Master Thesis

im Rahmen des
Universitätslehrganges „Geographical Information Science & Systems“
(UNIGIS MSc) am Interfakultären Fachbereich für GeoInformatik (Z_GIS)
der Paris Lodron-Universität Salzburg

zum Thema

Parametrisches (Geo)Design für Testplanungen

Vorgehensmodell zur Unterstützung von Planungen zur inneren Verdichtung

vorgelegt von

MSc Christoph Schaller
104551, UNIGIS MSc Jahrgang 2016

Zur Erlangung des Grades
„Master of Science (Geographical Information Science & Systems) – MSc(GIS)”

Dotzigen, 26.08.2018

Danksagung

Dank geht an Christine Seidler, William Fuhrer, Joachim Huber, Simon Gilgen, Jürg Bührer und Marcel Abegglen sowie Michael Walczak vom Kompetenzbereich Dencity des Departements Architektur, Holz und Bau der Berner Fachhochschule. Ohne die Inspiration durch die Arbeiten des Dencity sowie die Unterstützung durch Inputs und Feedbacks aus der Sicht von Architektur und Raumplanung in Rahmen fachlicher Diskussionen wäre diese Arbeit in der vorliegenden Form nicht möglich gewesen.
Spezieller Dank geht an Urs Sauter für seine Unterstützung als Vorgesetzter und Mentor während des Studiums. Sein Feedback zum vorliegenden Text hat mit seiner unabhängigen Perspektive zusätzliche Sichtweisen eröffnet und zur Verbesserung der Arbeit beigetragen.
Besten Dank auch an Christian Neuwirth für die Betreuung der Master Thesis und die immer prompten und hilfreichen Rückmeldungen bei Fragen und Problemen.

Christoph Schaller

Erklärung der eigenständigen Abfassung der Arbeit

Ich versichere hiermit, dass die vorliegende Master Thesis ohne fremde Hilfe und ohne Verwendung anderer als der angegebenen Quellen selbstständig verfasst und die den benutzten Quellen wörtlich und inhaltlich entnommenen Stellen als solche kenntlich gemacht habe. Die Arbeit wurde in gleicher oder ähnlicher Form keiner anderen Prüfungsbehörde vorgelegt.

Dotzigen, 26.08.2018 Christoph Schaller (eigenhändige Unterschrift)

Zusammenfassung

Diese Master Thesis beschäftigt sich mit der Entwicklung eines Vorgehensmodells zur Unterstützung von Testplanungen, um die Auswirkungen von Bauvorschriften auf die innere Verdichtung besser vermitteln zu können. Mit dem revidierten Schweizer Raumplanungsgesetz wurden die Einschränkung des Siedlungswachstums und eine Entwicklung nach innen als explizite Ziele festgeschrieben (RPG 2016). Dadurch ist in Planungsprozessen mehr Aufmerksamkeit nötig, da eine qualitätsvolle und nachhaltige Innenverdichtung gefordert ist, wobei GIS durch den inhärenten Raumbezug eine zentrale Rolle bei der Unterstützung und Bewertung spielt (Goodchild 2010; Angélil *et al.* 2016; Wissen Hayek *et al.* 2016).

Angelehnt an Arbeiten des Kompetenzbereichs Dencity für Urbane Entwicklung und Mobilität der Berner Fachhochschule wurde ein Vorgehensmodell entwickelt, das auf dem Geodesign Framework nach Steinitz (2012) basiert und systematisch GIS mit Parametrischem Design integriert. Zentrale Herausforderung dabei ist die Verknüpfung einer räumlichen Datenbasis mit parametrischen Modellen, sodass die Auswirkungen der Anpassung abstrakter Bauvorschriften auf die innere Verdichtung quantifiziert und vermittelt werden können. Insbesondere die hierzu entwickelten Visualisierungen und Modelle von Verdichtungsvarianten helfen, Anspruchsgruppen die räumlichen Auswirkungen solcher Änderungen in zielgruppengerechter Form zu vermitteln (Walz *et al.* 2008; Van Wezemael *et al.* 2014; Moura 2015; Wissen Hayek *et al.* 2016).

In Anlehnung an das Steinitzsche Framework ist das Vorgehensmodell, welches als integriertes Gesamtmodell zur Verdichtungsplanung mit aufeinander abgestimmtem Prozess, Datenmodell und technischer Unterstützung konzipiert ist, in zwei Phasen mit jeweils drei Schritten gegliedert. Die erste Phase zielt auf das Verständnis der Ist-Situation im betrachteten Projektgebiet. In einem ersten Schritt werden zu dessen Abbildung verschiedene Datenquellen (u.a. Amtliche Vermessung, harmonisierte Bauzonen des Bundesamtes für Raumplanung, GEOSTAT des Bundesamtes für Statistik, swissBUILDINGS3D) in eine PostGIS Datenbank importiert, die als zentraler Datenspeicher alle für das Modell benötigten Daten integriert. Auf dieser Basis werden in einem zweiten Schritt mit Hilfe von PostGIS, QGIS und R Kennzahlen abgeleitet, deren Auswahl (z.B. Struktur der Ist-Bebauung, bauliche Dichtekennziffern, Raumnutzerdichte, Haushaltsdichte) auf eine ganzheitliche Betrachtung der Dichte im Projektgebiet abzielt. In einem dritten Schritt werden die Resultate der Analyse zu Visualisierungen und Berichten aufbereitet, die als Grundlage zur Bewertung und Entscheidungsunterstützung dienen.

In der zweiten Phase geht es um die Auslotung und Vermittlung verschiedener Verdichtungsvarianten im Projektgebiet. Im vierten Schritt wird für die Definition von Szenarien und Generierung von Varianten ein in Rhino Grasshopper umgesetztes parametrisches Modell eingesetzt. Die zentrale Datenbank dient sowohl als Quelle für die räumlichen Inputdaten des Modells als auch der Speicherung der Szenarien und generierten Varianten. Das Modell integriert verschiedene bauliche Parameter wie z.B. Grenzabstände, Geschosszahl und Ausnützung, welche an die Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe angelehnt sind, wodurch die Übertragbarkeit begünstigt wird. In einem fünften Schritt werden die generierten Varianten anhand von Dichtekennziffern analysiert, um im Vergleich zur Ist-Situation deren räumliche Auswirkungen aufzeigen zu können. Die Resultate werden für die Entscheidungsfindung im sechsten Schritt wiederum als Visualisierungen und Berichte aufbereitet.

Im Rahmen der Verifikation des Vorgehensmodells zeigte dessen Anwendung auf zwei Testfälle in den Gemeinden Langenthal und Wohlen bei Bern die Übertragbarkeit des Modells auf, welche durch die Nutzung von Daten begünstigt wird, die schweizweit einheitlich oder ähnlicher Form verfügbar sind. Die Nutzung einer zentralen Datenbank zur Abbildung des Modells und von Python zur Schnittstellenprogrammierung stellt dabei die effiziente Integration zwischen GIS und parametrischen Designtools sicher. Resultate einer Sensitivitätsanalyse des parametrischen Modells sprechen dafür, dass dieses weitgehend den Erwartungen gemäss arbeitet und reale Zusammenhänge gut annähert. Zusammen mit den Rückmeldungen aus einem Workshop mit Architekten und Raumplanern des Dencity zur Bewertung des Vorgehensmodells kann der Schluss gezogen werden, dass das entwickelte Modell eine geeignete Grundlage für weitere Arbeiten darstellt. Gleichzeitig wurden allerdings einige Schwächen des Modells identifiziert, die vor einer Anwendung in der Praxis adressiert werden müssen. Dahingehend werden Empfehlungen für Anpassungen und Erweiterungen des Vorgehensmodells im Rahmen nachfolgender Arbeiten gegeben.

Inhaltsverzeichnis

Abbildungsverzeichnis

Tabellenverzeichnis

1 Einleitung

1.1 Motivation

Mit dem revidierten Raumplanungsgesetz kommt dem Thema innere Verdichtung in der Schweiz ein höherer Stellenwert zu. Aufgrund geographischer Eigenheiten kommen nur etwas über 30% der Schweizer Landesfläche als Siedlungsflächen in Betracht (BFS 2005). Effektiv konzentriert sich die überbaute Siedlungsfläche aktuell auf 7.5% der Fläche (ARE 2018a). Gleichzeitig hat bereits Mitte des 19. Jahrhunderts im Zusammenhang mit der fortschreitenden Industrialisierung eine Tendenz zur Ausbreitung des Siedlungsraumes eingesetzt, welche sich weiter fortsetzt und durch den steigenden Flächenverbrauch pro Kopf weiter verstärkt wird (BFS 2005; Schmid 2007). Dieses Siedlungswachstum trägt zusammen mit Prozessen der Verbuschung und Verwaldung im alpinen Raum zu einer fortschreitenden Zersiedlung und einem stetigen Verlust von Kulturland bei (ARE 2018a). Infolge dieser Entwicklungen hat schon vor einiger Zeit eine Diskussion über Urbanität, Verdichtung sowie deren Qualität in der Schweiz eingesetzt (vergleiche Beiträge in Lampugnani *et al.* (2007)).

Die Annahme der Teilrevision des Schweizer Raumplanungsgesetzes zeigt, dass inzwischen breite Teile der Bevölkerung eine Priorisierung der Entwicklung nach innen als Mindeststrategie zur Minimierung des Bodenverbrauchs akzeptieren (Grams 2015, s.7). Mit der Revision des Raumplanungsgesetzes wurden die Planungsziele angepasst, um die fortschreitende Zersiedelung zu bremsen und das Kulturland besser zu schützen (RPG, 2016). Das Siedlungswachstum soll eingedämmt und die Siedlungsentwicklung nach innen gelenkt werden (RPG, 2016). Mit dem Inkrafttreten der Revision müssen Bund, Kantone und Gemeinden ihre Planungsziele und -instrumente entsprechend anpassen. Dieser Anpassungsprozess und die Erreichung der gesetzten Ziele stellen in verschiedener Hinsicht eine Herausforderung dar.

Die Siedlungsentwicklung nach innen bedingt eine Verdichtung innerhalb der bestehenden Siedlungsfläche. Diese müssen durch Nachverdichtung, Schliessung von Baulücken und Umnutzung von (Industrie-)Brachen besser ausgenutzt werden (ARE ZH 2015; Grams 2015, s.16). Zusätzlich ist eine Abstimmung der Bauzonenreserven auf den tatsächlichen Bedarf nötig, wodurch sich Einschränkungen bei neuen Einzonungen ergeben (ARE ZH 2015; AGR 2016). Die notwendigen Abstimmungen für eine übergreifende Planung und die dazu benötigten planerischen Kapazitäten fordern mitunter eine regionale Zusammenarbeit von Behörden (Van Wezemael *et al.* 2014). Diese müssen im Rahmen der periodischen Überarbeitung der Planungsinstrumente die neuen Ziele des Raumplanungsgesetzes auf den lokalen Kontext übertragen und die zur Verfügung stehenden verbindlichen (Nutzungsplan, Baureglement) und optionalen (Leitlinien, Raumentwicklungskonzepte, Varianzverfahren, Sondernutzungsverfahren, Bodenpolitik) Policy Instrumente anpassen und kombinieren, um Anreize für die Entwicklung nach innen zu setzen und die Erreichung von Qualitätszielen positiv zu beeinflussen (Van Wezemael *et al.* 2014; ARE ZH 2015).

Der Qualität der Verdichtung kommt ein grosser Stellenwert zu. Die bauliche Dichte ist zwar ein Schlüsselelement für die Siedlungsentwicklung im Innern (Grams 2015, s.11ff.), für die Beurteilung der Qualität in verdichteten Siedlungsstrukturen ist aber eine ganzheitliche Betrachtung der Dichte unter Berücksichtigung weiterer Faktoren nötig (Schmid 2007; ROR 2012; Angélil *et al.* 2016). Bereits im revidierten Raumplanungsgesetz wird eine Siedlungsentwicklung nach innen unter Berücksichtigung einer angemessenen Wohnqualität gefordert (RPG 2016). Diese wird unter anderem durch die Durchmischung, Grünraum, soziale Aspekte, Verkehrserschliessung und Ortsbild beeinflusst (ARE 2007). Diese Faktoren spiegeln sich zudem teilweise in zusätzlichen Dichtebegriffen wie der Regelungsdichte (Anzahl gesetzlicher sowie sozialer Regeln), Einwohnerdichte (Anzahl Einwohner pro Hektare), Beschäftigtendichte (Anzahl Beschäftigte pro Hektare) und Interaktionsdichte/soziale Dichte (Anzahl der Interaktionen in einer bestimmten Bevölkerungsgruppe) wieder, welche die bauliche Dichte (bauliche Nutzung des Grundstücks) bei der Beurteilung der Dichtesituation ergänzen (Spiegel 2000; Häussermann 2007). Eine Beachtung dieser breit gefächerten Einflussfaktoren ist ausserdem nötig, weil sich daraus verschiedene Hindernisse für die Akzeptanz von Verdichtung ergeben, welche in die soziokulturelle, rechtliche, technische und wirtschaftliche Hindernisse unterteilen lassen (Bundesrat 2017).

Die Überwindung dieser Hindernisse und die Schaffung von Akzeptanz in Bevölkerung sowie Politik sind zentral für die erfolgreiche Umsetzung der Siedlungsentwicklung nach innen (ARE ZH 2015; Grams 2015, s.77f; Bundesrat 2017). Hierfür ist der Dialog mit der Bevölkerung und die Involvierung der breiten Öffentlichkeit entscheidend, was unmittelbaren Einfluss auf die Ausgestaltung der Prozesse zur Überarbeitung der Planungsinstrumente hat (ARE ZH 2015; AGR 2016). Statt einem rein linearen, projekthaften Vorgehen ist eine Überarbeitung mittels einer Kombination informeller und formeller Verfahren angebracht, welche die Öffentlichkeit in unterschiedlichem Grad involvieren (Van Wezemael *et al.* 2014; ARE ZH 2015; Grams 2015, s.123ff). Die informellen Verfahren unter Partizipation der breiten Öffentlichkeit haben durch Erarbeitung breit abgestützter Konzepte und Leitbilder eine vorbereitende und klärende Wirkung, welche zur Robustheit der nachfolgenden formellen Verfahren zur konkreten Festlegung der Planungsinstrumente beitragen (ARE ZH 2015; Grams 2015, pp.77f, 133, 156f; AGR 2016). Während diesen Verfahren kommt der Kommunikation und Vermittlung eine wichtige Rolle zu. Im Milizsystems der Schweizer Politik und in der breiten Öffentlichkeit fehlt meist das Fachwissen zum Verständnis von komplexen planerischen Fragestellungen und Fachbegriffen, was in Planungsprozessen eine auf die Zielgruppe abgestimmte Kommunikation nötig macht (Scholl *et al.* 2013; Van Wezemael *et al.* 2014). Neben einem übergreifenden Kommunikationskonzept zum gesamten Planungsprozess empfiehlt es sich, planerische Inhalte in einer für die Zielgruppe verständlichen Form aufzubereiten und zu vermitteln (Scholl *et al.* 2013; Van Wezemael *et al.* 2014; AGR 2016).

Modelle, Visualisierungen und Skizzen sind dabei wichtige Hilfsmittel für die Planung, deren Bedeutung mit zunehmender Komplexität der Planungsfragen zunimmt (Walz *et al.* 2008; Van Wezemael *et al.* 2014). In Bezug auf Verdichtung können insbesondere dreidimensionale Darstellungen helfen, den Bestand und Reserven anschaulich zu vermitteln sowie die Auswirkungen der Verdichtungsprozesse auf den Raum in intuitiv begreifbare Form zu bringen (Walz *et al.* 2008; Grams 2015, s.150). Gerade mit Hilfe parametrischer Modelle können durch die interaktive Anpassung von Parametern und die Visualisierung der resultierenden Änderungen die Auswirkungen einzelner Parameter auf den Raum besser kommuniziert und für beteiligte Anspruchsgruppen (Moura 2015; Wissen Hayek *et al.* 2016) wie auch für Architekten und Planer selbst (Speranza 2016) verständlich gemacht werden. Insgesamt können solche Modelle helfen, ein gemeinsames Verständnis zu entwickeln, Handlungsspielräume ins Bewusstsein von Grundeigentümern zu rufen und Zusammenarbeit anzuregen (Grams 2015, s.150). Bei komplexen Fragestellungen in Verdichtungs- und Umstrukturierungsgebieten bieten im Rahmen von informellen Verfahren zudem Methoden wie Wettbewerbe, Studienaufträge oder Testplanungen die Möglichkeit nach Lösungsansätzen zu suchen (ARE ZH 2015; AGR 2016). Insbesondere die Methode der Testplanung als kooperatives Planungsverfahren hat durch den Wettbewerb verschiedener Ideen das Potential, mögliche Lösungsansätze aufzuzeigen (Scholl *et al.* 2013; Grams 2015, s.64ff). Der Austausch zwischen Auftraggeber, Planern und oftmals der Öffentlichkeit im Rahmen der Überprüfung der Testentwürfe trägt zur Bildung eines gemeinsamen Verständnisses und der Beantwortung von Fragen bei, deckt manchmal aber auch neue Fragestellungen auf (Scholl *et al.* 2013; Grams 2015, s.64ff).

Da es sich bei Planungen im Zusammenhang mit der Siedlungsentwicklung und Verdichtung um gestaltende, raumwirksame Eingriffe handelt, kommt GIS bei deren Unterstützung und Bewertung eine zentrale Rolle zu (Dangermond 2010; Goodchild 2010; Wissen Hayek *et al.* 2016). Dabei bietet sich das Konzept des Geodesign an, um die gestalterischen Entscheide (Design) auf Basis wissenschaftlich fundierter Methoden und Informationen zu treffen, wobei Tools und Methoden der Geoinformatik zum Einsatz kommen (Goodchild 2010; Miller 2012). Der systemische Ansatz von Geodesign und die ganzheitliche Betrachtungsweise durch Integration verschiedener Disziplinen (Dangermond 2010; Steinitz 2012) deckt sich gut mit der Notwendigkeit einer umfassenderen Betrachtung von Dichte. Hinzu kommt eine starke Ausrichtung auf Partizipation der betroffenen Öffentlichkeit (Steinitz 2012). Diese Faktoren tragen dazu bei, dass insbesondere das von Steinitz (2012) vorgeschlagene Framework für Geodesign sich in der Praxis als geeignete Grundlage für die Gestaltung partizipativer Planungsprozesse herausgestellt hat (Moura 2015; Wissen Hayek *et al.* 2016).

1.2 Ziele

Unter anderem aufgrund der Flexibilität und einfachen Zugänglichkeit setzt der Kompetenzbereich Dencity für Urbane Entwicklung und Mobilität der Berner Fachhochschule Methoden des Parametrischen Designs im Rahmen von Testplanungen ein (Gilgen, 2016). Diesen Methoden werden grosses Potential bei der Generierung, Bewertung und Vermittlung von Planungsszenarien in Kooperation mit

Planern und Anspruchsgruppen zugeschrieben (Huber und Walczak 2016a). Dabei werden bereits diverse Geodaten als Grundlage verwendet (Huber und Walczak 2016b). Erfahrungen aus der Praxis zeigen aber, dass die Einbindung räumlicher Daten in den verwendeten Entwurfswerkzeugen oft unzureichend ist. Dahingehend wird im Rahmen der vorliegenden Master Thesis in Anlehnung an bisherige Arbeiten des Dencity ein Vorgehensmodell zur Unterstützung von Planungen entwickelt, das auf dem Geodesign Framework nach Steinitz (2012) basiert und systematisch GIS mit Parametrischem Design integriert. Ziel des Modells ist, durch die Quantifizierung und Visualisierung verschiedener Szenarien die durch Anpassung abstrakter Bauvorschriften verursachten Auswirkungen auf die innere Verdichtung einfacher bewerten und vermitteln zu können.

Die zentrale Forschungsfrage dieser Master Thesis lautet:

- Wie können basierend auf einer räumlichen Datenbasis sowie parametrischen Modellen die Auswirkungen auf die innere Verdichtung, welche durch Anpassung abstrakter Bauvorschriften verursacht werden, quantifiziert und vermittelt werden?

Die Beantwortung dieser Frage umfasst die Umsetzung der folgenden Hauptelemente:

- Vorgehensmodell: Angelehnt an das Geodesign Framework nach Steinitz (2012) wird ein Prozess zur Unterstützung von Testplanungen unter Einbezug eines parametrischen Modells erarbeitet.
- Parametrisches Modell: Es wird ein parametrisches Modell entwickelt, welches auf räumlichen Daten basiert, eine relevante Auswahl an baurechtlichen Parametern integriert und auf deren Basis die Generierung und Visualisierung von Verdichtungsvarianten ermöglicht.
- Analyse und Vermittlung der Dichte im Projektgebiet: Es werden Auswertungen entwickelt, welche im Rahmen des Vorgehensmodells die Analyse der Dichte im betrachteten Projektgebiet sowohl für die Ist-Situation als auch die generierten Varianten ermöglicht. Hierzu ist eine entsprechende Datenbasis zu definieren und aufzubauen.

Folgende Elemente sind explizit nicht Teil der Umsetzung:

- Partizipation der Öffentlichkeit: Im Rahmen der Arbeit ist keine Anwendung des Vorgehensmodells im Rahmen eines laufenden Planungsprojektes oder Partizipation öffentlicher Anspruchsgruppen geplant.

Im Zuge der Arbeit sind folgende Teilfragen zu bearbeiten:

- Anhand welcher Kennzahlen kann die Dichte in Siedlungsgebieten möglichst ganzheitlich beurteilt und vergleichbar gemacht werden?
- Welche baurechtlichen Parameter sind für die Generierung von Verdichtungsvarianten relevant und sind breit (d.h. schweizweit) anwendbar?
- Welche Daten sind für die Beurteilung der Dichte sowie als Input für das parametrische Modell nötig und wie können diese integriert werden?
- Über welche Mechanismen lassen sich GIS und Werkzeuge für Parametrisches Design effizient integrieren?

1.3 Methoden

In einem ersten Schritt werden die bestehenden Grundlagen für die Arbeit durch Literaturrecherche weiter vertieft. Dies betrifft insbesondere geeignete Indikatoren zur Beurteilung von Dichte aber auch die Analyse von Grundlagen für die Ausgestaltung und Anwendung des parametrischen Modells. Hierfür sind vor allem die gesetzlichen Grundlagen im Bau- und Planungsbereich sowie einschlägige Literatur zu analysieren. Hinzu kommen technische Abklärungen.

Als zweiter Schritt wird das Vorgehensmodell für die Arbeit angelehnt an Geodesign Prozesse ausgearbeitet. Als Basis dient das von Steinitz (2012) beschriebene Geodesign Framework. Die Ergebnisse aus der Grundlagenrecherche dienen dabei als Input. Die Arbeiten in diesem Schritt korrespondieren mit den ersten beiden Durchläufen des Geodesign Frameworks und umfassen mehrere Iterationen mit inkrementellen Verbesserungen des definierten Prozesses.

In einem dritten Schritt wird das ausgearbeitete Vorgehensmodell anhand eines ersten Fallbeispiels in der Gemeinde Langenthal umgesetzt. Parallel wird die technische Implementierung bearbeitet. Dieser Schritt korrespondiert weitgehend mit dem dritten Durchlauf des Geodesign Frameworks zur Durchführung der eigentlichen Studie. Wiederum ist ein Vorgehen mit mehreren Iterationen zur Verfeinerung vorgesehen, wobei es Überlappungen mit Schritt 2 gibt. Für die technische Umsetzung kommen Open Source GIS Werkzeuge (u.a. QGIS und PostgreSQL mit PostGIS Extension) und das kommerzielle Tool Rhino 3D mit der Grasshopper Erweiterung für Parametrisches Design zum Einsatz. Teil der Umsetzung ist ausserdem der Aufbau der Datenbasis für das parametrische Modell sowie die Analyse der Dichte.

In einem vierten und letzten Schritt wird das erarbeitete Vorgehensmodell verifiziert. Nach Fertigstellung einer ersten Version anhand des ersten Fallbeispiels, wird das Modell auf ein zweites Fallbeispiel in der Gemeinde Wohlen bei Bern angewendet. Ziel ist, die Übertragbarkeit des Vorgehensmodells und dessen Implementation aufzuzeigen. Zudem liefert die Ausarbeitung von Szenarien für die beiden Fallbeispiele Grundlagen für die Beurteilung der Ergebnisse aus fachlicher Sicht durch Architekten und Raumplaner des Kompetenzbereichs Dencity. Als dritter Teil der Verifizierung wird eine Sensitivitätsanalyse des zentralen parametrischen Modells in Bezug auf eine Auswahl relevanter Parameter durchgeführt.

1.4 Struktur der Arbeit

Die Thesis ist in sechs Kapitel gegliedert. Nach der Einführung in das Thema, der Forschungsfrage und das Vorgehen, werden in Kapitel 2 Grundlagen zur Arbeit vertieft. Neben Übersichten zum Thema Parametrisches Design und Geodesign wird eine kurze Einführung zum Thema Raumplanung und Siedlungsentwicklung in der Schweiz gegeben. In diesem Rahmen werden zudem Verknüpfungen zwischen dem Thema Raumplanung und Geodesign sowie Parametrischem Design aufgezeigt.

Das Kapitel 3 beschreibt das erarbeitete Vorgehensmodell aus methodischer Sicht. Nach einem Überblick zu dessen Aufbau werden die einzelnen Schritte und deren Grundlagen detailliert beschrieben, was die Definition des zentralen parametrischen Modells und dessen Zusammenhänge einschliesst.

Kapitel 4 behandelt die konkrete Umsetzung der Arbeit mit Fokus auf die technische Implementierung. Einerseits werden die Datengrundlagen, deren Integration und Analyse behandelt. Andererseits wird eine Übersicht der technischen Architektur gegeben und die Umsetzung des parametrischen Modells im Detail beschrieben.

Kapitel 5 beschreibt die Anwendung, Analyse und Ergebnisse der Arbeit. Neben einer Übersicht zu den beiden Testfällen in Langenthal und Wohlen werden zunächst allgemeine Resultate der Arbeit zusammengefasst. Ergänzt wird dies durch eine Übersicht der aufbereiteten Szenarien und die Ergebnisse der Beurteilung aus Expertensicht. Abschliessend folgt die Auswertung der Sensitivitätsanalyse.

Das Kapitel 6 fasst schliesslich die Ergebnisse der Arbeit zusammen und formuliert Empfehlungen für künftige Arbeiten.

2 Grundlagen

2.1 Parametrisches Design

2.1.1 Definition

Jabi (2013) definiert Parametrisches Design wie folgt:

> *Parametric Design: A process based on algorithmic thinking that enables the expression of parameters and rules that, together, define, encode and clarify the relationship between design intent and design response.*
> (Jabi 2013)

Dieser Definition zufolge handelt es sich bei Parametrischem Design also um einen Prozess, der auf algorithmischem Denken basiert und den Ausdruck von Parameter sowie Regeln ermöglicht, die zusammen die Beziehung zwischen Absicht und Auswirkung eines Entwurfs (d.h. Designs) definieren, kodieren und verdeutlichen (Jabi 2013). Frazer (2016) bezeichnet diese Definition als dienlich aber weit gefasst. Laut Monedero (2000) ist der Begriff Parametrisches Design ausserdem in dem Sinn eingeschränkt, dass die Verwendung von Parametern zur Definition einer Form impliziert wird, oftmals aber vielmehr die Definition und Ausnutzung von Beziehungen zwischen Formen eine Rolle spielt (z.B. dass eine Linie parallel zu einer anderen verlaufen muss). Zudem gibt es Überschneidungen zu anderen Begriffen wie Variational Design oder Constraint-Based Design (Monedero 2000).

Während Monedero (2000) Parametrisches Design als umfassenden Prozess anschneidet, steht in seinen Betrachtungen die Modellierung mittels parametrischen Werkzeugen im Zentrum. Darüber hinaus besteht keine klare Abgrenzung zwischen Parametrischem Design und den Scripting- und Programmier-Funktionen, die viele CAD und Computergrafik Programme bieten und so die parametrisierte Erzeugung von Geometrien erlauben (Monedero 2000). Frazer (2016) weist ebenfalls darauf hin, dass der Prozess des Parametrischen Designs im Kern von einem parametrischen Modell abhängt. In diesem Zusammenhang wird auch der Begriff Parametrische Modellierung (Parametric Modelling) verwendet (Janssen und Stouffs 2015). Hierzu gibt Frazer (2016) eine Definition von Janssen aus deren persönlichen Kommunikation wieder, die ein parametrisches Modell wie folgt definiert:

> *an algorithm that generates models consisting of geometry and attributes (e.g. material definitions). This algorithm uses functions and variables, including both dependent and independent variables. Some of the independent variables can be given a more prominent status, as the interface to the parametric model – these are referred to as the parameters of the model*
> (Patrick Janssen in Frazer 2016)

Das im Zusammenhang mit dem Parametrischen Design Prozess angesprochene algorithmische Denken wird also durch das parametrische Modell als konkreter Algorithmus implementiert. Dabei stellen die unabhängigen Input Variablen die eigentlichen Parameter des Modells dar. Die Festlegung dieser Parameter erfolgt durch den Benutzer des Modells (z.B. durch explizite Nachfrage nach Eingaben) (Monedero 2000).

2.1.2 Ansätze Parametrischer Modellierung

Monedero (2000) beschreibt fünf Ansätze der Parametrischen Modellierung, welche sich vor allem darin unterscheiden, auf welche Art Parameter und Einschränkungen definiert und bei der Generierung von Geometrien berücksichtigt werden:

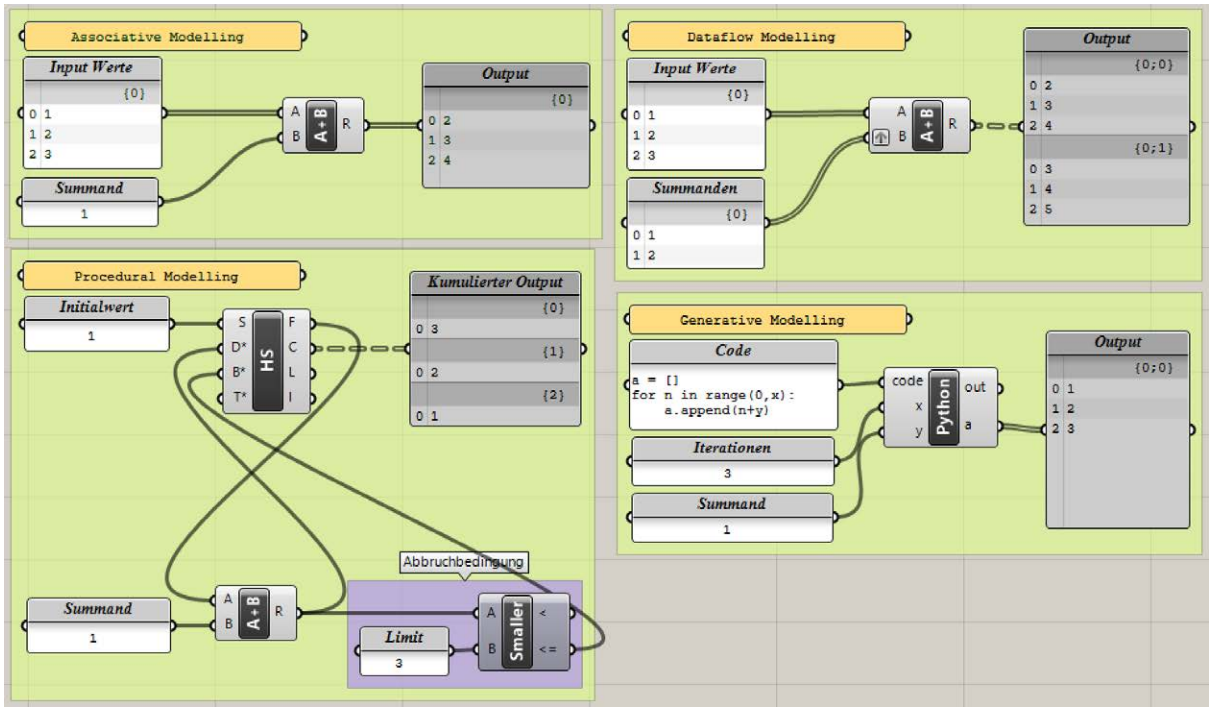
- *Variants programming by macros or procedural modeling*
  Gescriptete Befehle zur Erzeugung von Geometrien, welche in bestimmten Punkten parametrisiert werden können.
- *History-based constraint modelers*
  Generierung von Geometrien mittels eines Verlaufs von aufeinander aufbauenden geometrischen Primitiven und Operationen, welche parametrisiert und mit Einschränkungen versehen werden können.
- *Variational geometry and variational design*
  Erzeugung von Geometrien auf Basis einer Beschreibung mittels Gleichungen und Nebenbedingungen, welche durch einen Solver gelöst werden.

- *Rule-based variants: geometric reasoning by expert systems*
  Analog zu Variational geometry werden Geometrien durch Gleichungen und Bedingungen beschrieben. Deren Anzahl und Komplexität wird aber durch die Nutzung logischer Programmierung und die Einbindung von Inferenz-Engines stark reduziert.
- *Parametric feature-based design*
  Erzeugung von Geometrien basierend auf der Kombination von Primitiven und Features (Formen mit einer bestimmten Funktion in einem bestimmten Kontext) wie z.B. Löcher, Überhänge, Anschrägungen oder Vorsprüngen, welche jeweils in bestimmten Punkten parametrisiert werden können (z.B. Radius, Breite, Länge).

Die Kategorisierung von Monedero ist zwar als Review und Überblick nützlich, aber nur bedingt systematisch. Janssen und Stouffs (2015) hingegen schlagen ein generalisiertes Modell für Parametrische Modellierung vor und leiten daraus eine Taxonomie für parametrische Modellierungsansätze ab. Das generalisierte Modell basiert auf Gerichteten Azyklischen Graphen (DAG: Directed Acyclic Graph) bestehend aus Operations- und Daten-Knoten sowie verbindenden Kanten zur Repräsentation der Datenflüsse zwischen den Knoten (Janssen und Stouffs 2015). Schleifen sind aufgrund der azyklischen Natur von DAGs in diesem Modell nicht möglich, wohl aber Operationen, welche über Listen von Daten iterieren (Janssen und Stouffs 2015). Die Art der Unterstützung von Iterationen (entweder implizit durch Wiederholung einer Operation für alle Elemente im Input oder explizit durch Rekursion oder Knoten mit spezieller Schleifen-Semantik) ist zentrales Unterscheidungskriterium in der Taxonomie (Janssen und Stouffs 2015). Abbildung 1 illustriert am Beispiel von Rhino Grasshopper verschiedene Arten der Iteration aus der Taxonomie (man beachte, dass Art und Länge des Outputs von der Semantik der Operation sowie Länge und Art der Inputs abhängen). Die Taxonomie von Janssen und Stouffs (2015) unterscheidet folgende Kategorien:

- *Object Modelling*
  Methoden, welche keine Iteration unterstützen und nur einen implizit definierten Objekt Graphen verwenden.
- *Associative Modelling*
  Methoden, welche die implizite Iteration von Operationen mit einfachen Inputs unterstützen (d.h. der Dateninput kann eine Liste sein, während zusätzliche Parameter nur einen Wert aufweisen können).
- *Dataflow Modelling*
  Methoden, welche die implizite Iteration von Operationen mit mehrfachen Inputs unterstützen (d.h. der Dateninput und die zusätzlichen Parameter können Listen sein).
- *Procedural Modelling*
  Methoden, welche mittels spezialisierter Knoten die explizite Iteration von Operationen mit mehrfachen Inputs unterstützen (vergleiche Beispiel mit Rekursion in Abbildung 1).
- *Generative Modelling*
  In dieser Kategorie werden Methoden zur parametrischen Generierung auf Basis von imperativer Programmierung eingeordnet. Diese erlauben Schleifen, weshalb sie nicht mittels eines DAG des generischen Modells abgebildet werden können.

In Abbildung 1 nicht vertreten ist der Object Modelling Ansatz, der oft bei einfacheren Feature- oder Szenen-basierten Ansätzen zu finden ist. Solche Ansätze werden vor allem in 3D Applikationen und Animationswerkzeugen wie Autodesk Maya, Autodesk 3DS Max oder Blender verwendet und lassen sich mit dem generalisierten Modell von Janssen und Stouffs (2015) abbilden, wobei im eigentlichen Programm aber oft nur implizit ein Graph verwendet wird. Komplexere Feature- oder Szenen-basierten Ansätze können mitunter schon zur Kategorie Associative Modelling gehören (Janssen und Stouffs 2015). Verschiedene Werkzeuge zur Parametrischen Modellierung verwenden Graphen zur Definition von Modellen. Dazu gehören unter anderem Rhino Grasshopper, Sidefix Houdini, Autodesk Dynamo oder Blender Sverchok. Diese Systeme lassen sich gut mit dem generalisierten Modell abbilden und sind abhängig von der Unterstützung von Iterationen meist der Dataflow oder Procedural Modelling Kategorie zuzuordnen (Janssen und Stouffs 2015).



Quelle: eigene Darstellung

Abbildung 1: Illustration Modellierungsansätze nach Janssen und Stouffs (2015) in Grasshopper

Das unterliegende generalisierte Modell und die klare Definition der Taxonomie von Janssen und Stouffs (2015) erleichtert deren Anwendung und kann beim Verständnis der Funktion solcher Methoden helfen. Wie das Beispiel in Abbildung 1 zeigt, kann es bei der Einordnung von Programmen und deren Modellierungsansätzen mittels der Taxonomie aber zu Überschneidungen kommen. So ist zum Beispiel Rhino Grasshopper in seiner Basisversion in die Kategorie Dataflow Modelling einzuordnen, da nur implizite Iterationen mit mehrfachen Operationen möglich sind (darin ebenfalls enthalten sind implizite Iteration mit einfachen Inputs, wie sie bei Associative Modelling unterstützt werden). Spezielle Erweiterungen wie Hoopsnake ermöglichen überdies die Unterstützung expliziter Iterationen in Grasshopper, wodurch sich die Einordnung in Richtung Procedural Modelling verschiebt. Durch Nutzung von Scripts in speziellen C# oder Python Knoten können Mittel der imperativen Programmierung genutzt werden, wodurch sich die Möglichkeiten stark erweitern und es zu einer Überschneidung mit der von Janssen und Stouffs (2015) als Generative Modelling bezeichneten Kategorie kommt.

2.1.3 Anwendung in Architektur und Planung

Parametrisches Design und Parametrische Modellierung finden in Architektur und Planung verschiedentlich Anwendung. In Bezug auf Architektur steht meist die gestalterische Seite des Designs im Vordergrund. Zu den bekannteren Beispielen hierfür gehören die Arbeiten von Zaha Hadid Architects (2018). Unter dem Titel Parametricism hat Schumacher (2008) sogar einen eigenständigen Architekturstil proklamiert, der auf parametrischen Methoden fusst. Neben der Gestaltung einzelner Gebäude ist in diesem Zusammenhang auch von übergreifenden Planungen in der Form von Parametric Urbanism die Rede (Schumacher 2009). Insgesamt steht hier aber der ästhetische Anspruch im Vordergrund, dem andere Überlegungen untergeordnet werden. Dies hat verschiedentlich für Kritik und kontroverse Diskussionen gesorgt. So weist Frazer (2016) darauf hin, dass sich nicht zwingend ein spezifischer Architekturstil ergibt, wenn die Anwendung von parametrischen Methoden mit dem primären Ziel der Generierung variierter Geometrien erfolgt. Vielmehr adressiere Architektur komplexere Probleme, welche nicht alleine über die Generierung von Formen mittels Algorithmen gelöst werden können (Frazer 2016). Bei einer weiter gefassten Anwendung mit dem Ziel der besseren Lösung von Problemen unter Einbezug von Einflüssen aus Gesellschaft und Umwelt macht Frazer (2016) aber durchaus ein Potential für parametrische Methoden in der Architektur aus. In eine ähnliche Kerbe schlägt Mehaffy (2011), der ebenfalls grosses Potential in parametrischen und generativen Methoden für Planungen und architektonische Entwürfe sieht. Dies aber vor allem im Fall der Nutzung solcher

Methoden zur besseren Lösung komplexer Probleme und nicht zur reinen Erreichung ästhetischer Effekte, welche primär der Befriedigung künstlerischer Ziele dienen (Mehaffy 2011).

Bei architektonischen Planungen bestehen Überschneidungen zur Raumplanung auf übergeordneter Ebene, da bei der Planung im Detail das Umfeld zu berücksichtigen ist (Mehaffy 2011). Bei der Raumplanung werden aber die übergeordneten Ziele und Grundlagen in einem grösseren Zusammenhang betrachtet, wodurch die architektonische Detailplanung und Gestaltung weitgehend ausgeklammert wird. Die Betrachtung von Anwendungsbeispielen parametrischer Methoden für Raum- und Siedlungsplanung zeigen denn auch, dass diese praktisch ausschliesslich wie von Frazer und Mehaffy gefordert für die Lösung von grundlegenden Problemen und Zusammenhängen im Raum genutzt werden. Dies schliesst die Generierung von Entwürfen ganzer Städte oder Stadtteile mittels parametrischer Modelle mit ein (Beirão *et al.* 2012; Grêt-Regamey *et al.* 2013; Moura 2015; Taleb und Musleh 2015; Janssen *et al.* 2016). Dabei können sowohl rein parametrische Modelle (Beirão *et al.* 2012; Taleb und Musleh 2015; Janssen *et al.* 2016), welche die eigentliche architektonische Detailgestaltung stark abstrahieren (z.B. durch Extrusion einfacher Volumen), als auch stärker formalisierte Modelle auf Basis von Shape Grammars (Wissen Hayek *et al.* 2013; Neuenschwander *et al.* 2014; Moura 2015) verwendet werden. Bei letzteren wird die architektonische Gestaltung teilweise ebenfalls nur abstrahiert dargestellt (Moura 2015). In anderen Fällen werden durch Anwendung entsprechender Regeln mehr gestalterische Details eingebracht, um «realistischere» Darstellungen zu erzeugen (Wissen Hayek *et al.* 2013; Neuenschwander *et al.* 2014). Gemein ist den Beispielen der Bezug auf eine räumliche Fragestellung, deren Lösung mit Hilfe parametrischer Modelle bearbeitet wird. Inhalt und Ziele der Fragestellung unterscheiden sich dabei teilweise stark. Darunter sind Beispiele, die parametrisch aufbereitete Zonenpläne umsetzen, um die Auswirkung von Planungen besser vermitteln zu können (Halatsch 2013; Wissen Hayek *et al.* 2014; Moura 2015). Andere Beispiele setzen sich mit spezifischen Umweltfragen wie dem Grünraum in einem Siedlungsgebiet (Neuenschwander *et al.* 2014) oder der Optimierung der mikroklimatischen Bedingungen in heissen Klimaten (Taleb und Musleh 2015) auseinander. Wieder andere Beispiele behandeln die grundsätzliche Anwendung parametrischer Methoden für Entwürfe von Städten (Beirão *et al.* 2011; Koenig *et al.* 2017) und darüber hinaus deren Analyse (de Monchaux 2010; Beirão *et al.* 2012).

2.2 Geodesign

2.2.1 Definition

In der Literatur sind verschiedene Definitionen des Begriffs Geodesign zu finden. Goodchild (2010) verweist zunächst auf ein Zitat von Carl Steinitz, für eine simple Definition: «Geodesign is geography by design». Geodesign versucht durch aktive Intervention die Geographie (d.h. die Menge aus Form und Prozessen nahe der Erdoberfläche) zu verbessern, was in Kontrast zum traditionellen Vorgehen in der Wissenschaft steht, das primär auf das Verständnis der Geographie ausgerichtet ist (Goodchild 2010). Für eine praktischer ausgerichtete Definition bezieht sich Goodchild (2010) auf Flaxman (2010) und definiert Geodesign als Planung auf Basis wissenschaftlicher Erkenntnisse über die Funktion der Welt, welche in GIS-basierten Simulationen ausgedrückt werden. Dangermond (2010) beschreibt Geodesign etwas allgemeiner als einen interdisziplinären, synergetischen Ansatz, der zur Lösung kritischer Probleme sowie zur Optimierung des Standorts, der Ausrichtung und der Merkmale von Projekten auf lokaler und globaler Ebene dient. Dabei weist er auf Parallelen zu Vorgehensweisen in GIS und Ansätzen aus der Ökologie hin (Dangermond 2010). Laut Miller (2012) können im weitesten Sinne sogar alle entwurfsbezogenen Aktivitäten, die vom räumlichen Kontext abhängen oder diesen auf irgend eine Weise ändern, als Geodesign angesehen werden.

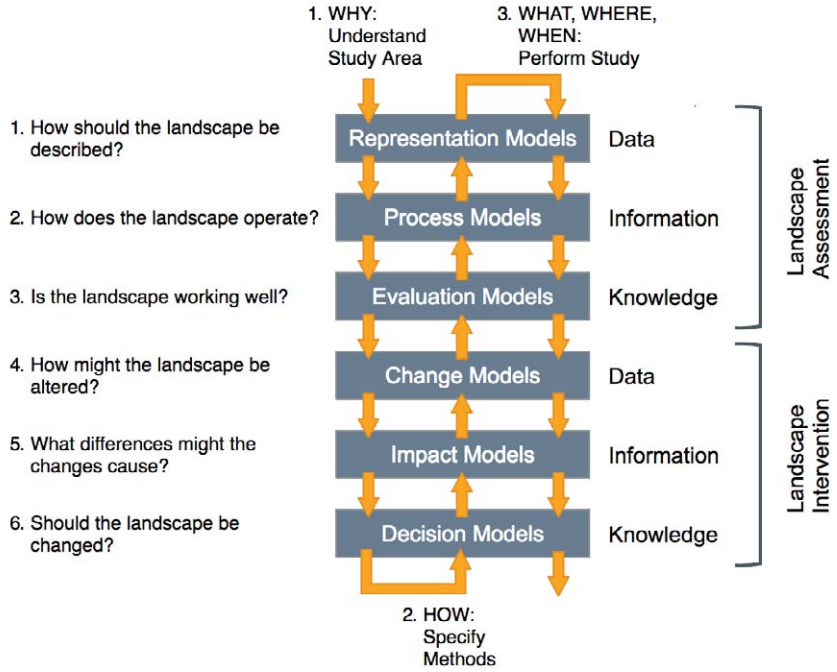
Aus den verschiedenen Definitionen und Beschreibungen von Geodesign können zwei zentrale Punkte als Gemeinsamkeit identifiziert werden. Erstens ist die Verknüpfung von Design und Wissenschaft durch Integration wissenschaftlich gestützter Daten, Analysen und Methoden in den Entwurfsprozess zu nennen (Dangermond 2010; Goodchild 2010; Steinitz 2012; Foster 2016). Die Qualität der Resultate eines Entwurfsprozesses und deren Nachhaltigkeit wird insgesamt verbessert, da von Beginn weg kritische Faktoren aus Umwelt, Gesellschaft und Wirtschaft mitberücksichtigt und mögliche Auswirkungen des Vorhabens analysiert werden, statt erst im Nachhinein solche Analysen anzustellen und allenfalls erneute Korrekturen vornehmen zu müssen (Dangermond 2010). Eine solche ganzheitliche Betrachtungsweise hebt Steinitz (2016) zudem als Unterschied von Geodesign zu traditionellen Planungsmethoden hervor und verweist darauf, dass hier «Systemdenken» in die Praxis umgesetzt und Planungen

auf lokaler Ebene mit Planungen auf systemischer Ebene kombiniert werden. Ein Potential zur Verbesserung der Resultate ergibt sich ausserdem aus der gegenseitigen Befruchtung von wissenschaftlichen und kreativen Methoden sowie Herangehensweisen (Foster 2016). Als zweiter Punkt ist die tragende Rolle von GIS im Geodesign Prozess zu nennen (Dangermond 2010; Goodchild 2010; Steinitz 2012; Batty 2013). Geographische Informationssysteme sind für die Prozessunterstützung zentral, wenn es um die Verwaltung, Verarbeitung, Analyse und Visualisierung der benötigten räumlichen Daten geht (Dangermond 2010). Aus diesen Gründen hat die steigende Verbreitung von GIS sowie die Maturität der zugehörigen Werkzeuge und Methoden einen grossen Anteil daran, dass Geodesign sich in der letzten Zeit breiter etablieren konnte (Dangermond 2010; Batty 2013). Mit der Verfügbarkeit Web- und Cloud-basierter GIS Werkzeuge mit Unterstützung für die Zusammenarbeit vieler Nutzer ist darüber hinaus die Voraussetzung für den erfolgreichen Einsatz von Geodesign in partizipativen Prozessen gegeben (Batty 2013).

Trotz steigender Verbreitung ist Geodesign eine relativ junge Disziplin, die aber auf eine längere Entwicklungsgeschichte zurückblicken kann (Tulloch 2012). GIS und Geodesign haben gemeinsame Wurzeln im Bereich Planung und Landschaftsarchitektur (Batty 2013). Goodchild (2010) gibt einen kurzen Abriss zur Entwicklung und nennt insbesondere Ian McHarg, einen Pionier der modernen Landschaftsarchitektur, als zentralen Einfluss. Die von McHarg in seinem Buch *Designing with Nature* (McHarg 1969) beschriebenen Methoden für Planungen in der Landschaftsarchitektur unter Berücksichtigung räumlicher Gegebenheiten nehmen nicht nur zentrale Aspekte von Geodesign sondern auch moderner GIS Anwendungen vorweg (Dangermond 2010; Goodchild 2010). Ein weiterer bedeutender Faktor war die Etablierung der Geographic Information Sciences (GIScience) (Goodchild 1992), die eine wichtige Rolle bei der Umsetzung von Geodesign spielen (Goodchild 2010). Weitere Beiträge zur Etablierung von Geodesign in seiner heutigen Form sind auf die Ideen und Arbeiten von Carl Steinitz zurückzuführen (Dangermond 2010), welche in der Entwicklung und Publikation eines umfassenden Frameworks für Geodesign mündeten (Steinitz 2012).

2.2.2 Geodesign Framework nach Steinitz

Carl Steinitz beschreibt im Buch *A Framework for Geodesign: Changing Geography by Design* umfassend ein Rahmenwerk für die Ausarbeitung und Durchführung von Geodesign Prozessen (Steinitz 2012). Abbildung 2 zeigt eine Übersicht des Kerns dieses Frameworks.



Quelle: adaptiert von Steinitz (2012)

Abbildung 2: Prozessstruktur des Geodesign Frameworks nach Steinitz (2012)

Der Kern besteht aus sechs von Steinitz als Modelle bezeichneten Komponenten, welche jeweils eine bestimmte Frage zum Studiengebiet beantworten. Die Modelle können in zwei Gruppen unterteilt werden, wobei es bei der ersten Gruppe um das Verständnis der Ist-Situation und bei der zweiten Gruppe um die Soll-Situation (d.h. die Auswirkung der geplanten Eingriffe) geht. In der ersten Gruppe bilden Representation Models die aktuelle Situation im Studiengebiet ab, die Process Models analysieren die Prozesse und Zusammenhänge darin und die Evaluation Models helfen bei der Beurteilung der aktuellen Situation. In der zweiten Gruppe beschreiben die Change Models die Möglichkeiten der Veränderungen und Eingriffe im Raum, die Impact Models beurteilen die Auswirkungen dieser Eingriffe und die Decision Models helfen bei der Beurteilung und der Entscheidungsfindung, welche Eingriffe ausgeführt werden sollen. Diese Gruppen bilden jeweils einen Dreischritt, bei dem im ersten Schritt über Daten die Situation beschrieben, diese im zweiten Schritt zu Informationen verdichtet und im dritten Schritt daraus Wissen abgeleitet wird. Die sechs Modelle bauen aufeinander auf und Steinitz sieht vor, dass diese in drei Phasen in jeweils unterschiedlicher Reihenfolge durchlaufen werden. Der erste Durchlauf erfolgt in aufsteigender Reihenfolge und soll das Verständnis des Studiengebiets ermöglichen sowie die Antwort auf die Frage geben, WESHALB die Studie durchgeführt wird. Der zweite Durchlauf erfolgt in umgekehrter Richtung, wobei auf Basis der Antworten aus dem ersten Durchlauf die Methoden für die Studie festgelegt werden. Es wird also festgelegt, WIE die Studie durchgeführt werden soll. Im letzten Durchlauf wird die eigentliche Studie beziehungsweise der Design Prozess ausgeführt, wobei die Schritte wieder in aufsteigender Reihenfolge durchlaufen werden. Dabei sollen die Fragen WAS, WO und WANN für die Eingriffe beantwortet werden.

Das Framework von Steinitz zeigt in seiner Strukturierung und den gestellten Fragen grosse Parallelen zu anderen Vorgehensmodellen für Design (Foster 2016). Trotzdem ist dieses nicht zwingend als fixes Vorgehen, sondern als offenes Rahmenwerk zur Definition individueller Geodesign Prozesse für spezifische Situationen zu sehen. In diesem Sinne kann das Framework von Steinitz auch als Anleitung zum «Entwurf des Designs» («designing the design») mit starker Ausrichtung auf die jeweiligen Entscheidungsträger gesehen werden, was eine Neuerung und Verbesserung im Vergleich zu anderen Design Ansätzen darstellt (Foster 2016). Dies deckt sich mit den zahlreichen Hinweisen von Steinitz (2012), dass der Entwurf von Änderungen nicht durch einzelne Personen bestritten wird, sondern entsprechend dem ganzheitlichen Ansatz von Geodesign zahlreiche Akteure aus den Bereichen Design und Wissenschaft beteiligt sind. Zudem wird ein starkes Gewicht auf den Einbezug von Personen gelegt, die vor Ort von den Veränderungen betroffen sind («people of the place») (Steinitz 2012). Darüber hinaus wird an mehreren Stellen darauf hingewiesen, dass die eigentliche Entscheidung über die durchzuführenden Eingriffe schlussendlich nicht durch die Designer, sondern durch die verantwortlichen Entscheidungsträger vor Ort erfolgt (Steinitz 2012).

Neben einer Individualisierung der einzelnen Schritte des Geodesign Prozesses ist auch der Ablauf zu dessen Ausarbeitung als weniger starr anzusehen, als es die klare Strukturierung des Frameworks vielleicht denken lässt. Die Ausarbeitung beschränkt sich nicht auf einen einfachen Durchlauf der drei Phasen mit jeweils sechs Schritten. Vielmehr hat bereits der Entwurf des Vorgehens für die eigentliche Studie einen iterativen Charakter, der sich mitunter in mehreren Schleifen mit Verfeinerungen und Anpassungen äussern kann (Steinitz 2012). Dies hat das Framework mit anderen Design Vorgehen gemein (Foster 2016). Zudem sind die einzelnen Schritte und Phasen nicht zwingend sequentiell und streng getrennt, sondern sind vielmehr als sich überlappende Bereiche zu betrachten (Foster 2016). Gleichfalls betont Steinitz (2016), dass bei der Durchführung der eigentlichen Studie mitunter mehrere Durchläufe in schneller Folge mit jeweiliger Analyse und Bewertung der Auswirkungen der Entwürfe und anschliessenden Anpassungen nötig sind.

2.2.3 Anwendungsbeispiele

In der Literatur finden sich verschiedene Beispiele der Anwendung von Geodesign. Eine zunehmende Zahl an publizierten Projekten deutet darauf hin, dass Geodesign nicht nur ein kurzlebiger akademischer Trend ist, sondern dass sich die Methode etabliert (Tulloch 2017). Tulloch (2017) untersucht 28 Publikationen zu Geodesign Projekten und kategorisiert diese in Hinblick darauf, ob 1) ein Teil der Kontrolle über den Entwurf dem Computer überlassen wird, 2) ob öffentliche Partizipation Teil des Prozesses war und 3) ob komplexe inhaltliche Modelle (Substantive Models) zum Einsatz kamen. Abhängig davon, wie und in welcher Form diese Kriterien erfüllt werden, können Klassen ähnlicher Projekte identifiziert werden, welche bei Vergleichen und der Suche nach Beispielen hilfreich sein können

(Tulloch 2017). Computer spielen bei vielen Projekten eine Rolle im Entwurfsprozess, wenn auch mit unterschiedlichem Grad an Kontrolle über den Prozess oder nur in unterstützender Form (Tulloch 2017). Bei der Partizipation reicht das Spektrum von fehlender über einfache bis hin zu sehr intensiver Partizipation (Tulloch 2017). Bei den inhaltlichen Modellen sind verschiedene Grade an Komplexität unterscheidbar, wobei einige Projekte sogar komplett auf solche Modelle verzichten (Tulloch 2017).

Campagna und Di Cesare (2016) analysieren Parallelen zwischen dem Geodesign Framework nach Steinitz (2012) und verschiedenen Planungsprozessen in Italien in Hinblick auf Regularien auf europäischer, nationaler und regionaler Ebene. Während bei aktuellen Planungsprozessen unterschiedliche Grade an Überlappung mit dem Geodesign Framework auszumachen sind, wird dessen Anwendung Potential zugeschrieben, die Lücke zwischen den gesetzlichen Regularien und deren Umsetzung zu schliessen (Campagna und Di Cesare 2016). Dies insbesondere in Bezug auf Regularien, die auf eine ganzheitliche Betrachtung und eine nachhaltige Entwicklung mit Einbezug der Regionen bei der Planung abzielen (Campagna und Di Cesare 2016).

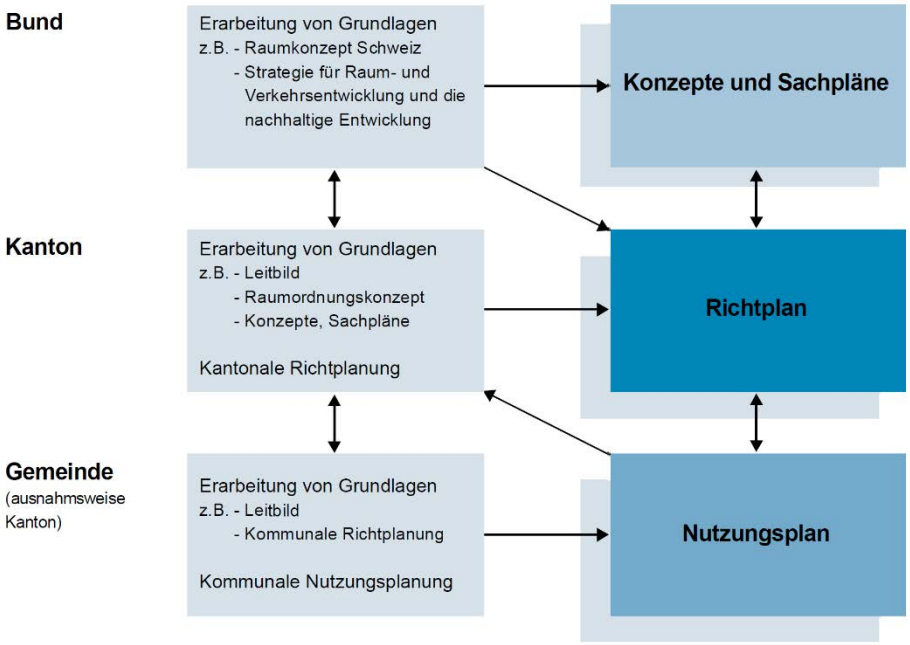
Moura (2015) hat das Geodesign Framework von Steinitz (2012) für eine Studie zur Nutzungsplanung in Pampulha (Brasilien) angewendet, wobei die einzelnen Schritte weitgehend direkt adaptiert wurden. Die Wirkung der aktuellen gesetzlichen Bestimmungen und die Folgen möglicher Anpassungen dieser Bestimmungen auf die urbane Landschaft im Studiengebiet sollten aufgezeigt und analysiert werden, wobei ein parametrisches Modell auf Basis der Esri CityEngine für die Umsetzung und Visualisierung verwendet wurde (Moura 2015). Geodesign wurde dabei als geeignetes Rahmenwerk für die Gestaltung von Prozessen zur Nutzungsplanung eingestuft (Moura 2015). Dabei werden vor allen dessen Offenheit und die Möglichkeit zur partizipativen Beteiligung der betroffenen Öffentlichkeit sowie der Berücksichtigung derer Meinung im Rahmen der Planung hervorgehoben (Moura 2015).

Als Teil des Nationalen Forschungsprogramms NFP65 wurde innerhalb des Projektes Sustainable Urban Patterns (Wissen Hayek *et al.* 2013) unter anderem Geodesign als Methode eingesetzt. Mit dem Ziel der Entwicklung nachhaltiger urbaner Muster für die künftige Entwicklung von Städten und Agglomerationen in der Schweiz wurde eine Studie im Limmattal (Schweiz) durchgeführt (Wissen Hayek *et al.* 2012). Hierzu wurde eine interdisziplinär angelegte Kollaborationsplattform aufgebaut, welche an das Geodesign Konzept angelehnt ist (Wissen Hayek *et al.* 2012). Der Ansatz setzt stark auf die Zusammenarbeit zwischen Wissenschaft und verschiedenen Anspruchsgruppen aus Gesellschaft, Wirtschaft und Politik, um gesellschaftlich akzeptierte Lösungen für die Transformation und Entwicklung in Agglomerationsgebieten zu entwickeln (Wissen Hayek *et al.* 2014). Kern der Methodik bilden ein parametrischer Nutzungsplan mit 3D Visualisierung auf Basis der Esri CityEngine zur Exploration von Optionen, Methoden der multikriteriellen Entscheidungsanalyse zur Abstimmung sowie Abwägung zwischen heterogenen Zielsetzungen sowie ein integriertes Modell für Landnutzung und Transport zur Simulation möglicher Entwicklungsszenarien (Wissen Hayek *et al.* 2014). Während explizit auf das Geodesign Framework von Steinitz (2012) Bezug genommen wird (Wissen Hayek *et al.* 2014; Wissen Hayek *et al.* 2016) zeigt der schlussendliche Workflow im Wesentlichen eine eigenständige Struktur, welche stark auf partizipative Prozesse ausgerichtet ist (Wissen Hayek *et al.* 2016). Die Integration von Methoden und Modellen aus verschiedenen Bereichen, deren Nutzung einer gemeinsamen GIS Datenbasis zur Repräsentation des Raumes im Studiengebiet sowie deren Nutzung und iterative Anreicherung im Rahmen kollaborativer Prozesse sind aber in Einklang mit Kernelementen des Geodesign Konzepts (Wissen Hayek *et al.* 2016).

2.3 Raumplanung und Siedlungsentwicklung

2.3.1 Grundlagen der Raumplanung in der Schweiz

Die Raumplanung in der Schweiz unterliegt der föderalistischen Aufgabenteilung zwischen Bund, Kantonen und Gemeinden (GS-UVEK 2013). Abbildung 3 zeigt eine Übersicht der beteiligten Ebenen und der zugehörigen Instrumente.



Quelle: ARE (2012)

Abbildung 3: Raumplanungsinstrumente des Bundes

Basierend auf Artikel 75 der Bundesverfassung hat der Bund die Kompetenz zur Festlegung von Grundsätzen der Raumplanung (BV 1999). Darüber hinaus hat dieser Planungskompetenzen in gewissen Sachgebieten wie z.B. Eisenbahn- oder Luftverkehr (VLP-ASPAN 2014). Neben dem Erlass von gesetzlichen Grundlagen wie dem Raumplanungsgesetz werden auf Stufe Bund vor allem Konzepte und Sachpläne erarbeitet, die als Grundlagen für die Raumplanung in den Kantonen dienen. Die eigentliche Kompetenz der Raumplanung liegt laut dem Raumplanungsartikel der Verfassung bei den Kantonen und verfolgt das Ziel einer zweckmässigen und haushälterischen Nutzung des Bodens sowie der geordneten Besiedlung des Landes (BV 1999). Die auf Ebene Kanton erarbeiteten Richtpläne sind behördenverbindliche Instrumente mit dem Ziel einer umfassenden Gebietsplanung (GS-UVEK 2013). Zudem werden auf dieser Ebene Planungs- und Baugesetze erlassen, welche ebenso wie die Richtpläne durch den Bund genehmigt werden müssen (VLP-ASPAN 2014). Die unterste Stufe der Planung in Form von Nutzungsplänen erfolgt auf Ebene Gemeinde oder ausnahmsweise auf Ebene Kanton (GS-UVEK 2013). Diese Nutzungspläne sind allgemeinverbindlich und legen parzellenscharf fest, wie der Grund genutzt werden darf (VLP-ASPAN 2014). Insbesondere legen die Nutzungspläne fest, wo und in welchem Umfang gebaut werden darf (GS-UVEK 2013). Sie sind mit den kantonalen Richtplänen abzustimmen und durch den Kanton zu genehmigen (VLP-ASPAN 2014). Darüber hinaus können in den kommunalen Baugesetzen und Plänen die Bestimmungen der kantonalen Gesetzgebung ergänzt werden. Dabei können zudem gestalterische Vorschriften zur Abstimmung auf den lokalen Kontext eine Rolle spielen (Hagmann 2007). Speziell hervorzuheben ist, dass bereits im Raumplanungsartikel der Bundesverfassung ein Gebot zur Koordination der Bestrebungen von Bund und Kantonen festgehalten wird (BV 1999). Zusammen mit dem Ziel der geordneten Besiedlung des Raumes resultiert dies in einer Reihe von Mechanismen zur Abstimmung der raumwirksamen Tätigkeiten von Bund, Kantonen und Gemeinden (VLP-ASPAN 2014). Neben der Genehmigung von Planungen und Gesetzen gehören dazu auch Mitwirkungsverfahren, Vernehmlassungen sowie in vielen Fällen die Möglichkeit der direktdemokratischen Mitbestimmung (z.B. durch Abstimmungen oder Referenden) (VLP-ASPAN 2014).

2.3.2 Siedlungsentwicklung in der Schweiz

Bereits Mitte des 19. Jahrhunderts hat in der Schweiz im Zusammenhang mit der fortschreitenden Industrialisierung eine Tendenz zum Wachstum der Siedlungsfläche und damit einhergehend ein Prozess der Urbanisierung eingesetzt (Schmid 2007). Dabei sorgten geographische Eigenheiten und die föderalistische Struktur der Schweiz dafür, dass die Urbanisierung als dezentraler Prozess ablief, bei dem Siedlungsstrukturen immer mehr zusammenwuchsen (Schmid 2007). Als Folge gehörten 2014 fast die Hälfte der Schweizer Gemeinden zu einer Agglomeration (Goebel und Kohler 2014). Diese konzentrie-

ren sich auf vergleichsweise kleinem Gebiet. Je nach Quelle wird der Anteil der theoretisch überbaubaren Fläche der Schweiz mit 30% bis 42% angegeben (BFS 2005; ARE 2018a). Der restliche Anteil entfällt auf unproduktive und bestockte Flächen (ARE 2018a). Die effektiv überbaute Siedlungsfläche liegt bei 7.5%, Tendenz steigend (ARE 2018a). Gleichzeitig kann eine Entwicklung hin zu einem höheren Verbrauch von Wohnbaufläche pro Kopf und damit sinkender Einwohnerdichte beobachtet werden (Häussermann 2007). Infolge resultiert ein Siedlungswachstum nach aussen, das zu mehr Zersiedlung und einem weiteren Zusammenwachsen der Siedlungsstrukturen führt. Zusammen mit Tendenzen zu Verbuschung und Verwaldung im alpinen Raum trägt dieses Siedlungswachstum zu einem stetigen Verlust von Kulturland bei (ARE 2018a). Im Zusammenhang mit diesen Entwicklungen hat schon vor einiger Zeit eine Diskussion über Urbanität, Verdichtung sowie deren Qualität in der Schweiz eingesetzt (vergleiche Beiträge in Lampugnani *et al.* (2007)).

Die Annahme der Teilrevision des Schweizer Raumplanungsgesetzes zeigt, dass inzwischen in breiten Teilen der Bevölkerung eine Strategie mit Priorisierung einer Entwicklung nach innen als Mindeststrategie zur Minimierung des Bodenverbrauchs akzeptiert wird (Grams 2015, s.7). Mit der Revision des Raumplanungsgesetzes wurden die Planungsziele angepasst, um die fortschreitende Zersiedelung zu bremsen und Kulturland besser zu schützen (RPG, 2016). Das Siedlungswachstum soll eingedämmt und die Siedlungsentwicklung nach innen gelenkt werden (RPG, 2016). Mit dem Inkrafttreten der Revision müssen Bund, Kantone und Gemeinden ihre Planungsziele und -instrumente entsprechend anpassen, was diese vor verschiedene Herausforderungen stellt.

2.3.3 Dichte und Qualität

Ist von Dichte die Rede, steht oft die bauliche Dichte im Zentrum der Diskussion. Während die bauliche Dichte ein Schlüsselelement für die Siedlungsentwicklung im Innern darstellt (Grams 2015, s.11ff.), ist deren alleinige Betrachtung im Hinblick auf die Qualität der resultierenden Siedlungsstrukturen nicht ausreichend (Schmid 2007; Angélil *et al.* 2016). Auch das revidierte Raumplanungsgesetz hält fest, dass die Siedlungsentwicklung nach innen unter Berücksichtigung einer angemessenen Wohnqualität zu erfolgen hat (RPG 2016). Die bestimmenden Faktoren für die Wohnqualität werden im Gesetz nicht weiter ausgeführt. Das Bundesamt für Raumentwicklung führt aber die Siedlungs- und Wohnqualität im Kriteriensystem für Nachhaltige Raumentwicklung in der Schweiz als Teil der Zieldimension Gesellschaft auf, wobei als Kriterium die Durchmischung, Grünraum, soziale Aspekte, Verkehrserschliessung und Ortsbild genannt werden (ARE 2007). Andere Quellen betonen gleichfalls, dass eine umfassendere Betrachtung unter Berücksichtigung verschiedener Dichtebegriffe nötig ist (ROR 2012; Angélil *et al.* 2016). Häussermann (2007) nennt hierzu mit Verweis auf Spiegel (2000) die Begriffe bauliche Dichte (Mass der baulichen Nutzung des Grundstücks), Regelungsdichte (Anzahl gesetzlicher und sozialer Regeln), Einwohnerdichte (Aussendichte: Anzahl Einwohner pro Hektare; Innendichte: Anzahl Einwohner pro Wohnfläche), Beschäftigtendichte (Anzahl Beschäftigte pro Hektare) und Interaktionsdichte oder: soziale Dichte (Anzahl der Interaktionen in einer bestimmten Bevölkerungsgruppe). Zudem wird die «ökologische» Dichte als Mass für die zeitlichen und monetären Kosten zur Überwindung von Raumeinheiten genannt (Häussermann 2007). Grams (2015, s.18) gibt eine gute Übersicht zu diesen und weiteren Dichtebegriffen sowie alternativen Bezeichnungen.

Betrachtet man diese verschiedenen Faktoren, welche Dichte und Qualität in Siedlungsstrukturen beeinflussen, zeigt sich, dass die Thematik an einem Schnittpunkt verschiedener wissenschaftlicher Disziplinen angesiedelt ist. Dazu gehören unter anderem Architektur, Landschaftsplanung, Politikwissenschaften, Soziologie, Ökonomie und Geographie. Dies spiegelte sich auch im Nationalen Forschungsprogramm NFP 65 *Neue urbane Qualität* wider, in dessen Rahmen das Thema der Qualität in Siedlungsstrukturen im urbanen Bereich von Städten und Agglomerationen von verschiedenen Seiten beleuchtet wurde (NFP65 2017). Das Programm umfasste fünf Projekte, welche unterschiedliche Themen und Aspekte der urbanen Qualität behandelten. Dazu gehören allgemein Nachhaltige Siedlungsentwicklungsmuster (Wissen Hayek *et al.* 2013), die Analyse urbaner Potenziale und Entwicklung von Strategien in metropolitanen Territorien (Angélil 2013; Angélil *et al.* 2016), die Analyse kommunaler Entwicklungsprozesse und politisch-planerischer Strategien (Van Wezemael *et al.* 2014), die zukünftige Stadt- und Landschaftsgestaltung öffentlicher Räume im Tessin (Arnaboldi 2015) sowie das Thema Urban Gardening im Rahmen der Food Urbanism Initiative (Verzone 2015). Die Projektteams waren meist interdisziplinär aufgestellt, wobei ein Schwerpunkt im Bereich Architektur und Planung festzustel-

len ist. Die Ergebnisse des Programms fielen ebenfalls recht divers aus und wurden teilweise kontrovers diskutiert. Insbesondere die von der Programmleitung publizierten Synthesen (Sulzer und Desax 2015; Wehrli-Schindler 2015) wurden dahingehend kritisiert, dass Planung mit Forschung gleichgesetzt werde und teilweise Behauptungen aufgestellt werden, die kaum durch empirische Ergebnisse gedeckt seien und eher normativen Charakter hätten (Gantenbein 2015). Ähnliche Kritik wurde von verschiedenen Seiten geäussert (Honegger 2015). Einer der Teilprojektleiter hat ebenfalls darauf hingewiesen, dass die «Synthese» wenig mit den Erkenntnissen und Empfehlungen der Projekte zu tun habe (Angélil 2015). In verschiedenen Debatten sei man sich einig geworden, dass eine differenziertere Betrachtung angemessen wäre und eine wissenschaftlich begründete Terminologie erarbeitet werden sollte, wobei auch der Programmtitel im Allgemeinen hinterfragt wurde (Angélil 2015).

Mit einem Anspruch in diese Richtung wurde auf Basis des Projektes *Urbane Potenziale und Strategien in metropolitanen Territorien* ein Handbuch publiziert, dessen Fokus vorwiegend auf sozialen Aspekten der Dichte liegt (Angélil *et al.* 2016). Es wird versucht eine ganzheitlicher Sicht auf urbane Qualität unter Berücksichtigung der Faktoren Zentralität, Diversität, Interaktion, Zugänglichkeit, Adaptierbarkeit und Aneignung zu ermöglichen (Angélil *et al.* 2016). In drei Schritten werden die Ist-Situation analysiert, der Bestand beurteilt und schliesslich Strategien für künftige Entwicklungen abgeleitet (Angélil *et al.* 2016). Als Instrument für die Analyse, Beurteilung und Vergleich dienen dabei Profile, welche verschiedene Kriterien zu den oben genannten Faktoren zusammenfassen. Neben einigen quantitativen Kriterien liegt der Schwerpunkt dabei auf eher qualitativen Kriterien und weicheren Faktoren, welche vorwiegend im Bereich Architektur und Soziologie anzusiedeln sind.

Auch andere Ergebnisse von NFP65 Teilprojekten lassen wissenschaftlich unterlegte Vorgehensweisen und Ergebnisse erkennen. So liefert insbesondere das Projekt *Nachhaltige Siedlungsentwicklungsmuster* Hinweise auf die nachhaltige Planung und Entwicklung, wobei vor allem partizipative Prozesse im Mittelpunkt stehen (Wissen Hayek *et al.* 2013). Diese Prozesse basieren auf dem Geodesign Ansatz und binden wissenschaftliche Methoden und Erkenntnisse ein, um die Qualität der resultierenden Entwicklungen zu beurteilen und abzusichern (Wissen Hayek *et al.* 2012). Ebenfalls zu erwähnen sind die Ergebnisse aus den empirischen Analysen des Projekts *Urbane Brüche / lokale Interventionen,* welche in Empfehlungen zur Gestaltung in kommunalen Planungs-, Bau- und Entwicklungsprozessen münden (Van Wezemael *et al.* 2014). In Bezug auf Qualität wird hier eine Ausrichtung der Zielsetzung auf Basis einer differenzierten Sicht auf die lokale Situation statt auf den unreflektierten Rückgriff auf städtebauliche Mainstream-Konzepte gefordert, wozu auch die Partizipation der lokalen Bevölkerung und eine Ausrichtung der Kommunikation auf deren Bedürfnisse beiträgt (Van Wezemael *et al.* 2014).

2.3.4 Umsetzung von Siedlungsentwicklung nach innen

Wie schon erwähnt, müssen Bund, Kantone und Gemeinden ihre Planungsinstrumente infolge der Revision des Raumplanungsgesetzes anpassen und auf dessen Zielsetzungen abstimmen. Hierzu wurden von Bund und Kantonen verschiedene Empfehlungen und Dokumente publiziert, welche die Siedlungsentwicklung nach innen behandeln und Gemeinden bei deren Umsetzung unterstützen sollen (z.B. ARE ZH (2015); AGR (2016); Bundesrat (2017)). Es stehen verschiedene verbindliche (Nutzungsplan, Baureglement) und optionale (Leitlinien, Raumentwicklungskonzepte, Varianzverfahren, Sondernutzungsverfahren, Bodenpolitik) Policy Instrumente zur Verfügung, um Anreize für die Entwicklung nach innen zu setzen und die Erreichung von Qualitätszielen zu beeinflussen (Van Wezemael *et al.* 2014). Im Rahmen der periodischen Überarbeitung der Planungsinstrumente sind auf die lokalen Verhältnisse angepasste Ziele für die künftige Entwicklung zu definieren sowie die Policy Instrumente so anzupassen und zu kombinieren, dass die Erreichung dieser Ziele unterstützt und dichteres Bauen begünstigt wird (Van Wezemael *et al.* 2014; ARE ZH 2015). Zu den möglichen Stellschrauben gehören unter anderem Grenzabstände, erlaubte Höhen beziehungsweise Geschosszahlen, höhere Nutzungsziffern oder die Verwendung von Überbauungsziffern statt Ausnützungsziffern (Gilgen 2016; Gilgen und Walczak 2017). Gleichermassen können flankierende Bestimmungen Anreize für eine verdichtete Bauweise geben. In einem Positionspapier empfiehlt der Rat für Raumordnung zum Beispiel, dass für Neueinzonungen mit Wohnnutzung W3 (Wohnnutzung dreigeschossig) als Mindeststandard angestrebt wird und Bauten <W3 einer Begründung bedürfen (ROR 2012).

Um eine Siedlungsentwicklung im Inneren zu bewerkstelligen, müssen bestehende Siedlungsflächen besser ausgenutzt und bestehende Baulücken geschlossen werden (Grams 2015, s.7). Letzteres beinhaltet die Umnutzung von (Industrie-)Brachen (Grams 2015, s.16). Für die Nachverdichtung und damit besseren Ausnützung bereits bebauter Siedlungsflächen stehen als Optionen Aufstockungen, Ergänzungsbauten und Ersatzneubauten zur Verfügung (ARE ZH 2015; Grams 2015, s.16). Ganz allgemein ist zudem eine Abstimmung der Bauzonenreserven auf den tatsächlichen Bedarf nötig, wodurch sich Einschränkungen bei neuen Einzonungen ergeben (ARE ZH 2015; AGR 2016). So können neue Einzonungen mitunter nur noch in begründeten Fällen und in Regionalen Zentren vorgenommen werden (AGR 2016). Fallweise ist bei der Einzonung von Bauland eine Kompensation durch Auszonungen an anderen Stellen zu prüfen, um den Erhalt von Kulturland zu gewährleisten (Grams 2015, s.7). Zudem ist mitunter eine regionale Zusammenarbeit einerseits in Hinblick auf den Aufbau planerischer Kapazitäten und Prozesse sowie andererseits in Bezug auf eine übergreifende Planung und Diversifizierung bei Nutzungen zu empfehlen (Van Wezemael *et al.* 2014).

Während die Zielrichtung bei der Siedlungsentwicklung nach innen weitgehend klar ist, stellt die erfolgreiche Umsetzung oft eine Herausforderung dar. Aus den breit gefächerten Einflussfaktoren der Dichte ergeben sich verschiedene Hindernisse, welche sich in vier Hauptkategorien unterteilen lassen (Bundesrat 2017):

1. Soziokulturelle Hindernisse (gesellschaftlicher Widerstand, negative Wahrnehmung)
2. Rechtliche Hindernisse (Garantie des Privateigentums, unangemessene Reglemente und Pläne, komplexe und nicht abgestimmte Raumplanungsbestimmungen und -verfahren)
3. Technische Hindernisse (unklare oder fehlende Vorstellung der räumlichen Entwicklung, fehlende Übereinstimmung von Planung und Nachfrage, komplexes politisches Umfeld, ungenügende Kompetenzen, Gewohnheiten und Ressourcen der Gemeinwesen)
4. Wirtschaftliche Hindernisse (Komplexität der Finanzierung der Infrastruktur, mangelnde wirtschaftliche Vorteile, fehlende Investoren in risikobehafteten Situationen)

Die Schaffung von Akzeptanz in der Bevölkerung und Politik ist zentral für die Überwindung dieser Hindernisse und die erfolgreiche Umsetzung der Siedlungsentwicklung nach innen (ARE ZH 2015; Grams 2015, s.77f; Bundesrat 2017). Der Dialog mit der Bevölkerung und die Involvierung der breiten Öffentlichkeit sind hierfür entscheidend (ARE ZH 2015; AGR 2016). Dies hat unmittelbaren Einfluss auf die Ausgestaltung der Prozesse, welche zur Überarbeitung der Planungsinstrumente führen. Diese können nicht einfach als rein lineare Verfahren oder Projekte ausgestaltet werden (Van Wezemael *et al.* 2014; ARE ZH 2015). Vielmehr ist für die Überarbeitung eine Kombination aus informellen und formellen Verfahren angebracht, welche die Öffentlichkeit in unterschiedlichem Grad involvieren (Van Wezemael *et al.* 2014; Grams 2015, s.123ff). Zudem ist über diese Prozesse hinaus eine Fortschreibung im Sinne eines laufenden Monitorings der Umsetzung und einer kontinuierlichen Verbesserung angebracht (ARE ZH 2015; AGR 2016).

Die genaue Ausgestaltung des Überarbeitungsprozesses hängt jeweils von den lokalen Bedingungen ab. Eine Rolle spielen unter anderem gesetzliche Vorgaben, die Grösse der Gemeinde, die politische Landschaft, die verfügbaren Mittel und der angestrebte Grad an Partizipation (Van Wezemael *et al.* 2014; ARE ZH 2015; Grams 2015, s.137ff; AGR 2016). Im Allgemeinen werden aber zuerst informelle Verfahren durchgeführt, bevor die formalen Verfahren zur Überarbeitung der Planungsinstrumente in Angriff genommen werden (ARE ZH 2015; Grams 2015, s.123ff; AGR 2016). Den informellen Verfahren kommt dabei eine vorbereitende und klärende Wirkung zu, welche zur Robustheit der nachfolgenden formellen Verfahren beitragen (Grams 2015, pp.77f, 133). Unter Partizipation der Öffentlichkeit werden im informellen Rahmen zunächst abstrakte Konzepte und Leitbilder erarbeitet, die in anschliessenden Schritten konkretisiert werden (ARE ZH 2015; Grams 2015, s.156f; AGR 2016).

Gerade bei komplexen Fragestellungen in Verdichtungs- und Umstrukturierungsgebieten und dem Fehlen eines gemeinsamen Verständnisses bieten Methoden wie Wettbewerbe, Studienaufträge oder Testplanungen im Rahmen von informellen Verfahren die Möglichkeit nach Lösungsansätzen zu suchen (ARE ZH 2015; AGR 2016). Insbesondere die Methode der Testplanung als kooperatives Planungsverfahren ist dabei hervorzuheben (Scholl *et al.* 2013). Im Sinne eines Wettbewerbs der Ideen wird dabei mit einem hohen Grad an Freiheit nach möglichen Lösungsansätzen für eine konkrete planerische Fragestellung gesucht (Scholl *et al.* 2013; Grams 2015, s.64ff). Analog zur Hypothesenbildung

bilden die resultierenden Entwürfe Prozess- und Handlungsvorschläge, die es zu testen und widerlegen gilt (Grams 2015, s.174). Der im Rahmen der Überprüfung der Testentwürfe stattfindende Austausch zwischen Auftraggeber, Planern und meist auch Öffentlichkeit trägt zur Bildung eines gemeinsamen Verständnisses und zur Beantwortung von Fragen bei, deckt mitunter aber auch neue Fragestellungen auf (Scholl *et al.* 2013; Grams 2015, s.64ff). Die Konkurrenz zwischen verschiedenen Entwürfen, deren Kritik und Verfeinerung erhöht insgesamt die Qualität der Ergebnisse (Scholl *et al.* 2013; Grams 2015, s.76f; AGR 2016). Allgemein passt die Methode des Testentwurfs gut zu solchen raumplanerischen Verfahren, da starke Parallelen zu der in der Schweiz verankerten demokratischen Kultur und deren Verfahren bestehen (Scholl *et al.* 2013). Andererseits ist in der Raumplanung weder eine reine Abstützung auf Prognosen, die auf Basis schwer nachvollziehbarer Modelle erstellt wurden, noch die rein lineare Fortschreibung der historischen Entwicklung ratsam (Grams 2015, s.174). Vielmehr sollte die Raumplanung Ideen für mögliche Zukünfte liefern, was im Ideenwettbewerb einer Testplanung umgesetzt werden kann (Grams 2015, s.174). Im konkreten Fall der Siedlungsentwicklung nach innen können Testentwürfe helfen aufzuzeigen, warum eine Erhöhung der Dichte im lokalen Kontext sinnvoll ist (Grams 2015, s.157). Die Auslotung planerischer Fragestellungen kann zudem Grundlagen für fachliche und politische Diskussionen liefern (Gilgen 2016). Insgesamt helfen Testplanungen sowie die Kombination von informellen und formellen Verfahren bei der Schaffung eines gemeinsamen Referenzrahmens (Grams 2015, s.162). Sie erhöhen die Akzeptanz von Verdichtung und der damit zusammenhängenden Prozesse (Grams 2015, s.162).

2.3.5 Kommunikation

Spezielles Augenmerk verdienen die Kommunikation und Vermittlung von Informationen in Planungsprozessen. Aufgrund des Milizsystems in Schweizer Politik und der vermehrten Involvierung der Öffentlichkeit, welche nicht über Fachwissen zum planerischen Fachjargon verfügen, ist für den Erfolg von Planungsprozessen eine auf die Zielgruppe abgestimmte Kommunikation nötig (Scholl *et al.* 2013; Van Wezemael *et al.* 2014). Einerseits sollte ein übergreifendes Konzept und eine durchgehende Planung der Kommunikation mit der Öffentlichkeit über den ganzen Planungsprozess hinweg erstellt werden (Scholl *et al.* 2013; AGR 2016). Andererseits sind planerische Inhalte in einer für die Zielgruppe verständlichen Form aufzubereiten und zu vermitteln (Scholl *et al.* 2013; Van Wezemael *et al.* 2014). Gerade Modelle, Visualisierungen und Skizzen sind dabei wichtige Hilfsmittel, deren Bedeutung mit zunehmender Komplexität der Planungsfragen zunimmt (Walz *et al.* 2008; Van Wezemael *et al.* 2014). Visualisierungen von Modellen können zur Kommunikation zusätzlicher Informationen genutzt werden, um (politische) Entscheidungsfindungsprozesse zu unterstützen (Walz *et al.* 2008). Sowohl grosse Einzeleingriffe in der Landschaft als auch kleinere Änderungen oder schleichende Veränderungsprozesse können durch solche Visualisierungen in eine intuitiv begreifbare Form gebracht werden (Walz *et al.* 2008). Die Etablierung von Modellen in der Planungspraxis wird aber durch technische Probleme (Datenverfügbarkeit und -qualität, fehlende Schnittstellen zwischen Software-Paketen) sowie mangelnde Erfahrung im Umgang mit Modellen behindert (Walz *et al.* 2008).

In Bezug auf Verdichtung können dreidimensionale Darstellungen helfen, den baulichen Bestand, Flächenreserven und nicht ausgeschöpfte Potentiale anschaulich zu vermitteln (Grams 2015, s.150). Differenzen zwischen realisierter und zulässiger Dichte lassen sich gemeinsam darstellen und lokale Häufungen sowie Muster sind einfacher zu erkennen (Grams 2015, s.150). Dies kann helfen, Handlungsspielräume in das Bewusstsein von Grundeigentümern zu rufen und Zusammenarbeit anzuregen (Grams 2015, s.150). Darüber hinaus eröffnet der Einsatz von parametrischen Modellen zusätzliche Möglichkeiten. Durch Veränderung der Parameter können verschiedene Varianten einer Planung erzeugt und visualisiert werden (Beirão *et al.* 2012; Neuenschwander *et al.* 2014). Insbesondere durch die interaktive Anpassungen und Visualisierung der resultierenden Änderungen können die Auswirkungen einzelner Parameter auf den Raum besser kommuniziert und für beteiligte Anspruchsgruppen verständlich gemacht werden (Moura 2015; Wissen Hayek *et al.* 2016). Für beteiligte Architekten und Planer kann zudem das Verständnis und die Zugänglichkeit solcher Prozesse vereinfacht werden, wenn parametrische Design Tools als Grundlage genutzt werden, mit denen sie aus ihrem Arbeitsalltag vertraut sind (Speranza 2016).

2.3.6 Geodesign in Planungsprozessen

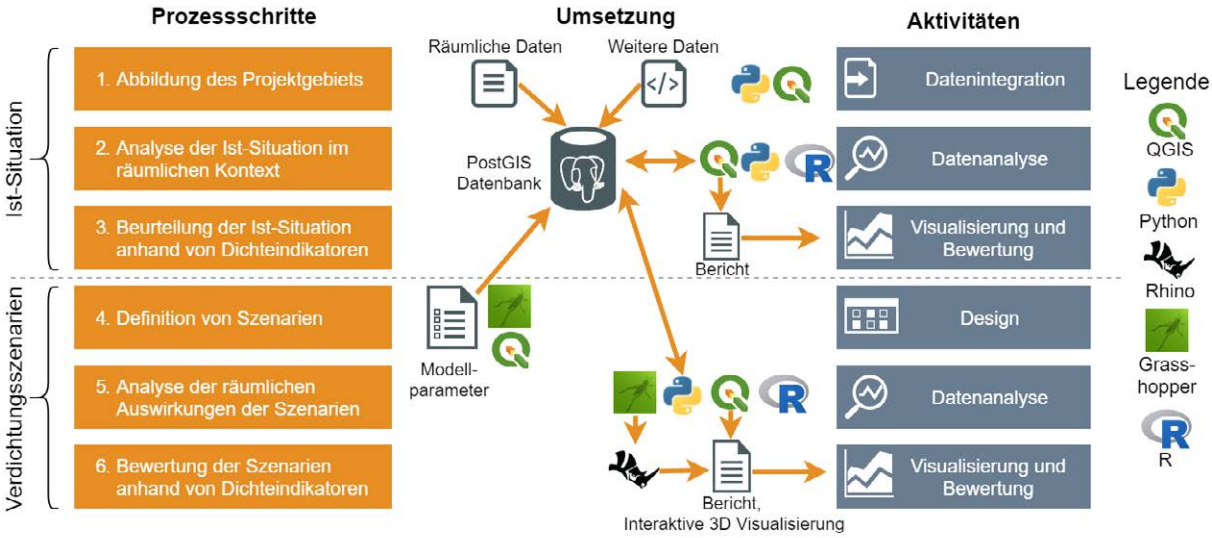
Planungsprozesse betreffen gestaltende, raumwirksame Eingriffe, die von einer Vielzahl von Faktoren beeinflusst werden (Dangermond 2010; Goodchild 2010; Wissen Hayek *et al.* 2016). Durch den inhärenten Raumbezug spielt GIS bei der Unterstützung und Bewertung von Planungsprozessen eine zentrale Rolle (Dangermond 2010; Wissen Hayek *et al.* 2016). In vielen Fällen geschieht dies implizit durch die Bereitstellung von Grundlagen und Plänen. Darüber hinaus bietet sich das Konzept des Geodesign an, um die gestalterischen Entscheide (Design) auf Basis wissenschaftlich fundierter Methoden und Informationen zu treffen, wobei Tools und Methoden der Geoinformatik zum Einsatz kommen (Goodchild 2010; Miller 2012). Werden parametrische Modelle im Rahmen des Geodesign Prozesses eingesetzt, so stellt deren Verknüpfung mit Inputs aus GIS und aus anderen Modellen sicher, dass die resultierenden Design Entscheidungen auf gesicherten Informationen und in Abstimmung mit dem räumlichen Kontext erfolgen und das Resultat nicht nur von rein ästhetischen Überlegungen geleitet wird (Goodchild 2010; Mehaffy 2011; Beirão *et al.* 2012). Ganz allgemein trägt der systemische Ansatz und die Integration von Methoden verschiedener Disziplinen (Dangermond 2010; Steinitz 2016) dazu bei, dass im Prozess eine umfassende und auf Nachhaltigkeit ausgerichtete Betrachtungsweise verankert wird, die im Zusammenhang mit der qualitätsvollen Siedlungsentwicklung nötig ist.

Das von Steinitz (2012) vorgeschlagene Framework für Geodesign hat sich als geeignete Grundlage für die Gestaltung solcher Prozesse herausgestellt (Moura 2015; Wissen Hayek *et al.* 2016). Die Bandbreite der im Rahmen von Geodesign eingesetzten Methoden der Geoinformatik ist vielfältig und abhängig vom jeweiligen Planungsziel. Beispiele umfassen unter anderem räumliche Optimierung (Goodchild 2010), Simulationen (Efthymiou *et al.* 2014), Sichtbarkeitsanalysen (Wissen Hayek *et al.* 2013), Szenario Analyse (Wissen Hayek *et al.* 2016) und Multikriterien-Evaluation (Moura 2015).

3 Vorgehensmodell zur Planungsunterstützung

3.1 Übersicht

Das Geodesign Framework nach Steinitz (2012) diente in der vorliegenden Master Thesis sowohl als Grundlage für das auszuarbeitende Vorgehensmodell als auch der Strukturierung des Ablaufs der Arbeit selbst. In Anlehnung an das Steinitzsche Framework wurde ein Prozess mit sechs Schritten ausgearbeitet, welche in der Übersicht in Abbildung 4 auf der rechts zu sehen sind. Der erste Teil des Ablaufs ist auf die Abbildung und Analyse der Ist-Situation im Projektgebiet ausgerichtet. Hierfür werden 1. die benötigten Daten zum Projektgebiet integriert, 2. im räumlichen Kontext analysiert und 3. für die Beurteilung visualisiert und aufbereitet. Der zweite Teil zielt auf die Erstellung und Bewertung verschiedener Szenarien für Verdichtung im Projektgebiet, wobei 4. die Parameter für verschiedene Szenarien definiert sowie die resultierenden Verdichtungsvarianten generiert, 5. deren räumliche Auswirkungen analysiert und 6. die Resultate für die Bewertung und den Vergleich aufbereitet und visualisiert werden.



Quelle: eigene Darstellung

Abbildung 4: Übersicht Vorgehensmodell zur Planungsunterstützung

Für die Ausarbeitung des Vorgehensmodells wurden wie von Steinitz (2012) vorgeschlagen mehrere Iterationen durchlaufen, bei denen Inputs gesammelt, strukturiert sowie schrittweise das Vorgehen definiert und verfeinert wurde. Strukturiert anhand der sechs Schritte für einen Geodesign Prozess wurde in einem ersten Durchlauf das Verständnis für das Untersuchungsgebiet und den Hintergrund der Analyse vertieft sowie versucht, diese genauer zu verstehen (WHY). In einem zweiten Durchlauf wurden die zugehörigen Methoden für die Analyse definiert (HOW). Im dritten Durchlauf wurde das eigentliche Vorgehen bei der Untersuchung genauer umrissen (WHAT WHERE WHEN). Diese Abfolge wurde mehrmals durchlaufen, wobei jedes Mal Verfeinerungen und Ergänzungen vorgenommen wurden.

Das so definierte Vorgehensmodell ist eine auf die Fragestellung der inneren Verdichtung zugeschnittene Version eines Geodesign Prozesses. Es ist als integriertes Gesamtmodell für Testplanungen zum Thema innere Verdichtung konzipiert, in welchem Vorgehen, Datenmodell und technische Umsetzung aufeinander abgestimmt sind. Die analog zum Steinitzschen Framework definierten Prozessschritte beleuchten jeweils spezifische Teilaspekte und Zusammenhänge des Gesamtmodells und die jeweils zugehörigen Modelle können entsprechend als Teilmengen des Gesamtmodells betrachtet werden. Wie in Abbildung 4 in der Mitte angedeutet ist, werden die Schritte durchgehend von einer technischen Lösung unterstützt. Details zu deren Umsetzung können Kapitel 4 entnommen werden. Das Gesamtsystem setzt sich aus verschiedenen Komponenten aus dem Bereich GIS (u.a. QGIS, PostGIS), Parametrisches Design (Rhino und Grasshopper) und Statistik (R) zusammen. Zentral für deren Integration ist die PostGIS Datenbank, welche alle für das Vorgehensmodell benötigten Daten umfasst. Abgesehen von den Werkzeugen zur initialen Datenintegration müssen die einzelnen Systemkomponenten nur noch eine einzelne Schnittstelle zur Datenbank unterstützen und Probleme bei Datenaustausch wie z.B. fehlende Formatunterstützung oder Versionskonflikte werden vermieden. Eine weitere wichtige

Rolle für die Integration der Systemkomponenten spielt Python, das für die Verknüpfung und Koordination der verschiedenen Programme auf funktionaler Ebene genutzt wird. Dank der Abstimmung von Vorgehen, Datenmodell und technischer Unterstützung können effizient Verdichtungsplanungen erstellt werden.

Das Vorgehensmodell zielt dabei nicht auf die Erstellung vollständig ausgearbeiteter Planungen im Rahmen formeller Verfahren. Absicht ist vielmehr die effiziente Abwicklung von Testplanungen, die im Rahmen von informellen Verfahren unter Einbezug verschiedener Anspruchsgruppen die Erarbeitung von Ideen, die Schaffung eines gemeinsamen Referenzrahmens und die Formulierung von Argumenten für nachfolgende politische Entscheide oder formelle Festlegungen unterstützen sollen (Scholl *et al.* 2013; Grams 2015, s.162; Gilgen und Walczak 2017). Es ist anzumerken, dass die Umsetzung des Prozesses in der Arbeit nicht im Rahmen eines laufenden Planungsverfahrens erfolgte und darüber hinaus keine Beteiligung der Öffentlichkeit stattfand. Sobald der Prozess aber konkret genug umrissen war, wurde mit der Umsetzung anhand eines Fallbeispiels in der Gemeinde Langenthal begonnen. Das betrachtete Projektgebiet war bereits Subjekt von Testplanungen und einer Studie seitens des Kompetenzbereichs Dencity der Berner Fachhochschule (Gilgen 2016; Gilgen und Walczak 2017). Während der Umsetzung fand zudem ein Austausch mit den Architekten des Dencity zum erarbeiteten Modell statt. Aus diesem Austausch und der Implementierung selbst ergaben sich weitere Inputs, welche in das Vorgehensmodell eingeflossen sind. In den nachfolgenden Abschnitten werden die einzelnen Schritte des Vorgehens aus methodischer und konzeptioneller Sicht genauer beschrieben.

3.2 Abbildung des Projektgebiets

3.2.1 Representation Model

Im ersten Schritt des Prozesses wird das Projektgebiet in Form eines Representation Models abgebildet. Dieses beantwortet die Frage, wie das Projektgebiet inhaltlich, zeitlich und räumlich beschrieben werden soll (Steinitz 2012). Die dadurch entstehende Abbildung der Realität schafft für die am Geodesign Prozess beteiligten Parteien die Grundlage für das Verständnis der aktuellen Situation sowie für die Auswirkungen der vorgeschlagenen Eingriffe in die Landschaft (Steinitz 2012; Moura 2015). Tabelle 1 gibt eine Übersicht der Elemente des Representation Models. Räumlich konzentriert sich dieses auf die baurechtlich relevante Unterteilung des Projektgebiets und des umgebenden Siedlungsgebiets. Der inhaltliche Fokus liegt auf der Nutzung des Siedlungsgebietes. Zeitlich wird auf den aktuellen Stand fokussiert. Die historische Entwicklung wird ausgeklammert. Die Elemente des Representation Models werden als Teil des Berichts zur Analyse der Ist-Situation aufbereitet und dem Benutzer zur Verfügung gestellt (siehe Kapitel 4.2.3 für Details zur Umsetzung).

Tabelle 1: Elemente des Representation Models

Element	Definition und Einsatzgebiet	Flächeneinheit
Parzellierung	Baurechtliche Aufteilung des Projektgebiets und Umgebung	Parzelle
Perimeter	Parzellen des Projektperimeters	-
Nutzungszonen	Baurechtliche Nutzung im Projektgebiet anhand generalisierter Bauzonen	Bauzonen
Bodenbedeckung	Bodennutzung nach Kategorisierung gemäss Informationsebene Bodenbedeckung der Amtlichen Vermessung (KKVA 2011)	-
Gebäude	Grundrisse sowie 3D Visualisierung bestehender Gebäude	Parzelle
Terrain	Höhenmodell für Topographie und als Kontext für 3D Visualisierung	-
Strassenflächen	Strassenflächen als Kontext für die 3D Visualisierung	-

3.2.2 Aufteilung des Projektgebietes

Das Representation Model zielt auf die Abbildung des aktuellen Zustandes im Siedlungsgebiet, in dem sich der Projektperimeter befindet. Der Hauptteil des ersten Prozessschrittes besteht deshalb darin, entsprechende Daten zu sammeln, zu integrieren und zu verstehen. Bei der Auswahl der Daten wurde darauf geachtet, dass diese in gleicher oder ähnlicher Form möglichst für die ganze Schweiz verfügbar sind, um die Übertragbarkeit des Vorgehens auf andere Orte zu vereinfachen. Das Schwergewicht der integrierten Daten liegt auf der Aufteilung und Nutzung des Siedlungsgebietes.

Die Parzellen beziehungsweise Grundstücke zeigen die grundrechtliche Unterteilung des Siedlungsgebietes auf. In Hinblick auf die Bebauung bildet die einzelne Parzelle den Ausgangspunkt für die Bestimmung der bebaubaren Flächen. Die Festlegung des Studiengebiets für den Geodesign Prozess erfolgt deshalb durch Ausscheidung einer Gruppe von Parzellen, welche gemeinsam den Projektperimeter bilden. Diese Ausscheidung erfolgt zu Beginn des ersten Prozessschrittes auf Basis der integrierten Daten zu den Parzellen. Der resultierende Perimeter beziehungsweise dessen Parzellen werden im weiteren Verlauf des Prozesses an verschiedenen Stellen als Input genutzt oder aber als Kontextinformation angezeigt, um dem Benutzer das Studiengebiet zu verdeutlichen.

3.2.3 Nutzung des Siedlungsgebietes

Neben der Aufteilung des Siedlungsgebiets liegt der zweite Schwerpunkt des Representation Models auf dessen Nutzung. Nutzungszonen geben Auskunft über die baurechtliche Nutzung des Siedlungsgebietes (Wohnen, Gewerbe, Mischzonen, öffentliche Nutzungen, Tourismus usw.). Da die Nutzungsplanung in der Verantwortung der Gemeinden liegt (GS-UVEK 2013), ergibt sich hier fast in jeder Gemeinde eine individuelle Situation mit eigenen Zonen und Vorschriften. Dies schränkt die Vergleichbarkeit ein und führt zu grösseren Herausforderungen bei der Erstellung eines standardisierten Prozesses. Aus diesem Grund wurden für das Vorgehen aggregierte Daten einheitlicher Qualität von einer möglichst hoch angesiedelten Verwaltungseinheit als Quelle ausgesucht. Deshalb werden für das Representation Model vorrangig die harmonisierten Bauzonen gemäss dem minimalen Geodatenmodell Nutzungsplanung (ARE 2011) eingebunden. Für diese liegt ein schweizweiter Datensatz vor, der durch die Aggregation und Vereinheitlichung von Daten der Gemeinden durch Kantone und Bund entsteht. Dieser Datensatz unterscheidet neun verschiedene Kategorien von Bauzonen. Während durch die Harmonisierung einerseits eine Vergleichbarkeit bei Analysen sichergestellt wird, gehen durch den Prozess der Aggregation und Zuordnung der Zonen zu den harmonisierten Kategorien andererseits Informationen verloren, die normalerweise im z.B. im Zonentitel oder zusätzlichen Attributen enthalten sind. Insbesondere betrifft dies die erlaubte Geschosszahl in den Zonen. Deshalb werden im Representation Model als zusätzliche Ebene noch die Bauzonen des Übersichtszonenplan des Kantons Bern eingebunden. Dieser enthält durch den Kanton aggregierte und standardisierte Bauzonen, welche in 24 Kategorien unterteilt sind. Diese stellen einen Kompromiss zwischen den lokalen, vollständig detaillierten und den harmonisierten Bauzonen dar. Die Kategorien des Übersichtszonenplans unterscheiden z.B. vier Typen von Wohnzonen mit unterschiedlicher Geschosszahl statt nur einem Typ, wie es bei den harmonisierten Kategorien des ARE Datensatzes der Fall ist. Andere Kantone stellen ähnliche Übersichtspläne zur Verfügung. Die Überlagerung der Bauzonen mit den Parzellen gibt Auskunft darüber, welche Flächen tatsächlich bebaut werden dürfen und wie diese genutzt werden können.

Unabhängig von der baurechtlichen Nutzung gibt die Bodenbedeckung Auskunft über die Art der Bodennutzung und vermittelt einen Eindruck von der Situation im Siedlungsgebiet. Die Bodenbedeckung wird als Teil der Amtlichen Vermessung gemäss der Richtlinie zum Detaillierungsgrad der Informationsebene Bodenbedeckung der Amtlichen Vermessung (KKVA 2011) weitgehend einheitlich und flächendeckend erfasst. Aktuell werden die Daten zu flächenhaften Elementen integriert, welche die Bodennutzung in 26 Kategorien einteilt (u.a. Gebäude, Strasse, Trottoir, Grünflächen). Die erfassten Flächen können dabei über Parzellengrenzen hinweggehen. Aus den Informationen der Bodenbedeckung werden zusätzlich die Gebäudegrundrisse extrahiert. Diese werden an verschiedenen Stellen im Kontext der Parzellen angezeigt, um einen Eindruck von der aktuellen Bebauung zu vermitteln.

3.2.4 Dreidimensionale Visualisierung

Die bisher erwähnten Bestandteile des Representation Models basieren auf klassischen GIS Daten und werden als zweidimensionale Karten visualisiert. Das dreidimensionale, parametrische Modell des Impact Models, setzt auch eine dreidimensionale Visualisierung eines Teils der Ist-Situation um, welche als Teil des Representation Models betrachtet werden kann. Einerseits werden vereinfachte 3D Gebäude dargestellt, welche aus dem swissBUILDINGS3D 2.0 Datensatz der swisstopo stammen (swisstopo 2017). Diese sind Teil des Topographischen Landschaftsmodells der swisstopo und umfassen vektorielle Gebäudemodelle für Gebäude ab 8m mal 3m Meter, welche den LoD2 (Level of Detail) Kriterien der CityGML Spezifikationen (Gröger *et al.* 2012) entsprechen (swisstopo 2017). Andererseits wird als «Unterlage» für die Gebäude ein Höhenmodell angezeigt, das einen Eindruck der Topographie im Siedlungsgebiet vermittelt. Aus der Bodenbedeckung extrahierte Strassenflächen, die über das Terrain gelegt werden, bieten eine zusätzliche Orientierungshilfe.

3.3 Analyse der Ist-Situation im räumlichen Kontext

3.3.1 Process Model

Im zweiten Schritt des Prozesses geht es darum, zu verstehen, wie das Projektgebiet funktioniert (Steinitz 2012). Das so genannte Process Model, stellt hierzu Informationen zu den funktionellen und strukturellen Beziehungen zwischen Elementen im Projektgebiet bereit, welche als Grundlage für die Analysen und Beurteilung des Gebietes dienen (Steinitz 2012). Tabelle 2 unten gibt eine Übersicht der Elemente des Process Models. Dieses fokussiert einerseits auf die Analyse der aktuellen baulichen Struktur. Andererseits werden verschiedene Aspekte der Dichte im Projektgebiet analysiert. Die Ergebnisse sind Hauptbestandteil des Berichtes zur Analyse der Ist-Situation, der als Grundlage für die Bewertung im anschliessenden Schritt dient (siehe Kapitel 4.2.3 für Details zu Analyse und Umsetzung). Die Definition und Umsetzung der Elemente des Process Models, insbesondere in Bezug auf die baulichen Grössen, sind dabei auf die Definitionen der Parameter im Change Model und die Elemente im Impact Model abgestimmt.

Tabelle 2: Elemente des Process Models

Element	Definition und Einsatzgebiet	Flächeneinheit	Quantifizierung
Art der Bebauung	Beurteilung der Gebäudetypologie der aktuellen Bebauung	-	Typ pro Gebäude
	Beurteilung von ungefährer Gebäudehöhe, Geschosszahl und Geschossfläche des aktuellen Gebäudebestandes	Bauzone	Durchschnittliche Gebäudehöhe in m
		Bauzone	Durchschnittliche Geschosszahl
		Bauzone	Durchschnittliche Geschossfläche in m² pro Gebäude
Bauliche Dichte	Abschätzung der überbauten Fläche, der Geschossfläche und dem überbauten Volumen pro anrechenbarer Grundstücksfläche	Parzelle	Überbauungsziffer
		Parzelle	Geschossflächenziffer
		Parzelle	Baumassenziffer
	Überbautes Volumen pro Flächeneinheit	Hektare	m³/ha
Freiflächenanteil	Anteil nicht überbauter Fläche pro Flächeneinheit	Hektare	%
Erschliessungsgüte öffentlicher Verkehr	Verkehrserschliessung öffentlicher Verkehr nach Güteklassen (ARE 2018b)	Zonen	Güteklasse pro Zone
Raumnutzerdichte	Anzahl Einwohner und Beschäftigte pro Flächeneinheit	Hektare	Raumnutzer pro ha
Einwohnerdichte	Anzahl ständige Einwohner pro Flächeneinheit und deren Verteilung pro Altersklasse gemäss Statistik der Bevölkerung und der Haushalte des BFS	Hektare	Einwohner pro ha
		Hektare	Verteilung der Einwohner nach Altersklassen
Beschäftigtendichte	Anzahl Beschäftigte pro Flächeneinheit und deren Verteilung nach Sektor gemäss Statistik der Unternehmensstruktur des BFS	Hektare	Beschäftigte pro ha
		Hektare	Verteilung Beschäftigte nach Sektor
Wohnungsdichte	Anzahl Wohnungen pro Flächeneinheit sowie deren Verteilung nach Anzahl Zimmern gemäss Gebäude- und Wohnungsstatistik des BFS	Hektare	Anzahl Wohnungen pro ha
		Hektare	Verteilung der Wohnungen nach Kategorien
Durchschnittliche Wohnfläche	Durchschnittliche Wohnfläche pro Einwohner	Hektare	m² pro Einwohner
Haushaltsdichte	Abschätzung der tatsächlichen Wohnverhältnisse anhand der Anzahl Haushalte pro Flächeneinheit und deren Verteilung nach Haushaltsgrösse gemäss Statistik der Bevölkerung und der Haushalte des BFS	Hektare	Anzahl Haushalte pro ha
		Hektare	Verteilung der Haushalte nach Kategorien

3.3.2 Art der Bebauung

Ein erster Teil des Process Models zielt auf das Verständnis der Struktur des aktuellen Gebäudebestandes. Die Siedlungs- oder Gebäudetypologie steht im Zusammenhang mit der Qualität städtischer Gestaltungskriterien wie Zugänglichkeit, Zusammenhang, Lesbarkeit und Identität (Kunze *et al.* 2012). Diese Faktoren beeinflussen wiederum massgeblich die Wahrnehmung des Raumes und dessen Qualität in architektonischer und gestalterischer Hinsicht (Grams 2015; Angélil *et al.* 2016). Zudem besteht ein Zusammenhang zwischen der baulichen Dichte und der Typologie, da die zulässigen Dichteziffern die umsetzbaren Typologien beeinflussen und umgekehrt gewisse Dichtewerte nur mit bestimmten Typologien überhaupt umgesetzt werden können (Grams 2015, s.23ff). Der Kontext der vorhandenen Typologie beeinflusst zudem die Optionen und Handlungsspielräume für die Verdichtung (Grams 2015, pp.133-136). Im Rahmen der Arbeit wurde auf eine detaillierte Bau- und Siedlungsstrukturanalyse verzichtet (vergleiche Schwalbach (2009) für eine detaillierte Beschreibung). Vielmehr wurde zur Reduktion der Komplexität ein Ansatz auf Basis einer vereinfachten Gebäudetypologie gewählt, wie sie in ähnlicher Form auch in anderen Arbeiten zur Anwendung kommt (Wissen Hayek *et al.* 2013; Huber und Walczak 2016a; Walczak 2017). Als Kompromiss zwischen Detaillierung und einer guten Übersicht wurde basierend auf den recht zugänglichen Beschreibungen von Bürklin und Peterek (2008) eine Typologie definiert, welche folgende Typen unterscheidet:

1) Einzelhäuser
   Freistehende Gebäude mit kleiner bis mittlerer Grundfläche (bis 2500m^2) und einfachem, annähernd rechteckigem Grundriss.
2) Doppelhäuser
   «Zweifamilienhäuser»: zwei aneinander gebaute Häuser mit einer kleinen Grundfläche (bis 250m^2) und einfachem, annähernd rechteckigem Grundriss.
3) Reihenhäuser
   Mehrere aneinander gebaute Häuser in annähernd linearer Anordnung.
4) geschlossene Blockränder
   Eine Reihe aneinander gebaute Häuser, welche einen einzelnen Innenhof komplett umschliessen.
5) Offene Blockränder
   Eine Reihe aneinander gebaute Häuser, welche einen Bereich teilweise umschliessen.
6) Komplexe Gebäude und Gruppen
   Einzelgebäude mit komplexer Struktur (grosse und unregelmässige Grundrisse, viele Eingänge) und komplexe Gruppen aneinander gebauter Gebäude.

Diese Typologie deckt sich weitgehend mit ähnlichen Typologien, die von Dencity verwendet werden (vergleiche Huber und Walczak (2016b) und Walczak (2017)). Die Kriterien für die Einteilung sowie die Methodik zur Klassifizierung wurden aber unabhängig erarbeitet und unterscheiden sich entsprechend. Auf Basis der Gebäudegrundrisse aus der amtlichen Vermessung und weiterer Daten zu den Gebäudeadressen wird für die aktuelle Bebauung eine Klassifizierung gemäss der beschriebenen Gebäudetypologie vorgenommen. Ergänzend wird für die Beurteilung der Struktur der Bebauung die Gebäudehöhe, Geschosszahl und Geschossfläche aufgezeigt. Zwar wären Teile dieser Daten im Gebäude- und Wohnungsregister erfasst, öffentlich sind diese aber nur in aggregierter Form als Teil der Gebäude- und Wohnungsstatistik (BFS 2016a) verfügbar. Zudem sind in diesem Register aktuell nur Gebäude mit Wohnnutzung zwingend erfasst. Deshalb wird für die Analyse eine Abschätzung dieser Grössen auf Basis der swisBUILDINGS3D 2.0 Daten der swisstopo vorgenommen. Aus den 3D Modellen wird die Gebäudehöhe und der Grundriss ermittelt. Auf Basis der Gebäudehöhe wird die Anzahl Geschosse abgeschätzt, welche zusammen mit der Grundfläche wiederum für eine Schätzung der Geschossfläche verwendet wird. Diese drei Grössen werden als Durchschnittswerte pro Bauzone aggregiert dargestellt. Zusätzlich wird deren Verteilung im Projektgebiet und dem gesamten Gemeindegebiet als Histogramme aufgezeigt, um einen Vergleich zwischen der Struktur im Projektgebiet und dem umgebenden Gebiet zu ermöglichen. Die Verteilungen werden unterteilt nach harmonisierten Bauzonen, um Unterschieden in der Bebauungsstruktur Rechnung zu tragen, welche sich aus verschiedenen Nutzungen ergeben.

3.3.3 Bauliche Dichte

Zusätzliche Hinweise zur Struktur der Bebauung ergeben sich auch aus den Kenngrössen zur baulichen Dichte. Hier kann zwischen Elementen unterscheiden werden, welche bezogen auf die Parzelle oder bezogen auf ein Hektarraster ausgewertet werden. Bezogen auf die Parzelle werden die Baumassenziffer, die Geschossflächenziffer und die Überbauungsziffer abgeschätzt, welche direkte Kennzahlen für die bauliche Dichte darstellen (Grams 2015, s.18). Diese drei Kennzahlen korrespondieren mit drei Nutzungsziffern, welche im Change Model als Option zur Beschränkung der Ausnützung zur Verfügung stehen. Deren Berechnung basiert auf den Definitionen in der Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe (IVHB 2005b), weicht aber an einigen Punkten leicht von diesen ab. Die Baumassenziffer als Mass für die Volumendichte ist das Verhältnis der oberirdischen Baumasse zur anrechenbaren Grundstücksfläche (IVHB 2005b). Die Baumasse wird dabei durch Multiplikation der Grundrissfläche und Gebäudehöhe berechnet, welche aus den swissBUILDINGS3D Daten abgeleitet wurden. Das Bauvolumen wird also vereinfacht als «kubisches» Volumen berechnet, welches im Vergleich zur IVHB Definition keine Rücksicht auf vor- oder rückspringende Gebäudeteile, (halb-)offene Gebäudeteile, Dachschrägen oder dem Terrainverlauf nimmt. Die anrechenbare Grundstücksfläche wird vereinfacht als Fläche der Parzelle innerhalb der Bauzone betrachtet und Erschliessungsflächen werden nicht abgezogen. Diese Grösse wird durch Verschnitt der Parzellen mit den harmonisierten Bauzonen bestimmt. Für Gebäude ausserhalb von Bauzonen werden keine Werte berechnet. Die Geschossflächenziffer als Mass für die Ausnützung der Grundstücksfläche wird als Verhältnis der gesamten Geschossflächen zur anrechenbaren Grundstücksfläche berechnet (IVHB 2005b). Für die Geschossfläche werden wiederum die aus den swissBUILDINGS3D Daten abgeleiteten Werte verwendet. Aufgrund der Natur der swissBUILDINGS3D Daten werden im Vergleich zur IVHB Definition nur oberirdische Geschossflächen mitberücksichtigt und unterirdische Geschosse bleiben aussen vor. Die anrechenbare Grundstücksfläche entspricht dem analogen Wert bei der Baumassenziffer. Die Überbauungsziffer ist schliesslich eine Flächenanteilsziffer, die aus dem Verhältnis der anrechenbaren Gebäudefläche zur anrechenbaren Grundstücksfläche berechnet wird (IVHB 2005b). Die Berechnung der anrechenbaren Grundstücksfläche erfolgt wie bei den anderen zwei Kennziffern. Für die anrechenbare Gebäudefläche wird die Fläche des aus den swissBUILDINGS3D Daten abgeleiteten Grundrisses verwendet. Im Vergleich zur IVHB werden vor- und rückspringende oder halboffene, anteilig anrechenbare Gebäudeteile nicht speziell behandelt.

Zwei weitere Masse für die bauliche Dichte werden nicht in Bezug auf die Parzelle, sondern aggregiert auf ein Hektarraster berechnet. Dieses Raster mit Referenz auf den Schweizer Bezugsrahmen LV03 nimmt zwar keine Rücksicht auf administrative (z.B. Bauzonen, Gemeindegrenzen) oder gewachsene (z.B. Quartiere) Gebietseinheiten, hat aber den Vorteil einer verallgemeinerten, einfach handhabbaren Struktur. Zudem sind für dieses Raster verschiedene Daten des Bundesamtes für Statistik verfügbar, welche bei den weiteren Dichtekennzahlen eine Rolle spielen. Als direktes Mass für die bauliche Dichte wird das überbaute Volumen pro Hektare ausgewiesen. Die Berechnung des Volumens ist dabei analog zur Berechnung der Baumassenziffer. Unterschied ist hingegen die Bezugsfläche und die Tatsache, dass die Auswertung auf die Gesamtfläche unabhängig von Bauzonen erfolgt. Entsprechend werden hier Gebäude ausserhalb der Bauzonen mitberücksichtigt. Als indirektes Mass für die bauliche Dichte wird der Anteil der Freifläche pro Hektare ausgewertet. Als Freifläche werden dabei nicht überbaute Flächen gemäss der Bodenbedeckung der Amtlichen Vermessung angesehen. Das Verhältnis zwischen bebauter und nicht bebauter Fläche trägt zur Qualität der Dichte bei. Durch Aneignung und Nutzung von Freiräumen entsteht ein Teil der (urbanen) Qualität (Angélil *et al.* 2016). Teil dieser Nutzung sind auch soziale Kontakte, welche zur sozialen Dichte beitragen (Spiegel 2000). Die Art der Freifläche beeinflusst zudem deren Potential zur Aneignung sowie deren Wahrnehmung (Angélil *et al.* 2016). Öffentliche Parkanlagen und Plätze können z.B. hohes Potential für Aneignung und verschiedene Nutzungen bergen, sofern deren Gestaltung und regulatorische Dichte diese zulassen (Angélil *et al.* 2016). Ausserdem kann der Anteil an Frei- und Grünflächen einen grossen Einfluss auf die wahrgenommene Qualität, die Regulierung des lokalen Mikroklimas und die ökologische Qualität haben (Wissen Hayek *et al.* 2010).

3.3.4 Weitere Dichtefaktoren

Neben der baulichen Dichte werden zusätzliche Dichtekennzahlen ausgewertet, welche sich auf den Grad und die Art der Nutzung des Projektgebiets beziehen. Die Raumnutzerdichte entspricht der An-

zahl Personen (zusammengesetzt aus Einwohnern und Beschäftigten) pro Flächeneinheit und gibt Aufschluss über die Intensität der Nutzung (Grams 2015, s.18 und 83). Im Unterschied zur Nutzungsdichte ist die Raumnutzerdichte unabhängig von der Bauzone beziehungsweise deren Fläche (Grams 2015, s.18). Die Raumnutzerdichte wird auf Basis von Angaben aus der Statistik der Bevölkerung und der Haushalte (STATPOP) (BFS 2016b) sowie der Statistik der Unternehmensstruktur (STATENT) (BFS 2017c) ermittelt. Die Auswertung findet in einem Hektarraster statt, welches der feinsten Auflösung entspricht, in der die Statistikdaten öffentlich verfügbar sind. Die Einwohnerdichte und Beschäftigtendichte als Komponenten der Raumnutzerdichte werden zusätzlich separat ausgewertet, da diese im Hinblick auf Städtebau, Stadtgeographie und Stadtsoziologie eine wichtige Rolle spielen (Spiegel 2000). Die Beschäftigtendichte gibt Aufschluss über die Intensität der wirtschaftlichen Nutzung. Deren Verteilung auf die drei Wirtschaftssektoren gibt Hinweise auf die Art der Nutzung. Darüber hinaus ist die Beschäftigtendichte von Belang, da die Beschäftigten als erwachsene Personen an verschiedenen lokalen Interaktionsprozessen beteiligt sind (z.B. Erbringung und Nutzung von Leistungen in den Bereichen Wirtschaft, Dienstleistung und Kultur) und so zur sozialen Dichte oder Interaktionsdichte (d.h. der Anzahl Interaktionen innerhalb einer bestimmten Bevölkerungsgruppe, unabhängig von einer Raumabgrenzung) beitragen (Spiegel 2000; Häussermann 2007). Zudem trägt die Bevölkerungsdichte, deren Nutzung des Raumes sich vor allem im privaten Bereich abspielt, zur sozialen Dichte bei (Spiegel 2000). Hier gibt die Verteilung der Einwohner nach Altersklassen zusätzlich Hinweise auf die Demographie der Bevölkerung. Wichtig ist von Belang, weil eine hohe bauliche Dichte oder eine hohe Einwohnerdichte ohne Durchmischung zu monotonen Orten führen und Potential für Nutzungskonflikte sowie negative Auswirkungen auf der sozialen Ebene bergen kann (Spiegel 2000; Angélil *et al.* 2016).

Im Zusammenhang mit der Bevölkerungsdichte und der baulichen Dichte steht die Wohnungsdichte, welche die Anzahl der verfügbaren Wohneinheiten pro Hektare aufzeigt (Grams 2015, s.18). Die Verteilung der Wohnungen nach Anzahl Zimmer gibt Hinweise auf die Wohnbaustruktur. Diese Grössen können den Daten der Gebäude- und Wohnungsstatistik (BFS 2016a) entnommen werden, die ebenfalls als Hektarraster verfügbar sind. Ergänzend gibt die Haushaltsdichte mit der Anzahl Haushalte pro Hektare sowie deren Verteilung nach Haushaltsgrösse Hinweise zur Anzahl der tatsächlichen Wohnverhältnisse sowie deren demographischer Struktur. Zusammen können Wohnungsdichte und Haushaltsdichte als Mass für die Zahl der Wohn- und Lebensverhältnisse dienen (Spiegel 2000). Aus dem Verhältnis der verfügbaren Wohnfläche und der Anzahl Bewohner lässt sich grob die durchschnittliche Wohnfläche pro Bewohner abschätzen, welche der Belegungsdichte der Wohnungen und indirekt mit der baulichen Dichte in Verbindung steht.

Als zusätzliche Ergänzung zu den restlichen Dichtemassen wird die Erschliessungsgüte des Öffentlichen Verkehrs (ÖV) nach der Methodik des Bundesamtes für Raumentwicklung ausgewiesen (ARE 2018b). Hierbei werden Zonen mit vier verschiedenen Klassen der Erschliessungsgüte A bis D gebildet. Die Güte der Erschliessung hängt von der Distanz zur nächsten Haltestelle sowie der Frequenz und Art deren Bedienung an. Die resultierenden Zonen werden vom ARE als Vektordaten x§zur Verfügung gestellt. Die ÖV-Erschliessungsgüte steht indirekt mit anderen Dichtefaktoren in Zusammenhang. So sind zentrumsnahe Gebiete meist besser erschlossen (d.h. sie weisen eine höhere ÖV-Erschliessungsgüte auf) und weisen eine höhere bauliche Dichte sowie eine höhere Raumnutzerdichte auf. Für die Verdichtungsdiskussion kann die Qualität der ÖV-Erschliessung ein Faktor für die Attraktivität eines Standortes und ein Anreiz zur höheren Verdichtung sein. Umgekehrt können sich aus der Betrachtung der Dichten im Verhältnis zur Erschliessung Hinweise auf Defizite und Ausbaubedarf beim ÖV ergeben

3.4 Beurteilung der Ist-Situation anhand von Dichtekennzahlen

3.4.1 Evaluation Model

Im dritten Schritt wird anhand so genannter Evaluation Models der aktuelle Zustand im Projektgebiet beurteilt (Steinitz 2012). Es soll die Frage beurteilt werden, ob das Projektgebiet aktuell gut funktioniert, wobei diese Beurteilung von den Erwartungen sowie dem kulturellen Wissen der Anspruchsgruppen und Entscheidungsträger abhängt (Steinitz 2012; Moura 2015). Die Kriterien für die Beurteilung hängen zudem vom konkreten Projekt, dessen Fragestellungen und den Entscheidungsträgern ab. Zu den möglichen Kriterien gehören ökologische, ökonomische, soziale und politische Faktoren (z.B. Entwicklungen in der Nähe von Naturschutzgebieten, Ansiedlung von Industrie, genossenschaftlicher Wohnungsbau), Veränderungen in diesen Bereichen (z.B. abnehmende Bevölkerungszahl, Ladensterben,

Gentrifizierungsprozesse), aber auch die Wahrnehmung und Attraktivität des Projektgebiets für verschiedene Anspruchsgruppen (Steinitz 2012). Im Rahmen der Arbeit wurden die Ergebnisse des Representation und des Process Models so aufbereitet, dass diese für die Unterstützung der Entscheidungsträger genutzt werden können. Dies insbesondere durch die Bereitstellung möglichst einfach verständlicher zweidimensionaler oder dreidimensionaler Visualisierungen der Situation im Projektgebiet, welche das Verständnis der Ausgangssituation für die Planung erleichtern, zur Bildung einer gemeinsamen Verständnisbasis beitragen und den Prozess der Bewertung sowie der Entscheidungsfindung unterstützen können (Walz *et al.* 2008; Van Wezemael *et al.* 2014). Der konkrete Entscheidungsprozess wird dabei offengelassen und hängt von der jeweiligen Projektorganisation ab. Im Rahmen der Arbeit wurden diese Hilfsmittel zur Orientierung im Projektgebiet und als Grundlage für die Entwicklung der Szenarien anhand des Change Models genutzt.

3.4.2 Hilfsmittel zur Entscheidungsunterstützung

Einerseits wurde für die Entscheidungsunterstützung ein aufbereiteter Bericht mit den Resultaten aus den Representation und Process Models erarbeitet (Abbildung 6 sowie die Abbildung 35 bis Abbildung 38 zeigen Auszüge). Dieser ist für die Abgabe in gedruckter Form auf Papier oder in elektronischer Form als PDF gedacht. Der Bericht ist als QGIS Layout umgesetzt und integriert neben Karten aus QGIS extern erzeugte Grafiken. Während eine initiale Version des QGIS Projekts und des Reportlayouts automatisch generiert wird, kann abhängig vom Projekt eine Anpassung im Hinblick auf die spezifische Situation nötig sein (z.B. Weglassen bestimmter Inhalte, Anpassung von Skalen in Legenden). Das QGIS Projekt definiert Layer mit den Elementen des Representation Models und mit den Ergebnissen des Process Models. Entsprechend kann dieses Projekt auch für die interaktive Präsentation und Analyse des Projektgebiets genutzt werden. Für die interaktive Betrachtung steht zudem die Möglichkeit der dreidimensionalen Visualisierung der Ist-Situation im Projektgebiet mit den Parzellen des Projektperimeters, bestehenden Gebäuden, Strassen und Terrain zur Verfügung (siehe Abbildung 31). Zudem bestehen eingeschränkte Möglichkeiten, interaktiv verschiedene Dichtefaktoren zur Ist-Situation in als dreidimensionale Visualisierung mit der Situation im Projektgebiet zu überlagern (siehe Abbildung 33). Diese dreidimensionalen Visualisierungen zur Ist-Situation sind als zusätzliche Option im parametrischen Modell für die Umsetzung des Change Models implementiert. Dieses in Rhino Grasshopper implementierte Modell setzt zudem die Generierung des QGIS Projektes und der benötigten, externen Grafiken für den Bericht um (vergleiche Architekturübersicht in Kapitel 4.1). Man beachte, dass für die Vereinfachung der Umsetzung die Generierung des QGIS Projektes für das Evaluation Model und das Decision Model kombiniert wurde. Das Projekt umfasst deshalb neben den Layern für die Ist-Situation immer auch Layer für ein Szenario.

Ein erster Teil des Berichts konzentriert sich auf die Darstellung der Ist-Situation entsprechend dem Representation Model. Eine Karte mit der Parzellierung, dem Projektperimeter und den Grundrissen des aktuellen Gebäudebestandes gibt eine Übersicht zum Projektgebiet. Diese wird im Berichtslayout durch Exporte der 3D Ansicht mit Terrain, Gebäuden und Parzellen des Projektgebietes ergänzt. Eine weitere Karte zeigt die Bodennutzung gemäss Bodenbedeckung der Amtlichen Vermessung. Zwei weitere Karten zeigen die baurechtliche Nutzung anhand der harmonisierten Bauzonen und der Bauzonen des Übersichtszonenplans des Kantons Bern. Für die Darstellung der Harmonisierten Bauzonen wurden die Vorgaben des zugrundeliegenden Minimalen Modells Nutzungsplanung übernommen (ARE 2011). Für die Bodenbedeckung und den Übersichtszonenplan wurden jeweils das Darstellungsmodell des Kantons Bern, welches als Styling in ArcGIS vorlag, in QGIS nachgebildet.

Der restliche Bericht setzt sich aus den Elementen des Process Models zusammen. Die Aufbereitung dieser Elemente zu Karten und Visualisierungen nimmt keine direkte Bewertung der jeweiligen Messgrössen vor, versucht aber den Entscheidungsträger durch die Verwendung geeigneter Skalen und Symbolisierungen bei diesem Prozess zu unterstützen. Wenn möglich wurde zudem auf gebräuchliche Darstellungsmodelle mit geeigneten Skalen abgestützt, welche z.B. auf dem Geoportal des Bundes und von Kantonen oder beim Bundesamt für Statistik in Gebrauch sind. In einem ersten Teil werden die geschätzte durchschnittliche Gebäudehöhe, Geschosszahl und Geschossfläche aggregiert auf die harmonisierten Bauzonen als Karten visualisiert. Diese Karten werden jeweils um eine Serie von Histogrammen ergänzt, welche für das Projektgebiet sowie für das gesamte Gemeindegebiet die Verteilung der jeweiligen Grössen aufzeigt (vergleiche Abbildung 5). Unterteilt nach den Klassen der

harmonisierten Bauzonen ist pro Klasse jeweils ein separates Histogramm vorhanden, dessen Visualisierung sich farblich an der Visualisierung der Bauzone in der Übersichtskarte des Representation Models anlehnt. In einem nächsten Teil folgen die Visualisierung der geschätzten Überbauungsziffer, Baumassenziffer und Geschossflächenziffer bezogen auf die Parzelle als Karten. Begleitet werden diese ebenfalls jeweils von einer Serie Histogramme zu den Verteilungen im Projekt- und Gemeindegebiet. Die Histogramme sind wiederum nach harmonisierten Bauzonen unterteilt, wofür die Parzellen mit den Bauzonen verknüpft werden. Als letzte Grösse zur aktuellen Bebauung enthält der Bericht eine Karte zur Gebäudetypologie.

Quelle: eigene Darstellung

Abbildung 5: Visualisierung der Überbauungsziffer in Langenthal als Karte (o.) und Hitogramme (u.)

Bei den weiteren Elementen wird zunächst die ÖV-Erschliessungsgüte dargestellt, wobei das Darstellungsmodell aus dem Geoportal des Bundes übernommen wurde (ARE 2018b). Die restlichen Visualisierungen beziehen sich auf das Hektarraster, welches für das Projektgebiet und die unmittelbare Umgebung angezeigt wird. Diese Karten umfassen die bauliche Dichte und den Freiflächenanteil auf Seiten der baulichen Dichtefaktoren sowie die Raumnutzerdichte, Einwohnerdichte, Beschäftigtendichte, Wohnungsdichte, Haushaltsdichte und durchschnittliche Wohnfläche pro Einwohner auf Seiten der ergänzenden Dichtefaktoren. Die Verteilungen nach Kategorien für die Einwohnerdichte, Beschäftigtendichte, Wohnungsdichte und Haushaltsdichte werden als separate Karten der jeweiligen Dichten im Hektarraster mit überlagertem Kartodiagramm visualisiert, welches die Verteilung als Kuchendiagramm anzeigt. Dieses Vorgehen orientiert sich an der Aufbereitung von Daten zur Siedlungsentwicklung nach innen (SEin) des Kantons Bern, wo ebenfalls Verteilungen in dieser Form als Kontextinformation verfügbar sind (AGR 2017). Die verwendeten Kategorien für die Verteilungen richten sich allerdings nach Einteilungen, welche vom Bundesamt für Statistik definiert wurden.

3.5 Definition von Szenarien

3.5.1 Change Model

Im vierten Schritt des Prozesses wird die Frage gestellt, wie das Projektgebiet verändert werden kann und welche Policy Instrumente oder Eingriffe hierfür wann und wo angewendet werden sollen (Steinitz 2012). Im Rahmen des Geodesign Prozesses entwickelte Change Models dienen der Beantwortung dieser Fragen (Steinitz 2012). Die von den Change Models generierten Daten werden für die Repräsentation des künftigen Soll-Zustandes genutzt und erlauben den Vergleich verschiedener Szenarien untereinander (Steinitz 2012). Das in der Arbeit entwickelte Vorgehensmodell zielt im Unterschied zu einem klassischen Geodesign Prozess nicht auf den Entwurf einer konkreten Planung ab, welche tatsächlich so umgesetzt werden soll. Vielmehr soll die Erstellung verschiedener Varianten für eine verdichtete Bauweise im Projektgebiet, die Vermittlung der Auswirkungen dieser Varianten und deren Vergleich erleichtert werden. Hierfür wird ein parametrisches Modell verwendet, welches verschiedene bau- und planungsrechtliche Parameter integriert und auf deren Basis dreidimensionale Visualisierungen für eine Bebauung im Projektgebiet entsprechend der festgelegten Planungsparameter generiert. Modelle dieser Art helfen, die Auswirkung von Anpassungen der sonst nur in abstrakter Textform vorhandenen Parameter für Anspruchsgruppen und Entscheidungsträger zugänglich und einfach fassbar zu machen (Walz *et al.* 2008; Wissen Hayek *et al.* 2013; Moura 2015). Durch Einstellung unterschiedlicher Parameter erlaubt das Modell die Betrachtung verschiedener Szenarien für eine künftige verdichtete Bauweise, wobei jeweils die höchstmögliche Dichte für das betrachtet Szenario aufgezeigt wird. Die vom parametrischen Modell generierten Resultate dienen in den nachfolgenden Schritten als Grundlage für die Analyse der Auswirkungen der verschiedenen Verdichtungsszenarien und für deren Vergleich untereinander.

3.5.2 Parametrisches Modell zur Generierung von Verdichtungsszenarien

Infolge der Zuständigkeit der Gemeinden für die Nutzungsplanung und deren Möglichkeiten bei der Definition von Bauvorschriften ergibt sich eine sehr heterogene Landschaft gesetzlicher Regularien. Das parametrische Modell für die Umsetzung des Change Models ist deshalb nicht auf eine bestimmte Gemeinde und deren Vorschriften zugeschnitten, da eine zu spezifische Umsetzung die Übertragbarkeit einschränken würde. Um letztere zu vereinfachen integriert das Modell vielmehr eine Auswahl breit abgestützter und gut übertragbarer Planungsparameter in weitgehend generischer Form. Die Auswahl und Umsetzung der abgebildeten Parameter orientiert sich an der Interkantonalen Vereinbarung über die Harmonisierung der Baubegriffe (IVHB 2005a). Die teilnehmenden Kantone dieses Konkordats sind zur Übernahme der im Anhang der Vereinbarung definierten Begriffe und Messweisen (IVHB 2005b) verpflichtet. Diese umfassen 30 formelle Baubegriffe (Definitionen) wie Höhen, Abstände, Geschossigkeit, welche in allen Kantonen gleich verstanden werden sollen (IVHB 2018). Stand Anfang 2018 sind bereits 18 Kantone dem Konkordat beigetreten, darunter auch der Kanton Bern, in dem die Testgebiete für die Arbeit liegen (IVHB 2018). Mit der Verordnung über die Begriffe und Messweisen im Bauwesen (BMBV 2011) setzt Bern die IVHB in kantonales Recht um.

Das parametrische Modell ist in Grasshopper, einer Erweiterung für Parametrisches Design des CAD und 3D Tools Rhino, umgesetzt. Die Eingabe der meisten Parameter erfolgt interaktiv über Parameterkontrollen innerhalb der Rhino Grasshopper Benutzeroberfläche. Die Auswirkungen geänderter Parameter werden vom Modell direkt umgesetzt und in einer 3D Ansicht visualisiert, wodurch die räumlichen

Auswirkungen der Änderungen an den Parametern unmittelbar sichtbar werden. Die Entwicklung von Szenarien beziehungsweise der resultierenden Varianten für Verdichtung kann abhängig vom jeweiligen Projekt entweder durch die Planer selbst oder im interaktiven Dialog mit Anspruchsgruppen (z.B. im Rahmen von Workshops) erfolgen. Wenn in einem Szenario der gewünschte Zustand konfiguriert ist, können einerseits die eingestellten Parameterwerte gesichert werden, damit diese später wieder abgerufen werden können. Andererseits können die vom Modell generierten Resultate abgespeichert werden, so dass diese anschliessend mittels des Impact Models analysiert werden können.

3.5.3 Modellparameter

3.5.3.1 Übersicht

Die nachfolgenden Unterkapitel erläutern die vom Modell umgesetzten Parameter. Diese decken zentrale Begriffe der IVHB ab, welche für die Generierung als relevant betrachtet werden. Weitere Parameter orientieren sich an den Elementen des Process und Evaluation Models, um Inputs für die Analyse und Bewertung der Szenarien im Rahmen der Impact und Decision Models bereitzustellen. Darüber hinaus bietet das Modell einige Konfigurationsmöglichkeiten zur Beeinflussung der generierten Gebäude, welche teils auf gestalterische Überlegungen (z.B. Typ des Dachaufbaus) oder aber auf die Generierung der Gebäudegrundrisse (z.B. Zusammenfassung von Parzellen für Mehrfamilienhäuser) auswirken. Einige weitere Parameter wie z.B. Pfadangaben oder Farbwerte, welche für die Konfiguration des Modells und dessen Ausführung nötig sind, werden nicht speziell behandelt.

3.5.3.2 Projekt

Aufgrund seiner generischen Auslegung kann das gleiche parametrische Modell für verschiedene Projekte eingesetzt werden. Deshalb kann das Projekt im Modell parametrisiert werden. Technisch gesehen handelt es sich bei einem Projekt um einen Eintrag in der Datenbank, welcher Einträge für die Parzellen des Projektperimeters und die Szenarien zugeordnet sind. Diese Daten werden nicht direkt im parametrischen Modell definiert. Vielmehr müssen diese im Rahmen des ersten Prozessschrittes als Teil des Representation Models entsprechend aufbereitet werden. Im Modell beeinflusst das gewählte Projekt die verfügbare Auswahl von Szenarien und die Parzellen, die als Input aus der Datenbank geladen werden.

3.5.3.3 Szenario

Der Szenario Parameter legt das Set von baulichen Parametern fest, welche auf die Parzellen eines Projektes angewendet werden. Damit definiert jedes Szenario eine spezifische Variante für das verdichtete Bauen im Projektperimeter. Technisch gesehen ist ein Szenario ein Datenbankeintrag, mit dem weitere Einträge für die Speicherung der baulichen Parameterwerte assoziiert sind. Abhängig vom gewählten Projekt können im Modell die in der Datenbank verfügbaren Szenarien ausgewählt werden. Neue Einträge müssen aber ausserhalb des Modells direkt in der Datenbank angelegt werden. Die assoziierten Parameterwerte können anschliessend innerhalb des Modells direkt abgespeichert sowie geladen und auf die Parameterkontrollen angewendet werden. Man beachte, dass beim Laden eines Szenarios, wenn keine Parameterwerte gespeichert sind, Standardwerte eingesetzt werden, die im Modell hinterlegt sind. Die vom Modell generierten Resultate, welche in die Datenbank zurückgespeichert werden, sind ebenfalls mit dem jeweils parametrisierten Szenario assoziiert.

3.5.3.4 Parzelle

Parzellen können als indirekter Parameter des Modells angesehen werden. Diese werden abhängig vom gewählten Projekt geladen. Deren Geometrien legen den Umfang des Projektperimeters fest und dienen als Ausgangspunkt für die Ableitung der bebaubaren Bereiche (und somit der Gebäudegrundrisse). Im Modell wird vereinfachend angenommen, dass jede Parzelle genau einer Planungszone angehört. Zudem weist jede Parzelle eine spezifische anrechenbare Grundstücksfläche auf. Diese beiden Grössen sind bei der Aufbereitung der Parzellen festzulegen und auf dem Eintrag der Parzelle in der Datenbank zu hinterlegen.

3.5.3.5 Anrechenbare Grundstücksfläche

Die anrechenbare Grundstücksfläche ist ebenfalls ein indirekter Parameter, welcher für jede Parzelle festzulegen ist. Dieser Wert ist im ersten Prozessschritt zusammen mit der Aufbereitung der Parzellen des Projektperimeters festzulegen und in der Datenbank abzulegen. Die anrechenbare Grundstücksfläche entspricht der Fläche, die bebaut werden darf und muss deshalb kleiner oder gleich der gesamten Parzellenfläche sein. Werte kleiner der Parzellenfläche entsprechen der Situation, wenn eine

Parzelle nur teilweise in einer Bauzone liegt. Im Modell wird aktuell die gesamte Parzellenfläche als Ausgangspunkt für die Ableitung des Baubereichs verwendet. Ist die anrechenbare Grundstücksfläche kleiner als die gesamte Grundstücksfläche, wird der Baubereich aufgrund der Nutzungsziffern entsprechend kleiner dimensioniert.

Im Rahmen des Process Models werden bei der Bestimmung der anrechenbaren Grundstücksfläche die Erschliessungsflächen nicht berücksichtigt. Für das Impact Model wird im Projektgebiet der auf der Parzelle hinterlegte Wert für die anrechenbare Grundstücksfläche verwendet. Wenn hier Parzellenflächen nicht angerechnet werden sollen, weil sie z.B. zur Basis- beziehungsweise Detailerschliessung gehören oder Teile der Parzelle ausserhalb der Bauzonen liegen, kann dies im Vorfeld bei der Bestimmung der anrechenbaren Grundstücksfläche berücksichtigt werden.

3.5.3.6 Modellzone

Im Modell können verschiedene Modellzonen definiert werden, für die jeweils ein eigener Satz von unabhängigen baulichen Parametern gilt. Diese Modellzonen entsprechen nicht direkt Bauzonen, sondern sind nur eine vereinfachte Annäherung, die primär der Organisation verschiedener Parametersätze im Projektgebiet dient. Gleichfalls kennt das Modell keine expliziten Konzepte für Überbauungsordnungen oder Zonen mit Planungspflicht. Diese können ebenfalls nur über die Definition eigener Modellzonen angenähert werden.

Für jede Parzelle wird angenommen, dass diese nur genau einer Modellzone angehört, während sich in der Realität eine Parzelle mit verschiedenen Bauzonen überschneiden kann. Entsprechend ist im Modell pro Parzelle auch nur eine Nutzungsart vorgesehen (Wohnen, Arbeit oder Mischnutzung). Zudem haben die Modellzonen keine räumliche Ausdehnung, welche die bebaubaren Flächen einschränkt. Liegt ein Teil einer Parzelle in der Realität ausserhalb einer Bauzone, wird dies im Modell zuerst nur über die tiefere anrechenbare Grundstücksfläche abgebildet. Bei der Ableitung der Baufläche wird diese zwar anteilsmässig kleiner, kann sich aber mit dem eigentlich nicht bebaubaren Bereich überschneiden. Sollen Parzellenteile im Modell explizit nicht bebaubar sein, müssen diese entweder mittels Abstandslinien abgetrennt oder bei der Datenaufbereitung von der Geometrie der Parzelle abgezogen werden.

3.5.3.7 Kleiner und grosser Grenzabstand

Pro Modellzone können separat Werte für einen kleinen und grossen Grenzabstand in Metern parametrisiert werden. Der Grenzabstand entspricht dem minimalen Abstand zwischen Parzellengrenze und projizierter Fassadenlinie des generierten Gebäudes. Im Modell wird der grosse Grenzabstand auf den längsten zusammenhängenden Grenzabschnitt angewendet, der eine Ausrichtung nach Osten bis Südwesten aufweist. Dieser Abschnitt wird als besonnte Hauptwohnseite des Gebäudes angesehen. Auf alle restlichen Grenzabschnitte wird der kleine Grenzabstand angewendet.

3.5.3.8 Abstandslinien

Zur Abbildung weiterer Abstandsregelungen können im Modell pro Szenario Abstandslinien definiert werden. Diese sind ausserhalb des Modells als Polyline Objekte in der Datenbank zu erfassen und dem jeweiligen Szenario zuzuordnen. Die Linien dürfen nicht geschlossen sein. Im Modell werden die Abstandslinien mit den Parzellengrenzen verschnitten und die Parzellenbereiche rechts der Linie (vom Beginn der Linie aus in Richtung Ende gesehen) als nicht bebaubar ausgeschieden. Deshalb ist wichtig, dass die Geometrien der Abstandslinien die betreffenden Parzellengrenzen schneiden oder berühren. Auf die ausgeschiedenen Grenzabschnitte werden keine Grenzabstände angewendet, wodurch die Abstandslinien an Stelle des grossen oder kleinen Grenzabstandes treten. Die Abstandslinien haben im Modell keine gestalterische Wirkung (d.h., dass das generierte Gebäude zwingend die Abstandslinie als Teil der projizierten Fassadenlinie einschliesst beziehungsweise an diese heranreicht).

3.5.3.9 Anzahl Geschosse

Dieser Parameter legt die maximale Anzahl Geschosse über dem Boden innerhalb der jeweiligen Modellzone fest. Diese werden immer als Vollgeschosse angesehen. Dach- und Attikageschosse werden vom Modell momentan nicht unterstützt. Für jedes Gebäude wird zwar ein Untergeschoss mit gleicher Fläche und Höhe wie das unterste Vollgeschoss erzeugt, welches aber der garantierten Versenkung ins Terrain dient und nicht gegen das Limit gezählt wird.

Das Modell versucht immer die maximale Anzahl Geschosse auszuschöpfen. Die Anzahl wird reduziert, wenn zusätzlich eine maximale Gesamthöhe für Gebäude definiert ist und das Produkt von Anzahl Geschossen mal Geschosshöhe dieses Limit überschreiten würde.

3.5.3.10 Geschosshöhe

Dieser Parameter legt die Höhe der Geschosse innerhalb der jeweiligen Modellzone fest. Die Geschosshöhe wird auf alle Geschosse gleich angewendet. Eine abweichende Geschosshöhe für das Erdgeschoss, wie sie z.B. für Erdgeschossnutzungen in gewissen Städten zulässig ist, wird vom Modell nicht unterstützt.

3.5.3.11 Gesamthöhe

Dieser Parameter legt die maximal mögliche Höhe eines Gebäudes innerhalb einer Modellzone fest. Das Modell ignoriert diesen Inputparameter, wenn er den Wert 0 aufweist oder kleiner als die Geschosshöhe ist.

3.5.3.12 Art der Ausnützung

Für jede Modellzone kann festgelegt werden, ob eine Nutzungsziffer zur Dimensionierung der Gebäude genutzt werden soll. Nach Abzug der Abstandsbereiche von den Parzellengrenzen und der Anpassung der anrechenbaren Grundstücksfläche durch die Grünflächenziffer wird die Nutzungsziffer für die Dimensionierung der Baufläche und damit der Gebäudegrundfläche herangezogen, wobei Anzahl Geschosse, Geschosshöhe und Maximalhöhe ebenfalls eine Rolle spielen. Dabei betrachtet das Modell nur die Ausnützung über dem Boden. Aktuell werden folgenden Optionen unterstützt:

Tabelle 3: Definition Arten der Ausnützung

Bezeichnung	Definition
Keine	Es wird keine Nutzungsziffer berücksichtigt. Die Dimensionierung der Baufläche hängt damit nur von den Abstandsbereichen sowie der anrechenbaren Grundstücksfläche und das Volumen von der Anzahl Geschosse, der Geschosshöhe sowie der maximalen Gesamthöhe ab.
Geschossflächenziffer (GFZ)	$$Geschossflächenziffer = \frac{Summe\ aller\ Geschossflächen}{antechenbare\ Grundstücksfläche}$$ Die Summe aller Geschossflächen wird als Produkt der Gebäudegrundfläche und Anzahl Geschosse berechnet. Im Unterschied zur IVHB Definition werden keine unterirdischen Geschosse berücksichtigt. Der Baubereich entspricht der nach dem Abzug der Abstandsbereiche verbleibenden Parzellenfläche, wenn die Geschossflächenziffer gemäss Gleichung bereits unterschritten wird. Ansonsten reduziert das Modell die Fläche, bis die Geschossflächenziffer erreicht oder leicht unterschritten wird.
Baumassenziffer (BMZ)	$$Baumasseziffer = \frac{Baumasse\ über\ dem\ Boden}{anrechenbare\ Grundstücksfläche}$$ Die Baumasse über dem Boden wird als Produkt der Gebäudegrundfläche sowie der Anzahl Geschosse und Geschosshöhe berechnet. Die Gebäudegrundfläche beziehungsweise der Baubereich entspricht der nach dem Abzug der Abstandsbereiche verbleibenden Parzellenfläche, wenn die Baumassenziffer gemäss Gleichung bereits unterschritten wird. Ansonsten reduziert das Modell die Fläche, bis die Baumassenziffer erreicht oder leicht unterschritten wird.
Überbauungsziffer (ÜZ)	$$Überbauungsziffer = \frac{anrechenbare\ Gebäudefläche}{anrechenbare\ Grundstücksfläche}$$ Die anrechenbare Gebäudefläche entspricht der Gebäudegrundfläche bzw. dem Baubereich. Der Baubereich entspricht der nach dem Abzug der Abstandsbereiche verbleibenden Parzellenfläche, wenn die Überbauungsziffer gemäss Gleichung bereits unterschritten wird. Ansonsten reduziert das Modell die Fläche, bis die Überbauungsziffer erreicht oder leicht unterschritten wird.

3.5.3.13 Nutzungsziffer

Die Nutzungsziffer legt innerhalb einer Modellzone den zulässigen Verhältniswert fest, wenn die Ausnützung anhand einer Nutzungsziffer bemessen wird. Dieser Input Parameter wird ignoriert, wenn keine Nutzungsziffer verwendet wird. Bei Verwendung einer Überbauungsziffer ist ein Wert von 0 bis 1 zulässig. Werte über 1 werden vom Modell als 1 interpretiert. Bei Verwendung der Geschossflächenziffer oder der Baumassenziffer sind Verhältniswerte grösser 1 zulässig.

3.5.3.14 Grünflächenziffer

Die Grünflächenziffer wirkt wie eine indirekte Nutzungsziffer. Sie legt fest, welcher Anteil der anrechenbaren Grundstücksfläche innerhalb einer bestimmten Modellzone nicht überbaut werden darf und eine

natürliche, bepflanzte Bodenfläche aufweisen muss. Für das Modell wird vereinfachend angenommen, dass die Parzellenfläche ausserhalb der Gebäudegrundrisse nicht überbaut und nicht versiegelt ist. Die Grünflächenziffer wird unabhängig von den anderen Nutzungsziffern behandelt. Ist der Wert 0 für die Grünflächenziffer eingestellt, wird dieser ignoriert. Ansonsten wird vor der Anwendung der Ausnützung für die Berechnung des Baubereichs der Wert der anrechenbaren Grundstücksfläche um den von der Grünflächenziffer festgelegten Anteil reduziert.

3.5.3.15 Gruppierung

Das Modell unterstützt optional die Möglichkeit der Gruppierung von Parzellen, wodurch diese bei den weiteren Schritten wie eine einzige Parzelle behandelt werden. Das heisst, dass z.B. die Abstandsvorschriften nur auf die Aussengrenze der Vereinigung aller Parzellen aber nicht auf die internen Parzellengrenzen angewendet werden und ausserdem die Nutzungsziffer auf die Gruppe als Ganzes angewendet wird. Dies erlaubt die Annäherung von Situationen wie z.B. bei Reiheneinfamilienhäusern auf einer Gruppe schmaler Parzellen, wo durch separate, im Grundbuch festgehaltene Regelungen zu Aspekten wie Näherbaurecht und Abtausch bei Ausnutzungen geschlossene Bebauungen über Parzellengrenzen hinweg ermöglicht werden, welche eigentlich punktuell die Bauvorschriften verletzen.

Unterstützt wird aktuell einerseits die Gruppierung nach dem Typ Reihenhaus, wobei die Dimensionierung des Baubereiches für die Gruppe analog zu einer Einzelparzelle erfolgt. Dies ermöglicht z.B. eben die Generierung geschlossener Reihenhäuser auf einer Gruppe schmaler Parzellen. Andererseits wird die Gruppierung als geschlossener Blockrand unterstützt. Die Dimensionierung des Baubereiches erfolgt dabei nicht durch Reduktion vom Rand her, sondern durch Subtraktion einer Fläche im Inneren des Blocks, welche einen Innenhof erzeugt. Die Definition der Gruppen erfolgt durch den Benutzer in Form eines JSON Fragments, in dem der Gruppierungstyp und die Parzellennummern der beteiligten Parzellen angegeben wird. Voraussetzung für das korrekte Funktionieren der Gruppierung ist, dass die Parzellen aneinandergrenzen und in derselben Modellzone liegen.

3.5.3.16 Dach Typ

Das Modell unterstützt optional die Möglichkeit den Dach Typ für Gebäude innerhalb bestimmter Parzellen explizit festzulegen. Aktuell kann der Typ als Steildach oder Flachdach festgelegt werden. Ähnlich wie bei den Gruppierungen erfolgt die Definition der Dach Typen in Form eines JSON Fragments, in dem pro Dach Typ die Parzellennummern angegeben werden. Wenn kein expliziter Typ festgelegt ist, wird als Standard ein Steildach verwendet. Das Modell legt ein Flachdach als Typ fest, wenn die Fläche des Baubereichs unter 20m^2 oder über 500m^2 liegt. Diese Grenzen sind global konfigurierbar und schliessen unvorteilhafte Dach-Geometrien aus. Zudem verwendet das Modell ein Flachdach, wenn das Produkt aus Geschosszahl und Geschosshöhe plus der Höhe für den Dachstock die zulässige Gesamthöhe in der Zone überschreiten würde.

3.5.3.17 Anteil der Arbeitsnutzung

Dieser Parameter legt fest, welcher Anteil der Geschossflächen in einer Modellzone für Arbeitsnutzungen verwendet wird. Ein Wert von 0 entspricht einer reinen Wohnnutzung. Werte zwischen 0 und 1 entsprechen Mischnutzungen und ein Wert gleich 1 entspricht einer reinen Gewerbe- oder Industrienutzung (Arbeitsnutzung).

3.5.3.18 Flächenbedarf Wohnnutzung pro Einwohner

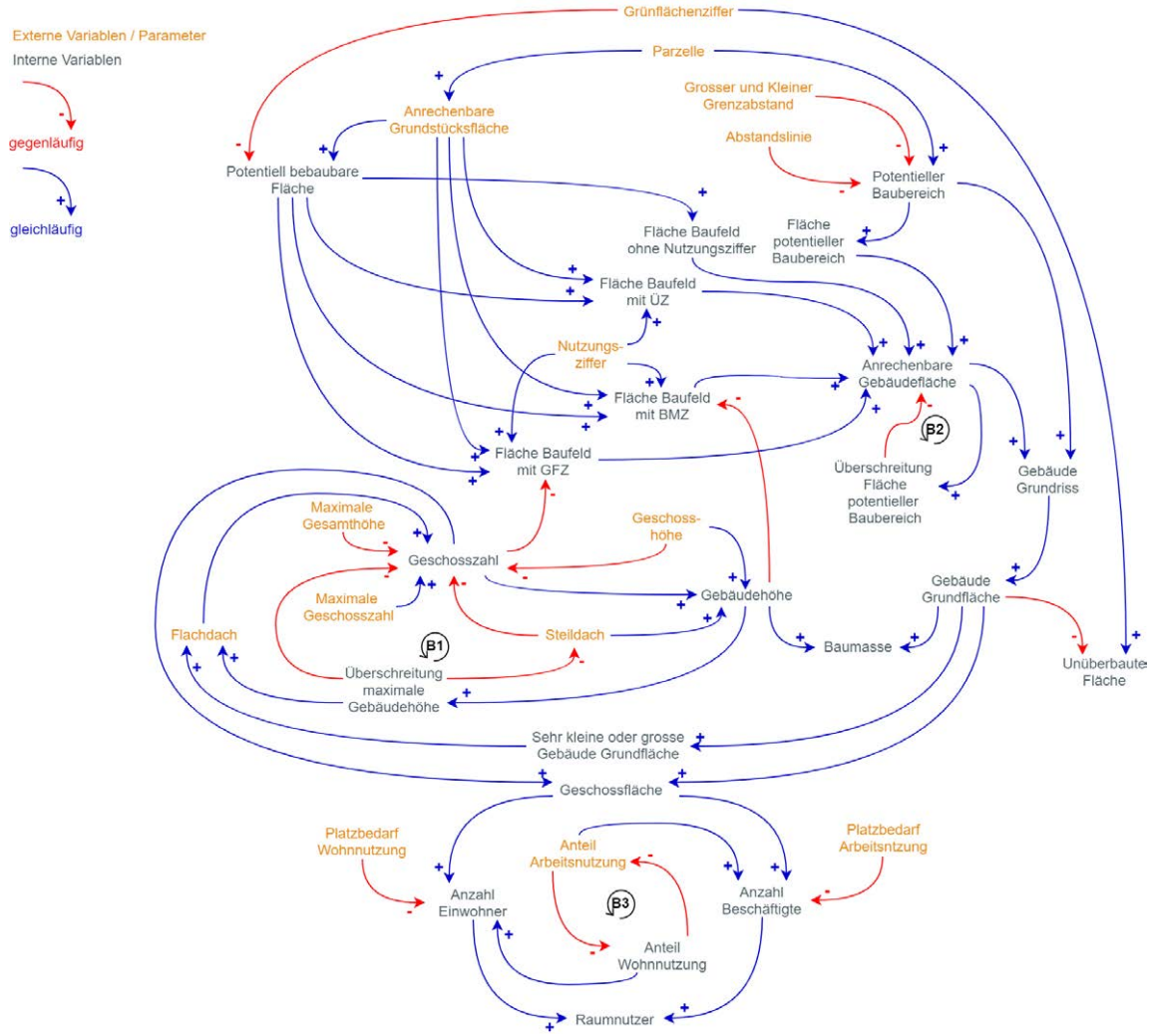
Dieser Parameter legt den angenommenen, durchschnittlichen Wohnflächenbedarf pro Einwohner innerhalb einer Modellzone fest. In Modellzonen mit einem Anteil Wohnnutzung wird dieser Parameterwert zusammen mit der Geschossflächenzahl für die Abschätzung der Anzahl Einwohner und damit im Impact Model schlussendlich für die Abschätzung der Einwohnerdichte genutzt.

3.5.3.19 Flächenbedarf Arbeitsnutzung pro Beschäftigtem

Analog zum Flächenbedarf für Wohnnutzung legt dieser Parameter den angenommenen, durchschnittlichen Flächenbedarf pro Beschäftigtem innerhalb einer Modellzone fest. In Modellzonen mit einem Anteil Arbeitsnutzung wird dieser Parameterwert zusammen mit der Geschossflächenzahl für die Abschätzung der Anzahl Beschäftigten und damit im Impact Model schlussendlich für die Abschätzung der Beschäftigtendichte genutzt.

3.5.4 Modellbeschreibung

Die Abbildung 6 zeigt eine Übersicht der wichtigsten Einflüsse und Zusammenhänge im Impact Model in Form eine Causal Loop Diagramms. Mit **+** markierte Verbindungen bezeichnen gleichläufige und mit **-** markierte Verbindungen bezeichnen gegenläufige Beziehungen.



Quelle: eigene Darstellung

Abbildung 6: Causal Loop Diagram der wichtigsten Zusammenhänge im parametrischen Modell

Da jeweils die höchstmögliche Dichte in einem Szenario aufgezeigt werden soll, versucht das Modell die Ausnützung unter den gegebenen Parametern so weit möglich zu maximieren. Um trotz verdichteter Bauweise den Freiraumanteil möglichst hoch zu halten, wird das Bauen in die Höhe gegenüber dem Bauen in die Fläche priorisiert.

Die Gebäudehöhe nimmt mit der Geschosszahl und -höhe sowie beim Vorhandensein eines Steildaches zu. Aufgrund der Priorität des Bauens in die Höhe, versucht das Modell die maximal mögliche Geschosszahl auszureizen. Durch Geschosshöhe und maximale Gesamthöhe des Gebäudes kann die Geschosszahl aber begrenzt werden. Würde die gesamte Gebäudehöhe bei maximaler Geschosszahl die erlaubte Maximalhöhe überschreiten, versucht das Modell dies auszubalancieren (siehe Loop B1). Sofern nicht explizit ein Steildach konfiguriert ist, verzichtet das Modell zugunsten der Geschosszahl auf das Steildach. Stattdessen wird ein Flachdach verwendet, wodurch sich die Gebäudehöhe reduziert. Erst wenn die maximale Gebäudehöhe immer noch überschritten wird, reduziert das Modell die Geschosszahl auf den unter den gegebenen Parametern höchstmöglichen Wert.

Bei den Flächen ist zunächst die Parzelle der primäre Input. Einerseits wird der potentiell bebaubare Bereich innerhalb der Parzelle durch die Grenzabstände und Abstandslinien begrenzt. Durch deren

Anwendung auf die Parzelle wird diese verkleinert und daraus Form und Fläche des potentiellen Baubereichs abgeleitet. Andererseits beeinflusst die Parzellenfläche direkt die mögliche anrechenbare Grundstücksfläche. Diese entspricht dem potentiell bebaubaren Flächenanteil, der in der Realität der Parzellenfläche innerhalb der Bauzone entspricht. Der potentiell bebaubare Flächenanteil wird allenfalls durch die Grünflächenziffer reduziert. Zusammen mit der Nutzungsziffer dient die potentiell bebaubare Fläche der Bestimmung der anrechenbaren Gebäudefläche. Wird keine Ausnützungsziffer verwendet, entspricht die anrechenbare Gebäudefläche der potentiell bebaubaren Fläche. Bei Verwendung einer Überbauungsziffer entspricht die anrechenbare Gebäudefläche dem Anteil der bebaubaren Fläche gemäss Nutzungsziffer. Wird die Baumassenziffer für die Dimensionierung verwendet, entspricht die anrechenbare Gebäudefläche der potentiell bebaubaren Fläche, welche auf Basis des Verhältnisses zwischen Nutzungsziffer und Gebäudehöhe reduziert wird. Ist eine Geschossflächenziffer definiert, so entspricht die anrechenbare Gebäudefläche der potentiell bebaubaren Fläche, die anhand des Verhältnisses zwischen Nutzungsziffer und Geschosszahl reduziert wird. Überschreitet die anrechenbare Gebäudefläche den Flächeninhalt des potentiellen Baubereiches, so reduziert das Modell die Fläche entsprechend (siehe Loop B2). Der potentielle Baubereich und die anrechenbare Gebäudefläche dienen als Input für die Ableitung des Gebäudegrundrisses. Da das Modell versucht, ansprechende Gebäude zu generieren, kann die Fläche der Gebäudegrundrisse in Grenzfällen minimal kleiner sein, als die berechnete anrechenbare Gebäudefläche. Aus dem Gebäudegrundriss leitet sich die letztendliche Grundfläche des Gebäudes ab. Diese beeinflusst wiederum zusammen mit der Gebäudehöhe das Bauvolumen und die Baumassenziffer. In Kombination mit der Geschosszahl beeinflusst die Grundfläche die Geschossfläche und damit die Geschossflächenziffer. Zudem reduziert die Grundfläche den nicht überbauten Bereich und somit den Freiflächenanteil. Aus ästhetischen Gründen wandelt das Modell bei Gebäuden mit sehr grosser oder sehr kleiner Grundfläche Steildächer (mit Ausnahme von explizit konfigurierten Steildächern) in Flachdächer um, wodurch die Gebäudehöhe und das Bauvolumen reduziert, die Geschosszahl aber nicht beeinflusst wird.

Die Arbeits- und Wohnnutzung im Projektgebiet werden von der verfügbaren Geschossfläche und dem jeweiligen Platzbedarf pro Raumnutzer beeinflusst. Während eine höhere Geschossfläche zu einer Steigerung bei der Anzahl Beschäftigten und Einwohnern führt, reduziert ein höherer Platzbedarf deren Anzahl. Die Verteilung der Raumnutzer auf Einwohner und Beschäftigte wird durch das Nutzungsverhältnis bestimmt, welches durch die Parametrisierung des Anteils Arbeitsnutzung in Prozent der verfügbaren Geschossfläche bestimmt wird. Entsprechend balancieren sich der Anteil der Arbeitsnutzung und der Anteil der Wohnnutzung automatisch aus (siehe Loop B3). Steigende Zahlen bei Einwohnern und Beschäftigten erhöhen die jeweiligen Dichten sowie die Raumnutzerdichte.

3.6 Analyse der räumlichen Auswirkungen der Szenarien

3.6.1 Impact Model

Im fünften Schritt des Prozesses geht es um die Frage, welche Änderungen sich durch die mittels des Change Models entwickelten Szenarien im Projektgebiet ergeben (Steinitz 2012). Impact Models liefern Antworten auf diese Frage (Steinitz 2012). Diese können als Analysen der Situation mittels Process Models unter den geänderten Bedingungen im Projektgebiet angesehen werden (Steinitz 2012). Tabelle 4 zeigt eine Übersicht der Elemente des Impact Models. Dieses orientiert sich am Process Model und gibt anhand verschiedener Dichtekennzahlen eine Übersicht zur Situation im Projektgebiet unter den Bedingungen der im Change Model definierten Szenarien. Einerseits wird auf die resultierende, bauliche Struktur fokussiert. Andererseits werden verschiedene Aspekte der Dichte im Projektgebiet analysiert. Um die Vergleichbarkeit der Analyse der Szenarien mit der Ist-Situation zu gewährleisten, sind die Elemente des Impact Models sowohl in Bezug auf Inhalt als auch Methodik der Analyse auf die Elemente des Process Models abgestimmt. Die Ergebnisse (siehe Kapitel 4.2.4 für Details zu Analyse und Umsetzung) sind Hauptbestandteil des Berichtes zur Analyse der verschiedenen Verdichtungsszenarien, der als Grundlage für die Bewertung der Szenarien und für die Entscheidungsfindung im anschliessenden Schritt dient.

Tabelle 4: Elemente des Impact Models

Element	Definition und Einsatzgebiet	Flächeneinheit	Quantifizierung
Gebäude	Grundrisse sowie dreidimensionale Visualisierung bestehender Gebäude	Parzelle	-
Art der Bebauung	Beurteilung von Gebäudehöhe, Geschosszahl und Geschossfläche des Gebäudebestandes	Bauzone	Durchschnittliche Gebäudehöhe in m
		Bauzone	Durchschnittliche Geschosszahl
		Bauzone	Durchschnittliche Geschossfläche in m² pro Gebäude
Bauliche Dichte	Abschätzung der überbauten Fläche, der Geschossfläche und dem überbauten Volumen pro anrechenbarer Grundstücksfläche	Parzelle	Überbauungsziffer
		Parzelle	Geschossflächenziffer
		Parzelle	Baumassenziffer
	Überbautes Volumen pro Flächeneinheit	Hektare	m³/ha
Freiflächenanteil	Anteil nicht überbauter Fläche pro Flächeneinheit	Hektare	%
Raumnutzerdichte	Anzahl Einwohner und Beschäftigte pro Flächeneinheit	Hektare	Raumnutzer pro ha
Einwohnerdichte	Anzahl Einwohner pro Flächeneinheit	Hektare	Einwohner pro ha
Beschäftigtendichte	Anzahl Beschäftigte pro Flächeneinheit	Hektare	Beschäftigte pro ha
Durchschnittliche Wohnfläche	Durchschnittliche Wohnfläche pro Einwohner	Hektare	m² pro Einwohner

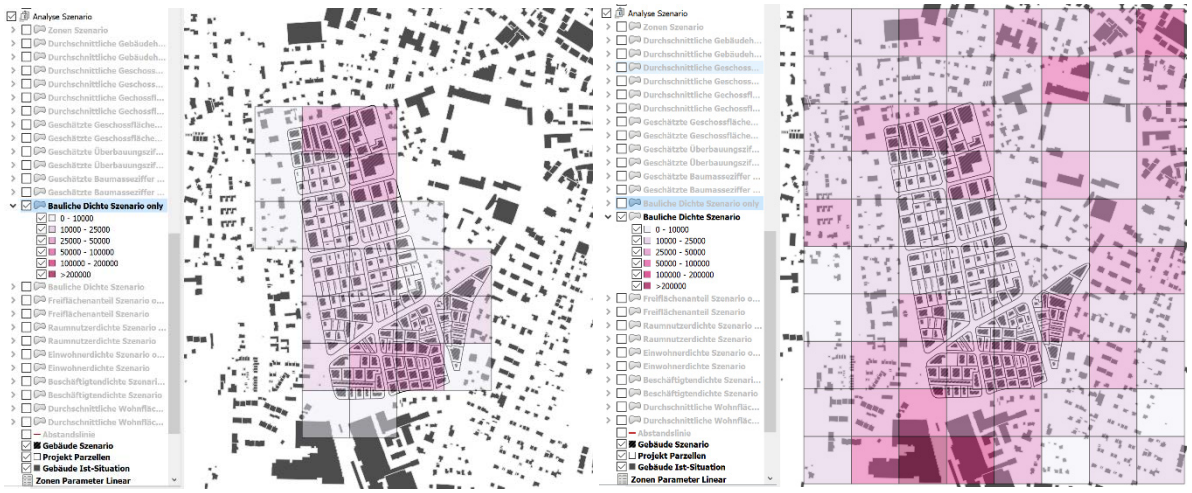
3.6.2 Art der Bebauung und bauliche Dichte

Innerhalb des Projektperimeters werden die Art der Bebauung und die bauliche Dichte jeweils in Bezug auf ein spezifisches Szenario untersucht. Als Input dienen die Parzellen des Projektperimeters und die vom parametrischen Modell generierten Resultate. Letztere umfassen den Grundriss des generierten Gebäudes und einige Kennwerte wie die Anzahl Geschosse, Geschossflächen, Anzahl Einwohner und Anzahl Beschäftigte. Die 3D Darstellungen der Gebäude werden nicht gespeichert, können aber im parametrischen Modell interaktiv im Kontext des umgebenden Terrains und des aktuellen Gebäudebestandes betrachtet werden. Als ergänzende Information zur Struktur der Bebauung werden analog zum Process Model die Gebäudehöhe, die Geschosszahl und die Geschossfläche ausgewertet. Für die Parzellen des Projektperimeters werden diese Grössen direkt aus den Ergebnissen des Change Models übernommen. Für die Parzellen in der Umgebung des Projektgebietes werden die Werte des aktuellen Gebäudebestandes übernommen, welche im Process Model aus den swissBUILDINGS3D Daten abgeleitet wurden. Die Werte der Parzelle werden als Durchschnitt auf die harmonisierten Bauzonen aggregiert. Die Modellzonen und deren Nutzungen werden dabei nicht berücksichtigt. Zusätzlich wird die Verteilung von Gebäudehöhe, Geschosszahl und Geschossfläche im Projektgebiet aufgezeigt, wobei die Ist-Situation der Situation bei angewendetem Szenario gegenübergestellt wird. In Bezug auf Gebäudehöhe und Geschosszahl ist anzumerken, dass diese Werte innerhalb des Projektperimeters prinzipbedingt nahe an den Parameterwerten für maximale Geschosszahl und/oder Gesamthöhe liegen, da das parametrische Modell die Ausreizung der maximalen Geschosszahl priorisiert.

Recht ähnlich werden die Elemente zur baulichen Dichte gehandhabt. Als Kenngrössen werden die Baumassenziffer, die Geschossflächenziffer und die Überbauungsziffer pro Parzelle abgeschätzt. Dabei werden für die Parzellen des Projektperimeters die Outputs des parametrischen Modells zur Berechnung herangezogen, welche analog zum Vorgehen im Process Model erfolgt. Anzumerken ist, dass für die anrechenbare Grundstücksfläche der auf den Projektparzellen hinterlegte Wert verwendet wird. Insgesamt ist dadurch die Berechnung der Kennwerte für die Szenarien besser nachvollziehbar als die Berechnung der abgeschätzten Werte für die Ist-Situation. Für die umgebenden Parzellen wird wiederum der Wert des aktuellen Gebäudebestandes herangezogen, der im Rahmen des Process Models berechnet wurde. Gleichfalls wird für Baumassenziffer, Geschossflächenziffer und Überbauungsziffer zusätzlich die Verteilung im Projektgebiet für Ist-Situation und das jeweilige Szenario gegenübergestellt. In Bezug auf diese drei Kennzahlen ist anzumerken, dass im Projektperimeter bei Verwendung einer Nutzungsziffer zur Dimensionierung der Gebäude innerhalb einer Modellzone die

Werte der korrespondierenden Kennzahl aus der Analyse prinzipbedingt oft nahe am Parameterwert liegen. Andere Parameter wie Grenzabstände, Geschosszahl, Gesamthöhe und Grünflächenziffern können allerdings zu Werten führen, welche vom Parameterwert deutlich abweichen.

Auch für das Impact Model werden zusätzlich die bauliche Dichte in Form des überbauten Volumens sowie der Freiflächenanteil in Bezug auf das Hektarraster ausgewertet. Für das Bauvolumen über dem Projektperimeter werden die Daten aus dem parametrischen Modell herangezogen. Die Auswertung erfolgt in einer Version rein in Bezug auf den Projektperimeter. In den überdeckenden Hektarzellen werden dabei allfällige Parzellen ausserhalb des Perimeters unberücksichtigt gelassen (siehe Abbildung 7). Gerade in Zellen am Rand des Projektgebietes führt dies zu einer Unterschätzung, weshalb in einer zweiten Variante für die Parzellen ausserhalb des Perimeters die Volumina des aktuellen Gebäudebestandes auf Basis des Process Models mitberücksichtigt werden. Die Auswertung des Freiflächenanteils erfolgt analog in einer Variante mit und ohne Berücksichtigung der Ist-Situation. Für die Bestimmung des Freiflächenanteils in den Parzellen des Projektperimeters wird die Parzellenfläche abzüglich der überbauten Flächen in Form der Gebäudegrundrisse aus den Modelldaten als Freifläche angenommen. Erschliessungsflächen oder andere versiegelte Flächen werden also nicht gesondert behandelt.



Quelle: eigene Darstellung

Abbildung 7: Auswertung bauliche Dichte pro Hektare mit (r.) und ohne (l.) Berücksichtigung des Umfelds

3.6.3 Weitere Dichtefaktoren

Für die weiteren Dichtefaktoren werden analog zum Process Model die Einwohnerdichte und Beschäftigtendichte sowie die aus deren Kombination resultierende Raumnutzerdichte ausgewertet. Für die Parzellen des Projektperimeters wird aufgrund der im parametrischen Modell generierten Gebäude und deren Geschossfläche auf der einen Seite und dem parametrisierten Nutzungsverhältnis zwischen Wohnen und Arbeit sowie dem jeweiligen Platzbedarf auf der anderen Seite die Werte für die Anzahl Einwohner und Beschäftigte auf dem Grundstück abgeschätzt. Im Rahmen der Berechnung werden diese Werte nicht gerundet, was bei der Visualisierung aber durch die Klasseneinteilung mit ganzzahligen Grenzen ausgeglichen wird. Durch Addition der Werte für die Einwohner und Beschäftigte wird der Wert für die Raumnutzer abgeleitet. Ähnlich wie bei der Aufbereitung der baulichen Dichtekennziffern im Hektarraster, werden für Einwohnerdichte, Beschäftigtendichte und Raumnutzerdichte einerseits eine Auswertung nur unter Berücksichtigung des Projektperimeters und andererseits eine Auswertung in Kombination mit den Ist-Werten aus dem Umfeld durchgeführt. Die Aggregierung der Werte aus dem Projektperimeter erfolgt anteilig zur Gebäudefläche innerhalb der jeweiligen Hektarzelle. Da die Ist-Werte aus dem Umfeld im Hektarraster aggregiert vorliegen und eine vollständige Disaggregation kaum zu bewerkstelligen ist, wird für die kombinierte Auswertung von Ist-Situation und Szenario eine anteilige Anrechnung der Ist-Werte zu den Werten aus dem Projektperimeter vorgenommen. Weil die Ist-Werte bezogen auf die Gebäude erfasst und anschliessend auf die Hektare aggregiert werden, wird der Anteil der Gebäudefläche, der innerhalb der Hektarzelle aber ausserhalb der Projektparzellen liegt, für die Bestimmung des Anteils zur Anrechnung verwendet.

Als weitere Grösse wird die durchschnittliche Wohnfläche pro Einwohner bezogen auf die Hektare ausgewiesen. Die Aggregation der Werte aus dem Projektperimeter erfolgt wiederum anteilig über die Gebäudefläche innerhalb der Hektarzelle. Für die durchschnittliche Wohnfläche pro Einwohner wird ebenfalls wie für die oben erwähnten Kennzahlen eine Auswertung nur bezogen auf den Projektperimeter sowie eine Auswertung in Kombination mit den Ist-Werten des Umfeldes vorgenommen. Die anteilige Anrechnung der Ist-Werte erfolgt analog. Es ist anzumerken, dass im Projektgebiet der Wert für die durchschnittliche Wohnfläche pro Einwohner prinzipbedingt stark von den parametrisierten Werten für den Platzbedarf Wohnnutzung abhängt. Überschneidet sich eine Hektarzelle nur mit Parzellen einer einzigen Planungszone oder mit Planungszonen mit dem gleichen Parameterwert für Platzbedarf Wohnnutzung, so ist der Durchschnitt mit dem Parameterwert identisch.

Aufgrund der relativ einfachen Ausgestaltung dieser Modellrechnung können keine Verteilungen nach Kategorien für diese weiteren Dichtefaktoren ausgegeben werden. Auch eine Angabe zur Haushaltsgrösse und der Anzahl Wohnungen ist deshalb nicht ohne weiteres möglich. Für solche Auswertungen wären komplexere Modell wie z.B. ein Agentenbasiertes Modell unter Berücksichtigung zusätzlicher Faktoren wie Demographie, wirtschaftliche Entwicklung und Verkehr nötig, um brauchbare Aussagen zu generieren. Solche Modelle können z.B. mit MatSim umgesetzt werden und fussen teilweise auf den gleichen Inputdaten, die schon im Rahmen des Process Models genutzt werden (Efthymiou *et al.* 2013; Wissen Hayek *et al.* 2016). Der Aufwand für Aufbau und Integration eines glaubwürdigen Modells dieser Art hätte aber den Rahmen dieser Arbeit gesprengt.

3.7 Bewertung der Szenarien anhand von Dichtekennzahlen

3.7.1 Decision Model

Der sechste und letzte Schritt des Prozesses dreht sich um die Frage, wie das Projektgebiet verändert werden soll (Steinitz 2012). Decision Models helfen bei der Beantwortung dieser Frage. Die Beurteilung der verschiedenen Optionen und die schlussendlichen Entscheidungen hängen wiederum von den Erwartungen sowie dem kulturellen Wissen der Entscheidungsträger ab (Steinitz 2012). Für den Entscheidungsfindungsprozess sind die Auswirkungen des Change Models zu beurteilen und zu vergleichen, um zu entscheiden, ob und welche Änderungen umzusetzen oder ob weitere Anpassungen am Studiengebiet beziehungsweise dem Change Model nötig sind (Steinitz 2012). Dieser Vorgang erfolgt im Dialog zwischen den Planern, welche die Studie durchführen, und den Entscheidungsträgern. Der genaue Ablauf dieses Prozesses wird im Rahmen dieser Arbeit offengelassen und muss abhängig vom Projekt individuell ausgestaltet werden. Infolge der Ausrichtung des Vorgehensmodells auf Testplanungen muss in diesem Schritt zudem nicht zwingend eine Entscheidung für eine einzelne Variante getroffen werden. Vielmehr ist die Auswahl mehrerer geeigneter Varianten als Grundlage für weitere Arbeiten denkbar.

Zur Unterstützung des Entscheidungsfindungsprozesses werden die Ergebnisse des Change und Impact Models so aufbereitet, dass diese im Dialog mit den Entscheidungsträgern als unterstützendes Hilfsmittel eingesetzt werden können. Dies umfasst einerseits die Möglichkeit der interaktiven Nutzung des Change Models für die dreidimensionale Visualisierung der Auswirkungen der verschiedenen Szenarien im Projektgebiet. Andererseits werden die Ergebnisse des Impact Models so aufbereitet und visualisiert, dass ein Vergleich zwischen den Szenarien sowie mit der Ist-Situation ermöglicht wird. Im Rahmen der Arbeit wurden diese Hilfsmittel zur Feinabstimmung der Szenarien und für die Auswertung der Ergebnisse verwendet.

3.7.2 Hilfsmittel zur Entscheidungsunterstützung

Das parametrische Modell, welches das Change Model umsetzt, und insbesondere dessen Fähigkeit zur dreidimensionalen Visualisierungen der Auswirkungen der Parameter auf die Situation im Projektgebiet werden als wichtiges Hilfsmittel im Dialog mit den Entscheidungsträgern angesehen, um die Konsequenzen der Entscheidungen einfach und intuitiv fassbar zu machen (Walz *et al.* 2008; Wissen Hayek *et al.* 2013; Moura 2015). Ein solcher Einsatz ist bereits vor der eigentlichen Entscheidungsphase im Rahmen partizipativer Prozesse unter Beteiligung von Öffentlichkeit und Entscheidungsträgern anzuraten, um breit abgestützte Grundlagen für die formalen Entscheidungen erarbeiten zu können (Van Wezemael *et al.* 2014; Moura 2015; Wissen Hayek *et al.* 2016). Dieser Beitrag zur Erarbeitung von Grundlagen wird als Haupteinsatzgebiet des erarbeiteten Modells angesehen, zielt dieses

doch auf die effiziente Erarbeitung von Varianten für Testplanungen im Rahmen informeller Verfahren ab.

Spezifisch für die Entscheidungsunterstützung wird als Teil des Decision Models einerseits ein Bericht mit den Resultaten aus den Change und Impact Models aufbereitet. Dieser ist analog zum Bericht für das Evaluation Model gestaltet und ist für die Abgabe in gedruckter Form oder als PDF gedacht. Die Umsetzung des Berichts erfolgt erneut in einem QGIS Projekt mit Layern für die Analyseergebnisse und einem Kartenlayout zur Gestaltung des eigentlichen Berichts. Dieses Layout integriert neben den Karten aus QGIS extern erzeugte Grafiken. Das aus dem Change Model heraus generierte Projekt umfasst neben der Auswertung des Impact Models zu einem spezifischen Szenario jeweils auch die Layer und das Layout für die Analyse der Ist-Situation im Projektgebiet. Zusätzlich zur Erzeugung der Berichte kann das QGIS Projekt also für die interaktive Präsentation und Analyse des Projektgebiets sowie den Vergleich zwischen Ist-Situation und Szenario genutzt werden.

Ein erster Teil des Berichts zeigt eine Übersicht des Projektgebiets nach Anwendung des Szenarios. Eine Karte zeigt eine erste Übersicht mit den Parzellen des Projektperimeters, den vom Modell generierten Gebäudegrundrissen sowie den Grundrissen der aktuellen Bebauung ausserhalb des Perimeters. Eine weitere Karte zeigt die Aufteilung der Parzellen in Modellzonen. Diese Ansicht wird durch eine Aufstellung der Parameterwerte pro Modellzone in tabellarischer Form ergänzt. Des Weiteren integriert das Berichtslayout Exporte der 3D Ansichten aus dem Change Model. Diese umfassen die Parzellen des Projektperimeters, die vom Modell generierten Gebäude sowie das Terrain, Strassen und die Gebäude der aktuellen Bebauung im Umfeld als Kontext.

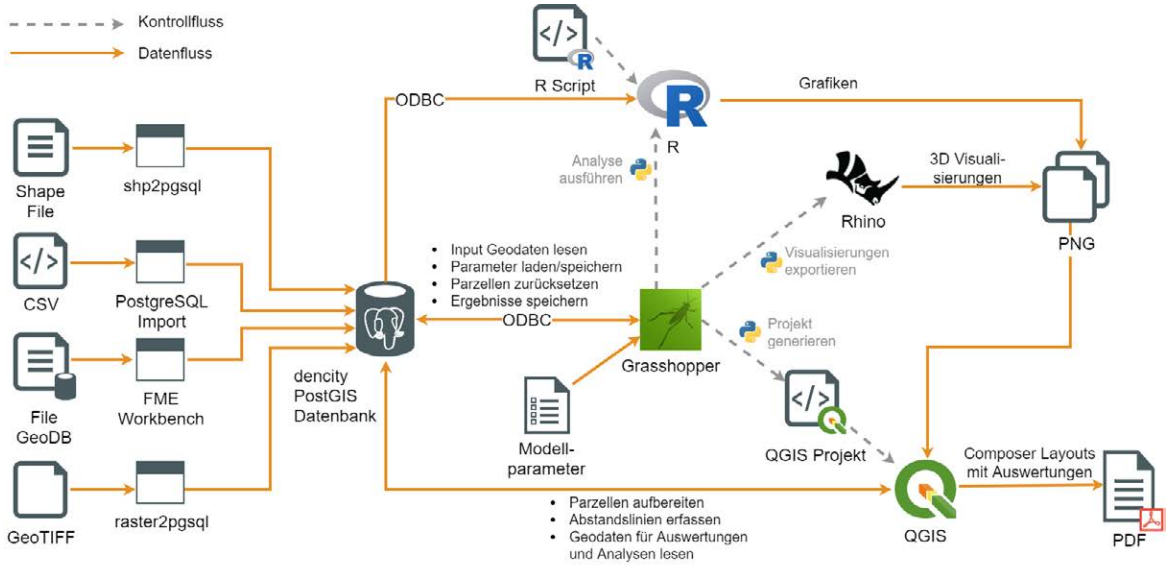
Die Übersicht der Szenario Resultate wird gefolgt von den visualisierten Elementen des Impact Models. Die Reihenfolge, Aufbereitung und Visualisierung dieser Elemente wurde möglichst nahe am Bericht zum Evaluation Model gehalten, um eine Vergleichbarkeit zu gewährleisten. Aspekte wie Skalen, Farbgebung und Bezugsflächen für die Aggregation wurden direkt übernommen. In einem ersten Teil werden die durchschnittliche Gebäudehöhe, Geschosszahl und Geschossfläche aggregiert auf die harmonisierten Bauzonen als Karten visualisiert. Dabei werden im Bericht nur die Analyse-Layer eingebunden, welche die Werte des Szenarios für die Parzellen im Projektperimeter mit den Werten der Ist-Situation für die Umgebung kombinieren. Die Layer, welche nur das Projektgebiet auswerten, sind aber bei interaktiver Nutzung im QGIS Projekt zugänglich. Wiederum werden die Karten zu den baulichen Kennzahlen um eine Zusammenstellung von Histogrammen ergänzt, um die Verteilungen aufzuzeigen. Im Unterschied zum Evaluation Model werden aber die Verteilungen im Projektgebiet im Szenario den Verteilungen der Ist-Situation des Projektgebiets gegenübergestellt. Für die Verteilung der Ist-Situation im gesamten Gemeindegebiet muss der Bericht zur Ist-Situation herangezogen werden. In einem nächsten Teil erfolgt die Visualisierung von Überbauungsziffer, Baumassenziffer und Geschossflächenziffer bezogen auf die Parzelle als Karten, welche wiederum von einer Serie Histogramme zu den Verteilungen begleitet werden.

In einem dritten Teil folgen schliesslich die Auswertungen mit Bezug auf das Hektarraster. Hier wurde ebenfalls die Gestaltung und Reihenfolge der Elemente aus dem Bericht zum Evaluation Model beibehalten. Analog zu den baulichen Grössen werden im Bericht für die Hektarraster die Karten eingebunden, welche die Werte des Szenarios für den Projektperimeter mit den Ist-Werten für die Umgebung kombinieren. Bei den Rastern macht die bauliche Dichte mit dem überbauten Volumen pro Hektare den Anfang, gefolgt vom Freiflächenanteil. Es folgen die Raster für Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte, wobei für die letzten beiden im Vergleich zum Bericht des Evaluation Models keine zusätzlichen Karten mit Verteilungen nach Altersklasse und Wirtschaftssektoren vorliegen. Den Abschluss bildet die Karte zur durchschnittlichen Wohnfläche.

4 Umsetzung

4.1 Übersicht technische Architektur

Abbildung 8 zeigt eine Übersicht der technischen Architektur zur Umsetzung des in Kapitel 3 beschriebenen Vorgehensmodells.



Quelle: eigene Darstellung

Abbildung 8: Übersicht technische Architektur

Eine zentrale Komponente der Umsetzung ist die Datenbank, die als Datenspeicher für alle Daten der entwickelten Lösung dient. Hierfür wird im Rahmen dieser Arbeit der PostgreSQL Server in der Version 9.6.5 (PostgreSQL Global Development Group 2017) als relationaler Datenbankserver mit dem zusätzlichen PostGIS Aufsatz in der Version 2.4.1 (PostgreSQL Global Development Group 2017) zur Unterstützung räumlicher Daten eingesetzt. Auf der linken Seite von Abbildung 8 sind die Geodaten und Formate angedeutet, welche für das Projekt mit Hilfe verschiedener Werkzeuge in die Datenbank integriert werden. Diese Integration, die in Kapitel 4.2 näher erläutert wird, erfolgt zu Beginn des Prozesses, da in den nachfolgenden Schritten die Daten als Grundlage verwendet werden. Die Komponenten für die Umsetzung der weiteren Prozessschritte werden in der Abbildung rechts von der Datenbank aufgezeigt.

Für Bearbeitung und Auswertung der räumlichen Daten wurde im Rahmen der Arbeit zunächst QGIS in der Version 2.18 (QGIS Development Team 2017) verwendet. Im späteren Verlauf erfolgte aufgrund von Verbesserungen beim Layout von Karten und Berichten der Wechsel auf QGIS Version 3.0 (QGIS Development Team 2018). Dank der integrierten Unterstützung von PostGIS Datenbanken können räumliche Daten aus der zentralen Datenbank des Projekts direkt angezeigt und bearbeitet werden. Neben QGIS und dessen integriertem Datenbank Client wurde für die Ausführung von SQL Befehlen im Rahmen von Datenimport und -aufbereitung der *psql* Kommandozeilen-Client des PostgreSQL Servers genutzt, der in Abbildung 8 nicht speziell gezeigt wird.

In QGIS wird intensiv Gebrauch von der Definition von Layern auf Basis von SQL Abfragen gemacht, um einen grossen Teil der räumlichen Analysen direkt auf der Datenbank auszuführen. Die Zusammenstellung der Analyseergebnisse als Bericht erfolgt ebenfalls in QGIS, indem Layouts definiert werden, welche bei Bedarf als PDF Dokumente exportiert werden können. Neben Karten auf Basis der in QGIS definierten Layer binden diese Layouts darüber hinaus Bilder im PNG Format mit 3D Visualisierungen sowie Auswertungen in Form von Diagrammen ein, welche extern mit Hilfe von Rhino beziehungsweise R erstellt wurden.

Die Statistikumgebung R in der Version 3.4.0 (R Core Team 2017) wird genutzt, um zusätzliche Auswertungen durchzuführen und zugehörige Grafiken zu erzeugen. Hier ergänzt R die Funktionen von

QGIS, welches praktisch keine Möglichkeit zur Erzeugung von Diagrammen bietet, welche direkt in Kartenlayouts eingebunden werden können. In R können hingegen verschiedene (statistische) Auswertungen ausgeführt, Diagramme erzeugt und als Bilddatei exportiert werden. Konkret werden verschiedene Verteilungen als Histogramme visualisiert und als PNG Bilddateien exportiert, welche anschliessend von den QGIS Kartenlayouts eingebunden werden. Die Daten für die Auswertungen in R werden über eine ODBC (Open Database Connectivity) Verbindung mittels SQL Statements direkt auf der zentralen Datenbank abgefragt.

Eine weitere zentrale Komponente der technischen Umsetzung ist Grasshopper (Robert McNeel & Associates 2017b). Dabei handelt es sich um eine Erweiterung für Parametrisches Design für die CAD und 3D Computergrafik Software Rhinoceros (Rhino) in der Version 5 (Robert McNeel & Associates 2017c). Einerseits ist der Kern des parametrischen Modells zur Erzeugung der Szenarien im vierten Schritt des Vorgehensmodells in Grasshopper umgesetzt. Andererseits sind im diesem parametrischen Modell auch eine Reihe von Funktionen implementiert, welche Grasshopper zum koordinierenden und verbindenden Element der technischen Umsetzung machen. Für die Umsetzung des Modells werden neben eingebauten Funktionen eine Reihe von Add-ons für die Erweiterung des Funktionsumfangs von Grasshopper genutzt. Für die Anbindung von Daten aus der Datenbank kommt das Slingshot! Add-on (Miller 2015) zum Einsatz, das die Nutzung einer ODBC Verbindung ermöglicht. Über die ODBC Schnittstelle werden die Inputdaten für das Modell eingelesen, Parameterwerte für Szenarien gelesen und gespeichert sowie die vom Modell generierten Ergebnisse zurück in die Datenbank geschrieben. Zudem wird im Rahmen des Prozesses PostGIS für räumliche Verschnitt-Operationen bei der Anwendung von Grenzabständen auf Parzellen eingebunden. Da allerdings weder Grasshopper noch Slingshot! direkt räumliche GIS Datentypen unterstützen, wird GeoJSON (Butler *et al.* 2016) als Schnittstellenformat für die Geometrien eingesetzt. Mittels Python Scripting wird zwischen GeoJSON und den Rhino internen Geometrietypen übersetzt. Um solche Scripts zur Funktionserweiterung von Grasshopper einbinden zu können, wird das GhPython Add-on (Robert McNeel & Associates 2017a) genutzt. Neben der Anbindung der Datenbank werden Python Scripts an verschiedenen Stellen für die Koordination der Nutzung des parametrischen Modells eingesetzt. Dazu gehören unterschiedliche Funktionen für das Benutzerinterface, welche z.B. das Speichern und Laden von Modellparametern aus der Datenbank oder auch die Speicherung der Resultate auslösen. Weitere Scripts dienen der Integration von R und QGIS. Auf Knopfdruck im parametrischen Modell kann für das aktuell geladene Szenario automatisch ein QGIS Projekt erzeugt werden, welches die Layer für die räumliche Analyse sowie die Definitionen der Kartenlayouts der zugehörigen Berichte umfasst. Ebenfalls kann der Benutzer mittels eines anderen Python Scripts einen parametrisierten Aufruf eines R Scripts für die Generierung der Grafiken zum aktuellen Szenario anstossen. Auf ähnliche Weise werden mittels Python zudem aus Rhino heraus 3D Visualisierungen der generierten Verdichtungsvariante als Grafik exportiert. Die exportierten Grafiken aus R und Rhino werden wiederum von den Kartenlayouts im generierten QGIS Projekt eingebunden.

4.2 Datenintegration und Analyse

4.2.1 Datenquellen

Tabelle 5 gibt eine Übersicht der externen Daten und deren Quellen, welche als Datengrundlage für die Umsetzung des Geodesign Prozesses in die zentrale Datenbank integriert wurden. Bei der Auswahl der Quellen wurden möglichst aktuelle Daten verwendet. Bei den verschiedenen Statistikdaten wurde allerdings darauf geachtet, dass Daten aus dem gleichen Stichjahr verwendet werden, um die zeitliche Konsistenz der Resultate zu gewährleisten. Infolge der zeitlichen Verzögerung bei der Statistik der Unternehmensstruktur werden deshalb Daten zum Jahr 2015 als Quelle für die Analysen verwendet. Bei der Auswahl der Datenquellen wurde einerseits auf eine möglichst breite und einheitliche Verfügbarkeit für die ganze Schweiz geachtet. Andererseits wurde frei verfügbaren Datenquellen der Vorzug gegeben. Die Datensätze des Bundes sind schweizweit verfügbar. Die vom Kanton Bern bezogenen Daten der Amtlichen Vermessung sind dank eines einheitlichen Basisdatenmodells in ähnlicher Form gleichermassen von anderen Kantonen verfügbar (cadastre.ch 2018). Die swissBUILDINGS3D 2.0 Daten der swisstopo sind der einzige Datensatz, der nur kostenpflichtig verfügbar ist. Für das Höhenmodell wurde den SRTM Daten mit 30m Auflösung der Vorzug vor dem schweizweit frei verfügbaren Datensatz DHM20/200 mit 200m Auflösung gegeben. Von swisstopo sind zwar mit Datensätze mit höherer Auflösung erhältlich (DHM25 mit 25m und swissALTI3D mit 2m), diese sind aber kostenpflichtig. Da dem

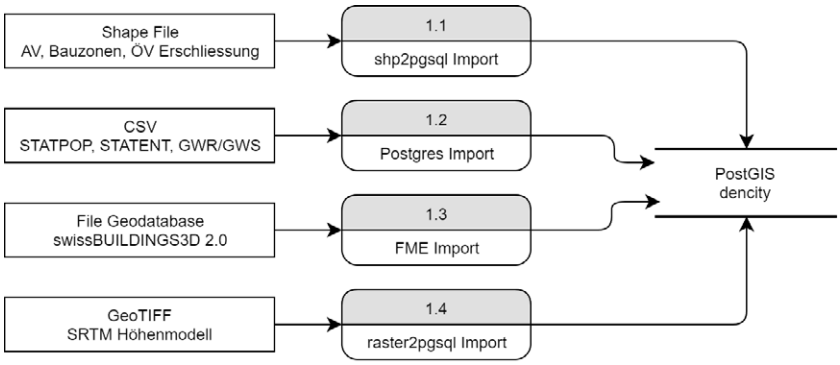
Höhenmodell im Modell nur unterstützender Charakter bei der Visualisierung zukommt, wurden Abstriche bei der Genauigkeit im Gegenzug zur freien Verfügbarkeit in Kauf genommen.

Tabelle 5: Übersicht Datenquellen

Bezeichnung	Format	Verfügbarkeit	Stand	Quelle
Statistik der Unternehmensstruktur (STATENT), Beschäftigte und Arbeitsstätten: Geodaten	CSV	frei / kommerziell	2015 (provisorisch)	Bundesamt für Statistik (BFS 2017d)
Neue Volkszählung, Bevölkerung, Privathaushalte (STATPOP): Geodaten	CSV	frei / kommerziell	2015 und 2016	Bundesamt für Statistik (BFS 2017b)
Gebäude- und Wohnungsstatistik (GWS): Geodaten	CSV	frei / kommerziell	2015	Bundesamt für Statistik (BFS 2017a)
Amtliche Vermessung vereinfacht (MOPUBE) • **BBF: Bodenbedeckung** • **GAEINP: Gebäudeeingang** • **HGF: Gemeindegrenzen** • **LIF: Liegenschaften: Grundstücke**	Shapefile / File Geodatabase	frei	10.08.2017	Amt für Geoinformation des Kantons Bern (AGI 2017a)
Übersichtszonenplan 1:25'000	Shapefile	frei	06.04.2017	Amt für Geoinformation des Kantons Bern (AGI 2017b)
Geodaten Bauzonen Schweiz (harmonisiert)	Shapefile	frei	2017	Bundesamt für Raumentwicklung ARE (ARE 2017)
Geodaten ÖV-Güteklassen	Shapefile	frei	2017	Bundesamt für Raumentwicklung ARE (ARE 2018b)
swissBUILDINGS3D 2.0	File Geodatabase	kommerziell	2017	Bundesamt für Landestopografie swisstopo (swisstopo 2017)
SRTM 90m Digital Elevation Data Version 4	GeoTIFF	Frei	2008	NASA/Consortium for Spatial Information (CGIAR-CSI) (NASA 2008)

4.2.2 Datenintegration

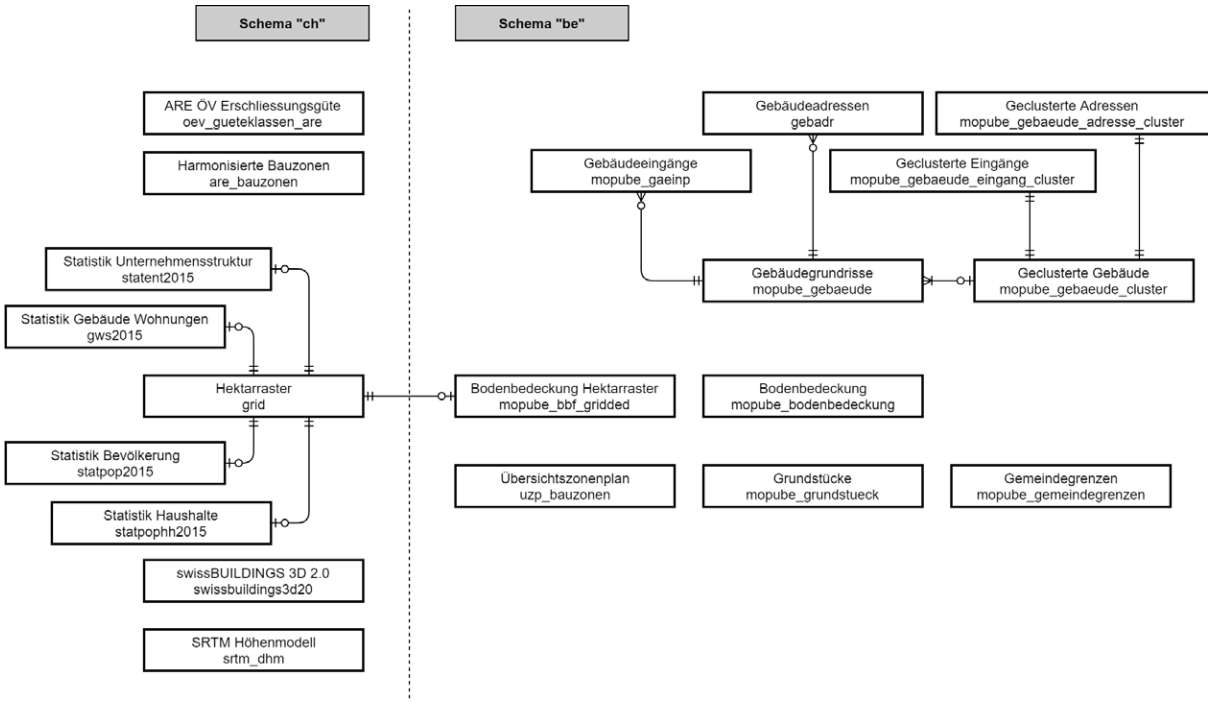
Abbildung 9 zeigt schematisch den Datenfluss für den initialen Import der externen Daten, welche als Input für den Geodesign Prozess dienen. Alle Inputdaten werden in die zentrale PostGIS Datenbank importiert. Abhängig vom Datenformat der Inputdaten, werden unterschiedliche Methoden für den Import eingesetzt.



Quelle: eigene Darstellung

Abbildung 9: Datenfluss initialer Import von Inputdaten

Abbildung 10 zeigt eine schematische Übersicht des Datenmodells für die Inputdaten in der PostgreSQL Datenbank. Die Tabellen für das Projekt werden mit Hilfe von Schemen organisiert. Die Inputdaten werden abhängig von ihrer geographischen Abdeckung entweder in das *ch* Schema für schweizweite Daten oder in das *be* Schema für Daten des Kantons Bern integriert. Für alle räumlichen Daten wird als Referenzsystem das Bezugssystem LV95 (EPSG:2056) der neuen Schweizer Landesvermessung verwendet. Die meisten der Daten liegen in diesem Bezugssystem vor. Wo nötig wird eine Reprojektion nach LV95 vorgenommen. Man beachte, dass das Datenmodell in Abbildung 10 neben den Tabellen mit den importierten Daten einige Tabellen zeigt, die im Zuge der Datenaufbereitung und -analyse ergänzt werden. Auch in den Tabellen mit den Inputdaten werden nach Bedarf einzelne Spalten für abgeleitete Werte sowie Indizes ergänzt.



Quelle: eigene Darstellung

Abbildung 10: Datenmodell Inputdaten

Als Esri Shape File vorliegende Daten werden mit Hilfe des *shp2pgsql* Tools sowie des PostgreSQL Kommandozeilenclients direkt in die Datenbank importiert. Das Schema der Zieltabellen wird von *shp2pgsql* automatisch auf Basis der Datentypen in der Quelldatei generiert. In diese Kategorie fallen die Daten der Amtlichen Vermessung und der Übersichtszonenplan des Kantons Bern, die harmonisierten Bauzonen sowie die Erschliessungsgüte des Öffentlichen Verkehrs.

Als zweite Gruppe liegen die Daten des Bundesamtes für Statistik als CSV Dateien vor. Diese werden mit dem in PostgreSQL integrierten COPY Befehl direkt aus der CSV Datei in die Zieltabellen innerhalb der Datenbank integriert. Diese Tabellen wurden vor dem Import explizit angelegt und beeinflussen mit den Datentypen ihrer Spalten die Konversion der Textdaten beim Import durch den COPY Befehl. Die importierten Daten des Bundesamtes für Statistik beziehen sich auf ein Hektarraster, enthalten aber keine expliziten Geometrien. Stattdessen ist jeweils die X und Y Koordinate der unteren rechten Ecke der korrespondierenden Hektarzelle vorhanden, aus denen zudem ein eindeutiger Identifikator abgeleitet wird. Um die Geometrie explizit zu machen, wird im Rahmen der Datenaufbereitung eine zusätzliche Datenbanktabelle erzeugt, welche das Hektarraster als Vektordaten abbildet. Für die Auswertung und Darstellung werden die importierten Daten über den eindeutigen Identifikator mit den Geometrien in der Tabelle des Hektarrasters verknüpft.

Die swissBUILDINGS3D 2.0 Daten der swisstopo liegen als Esri File Geodatabase vor und werden mit Hilfe eines Prozesses in die Datenbank integriert, der mittels der auf FME basierenden Data Interope-

rability Extension von ArcGIS Desktop umgesetzt ist. Dies einerseits aufgrund der guten Formatunterstützung und andererseits aufgrund der Fähigkeit von FME, die Multipatch Daten in den File Geodatabase in PolyhedralSurface Objekte für die Abbildung in PostGIS umzuwandeln. Das Schema der Zieltabelle in der Datenbank wird von FME auf Basis der im Prozess hinterlegten Datentypen automatisch erzeugt. Die swissBUILDINGS3D Daten liegen in der File Geodatabase gekachelt als separate Tabellen vor, wobei Inhalt und Namensgebung der Tabellen mit der Kachelaufteilung von SWISSIMAGE korrespondieren (swisstopo 2010). Der Importprozess überführt die Daten aus den Quelltabellen in eine einzelne Zieltabelle in der PostgreSQL Datenbank, wobei der Name der Quelltabelle als neues Attribut zu den Daten hinzugefügt wird. Dieses Attribut kann bei der Abfrage und Verarbeitung der Daten als Selektions- und Gruppierungskriterium genutzt werden.

Als vierte und letzte Kategorie liegen die SRTM Daten als Raster im GeoTIFF Format vor. Diese Daten werden mittels des *raster2pgsql* Tools sowie des PostgreSQL Kommandozeilenclients direkt in die Datenbank importiert. Die Zieltabelle wird dabei automatisch vom Tool generiert. Die Daten werden in der Datenbank als Raster abgespeichert. Beim Import wird das Raster, welches etwas mehr als die Schweiz abdeckt, in Abschnitte von 20 mal 20 Pixeln (d.h. 600 mal 600 Meter) unterteilt. Vor dem Import wurden die SRTM Daten in QGIS aufbereitet, wobei zwei SRTM Kacheln zu einem einzelnen Raster zusammengefügt und in das Schweizer Landeskoordinatensystem transformiert wurden.

4.2.3 Analyse Ist-Situation

4.2.3.1 Übersicht

Für die Analyse der Ist-Situation werden in QGIS und R verschiedene Auswertungen auf der Datenbank durchgeführt. Die Auswertungen in QGIS sind in Form von Layern auf Basis von SQL SELECT Statements umgesetzt. Diese Layer sind jeweils zusammen mit Layern zur Analyse eines bestimmten Szenarios Teil eines QGIS Projektes, welches mittels des parametrisierten Modells generiert werden kann. In R werden ebenfalls SQL SELECT Statements auf der Datenbank abgesetzt, deren Resultate anschliessend zu Grafiken aufbereitet werden. Die SELECT Statements dienen in erster Linie dazu, das gewünschte Subset von Daten aus der zentralen PostGIS Datenbank abzufragen und in eine für die Visualisierung geeignete Form zu bringen. Dies umfasst insbesondere auch JOIN Operationen zwischen Tabellen und die Aggregation auf verschiedene Flächeneinheiten. Um den Aufwand bei der Abfrage für die eigentliche Auswertung zu reduzieren, werden aufwändigere Berechnungs- und Analyseschritte im Vorfeld während der Datenintegration ausgeführt (vergleiche Anhang B.1). Dies beinhaltet Schritte wie die Generierung des Hektarrasters, die Abschätzung von Gebäudehöhe, Geschosszahl, Geschossfläche sowie Volumen aus 3D Gebäudemodellen, die Bestimmung der Gebäudetypologie und die Abschätzung von Kennziffern. Die nachfolgenden Unterabschnitte geben eine Übersicht zu den involvierten Schritten und Auswertungen. Details zu den Datenflüssen können dem Anhang B entnommen werden. Nicht speziell behandelt werden die Analyse der harmonisierten Bauzonen, Übersichtszonenplan Bauzonen, Bodenbedeckung, ÖV Güteklasse, Gebäude und Grundstücke, welche für die Analyse unverändert aus der Datenbank übernommen werden. Einzig deren Darstellungsmodell wird in QGIS jeweils individuell konfiguriert. Die Projekt Parzellen werden ebenfalls unverändert aus der Datenbank übernommen, wobei aber eine Filterung auf ein spezifisches Projekt erfolgt. Der Layer Projektperimeter ist eine Variante der Projekt Parzellen, bei dem die Geometrien vereinigt und das Styling für eine Hervorhebung angepasst wurde.

4.2.3.2 Gebäudehöhe, Geschosszahl und Geschossfläche

Für die Auswertung von Gebäudehöhe, Geschosszahl und Geschossfläche werden die swissBUILDINGS3D 2.0 Daten der swisstopo als Quelle herangezogen. Während der Datenintegration werden diese Grössen aus den 3D Modellen der Gebäude abgeschätzt. Das detaillierte Vorgehen kann dem Anhang B.1.6 entnommen werden. Die Gebäudehöhe wird als Differenz des Z-Wertes des höchsten und niedrigsten Punkt des Gebäudemodells bestimmt. Wenn die Differenz grösser als drei Meter ist, wird der Wert um 3 reduziert, da Gebäude laut Produktdokumentation um diesen Wert in den Boden versenkt werden (swisstopo 2016). Unter der Annahme einer Geschosshöhe von 3m wird die Geschosszahl durch Division und Abrundung aus der Gebäudehöhe abgeleitet. Für jedes Gebäude wird ein Grundriss abgeleitet, indem die Z-Dimension aller Facetten (Faces) auf 0m gesetzt und eine Vereinigung der resultierenden Flächen vorgenommen wird. Daraus wird die Grundfläche bestimmt, deren Multiplikation mit der Geschosszahl die Geschossfläche ergibt. Die Multiplikation von Grundfläche und

Gebäudehöhe ergibt die Abschätzung der Baumasse. Die so bestimmten Werte werden als neue Felder auf den swissBUILDINGS3D Daten hinterlegt.

Zusätzlich werden aus der Bodenbedeckung die Gebäudegrundrisse isoliert, welche ergänzend Gebäude enthalten, die noch nicht in den swissBUILDINGS3D Daten vorhanden sind. Auf diesen Grundrissen wird durch Verschnitt mit den zuvor aufbereiteten swissBUILDINGS3D Daten die Höhe bestimmt. Wird keine direkte Übereinstimmung gefunden, wird der Median der Gebäude im Umkreis von 100m eingesetzt. Analog zu den swissBUILDINGS3D Daten werden für die Gebäudegrundrisse aus der Amtlichen Vermessung die Geschosszahl, Geschossfläche und das Bauvolumen ermittelt. Diese wurden als Vergleichswert zu den swissBUILDINGS3D Daten verwendet, wobei sich generell gute Übereinstimmungen zeigten.

Für die Auswertung werden schliesslich die swissBUILDINGS3D Daten als Quelle herangezogen. Für die Auswertung in QGIS werden die Daten nach harmonisierten Bauzonen aggregiert und der Durchschnitt für Gebäudehöhe, Geschosszahl und Geschossfläche gebildet. In R werden für diese Grössen die Verteilungen gruppiert nach harmonisierten Bauzonen aufgezeigt (vergleiche Anhang B.5.1).

4.2.3.3 Geschossflächenziffer, Überbauungsziffer und Baumassenziffer

Die Geschossflächenziffer, Überbauungsziffer und Baumassenziffer werden als Kennzahlen der baulichen Dichte bezogen auf die Parzelle ausgewertet. Die nötigen Berechnungen für diese Kennziffern werden während der Datenaufbereitung ausgeführt und als neue Spalten auf den Parzellen hinterlegt (vergleiche Anhang B.1.7). In einem ersten Schritt werden die Geometrien der Parzellen korrigiert, um Probleme mit nachfolgenden Overlay Operationen zu vermeiden. Die anrechenbare Grundstücksfläche als Referenzfläche für die Kennzahlen wird ermittelt, indem durch Intersection der Parzelle mit den harmonisierten Bauzonen der Flächenanteil innerhalb der Zonen bestimmt wird. Durch Überlagerung mit den zuvor aufbereiteten Gebäudegrundrissen aus der Amtlichen Vermessung wird pro Parzelle die summierte Gebäudegrundfläche, Geschossfläche und das Volumen bestimmt. Diese werden ins Verhältnis zur anrechenbaren Grundstücksfläche gesetzt, um Überbauungsziffer, Geschossflächenziffer und Baumassenziffer abzuschätzen.

Für die Auswertung in QGIS können die Parzellen direkt abgefragt und die zuvor berechneten Werte für Geschossflächenziffer, Überbauungsziffer und Baumassenziffer direkt visualisiert werden. In R wird für diese Grössen wiederum die Verteilung gruppiert nach harmonisierten Bauzonen aufgezeigt, wozu ein Overlay der Parzellen mit den Bauzonen vorgenommen wird (vergleiche Anhang B.5.1).

4.2.3.4 Bauliche Dichte

Zusätzlich zur Auswertung der baulichen Dichtekennziffern in Bezug auf die Parzelle wird das überbaute Volumen bezogen auf die Hektare ausgewertet (vergleiche Anhang B.6.2). Hierzu werden die Gebäudegrundrisse der swissBUILDINGS3D mit den Zellen des Hektarrasters verschnitten und anschliessend das Volumen aggregiert. Durch den Verschnitt kann es vorkommen, dass das Volumen eines einzelnen Gebäudes auf mehrere Hektarzellen verteilt wird. Das Ergebnis des Verschnitts wird in QGIS als Layer visualisiert. Das abgefragte Gebiet wird dabei auf die Bounding Box des Projektperimeters plus einem Buffer von rund 200m zur Berücksichtigung der Umgebung beschränkt (die genaue Ausdehnung hängt von der Lage des Projektperimeters im Hektarraster und den angeschnittenen Zellen ab).

4.2.3.5 Freiflächenanteil

Komplementär zur Auswertung des überbauten Volumens wird auch der Anteil Freiflächen pro Hektare ausgewertet (vergleiche Anhang B.6.3). Bereits bei der Datenaufbereitung wird hierfür ein Verschnitt der Bodenbedeckung mit den Hektarzellen vorgenommen. Für die eigentliche Auswertung müssen aus den daraus resultierenden Daten dann nur noch die Freiflächen selektiert, auf die Hektarzelle aggregiert und in Verhältnis zur Grundfläche gesetzt werden. Das Resultat wird als Prozentwert in einem QGIS Layer visualisiert. Der Bereich wird dabei analog zur baulichen Dichte auf den Projektperimeter und Umgebung beschränkt. Als Freiflächen wurden die Klassen «Acker, Wiese, Weide», «Gartenanlage», «übrige humusierte», «stehendes Gewässer», «fliessendes Gewässer», «Schilfgürtel», «übrige vegetationslose» aus der Klassierung der Bodenbedeckung ausgewählt. Diese Klassen werden als potentielle Freiflächen im Siedlungsgebiet betrachtet.

4.2.3.6 Gebäudetypologie

Die Gebäudetypologie wird während der Datenaufbereitung in Rahmen eines Prozesses mit mehreren Schritten abgeleitet (vergleiche Anhang B.1.6). Die Ableitung erfolgt auf Basis der Analyse verschiedener Eigenschaften der Formen und Muster von Gebäudegrundrissen, Gebäudeeingängen sowie Clustern von Gebäuden. Als Input dienen die Gebäudegrundrisse (Flächen) und Gebäudeeingänge (Punkte) aus der Amtlichen Vermessung sowie die Gebäudeadressen (Punkte). In einem ersten Schritt werden Gebäude, die aneinandergebaut sind, zu Gebäudeclustern zusammengefasst. Die eigentliche Typologisierung erfolgt anhand dieser Cluster. Alleinstehende Gebäude bilden dabei eigene Cluster. Für jeden Cluster werden verschiedene Eigenschaften wie die Anzahl Geometrien (Häuser), die Anzahl innerer Ringe («Löcher» wie z.B. Innenhöfe), die Fläche mit und ohne innere Ringe, die Fläche der konkaven und konvexen Hülle sowie die Länge des äusseren Perimeters bestimmt. Anhand der Gebäudecluster werden in einem zweiten Schritt die Gebäudeeingänge ebenfalls zu Clustern zusammengefasst. Für diese Eingangscluster werden Kennzahlen wie die Anzahl Eingänge pro Cluster sowie die Fläche und der Umfang der konkaven und konvexen Hülle bestimmt. Die gleiche Clusterung und Ableitung von Kennzahlen wird zudem noch für die Gebäudeadressen vorgenommen. Der Punkt für die Adresse entspricht meist einem der Eingänge. Unterschied zwischen den Eingängen und Adressen ist allerdings, dass ein Haus nur einen Adresspunkt aber mehrere Eingangspunkte haben kann, wodurch sich je nach Konfiguration der Cluster und Anzahl Eingänge leicht andere Kennzahlen als bei den Adressen ergeben.

Die Kennzahlen aus den Gebäude-, Eingangs- und Adressclustern dienen als Input für eine Klassifikation, die jedem Cluster einen Typ zuweist. Dieser Gebäudetyp wird anschliessend auf die ursprünglichen Gebäude übertragen, welche dann in QGIS direkt visualisiert werden. Die Kriterien der Klassifikation basieren einerseits aus manuell gesetzten Kriterien, welche sich aus der Definition der Gebäudetypologien und aus Erfahrungswerten ableiten (z.B. die Anzahl Gebäude pro Cluster, die Anzahl innerer Ringe oder die Schwellwerte bei der Grundfläche). Weitere Kriterien leiten sich aus der Analyse der Inputdaten in R ab, wobei einerseits mit grundlegenden statistischen Analysen und andererseits mit der automatischen Ableitung von Modellen in Form von Entscheidungsbäumen gearbeitet wurde. Für diese Untersuchungen wurde vorgängig ein Datensatz mit 5600 manuell klassifizierten Gebäudeclustern als Referenz angelegt. Die in R untersuchten Kennzahlen und daraus abgeleiteten Kriterien umfassen neben den rohen Kennzahlen verschiedene Verhältnisse wie z.B. das der Grundfläche zur Fläche der konkaven und konvexen Hülle, welche sich abhängig von der Konfiguration der Gebäudecluster unterscheiden. So wird z.B. das Verhältnis der Grundfläche zur Fläche der konvexen Hülle bei einer geraden Reihe nahe 1 liegen, während sich bei einem offenen Blockrand infolge der gekrümmten Form bei der konvexen Hülle eine grössere Fläche und damit ein niedrigeres Verhältnis ergibt.

4.2.3.7 Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte

Die Anzahl Einwohner und die Anzahl Beschäftigte pro Hektare liegen in den Geodaten des Bundesamtes für Statistik zur neuen Volkszählung, Bevölkerung und Privathaushalten (STATPOP) sowie in den Geodaten zur Statistik der Unternehmensstruktur (STATENT) bereits bezogen auf die Hektare vor. Für die Auswertung und Visualisierung in QGIS werden diese tabellarischen Daten anhand des Identifikators für die Hektarzelle miteinander und mit dem Hektarraster verknüpft, das die Geometrie beisteuert. Die Anzahl Raumnutzer pro Hektare wird in der gleichen Abfrage durch Addition der Einwohner und Beschäftigten abgeleitet (vergleiche Anhang B.6.4).

Die Verteilung der Beschäftigten nach Sektoren kann ebenfalls direkt aus den Quelldaten übernommen werden. Die Darstellung erfolgt in diesem Fall über einen separaten Layer mit der Definition eines Kartogramms in Form eines Kuchendiagramms. Die Verteilung der Einwohner nach Altersklassen wird in gleicher Weise visualisiert. Allerdings ist hierfür eine vorgängige Aggregation der pro Altersjahr vorliegenden Daten auf die Altersklassen nötig. Die Einteilung der Altersklassen und die farbliche Gestaltung orientiert sich dabei an Veröffentlichungen des Bundesamts für Statistik (BFS 2018).

4.2.3.8 Wohnungsdichte und Haushaltsdichte

Die Anzahl der Wohnungen pro Hektare liegt in den Geodaten der Gebäude- und Wohnungsstatistik (GWS) bereits bezogen auf die Hektare vor. Die Anzahl der Haushalte kann hingegen der Statistik der Privathaushalte entnommen werden, welche Teil der STATPOP Geodaten ist. Für die Auswertung werden jeweils die tabellarisch vorliegenden Daten anhand des Identifikators für die Hektarzelle mit dem

Hektarraster verknüpft und als QGIS Layer visualisiert (vergleiche Anhänge B.6.6 und B.6.7). Die Verteilung der Wohnungen nach Zimmergrösse und der Haushalte nach Haushaltsgrösse können jeweils direkt den Quelldaten entnommen werden. Die Visualisierung erfolgt in beiden Fällen anhand eines separaten Layers mit einem Kartogramm in Form eines Kuchendiagramms.

4.2.3.9 Durchschnittliche Wohnfläche pro Einwohner

Die durchschnittliche Wohnfläche pro Einwohner wird als Summe der Wohnfläche pro Hektar geteilt durch die Anzahl der ständigen Einwohner pro Hektar als Näherung berechnet. Hierfür werden die Daten des GWS mit den Daten der STATPOP anhand des Identifikators für die Hektarzelle miteinander und mit dem Hektarraster verknüpft (vergleiche Anhang B.6.7). Die Ausgabe erfolgt in Form eines Hektarrasters in QGIS, wobei eine Skala mit sechs Klassen und einem divergierenden Farbverlauf von grün am unteren Ende (niedrige Werte) nach rot am oberen Ende (hohe Werte) verwendet wird. Dieses Farbschema soll die positive Assoziation von niedrigem Platzverbrauch pro Kopf und damit höherer Dichte im Gegensatz zu höherem Platzverbrauch und geringerer Dichte unterstützen.

4.2.4 Analyse Szenarien

4.2.4.1 Übersicht

Die Analyse der Szenarien erfolgt in Anlehnung an die Analyse der Ist-Situation. Wiederum kommen SQL Abfragen auf Basis der zentralen PostGIS Datenbank zum Einsatz, welche in QGIS für die Erzeugung von Layern und in R für die Generierung von Grafiken der Verteilung bestimmter Kennzahlen verwendet werden. Für die Analyse eines Szenarios kann im parametrischen Modell ein QGIS Projekt generiert werden, welches die entsprechenden Layer parametrisiert auf das jeweilige Projektgebiet und Szenario enthält. Die Layer für die Analyse der Szenarien umfassen nur ein Subset der Kennzahlen zur Analyse der Ist-Situation, welche aus den Resultaten des parametrischen Modells abgeleitet werden können. Im QGIS Projekt werden für jede Kennzahl jeweils Layer eingebunden, welche einerseits nur die Parzellen des Projektperimeters und andererseits eine Kombination der generierten Daten für den Projektperimeter mit den Daten zur Ist-Situation im Umfeld auswerten. Für die Erzeugung des Berichts werden jeweils die Layer verwendet, welche die Szenarien kombiniert mit dem Ist-Zustand im Umfeld zeigen. Unterschied zwischen den beiden Varianten ist in erster Linie, welche Datengrundlagen verwendet und wie diese kombiniert werden. Ausserdem werden gewisse Kennzahlen für den Projektperimeter zur Abfragezeit gerechnet und nicht wie im Fall der Ist-Situation vorgängig aufbereitet.

Eine Auswertung und Darstellung der Kennzahlen zum Projektgebiet direkt innerhalb des parametrischen Modells wurde zwar erwogen, aber schlussendlich verworfen. Die dreidimensionale Visualisierung von Kennzahlen in Rhino hat sich bereits bei Versuchen zur Ist-Situation als machbar aber eher schwer interpretierbar herausgestellt (siehe Kapitel *4.3.8*). Die Komplexität der Abfragen und Operationen für die Kombination von generierten Resultaten und Ist-Situation hätten zudem Probleme mit der Performanz mit sich gebracht, welche für eine interaktive Anwendung nicht akzeptabel gewesen wären. Die Umsetzung der Analyse der Szenarien mit den gleichen Mitteln wie die Analyse der Ist-Situation bringt hingegen der Vorteil gemeinsamer Grundlagen und eines einheitlichen Vorgehens.

Die nachfolgenden Unterabschnitte geben eine Übersicht zur Umsetzung der Auswertungen. Details zu Datenflüssen können dem Anhang B entnommen werden. Nicht speziell behandelt werden die Übersicht des Projektgebietes und die Darstellung der Modellzonen, welche weitgehend unverändert aus der Datenbank übernommen und in geeigneter Form dargestellt werden. Für den Layer mit den Modellzonen wird einzig eine Union der Parzellenflächen anhand des Zonencodes vorgenommen.

4.2.4.2 Gebäudehöhe, Geschosszahl und Geschossfläche

Die Auswertung der durchschnittlichen Gebäudehöhe, Geschosszahl und Geschossfläche erfolgt weitgehend analog zur Ist-Situation (vergleiche Anhang B.6.8). Für die Ist-Situation werden wiederum die aufbereiteten swissBUILDINGS3D Gebäude herangezogen, wobei Gebäude, die sich mit Parzellen des Projektgebietes überschneiden, ausgenommen werden. Für die Parzellen des Projektgebietes werden die generierten Gebäude des parametrischen Modells als Quelle verwendet, welche auf denen die drei ausgewerteten Grössen bereits hinterlegt sind. Die kombinierten Gebäude der beiden Quellen werden mit den harmonisierten Bauzonen verknüpft und als Durchschnitt aggregiert. Die Darstellung erfolgt dann analog zur Ist-Situation. Speziell ist, dass der Layer, welcher nur das Szenario ohne die Umgebung darstellt, nicht auf die harmonisierten Bauzonen, sondern auf die Modellzonen aggregiert wird.

4.2.4.3 Geschossflächenziffer, Überbauungsziffer und Baumassenziffer

Die Analyse der geschätzten Geschossflächenziffer, Überbauungsziffer und Baumassenziffer gestaltet sich relativ simpel, da diese bezogen auf die Parzelle ausgewertet werden (vergleiche Anhang B.6.9). Bei den Parzellen ausserhalb des Projektgebietes können für die Ist-Situation die zuvor aufbereiteten Daten direkt übernommen werden. Parzellen mit Überschneidung zum Projektgebiet werden entsprechend ausgeschlossen. Für die Parzellen innerhalb des Projektgebiets wird zunächst eine Verknüpfung mit den vom Modell generierten Gebäuden vorgenommen. Anschliessend werden die nötigen Grössen auf die Parzelle aggregiert und die drei Kennzahlen berechnet. Auf der einen Seite werden diese aggregierten Daten direkt als eigener Layer dargestellt. Auf der anderen Seite wird für die kombinierte Darstellung mit der Ist-Situation noch eine Mengenvereinigung vorgenommen.

4.2.4.4 Bauliche Dichte

Ähnlich wie bei der Auswertung der baulichen Dichtekennziffern gestaltet sich die Auswertung des überbauten Volumens pro Hektare relativ unkompliziert. Für die Situation innerhalb des Projektgebietes werden die vom parametrischen Modell generierten Gebäude direkt mit den Hektarzellen verschnitten und das Volumen aggregiert (vergleiche Anhang B.6.10). Zu beachten ist hier, dass in Zellen am Rand des Projektgebietes niedrigere Werte resultieren können, wenn die Gebäude der Ist-Bebauung im Umfeld nicht berücksichtigt werden. Für die kombinierte Darstellung werden die generierten Gebäude innerhalb des Projektperimeters mit den swissBUILDINGS3D Gebäuden ausserhalb des Perimeters zusammengefasst, welche ebenfalls mit den Hektarzellen verschnitten werden. Anschliessend findet eine Aggregation auf die Hektare statt.

4.2.4.5 Freiflächenanteil

Die Auswertung des Freiflächenanteils pro Hektare erfolgt ebenfalls ähnlich direkt wie bei der baulichen Dichte. Für die Parzellen des Projektgebietes werden die nicht von Gebäuden eingenommenen Flächen als Freifläche angesehen. Sowohl die Projektparzellen als auch die Grundrisse der Gebäude werden mit den Hektarzellen verschnitten und die Daten anschliessend anhand des Identifikators der Zelle miteinander und mit dem Hektarraster verknüpft. Bei der nachfolgenden Aggregation der Flächeninhalte wird die Fläche der Gebäude pro Zelle von den Flächen der Parzellen abgezogen, um den Freiraum zu erhalten (vergleiche Anhang B.6.11). Dieser wird noch ins Verhältnis zur Zellenfläche gesetzt, um einen Prozentwert für die Darstellung zu erzeugen. Wiederum ist bei Zellen am Rande des Projektgebietes Vorsicht bei der Interpretation der Werte geboten, da das Umfeld nicht betrachtet wird. Für die kombinierte Darstellung mit der Ist-Situation werden analog zur Analyse der Ist-Situation die bereits mit den Hektarzellen verschnittenen Flächen aus der Bodenbedeckung selektiert, die als Freifläche angesehen werden. Durch Verschnitt mit den Projektparzellen werden Flächen innerhalb des Projektperimeters ausgeschlossen, bevor die kombinierten Flächen aggregiert und ins Verhältnis zur Grundfläche gesetzt werden.

4.2.4.6 Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte

Die Auswertung der Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte bezogen auf das Projektgebiet erfolgt durch Umlegung der auf den generierten Häusern hinterlegten Werte für Bewohner und Beschäftigte auf die Hektarzellen. Dabei wird der Wert eines Gebäudes derjenigen Hektarzelle zugeschlagen, in der sich das Zentroid des Grundrisses befindet. Anschliessend werden die Werte auf die Hektarzelle aggregiert, wobei die Raumnutzer durch die Addition von Einwohnern und Beschäftigten abgeleitet werden (vergleiche Anhang B.6.12). Wie bei den anderen hektarbezogenen Auswertungen muss hier auf zu niedrige Werte am Rande des Projektgebietes geachtet werden. Die Kombination mit der Ist-Situation gestaltet sich wesentlich komplexer, da die bereits auf die Hektare bezogenen Werte anteilig umgelegt werden müssen. Als Kriterium dafür wurde der Anteil der Grundfläche der Gebäude der Ist-Bebauung ausserhalb des Projektperimeters verwendet. Zu dessen Bestimmung werden zunächst die Grundrisse der Gebäude aus der Amtlichen Vermessung mit den Hektarzellen verschnitten. Durch eine zweite Verschnitt-Operation mit den Projektparzellen wird bestimmt, welche Gebäudeteile sich innerhalb und welche ausserhalb des Perimeters befinden. Durch Aggregation auf die Hektarzelle wird der Anteil ausserhalb des Perimeters bestimmt. Dieser Anteil wird mit den STATPOP und STATENT Daten der Ist-Situation multipliziert. Das Resultat wird mit den Daten aus dem Projektperimeter addiert, um die kombinierte Darstellung zu erhalten. Man beachte, dass diese Methode der anteiligen Anrechnung keine exakten Werte liefert, sondern eine Näherung darstellt. Für genauere Werte

müssten die nicht aggregierten Daten der Statistiken herangezogen werden, welche aber nicht öffentlich verfügbar sind. Bei Vorliegen der Referenzpunkte für die Erfassung der STATPOP und STATENT Daten, welche als Grundlage für die räumliche Verortung und Aggregation der Daten dienen, könnte eventuell eine bessere Disaggregation und damit anteilige Anrechnung vorgenommen werden. Die Verwendung der Adresspunkte als Stellvertreter für die Haupteingänge der Gebäude könnten ein Ansatz für eine solche näherungsweise Disaggregation sein, welche aber aufgrund des damit verbundenen Aufwandes nicht weiter untersucht wurde.

4.2.4.7 Durchschnittliche Wohnfläche pro Einwohner

Die Auswertung der durchschnittlichen Wohnfläche pro Einwohner zeigt in mehrerlei Hinsicht Parallelen zur Auswertung der Raumnutzer-, Einwohner- und Beschäftigtendichte. Wenn nur das Szenario betrachtet wird, ist das Vorgehen analog, indem die generierten Gebäude anhand des Zentroids einer Hektare zugeordnet und die Werte anschliessend aggregiert werden. Vor der Aggregation werden aus der Geschossfläche und dem Anteil Arbeitsnutzung noch die Wohnfläche bestimmt, welche bei der Aggregation durch die Anzahl Bewohner geteilt wird (vergleiche Anhang B.6.13). Da im Modell die Anzahl Einwohner und Beschäftigte über den Flächenverbrauch pro Person berechnet werden, kommt es in Hektarzellen, die praktisch nur in einer Modellzone liegen, zu einer Annäherung des berechneten Wertes an den Parameterwert für den Platzverbrauch. Bei Randzellen ist wiederum Vorsicht geboten, da das Umfeld nicht berücksichtigt wird. Bei der kombinierten Darstellung von Szenario und Ist-Situation erfolgt wiederum eine anteilige Umlegung anhand der Gebäudeflächen ausserhalb des Projektperimeters. Die Bestimmung dieses Anteils erfolgt auf die gleiche Weise wie bei den Raumnutzerdichten. Der entsprechende Faktor wird mit den Ist-Werten aus GWS für die Wohnfläche und STATPOP für die Einwohner multipliziert. Die so berechneten Anteile werden zusammen mit den Werten aus dem Szenario aggregiert und daraus der kombinierte Wert für die Wohnfläche pro Einwohner bestimmt.

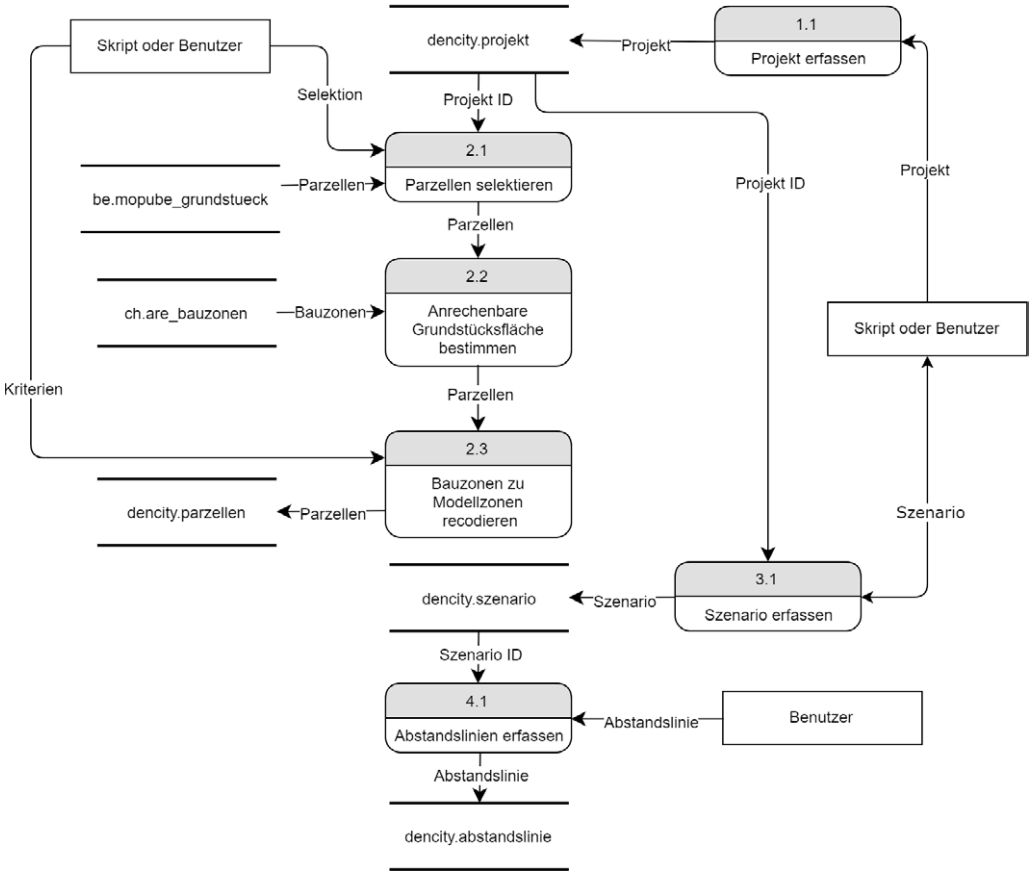
4.3 Parametrisches Modell

4.3.1 Übersicht

Das in Rhino Grasshopper implementierte parametrische Modell setzt das Change Model im Geodesign Prozess um. Der Grossteil des Modells ist mittels visueller Programmierung mit Grasshopper Komponenten umgesetzt. Dies beinhaltet Komponenten einiger Add-ons. Teile des Modells sind zudem in Python und C# Scripts umgesetzt, was vor allem bei der Integration von Grasshopper mit externen Systemteilen (Datenbank, R und QGIS) der Fall ist.

Von der Art her, wie die Daten im Modell verarbeitet werden, kann dieses in die Kategorie *Procedural Modelling* in der Taxonomie von Janssen und Stouffs (2015) eingeordnet werden. Im Modell werden zwar an verschiedenen Stellen Scripts mit imperativer Programmierung genutzt, welche auch Schleifen enthalten, diese dienen aber praktisch ausschliesslich der Iteration über die Eingabedaten. Ein Grenzfall ist die Generierung von Gebäudegrundrissen mit Hilfe von Quadtree Zerlegung der Bauflächen. Hier kommt Iteration mittels Rekursion zum Einsatz und es werden neue Geometrien generiert. Diese stehen allerdings in direkter Abhängigkeit zu den Inputwerten. Zudem könnte die Zerlegung in ähnlicher Form ebenfalls mit Grasshopper Komponenten und Zuhilfenahme von Erweiterungen wie Hoopsnake für iterative Prozesse umgesetzt werden.

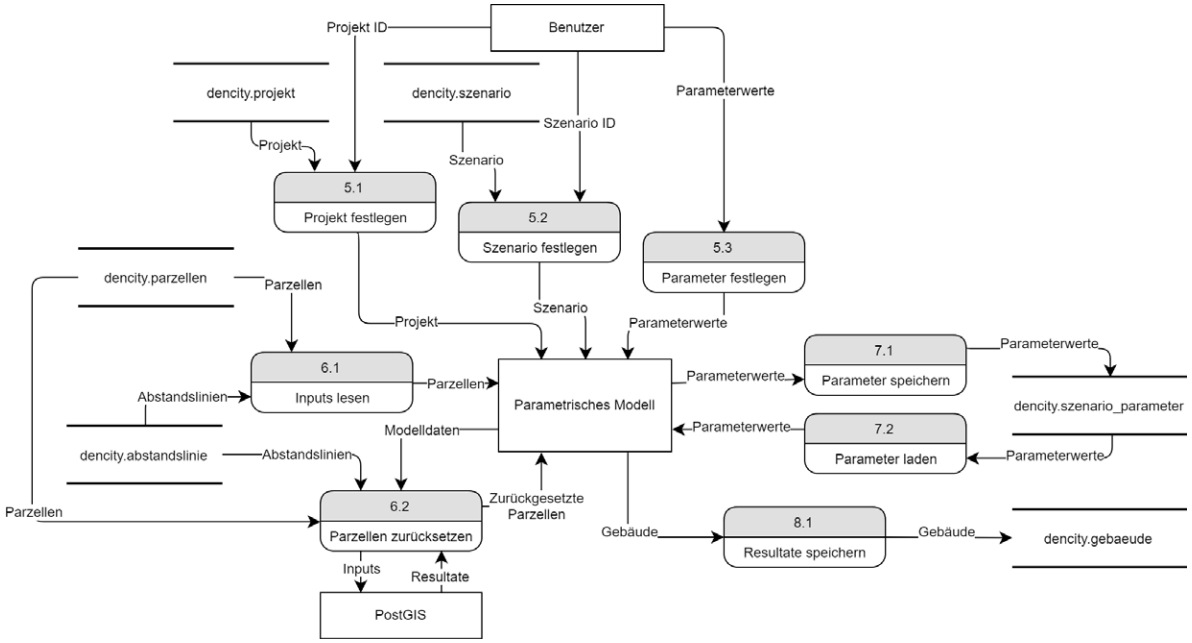
Aufgrund seiner Integration mit der zentralen PostGIS Datenbank und der daraus resultierenden Abhängigkeit setzt das parametrische Modell einige Aufbereitungsschritte ausserhalb Grasshopper voraus, bevor Verdichtungsvarianten als Output erzeugt werden können. Abbildung 11 zeigt den grundlegenden Datenfluss für diese vorbereitenden Schritte. Abgesehen von der manuellen Erfassung von Abstandslinien in QGIS wurden diese Schritte in der Arbeit mittels SQL Skripten automatisiert. Zunächst werden Projekte als Einträge in der entsprechenden Tabelle eingefügt. In einem zweiten Schritt werden die Parzellen aufbereitet, indem die Geometrien aus der Amtlichen Vermessung selektiert, in die Zieltabelle eingefügt und anschliessend die anrechenbare Grundstücksfläche gesetzt wird. Letzteres erfolgt über den Verschnitt mit den harmonisierten Bauzonen. Diese dienen zudem der Zuordnung der Modellzonen. Pro Projektgebiet wurde eine sinnvolle Abbildung der bestehenden Bauzonen auf die fünf verfügbaren Modellzonen vorgenommen, wobei ähnliche Zonen zusammengefasst wurden. In einem dritten Schritt werden Einträge für die Szenarien in die Datenbank eingefügt. Wo vorgesehen wurden anschliessend die Abstandslinien für die Szenarien in QGIS erfasst.



Quelle: eigene Darstellung

Abbildung 11: Datenfluss Datenaufbereitung parametrisches Modell

Abbildung 12 zeigt die Schritte und den vereinfachten Datenfluss für die Verwendung des Modells innerhalb von Grasshopper, nachdem die vorbereitenden Schritte abgeschlossen sind. Das Modell selbst ist dabei als externe Entität dargestellt, welche die detaillierten Verarbeitungsschritte kapselt. Durch Input des Benutzers wird im Modell nach der Auswahl eines Projektes zuerst die Liste der verfügbaren Szenarien geladen. Anschliessend kann ein Szenario gewählt werden. Für dieses werden die Parameter eingestellt, was entweder durch manuelle Eingabe oder das Laden eines gespeicherten Szenarios geschieht. Das Modell liest selbstständig die zum Projekt gehörenden Parzellen ein und verarbeitet diese, wobei die Datenbank auch für das Zurücksetzen der Parzellenflächen angesprochen wird. Wenn der Benutzer die eingestellten Parameter für ein Szenario speichern will, kann er eine entsprechende Funktion auslösen. Wenn er mit den generierten Resultaten zufrieden ist, kann er schliesslich deren Speicherung in der Datenbank veranlassen.

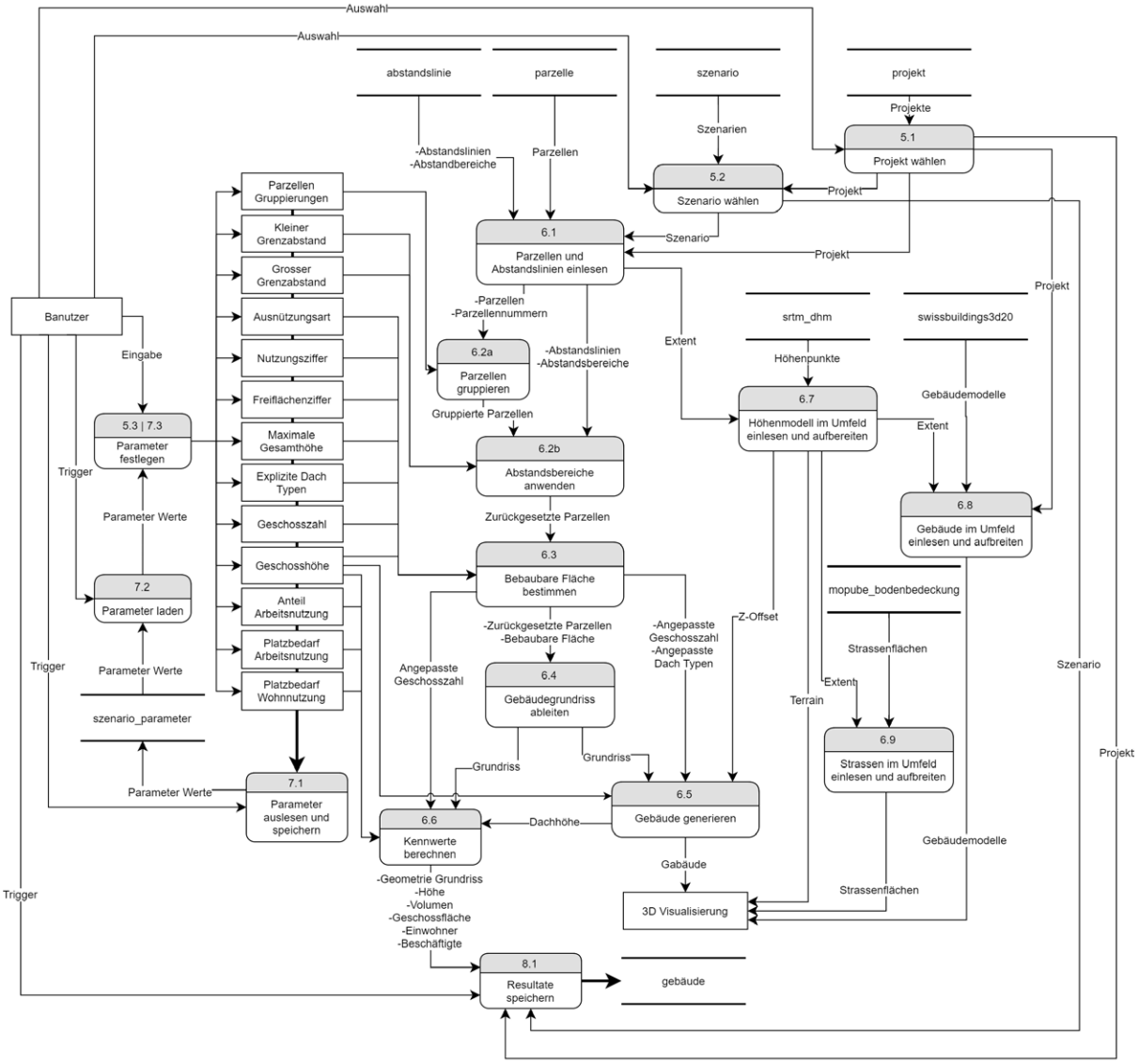


Quelle: eigene Darstellung

Abbildung 12: Datenfluss Nutzung parametrisches Modell

Die Abbildung 13 zeigt den Datenfluss des parametrischen Modells in ausführlicherer Form mit den wichtigsten internen Verarbeitungsschritten. Die Nummerierung der Operationen ist dabei mit der externen Sicht in Abbildung 12 abgestimmt. Die gesetzten Parameter sowie die eingelesenen Parzellenflächen und Abstandslinien stellen die Inputs des Modells dar. Bei der Generierung der Varianten können eine Reihe von Schritten unterschieden werden, welche in den nachfolgenden Abschnitten detaillierter behandelt werden. Zunächst werden die Abstandsbereiche auf die Parzellen angewendet (6.2). Auf dieser Basis sowie der weiteren Parameter wird die Grösse der maximal bebaubaren Fläche bestimmt (6.3). In diesem Schritt werden zudem die verschiedenen Parameter entsprechend der in Kapitel 3.5.4 beschriebenen Beziehungen aufeinander abgestimmt und einige Werte mitunter angepasst (z.B. Geschosszahl und Dach Typ). Die bebaubare Fläche und die um die Abstandsbereiche zurückgesetzten Parzellen dienen als Input zur Ableitung der Gebäudegrundrisse (6.4). Auf Basis der Grundrisse und der anderen Parameter können die Gebäudemodelle erzeugt werden (6.5). Die Grundrisse und Gebäude dienen wiederum als Input für die Berechnung der Kennzahlen des Gebäudes (6.6), welche der Benutzer zusammen mit der Geometrie der Grundrisse in der Datenbank speichern kann (8.1). Zusätzlich zu diesen Verarbeitungsschritten werden im Modell Daten zur Umgebung des Projektgebietes in Form eines Höhenmodells, Gebäuden und Strassen eingelesen (6.7 bis 6.9), welche allerdings nur der Visualisierung dienen.

In Abbildung 12 und Abbildung 13 nicht gezeigt sind Schritte beziehungsweise Datenflüsse für zusätzliche Funktionen. Dazu gehören die Generierung von Auswertungen und die dreidimensionale Visualisierung von Kennzahlen zur Ist-Situation, die in den nachfolgenden Kapiteln ebenfalls kurz behandelt werden.

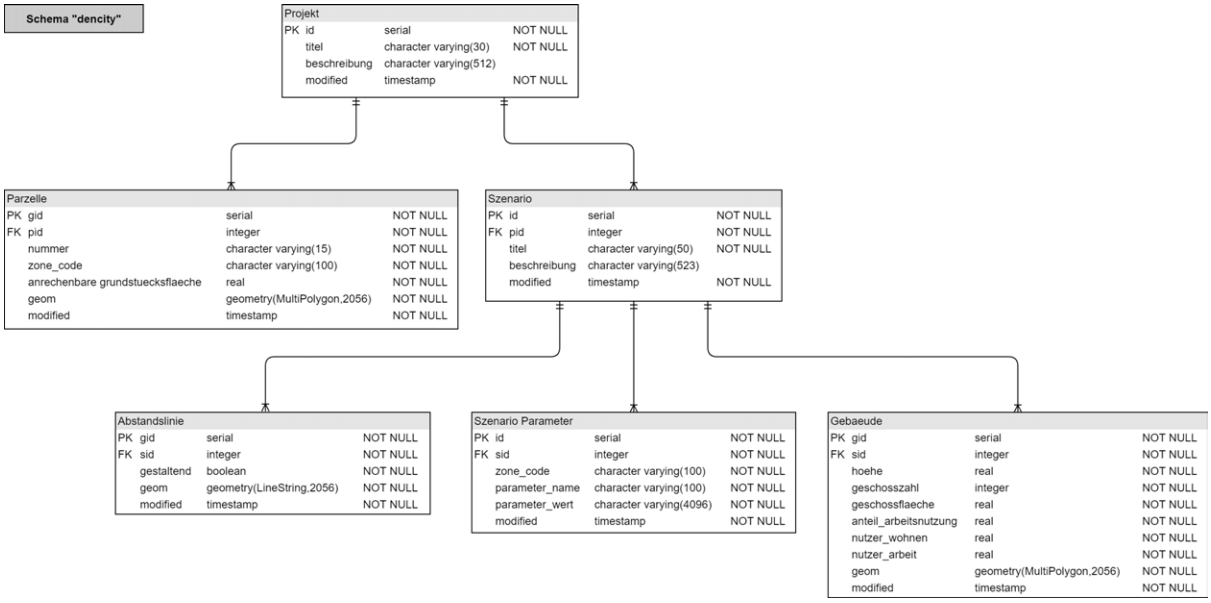


Quelle: eigene Darstellung

Abbildung 13: Vereinfachte Übersicht des Datenflusses des parametrischen Modells

4.3.2 Datenmodell

Abbildung 14 zeigt eine Übersicht des Datenmodells für die Speicherung der Daten des parametrischen Modells in der zentralen PostgreSQL Datenbank. Die Tabellen sind in einem eigenen Schema *dencity* innerhalb der Datenbank organisiert. Die folgenden Unterabschnitte geben eine kurze Übersicht zu den einzelnen Entitäten. Allen Tabellen ist gemein, dass diese einen nicht sprechenden Primärschlüssel in Form einer *id* (beziehungsweise *gid* bei Entitäten mit Geometrie) haben, deren Wert jeweils als fortlaufende Sequenz von Zahlen automatisch vergeben wird. Ebenfalls weisen alle Entitäten mit dem Attribut *modified* einen Zeitstempel auf, der das Datum der letzten Änderung angibt. Der Wert des Zeitstempels wird bei INSERTS mittels eines DEFAULT Werts automatisch gesetzt und bei UPDATES durch einen Trigger automatisch aktualisiert, so dass sich der Benutzer nicht um das Attribut kümmern muss. Die Tabellen des physischen Datenmodells nutzen verschiedene Foreign Key und Uniqueness Constraints zur Sicherstellung der (referentiellen) Integrität der Daten.



Quelle: eigene Darstellung

Abbildung 14: Datenmodell parametrisches Modell

4.3.2.1 Projekt

Das *Projekt* ist die zentrale Entität zur Abbildung von Untersuchungsgebieten, von der direkt oder indirekt alle anderen Entitäten abhängen. Zu einem Projekt gehören eine Reihe von Parzellen, die das Projektgebiet definieren, und Szenarien, die mit ihren Parametern sowie resultierenden Gebäuden verschiedene Verdichtungsvarianten im Projektgebiet repräsentieren. Auf dem Projekt selbst kann nur ein Titel und eine kurze Beschreibung gespeichert werden. Diese sind direkt in der Datenbank zu erfassen, wobei entweder SQL INSERT Statements oder ein geeigneter Datenbankclient verwendet werden muss.

4.3.2.2 Parzelle

Jeder Eintrag in der Tabelle *Parzelle* ist genau einem Projekt zugeordnet. Mit ihrer Geometrie definiert die Parzelle den zentralen räumlichen Input für die Erzeugung der Varianten mittels des parametrischen Modells. Zusammengenommen legen die zugeordneten Parzellen ausserdem den Projektperimeter fest. Für jede Parzelle ist eine Nummer als Identifikator anzugeben, welche innerhalb eines Projekts eindeutig sein muss. Diese Nummer wird insbesondere zur Gruppierung von Parzellen, zur Parametrisierung von Dachformen sowie für die Steuerung des Datenflusses im Modell verwendet. Zudem ist für jede Parzelle der Code der Modellzone anzugeben, zu der sie gehört. Ebenso muss die anrechenbare Grundstücksfläche hinterlegt werden. Die Eingabe der Parzellen mit einer Geometrie vom Typ MultiPolygon hat direkt in der Datenbank zu erfolgen. Während eine manuelle Erfassung mittels QGIS möglich wäre, empfiehlt sich die Ableitung aus den Daten der Amtlichen Vermessung zu den Grundstücken, wodurch die Geometrien übernommen werden können. Für das Setzen der anrechenbaren Grundstücksfläche kann ein Overlay mit den bestehenden Bauzonen als Input verwendet werden. Auf diese Art können Projekte und Szenarien mit grosser Nähe zur Ist-Situation umgesetzt werden. Es besteht zudem die Möglichkeit durch Bearbeitung der Geometrien bestehender Grundstücke (z.B. durch Dissolve oder durch Aufteilung) hypothetische Grundlagen zu definieren.

4.3.2.3 Szenario

Die Entität *Szenario* ist jeweils einem einzelnen Projekt zugeordnet. Sie stellt ein Sammelgefäss für die Inputs und Outputs des parametrischen Modells dar, wodurch jedes Szenario schlussendlich eine Verdichtungsvariante repräsentiert. Die Inputs werden in Form von Parameterwerten und Abstandslinien definiert, während die Outputs in Form der generierten Gebäude gespeichert werden. Auf dem Szenario selbst kann wie beim Projekt nur ein einfacher Titel und eine kurze Beschreibung hinterlegt werden. Das Anlegen von Szenario Einträgen hat direkt in der Datenbank zu geschehen.

4.3.2.4 Abstandslinie

Die *Abstandslinie* ist der zweite räumliche Input für das parametrische Modell. Die Geometrie einer Abstandslinie definiert einen Bereich auf einer oder mehreren Parzellen, in welchem die regulären Grenzabstände überschrieben werden. Als Input für das Modell ist jede Abstandslinie genau einem Szenario zugeordnet. Die Erfassung der Abstandslinien mit einer Geometrie vom Typ Polyline geschieht typischerweise manuell in QGIS. Hierfür kann im parametrischen Modell für ein Szenario ein einfaches QGIS Projekt mit den notwendigen Layern generiert werden. Bei der Erfassung der Abstandslinien ist auf die Richtung zu achten. Im Modell werden die Abstandslinien mit den Parzellengrenzen verschnitten und die Parzellenbereiche rechts der Linie (gesehen vom Beginn in Richtung Ende) als nicht bebaubar ausgeschieden. Entsprechend ist wichtig, dass die Geometrien der Abstandslinien die betreffenden Parzellengrenzen schneiden oder berühren. Die Abstandslinien haben im Modell im Moment keine direkte gestalterische Wirkung (d.h. dass generierte Gebäude zwingend die Abstandslinie als Teil der projizierten Fassadenlinie einschliessen beziehungsweise an diese heranreichen), eine Spalte für einen entsprechenden Parameter ist im Datenmodell aber bereits vorgesehen.

4.3.2.5 Szenario Parameter

Die Tabelle *Szenario Parameter* dient der Speicherung von Parameterwerten, die als Input für das parametrische Modell verwendet werden. Im Kern funktioniert die Speicherung nach dem Prinzip eines Key-Value Speichers. Die Spalte *parameter_name* repräsentiert den Key, dessen Wert sich mit der Bezeichnung eines Parameters im Modell decken muss. Die Spalte *parameter_wert* entspricht hingegen dem Value, der als Text gespeichert wird. Das parametrische Modell ist für die Konversion des Texts in den richtigen Datentyp zuständig. Zudem ist jedem Eintrag ein Zonen Code zugeordnet, da das parametrische Modell mehrere Modellzonen mit unterschiedlichen Parametern unterstützt. Die Kombination von *zone_code* und *parameter_name* muss deshalb innerhalb eines Szenarios eindeutig sein. Per Konvention wird die Kombination dieser beiden Werte auch für die Identifikation der GUI Inputs im parametrischen Modell verwendet, wenn ein Szenario geladen wird. Das Anlegen der Einträge in dieser Tabelle geschieht im Normalfall durch Speicherung der Parameterwerte eines Szenarios aus dem parametrischen Modell heraus. Der verwendete Ansatz mit einem Key-Value Speicher und einem generischen Mechanismus zum Laden bzw. Speichern der Parameterwerte erlaubt, das Modell um neue Parameter zu erweitern, ohne das Datenmodell anpassen zu müssen.

4.3.2.6 Gebäude

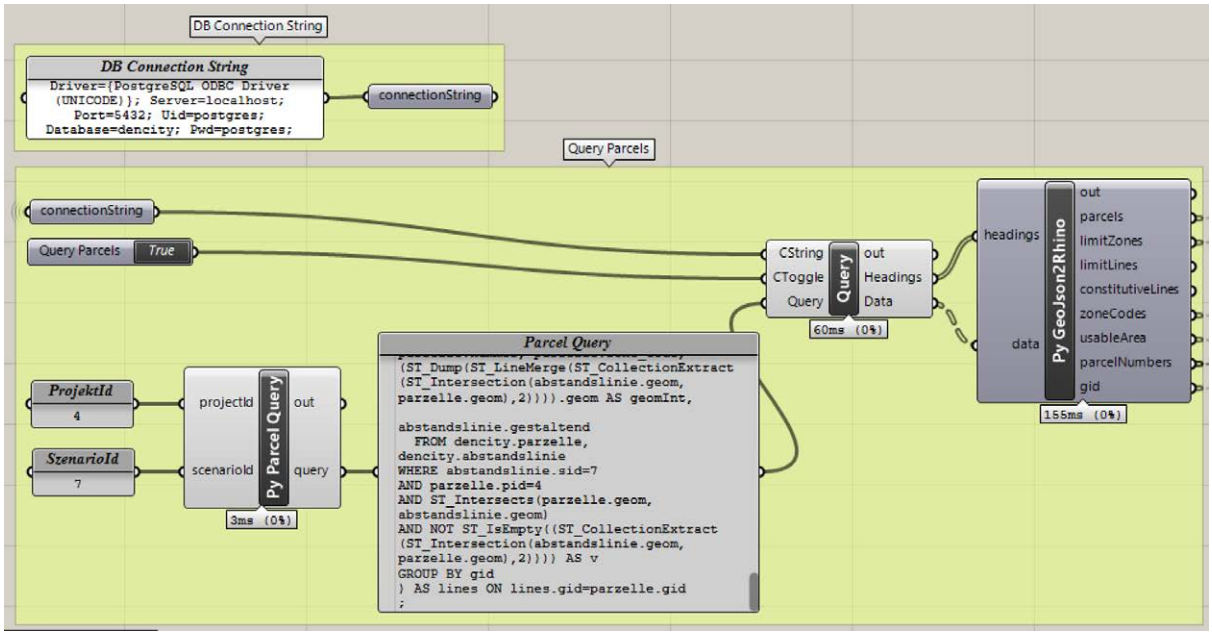
Die *Gebäude* repräsentieren die Resultate des parametrischen Modells. Jedes Gebäude ist einem Szenario zugeordnet. Gespeichert wird nicht die dreidimensionale Geometrie, die vom Modell generiert wird, sondern nur der zweidimensionale Grundriss des Gebäudes. Zusätzlich werden zusammen mit der Geometrie mehrere Werte wie Gebäudehöhe, Geschosszahl, Geschossfläche, Anzahl Einwohner und Anzahl Beschäftigte gespeichert, welche sich aus dem parametrischen Modell ergeben. Die Einträge in dieser Tabelle entstehen durch Speicherung der Resultate im parametrischen Modell. In Kombination mit den Parzellen des zugehörigen Projektes bilden sie die Grundlage zur Analyse des Szenarios.

4.3.3 Datenbankanbindung

Abgesehen von den manuellen Eingaben für Parameter werden alle Inputdaten für das parametrische Modell aus der zentralen PostgreSQL Datenbank gelesen. Die eigentlichen Outputs des Modells sowie die Parameterwerte für Szenarien werden ebenfalls in der Datenbank gespeichert. Für den Zugriff auf die Datenbank wird das Slingshot! Add-on (Miller 2015) verwendet. Dieses ermöglicht den Zugriff auf MySQL, ODBC oder OLE DB Verbindungen in Grasshopper. Im parametrischen Modell werden ODBC Verbindungen für den Datenbankzugriff verwendet. Slingshot! umfasst eine Reihe von Komponenten für Aufgaben im Zusammenhang mit Datenbankoperationen. Primär werden im parametrischen Modell die *Query* Komponente, welche die Resultate von SQL SELECT Statements als Grasshopper Datenstrukturen bereitstellt, und die *Command* Komponente für die Ausführung beliebiger SQL Befehle verwendet. Die nötigten SQL Statements werden in einfachen Fällen mittels der *QuerySFW* Komponente und in komplexeren Fällen mittels der *GH Python* Komponente erzeugt. Die benötigen ODBC Connection Strings werden manuell mittels eines Text Panels konfiguriert, da die Connection Strings, die mit den Slingshot! Konfigurationskomponenten für PostgreSQL Datenbanken generiert werden, mit aktuellen ODBC Treibern nicht funktionieren. Nachteil dieser Lösung ist, dass das Passwort im Klartext im

Connection String steht und dieses vor der Weitergabe der Grasshopper Definition entfernt werden muss.

Grob kann bei der Datenbankanbindung zwischen dem Lesen und Speichern von Daten unterschieden werden. Für beide Fälle gibt es jeweils ein Grundmuster aus mehreren Komponenten. Abhängig vom Kontext werden diese Muster an mehreren Stellen im Modell mit leichten Variationen eingesetzt. Unterschiede bestehen vor allem in der Ausgestaltung der Generierung der SQL Ausdrücke und in der Behandlung der Outputs. In Abbildung 15 wird das Grundmuster für das Lesen von Daten aus der Datenbank am Beispiel der Abfrage der Parzellendaten gezeigt. Eine solche Gruppe besteht jeweils aus Komponenten für die Erzeugung einer SQL SELECT Abfrage, der *Query* Komponente für deren Ausführung und aus einer *GH Python* Komponente für die Umwandlung sowie Ausgabe der Resultate zur weiteren Verarbeitung. Im gezeigten Beispiel wird eine *GH Python* Komponente für die Erzeugung der SELECT Abfrage verwendet (vergleiche Anhang B.4.2 für Details), da diese mit verschiedenen UNIONS und Sub-SELECTS zu komplex für die Slingshot! Komponenten ist. Das erzeugte SELECT wird in die *Query* Komponente gespiesen, welche als Input neben einer Abfrage als String, Die Verbindungsinformationen als Connection String und einen Boolean Wert als Trigger verlangt. Die Abfrage wird nur ausgeführt, wenn der Trigger auf TRUE gesetzt ist. Dies erlaubt es, gewisse Abfragen nur auszuführen, wenn der Benutzer dies explizit auslöst (z.B. um die gespeicherten Parameterwerte für ein Szenario zu laden). Die Outputs der *Query* Komponente in Form einer Liste mit den Namen der abgefragten Spalten und einem DataTree mit den eigentlichen Daten werden an eine *GH Python* Komponente übergeben, welche eine Umwandlung in Datenstrukturen und Datentypen für die weitere Verarbeitung umsetzt (vergleiche Anhang B.4.3). Die Resultate werden als Listen oder Trees ausgegeben, wobei gleichzeitig eine Typenkonversion stattfinden kann. Im einfachen Fall sind dies simple Konversionen wie z.B. von String nach Integer. Viel wichtiger ist aber die Konversion der räumlichen Datentypen in die Rhino internen Geometrietypen.



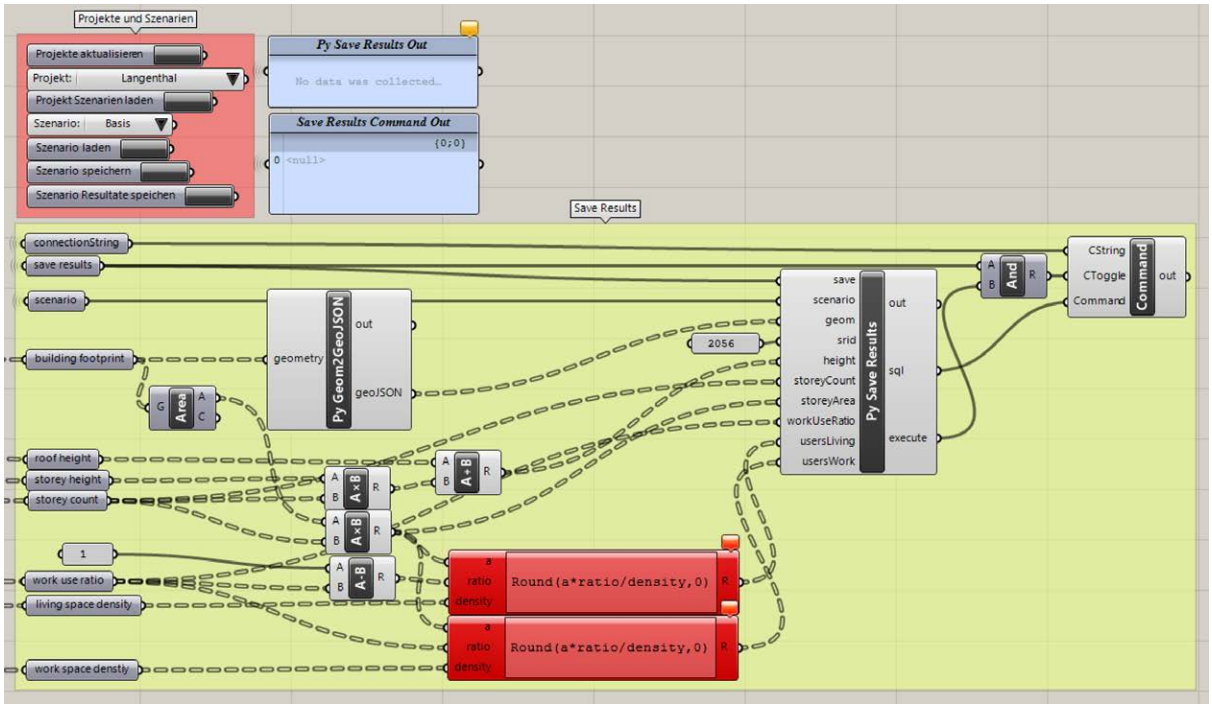
Quelle: eigene Darstellung

Abbildung 15: Lesen der Parzellendaten aus der Datenbank

Da weder Slingshot! noch Grasshopper nativ mit den Geometrie Datentypen von PostGIS umgehen können, müssen diese für die Integration in das Modell aufbereitet werden. Als standardisiertes Austauschformat wird GeoJSON (Butler *et al.* 2016) verwendet, das von PostGIS unterstützt und in Python einfach verarbeitet werden kann. Bei der Abfrage auf der Datenbank wird die eigentliche Geometrie mittels der Funktion ST_AsGeoJSON() in GeoJSON konvertiert. Der resultierende String wird als Inhalt des «geometry» Attributes in ein umgebendes «Feature» Objekt verpackt. Auf Seiten von Grasshopper werden GeoJSON Daten mittels eines Python Scripts in Rhino Geometrietypen konvertiert. Dieses Script ist eine Version des GeoJson2Rhino.py Scripts (Golder 2010) aus dem Local Code Projekt (de Monchaux *et al.* 2010), bei der das Handling von Geometry Collections und Multi Geometrie Objekten

ergänzt und verbessert wurden. Die Entsprechung der GeoJSON Geometrie Objekte und der Rhino internen Geometrietypen ist nicht eins zu eins. Einfache Point Objekte werden in Point3d Geometrien und LineString Objekte in Curve Geometrien übersetzt. Polygon Objekte können nicht direkt abgebildet werden, da Rhino kein genaues Äquivalent kennt. Vielmehr werden die Ringe der Polygone als geschlossene Curve Geometrien abgebildet, welche am ehesten dem Konzept eines geschlossenen Polygonzugs entsprechen. Wenn ein Polygon in GeoJSON mehr als einen Ring aufweist (d.h. Löcher vorhanden sind), wird eine Liste mit mehreren Curve Objekten ausgegeben. Um tatsächlich eine Oberfläche mit Löchern zu erhalten, müssen diese Kurven im Anschluss in BREP (Boundary Representation) Surfaces konvertiert werden. Zudem werden Multi Geometrie Objekte und Collection Objekte als (allenfalls geschachtelte) Listen von Geometrien abgebildet, da Rhino direkte Äquivalente kennt.

Abbildung 16 zeigt anhand des Beispiels der Szenario Resultate das Vorgehen zur Speicherung von Daten. Hier kann grob zwischen Komponenten zur Aufbereitung der Daten, Komponenten zur Erzeugung der nötigen SQL Befehle und der *Command* Komponente zu deren Ausführung unterschieden werden. Die Art der Aufbereitung der Daten hängt von der jeweiligen Situation ab. Die Inputs aus der Aufbereitung werden in eine *GH Python* Komponente gespiesen, welche die nötigen SQL Befehle für die Speicherung der Daten erzeugt. Abhängig von der Situation ist dies eine Mischung aus DELETE, INSERT und UPDATE Statements. Die erzeugten SQL Befehle werden schliesslich in eine *Command* Komponente für die Ausführung auf der Datenbank gespiesen. Neben den Befehlen in Form eines Strings benötigt diese Komponente als Input einen Connection String und einen Boolean Wert als Trigger. Im Falle der Speicherung von Komponenten wird der Trigger immer nur konditional auf Eingabe des Benutzers ausgelöst (siehe «Szenario Resultate Speichern» Button in Abbildung 16). Im gezeigten Beispiel wird durch Verkettung mit der *GH Python* Komponente zur Erzeugung der SQL Befehle zudem sichergestellt, dass die Ausführung erst erfolgt, wenn die Generierung der Befehle abgeschlossen ist.



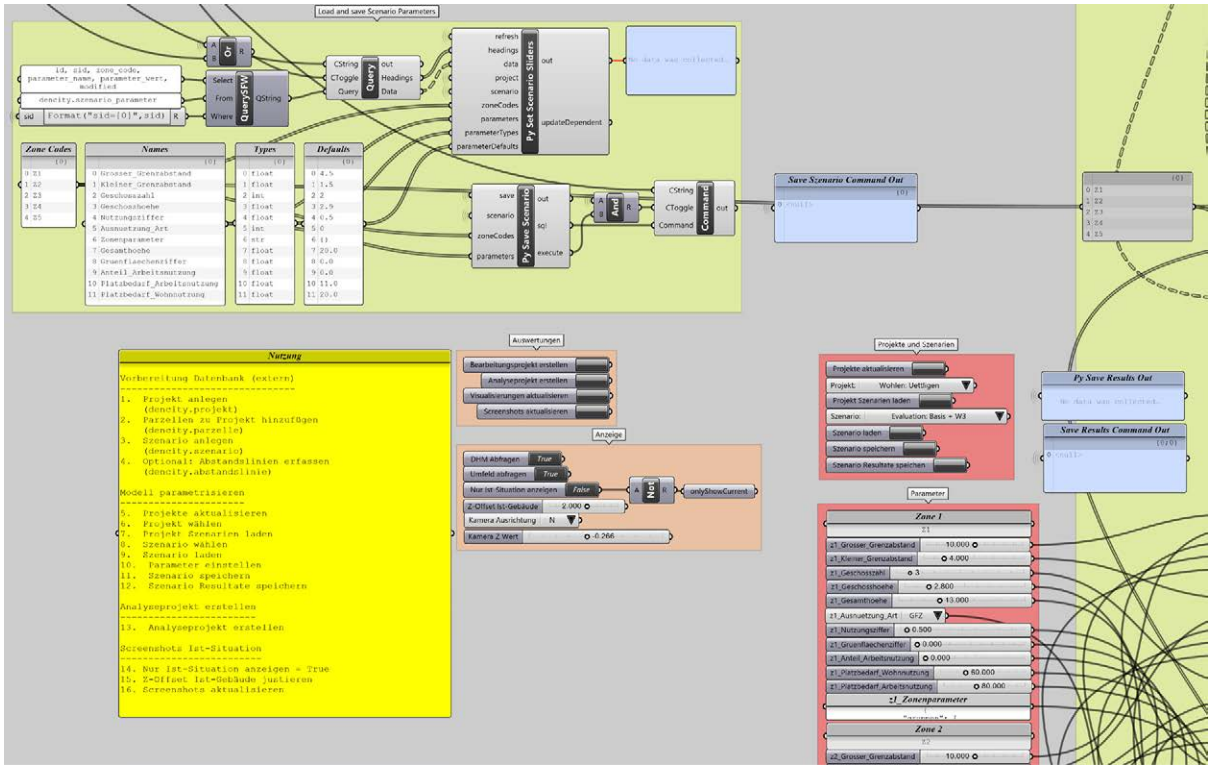
Quelle: eigene Darstellung

Abbildung 16: Speichern der Szenario Resultate in die Datenbank

4.3.4 Parameter

Die Parameter für das Modell werden vorrangig manuell über Parameter Input Komponenten im Grasshopper Modell festgelegt. Unten links in Abbildung 17 ist in der rot hinterlegten *Parameter* Gruppe ein Ausschnitt dieser Inputs für die erste Modellzone zu sehen. Bevor die Parameter eingestellt werden, müssen erst ein Projekt und anschliessend ein Szenario ausgewählt sowie geladen werden. Beim Laden eines Szenarios werden die in der Datenbank gespeicherten Parameterwerte abgefragt und auf die Inputs angewendet. Die hierfür zuständigen Komponenten sind oben rechts in Abbildung 17 in Form

der *Load and save Scenario Parameters* Gruppe zu sehen. Wenn in der Datenbank keine Werte gefunden werden, so werden auf den Inputs auf Standardwerte gesetzt, welche konfiguriert werden können.



Quelle: eigene Darstellung

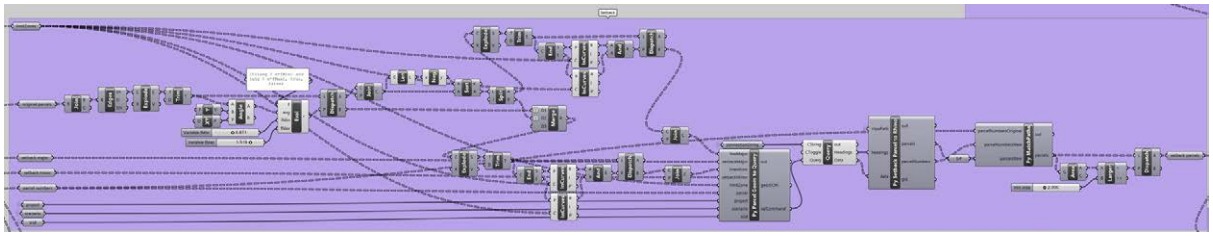
Abbildung 17: Parametrisierung in Grasshopper

Es ist keine direkte Verbindung zwischen den Komponenten zum Laden der Werte und den Inputs nötig, da die Werte auf den Input Komponenten programmatisch gesetzt werden. Der Code zum Laden und Speichern der Parameter ist generisch ausgelegt und wird durch eine Reihe von Konfigurationen gesteuert, die in der *Load and save Scenario Parameters* Gruppe in Form von Text Panels unten links zu sehen sind. Eine erste Konfiguration steuert die Anzahl der Modellzonen, indem deren Codes gelistet werden («Zone Codes»). Eine zweite Konfiguration legt die Namen der Parameter fest («Names»). Zwei weitere Konfigurationen bestimmen die Datentypen («Types») und Standardwerte («Defaults») der Parameter.

Die Namen der Parameter dienen bei der Speicherung in der Datenbank zudem als Schlüssel für die Key-Value Logik in der Tabelle *dencity.szenario_parameter*. In Kombination mit dem vorangestellten Code der Modellzone entsprechen die Parameternamen ausserdem dem Namen der Input Komponenten, was beim Laden und Speichern der Parameterwerte für den Abgleich genutzt wird. Solange die Parameter Namen eindeutig sind, kann deshalb das Modell durch Anpassen der Konfigurationen um neue Parameter ergänzt werden, ohne den Code oder das Datenmodell der Datenbank anpassen zu müssen.

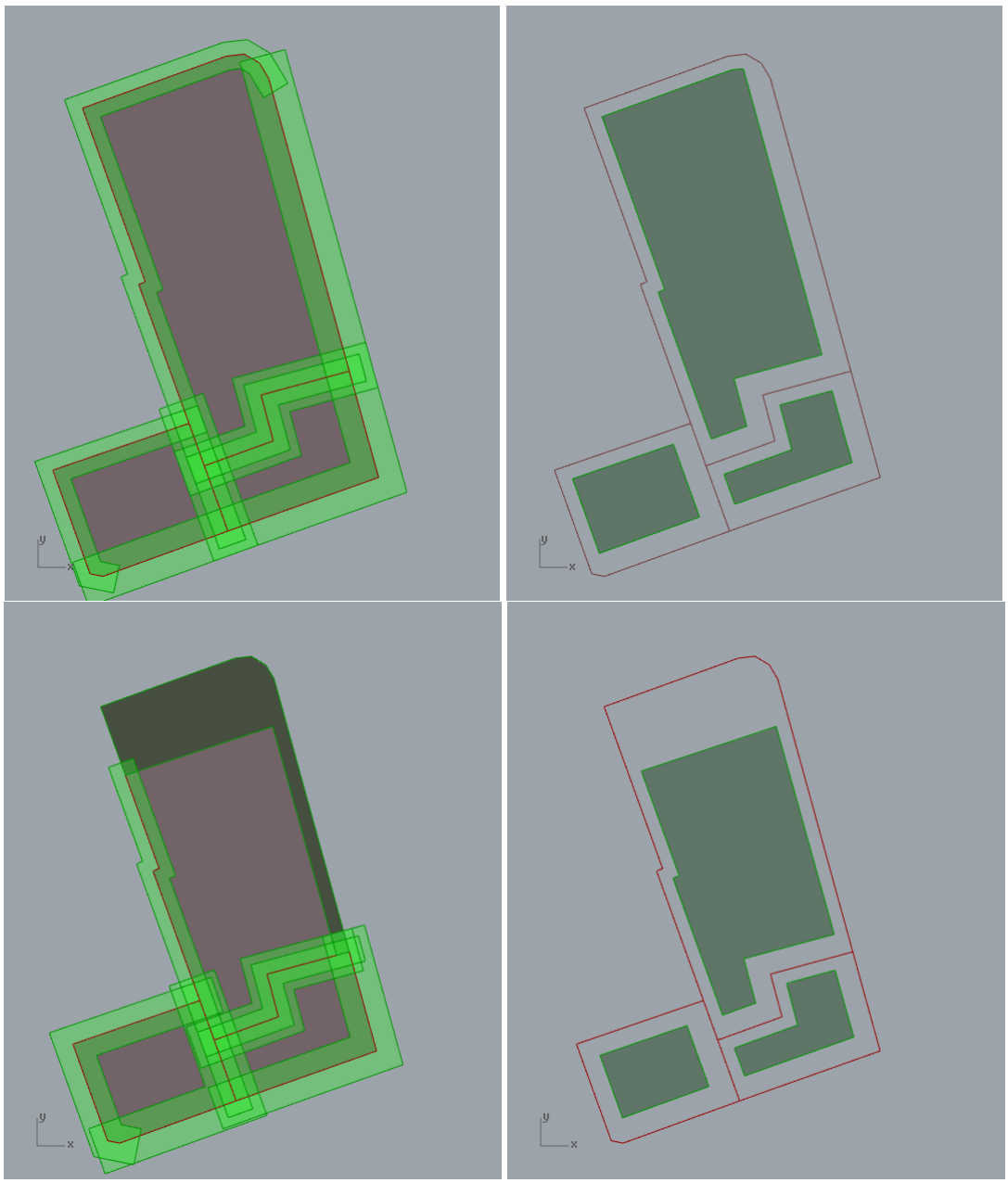
4.3.5 Ableitung bebaubare Parzellenfläche

Nach der Parametrisierung und dem Einlesen der Inputs werden zuerst die potentiell bebaubaren Parzellenflächen abgeleitet. Dies beinhaltet auf der einen Seite die Anwendung der Abstandsbereiche auf die Parzellen und auf der anderen Seite die Berechnung der Fläche, die gemäss Ausnützung tatsächlich überbaut werden darf. Abbildung 18 zeigt die Definition für das Zurücksetzen der Parzellenflächen um die Abstandsbereiche und Abbildung 19 illustriert deren Auswirkung.



Quelle: eigene Darstellung

Abbildung 18: Grasshopper Definition zum Zurücksetzen der Parzellenflächen



Quelle: eigene Darstellung

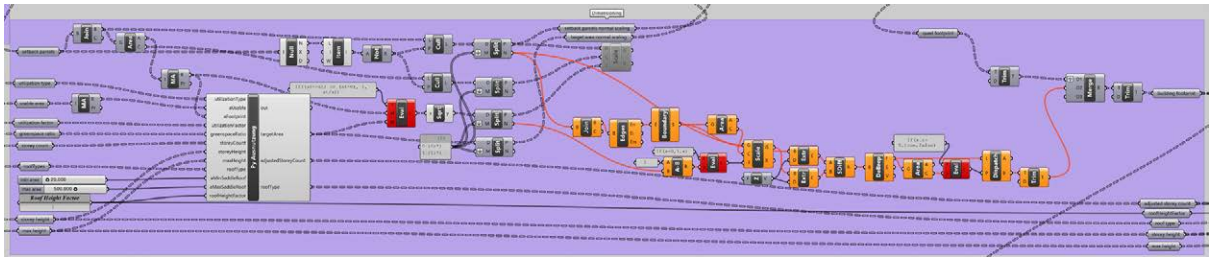
Abbildung 19: Auswirkung von Abstandsbereichen mit grossem und kleinem Grenzabstand (o.) sowie mit Abstandslinie (u.)

Die Abstandsbereiche werden durch die kleinen und grossen Grenzabstände sowie durch die Abstandslinien definiert. Zunächst werden die Parzellengrenzen unterteilt und diejenigen Grenzabschnitte mit einer Ausrichtung nach Osten bis Südwesten bestimmt. Auf den längsten dieser Abschnitte wird der grosse Grenzabstand und auf die restlichen der kleine Grenzabstand angewendet. Dies erfolgt durch ein Buffering mit dem jeweiligen Abstand auf die Grenzabschnitte und eine anschliessende Differenzbildung mit der originalen Parzellenfläche. In Abbildung 19 sind oben links die originalen Parzellenflächen mit überlagerten Grenzabständen und oben rechts die zurückgesetzten Parzellenflächen illustriert. Diese Fläche wird von der Geometrie her als der potentielle Baubereich angesehen. Durch vorherige Anwendung der Grenzabstände wird sichergestellt, dass jegliches Gebäude innerhalb dieser Fläche die nötigen Abstände einhält. Durch Definition von Abstandslinien können die Grenzabstände übersteuert werden. Auf Grenzabschnitte innerhalb der von den Abstandslinien abgetrennten Bereich werden keine Grenzabstände angewendet. Stattdessen wird der abgetrennte Bereich von der Parzellenfläche abgezogen, was im unteren Teil der Abbildung 19 illustriert ist. Die Parzelle oben rechts weist hier eine Abstandslinie auf, welche gegen oben und zur rechten Seite hin Abstände definiert, die klar von den grossen und kleinen Grenzabständen abweichen.

Speziell anzumerken ist, dass die Differenzbildung beim Zurücksetzen der Parzellenflächen nicht in Grasshopper, sondern in der Datenbank erfolgt. Versuche zur Umsetzung der Operation in Grasshopper haben verschiedene Probleme gezeigt. Differenzoperationen mit geschlossenen, planaren Kurven haben zuverlässig funktioniert, schliessen aber Parzellenflächen mit Löchern aus. Für die Repräsentation solcher Flächen werden Boundary Representation (BREP) Surfaces benötigt. Differenz Operationen auf planaren BREP Surfaces werden allerdings nicht unterstützt. Ein Workaround mit Extrusion der Surfaces, anschliessender Differenzbildung mit den resultierenden Volumen (Solid Difference) und erneuten Umwandlung in planare Surface Geometrien haben sich als unzuverlässig erwiesen. Diese Operationen liefern fehlerhafte Ergebnisse, wenn sich mehr als zwei Abstandsbereiche überschneiden oder die Parzelle von den Bereichen komplett bedeckt wird. Aus diesem Grund werden die Buffer und Differenz Operationen mittels PostGIS ausgeführt. Aus den abgeleiteten Grenzabschnitten und Abständen wird hierfür eine SQL Abfrage generiert (vergleiche Anhang B.4.5), die auf der Datenbank ausgeführt wird. Die resultierenden Geometrien der zurückgesetzten Parzellen werden dann wieder in Rhino Geometrien umgesetzt und in die richtige Reihenfolge gebracht. Trotz des Umwegs über die Datenbank, welche auf dem gleichen Computer wie Grasshopper lief, konnte bei der Delegation des Zurücksetzens an PostGIS sogar eine leicht bessere Performance als bei den Grasshopper internen Operationen erreicht werden. Zudem funktionieren bei dieser Lösung die Differenzoperationen zuverlässig, unabhängig von der Überlappung der Grenzbereiche und Parzellenflächen.

Man beachte, dass mehrere Parzellenflächen zu so genannten Gruppen zusammengefasst werden können. Dies hat durch den Benutzer mittels expliziter Konfiguration in den Zonen Parametern zu erfolgen. Umgesetzt wird dies durch eine JSON basierte Konfiguration, wie sie in Abbildung 21 auf Seite 59 gezeigt wird. Eine Gruppierung hat zur Folge, dass mehrere Parzellen bei der Anwendung der Grenzabstände und der Generierung der Gebäudegrundrisse wie eine einzelne Parzelle behandelt werden. Beim Zurücksetzen der Parzellen werden die Grenzabstände entsprechend nur auf die äussere Grenze der Gruppe angewendet.

Als zweiter Faktor zur Ableitung der bebaubaren Parzellenfläche muss anhand der weiteren Parameter wie anrechenbarer Grundstücksfläche, Art der Ausnützung, Nutzungsziffer, Geschosszahl und Maximalhöhe die Grösse der Fläche berechnet werden, welche tatsächlich bebaut werden darf. Abbildung 20 zeigt die Definition in Grasshopper, welche hierfür zuständig ist. Die eigentliche Berechnung ist in Python implementiert («Py Ausnutzung») und setzt die Zusammenhänge der Modelldefinition aus Kapitel 3.5.4 in Code um (vergleiche Anhang B.4.6). Der Output ist die Fläche, welche der Gebäudegrundriss maximal einnehmen darf. Zudem werden allenfalls angepasste Werte für die Geschosszahl und den Dach Typ ausgegeben, wenn diese infolge der gegebenen Einschränkungen angepasst werden mussten. Diese Outputs dienen als Parameter für die nachfolgende Generierung der Gebäude.



Quelle: eigene Darstellung

Abbildung 20: Grasshopper Definition zur Dimensionierung der bebaubaren Parzellenfläche

4.3.6 Generierung von Gebäuden

4.3.6.1 Ableitung Gebäudegrundrisse

Auf Basis der zurückgesetzten Parzellenflächen und des maximalen Wertes für die Gebäudegrundfläche werden im nächsten Schritt die Grundrisse der Gebäude abgeleitet. Hierfür wurden verschiedene Ansätze ausgelotet, wovon aktuell drei im Rahmen des Modells eingesetzt werden.

Der erste und einfachste Ansatz basiert auf der Skalierung der reduzierten Parzellenfläche auf die zulässige Fläche für den Grundriss. Diese Methode ist einfach umsetzbar, da nur ein Skalierungszentrum sowie ein Skalierungsfaktor bestimmt und auf dieser Basis eine lineare Skalierung entlang der X- und Y-Achse ausgeführt werden müssen. Als Skalierungszentrum wird das Zentroid der Polygonfläche verwendet. Der Skalierungsfaktor wird aus dem Verhältnis der zurückgesetzten Parzellenfläche zur erlaubten Grundfläche bestimmt. Dieser Ansatz hat grössere Nachteile, wenn die zu skalierende Fläche nicht ein einfaches, konkaves Polygon ist. Durch konvexe Formen kann das Skalierungszentrum so zu liegen kommen, dass die skalierte Fläche in die Grenzabstände oder sogar in andere Parzellen hineinragt. Falls die Parzelle durch die Anwendung der Grenzabstände in mehrere Flächen unterteilt wird, so muss zudem darauf geachtet werden, dass für jede Teilfläche ein eigenes Skalierungszentrum bestimmt wird, da es ansonsten zu ähnlichen Problemen wie bei konkaven Polygonen kommen kann. Nicht zuletzt hat die Methode der reinen Skalierung den Nachteil, dass die resultierende Form der reduzierten Parzellenflächen mitunter ästhetisch wenig ansprechend ist und kaum für ein reales Haus in Frage käme. Aus diesen Gründen wird im Modell die Nutzung der reduzierten Parzellenfläche und Skalierung als Basis für den Gebäudegrundriss nur verwendet, wenn entweder die resultierende Fläche bereits unter der erlaubten Fläche liegt (und damit die Skalierung entfällt) oder die Ableitung durch die Quadtree basierte Methode kein Ergebnis liefert. Letzteres ist vor allem bei sehr schmalen oder kleinen Flächen der Fall.

Die zweite Methode basiert auf der Skalierungsmethode, ist aber für die Generierung von Häusern mit Innenhof oder geschlossenen Blockrändern gedacht. Die Reduktion der Fläche erfolgt dabei nicht durch Skalierung der ganzen, zurückgesetzten Parzellenfläche, sondern durch Abziehen eines Innenhofs in deren Mitte (vergleiche rechter Teil der Definition in Abbildung 20 oben). Hierfür wird aus der Differenz zwischen der zurückgesetzten Parzellenfläche und der erlaubten Grundfläche ein Skalierungsfaktor bestimmt. Die zurückgesetzte Parzellenfläche wird um diesen Faktor herunterskaliert um die Differenzfläche des Innenhofes zu erhalten. Dies geschieht analog zur oben beschriebenen Skalierungsmethode mit dem Zentroid als Skalierungszentrum. Die skalierte Fläche wird in Grasshopper mittels Extrusion und Solid Difference von der originalen Fläche abgezogen. Wie bei der einfachen Skalierungsmethode kann es zu Fehlern kommen, wenn die Parzellenfläche für den Blockrand nicht ein konkaves Polygon ist, da die Differenzfläche teilweise oder ganz ausserhalb der zurückgesetzten Parzellenfläche zu liegen kommen kann. Als Folge ist das Resultat der Differenzbildung dahingehend inkorrekt, dass die Fläche zu gross ist und kein sauberer Hof gebildet wird. Aus diesem Grund wird diese Methode im Modell nicht automatisiert eingesetzt, sondern muss vom Benutzer explizit parametrisiert werden. Dieser trägt also die Verantwortung, dass die Methode nur auf geeignete Fälle angewendet wird. Die Parametrisierung erfolgt wie in Abbildung 21 gezeigt durch Definition einer Gruppe des Typs *Blockrand*. Bei einer Gruppe mit einer einzelnen Parzelle resultiert ein einzelnes Gebäude mit Innenhof, bei mehreren Parzellen resultiert ein geschlossener Blockrand, der entlang der Parzellengrenzen unterteilt wird. Bei Gruppen mit mehreren Parzellen ist neben der konkaven Form der ganzen Gruppe zudem darauf zu achten,

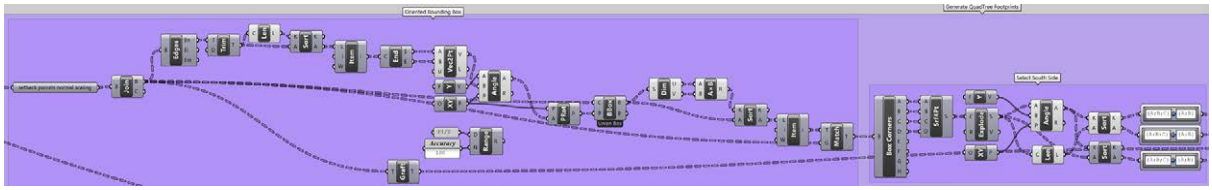
dass deren Fläche geschlossen ist und jede Parzelle an den Aussenrand grenzt. Parzellen, die vollständig von anderen Parzellen der Gruppe umgeben sind, werden durch die Differenzoperation unter Umständen vollständig eliminiert.

```
{
  "gruppen": [{
      "typ": "block",
      "parzellen_nummern": ["1214"]
    }, {
      "typ": "reihe",
      "parzellen_nummern": ["2430", "2722"]
    }
  ],
  "dachtypen": [{
      "typ": "flachdach",
      "parzellen_nummern": ["1214"]
    }, {
      "typ": "steildach",
      "parzellen_nummern": ["2430","2722"]
    }
  ]
}
```

Abbildung 21: JSON Konfiguration für Gruppierung und Dach Typen

Als weiterer Ansatz wurde die Bestimmung des flächengrössten, komplett in der zurückgesetzten Parzellenfläche enthaltenen Rechtecks untersucht. Algorithmen für die Bestimmung eines solchen Rechtecks für beliebige Polygone und Ausrichtungen weisen eine hohe Komplexität von $O(n^3)$ und damit hohen Rechenaufwand auf (Molano *et al.* 2012). Im vorliegenden Fall wurden Versuche mit zwei Algorithmen durchgeführt, welche das flächengrösste Rechteck annähern, aber nicht zwingend die optimale Lösung liefern. In einem ersten Schritt wurde ein ursprünglich in JavaScript geschriebener Algorithmus (Smilkov 2014) zunächst als Python und später als C# Script in Grasshopper adaptiert. Anstatt primitiver Datentypen und selbst geschriebener Routinen für geometrische Operationen wurden teilweise Geometrietypen und Operationen von Rhino Common verwendet. Die Umstellung von Python auf C# erfolgte aufgrund der besseren Integration von Rhino Common und zur Verbesserung der Performance. Für weitere Experimente wurde ein anderer Algorithmus leicht adaptiert, der bereits als C# Script in Grasshopper implementiert war (Delrieu 2016). Beide Algorithmen versuchen im Kern auf der Basis der gegebenen Parameter für eine Anzahl von Punkten innerhalb des Polygons das grösstmögliche Rechteck zu bestimmen und geben nach einer definierten Anzahl von Versuchen das grösste Rechteck als Resultat zurück. Beim ersten Algorithmus wird der Punkt als Schnittpunkt der maximalen Höhe und Breite und beim zweiten als Schnittpunkt der Diagonalen angenommen. Anschliessend wird für verschiedene Rotationswinkel das zugehörige Rechteck bestimmt und geprüft, ob dieses komplett innerhalb des Polygons der Parzelle liegt. Falls nicht, wird der Versuch verworfen. Ansonsten wird geprüft, ob die Fläche grösser als die des bisher grössten Rechtecks ist. Die generierten Rechtecke haben den Vorteil einer regelmässigen, gut für ein Haus geeigneten Form, welche bei Bedarf ohne Probleme durch Skalierung verkleinert werden können, da diese konvex und komplett in der zurückgesetzten Parzellenfläche enthalten ist. Die Experimente haben aber beträchtliche Nachteile gezeigt, weshalb der Ansatz schlussendlich verworfen wurde. Ein grosses Problem ist die lange Laufzeit der beiden Algorithmen. Obwohl es sich um eine Annäherungen mit besserer Laufzeit als bei den optimalen Algorithmen handelt, ist immer noch eine beträchtliche Anzahl an Versuchen pro Parzelle nötig, um geeignete Resultate zu erhalten. Insbesondere die Abfragen, ob ein Rechteck im Polygon enthalten ist, sind zeitaufwändig. Bei langgezogenen oder unregelmässigen Parzellenflächen sind die resultierenden Rechtecke oft kleiner als die erlaubte Fläche. Gleichzeitig können die Rechtecke nur verkleinert aber nicht vergrössert werden, da diese ansonsten in die Grenzabstände hineinragen würden. Insbesondere bei sehr schmalen Flächen oder Flächen mit minimal gekrümmten Seiten kann es ausserdem infolge numerischer Toleranzen zu Fehlern kommen, wodurch die resultierenden Rechtecke leicht in die Grenzabstände hineinragen. Ein weiterer Nachteil der untersuchten Algorithmen (wie auch des optimalen $O(n^3)$ Algorithmus) ist, dass diese nicht auf Löcher in den Polygonen ausgelegt sind. Deren Berücksichtigung würde die Komplexität zusätzlich erhöhen.

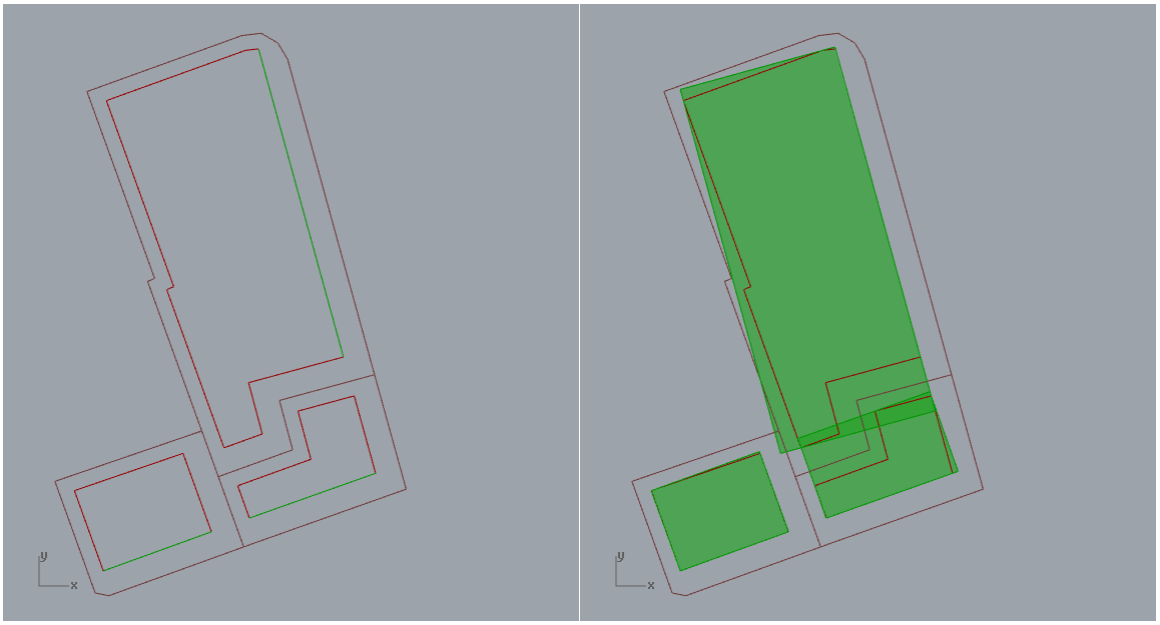
Zu guter Letzt wurden Ansätze zur Generierung der Gebäudegrundrisse auf Basis von Rastern untersucht. Dabei wird die reduzierte Parzellenfläche mit einem Raster angenähert und daraus der Grundriss abgeleitet. Zunächst ist die Ausrichtung des Rasters festzulegen. Eine globale Ausrichtung an der X- und Y-Achse für alle Parzellen hat sich bei Versuchen als suboptimal erwiesen. Einerseits sind dadurch alle Häuser weitgehend einheitlich orientiert, was in der Realität kaum der Fall ist. Andererseits ignoriert diese Ausrichtung die Form und Lage der reduzierten Parzellenfläche, wodurch mitunter kleinteilige und unnatürlich wirkende Grundrisse resultieren. Deshalb wird für die Generierung des Rasters ein lokal orientiertes Rechteck als Basis verwendet. Hierfür wurde eine Grasshopper Definition für die Generierung des kleinsten, orientierten, umgebenden Rechtecks (Minimum Oriented Bounding Box) adaptiert (Heumann 2011). Dieses berücksichtigt weitgehend die Lage der Parzellenfläche und liefert bereits wesentlich bessere Ergebnisse als die globale Orientierung. Bei unregelmässigen und aufgeteilten Flächen kann das Ergebnis aber weiterhin nicht ganz zufriedenstellend sein. Als weitere Verfeinerung wurde deshalb die in Abbildung 22 gezeigte Definition erarbeitet, bei der die Generierung des umgebenden Rechtecks so angepasst wurde, dass dieses am längsten Seitenabschnitt der Parzelle mit Orientierung zwischen Ost und Südwest orientiert ist.



Quelle: eigene Darstellung

Abbildung 22: Definition zur Bestimmung der Minimal Oriented Bounding Box

Abbildung 23 illustriert diesen Ansatz. Essenziell wird die Bounding Box an derjenigen Grundstücksseite aufgerichtet, welche der Hauptwohnseite entspricht und auf die der grosse Grenzabstand angewendet wird, was auch in der Realität meist der Hauptausrichtung des Gebäudes entspricht. Die auf Basis dieser Ausrichtung generierten Grundrisse zeigen generell eine nochmals leicht verbesserte Orientierung.

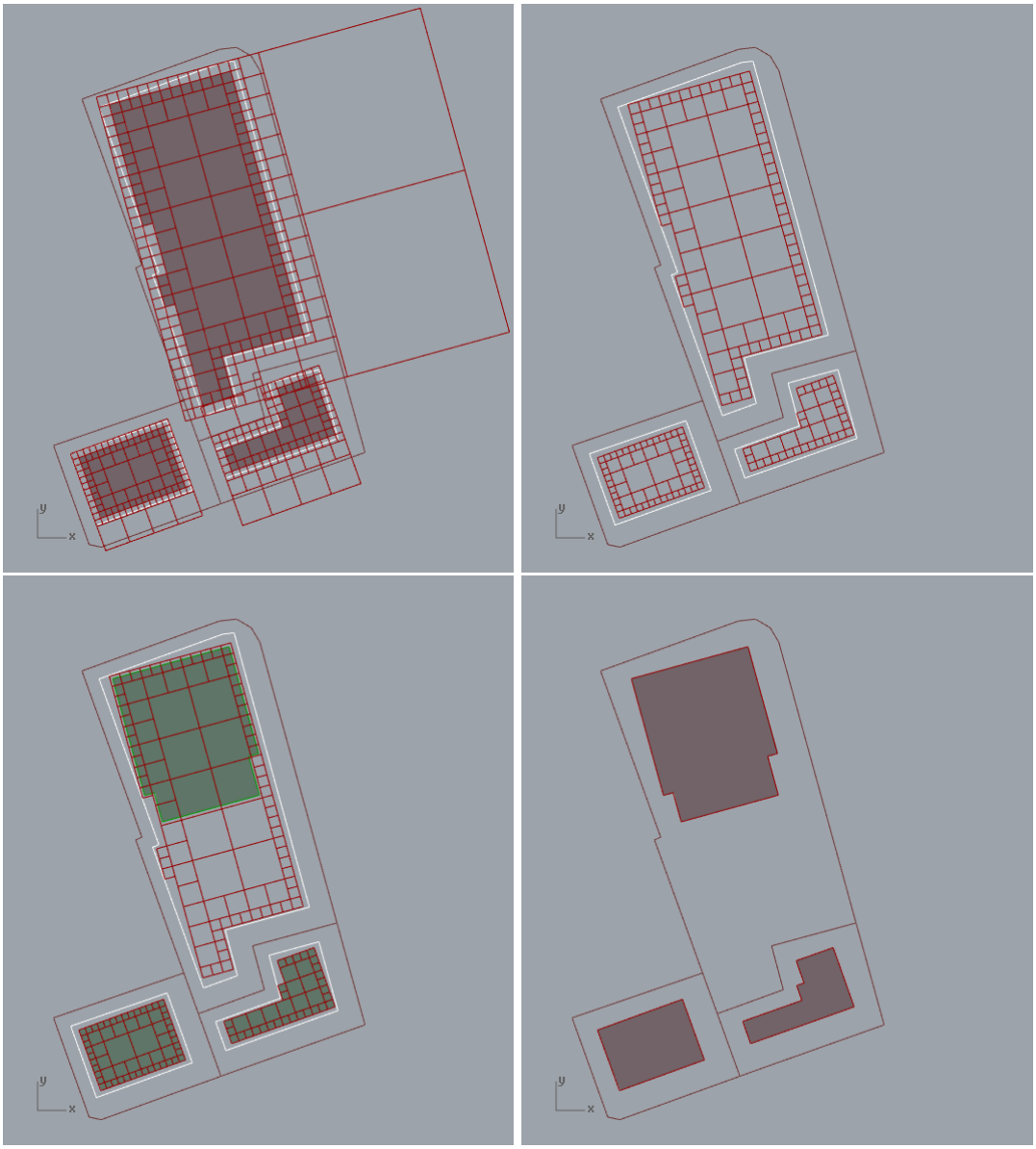


Quelle: eigene Darstellung

Abbildung 23: Oriented Minimal Bounding Boxes: längste Grundstückseite mit Ausrichtung nach Ost bis Südwest (l.) und entsprechend ausgerichtete Minimal Bounding Boxes (r.)

Für die eigentliche Generierung der rasterbasierten Grundrisse wurde zunächst ein bottom-up Ansatz getestet, bei dem zuerst das orientierte Rechteck in quadratische Rasterzellen unterteilt wurde. Danach wurde für alle Zellen bestimmt, ob diese innerhalb der zurückgesetzten Parzellenfläche liegen oder nicht. Aus den enthaltenen Rechtecken könnte dann durch Aggregation der Grundriss eines Gebäudes

abgeleitet werden. Die Umsetzung dieses ersten Ansatzes erfolgte rein in Grasshopper. Während die Machbarkeit gegeben ist, zeigten sich Performanceprobleme bei feineren Auflösungen der Rasterzellen. Tests zeigten, dass eine Seitenlänge von 1m oder kleiner nötig ist, um die Parzellenfläche genügend fein anzunähern. Dies führt insbesondere bei grossen Parzellen schnell zu sehr vielen Rasterzellen. Die Prüfung, ob diese innerhalb des umgebenden Polygons liegen, ist wiederum rechenintensiv. Gerade bei unregelmässig geformten Parzellen liegt dabei oft ein Grossteil der Rasterzellen ausserhalb der eigentlichen Parzellenfläche und wird somit nicht benötigt. Aus diesen Gründen wurde schlussendlich ein top-down Ansatz basierend auf Area Quadtrees umgesetzt, der in Abbildung 24 illustriert wird.



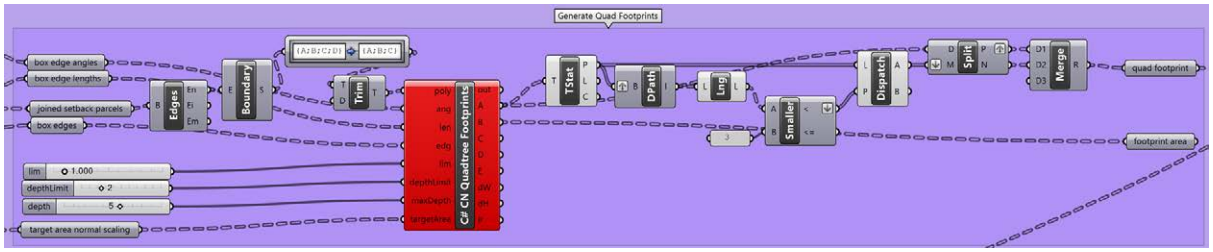
Quelle: eigene Darstellung

Abbildung 24: Ableitung von Gebäudegrundrissen mit Hilfe von Quadtree Zerlegung

Das umgebende Rechteck der reduzierten Parzellenfläche wird dabei zunächst an der kürzeren Seite zu einem Quadrat erweitert, dessen Seitenlänge der längeren Rechteckseite entspricht (vergleiche Abbildung 24 oben links). Dieses Quadrat dient als Basis für die Zerlegung mittels eines Area Quadtrees. Im vorliegenden Fall wird die Unterteilung der Fläche solange fortgesetzt, bis entweder 1) ein Quadrant komplett innerhalb oder ausserhalb der Parzellenfläche liegt, 2) bis die Seitenlänge des Quadranten eine definierte Schwelle unterschreiten würde oder 3) eine maximale Tiefe des resultierenden Quadtrees erreicht ist. Als Schwelle für die Seitenlänge hat sich wiederum 1m bewährt, während die Tiefe

zwischen 5 und 6 eine gute Balance zwischen Laufzeit und Annäherung der Parzellenfläche ergibt. Wie in Abbildung 24 oben links gut zu sehen ist, sind durch den Quadtree wesentlich weniger Quadrate zur Annäherung der Parzellenfläche nötig, da durch den top-down Ansatz schnell grössere Flächen identifiziert werden können, die innerhalb oder ausserhalb der bebaubaren Fläche liegen. Für die weitere Ableitung des Gebäudegrundrisses werden nur Quadrate innerhalb dieser Fläche berücksichtigt (Abbildung 24 oben links). Durch Selektion und Aggregation von Quadraten aus dieser Grundmenge werden die eigentlichen Grundrisse gebildet (Abbildung 24 unten links). Diese werden vor der weiteren Verarbeitung noch auf die erlaubte Fläche für den Grundriss herunterskaliert, falls die Aggregation zu einer Überschreitung dieses Wertes geführt hat.

In Abbildung 25 ist die Definition für die Ableitung der Gebäudegrundrisse auf Basis von Quadtrees zu sehen, in deren Zentrum ein C# Script zur Umsetzung des Algorithmus steht (siehe Anhang B.4.7 für Details). Für den Quadtree wurde ein Cardinal Neighbor Quadtree als Datenstruktur implementiert, der gegenüber normalen Quadtrees den Vorteil der Suche nach Nachbaren in linearer Zeit hat (Qasem und Touir 2015). Als Ausgangspunkt diente dabei eine in Go geschriebene Implementation von Cardinal Neighbor Quadtrees (Rainone 2017), die zunächst als Python Script in Grasshopper adaptiert wurde. Im Zuge dieser Adaptierung wurden wiederum Rhino spezifische Datentypen und Funktionen integriert, um z.B. die Quads abzubilden und deren Containment in der zurückgesetzten Parzellenfläche zu prüfen. Während weiterer Anpassungen und Erweiterungen wurde der Python Code für die Quadtrees in C# neu implementiert, wodurch sich die Performance annähernd verdoppelte. Die Datenstruktur wurde zudem um die Implementation eines Algorithmus für die Bestimmung der Connected Components im Quadtree ergänzt. Dieser basiert auf einem von Samet (1981) vorgeschlagenen Algorithmus, nutzt bei der Bestimmung von Nachbaren aber die Möglichkeiten des Cardinal Neighbor Quadtrees anstatt Tree Traversals.

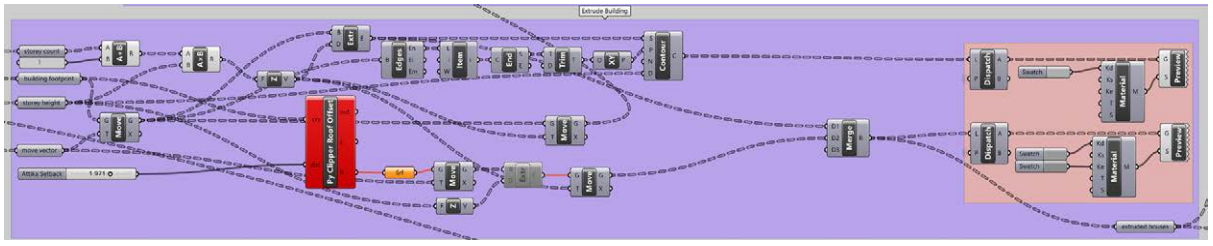


Quelle: eigene Darstellung

Abbildung 25: Definition zur Quadtree basierten Ableitung von Gebäudegrundrissen

4.3.6.2 Extrusion zur Erzeugung von Volumen

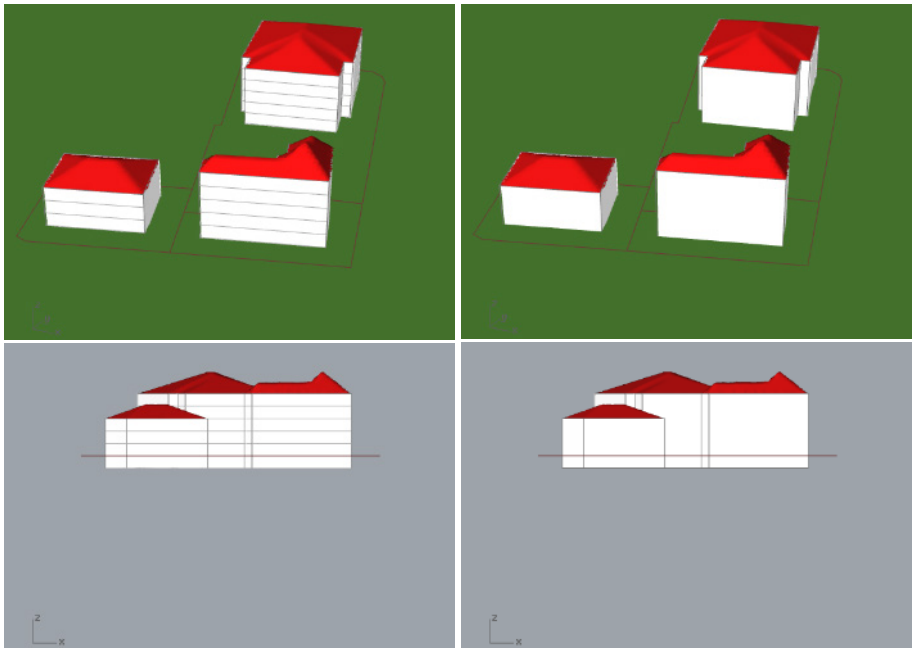
Die zuvor generierten Gebäudegrundrisse dienen zusammen mit der Geschosszahl und Geschosshöhe als Grundlage zur Erzeugung der Gebäudekörper. Die zugehörige Definition ist in Abbildung 26 zu sehen.



Quelle: eigene Darstellung

Abbildung 26: Definition zur Extrusion der Gebäudekörper

Aus der Geschosszahl und -höhe wird der Wert bestimmt, um den der Gebäudegrundriss in Z-Richtung extrudiert wird, um den Gebäudekörper zu erzeugen. Das Resultat entspricht den weissen Volumina, wie sie in Abbildung 27 zu sehen sind.



Quelle: eigene Darstellung

Abbildung 27: Darstellung von Gebäuden mit (l.) und ohne (r.) Hervorhebung von Stockwerken

Die Geschosszahl wird vor der Extrusion um ein Geschoss erhöht, um ein zusätzliches Untergeschoss zu erzeugen. Der Gebäudekörper wird entsprechend in Z-Richtung um eine Geschosshöhe nach unten versetzt, wie im unteren Teil der Abbildung 27 zu sehen ist. Dieser Versatz dient der kompletten Versenkung des Gebäudes in das Terrain, wenn ein Höhenmodell angezeigt wird.

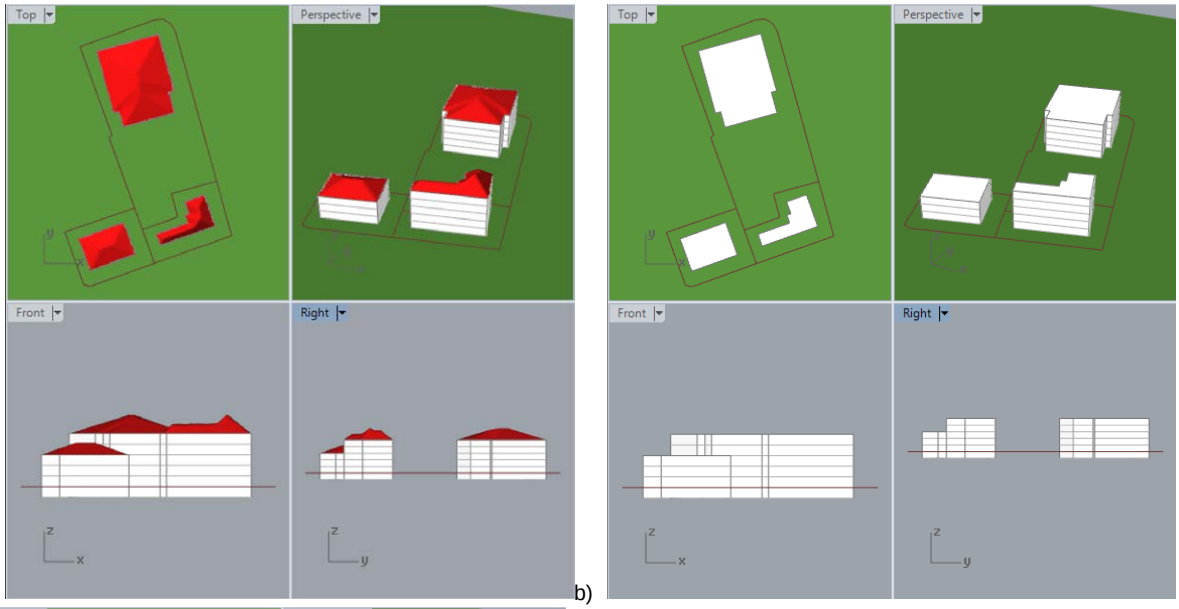
Um die Geschosszahl in der Visualisierung besser abschätzen zu können, werden die Geschosse mittels Linien auf den Fassaden angedeutet. Dies ist in Abbildung 27 links zu sehen, während rechts diese Linien ausgeblendet sind. Die Linien werden durch Generierung von Konturlinien mit Abstand von einer Geschosshöhe und mit Ausrichtung an der XY-Ebene erzeugt. Die Anzeige der Konturlinien kann durch Deaktivierung der zugehörigen *Preview* Komponente ausgeschaltet werden.

4.3.6.3 Generierung von Dächern

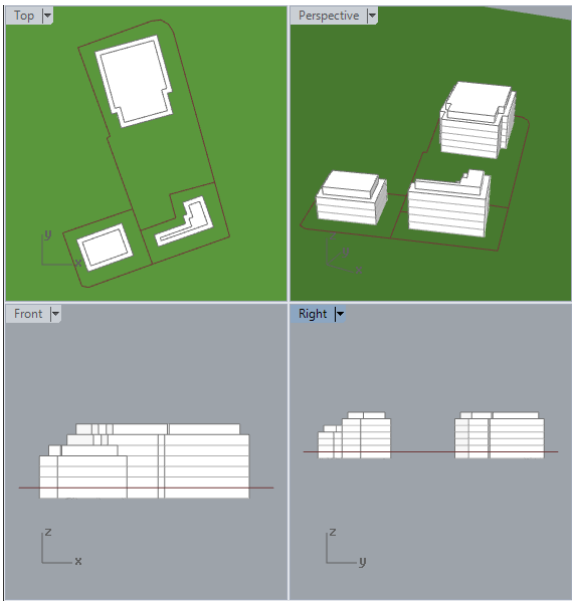
Nach Extrusion der Gebäudekörper bleibt die Erzeugung der Dächer als letzter Schritt zur Generierung der Gebäude. Im Modell sind die in Abbildung 28 gezeigten Dachformen implementiert, von denen das Steildach und das Flachdach effektiv genutzt werden. Als Standard wird für ein Gebäude ein Steildach generiert, sofern dessen Grundfläche nicht kleiner als 20m^2 oder grösser als 500 m^2 ist. In Anlehnung an von Kelly (2014, s.151) wurde für die Generierung der Steildächer ein Ansatz auf Basis der Medial Axis der Dachflächen gewählt. Hierfür wurde eine Grasshopper Definition adaptiert, welche einen Medial Axis Transform annähert (Delrieu 2018). Diese in Abbildung 29 gezeigte Definition wurde so angepasst, dass die Skalierung der Höhe des Daches mit der Parametrisierung des Modells im Einklang steht und deren Platzierung zu den generierten Gebäudekörpern passt. Die Dachgeometrie wird aktuell auf den eineinhalbfachen Wert der Geschosshöhe skaliert. Der Skalierungsfaktor (in Bezug auf die Geschosshöhe) kann global konfiguriert werden.

Liegt die Grundfläche des Daches ausserhalb der genannten Grenzen, wird ein Flachdach angenommen. In diesem Fall wird keine Geometrie generiert, sondern die Dachfläche des Gebäudevolumens leer gelassen. Die Grenzen für den Ausschluss von Steildächern können global konfiguriert werden, Diese wurden so gewählt, dass keine unvorteilhaften Steildächer generiert werden (entweder sehr steil oder sehr flach). Wie in Abbildung 21 ersichtlich, steht dem Benutzer darüber hinaus die Möglichkeit offen, für Parzellen explizit Dach Typen zu konfigurieren. Dadurch werden die gesetzten Limits für die Grundfläche übersteuert. Zudem wird eine Anpassung des Dach Typs von Steildach zu Flachdach zur Vermeidung der Überschreitung der maximalen Gebäudehöhe ebenfalls ausgeschlossen.

a) b)

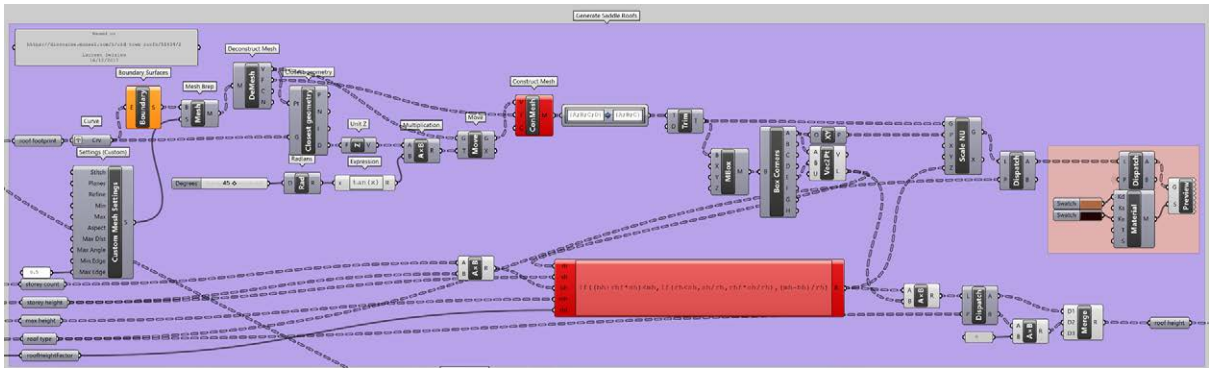


c)



Quelle: eigene Darstellung

Abbildung 28: Dachformen: a) Steildach, b) Flachdach, c) angenähertes Dachgeschoss



Quelle: eigene Darstellung

Abbildung 29: Definition zur Generierung von Steildächern auf Basis der angenäherten Medial Axis

Wie in Abbildung 29 unten links zu sehen ist, wurde zudem die Generierung angenäherter Dachgeschosse als Möglichkeit getestet und umgesetzt, wie sie in Modellen des Dencity schon Verwendung gefunden hat (Gilgen und Walczak 2017). Diese werden durch Zurücksetzen der Dachgrundfläche um einen bestimmten Betrag und Extrusion um eine Geschosshöhe erzeugt (zu sehen unten mittig in der Definition in Abbildung 26). Aufgrund der unklaren Semantik des Dachgeschosses in Hinblick auf die Geschossfläche und der Gefahr einer Verwechslung mit Attikageschossen wird diese Möglichkeit aber im Moment nicht genutzt.

4.3.7 Einbindung von Daten zur Umfeld Darstellung

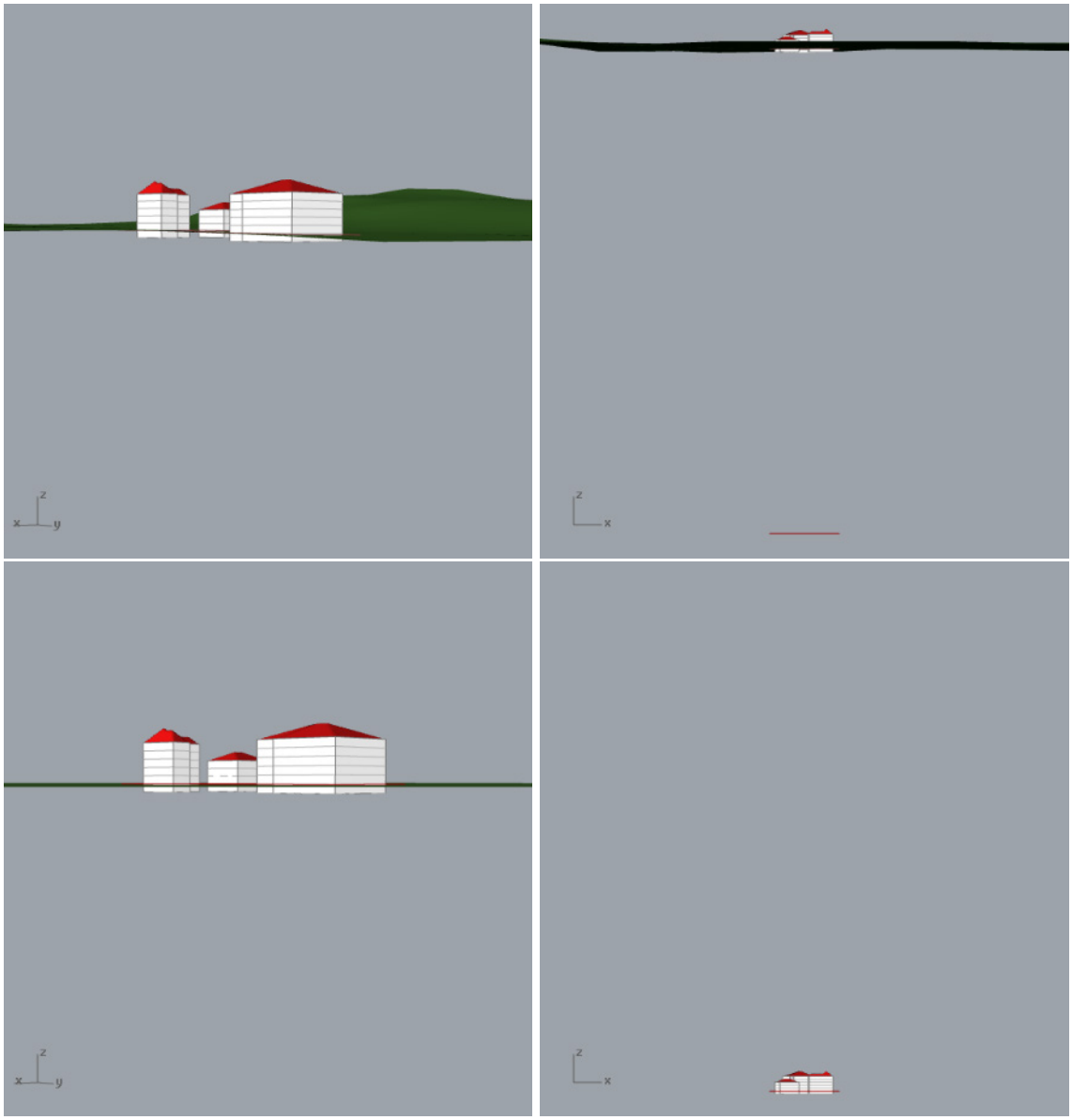
Als Ergänzung zu den generierten Gebäuden unterstützt das Modell die Einbindung und Darstellung von Terrain, Gebäuden und Strassen aus dem Umfeld des Projektperimeters. Die Darstellung dieser Objekte ermöglicht eine bessere Einschätzung der Situation im Projektgebiet als Ganzes, bietet Orientierungspunkte für dessen Verortung und lässt die Visualisierung insgesamt weniger abstrakt erscheinen. Aus diesem Grund werden die Objekte im Umfeld standardmässig angezeigt, können optional aber ausgeblendet werden. Die folgenden Unterabschnitte beschreiben die Einbindung der einzelnen Datenarten für die Darstellung des Umfeldes.

4.3.7.1 Höhenmodell

Die eigentliche Generierung der Gebäude erfolgt in der XY-Ebene. Dies liegt einerseits daran, dass die Parzellen in den Inputdaten keine Z-Koordinaten aufweisen. Andererseits unterstützen einige der verwendeten Grasshopper Komponenten nur planare Geometrien. Die Darstellung der Resultate würde deshalb auf einer flachen Ebene erfolgen, wie im unteren Teil der Abbildung 27 weiter oben zu sehen ist. Dabei werden alle Gebäude auf der gleichen Höhe dargestellt. Für eine realistischere Darstellung kann im Modell das Terrain in der Umgebung des Projektperimeters visualisiert werden.

Ein Höhenmodell auf Basis von SRTM Rasterdaten dient als Grundlage für die Darstellung des Terrains. Für die Bestimmung der Ausdehnung der Darstellung wird die Bounding Box um alle Parzellen im Projektgebiet gebildet und darauf ein zusätzlicher Buffer von 150 Meter angewendet. Für die Visualisierung werden zunächst die Pixel des SRTM Rasters innerhalb der bestimmten Ausdehnung abgefragt. Die Pixel werden dabei in Punkt Geometrien konvertiert und dann als GeoJSON ausgegeben, welches per Python Script in Rhino interne Geometrien umgewandelt wird. Für die Darstellung werden die resultierenden Punkte zunächst mit Hilfe einer Delaunay Vermaschung zu einem Mesh verknüpft. Um eine etwas geglättete Darstellung des Terrains zu erhalten, wird mittels des Weaverbird Add-On (Piacentino 2009) eine Catmull Clark Subdivision mit einem Unterteilungsschritt auf das Mesh angewendet.

Wie anhand der beiden Screenshots auf der rechten Seite von Abbildung 30 zu sehen ist, werden die Z-Werte aus dem Höhenmodell direkt übernommen, weshalb das Terrain auf seiner originalen Höhe oberhalb der XY-Ebene dargestellt wird. Deshalb wird bei der Darstellung des Terrains auf die generierten Häuser eine Translation in Z-Richtung angewendet. Für die Bestimmung des Betrages der Verschiebung wird zunächst das Zentroid des Gebäudegrundrisses ermittelt und dieses anschliessend in Z-Richtung auf das Terrain projiziert. Der Z-Wert des projizierten Punktes wird als Betrag für die Verschiebung des ganzen Gebäudes verwendet, wobei allerdings eine Geschosshöhe abgezogen wird. Bei den Gebäuden wird aus genau diesem Grund ein zusätzliches Untergeschoss generiert, welches die vollständige Versenkung in das Terrain gewährleisten soll. Dieses Vorgehen erfolgt in Analogie zu den swissBUILDINGS3D 2.0 Daten, wo die Häuser um 3 Meter in die Tiefe versetzt werden (swisstopo 2016). Eine ähnliche Versetzung auf das Terrain Mesh erfolgt für die Parzellengrenzen. Bei diesen werden allerdings die einzelnen Grenzpunkte individuell projiziert, damit im Resultat die Parzellengrenzen dem Verlauf des Terrains folgen.

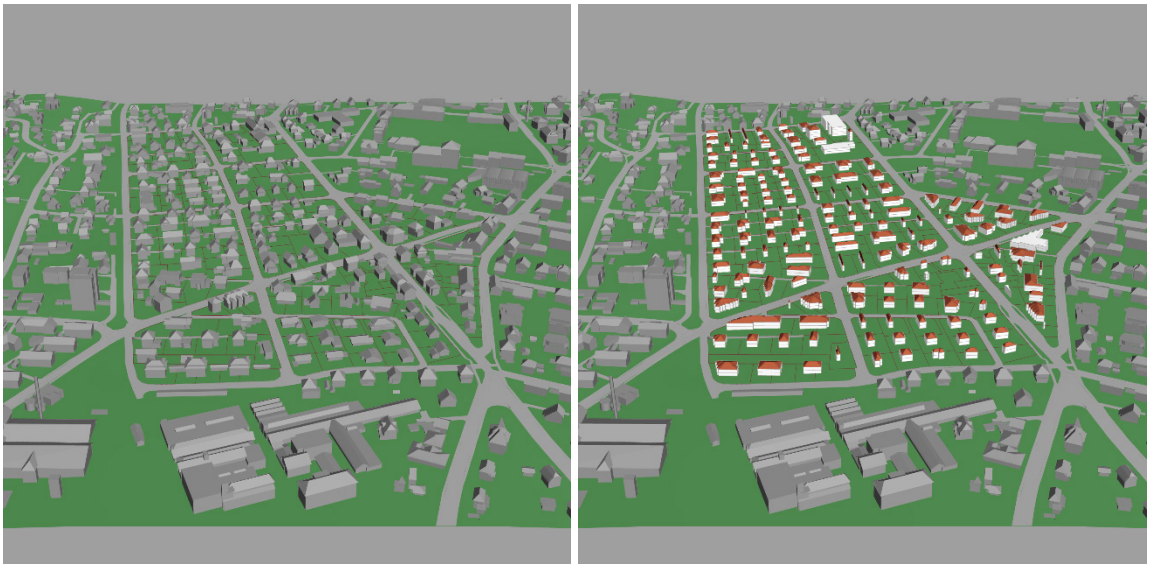


Quelle: eigene Darstellung

Abbildung 30: Einbindung Höhenmodell: Darstellung mit (o.) und ohne (u.) Höhenmodell

4.3.7.2 3D Gebäude

Neben dem Terrain können Gebäude der bestehenden Bebauung im Projektgebiet und dessen Umgebung visualisiert werden. Wie in Abbildung 31 auf der linken Seite ersichtlich ist, können entweder ausschliesslich die Gebäude der bestehenden Bebauung oder aber eine Kombination aus den generierten Gebäude im Projektperimeter mit den bestehenden Gebäuden im Umfeld angezeigt werden. Die kombinierte Darstellung ist dabei der Standard.



Quelle: eigene Darstellung

Abbildung 31: Darstellung Gebäude im Umfeld Ist-Situation (l.) und Kombination mit Szenario (r.)

Als Quelle für die Gebäudemodelle der bestehenden Bebauung dienen die swissBUILDINGS3D Daten der swisstopo, welche von Stereoluftbildern abgeleitet werden. Die Modelle wurden als PolyhedralSurface Geometrien in die zentrale PostGIS Datenbank integriert und werden für die Darstellung von dort abgefragt. Die Bounding Box des Terrain Mesh dient dabei als räumliche Einschränkung für die Abfrage. Bei der kombinierten Darstellung mit den generierten Gebäuden werden ausserdem alle Gebäude ausgeschlossen, die sich mit dem Projektperimeter überschneiden. Die PolyhedralSurface Geometrien müssen wie andere Geometrien in GeoJSON umgewandelt werden, um in Grasshopper importiert werden zu können. Allerdings unterstützt GeoJSON diesen Geometrietyp nicht. Deshalb macht sich das Modell die Tatsache zunutze, dass die Struktur der Well Known Text (WKT) Repräsentation der PolyhedralSurface Geometrien und von MultiPolygon Geometrien abgesehen vom Bezeichner des Geometrietyps identisch ist. Vor der Konversion in GeoJSON wird deshalb die Geometrie in WKT konvertiert, das Schlüsselwort POLYHEDRALSURFACE durch MULTIPOLYGON ersetzt und danach in GeoJSON konvertiert. Diesem Vorgehen kommt zudem entgegen, dass die swissBUILDINGS3D Modelle nicht als geschlossene PolyhedralSurfaces vorliegen, sondern im Kern eine Sammlung zusammengehöriger, dreieckiger Facetten (Faces) sind. In Grasshopper werden diese Dreiecke zunächst als Polyline Curves konvertiert, welche wiederum in einzelne Kanten zerlegt werden. Aus diesen Kanten werden anschliessend mit Hilfe der *Weave* Komponente des Weaverbird Add-on Mesh Objekte erzeugt.

Die swissBUILDINGS3D Gebäudemodelle weisen Z-Koordinaten auf, weshalb die Gebäude bereits auf der tatsächlichen Höhe über der XY-Ebene dargestellt werden. Entsprechend ist die gleichzeitige Anzeige des Terrains eine Voraussetzung für die korrekte Darstellung der swissBUILDINGS3D Gebäude. Allerdings versinken die Gebäude teilweise stärker als erwartet im Terrain, was auf vertikale Abweichungen des SRTM basierten Höhenmodells sowie dessen Projektion zurückgeführt wird. Für eine teilweise Korrektur dieses Umstandes kann ein Offset in Z-Richtung eingestellt werden, der in Form einer Translation auf alle Gebäude der Ist-Bebauung im Projektgebiet angewendet wird. Den Betrag des Offsets muss der Benutzer individuell abschätzen und einstellen. Durch Verwendung eines Höhenmodells der swisstopo, das mit den swissBUILDINGS3D Daten abgestimmt ist (insbesondere das swissTLM Höhenmodell), könnte dieses Phänomen vermutlich weitgehend vermieden werden.

4.3.7.3 Strassen

Als weiteres Mittel zur Orientierung werden Strassen im Umfeld des Projektgebietes visualisiert. Konkret werden die als «Strasse», «Weg» oder «Verkehrsinsel» klassifizierten Flächen aus der Bodenbedeckung der Amtlichen Vermessung zusammen mit dem Terrain und den Gebäuden der aktuellen Bebauung dargestellt.

Die darzustellenden Flächen werden als MultiPolygon Geometrien aus der zentralen PostGIS Datenbank ausgelesen, wobei die Bounding Box des Terrain Mesh als räumliche Einschränkung dient. Da die Strassenflächen in der Bodenbedeckung teilweise als ausgedehnte, zusammenhängende Flächen mit Löchern erfasst sind, resultieren Geometrien, welche Grasshopper nicht zuverlässig interpretieren kann. Aus diesem Grund werden die Flächen bei der Abfrage zuerst mit den Parzellen der Amtlichen Vermessung verschnitten, wodurch die komplexen Flächen in handhabbare Teilflächen unterteilt werden. Für die Integration in Grasshopper wird wiederum der Weg über GeoJSON als Schnittstellenformat genommen. Die resultierenden MultiPolygon Geometrien werden erst in Curve Objekte und dann in Mesh Geometrien übersetzt. Die Vertices dieses Meshes werden dann in Z-Richtung auf das Terrain projiziert. Da insbesondere in unebenen Bereichen mitunter Teile der Strassen im Terrain versinken, kann ähnlich wie bei den Gebäuden ein Z-Offset für die Strassen eingestellt werden. Bereits minimale Offsets beheben das Problem weitgehend. Beim betrachteten Massstab fällt es denn auch nicht weiter auf, dass die Strassen technisch gesehen leicht oberhalb des Terrains liegen.

4.3.8 Speicherung von Szenarien und Resultaten

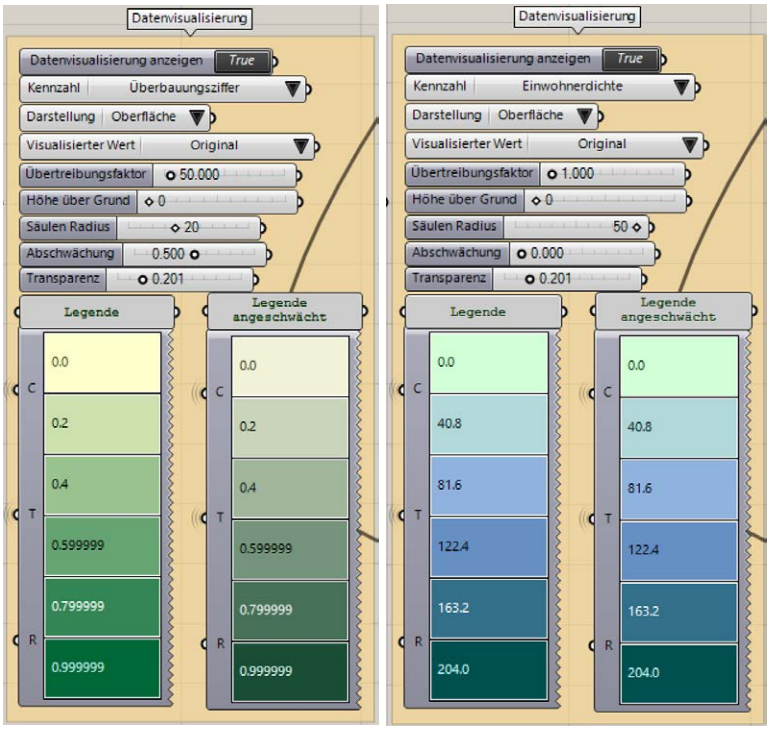
Das parametrische Modell erzeugt in erster Linie dreidimensionale Visualisierungen von Varianten von verdichtetem Bauen im Projektgebiet. Dies erfolgt in Form grafischer Outputs am Bildschirm, welche für die Auswertung zudem als Bilder exportiert werden. Zusätzlich wurde im Modell mit Hilfe von Python und Slingshot! die Möglichkeit zur Speicherung von Outputs in der Datenbank implementiert. Dabei kann zwischen der Speicherung von Szenarien und der Speicherung der generierten Resultate unterschieden werden.

Bei der Speicherung der Szenarien werden nur die in den Input Komponenten der Modellzonen festgelegten Parameterwerte in die Datenbank geschrieben. Wie in Kapitel 4.3.4 schon angedeutet, werden hierfür entsprechend der Konfiguration die Werte aus den Input Komponenten ausgelesen und auf dieser Basis SQL Statements für die Speicherung der Werte generiert, welche dann mittels Slingshot! auf der Datenbank ausgeführt werden. Für die Speicherung werden kombinierte INSERT/UPDATE Statements generiert, welche neue Werte einfügen und bestehende Einträge aktualisieren (vergleiche Anhang B.4.13).

Die Speicherung der generierten Resultate muss separat von der Speicherung der Parameterwerte ausgelöst werden. Ohne vorherige Speicherung der Resultate funktionieren die Analysen zur Auswertung des Szenarios nicht korrekt, da ihnen ein Teil der Inputs fehlt. Um diese Inputs zu erhalten, müssen die Outputs des parametrischen Modells zuerst aufbereitet werden. Die zugehörige Definition wurde bereits in Abbildung 16 (Seite 54) wiedergegeben. Einerseits werden eine Reihe von Berechnungen für Attribute wie Gebäudegrundfläche, Geschossfläche, Gebäudehöhe, Anzahl Beschäftigte und Anzahl Einwohner ausgeführt. Andererseits werden die Geometrien der Gebäudegrundrisse gespeichert, welche mit Hilfe eines Python Scripts in GeoJSON übersetzt werden. Der entsprechende Code für die Umwandlung der Rhino BREP Surfaces in MultiPolygon GeoJSON Objekte wurde selbst entwickelt (vergleiche Anhang B.4.14). Die verschiedenen Inputs aus der Aufbereitung werden anschliessend in eine weitere *GH Python* Komponente gespiesen, welche die nötigen SQL Befehle für das Speichern der Daten erzeugt. Im Fall der Szenario Resultate wird hierfür neben einem DELETE Statement zum Löschen der schon bestehenden für jedes Gebäude im Input ein einzelnes INSERT Statement erzeugt (vergleiche Anhang B.4.15). Diese SQL Befehle werden schliesslich in eine *Command* Komponente für die Ausführung auf der Datenbank gespiesen.

4.3.9 Dreidimensionale Datenvisualisierung

Im Rahmen der Arbeit wurde auch die Möglichkeit einer direkten Visualisierung von Kennzahlen zur Ist-Situation innerhalb von Rhino Grasshopper ausgelotet. Dazu wurden im parametrischen Modell entsprechende Funktionen ergänzt, deren Ergebnisse optional eingeblendet werden können. Abbildung 32 zeigt die Parametrisierung dieser Visualisierungsfunktionen.

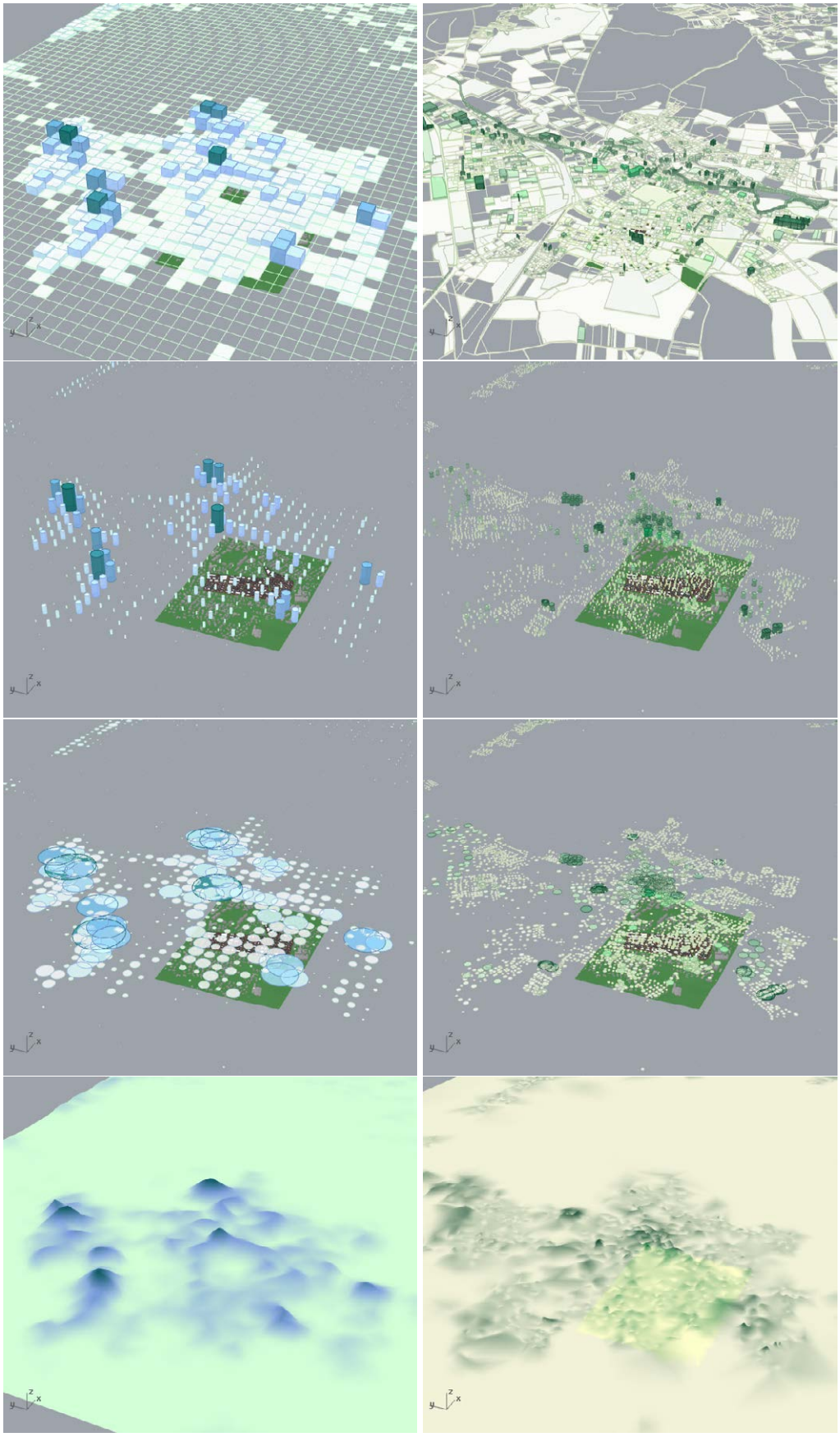


Quelle: eigene Darstellung

Abbildung 32: Parametrisierung Datenvisualisierung und generierte Legenden

Aktuell können bezogen auf die Hektare Einwohnerdichte, Beschäftigtendichte, Raumnutzerdichte, Wohnungsdichte, Haushaltsdichte, durchschnittliche Wohnfläche, bauliche Dichte und Freiraumanteil sowie bezogen auf die Parzelle die Überbauungsziffer visualisiert werden. Für jede dieser Kennzahlen ist im Modell eine SQL Abfrage hinterlegt, welche die Werte der Kennzahl und die zugehörige Geometrie ausliest. Für die Visualisierungen werden aus diesen Inputs dreidimensionale Darstellungen generiert. Wie in Abbildung 33 ersichtlich ist, kann zwischen vier Varianten von Visualisierungen ausgewählt werden. Die einfachste Variante ist die Extrusion der zugehörigen Geometrie auf Basis des Wertes der Kennzahl. Eine zweite Variante stellt am Zentroid der Geometrie eine Säule dar, deren Höhe und Durchmesser vom Kennzahlenwert abhängt. Der Säulendarstellung verwandt ist die Darstellung als Kreise, bei denen nur der Radius variiert wird, aber keine Extrusion in die Höhe erfolgt. Die vierte Variante besteht darin, dass die Zentroide als Höhenpunkte mit dem Wert der Kennzahl interpretiert werden, welche durch eine Delauney Triangulation in eine Oberfläche vermascht werden.

Allen Darstellungen ist gemein, dass diese im Raum oberhalb des höchsten Punktes des Projektgebiets dargestellt werden, wobei der Offset in Z-Richtung parametrisiert werden kann. Zudem wird bei allen Darstellungen die Farbe anhand des Kennzahlenwertes variiert. Hierfür können Farbverläufe mit zwei oder drei Farbwerten im Modell hinterlegt werden. Die Zuordnung der Farben erfolgt durch lineare Interpolation auf Basis des Wertebereiches der jeweiligen Kennzahl und den individuellen Werten. Während die Farben an die Skalen der gleichen Auswertungen in QGIS angelehnt sind, resultieren durch die lineare Interpolation andere farbwerte für die Visualisierung in Grasshopper. Da für die Visualisierung das ganze Gemeindegebiet berücksichtigt wird, besteht die Möglichkeit, für Datenpunkte ausserhalb des Projektgebietes und dessen unmittelbaren Umgebung die Sättigung der Farben zu reduzieren. Dies ist in Abbildung 33 unten rechts gut ersichtlich. Zudem kann global die Transparenz der Darstellung beeinflusst werden.



Quelle: eigene Darstellung

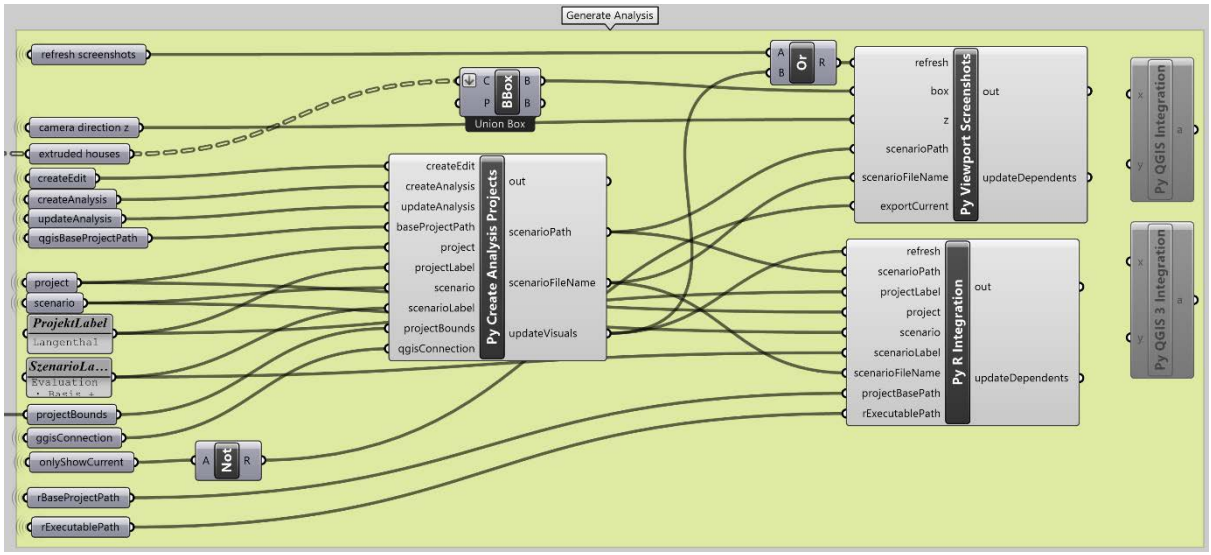
Abbildung 33: Datenvisualisierung Einwohnerdichte (l.) und Überbauungsziffer (r.) als Extrusion, Säulen, Kreise und Oberfläche (v.o.n.u.)

Die grossen Unterschiede bei den Wertebereichen der verschiedenen Kennzahlen erschweren eine einheitliche Visualisierung. Bei einer Extrusion der Überbauungsziffer, deren Wert zwischen 0 und 1 liegt, sind zum Beispiel im Unterschied zu Einwohnerzahlen kaum Unterschiede wahrnehmbar. Deshalb bietet das Modell zwei Stellschrauben, welche beeinflussen, wie die Werte für die Visualisierung interpretiert werden. Einerseits kann gewählt werden, ob der originale Wert dargestellt werden soll oder ob der Wertebereich, der vom Resultat der SQL Abfrage abgedeckt wird, in den Wertebereich 0 bis 1 abgebildet werden soll (Standardisierung). Als zweite Option steht ein so genannter Überhöhungsfaktor zur Verfügung, der als Multiplikator auf den dargestellten Wert angewendet wird. Ein Wert von 1 bewirkt keine Änderung, ein Wert über 1 führt zu einer Übertreibung des dargestellten Werts und ein Wert unter 1 zu einer Verkleinerung.

Die Freiheitsgrade dieses Modells zur Visualisierung von Daten haben den Vorteil, dass interaktiv verschiedene Eigenschaften der Kennzahlen herausgeschält und teilweise recht intuitiv dargestellt werden können. Allerdings fehlt an diesem Punkt die Möglichkeit, einen Parametersatz für die Visualisierung zu speichern und wieder zu laden, weshalb die Parametrisierung jedes Mal erneut vorgenommen werden muss. Während insbesondere gewisse relative Verhältnisse und deren Verteilung im Raum durch die Visualisierungen recht gut verdeutlicht werden können, ist die quantitative Interpretation der Darstellungen relativ schwierig. Ebenso muss man vorsichtig sein, dass durch die Stellschrauben mit der Standardisierung der Werte oder die Übertreibung nicht falsche Schwerpunkte gesetzt werden und es zu Fehlinterpretationen kommt. Zudem erwies sich der Aufbau der Visualisierungen als relativ langsam. Aus diesen Gründen wurde die dreidimensionale Visualisierung zwar im parametrischen Modell belassen, für das Vorgehensmodell aber klassischen Auswertungen in Form zweidimensionaler Karten der Vorzug gegeben.

4.3.10 Generierung von Auswertungen

Das parametrische Modell in Grasshopper implementiert mehrere Funktionen, welche der Generierung von Auswertungen zur Ist-Situation und den Szenarien dienen. Wie in Abbildung 34 ersichtlich ist, sind diese Funktionen allesamt mittels Python umgesetzt, da die inhaltliche Umsetzung der Analysen durch externe Komponenten erfolgt.



Quelle: eigene Darstellung

Abbildung 34: Grasshopper Definition zur Generierung von Auswertungen

Eine erste Komponente («Py Create Analysis Projects») generiert auf Basis verschiedener Input QGIS Projekte zum aktuell ausgewählten Szenario (siehe Anhang B.4.16 für Details). Einerseits kann die Komponente ein einfaches Projekt mit den Layern für Parzellen, Abstandslinien und Bauzonen generieren, welches für die Bearbeitung und Erfassung von Daten gedacht ist. Andererseits kann ein umfangreiches QGIS Projekt mit den Layern zur Analyse der Ist-Situation im Projektgebiet sowie des aktuellen Szenarios erzeugt werden, welches zudem Layouts für die Ausgabe von PDF Berichten um-

fasst. Die Projekte werden durch die Parametrisierung von speziell präparierten QGIS Projekten erzeugt, welche als Vorlage dienen. Bei der Parametrisierung werden aufgrund der Inputparameter das Projekt, das Szenario, zugehörige Bezeichnungen sowie die Extents des Projektgebietes festgelegt. Der Benutzer muss anschliessend das Projekt in QGIS öffnen und kann bei Bedarf manuell die PDF Berichte exportieren.

Damit die Berichte komplett dargestellt werden, müssen zuvor einige zusätzliche Inhalte in Form von Bilddateien generiert und zusammen mit dem generierten QGIS Projekt abgelegt werden. Diese Bilder werden von den Berichtslayouts eingebunden. Deshalb wird nach der Erzeugung des Auswertungsprojektes automatisch die Generierung dieser Inhalte angestossen. Einerseits ruft eine Komponente («Py Viewport Screenshots») Funktionen der laufenden Rhino Instanz auf, um Ansichten der 3D Visualisierung zu exportieren (vergleiche Anhang B.4.18). Es werden dabei keine echten Renderings erzeugt, sondern Ansichten des Preview Viewports als PNG exportiert. Die Grafikqualität der Preview ist zwar etwas weniger detailliert als bei einem Rendering, der Aufwand für die Erzeugung der Bilder ist aber wesentlich kleiner und die Ausgabe erfolgt schneller. Der Code für den Export stellt automatisch die gewünschten Parameter für die Preview ein, justiert die Kamera für die gewünschten Ansichten, exportiert die Bilder und stellt abschliessend die ursprüngliche Ansicht wieder her. Die vier Ansichten aus den vier Himmelsrichtungen sind in Form von Vektoren parametrisiert, welche mit einem automatischen Zoom der Ansicht auf die Bounding Box der erzeugten Häuser kombiniert werden, um die Kameraansicht einzustellen.

Eine weitere Komponente («Py R Integration») erzeugt für mehrere der Kennzahlen mit Hilfe der Statistikumgebung R eine Reihe von Grafiken. Die Integration von R ist dabei so gestaltet, dass nacheinander zwei Subprozesse gestartet werden, die den R Interpreter aufrufen und jeweils ein Script für die Erzeugung der Grafiken auf Basis der Daten aus der PostGIS Datenbank ausführen (vergleiche Anhang B.4.17). Eines dieser Scripts erzeugt Grafiken zur Ist-Situation (vergleiche Anhang B.5.1) und das andere Grafiken zum aktuellen Szenario (vergleiche Anhang B.5.2). Beide Scripts sind im Aufbau ähnlich, unterscheiden sich aber in den ausgeführten SQL Abfragen und dem ausgewerteten Extent (ganze Gemeinde beziehungsweise Projektgebiet). In beiden Fällen wird dem Script die ID des Projektes und bei der Auswertung des Szenarios zusätzlich dessen ID als Parameter übergeben. Das Script führt damit parametrisierte SQL Abfragen aus und exportiert darauf basierend Grafiken als PNG Dateien. Der Ausgabepfad wird dabei so parametrisiert, dass die Bilder zusammen mit dem generierten QGIS Projekt abgelegt werden.

Wie anhand zweier Komponenten am rechten Rand der Abbildung 34 ersichtlich ist, wurde zudem eine direkte Anbindung von QGIS mittels Python geprüft. Da QGIS eine relativ speziell konfigurierte Python Umgebung verwendet, deren Packages sich nicht direkt mit der IronPython Umgebung in Grasshopper integrieren liessen, wurde stattdessen eine Integration analog zum Aufruf von R gewählt. Es wird also ein externer Prozess mit dem QGIS eigenen Python Interpreter gestartet, der ein separates Python Script ausführt (vergleiche Anhang B.4.19). Aufgrund relativ langer Laufzeiten beim Export der PDF Berichte aus QGIS, was zu einem Blockieren in Grasshopper führt, sowie ungelösten Fragen bei der sicheren Speicherung der Zugangsdaten für die Datenbank, wurde auf diese direkte Integration von QGIS schlussendlich verzichtet.

5 Ergebnisse

5.1 Vorgehensmodell zur Vermittlung der Auswirkungen innerer Verdichtung

5.1.1 Relation zu Geodesign

Geodesign und speziell das Geodesign Framework von Steinitz (2012) diente in der Arbeit in zweierlei Hinsicht als Grundlage. Einerseits diente dieses der Strukturierung der Ausarbeitung des Vorgehensmodells zur Vermittlung der Auswirkungen der inneren Verdichtung. Die sechs Schritte des Frameworks wurden in mehreren Iterationen durchlaufen, um 1) ein Verständnis für die Thematik zu erlangen, 2) die Methoden und Modelle für das Vorgehensmodell zu spezifizieren sowie 3) Varianten für verdichtetes Bauen auf Basis des Vorgehensmodells zu generieren. Das Framework und die von Steinitz als Hilfestellung definierten Fragen zu einzelnen Aspekten des Geodesign Prozesses waren dabei eine wertvolle Unterstützung, welche die Einarbeitung in die Thematik Geodesign und das für den Autor neue Fachgebiet der Planung strukturierten und erleichterten. Andererseits diente das Steinitzsche Framework als direktes Vorbild für das ausgearbeitete Vorgehensmodell. Dessen Strukturierung leitet sich aus den Schritten des Frameworks ab und die entwickelten Systemteile sind direkt an die Modelle angelehnt, welche zu den Schritten gehören.

Die Arbeit und deren Ergebnisse sind somit eng mit Geodesign verknüpft, sind aber nicht als Geodesign Prozess im engeren Sinne zu betrachten. Die durchlaufenen Schritte decken zwar den Grossteil eines Geodesign Prozesses ab, es fehlen aber wichtige Elemente zu dessen Vervollständigung. Zunächst fehlt ein konkretes Planungsprojekt mit einer spezifischen Fragestellung, an dessen Beispiel das Vorgehensmodell erarbeitet und getestet worden wäre. Das Ziel der Vermittlung der Auswirkungen innerer Verdichtung, welches als Leitlinie diente, ist vielmehr als abstrahierte Version einer solchen Fragestellung anzusehen, die anhand von zwei Projektperimetern beispielhaft bearbeitet wurde. Da kein konkretes Planungsprojekt vorhanden war, fehlt ebenso das partizipative Element mit Einbezug betroffener Anspruchsgruppen. In einem Geodesign Prozess und in Planungsprozessen allgemein ist die Involvierung von vor Ort betroffenen Personen aber als integraler Punkt und zentraler Vorteil anzusehen, wenn es um die Schaffung eines gemeinsamen Verständnisses und breit abgestützter Lösungen geht (Steinitz 2012; Grams 2015; Moura 2015; Wissen Hayek *et al.* 2016). Auch Entscheidungsträger als spezielle Anspruchsgruppe waren in der Arbeit nicht involviert. Zusammen mit der lediglich abstrakt formulierten Fragestellung für den Designprozess ist dies der Hauptgrund dafür, dass im Rahmen der Arbeit die Beurteilung der Ist-Situation sowie der Entscheid für spezifische Verdichtungsvarianten im Rahmen der Evaluation und Decision Models weitgehend offengelassen wurden. Allerdings wurden für das Vorgehensmodell Grundlagen entwickelt, um die Bewertung und Entscheidungsfindung zu unterstützen. Diese nehmen Anleihen bei anderen Arbeiten und können im Rahmen der Anwendung in konkreten Testplanungen die Vermittlung der ansonsten abstrakten Zusammenhänge und Fragestellungen in Bezug auf innere Verdichtung unterstützen sowie erleichtern, wobei insbesondere dem parametrischen Modell und dessen interaktiver Nutzung im Rahmen von Workshops Potential zugeschrieben wird (Walz *et al.* 2008; Van Wezemael *et al.* 2014; Moura 2015; Wissen Hayek *et al.* 2016).

Auch wenn die Arbeit nicht als kompletter Geodesign Prozess anzusehen ist, kann die Erarbeitung des Vorgehensmodells als Design Prozess angesehen werden, der den Entwurf des Vorgehensmodells zum Ziel hatte. Das Durchlaufen des Frameworks nach Steinitz (2012) kann im Sinne von Foster (2016) als «Entwurf des Designs» («designing the design») bezeichnet werden. Das entworfene Vorgehensmodell kann als Blaupause oder eine auf Fragestellungen im Bereich der Verdichtung zugeschnittene Version des Steinitzschen Frameworks gesehen werden. Dessen Anwendung in einer konkreten Testplanung würde dann einen kompletten Geodesign Prozess konstituieren. Eine solche Anwendung steht bisher noch aus. Diese ist als Teil nachfolgender Arbeiten anzustreben, um die Verifizierungsschritte in der Masterarbeit zu ergänzen.

5.1.2 Vorgehen bei der Verifikation

In Absenz eines konkreten Planungsprojektes als Testfall für das Vorgehensmodell wurde ein mehrteiliges Vorgehen zu dessen Verifizierung konzipiert und umgesetzt. Dabei liegt der Schwerpunkt auf der Beurteilung des zentralen parametrischen Modells, welches das Change Model des Geodesign Frameworks umsetzt, sowie dessen Resultate.

Ein erster Teil der Verifikation zeigt die Übertragbarkeit des Modells auf. Hierzu wurde das anhand eines Fallbeispiels in Langenthal entwickelte Modell auf einen zweiten Fall in Wohlen bei Bern übertragen. Der zweite Teil der Verifizierung hat die Beurteilung der Resultate aus Expertensicht zum Ziel. Für die beiden Fallbeispiele in Langenthal und Wohlen wurden mit Hilfe des Modells jeweils eine Reihe repräsentativer Szenarien definiert, welche mit dem Dencity abgestimmt wurden. Die resultierenden Verdichtungsvarianten wurden aufbereitet und den Experten des Dencity in einem Workshop zur Beurteilung vorgelegt, um eine Einschätzung der Plausibilität aber auch der Qualität der Resultate aus Sicht von Planern und Architekten zu erhalten. Der dritte und letzte Teil der Verifizierung besteht aus einer Analyse der Sensitivität des parametrischen Modells in Bezug auf ausgewählte Modellparameter. Hierfür wurden diese Parameter innerhalb vordefinierter Wertebereiche variiert, daraus Verdichtungsvarianten generiert und die Resultate ausgewertet.

Zur Quantifizierung der Resultate für die Expertenbeurteilung und die Sensitivitätsanalyse werden die Geschossflächenziffer, die Baumassenziffer und die Überbauungsziffer als bauliche Dichtekennziffern herangezogen. In ihrer Definition entsprechen diese weitgehend den gleichnamigen Kennziffern, welche bereits in den Reporten zu den Evaluation und Decision Models verwendet werden. Unterschied ist, dass diese bei der Auswertung der Szenarien für den Expertenworkshop nicht bezogen auf die Parzelle, sondern bezogen auf den ganzen Projektperimeter und Modellzonen berechnet wurden.

Es ist zudem darauf hinzuweisen, dass die drei Kennziffern im Kern sowohl vom Wesen als auch von der Berechnung her den Ausnützungsarten mit gleichen Namen entsprechen, welche zusammen mit der Nutzungsziffer im parametrischen Modell für die Festlegung der maximal erlaubten Ausnützung verwendet werden. In der Auswertung werden die Kennzahlenwerte aus den Resultaten und die Ausnützungsart einander gegenübergestellt. Zur Unterscheidung zwischen den beiden Sichtweisen werden die Namen der Kennzahlen jeweils ausgeschrieben (Geschossflächenziffer, Baumassenziffer und Überbauungsziffer). Wenn die Ausnützungsart und die zugehörige Nutzungsziffer gemeint sind, wird hingegen die jeweilige Abkürzung verwendet (*GFZ*, *BMZ*, *ÜZ*). In Tabelle 6 werden die Begriffe und Berechnungsweisen definiert, welche der numerischen Auswertung der Szenarien zugrunde liegen.

Tabelle 6: Definition Kennzahlen für Auswertung

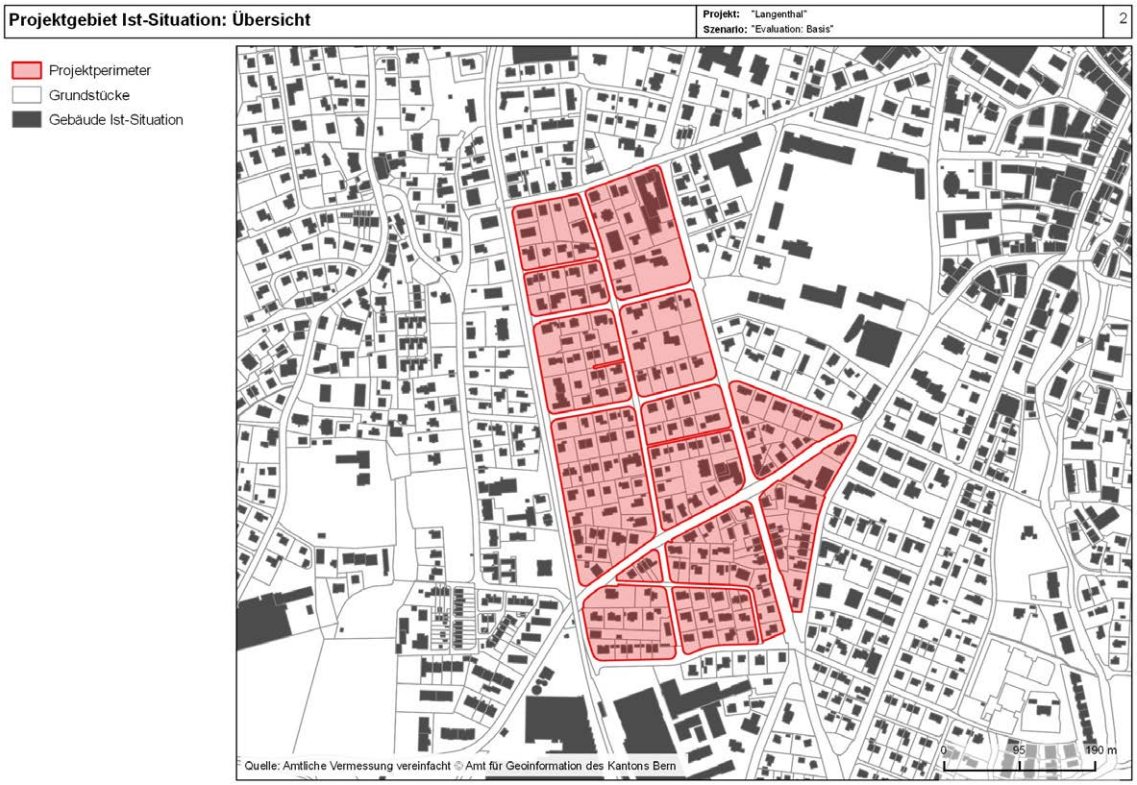
Kennzahl	Definition
Anrechenbare Grundstücksfläche	Anteil der Grundstücksfläche, welcher bebaut werden kann. Im Modell wird dieser Anteil über ein Attribut der Parzelle festgelegt.
Überbaute Fläche	Entspricht der Summe der Grundflächen der generierten Gebäude. Bei Gebäuden, welche Parzellengrenzen überschreiten, wird für die Auswertung auf Parzellenebene der Grundriss an der Parzellengrenze unterteilt.
Geschossfläche	Entspricht der gesamten Geschossfläche der generierten Gebäude. Die Geschossfläche eines Gebäudes wird berechnet als: $Geschossfläche = Gebäude\ Grundfläche * Geschosszahl$ Bei Auswertung auf Parzellenebene wird die Geschossfläche auf Basis der Unterteilung an der Parzellengrenze aufgeteilt.
Baumasse	Entspricht dem Volumen der Kubatur der generierten Gebäude. Die Baumasse wird berechnet als: $Baumasse = Gebäude\ Grundfläche * Gesamthöhe$ Die Gesamthöhe entspricht dabei dem Produkt Geschosszahl mal der Geschosshöhe plus der Höhe eines allfälligen Daches. Bei Auswertung auf Parzellenebene wird die Baumasse auf Basis der Unterteilung an der Parzellengrenze aufgeteilt.
Geschossflächenziffer	Das Verhältnis der gesamten Geschossfläche auf einer Parzelle zur anrechenbaren Grundstücksfläche: $Geschossflächenziffer = \frac{Geschossfläche}{Anrechenbare\ Grundstücksfläche}$ Die globale Geschossflächenziffer entspricht dem Verhältnis der gesamten Geschossfläche im Projektgebiet zur gesamten anrechenbaren Grundstücksfläche, ohne Berücksichtigung von einzelnen Parzellen: $Geschossflächenziffer_g = \frac{SUM(Geschossfläche_{Parzelle})}{SUM(Anrechenbare\ Grundstücksfläche_{Parzelle})}$
Baumassenziffer	Das Verhältnis der gesamten Baumasse auf einer Parzelle zur anrechenbaren Grundstücksfläche. $Baumassenziffer = \frac{Baumasse}{Anrechenbare\ Grundstücksfläche}$ Die globale Baumassenziffer entspricht dem Verhältnis der gesamten Baumasse im Projektgebiet zur gesamten anrechenbaren Grundstücksfläche, ohne Berücksichtigung von einzelnen Parzellen: $Baumassenziffer_g = \frac{SUM(Baumasse_{Parzelle})}{SUM(Anrechenbare\ Grundstücksfläche_{Parzelle})}$
Überbauungsziffer	Das Verhältnis der gesamten überbauten Fläche auf einer Parzelle zur anrechenbaren Grundstücksfläche. $Überbauungsziffer = \frac{Überbaute\ Fläche}{Anrechenbare\ Grundstücksfläche}$ Die globale Überbauungsziffer entspricht dem Verhältnis der gesamten überbauten Fläche im Projektgebiet zur gesamten anrechenbaren Grundstücksfläche, ohne Berücksichtigung von einzelnen Parzellen: $Überbauungsziffer_g = \frac{SUM(Überbaute\ Fläche_{Parzelle})}{SUM(Anrechenbare\ Grundstücksfläche_{Parzelle})}$

5.2 Entwicklung und Verifizierung der Übertragbarkeit des Modells

5.2.1 Entwicklung Prozess: Langenthal Ringstrasse

Als Fallbeispiel für die Entwicklung des Vorgehensmodells diente ein Projektperimeter, der an der Ringstrasse im Zentrum der Stadt Langenthal liegt. Abbildung 35 zeigt eine Übersicht des Perimeters mit den Parzellen und Gebäudegrundrissen der Ist-Situation. Bei der Abbildung handelt es sich um einen Auszug aus einem generierten Bericht aus dem Vorgehensmodell.

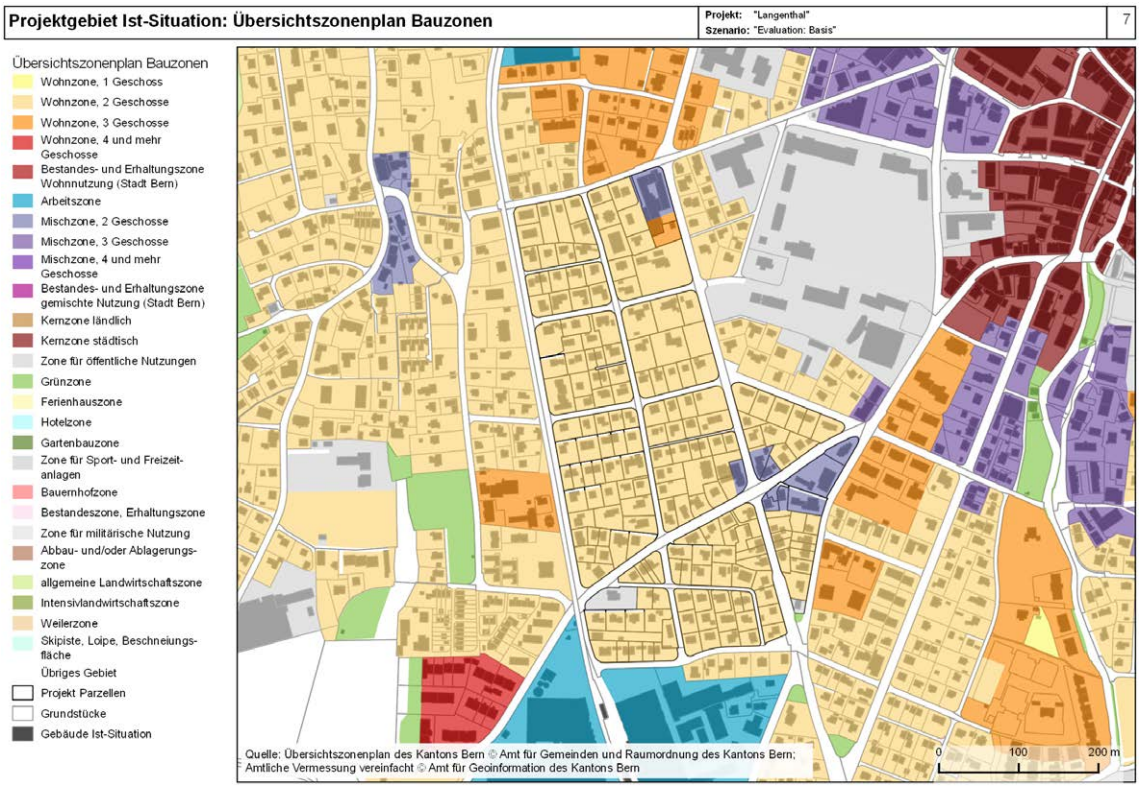
Der Perimeter an der Ringstrasse wurde ausgewählt, weil dieser im Rahmen einer Fallstudie zum Thema innere Verdichtung bereits vom Kompetenzbereich Dencity mit Methoden bearbeitet wurden, welche die Arbeit mit inspiriert haben (Gilgen 2016; Gilgen und Walczak 2017). Die bereits vorhandene Fallstudie stellt nicht nur die Vertrautheit der Experten mit dem ersten Projektperimeter sicher, sie lieferte zudem Anhaltspunkte für die Gestaltung der Szenarien zur Expertenbeurteilung und in eingeschränktem Mass auch Vergleichswerte für die Auswertung.



Quelle: eigene Darstellung

Abbildung 35: Übersicht Projektperimeter Langenthal Ringstrasse

Der Perimeter umfasst 199 Parzellen mit einer Gesamtfläche von rund 137'530m^2. Der Gebäudebestand besteht vor allem aus ein- und zweigeschossigen Wohngebäuden. Hinzu kommt ein kleinerer Teil des Perimeters, in dem Mischnutzung erlaubt ist. Zudem liegen zwei Parzellen in Zonen für öffentliche Nutzungen. Diese Verhältnisse sind anhand des Übersichtszonenplans in Abbildung 36 ebenfalls gut ersichtlich. Innerhalb des Perimeters wurden 18 Gebäude ausgemacht, die als Doppelfamilien oder Reihenhäuser eingestuft wurden, welche über Parzellengrenzen hinweg gebaut sind. Die zugehörigen Parzellen werden in verschiedenen Szenarien in Gruppen von zwei bis drei Parzellen zusammengefasst. Die 18 Gruppen umfassen insgesamt 38 Parzellen oder etwas über 19 Prozent der gesamten Parzellen.



Quelle: eigene Darstellung

Abbildung 36: Übersichtszonenplan (o.) und Einteilung Modellzonen (u.) des Projektperimeters Langenthal

Die Aufbereitung der Daten des Projektgebietes entspricht dem Vorgehen, das in Abbildung 11 auf Seite 48 gezeigt wurde. Die Umsetzung erfolgte mit Hilfe eines SQL Scripts (vergleiche Anhang B.3.1). Die anrechenbare Grundstücksfläche wurde auf Basis eines Verschnitts mit den Bauzonen des Übersichtszonenplans bestimmt. Diese summiert sich auf rund 135'760m^2 (d.h. praktisch die gesamte Parzellenfläche befindet sich innerhalb von Bauzonen). Für die Zuweisung der Modellzonen wurden die bestehenden Bauzonen gemäss der Zuordnung in der Tabelle 7 auf die fünf verfügbaren Modellzonen

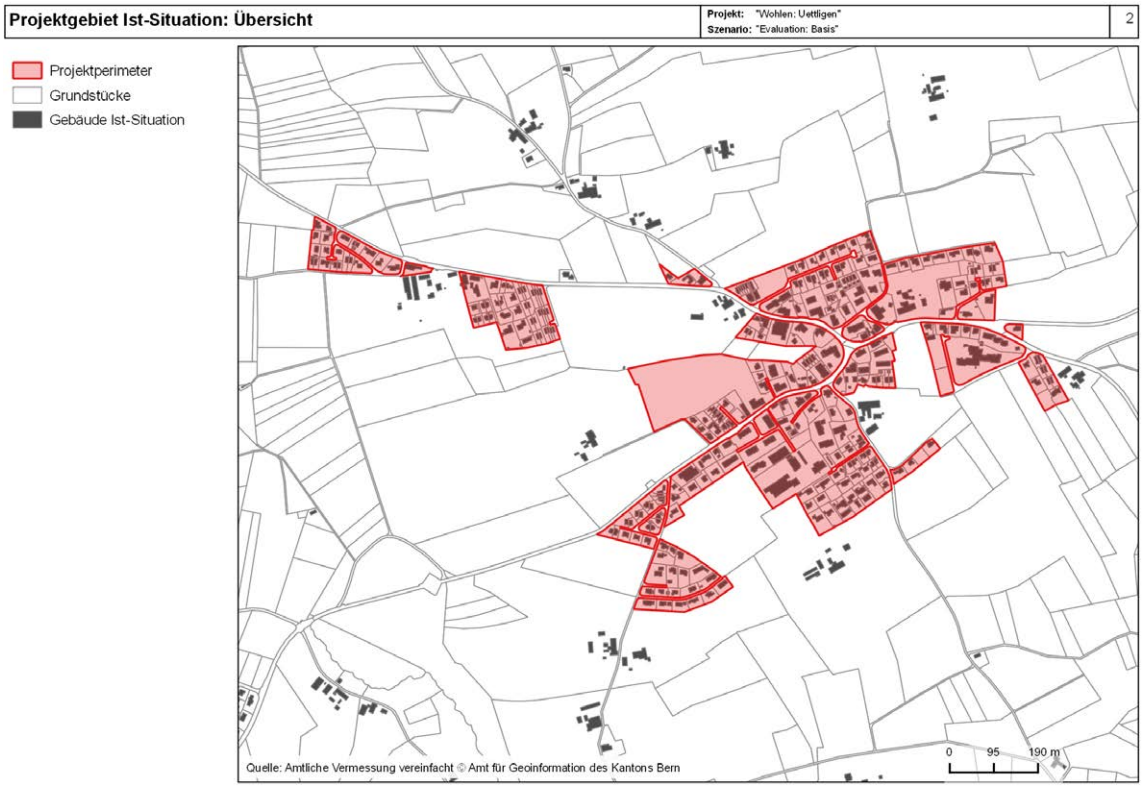
abgebildet. Dabei wurden Zonen mit ähnlichem Charakter in die gleiche Modellzone eingeordnet. Die resultierende Einteilung ist unten in Abbildung 36 zu sehen.

Tabelle 7: Zuordnung Bauzonen auf Modellzonen im Projektperimeter Langenthal Ringstrasse

Modellzone	Bauzone Ist-Situation
Z1	• Wohnzone W2/ B
Z2	• Wohnzone W2/ C
Z3	• Mischzone MZ 2
Z4	• UeO Nr. 41 "Areal Anliker" • UeO Nr. 41 "Areal Anliker" Baubereich A
Z5	• Zone für öffentliche Nutzung ZöN Art. 77 BauG • Alle weiteren Zonen

5.2.2 Verifizierung Übertragbarkeit: Wohlen Uettligen

Nachdem das Modell in einer ersten Version mit allen angestrebten Funktionen implementiert war, wurde dieses auf ein zweites Fallbeispiel angewendet, um die Übertragbarkeit zu prüfen. Als zweiter Perimeter wurde der Ortsteil Uettligen in Wohlen bei Bern ausgesucht, der in der Übersicht in Abbildung 37 zu sehen ist.

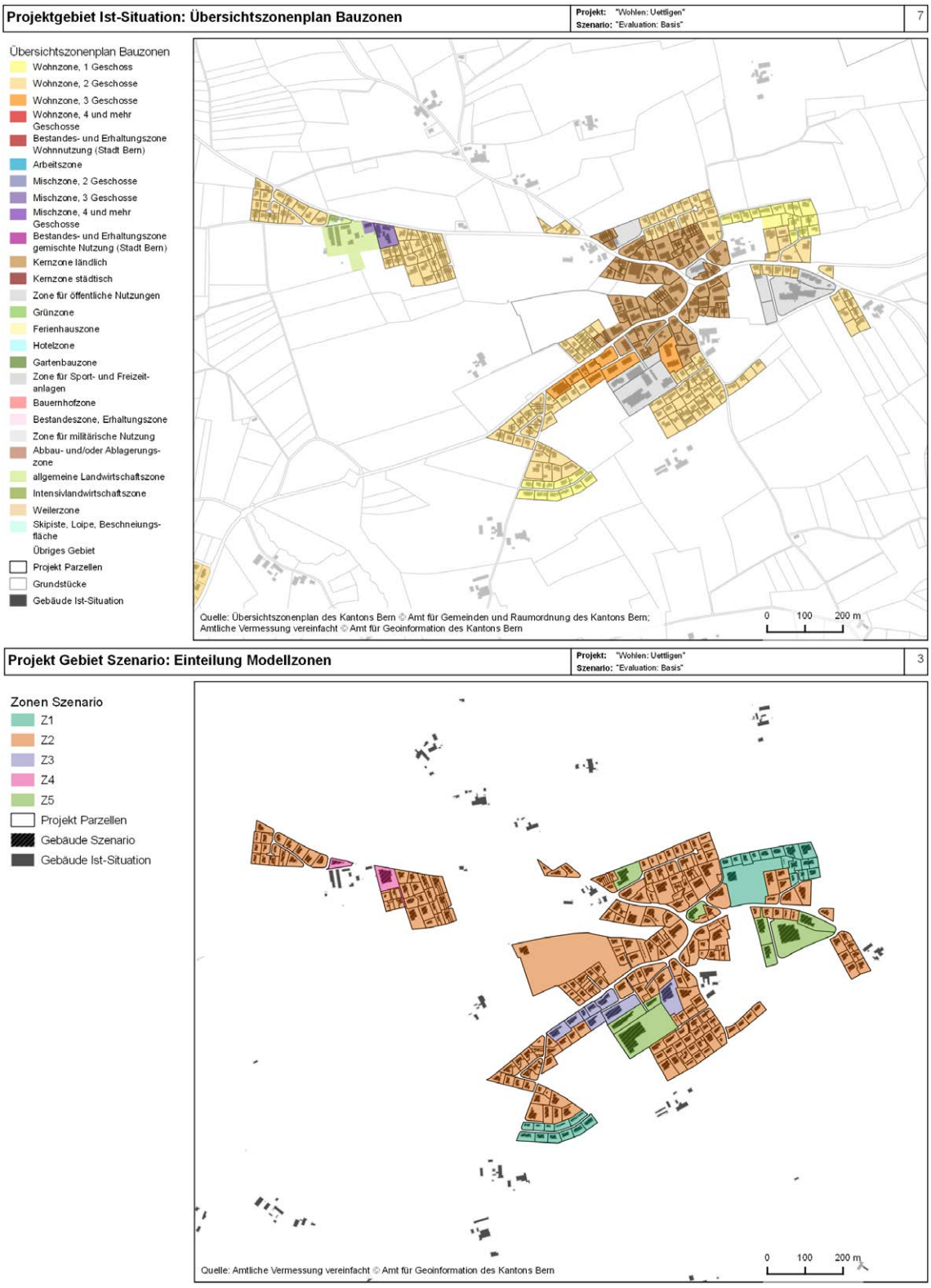


Quelle: eigene Darstellung

Abbildung 37: Übersicht Projektperimeter Wohlen Uettligen

Der Perimeter in Uettligen wurde ausgewählt, weil der Autor der Arbeit bereits aus anderen Projekten mit der Gemeinde Wohlen vertraut war. Die Gemeinde grenzt an die Stadt Bern und besteht aus mehreren unabhängigen Ortsteilen, welche periurbanen bis ländlichen Charakter aufweisen. Der Ortsteil Uettligen wurde ausgewählt, da er in Teilen der Bebauung zwar Parallelen zum Perimeter in Langenthal aufweist, aber durch seine eher ländliche Prägung auch abweichende Strukturen umfasst. Mit 327 Parzellen und einer Gesamtfläche von rund 321'616m^2 ist der Perimeter klar grösser als derjenige in Langenthal. Der Gebäudebestand wird auf der einen Seite von ein- bis zweigeschossigen Wohngebäuden dominiert. Hinzu kommt ein kleinerer Teil des Perimeters mit grösseren, dreigeschossigen Wohnbauten. Auf der anderen Seite ist eine Kernzone um den alten Dorfkern vorhanden, wo ein- bis dreigeschossige Gebäude zu finden sind. Neben Wohngebäuden umfassen diese Mischnutzungen sowie

Gewerbe. Hinzu kommen landwirtschaftliche Gebäude. Wie in der Übersicht in Abbildung 37 und anhand des Übersichtszonenplans in Abbildung 38 zu sehen ist, sind einige dieser Gebäude auf grösseren Parzellen, welche an sich nur teilweise in der Bauzone stehen. Abgerundet wird das Bild von mehreren Flächen für öffentliche Nutzungen (Schulen, Viehschauplatz, Altersheim).



Quelle: eigene Darstellung

Abbildung 38: Übersichtszonenplan (o.) und Einteilung Modellzonen (u.) des Projektperimeters Wohlen

Während die Struktur der Bebauung in den Wohnzonen grob mit derjenigen in Langenthal vergleichbar ist, sind diese Gebiete lockerer verstreut. Zudem ist ein höherer Anteil an Doppel- und Mehrfamilienhäusern vorhanden. Innerhalb des Perimeters wurden insgesamt 51 Gruppen von drei bis fünf Parzellen ausgemacht, welche in einigen Szenarien zusammengefasst werden. Diese Gruppen umfassen insgesamt 125 Parzellen (etwas über 38 Prozent).

Das Vorgehen für die Aufbereitung des Projektperimeters Wohlen Uettligen verlief analog zum Vorgehen in Langenthal. Das verwendete SQL Script ist in Anhang B.3.2 gelistet. Es wurden nur Parzellen in den Perimeter übernommen, welche zumindest zum Teil in einer Bauzone liegen. Parzellen mit landwirtschaftlichen Gebäuden ausserhalb der Bauzonen oder in der allgemeinen Landwirtschaftszone wurden nicht in den Perimeter übernommen. Die anrechenbare Grundstücksfläche wurde dann wiederum auf Basis eines Verschnitts mit den Bauzonen des Übersichtszonenplans bestimmt. Diese summiert sich auf rund 282'190m^2. Die klare Differenz zur gesamten Parzellenfläche ist der Tatsache geschuldet, dass einige Grundstücke nur teilweise innerhalb der Bauzone liegen. Für die Zuweisung der Modellzonen wurden die bestehenden Bauzonen gemäss der Zuordnung in der Tabelle 8 auf die fünf verfügbaren Modellzonen abgebildet. Dabei wurden Zonen mit ähnlichem Charakter in die gleiche Modellzone eingeordnet. Die resultierende Einteilung ist unten in Abbildung 37 zu sehen.

Tabelle 8: Zuordnung Bauzonen auf Modellzonen im Projektperimeter Wohlen Uettligen

Modellzone	Bauzone Ist-Situation
Z1	• Wohnzone 1-geschossig
Z2	• Dorfzone, 2 geschossig • Wohnzone 2-geschossig
Z3	• Wohnzone 3-geschossig
Z4	• Mischzone • Mischzone Landwirtschaft
Z5	• Zone für öffentliche Nutzung • Alle weiteren Zonen

5.2.3 Beurteilung der Übertragbarkeit und Nutzung des Vorgehensmodells

Die Übertragung des Vorgehensmodells vom Perimeter in Langenthal auf den zweiten Testperimeter in Wohlen verlief weitgehend problemlos und zügig. Der grösste Aufwand fiel während der Auswahl der Parzellen und der Definition des Mappings der Bauzonen auf die Modellzonen an, da das Vorgehen aus Langenthal individuell auf Wohlen Uettligen adaptiert werden musste. Dass sich die Übertragung so einfach gestaltete, ist wesentlich der Tatsache geschuldet, dass beide Perimeter im Kanton Bern liegen und die nötigen Inputdaten für Wohlen bereits im Rahmen der Ausarbeitung für Langenthal in die Datenbank integriert wurden. Entsprechend würde etwas mehr Aufwand anfallen, wenn ein Perimeter in einem anderen Kanton bearbeitet werden soll, da verschiedene Daten der Amtlichen Vermessung vorgängig neu integriert werden müssten. Eine weitergehende Recherche für das Beispiel des Kantons Zürich, der ähnlich wie Bern die Daten der Amtlichen Vermessung als Open Data frei zur Verfügung stellt, hat gezeigt, dass der Aufwand für die Integration dank Abstützung auf weitgehend identische Datenmodelle überschaubar wäre. Für den Kanton Zürich müssten bei der Integration insbesondere einige im Zürcher Modell zusätzlich hinzugefügten Felder ausgeschlossen und einige Felder anders zugeordnet werden. Zudem wären kleinere Anpassungen an der Datenaufbereitung nötig, da Zürich einige abweichende Codierungen bei der Bodenbedeckung verwendet.

Die Anwendung des Vorgehensmodells auf den Perimeter in Wohlen hat zudem einige Punkte aufgezeigt, welche für die Notwendigkeit individueller Anpassungen in gewissen Fällen sprechen. Infolge der wesentlich grösseren Ausdehnung des Perimeters in Wohlen im Vergleich zum Perimeter in Langenthal zeigten einige der Berichte in QGIS zunächst ungünstige Kartendarstellungen, welche durch Anpassungen am Template und den Generierungsroutinen weitgehend kompensiert werden konnten. Trotzdem sind die Berichte in der jetzigen Form nur etwa für Darstellungen mit einem Massstab von etwa 1:10'000 optimiert, womit Wohlen mit 1:8'800 schon nahe an der oberen Grenze liegt. Bei noch grösseren Perimetern wären zusätzliche Anpassungen am Template beziehungsweise die Erstellung eines spezifischen Projekttemplates mit entsprechenden Optimierungen in den Berichten nötig (u.a. Seitengrösse, Grösse der Kartendarstellung, Massstäbe, Sichtbarkeit und Gestaltung der Kartogramme).

Trotz Einschränkungen bei den Berichten haben sich die Auswertungen in QGIS als gut übertragbar erwiesen. Insbesondere im Rahmen der Definition der Szenarien für die Expertenbeurteilung wurden die Auswertungen intensiv genutzt, um in Wohlen eine Reihe von Parametern abzuschätzen. Während die Layer zur Ist-Situation kombiniert mit den Histogrammen aus R der Abschätzung dienten, konnten die Layer zur Auswertung der Szenarien im Rahmen der iterativen Verfeinerung der Parametrisierung der Szenarien zur Kontrolle genutzt werden. Dabei zeigte sich bereits, dass die generierten Varianten des Modells die Ist-Situation annähern, oft aber leicht unterhalb der erlaubten Dichten liegen. Wie die weiteren Arbeiten zeigten, liegt dies an einem Zusammenspiel der Parameter, der Form der Parzellen im Input und Eigenheiten des Modells. Insgesamt wird aufgrund der Erfahrungen mit dem Perimeter Wohlen Uettligen und den weitergehenden Recherchen zum Kanton Zürich die Übertragbarkeit des Modells als gegeben angesehen.

5.3 Beurteilung Modelloutputs aus Expertensicht

5.3.1 Vorgehen

Für die Beurteilung der Modelloutputs aus Expertensicht wurden für die beiden Projektperimeter in Langenthal und Wohlen eine Reihe von Szenarien definiert, mittels des parametrischen Modells aufbereitet und schliesslich den Planern und Architekten des Dencity als Input zur Beurteilung vorgelegt. Die Abläufe für die Aufbereitung der Szenarien dienten dabei auch als Test des Vorgehensmodells im Allgemeinen. Zunächst wurden für den Perimeter in Langenthal fünf Szenarien «Basis», «Basis ohne Ausnützungsziffer», «Basis + Grünflächenziffer 0.7», «W3» sowie «W3 + Abstandslinien» definiert, welche an Szenarien aus der Fallstudie Langenthal des Dencity angelehnt sind (Gilgen und Walczak 2017). Für den Perimeter in Wohlen wurden lokal adaptierte Versionen der vier ersten Langenthaler Szenarien erstellt. Die genauen Parameter wurden in einem iterativen Prozess festgelegt. Hierfür wurden im parametrischen Modell zunächst in einem Testszenario die groben Werte eingestellt. Die vom Modell generierten Auswertungen in Form von QGIS Projekten dienten dabei der Abschätzung von Parametern auf Basis der Ist-Situation. Die Auswertungen zu den Szenarien dienten hingehen der Kontrolle und schrittweisen Verfeinerung der Parameterwerte. Des Weiteren wurden die QGIS Projekte interaktiv genutzt, um die Nummern der zu gruppierenden Parzellen zu identifizieren.

Die vorgeschlagenen Szenarien wurden dem Dencity zu einer ersten Plausibilisierung vorgelegt, um abzuklären, ob deren Konzeption und deren Parameter Sinn machen. Im Anschluss wurden die Szenarien mit Hilfe des parametrischen Modells aufbereitet. Zuerst wurden die Szenarien in der Datenbank angelegt, anschliessend in Grasshopper die Parameterwerte entsprechend der Konzeption eigestellt und die Resultate gespeichert. Danach wurden den Experten des Dencity relevante Teile der Berichte aus den für die Szenarien generierten QGIS Projekten zusammen mit weiteren Auswertungen von Dichteziffern als Input für die Beurteilung zur Verfügung gestellt. Diese Inputs werden in den Abschnitten 5.3.2 und 5.3.3 dokumentiert. In Abschnitt 5.3.4 folgt eine Auswertung und Interpretation der Szenarien aus Sicht des Autors und in Abschnitt 5.3.5 werden die Resultate des Expertenworkshops zusammengefasst.

5.3.2 Szenarien Langenthal Ringstrasse

5.3.2.1 Übersicht

Die nachfolgenden Unterabschnitte dokumentieren die Szenarien für den Perimeter Langenthal Ringstrasse. Neben der Parametrisierung wird jeweils eine kurze Übersicht zu den resultierenden Varianten gegeben, wie sie in ähnlicher Form den Experten des Dencity zur Verfügung gestellt wurden. Die Parameter Geschosshöhe, Anteil Arbeitsnutzung, Platzbedarf Arbeitsnutzung, Platzbedarf Wohnnutzung, Gruppierung und Dachform wurden für alle Szenarien konstant gehalten. Der Anteil Arbeitsnutzung wurde pro Zone auf Grund der Ist-Situation abgeschätzt. Die Werte für Platzbedarf Arbeitsnutzung und Platzbedarf Wohnnutzung sind ebenfalls geschätzte Durchschnittswerte, welche aber für alle Szenarien und Modellzonen identisch sind, da die Anzahl Einwohner und Beschäftigte in der Auswertung nur am Rande betrachtet wurden. In der Realität wären hingegen lokale Abweichungen zu erwarten. Die maximale Gesamthöhe wurde jeweils so gesetzt, dass die volle Anzahl Geschosse und ein Dach realisiert werden können (d.h. der Parameter hat keine einschränkende Wirkung).

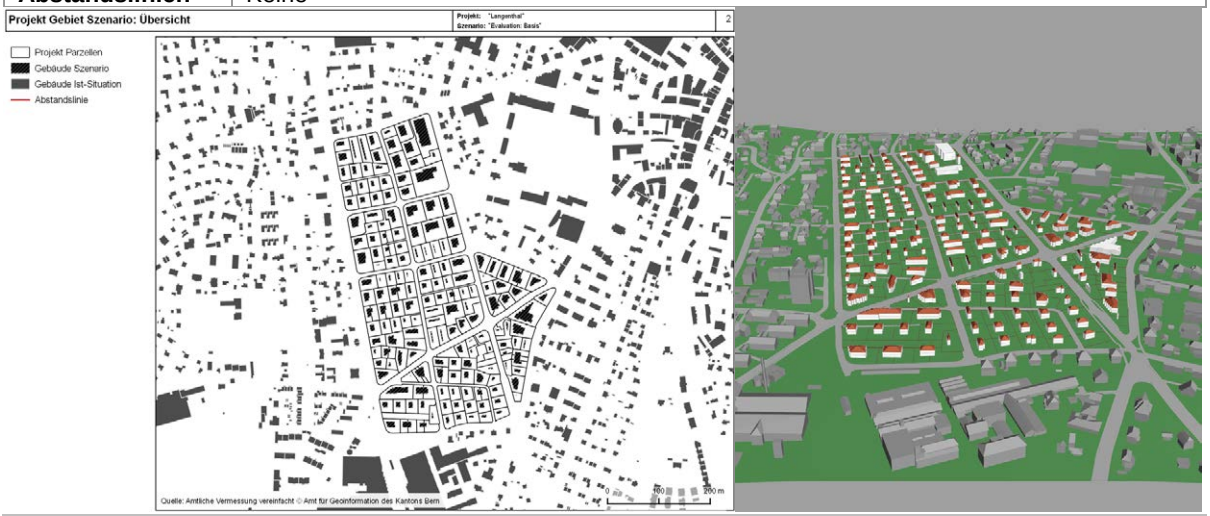
5.3.2.2 Szenario «Basis»

Dieses an die Ist-Situation angelehnte Szenario bildet den Ausgangspunkt für die anderen Szenarien. Für die Modellzonen Z1, Z2 und Z3 wurden bei den Grenzabständen, Geschosszahl und Nutzungsziffer

die Werte aus dem Bestandes-Szenario der Fallstudie Langenthal als Referenz verwendet. Allerdings wurden die Werte der Nutzungsziffern als Ausnützungsart *GFZ* im Sinne des Modells interpretiert und nicht als *Ausnützungsziffer* (einer weiteren Dichtekennziffer, die mit der IVHB weitgehend durch die Geschossflächenziffer abgelöst wird). Für die Modellzonen Z4 und Z5 wurden die Grenzabstände identisch gesetzt wie in den anderen Zonen. Die Nutzungsziffern und Geschosszahlen wurden auf Basis der Auswertung zur Ist-Situation geschätzt. Die Ausnützungsart *BMZ* in der Modellzone Z4 entspricht nicht der Situation in der Realität, sondern wurde als Test der entsprechenden Funktion des Modells so gewählt, da es sich um ehemalige Industriebauten handelt und diese Ausnützungsart für Gewerbe- und Arbeitszonen geeignet ist (BSIG, 2011, 16). Aufgrund der Nutzungsart in der Realität wurden in den Modellzonen Z4 und Z5 auch die Geschosshöhen leicht höher gesetzt. Insgesamt resultiert durch die Parametrisierung eine Situation, welche nahe am maximal möglichen Ausbau im aktuellen Bestand im Perimeter Ringstrasse liegen sollte.

Tabelle 9: Langenthal Ringstrasse Szenario «Basis»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	GFZ	GFZ	GFZ	BMZ	GFZ
Nutzungsziffer	0.4	0.5	0.6	6	0.5
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	2	2	2	4	2
Geschosshöhe	2.8m	2.8m	2.8m	3m	3m
Gesamthöhe	10m	10m	10m	17m	10m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0.1	0.2	0.8	1	1
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				



- Anrechenbare Grundstücksfläche Total [m2]: 135'763
- Geschossfläche Total [m2]: 46'284
- Bauvolumen Total [m3]: 184'694
- Überbaute Fläche Total [m2]: 21'732

- Geschossflächenziffer$_g$: 0.341
- Geschossflächenziffer$_{avg}$: 0.290
- Baumassenziffer$_g$: 1.360
- Baumassenziffer$_{avg}$: 1.198
- Überbauungsziffer$_g$: 0.160
- Überbauungsziffer$_{avg}$: 0.142

Quelle: eigene Darstellung

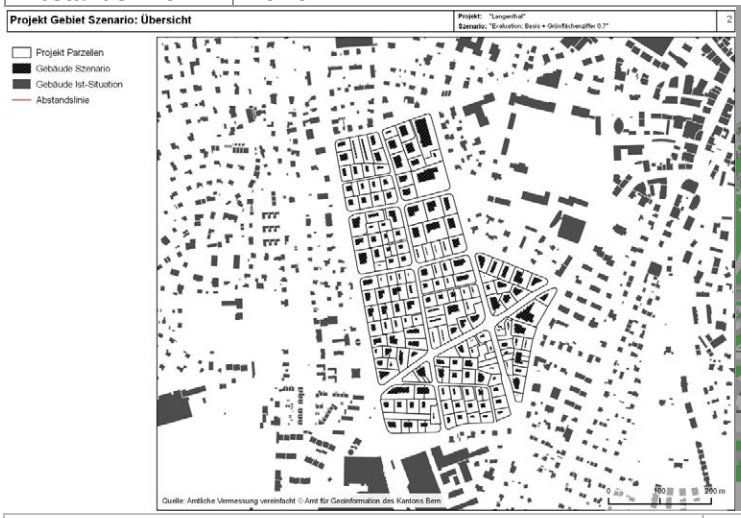
Abbildung 39: Übersicht Modelloutput Langenthal Ringstrasse Szenario «Basis»

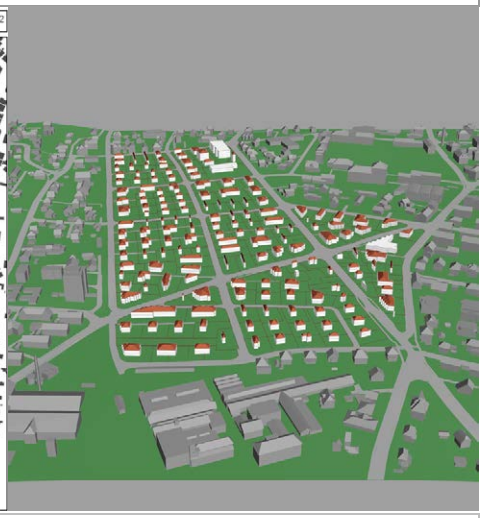
5.3.2.3 Szenario «Basis + Grünflächenziffer 0.7»

In diesem Szenario soll aufgezeigt werden, welche Dichte realisiert werden kann, wenn die aktuellen Parameter beibehalten werden, aber ein Minimum an nicht überbauter Freifläche festgelegt wird. In einem analogen Szenario der Fallstudie wurde hierfür in den Zonen mit Wohnnutzung eine Freiflächenziffer von 0.7 festgelegt (d.h. mindestens 70% der Parzellenfläche müssen frei bleiben). Im Szenario wird die sich weitgehend analog auswirkende Grünflächenziffer in den Modellzonen Z1, Z2 und Z3 auf den Wert 0.7 gesetzt. Die anderen Parameter sowie die Parametrisierung der Modellzonen Z4 und Z5 werden wiederum gleich belassen wie im Szenario «Basis».

Tabelle 10: Langenthal Ringstrasse Szenario «Basis + Grünflächenziffer 0.7»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	GFZ	GFZ	GFZ	BMZ	GFZ
Nutzungsziffer	0.4	0.5	0.7	6	0.5
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	2	2	2	4	2
Geschosshöhe	2.8m	2.8m	2.8m	3m	3m
Gesamthöhe	10m	10m	10m	17m	10m
Grünflächenziffer	0.7	0.7	0.7	0	0
Anteil Arbeitsnutzung	0.1	0.2	0.8	1	1
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				





- Anrechenbare Grundstücksfläche Total [m2]: 135'763
- Geschossfläche Total [m2]: 46'284
- Veränderung Geschossfläche im Vergleich Szenario zu «Basis»: 0%
- Bauvolumen Total [m3]: 184'694
- Veränderung Bauvolumen im Vergleich Szenario zu «Basis»: 0%
- Überbaute Fläche Total [m2]: 21'732
- Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»: 0%

- Geschossflächenziffer$_g$: 0.341
- Geschossflächenziffer$_{avg}$: 0.290
- Baumassenziffer$_g$: 1.360
- Baumassenziffer$_{avg}$: 1.198
- Überbauungsziffer$_g$: 0.160
- Überbauungsziffer$_{avg}$: 0.142

Quelle: eigene Darstellung

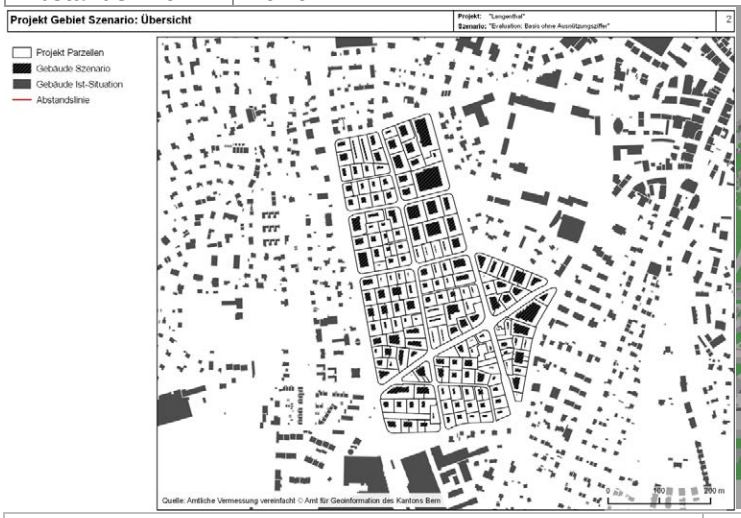
Abbildung 40: Übersicht Modelloutput Langenthal Ringstrasse Szenario «Basis + Grünflächenziffer 0.7»

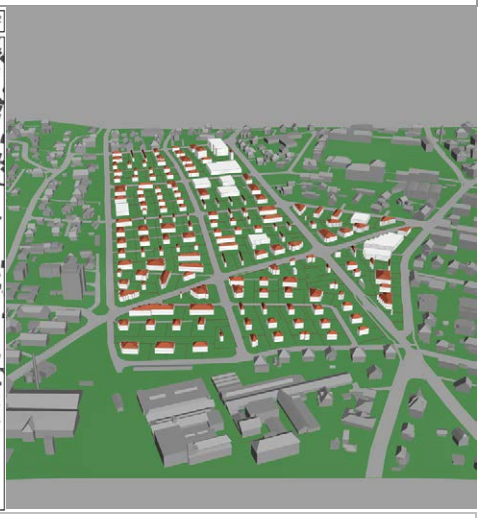
5.3.2.4 Szenario «Basis ohne Ausnützungsziffer»

Wie in einem analogen Szenario in der Fallstudie Langenthal soll aufgezeigt werden, welche Dichte realisiert werden kann, wenn die Parameter der aktuellen Situation beibehalten werden, aber die Ausnützung nicht beschränkt wird. Wie in der Fallstudie wird diese Änderung nur auf die Zonen mit überwiegender Wohnnutzung angewendet. Deshalb wird in den Modellzonen Z1, Z2 und Z3 die Ausnützungsart auf den Wert «Keine» gesetzt, die anderen Parameter wie Grenzabstände, Geschosszahl und Gesamthöhe aber so wie im Szenario «Basis» belassen. Analog zur Fallstudie wird die Parametrisierung der Modellzonen Z4 und Z5 beibehalten.

Tabelle 11: Langenthal Ringstrasse Szenario «Basis ohne Ausnützungsziffer»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	Keine	Keine	Keine	BMZ	GFZ
Nutzungsziffer	0	0	0	6	0.5
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	2	2	2	4	2
Geschosshöhe	2.8m	2.8m	2.8m	3m	3m
Gesamthöhe	10m	10m	10m	17m	10m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0.1	0.2	0.8	1	1
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				





- Anrechenbare Grundstücksfläche Total [m2]: 135'763
- Geschossfläche Total [m2]: 54'341
- Veränderung Geschossfläche im Vergleich Szenario zu «Basis»: 17%
- Bauvolumen Total [m3]: 204'935
- Veränderung Bauvolumen im Vergleich Szenario zu «Basis»: 11%
- Überbaute Fläche Total [m2]: 25'760
- Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»: 19%

- Geschossflächenziffer$_g$: 0.400
- Geschossflächenziffer$_{avg}$: 0.323
- Baumassenziffer$_g$: 1.510
- Baumassenziffer$_{avg}$: 1.293
- Überbauungsziffer$_g$: 0.190
- Überbauungsziffer$_{avg}$: 0.158

Quelle: eigene Darstellung

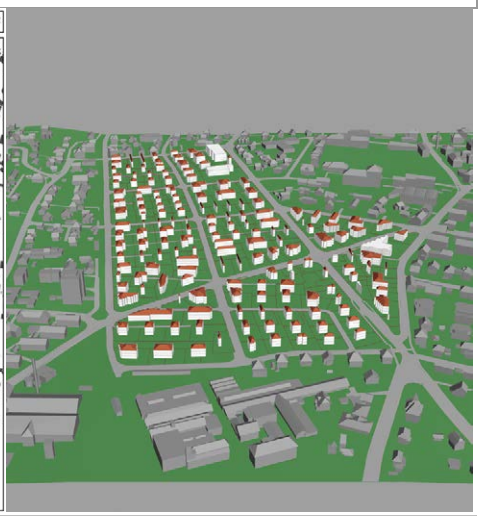
Abbildung 41: Übersicht Modelloutput Langenthal Ringstrasse Szenario «Basis ohne Ausnützungsziffer»

5.3.2.5 Szenario «W3»

Das Szenario «W3» soll aufzeigen, welche Dichte mit einer Aufzonung in den Wohnzonen realisiert werden kann. Die Verwendung einer W3 Zone erfolgt in Anlehnung an die Empfehlungen des Rats für Raumordnung (ROR 2012). Durch Bauen in die Höhe soll mehr Dichte realisiert werden, ohne Freiraum zu opfern. In den Modellzonen Z1, Z2 und Z3 wird entsprechend die Geschosszahl auf 3 erhöht. Gleichzeitig werden die Nutzungsziffer und die maximale Gesamthöhe jeweils rund um ein Drittel erhöht, um tatsächlich eine dichtere Bebauung realisieren zu können. Die anderen Parameter wie z.B. die Grenzabstände sowie die Parameter der Modellzonen Z4 und Z5 werden aus dem Szenario «Basis» übernommen.

Tabelle 12: Langenthal Ringstrasse Szenario «W3»

Parameter	Z1	Z2	Z3	Z4	Z5
Art der Ausnützung	GFZ	GFZ	GFZ	BMZ	GFZ
Nutzungsziffer	0.6	0.75	0.9	6	0.5
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	3	3	3	4	2
Geschosshöhe	2.8m	2.8m	2.8m	3m	3m
Gesamthöhe	13m	13m	13m	17m	10m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0.1	0.2	0.8	1	1
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				



- Anrechenbare Grundstücksfläche Total [m2]: 135'763
- Geschossfläche Total [m2]: 66'207
- Veränderung Geschossfläche im Vergleich Szenario zu «Basis»: 43%
- Bauvolumen Total [m3]: 240'477
- Veränderung Bauvolumen im Vergleich Szenario zu «Basis»: 30%
- Überbaute Fläche Total [m2]: 21'732
- Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»: 0%

- Geschossflächenziffer$_g$: 0.488
- Geschossflächenziffer$_{avg}$: 0.427
- Baumassenziffer$_g$: 1.771
- Baumassenziffer$_{avg}$: 1.582
- Überbauungsziffer$_g$: 0.160
- Überbauungsziffer$_{avg}$: 0.142

Quelle: eigene Darstellung

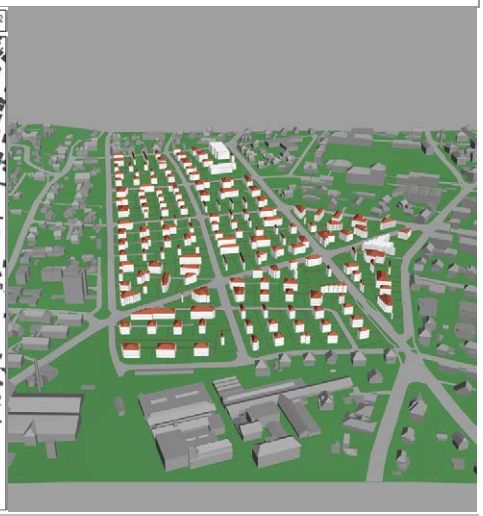
Abbildung 42: Übersicht Modelloutput Langenthal Ringstrasse Szenario «W3»

5.3.2.6 Szenario «W3 + Abstandslinien»

Das Szenario «W3 + Abstandslinien» orientiert sich lose an einem Szenario der Fallstudie, welches am Kreisel Ringstrasse/Thunstettenstrasse (grosse Kreuzung rechts aussen im Projektperimeter) eine Verdichtung hin zur Strasse sowie die Verwendung von Baulinien zur Festlegung des Abstandes zur Strasse vorsieht. An der Kreuzung wurden hin zur Strasse Abstandslinien mit einer Distanz von etwa 2m erstellt. Dieser Wert wäre in der Realität zu klein, ist aber geeignet zum Test der Auswirkung solcher Abstandslinien. Während in der Fallstudie bei Gebäuden an der Strasse höhere Ausnutzungen zugelassen wurden, ist dies im parametrischen Modell nicht möglich. Stattdessen wurden abgesehen von den Abstandslinien die Parameter des Szenarios «W3» beibehalten.

Tabelle 13: Langenthal Ringstrasse Szenario «W3 + Abstandslinien»

Parameter	Z1	Z2	Z3	Z4	Z5
Art der Ausnützung	GFZ	GFZ	GFZ	BMZ	GFZ
Nutzungsziffer	0.6	0.75	0.9	6	0.5
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	3	3	3	4	2
Geschosshöhe	2.8m	2.8m	2.8m	3m	3m
Gesamthöhe	13m	13m	13m	17m	10m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0.1	0.2	0.8	1	1
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Abstandslinien mit ca. 2m Distanz zur Strasse am Kreisel Ringstrasse/Thunstettenstrasse.				



• Anrechenbare Grundstücksfläche Total [m2]:	135'763	• Geschossflächenziffer$_g$:	0.499
• Geschossfläche Total [m2]:	67'770	• Geschossflächenziffer$_{avg}$:	0.439
• Veränderung Geschossfläche im Vergleich Szenario zu «Basis»:	46%	• Baumassenziffer$_g$:	1.815
• Bauvolumen Total [m3]:	246'402	• Baumassenziffer$_{avg}$:	1.629
• Veränderung Bauvolumen im Vergleich Szenario zu «Basis»:	33%	• Überbauungsziffer$_g$:	0.164
• Überbaute Fläche Total [m2]:	22'253	• Überbauungsziffer$_{avg}$:	0.146
• Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»:	2%		

Quelle: eigene Darstellung

Abbildung 43: Übersicht Modelloutput Langenthal Ringstrasse Szenario «W3 + Abstandslinien»

5.3.3 Szenarien Wohlen Uettligen

5.3.3.1 Übersicht

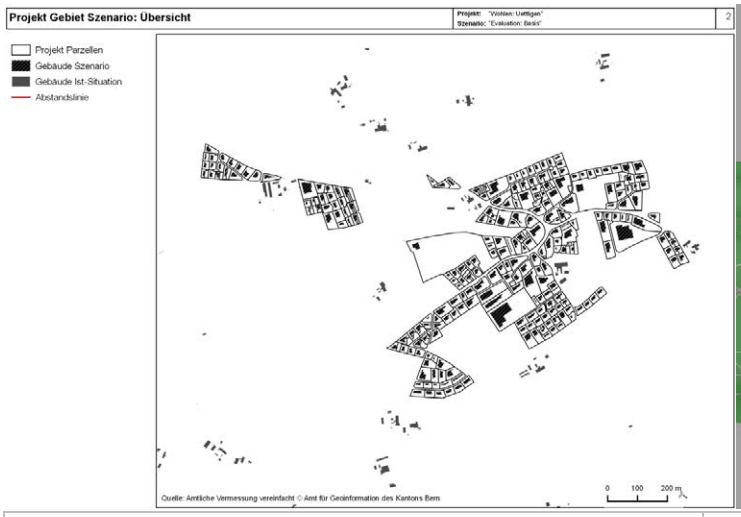
Die nachfolgenden Unterabschnitte geben die Parametrisierungen für den Perimeter Wohlen Uettligen wieder. Diese sind unmittelbar an die vier ersten Szenarien des Perimeters in Langenthal angelehnt. Während einige Parameter wie Grenzabstände, Geschosshöhe, Platzbedarf Arbeitsnutzung und Platzbedarf Wohnnutzung aus den Langenthaler Szenarien übernommen wurden, sind andere auf Basis der Ist-Situation in Wohlen festgelegt worden. Grundsätzlich verfolgen die gleichnamigen Szenarien in den beiden Perimetern aber jeweils die gleichen Ziele und sollen zumindest eine eingeschränkte Vergleichbarkeit ermöglichen.

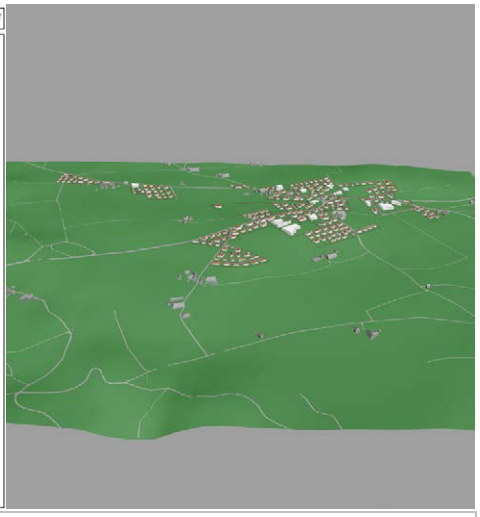
5.3.3.2 Szenario «Basis»

Das Szenario «Basis» ist ähnlich wie das gleichnamige Szenario in Langenthal eine Annäherung an den maximal möglichen Ausbau in der Ist-Situation und bildet den Ausgangspunkt für die anderen Szenarien im Projektperimeter. Die Grenzabstände wurden von den Zonen mit Wohnnutzung in Langenthal übernommen. Als Art der Ausnutzung wurde überall *GFZ* angenommen und die Nutzungsziffer auf Basis der Ist-Situation geschätzt. Die Anzahl Geschosse wurde möglichst dem Übersichtszonenplan entnommen oder aber abgeschätzt. Der Anteil Arbeitsnutzung wurde auf Basis der Auswertung zur Ist-Situation ebenso abgeschätzt. Die Geschosshöhe, der Platzbedarf Arbeitsnutzung und der Platzbedarf Wohnnutzung wurden aus Langenthal übernommen. Die maximale Gesamthöhe wurde wiederum so gesetzt, dass die volle Geschosszahl plus Dach realisiert werden kann.

Tabelle 14: Wohlen Uettligen Szenario «Basis»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	GFZ	GFZ	GFZ	GFZ	GFZ
Nutzungsziffer	0.3	0.4	0.7	0.6	0.7
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	1	2	3	2	3
Geschosshöhe	2.8m	2.8m	2.8m	2.8m	2.8m
Gesamthöhe	7m	10m	13m	10m	13m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0	0.2	0.1	0.8	0.2
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				





• Anrechenbare Grundstücksfläche Total [m2]:	282'196	• Geschossflächenziffer$_g$:	0.423
• Geschossfläche Total [m2]:	119'333	• Geschossflächenziffer$_{avg}$:	0.351
• Bauvolumen Total [m3]:	453'055	• Baumassenziffer$_g$:	1.605
• Überbaute Fläche Total [m2]:	55'780	• Baumassenziffer$_{avg}$:	1.455
		• Überbauungsziffer$_g$:	0.198
		• Überbauungsziffer$_{avg}$:	0.178

Quelle: eigene Darstellung

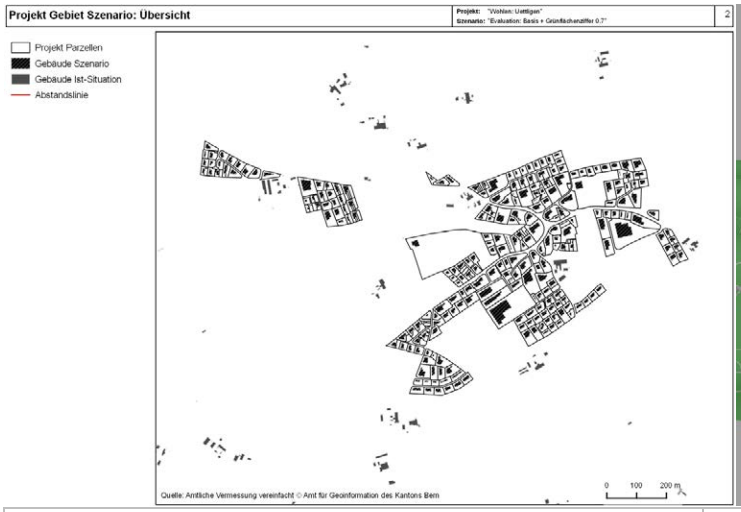
Abbildung 44: Übersicht Modelloutput Wohlen Uettligen Szenario «Basis»

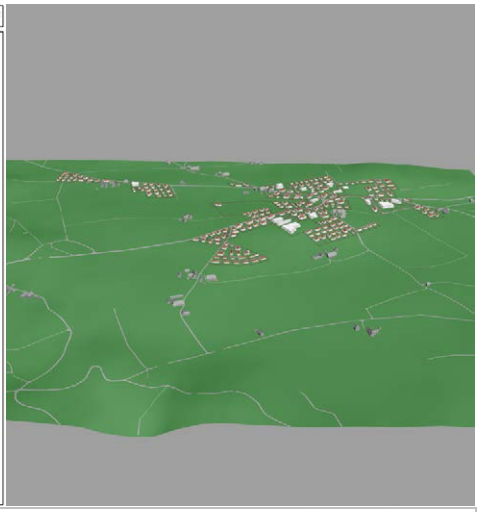
5.3.3.3 Szenario «Basis + Grünflächenziffer 0.7»

Das Szenario «Basis + Grünflächenziffer 0.7» zeigt wiederum auf, welche Dichte unter Festlegung einer Grünflächenziffer von 0.7 und Beibehaltung der restlichen Parameter realisiert werden kann. Entsprechend werden die Parameter des Szenarios «Basis» beibehalten, aber in allen Modellzonen die Grünflächenziffer auf den Wert 0.7 gesetzt.

Tabelle 15: Wohlen Uettligen Szenario «Basis + Grünflächenziffer 0.7»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	GFZ	GFZ	GFZ	GFZ	GFZ
Nutzungsziffer	0.3	0.4	0.7	0.6	0.7
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	1	2	3	2	3
Geschosshöhe	2.8m	2.8m	2.8m	2.8m	2.8m
Gesamthöhe	7m	10m	13m	10m	13m
Grünflächenziffer	0.7	0.7	0.7	0.7	0.7
Anteil Arbeitsnutzung	0	0.2	0.1	0.8	0.2
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				





• Anrechenbare Grundstücksfläche Total [m2]:	282'196	• Geschossflächenziffer$_g$:	0.423
• Geschossfläche Total [m2]:	119'333	• Geschossflächenziffer$_{avg}$:	0.351
• Veränderung Geschossfläche im Vergleich Szenario zu «Basis»:	0%	• Baumassenziffer$_g$:	1.605
• Bauvolumen Total [m3]:	453'055	• Baumassenziffer$_{avg}$:	1.455
• Veränderung Bauvolumen im Vergleich Szenario zu «Basis»:	0%	• Überbauungsziffer$_g$:	0.198
• Überbaute Fläche Total [m2]:	55'780	• Überbauungsziffer$_{avg}$:	0.178
• Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»:	0%		

Quelle: eigene Darstellung

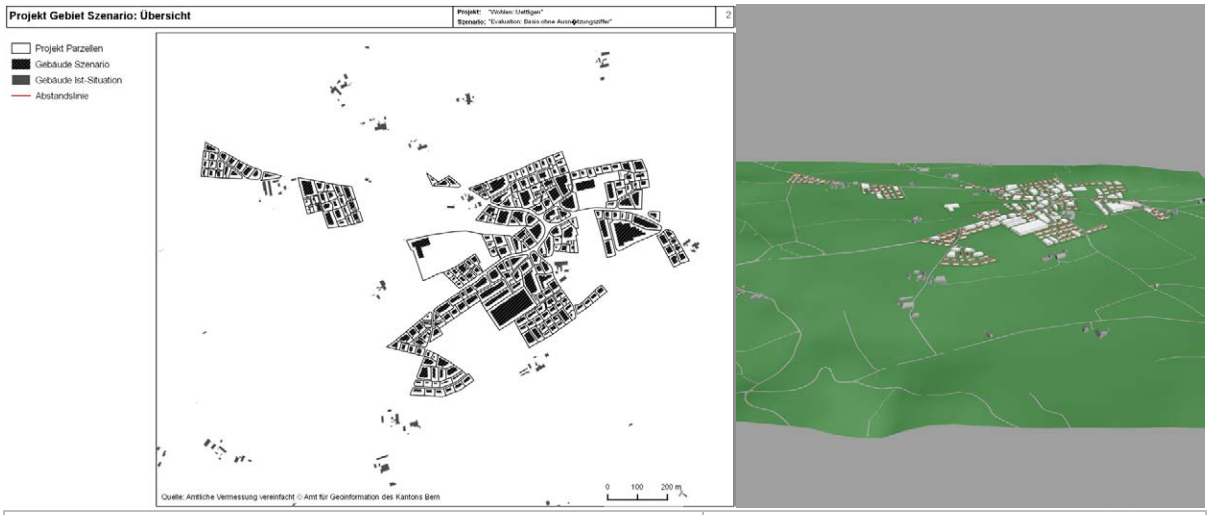
Abbildung 45: Übersicht Modelloutput Wohlen Uettligen Szenario «Basis + Grünflächenziffer 0.7»

5.3.3.4 Szenario «Basis ohne Ausnützungsziffer»

Das Szenario «Basis ohne Ausnützungsziffer» ist analog zum gleichnamigen Szenario in Langenthal angelegt und soll die realisierbare Dichte unter Beibehaltung der Parameter der aktuellen Situation aber ohne Beschränkung der Ausnützung aufzeigen. Konkret wird hierfür in allen Modellzonen die Art der Ausnützung auf den Wert «Keine» gesetzt, die anderen Parameter wie Grenzabstände, Geschosszahl und Gesamthöhe aber wie im Szenario «Basis» belassen.

Tabelle 16: Wohlen Uettligen Szenario «Basis ohne Ausnützungsziffer»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	Keine	Keine	Keine	Keine	Keine
Nutzungsziffer	0	0	0	0	0
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	1	2	3	2	3
Geschosshöhe	2.8m	2.8m	2.8m	2.8m	2.8m
Gesamthöhe	7m	10m	13m	10m	13m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0	0.2	0.1	0.8	0.2
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				



• Anrechenbare Grundstücksfläche Total [m2]:	282'196	• Geschossflächenziffer$_g$:	0.719
• Geschossfläche Total [m2]:	202'892	• Geschossflächenziffer$_{avg}$:	0.512
• Veränderung Geschossfläche im Vergleich Szenario zu «Basis»:	70%	• Baumassenziffer$_g$:	2.341
• Bauvolumen Total [m3]:	660'589	• Baumassenziffer$_{avg}$:	1.881
• Veränderung Bauvolumen im Vergleich Szenario zu «Basis»:	46%	• Überbauungsziffer$_g$:	0.325
• Überbaute Fläche Total [m2]:	91'715	• Überbauungsziffer$_{avg}$:	0.255
• Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»:	64%		

Quelle: eigene Darstellung

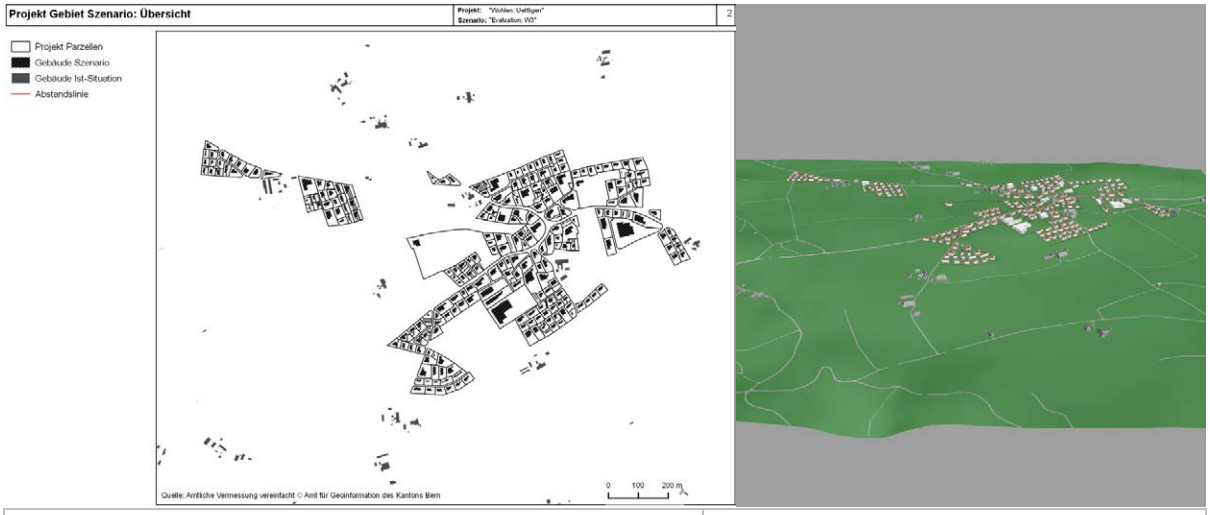
Abbildung 46: Übersicht Modelloutput Wohlen Uettligen Szenario «Basis ohne Ausnützungsziffer»

5.3.3.5 Szenario «W3»

Das Szenario «W3» zeigt analog zu Langenthal auf, welche Dichte mit einer Aufzonung realisiert werden kann. Wiederum wird eine Wz Zone angenommen, um mehr Dichte durch Bauen in die Höhe zu realisieren. In allen Modellzonen eine Geschosszahl von 3 parametrisiert und die maximale Gesamthöhe hinaufgesetzt, wodurch es in allen Modellzonen zu einer Angleichung kommt. Wo im Szenario «Basis» eine Geschosszahl kleiner 3 festgelegt war, wird zwar ebenfalls die Nutzungszahl entsprechend erhöht, es bleiben hier aber Unterschiede erhalten. Insbesondere für die W1 und W2 Quartiere, die von Ein- und Mehrfamilienhäusern dominiert werden, wird eine etwas geringere Dichte beibehalten.

Tabelle 17: Wohlen Uettligen Szenario «W3»

Parameter	**Z1**	**Z2**	**Z3**	**Z4**	**Z5**
Art der Ausnützung	GFZ	GFZ	GFZ	GFZ	GFZ
Nutzungsziffer	0.5	0.6	0.7	0.7	0.7
Grosser Grenzabstand	10m	10m	10m	10m	10m
Kleiner Grenzabstand	4m	4m	4m	4m	4m
Geschosszahl	3	3	3	3	3
Geschosshöhe	2.8m	2.8m	2.8m	2.8m	2.8m
Gesamthöhe	13m	13m	13m	13m	13m
Grünflächenziffer	0	0	0	0	0
Anteil Arbeitsnutzung	0	0.2	0.1	0.8	0.2
Platzbedarf Arbeitsnutzung	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person	80m^2/Person
Platzbedarf Wohnnutzung	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person	60m^2/Person
Gruppierung	Pro Zone Gruppen des Typs «Reihe» wie in der Realität				
Dachform	Keine Parametrisierung/Modelldefault				
Abstandslinien	Keine				



• Anrechenbare Grundstücksfläche Total [m2]:	282'196	• Geschossflächenziffer$_g$:	0.579
• Geschossfläche Total [m2]:	163'317	• Geschossflächenziffer$_{avg}$:	0.523
• Veränderung Geschossfläche im Vergleich Szenario zu «Basis»:	37%	• Baumassenziffer$_g$:	2.036
• Bauvolumen Total [m3]:	574'477	• Baumassenziffer$_{avg}$:	1.929
• Veränderung Bauvolumen im Vergleich Szenario zu «Basis»:	46%	• Überbauungsziffer$_g$:	0.193
• Überbaute Fläche Total [m2]:	54'439	• Überbauungsziffer$_{avg}$:	0.174
• Veränderung überbaute Fläche im Vergleich Szenario zu «Basis»:	-2%		

Quelle: eigene Darstellung

Abbildung 47: Übersicht Modelloutput Wohlen Uettligen Szenario «W3»

5.3.4 Auswertung Szenarien

Betrachtet man die Dichtekennzahlen der beiden Szenarien in der Übersicht in Tabelle 18, so zeigen sich in beiden Perimetern ähnliche Tendenzen. Der Durchschnitt der Kennzahlen, der auf Basis der Werte der Einzelparzellen gebildet wird, ist jeweils kleiner als die über den ganzen Perimeter gerechnete Kennzahl. Dies ist aufgrund der Durchmischung mit kleineren und grösseren Parzellen soweit zu erwarten.

In beiden Perimetern zeigt die Definition einer Grünflächenziffer keinen Unterschied im Vergleich zum «Basis» Szenario. Eine Analyse der detaillierten Zahlen hat gezeigt, dass dies darauf zurückzuführen ist, dass die entsprechenden Freiräume bereits durch die Grenzabstände freigehalten werden. Hierfür spricht auch, dass die Überbauungsziffern im «Basis» Szenario unter 0.3 liegen, was umgekehrt bedeutet, dass über 70% der Fläche nicht bebaut sind.

Das Szenario ohne Ausnützung führt in beiden Perimetern zu einer klaren Steigerung der Dichte. Da in diesen Fällen die nicht von den Abstandsbereichen absorbierten Flächen voll ausgenutzt werden, steigt die Überbauungsziffer klar. Dies äussert sich in Langenthal in einer Steigerung der überbauten Fläche von rund 19% gegenüber dem Szenario «Basis», während in Wohlen die Steigerung sogar rund 64% beträgt. Dies zeigt auch, dass in Wohlen mehr grössere Grundstücke mit noch nicht ausgenutzten Reserven vorhanden sind. Die Steigerung der realisierbaren Geschossflächen in Langenthal und Wohlen bewegen sich mit 17% beziehungsweise 70% in ähnlichen Grössenordnungen, während die Veränderung bei den Volumen mit 11% und 45.8% etwas tiefer ausfallen. Diese etwas tieferen Werte bei den Volumen sind unter anderen auf die verschiedene Geschossigkeit in den unterschiedlichen Modellzonen zurückzuführen.

Tabelle 18: Übersicht Kennzahlen der Perimeter Langenthal und Wohlen

Projekt	**Langenthal**					**Uettligen**			
Szenario	Basis	Basis + Grünflächenziffer 0.7	Basis ohne Ausnützungsziffer	W3	W3 + Abstandslinien	Basis	Basis + Grünflächenziffer 0.7	Basis ohne Ausnützungsziffer	W3
Anrechenbare Grundstücksfläche Total [m^2]	135'763	135'763	135'763	135'763	135'763	282'196	282'196	282'196	282'196
Überbaute Fläche Total [m^2]	21'732	21'732	25'760	21'732	22'253	55'780	55'780	91'715	54'439
Veränderung Überbaute Fläche Total zu Szenario «Basis»	-	0.0%	18.5%	0.0%	2.4%	-	0.0%	64.4%	-2.4%
Bauvolumen Total [m^3]	184'694	184'694	204'935	240'477	246'402	453'055	453'055	660'589	574'477
Bauvolumen Fläche Total zu Szenario «Basis»	-	0.0%	11.0%	30.2%	33.4%	-	0.0%	45.8%	26.8%
Geschossfläche Total [m^2]	46'284	46'284	54'341	66'207	67'770	119'333	119'333	202'892	163'317
Geschossfläche Fläche Total zu Szenario «Basis»	-	0.0%	17.4%	43.0%	46.4%	-	0.0%	70.0%	36.9%
Geschossflächenziffer$_g$	0.341	0.341	0.400	0.488	0.499	0.423	0.423	0.719	0.579
Geschossflächenziffer$_{avg}$	0.290	0.290	0.323	0.427	0.439	0.351	0.351	0.512	0.523
Bau-massenziffer$_g$	1.360	1.360	1.510	1.771	1.815	1.605	1.605	2.341	2.036
Baumassenziffer$_{avg}$	1.198	1.198	1.293	1.582	1.629	1.455	1.455	1.881	1.929
Überbauungsziffer$_g$	0.160	0.160	0.190	0.160	0.164	0.198	0.198	0.325	0.193
Überbauungsziffer$_{avg}$	0.142	0.142	0.158	0.142	0.146	0.178	0.178	0.255	0.174

Die Aufzonung auf W3 führt ebenfalls in beiden Perimetern zu einer Steigerung der Dichte, die aber weniger stark ausfällt als beim Verzicht auf eine Begrenzung der Ausnützung. In Wohlen Uettligen sinkt bei diesem Szenario die Überbauungsziffer leicht, was sich ebenfalls in einer Verkleinerung der überbauten Fläche um rund 2% äussert. Dies ist darauf zurückzuführen, dass im parametrischen Modell das Bauen in die Höhe priorisiert wird d.h. eher ein zusätzliches Geschoss realisiert als in die Fläche gebaut wird. Als Konsequenz verändern sich in einigen Fällen auch die Grundrisse der Gebäude. In den Detaildaten ist das Phänomen der Abnahme der Überbauungsziffer bei einigen Parzellen in Langenthal ebenfalls zu sehen. Dass es sich aber allenfalls in den Nachkommastellen der Kennzahlen niederschlägt, ist darauf zurückzuführen, dass in Langenthal tendenziell kleinere Parzellen vorhanden sind, welche nach Abzug der Abstandsbereiche im «Basis» Szenario die erlaubte Ausnutzung infolge der begrenzten Fläche nicht ausreizen können. Entsprechend verändert sich in Langenthal die überbaute Fläche ebenfalls nicht. Durch die Aufzonung kann auf der gleichen Grundfläche durch zusätzliche Geschosse aber mehr Dichte realisiert werden. Dies äussert sich in Steigerungen der Geschossfläche in Langenthal und Wohlen um 43% und 37% sowie in einer Steigerung die Volumina um 30% und 27%.

Die Definition von Abstandslinien im Perimeter Langenthal hat kleine aber messbare Auswirkungen. Da die Abstandslinien innerhalb der Abstandsbereiche des normalen «W3» Szenario liegen, kann auf den betroffenen Parzellen mehr Fläche bebaut werden, was das Modell ausnutzt. In den Übersichtskarten in Abbildung 42 und Abbildung 43 äussert sich das in leicht anderen Grundrissen der Gebäude auf den betroffenen Parzellen. Diese Grundrisse sind leicht näher an der Strasse platziert und haben

eine grössere Grundfläche. Dies schlägt sich in allen Kennzahlen in einer kleinen Erhöhung nieder. Die überbaute Fläche steigt um rund 2% und die Geschossfläche sowie das Volumen jeweils um rund 3%.

Betrachtet man die durchschnittlichen Dichtekennzahlen und die Kennzahlen über den ganzen Perimeter gerechnet, so liegen diese unterhalb der Nutzungsziffern, die von den Parametern der Szenarien festgelegt wurden. Tabelle 19 und Tabelle 20 zeigen detaillierter aufgeschlüsselte Kennzahlen für die Perimeter in Langenthal und Wohlen. Einerseits sind die Dichten pro Modellzone aufgeschlüsselt. Zudem werden neben dem Durchschnitt der Dichtekennzahlen das Minimum und Maximum aufgeführt. Betrachtet man die Minimalwerte, so fallen in beiden Perimetern Nullwerte auf. Dies ist auf einzelne Parzellen zurückzuführen, welche durch die Abstandsbereiche komplett abgedeckt und dadurch nicht mehr bebaut werden können oder aber durch einen Abtausch innerhalb einer Gruppe nicht bebaut wurden. Langenthal ist davon stärker betroffen, da hier mehr Parzellen mit kleiner Grundfläche und/oder komplexer Form vorhanden sind, welche eher durch die Abstandsbereiche komplett absorbiert werden.

Die Maximalwerte sind teilweise klar höher als die Durchschnitte. Vergleicht man hier jeweils den Wert der Kennzahl, welche der Art der Ausnützung der jeweiligen Modellzone entspricht, so liegt deren Wert oft näher an der parametrisierten Nutzungsziffer. In einigen Fällen wird diese sogar überschritten. Dies ist darauf zurückzuführen, dass gruppierte Parzellen wie eine einzige Parzelle behandelt werden, welche als ganzes die Ausnützung einhält. Allerdings kann in einigen Parzellen der Gruppe die Nutzungszahl überschritten und in anderen als Ausgleich unterschritten werden.

Tabelle 19: Kennzahlen Szenarien Langenthal Ringstrasse aufgeschlüsselt nach Modellzonen

Szenario	Zone Code	Anrechenbare Grundstücks-fläche Total [m²]	Überbaute Fläche Total [m²]	Bauvolumen Total [m³]	Geschoss-fläche Total [m²]	Geschossflächen -ziffer$_{avg}$	Geschossflächen -ziffer$_{min}$	Geschossflächen -ziffer$_{g}$	Baumassen -ziffer$_{avg}$	Baumassen -ziffer$_{min}$	Baumassen -ziffer$_{g}$	Baumassen -ziffer$_{max}$	Überbauungs -ziffer$_{g}$	Überbauungs -ziffer$_{min}$	Überbauungs -ziffer$_{avg}$	Überbauungs -ziffer$_{max}$	Art der Ausnnützung	Nutzungsziffer
Basis	Z1	68'816	10'464	85'856	20'927	0.675	0.000	1.248	1.139	0.000	1.248	2.833	0.152	0.000	0.136	0.337	GFZ	0.40
	Z2	52'879	7'778	65'234	15'556	0.957	0.000	1.234	1.193	0.000	1.234	4.019	0.147	0.000	0.142	0.478	GFZ	0.50
	Z3	7'495	1'681	12'197	3'361	0.785	0.000	1.627	1.319	0.000	1.627	3.297	0.224	0.000	0.170	0.393	GFZ	0.60
	Z4	4'716	1'411	17'824	5'642	1.561	0.195	3.780	2.939	0.730	3.780	4.683	0.299	0.049	0.222	0.390	BMZ	6.00
	Z5	1'856	399	3'583	797	0.500	0.012	1.930	1.143	0.035	1.930	2.250	0.215	0.006	0.128	0.250	GFZ	0.50
Basis + Grünflächen-ziffer 0.7	Z1	68'816	10'464	85'856	20'927	0.675	0.000	1.248	1.139	0.000	1.248	2.833	0.152	0.000	0.136	0.337	GFZ	0.40
	Z2	52'879	7'778	65'234	15'556	0.957	0.000	1.234	1.193	0.000	1.234	4.019	0.147	0.000	0.142	0.478	GFZ	0.50
	Z3	7'495	1'681	12'197	3'361	0.785	0.000	1.627	1.319	0.000	1.627	3.297	0.224	0.000	0.170	0.393	GFZ	0.60
	Z4	4'716	1'411	17'824	5'642	1.561	0.195	3.780	2.939	0.730	3.780	4.683	0.299	0.049	0.222	0.390	BMZ	6.00
	Z5	1'856	399	3'583	797	0.500	0.012	1.930	1.143	0.035	1.930	2.250	0.215	0.006	0.128	0.250	GFZ	0.50
Basis ohne Ausnütz-ungsziffer	Z1	68'816	13'593	102'179	27'186	0.982	0.321	1.485	1.287	0.000	1.485	3.157	0.198	0.000	0.160	0.491	Keine	0.00
	Z2	52'879	8'145	67'071	16'289	0.973	0.300	1.268	1.233	0.000	1.268	4.086	0.154	0.000	0.150	0.486	Keine	0.00
	Z3	7'495	2'214	14'278	4'427	0.920	0.413	1.905	1.440	0.000	1.905	3.340	0.295	0.000	0.207	0.460	Keine	0.00
	Z4	4'716	1'411	17'824	5'642	1.561	0.888	3.780	2.939	0.730	3.780	4.683	0.299	0.049	0.222	0.390	BMZ	6.00
	Z5	1'856	399	3'583	797	0.500	0.256	1.930	1.143	0.035	1.930	2.250	0.215	0.006	0.128	0.250	GFZ	0.50
W3	Z1	68'816	10'464	115'154	31'391	1.012	0.409	1.673	1.520	0.000	1.673	3.778	0.152	0.000	0.136	0.337	GFZ	0.60
	Z2	52'879	7'778	87'013	23'334	1.435	0.427	1.646	1.592	0.000	1.646	5.358	0.147	0.000	0.142	0.478	GFZ	0.75
	Z3	7'495	1'681	16'903	5'042	1.178	0.509	2.255	1.794	0.000	2.255	4.396	0.224	0.000	0.170	0.393	GFZ	0.90
	Z4	4'716	1'411	17'824	5'642	1.561	0.888	3.780	2.939	0.730	3.780	4.683	0.299	0.049	0.222	0.390	BMZ	6.00
	Z5	1'856	399	3'583	797	0.500	0.256	1.930	1.143	0.035	1.930	2.250	0.215	0.006	0.128	0.250	GFZ	0.50
W3 + Abstands-linien	Z1	68'816	10'514	115'712	31'541	1.012	0.413	1.681	1.534	0.000	1.681	3.778	0.153	0.000	0.138	0.337	GFZ	0.60
	Z2	52'879	8'089	90'544	24'266	1.435	0.448	1.712	1.672	0.000	1.712	5.358	0.153	0.000	0.149	0.478	GFZ	0.75
	Z3	7'495	1'841	18'738	5'524	1.350	0.533	2.500	1.896	0.000	2.500	5.039	0.246	0.000	0.178	0.450	GFZ	0.90
	Z4	4'716	1'411	17'824	5'642	1.561	0.888	3.780	2.939	0.730	3.780	4.683	0.299	0.049	0.222	0.390	BMZ	6.00
	Z5	1'856	399	3'583	797	0.500	0.256	1.930	1.143	0.035	1.930	2.250	0.215	0.006	0.128	0.250	GFZ	0.50

Tabelle 20: Kennzahlen Szenarien Wohlen Uettligen aufgeschlüsselt nach Modellzonen

Szenario	Zone Code	Anrechenbare Grundstücks-fläche Total [m²]	Überbaute Fläche Total [m²]	Bauvolumen Total [m³]	Geschoss-fläche Total [m²]	Geschossflächen -ziffer$_{g}$	Geschossflächen -ziffer$_{min}$	Geschossflächen -ziffer$_{avg}$	Baumassen -ziffer$_{g}$	Baumassen -ziffer$_{min}$	Baumassen -ziffer$_{avg}$	Baumassen -ziffer$_{max}$	Überbauungs-ziffer$_{g}$	Überbauungs-ziffer$_{min}$	Überbauungs-ziffer$_{avg}$	Überbauungs-ziffer$_{max}$	Art der Ausnnützung	Nutzungsziffer
Basis	Z1	23'378	4'652	24'639	4'652	0.199	0.024	0.187	1.054	0.136	1.019	2.711	0.199	0.024	0.187	0.484	GFZ	0.30
	Z2	201'386	37'501	308'438	75'001	0.372	0.000	0.346	1.532	0.000	1.447	3.760	0.186	0.000	0.173	0.448	GFZ	0.40
	Z3	15'348	3'581	35'327	10'743	0.700	0.700	0.700	2.302	1.960	2.450	2.613	0.233	0.233	0.233	0.233	GFZ	0.70
	Z4	4'179	1'201	7'344	2'402	0.575	0.485	0.542	1.757	1.680	1.858	2.035	0.287	0.242	0.271	0.300	GFZ	0.60
	Z5	37'906	8'845	77'307	26'534	0.700	0.700	0.700	2.039	1.960	2.178	2.613	0.233	0.233	0.233	0.233	GFZ	0.70
Basis + Grünflächen-ziffer 0.7	Z1	23'378	4'652	24'639	4'652	0.199	0.024	0.187	1.054	0.136	1.019	2.711	0.199	0.024	0.187	0.484	GFZ	0.30
	Z2	201'386	37'501	308'438	75'001	0.372	0.000	0.346	1.532	0.000	1.447	3.760	0.186	0.000	0.173	0.448	GFZ	0.40
	Z3	15'348	3'581	35'327	10'743	0.700	0.700	0.700	2.302	1.960	2.450	2.613	0.233	0.233	0.233	0.233	GFZ	0.70
	Z4	4'179	1'201	7'344	2'402	0.575	0.485	0.542	1.757	1.680	1.858	2.035	0.287	0.242	0.271	0.300	GFZ	0.60
	Z5	37'906	8'845	77'307	26'534	0.700	0.700	0.700	2.039	1.960	2.178	2.613	0.233	0.233	0.233	0.233	GFZ	0.70
Basis ohne Ausnütz-ungsziffer	Z1	23'378	6'082	26'163	6'082	0.260	0.024	0.217	1.000	0.136	1.019	2.800	0.260	0.024	0.217	1.000	Keine	0.00
	Z2	201'386	58'427	407'811	116'853	0.580	0.000	0.500	2.000	0.000	1.871	6.301	0.290	0.000	0.250	1.000	Keine	0.00
	Z3	15'348	6'347	54'652	19'042	1.241	0.779	1.147	3.561	2.910	3.415	3.911	0.414	0.260	0.382	0.466	Keine	0.00
	Z4	4'179	1'662	9'925	3'324	0.795	0.485	0.683	2.375	2.035	2.252	2.470	0.398	0.242	0.342	0.441	Keine	0.00
	Z5	37'906	19'197	162'039	57'591	1.519	0.750	1.265	4.275	2.764	3.619	5.622	0.506	0.250	0.422	0.669	Keine	0.00
W3	Z1	23'378	3'537	39'617	10'612	0.454	0.073	0.446	1.695	0.272	1.666	3.539	0.151	0.024	0.149	0.316	GFZ	0.50
	Z2	201'386	37'501	413'440	112'502	0.559	0.000	0.520	2.053	0.000	1.932	5.013	0.186	0.000	0.173	0.448	GFZ	0.60
	Z3	15'348	3'581	35'327	10'743	0.700	0.700	0.700	2.302	1.960	2.450	2.613	0.233	0.233	0.233	0.233	GFZ	0.70
	Z4	4'179	975	8'785	2'925	0.700	0.700	0.700	2.102	1.960	2.287	2.613	0.233	0.233	0.233	0.233	GFZ	0.70
	Z5	37'906	8'845	77'307	26'534	0.700	0.700	0.700	2.039	1.960	2.178	2.613	0.233	0.233	0.233	0.233	GFZ	0.70

5.3.5 Ergebnisse Expertenworkshop

5.3.5.1 Organisation Workshop

Am Workshop nahmen neben dem Autor dieser Arbeit vier Vertreter des Kompetenzbereichs Dencity für Urbane Entwicklung und Mobilität des Departements Architektur, Holz und Bau der Berner Fachhochschule (BFH AHB) teil:

- Christine Seidler, Ing. Raumplanung B.ScHTR FSU / Professorin und Co-Leitung Dencity
- William Fuhrer, Architekt MA ZFH / Professor und Co-Leitung Dencity
- Joachim Huber, Prof. Dr. Ing EMBA HSG / Professor für Architektur
- Simon Gilgen, Dipl. Architekt ETH SIA / Wissenschaftlicher Mitarbeiter

Vom Ablauf her wurden zunächst die Masterarbeit allgemein sowie die Ergebnisse aus den aufbereiteten Szenarien präsentiert. In der begleitenden Diskussion wurden neben der Beurteilung der Resultate aus Sicht von Raumplanern und Architekten nötige Anpassungen und Ergänzungen des Modells besprochen. Teilweise beinhaltete dies eine kurze Diskussion möglicher Lösungsansätze für die Anpassung des Modells. Die aufbereiteten Ergebnisse und Schlussfolgerungen aus dem Workshop sind in den nachfolgenden Unterabschnitten zusammengefasst.

5.3.5.2 Vorgehensmodell allgemein

Aussagen über die Anwendbarkeit des Vorgehensmodells in Praxisprojekten können ohne vorherige Erprobung nicht pauschal gemacht werden. Angemerkt wurde, dass die Schritte des Vorgehensmodells den Schritten nahekommen, welche für die Erstellung der Fallstudie Langenthal durchlaufen wurden. Tests des Vorgehensmodells in Projekten wären aber notwendig, um konkrete Aussagen für die Anwendbarkeit machen und den nötigen Anpassungsbedarf im Detail definieren zu können. Es wurde aber bereits die Feststellung gemacht, dass das Modell für die Anwendung an bestehende Prozesse beziehungsweise Verfahren angedockt und auf diese abgestimmt werden müsste. Als möglicher Ansatzpunkt dafür wurde das Aufzeigen von Gemeinsamkeiten des Vorgehensmodells beziehungsweise des zugrundeliegenden Geodesign Frameworks mit etablierten Prozessen genannt. So wurde auf Parallelen der im Geodesign Framework vorgesehenen Schritte und Iterationen zu den ersten Phasen mit strategischen Planungen, Vorstudien und Planungen in der Norm SIA 112 hingewiesen, welche als Modell zur Bauplanung in der Schweizer Baubranche eine wichtige Rolle bei Planungen und Ausschreibungen einnimmt (SIA 2014). Das Aufzeigen solcher Anknüpfungspunkte würde potentiell die gemeinsame Umsetzung eines individuellen Geodesign Prozesses mit Auftraggebern erleichtern. Generell sei es wichtig, die Konzepte des Geodesigns sowie des Vorgehensmodells in einer zielgruppengerechten Sprache aufzubereiten und zu präsentieren. Dies kann je nach Fragestellung, Auftraggeber und beteiligten Anspruchsgruppen individuelle Anpassungen für jedes Projekt bedingen.

Aus Sicht der Raumplanung wurde der partizipative Aspekt des Geodesign Ansatzes positiv und wichtig bewertet. Insbesondere die Auswertungen zu den Dichtekennzahlen in Kombination mit Kartogrammen, welche über die bauliche Dichte hinausgehende Aspekte aufzeigen, wurden sehr gut aufgenommen. Diese wurden als mögliche Entscheidungsgrundlage für die Diskussion mit Anspruchsgruppen und Auftraggebern genannt, mittels derer mögliche Handlungsoptionen gemeinsam entwickelt und anhand von verschiedenen Zusammenhängen und Faktoren begründet werden können. Abhängig von der Grösse des Projektgebiets wären allerdings feinere Auflösungen als ein Hektarraster wünschenswert. Zudem wären in Partizipationsprozessen zusätzliche, qualitative Auswertungen von Nutzen, wofür das von Angélil *et al.* (2016) beschriebene Vorgehen geeignete Ansätze liefern könnte. Aus Sicht der Architektur wurde bei den Rückmeldungen eher das parametrische Modell und dessen Fähigkeit der Generierung baulicher Varianten fokussiert.

5.3.5.3 Parametrisches Modell

Zum parametrischen Modell wurde allgemein angemerkt, dass dieses in mehreren Punkten weitergeht, als das bisherige Modell des Dencity, welches z.B. bei der Fallstudie Langenthal eingesetzt wurde. Wichtiger Unterschied ist die stärkere Betonung der Automatisierung und der weitgehende Verzicht auf manuelle Korrekturen. Der Ansatz zur Ableitung der Grundrisse wurde sehr positiv beurteilt. In der Diskussion hat sich aber auch gezeigt, dass bei der Zusammenstellung der Grundrisse aus den Kandidatenflächen (Quads) noch weitere Strategien erforscht werden solletn, wobei sich diese je nach

Zielsetzung unterscheiden können (z.B. Generierung kompakter Gebäude ohne «Fortsätze», Beschränkung auf nur ein zusammenhängendes Gebäude pro Parzelle, gezielte Generierung von mehreren, ähnlich grossen Gebäuden).

Die Generierung eines angenäherten Steildaches wurde ebenfalls als Fortschritt angesehen, welcher im Vergleich zu den angenäherten Dachgeschossen, wie sie im Modell für die Fallstudie Langenthal genutzt wurden, zu einer realistischeren Darstellung beiträgt. Insbesondere entfällt durch die Steildächer die Verwechslungsgefahr mit Attikageschossen. Zu den Dächern wurde ergänzend die Frage gestellt, ob aus deren Geometrie die Fläche mit mindestens 1.5m Höhe abgeleitet werden kann, welche als Geschossfläche im Dachgeschoss anzurechnen wäre. Genau solche Dachgeschosse werden im parametrischen Modell der vorliegenden Arbeit im Moment explizit nicht betrachtet, wurden aber bei der Fallstudie Langenthal durch angenäherte Dachgeschosse berücksichtigt. Dies ist höchstwahrscheinlich einer der Faktoren, welcher in den aufbereiteten Szenarien im Vergleich zur Fallstudie zu tieferen Geschossflächen führt.

Die Orientierung der Parameter an der Interkantonalen Vereinbarung zur Harmonisierung der Baubegriffe wurde als sinnvoll eingestuft. Nach Ablauf der Übergangsfristen, die kantonal unterschiedlich geregelt sind, seien alle Gemeinden dazu verpflichtet, ihre Baugesetzgebung an die IVHB beziehungsweise deren kantonale Umsetzung anzupassen. Dadurch ergibt sich aufgrund der breiten Übertragbarkeit der Kernkonzepte aus der IVHB ein grosses Potential in Bezug auf die Anwendung des Modells in künftigen Projekten.

In Bezug auf die Anwendung der Grenzabstände zeigte sich in der Diskussion, dass die Positionierung des grossen Grenzabstandes verfeinert werden sollte. Da dieser auf dem ganzen längsten zusammenhängenden Grenzabschnitt mit Ausrichtung nach Osten bis Südwesten angewendet wird und nicht nur auf den Abschnitt mit der Hauptwohnseite des Gebäudes, geht im Modell potentiell bebaubare Fläche verloren. Infolge fallen im Modell die potentiellen Bauflächen kleiner aus und die realisierbare Ausnutzung bleibt unter den Erwartungen. Zudem dürften die zu grosszügig angewendeten Grenzabstände mit dafür verantwortlich sein, dass die Grünflächenziffer kaum einen Einfluss auf die Kennzahlen hat. In der Diskussion wurde darauf hingewiesen, dass beim Ableiten der Gebäudegrundrisse für die Ausrichtung der minimalen, orientierten Bounding Box quasi die Hauptwohnseite bestimmt wird, welche nicht den ganzen Grenzabschnitt abdeckt. Der grosse Grenzabstand könnte analog nur auf den längsten Einzelabschnitt angewendet werden.

In direktem Zusammenhang mit den Grenzabständen steht der Hinweis auf eine Reihe langgezogene, sehr schmale Gebäude in den Verdichtungsvarianten, die so in der Realität nicht gebaut werden könnten. Diese entstehen in den meisten Fällen durch Anwendung des grossen Grenzabstandes an der Längsseite schmaler Parzellen. Um die Outputs des Modells glaubhaft zu machen, müssten diese Gebäude eliminiert werden. Gebäude dieser Art seien beim Modell des Dencity für die Fallstudie Langenthal ebenfalls aufgetreten und wurden dort nachträglich manuell korrigiert. Vom Vorgehen her wurde dafür der grosse Grenzabstand auf der schmalen Seite des Grundstücks angewendet, was auch in der Realität bei solchen Parzellen oft gemacht werde. Dadurch entstehen Baubereiche mit günstigerer Form, die Gebäude mit «normalen» Abmessungen zulassen. Die Eliminierung solch langgezogener Gebäude würde im parametrischen Modell beträchtliche Anpassungen bedingen, welche vermutlich eng mit den Anpassungen für die verfeinerte Handhabung des grossen Grenzabstandes kombiniert werden müssten. Dabei wäre ein Schwellwert für die Elongation der zurückgesetzten Parzellenfläche aus dem ersten Durchlauf zu definieren, bei dessen Überschreiten der grosse Grenzabstand in einem zweiten Schritt auf die schmale Seite des Grundstücks angewendet wird.

Als wichtiger Parameter, der im Modell noch fehlt, wurde der Strassenabstand genannt. Abhängig von der Art der Strasse müssen Gebäude einen Mindestabstand einhalten (im Kanton Bern z.B. 5m bei Kantonsstrassen und 3.6m bei Gemeinde- und Quartierstrassen). Durch Integration der Geometrien der Strassen sowie eines Abstandswerts liessen sich Strassenabstände als zusätzlicher Abstandsbereich in das Modell integrieren. Als weiterer Punkt in Bezug auf Abstände wurden einzuhaltende Baulinien genannt, welche Abstandslinien mit gestalterischer Wirkung entsprechen würden. Deren Integration wird als machbar aber aufwändig eingestuft.

Eine weitere Ergänzungsmöglichkeit im Modell wäre die Festlegung von maximalen Breiten und Längen für Gebäude in einer Zone, welche mancherorts zur Grössenbegrenzung genutzt werden. Bei der Ableitung der Gebäudegrundrisse aus dem Quadtree basierten Raster für die Annäherung der Baufläche liessen sich diese Grössen bei der Auswahl der Quads voraussichtlich gut als Nebenbedingung integrieren.

Kurz angeschnitten wurde die Unterteilung grösserer Parzellen z.B. bei einer Einzonung. Während verschiedene Ansätze zur prozeduralen Unterteilung z.B. anhand von Zerschneidung auf Basis einer ausgerichteten Bounding Box (Koenig *et al.* 2017) oder der Medial Axis der Polygone existieren (Vanegas *et al.* 2012), wurde in der Diskussion eine manuelle Unterteilung im Rahmen der Datenaufbereitung für ein Projektgebiet favorisiert.

Als nicht zulässig wurde beurteilt, dass in den aufbereiteten Varianten für Wohlen in einigen Parzellen Gebäude ausserhalb der eigentlichen Bauzone zu liegen kommen. Dies verletze Grundsätze der Schweizer Raum- und Nutzungsplanung und ist als Hindernis für die Anwendbarkeit im Rahmen konkreter Projekte anzusehen. Die reine Begrenzung über die anrechenbare Grundstücksfläche, wie es bei den aufbereiteten Szenarien der Fall war, ist also als unzureichend anzusehen. In der Diskussion wurde die Integration echter Bauzonen in das parametrische Modell vom Autor als aufwändig eingeschätzt. Es wurde als geeignete Alternative angesehen, wenn im Rahmen der Übernahme der Parzellen bei der Datenaufbereitung des Projektperimeters ein Verschnitt der Parzellen mit den Bauzonen vorgenommen und nur die Parzellenteile innerhalb der Bauzone übernommen würden. Dieses Vorgehen würde bei Parzellen, die in mehreren Bauzonen liegen, eine Aufteilung zur Folge haben, welche sogar näher an der Realität liegen würde, als das bisherige Vorgehen bei der Aufbereitung der Szenarien, wo solche Parzellen als Ganzes behandelt und pauschal nur einer Modellzone zugeordnet wurden. Durch die Aufteilung könnten den einzelnen Teilen unterschiedliche Modellzonen zugeordnet werden. Eine weitere Möglichkeit zur Umgehung des Problems wäre die Abtrennung von Flächen ausserhalb der Bauzone mittels einer Abstandslinie, was aber die Anwendung regulärer Grenzabstände an der entstehenden Grenze ausschliessen würde.

Weitere Anmerkungen betrafen die Wortwahl und Kommunikation im Zusammenhang mit dem Modell. Für den Parameter Grünflächenziffer sollte künftig besser der Begriff Freiflächenziffer verwendet werden. Von der Natur her sind sich die beiden Ziffern zwar ähnlich, die Flächen der Grünflächenziffer müssen aber zwingend humusiert sein. Abhängig vom Gesprächspartner sei dies ein wichtiger Unterschied. In Bezug auf das parametrische Modell hat der jetzige Parameter Grünflächenziffer eher den Charakter Freiflächenziffer. Eine Anpassung wäre zudem konsistenter mit der Definition der Auswertungen zum Freiflächenanteil. Während die Anpassung vor allem eine Umbenennung erfordert und somit einfach umzusetzen ist, wurde in Bezug auf den vorliegenden Text und die erarbeiteten Ergebnisse aus Gründen der Konsistenz darauf verzichtet.

Des Weiteren wurde angeregt in Bezug auf die Outputs des Modells von Varianten statt von Entwürfen zu sprechen, da Letztere schon als wesentlich konkreter anzusehen wären. Diese Anpassung wurde im vorliegenden Text bereits berücksichtigt. Aufgrund der Verwendung des Begriffes Landscape beziehungsweise Landschaft im Rahmen des Geodesign Prozesses solle zudem in der Kommunikation klar hervorgehoben werden, dass das Modell auf Raumplanung und nicht auf Landschaftsplanung abzielt, um Missverständnisse bei Laien zu vermeiden. Bei der Aufbereitung und Präsentation von mehreren Varianten wurde zudem angeregt, nicht nur mit absoluten Zahlen zu arbeiten, sondern auch mit Prozentzahlen. Dabei ist eine Variante als Basis zu definieren, wie dies bei den hier aufbereiteten Szenarien bereits der Fall ist. Die Erfahrungen aus der Fallstudie Langenthal hätten gezeigt, dass dabei die Unterschiede bei den realisierbaren Geschossflächen und Volumen im Zentrum stünden, während die Anzahl Einwohner und Beschäftigte kaum eine Rolle spielten. Entsprechend wurden in der Dokumentation der Szenarien in den vorangegangenen Abschnitten bereits Prozentangaben ergänzt.

5.3.5.4 Schlussfolgerungen

Die Rückmeldungen zum Vorgehensmodell und dem parametrischen Modell weisen darauf hin, dass dieses eine gute erste Grundlage darstellt und zumindest in Teilen Potential für den Einsatz in konkreten Planungsprojekten hat. Die aufgezeigten Schwachpunkte zeigen aber auch, dass zuvor noch Verbesserungen nötig sind. Beim Vorgehensmodell als Ganzes ist davon auszugehen, dass dieses für

jedes Projekt leicht angepasst und auf die jeweilige Fragestellung sowie die beteiligten Anspruchsgruppen abgestimmt werden muss. Aus dieser Perspektive macht eine zu starke Formalisierung des Vorgehensmodells keinen Sinn. Vielmehr sollte das Modell als Framework und die damit zusammenhängenden Systemteile als Werkzeugkasten angesehen werden, die als Grundlage für die Ausgestaltung eines konkreten Planungsprozesses dienen. Bei der Weiterentwicklung wäre entsprechend eine Flexibilisierung der einzelnen Komponenten und deren Kombination anzustreben.

Die vorgeschlagenen Korrekturen und Ergänzungen am parametrischen Modell werden als sinnvoll und teilweise zwingend angesehen. Wie im obenstehenden Text zu den Rückmeldungen angedeutet, wurden im Rahmen des Workshops zudem schon einige Lösungsvarianten andiskutiert. Auf dieser Basis sowie aufgrund einiger zusätzlicher Abklärungen wird eine entsprechende Anpassung des Modells als machbar angesehen. Einige der angemerkten Punkte wie das Ausscheiden der Flächen ausserhalb der Bauzonen im Rahmen der Datenaufbereitung, die Verwendung der Freiflächenziffer sowie die Bestimmung der Geschossfläche innerhalb der generierten Dächer sind relativ einfach umsetzbar. Infolge der Abhängigkeiten mit anderen Teilen des Vorgehensmodells wie dem Datenmodell und den Auswertungen fällt aber der insgesamt nötige Aufwand grösser aus. Andere Anpassungen wie die Verfeinerung der Platzierung des grossen Grenzabstandes, die Eliminierung langgezogener Gebäude und die Integration von Strassenabständen sind mit mehr Aufwand verbunden. Bei diesen Anpassungen muss ausserdem einschränkend gesagt werden, dass vermutlich nicht für alle Fälle eine komplett automatisierte Lösung machbar ist und das parametrische Modell um Möglichkeiten für gezielte manuelle Korrekturen durch Übersteuern von Parametern erweitert werden sollte. Auf Integration der in der Expertenrunde vorgeschlagenen Ergänzungen am parametrischen Modell wurde im Rahmen der Masterarbeit verzichtet, da der Aufwand für deren Integration die Grenzen der Arbeit sprengen würde.

5.4 Sensitivitätsanalyse

5.4.1 Vorgehen

Während die Beurteilung aus Expertensicht auf Basis einer überschaubaren Zahl an Szenarien umgesetzt werden konnte, ist die Auswertung einer grösseren Menge an Varianten nötig, um die Sensitivität des parametrischen Modells auf bestimmte Parameter verlässlich beurteilen zu können. Um eine bessere Vergleichbarkeit der Resultate zwischen den beiden Projektperimetern zu gewährleisten und eine allfällige Sensitivität in die Situation im Projektgebiet zu erhalten, wird keine Rücksicht auf lokale Gegebenheiten genommen. Deshalb werden auf alle Parzellen die gleichen Parameter angewendet (d.h. es ist effektiv nur eine Modellzone vorhanden). Zudem werden die Parametrisierungen in Langenthal und Wohlen Uettligen identisch angewendet. Ausnahme ist die Gruppierung von Parzellen, welche für den Perimeter spezifisch ist. In der Hälfte der Fälle werden diejenigen Parzellen in Gruppen des Typs «Reihe» zusammengefasst, die auch in der Realität zusammenhängende Gebäude aufweisen, welche über Parzellengrenzen hinausgehen. Dies insbesondere, um nicht eine grössere Zahl schmaler Parzellen komplett auszuschliessen und «unbebaut» zu lassen.

Für die Analyse werden die Verteilungen baulicher Dichtekennzahlen in den generierten Verdichtungsvarianten ausgewertet, um den Einfluss einzelner Parameter zu untersuchen. Die Outputs zu Anzahl Einwohner und Anzahl Beschäftigte werden nicht ausgewertet, da diese über die Geschossfläche in direkter Funktion von den generierten Gebäuden abhängen, weshalb keine nennenswerte Zusatzinformation aus der Analyse dieser Grössen zu erwarten ist.

5.4.2 Definition Parameterraum

Die Erzeugung der Verdichtungsvarianten mittels komplett randomisierten Parameterwerten wurde als sehr aufwändig und wenig zielführend eingestuft. Stattdessen wurden in einem gezielteren Ansatz die zu untersuchenden Parameter ausgewählt und jeweils ein Wertebereich festgelegt. Diese in Tabelle 21 ersichtlichen Wertebereiche spannen den Parameterraum für die Sensitivitätsanalyse auf. Die Parameterwerte und deren Kombination müssen für diesen Teil der Verifikation nicht zwingend realistisch sein. Vielmehr können explizit Kombinationen von extremen Parameterwerten auftreten, welche in der Realität nur bedingt Sinn ergeben würden, um das Verhalten des Modells in diesen Situationen mit zu berücksichtigen.

Tabelle 21: Parameterraum für Sensitivitätsanalyse

Parameter	Wertebereich
Art der Ausnutzung	[GFZ, BMZ, ÜZ, Keine]
Nutzungsziffer	[0.5,0.75,1.0,1.25,1.5]
Kleiner Grenzabstand	[2,4,6]
Grosser Grenzabstand	[6,8,10]
Grünflächenziffer	[0,0.5]
Geschosszahl	[2,3,4]
Gruppierung	[ohne Gruppierungen, Gruppierung «Reihe» wie in Realität]

Die nicht explizit untersuchten Parameter des Modells werden auf sinnvolle Standardwerte gesetzt, so dass diese bei der Generierung der Varianten nicht einschränkend wirken:

- Die Werte der Parameter Anteil Arbeitsnutzung, Platzbedarf Arbeitsnutzung und Platzbedarf Wohnnutzung werden konstant gehalten, da die Anzahl Einwohner und Anzahl Beschäftigte nicht ausgewertet werden.
- Die maximale Gesamthöhe wird so gesetzt, dass die volle Geschosszahl ausgereizt und ein Steildach generiert werden kann. Es wird davon ausgegangen, dass in der Realität ebenfalls die Maximalhöhe so definiert wird, dass die maximale Geschosszahl realisiert werden kann.
- Die Geschosshöhe wird konstant auf 2.8m gehalten. Bei der Baumasse hat dieser Parameter die Funktion eines linearen Faktors ($\text{Volumen} = \text{Gebäude Grundfläche} * (\text{Anzahl Geschosse} * \text{Geschosshöhe} + \text{Dach Höhe})$). Im Zusammenspiel mit der maximalen Gesamthöhe könnte dieser Faktor aber trotzdem limitierend wirken und zur Einschränkung der Anzahl Geschosse oder zur Generierung eines Flachdaches führen, da das Bauen in die Höhe priorisiert wird.
- Dachtypen werden nicht explizit parametrisiert. Eine arbiträre Mischung ergibt keinen Sinn. Wenn nur Flachdächer generiert werden, würde das Bauvolumen leicht sinken. Dies wäre allenfalls sinnvoll, wenn Limitierungen bei der Gesamthöhe unterschritten werden sollen. Wenn explizit nur Steildächer generiert werden, würde sich das Bauvolumen entsprechend erhöhen. Da insbesondere grosse Gebäude betroffen wären, welche vom Modell mit Flachdach generiert werden, würde dies ins Gewicht fallen.
- Abstandslinien werden nicht eingesetzt. Deren Effekt ist sehr lokalisiert und spezifisch. Zudem wäre deren Festlegung teilweise arbiträr und deren Erfassung vergleichsweise aufwändig.

5.4.3 Grundlagen der Auswertung

Zur Berechnung der notwendigen Grundlagedaten für die Analyse wurden für jede Permutation der Werte im Parameterraum eine Variante berechnet und deren Resultate in einer Datenbank gespeichert. Für die Umsetzung wurden zunächst mittels SQL Script die Wertekombinationen des Parameterraums erzeugt und in der zentralen Datenbank gespeichert. Dabei wurde jede Permutation als separates Szenario abgebildet. Für die eigentliche Generierung wurde eine modifizierte Version des parametrischen Modells erstellt, das mit Hilfe des Hoopsnake Add-ons (Chatzikonstantinou 2014) so erweitert wurde, dass dieses über die zuvor erzeugten Parametersätze in der Datenbank iteriert. Dabei werden die Parameterwerte ausgelesen, auf das Modell angewendet und die Ergebnisse der Berechnung in der Datenbank gespeichert, bevor die Iteration mit dem nächsten Parametersatz weiterfährt. Auf diese Art wurden für beide Projektperimeter alle Permutationen des Parameterraums durchgerechnet.

Die resultierenden Verdichtungsvarianten bilden die Grundlage für die eigentliche Sensitivitätsanalyse. Aus der Permutation der definierten Parameterwerte ergeben sich 2160 unterschiedliche Kombinationen. Durch die Berechnung für die beiden Projektgebiete ergeben sich 4320 Varianten als Grundlage. Einschränkend muss gesagt werden, dass aus den 2160 Kombinationen effektiv nur 1512 distinkte Varianten resultieren, da bei der Ausnützungsart *ÜZ* Nutzungsziffern mit einem Wert von über 1 als 1 behandelt werden und da bei der Ausnützungsart *Keine* die Nutzungsziffer ignoriert wird. Dieser Umstand spiegelt sich auch in den Resultaten wider.

Eine erste Auswertung mit globalen Dichtekennziffern über das ganze Projektgebiet hat gezeigt, dass diese Durchschnittswerte zwar generell die Tendenzen des Modells aufzeigen, Extremwerte aber ausklammern. Da diese aber für die genaue Einschätzung einiger Eigenheiten des Modells wichtig sind, wurden pro Variante und Parzelle die individuellen Dichtekennziffern berechnet. Wohlen Uettligen weist

im Projektperimeter 327 und Langenthal 199 Parzellen auf. Mit den 2160 Szenarien pro Perimeter ergeben sich 1’136’160 Datenpunkte, welche als Grundlage für die Auswertungen in diesem Kapitel dienen.

Tabelle 22 fasst die Zusammenhänge zwischen den untersuchten Parametern und den betrachteten baulichen Dichtekennziffern zusammen. Dabei ist ersichtlich, dass einige Parameter einen gleichläufigen (d.h. bei steigendem Wert steigt die Dichte) andere hingegen einen gegenläufigen Einfluss (d.h. bei steigendem Wert sinkt die die Dichte) haben. Dabei verhalten sich Geschossflächenziffer und Baumassenziffer ähnlich, während es bei der Überbauungsziffer kleine Unterschiede in Bezug auf die Geschosszahl gibt. Die gezeigten Zusammenhänge geben die generellen Tendenzen wieder. Wie die Visualisierungen zu den einzelnen Parametern in den nachfolgenden Unterabschnitten aber aufzeigen, hängt die Stärke des Effektes sowohl vom jeweiligen Projektgebiet als auch von der Ausnützungsart ab. Zusammengefasst kann aber gesagt werden, dass die Zusammenhänge zwischen Parametern und der Dichte den Erwartungen entspricht und sich diese mit den Zusammenhängen in der Realität decken.

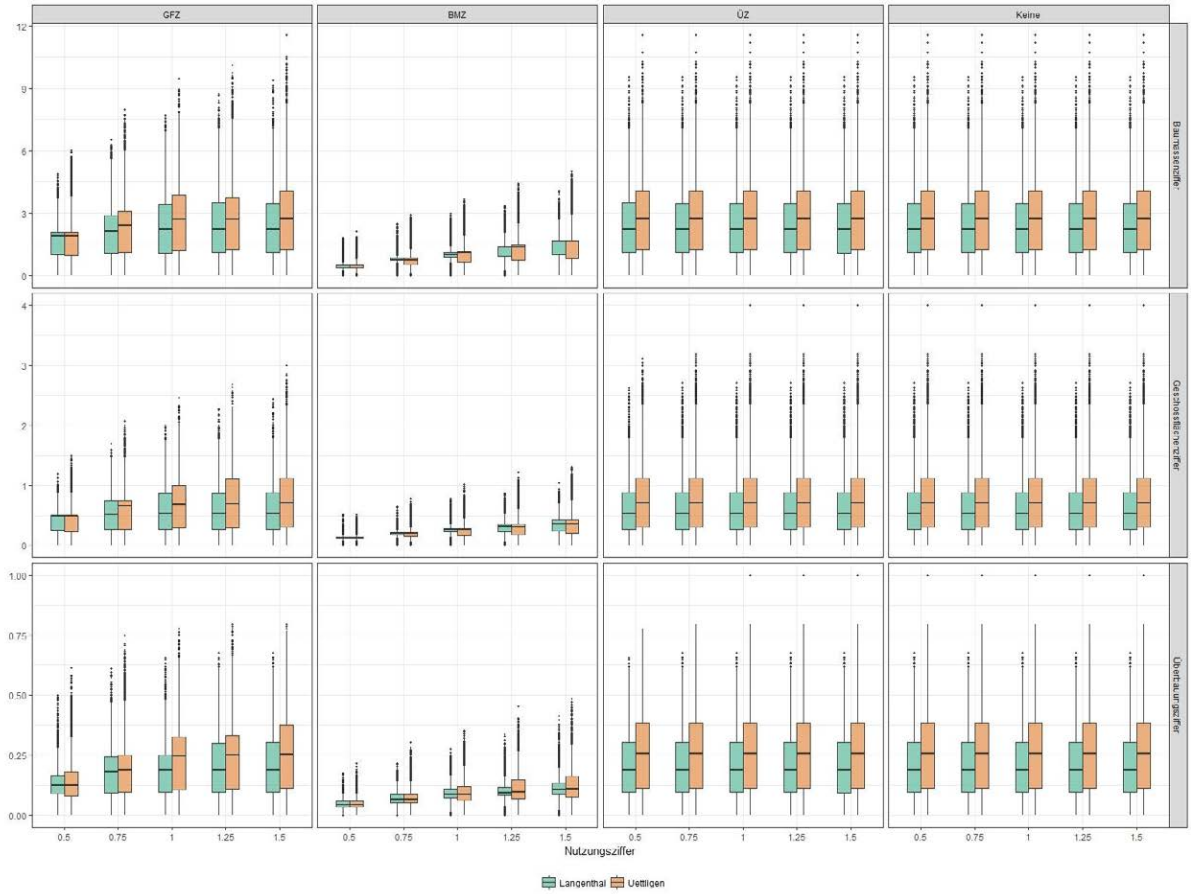
Tabelle 22: Einfluss Parameter auf bauliche Dichtekennziffern

Parameter	Geschossflächenziffer	Baumassenziffer	Überbauungsziffer
Nutzungsziffer	++	++	++
Kleiner Grenzabstand	--	--	--
Grosser Grenzabstand	--	--	--
Grünflächenziffer	-	-	-
Geschosszahl	++	++	-
Gruppierung	+	+	+

Einfluss	gleichläufig	gegenläufig
stark	++	--
schwach	+	-

5.4.4 Einfluss der Art der Ausnützung

Abbildung 48 zeigt die Verteilungen der Dichtekennzahlen pro Parzelle für das Modell als Box-Plots aufgeschlüsselt nach Art der Ausnützung und Projekt. Die Ausnützungsart ist in den Spalten der Matrix aufgeschlüsselt und die zugehörigen Nutzungsziffern sind auf der X-Achse aufgetragen. Die drei Kennzahlen Geschossflächenziffer, Baumassenziffer und Überbauungsziffer sind in den Zeilen aufgeschlüsselt und der zugehörige Wert wird jeweils auf der Y-Achse dargestellt. Die Unterscheidung nach Projekt ist schliesslich durch die Gruppierung der Box-Plots wiedergegeben. Mit dieser Aufschlüsselung zeigt die Abbildung nicht nur den Einfluss der Art der Ausnützung, sondern auch eine Übersicht des Modells im Allgemeinen. Man sieht in den Plots, dass generell mit zunehmender Nutzungsziffer die Dichte zunimmt, was den Erwartungen entspricht. Ausnahmen zeigen sich bei der Ausnützungsart *Keine* und *ÜZ*. Für *ÜZ* führen Nutzungsziffern über 1 zum gleichen Ergebnis. Da ein Grundstück maximal zu 100% ausgenutzt werden kann, behandelt das parametrische Modell höhere Nutzungsziffern gleich wie den Wert 1. Die Ausnützungsart *Keine* ignoriert hingegen die Nutzungsziffer komplett und verhält sich im Prinzip immer wie eine Bemessung der Ausnützung nach *ÜZ* mit einer Nutzungsziffer von 1 (d.h. die Fläche des Grundstücks wird nach Anwendung der Abstandsbereiche immer maximal ausgenutzt). Infolge ist das Ergebnis für alle Nutzungsziffern identisch.



Quelle: eigene Darstellung

Abbildung 48: Verteilung Dichtekennzahlen pro Parzelle nach Ausnützungsart und Projekt

Für die Fälle, wo die Art der Ausnützung und die Dichtekennziffer übereinstimmen, ist ersichtlich, dass der Wert der Dichtekennzahl im Schnitt kleiner ist, als der Wert der Nutzungsziffer. Neben der Situation auf den einzelnen Grundstücken ist dies auf den einschränkenden Einfluss der anderen Parameter und die Eigenheiten des Modells (insbesondere Platzierung des grossen Grenzabstandes und Ignorierung von Dachgeschossen) zurückzuführen. Zudem überschreitet die Kennzahl in der Regel die Nutzungsziffer nicht, was grundsätzlich zu erwarten wäre. Dass es trotzdem Ausreisser nach oben gibt, ist auf die Gruppierung von Parzellen zurückzuführen, was in Abschnitt 5.4.8 im Detail ausgeführt wird.

Es ist ebenfalls ersichtlich, dass sich die Baumassenziffer weitgehend ähnlich verhält wie die Geschossflächenziffer, aber um einen Faktor kleiner. Dieser Faktor ergibt sich aus der ähnlichen Definition der beiden Kennzahlen. Im Zähler stehen mit der Geschossfläche und dem Bauvolumen zwei Grössen, die von der Grundfläche abgeleitet sind. Die Grösse des Faktors zwischen den beiden Grössen ist etwas grösser als die Geschosshöhe, welche beim Bauvolumen als zusätzlicher Faktor zur Geschosszahl hinzukommt (plus der Dach Höhe als Summand). Für die Analysen der weiteren Parameter werden deshalb der Übersichtlichkeit halber nur die Geschossflächenziffer und Überbauungsziffer betrachtet.

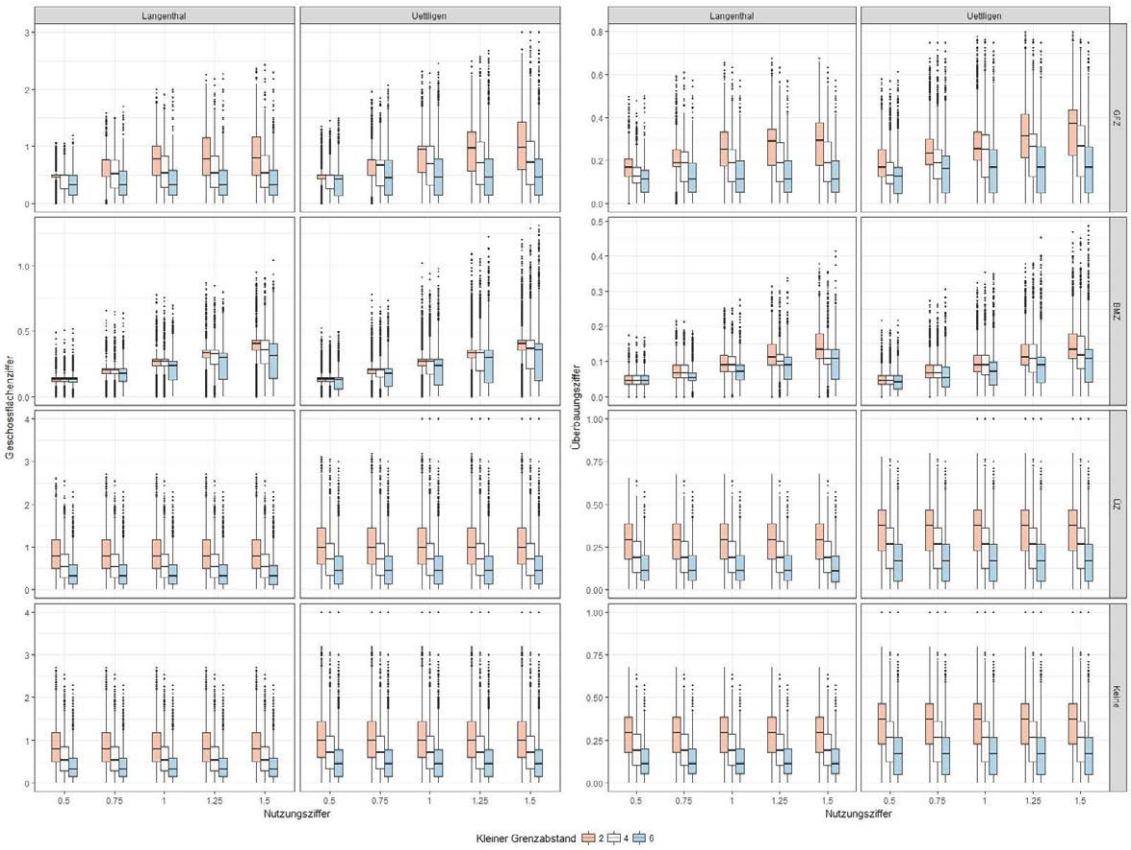
Durch die Gruppierung nach Projekten wird ersichtlich, dass in Wohlen Uettligen die Dichte im Mittel leicht höher liegt und der Wertebereich (insbesondere nach oben) etwas grösser ist als in Langenthal. Dies zeigt, dass das Projektgebiet und dessen Parzellen einen Einfluss auf die mögliche Dichte haben. Langenthal hat weniger Grundstücke, wovon viele eine kleinere bis mittlere Grösse aufweisen. Zudem befinden sich der Grossteil der Parzellenfläche in der Bauzone. In Uettligen hat es hingegen mehr grössere Parzellen. Gerade einige der grossen Parzellen liegen nur teilweise in einer Bauzone und weisen deshalb eine anrechenbare Grundstücksfläche kleiner als die Grundstücksfläche auf. Ein weiterer Unterschied ist, dass es in Uettligen mehr kleine, schmale Parzellen gibt, die zu Gruppen zusammengefasst werden.

5.4.5 Einfluss der Grenzabstände

Abbildung 49 und Abbildung 50 zeigen die Verteilung der Geschossflächenziffer und Überbauungsziffer pro Parzelle für die kleinen sowie die grossen Grenzabstände als Box-Plots. Ein Vergleich der Abbildungen zeigt, dass die Auswirkungen sich sehr ähneln. Mit der Erhöhung der Abstände geht eine Verringerung der Dichte bei allen Ausnützungsarten und bei beiden Kennzahlen einher, wobei beim grossen Grenzabstand die Verteilungen etwas stärker nach unten verschoben sind. Dies wird auf die stärker einschränkende Wirkung der grossen Grenzabstände zurückgeführt. Die Unterscheide zwischen den beiden Projektgebiete sind relativ klein, Wohlen Uettligen zeigt aber tendenziell einen leicht höheren Median und einen etwas grösseren Interquartilsabstand.

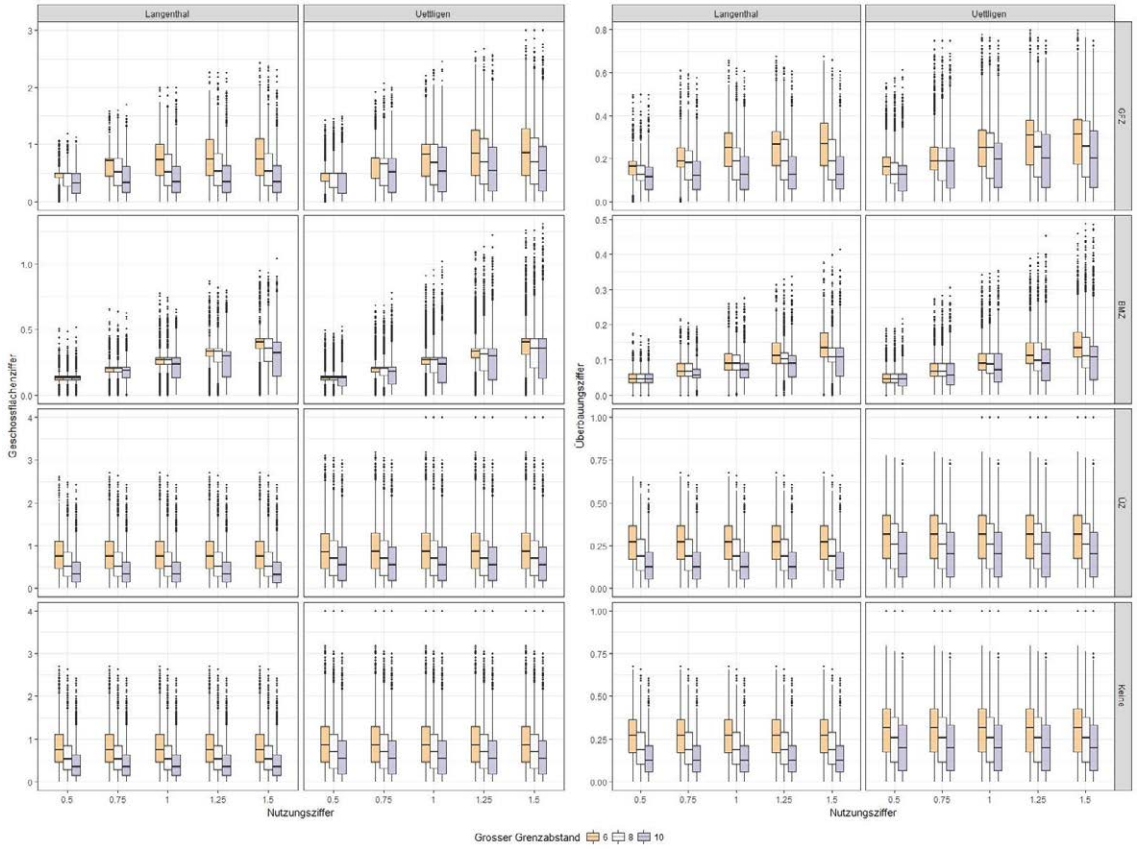
Bei den Ausnützungsarten *ÜZ* und *Keine* ist der Effekt der Abnahme in der Dichte mit zunehmendem Grenzabstand sehr deutlich, da sich der Schnitt mit jedem Erhöhungsschritt klar nach unten verschiebt. Dies deshalb, weil bei beiden Ausnützungsarten die Grundfläche für die Bemessung relevant ist und die potentiell bebaubare Fläche durch die Erhöhung der Grenzabstände direkt reduziert wird. Dass bei Uettligen trotzdem klare Ausreisser nach oben vorhanden sind, hängt damit zusammen, dass es hier einige grössere Parzellen gibt, die nur zum Teil in einer Bauzone liegen. Das parametrische Modell in der verwendeten Form kennt aber keine Beschränkung des Baufeldes aufgrund einer Zonengrenze, sondern nur eine Beschränkung anhand der anrechenbaren Grundstücksfläche. Infolge bleibt bei besagten Parzellen wegen ihrer Grösse trotz maximalen Grenzabständen und anderen Einschränkungen immer noch genug Fläche, damit das Modell die Ausnützung immer ausreizen kann.

Dass bei den Ausnützungsarten *GFZ* und *BMZ* die Effekte der Verschiebung der Verteilung der Dichte weniger ausgeprägt ausfällt und die Medianwerte teilweise kaum verschieben, liegt daran, dass hier bei der Bemessung auch die dritte Dimension eine Rolle spielt. Das Modell priorisiert das Bauen in die Höhe, wodurch teilweise die Grundfläche zugunsten der Höhe nicht vollständig ausgenutzt wird. Dadurch ist der Impact der Erhöhung der Grenzabstände nicht in allen Fällen gleich hoch und es kommt zu einer grösseren Überlappung der Verteilungen.



Quelle: eigene Darstellung

Abbildung 49: Verteilung von Geschossflächenziffer und Überbauungsziffer pro Parzelle nach Projekt, Ausnützungsart und kleinem Grenzabstand

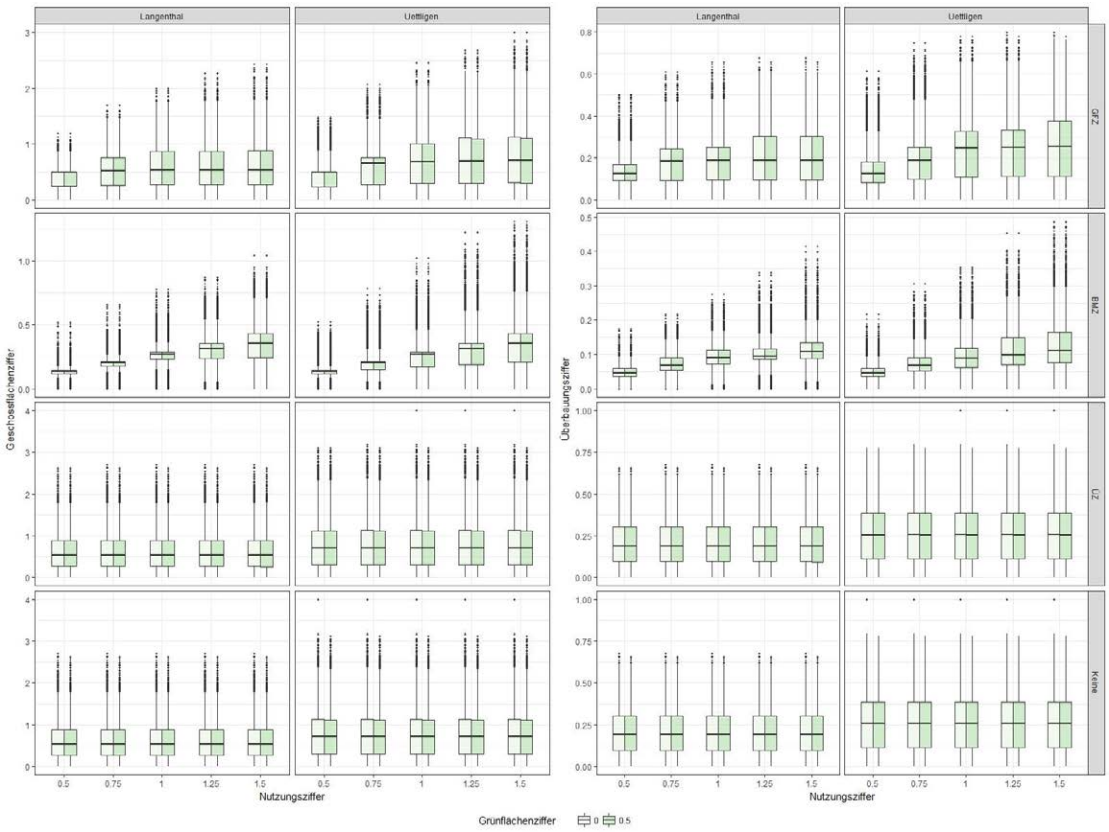


Quelle: eigene Darstellung

Abbildung 50: Verteilung von Geschossflächenziffer und Überbauungsziffer pro Parzelle nach Projekt, Ausnützungsart und grossem Grenzabstand

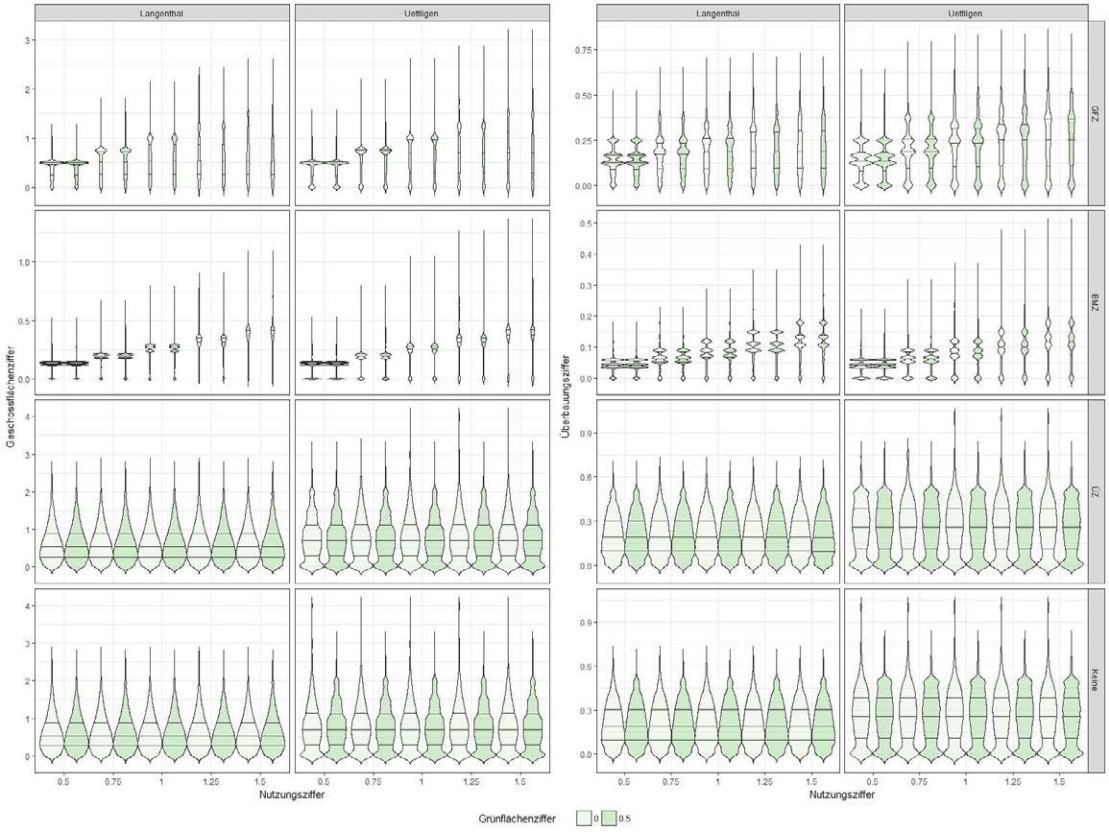
5.4.6 Einfluss der Grünflächenziffer

Abbildung 52 zeigt die Verteilung der Geschossflächenziffer und Überbauungsziffer pro Parzelle für die Situation mit und ohne Grünflächenziffer als Box-Plots. Es ist ersichtlich, dass die Definition einer Grünflächenziffer tendenziell eine Reduktion der Dichte zur Folge hat, da ein Anteil der anrechenbaren Grundstücksfläche nicht überbaut werden darf. In den generierten Szenarien ist in der Hälfte der Fälle eine Grünflächenziffer von 0.5 definiert, weshalb 50% der Fläche frei bleiben müssen. Wie sich schon im Rahmen der Szenarien für die Expertenbeurteilung angedeutet hatte, fällt die Auswirkung auf die Verteilung im Rahmen der Sensitivitätsanalyse allerdings geringer aus als ursprünglich erwartet. Es ist jeweils ein kleiner Unterschied am oberen Ende der Verteilungen zu sehen, wo sich die Ausreisser sowie das 3. Quartil leicht nach unten verschieben. Betrachtet man die Dichtefunktionen in den Violin-Plots in Abbildung 53, so sind hier gleichfalls leichte Verschiebungen im oberen Wertebereich der Verteilung ersichtlich. Als Hauptgrund für diese Resultate wurden bei Betrachtung der detaillierten Daten die Abstandsbereiche ausgemacht, welche in den meisten Fällen bereits die nötigen Freiflächen von 50% abdecken. Die Verschiebung im oberen Teil ist auf (tendenziell grössere) Parzellen zurückzuführen, bei denen durch die Grünflächenziffer tatsächlich eine Einschränkung der bebaubaren Fläche eintritt. Es ist zu erwarten, dass der Einfluss der Grünflächenziffer bei höheren Werten oder aber bei kleineren Grenzabständen stärker ausfallen würde. Ein stärkerer Einfluss ist ebenfalls zu erwarten, wenn im Modell die Platzierung des grossen Grenzabstandes gezielter erfolgt, wie sie im Rahmen der Resultate des Expertenworkshops skizziert wurde. Durch die Anpassung würde weniger Baufläche durch Abstandbereiche absorbiert, was wiederum den Effekt der Grünflächenziffer verstärken würde.



Quelle: eigene Darstellung

Abbildung 51: Verteilung von Geschossflächenziffer und Überbauungsziffer pro Parzelle nach Projekt, Ausnützungsart und Grünflächenziffer



Quelle: eigene Darstellung

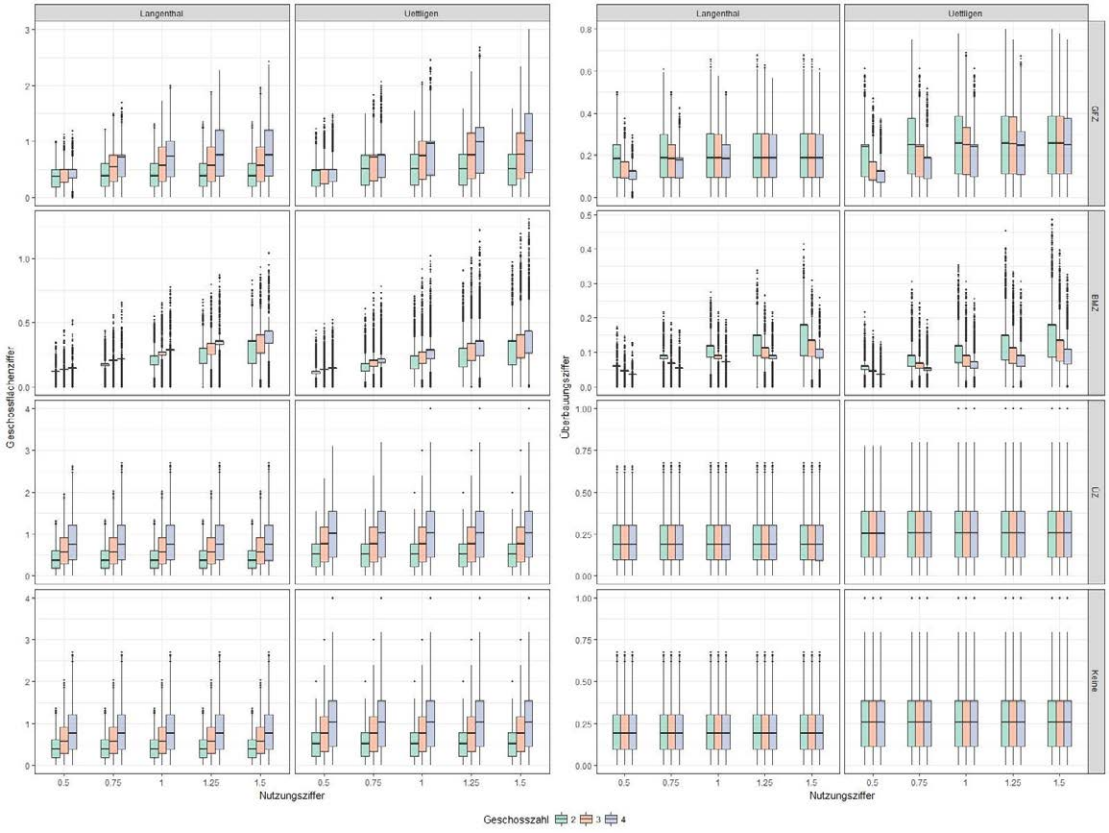
Abbildung 52: Dichtefunktionen von Geschossflächenziffer und Überbauungsziffer pro Parzelle nach Projekt, Ausnützungsart und Grünflächenziffer

5.4.7 Einfluss der Geschosszahl

Abbildung 54 zeigt die Verteilung der Geschossflächenziffer und Überbauungsziffer pro Parzelle für die verschiedenen Geschosszahlen als Box-Plots. Bei der Geschossflächenziffer führt die Erhöhung der Geschosszahl durchwegs zu einer Verbreiterung der Spannweite der Dichte nach oben. Diese Verbreiterung hängt damit zusammen, dass in Fällen mit bereits ausgereizter Grundfläche aber einer Ausnutzung unterhalb des eigentlich erlaubten Wertes durch zusätzliche Geschosse tatsächlich mehr Geschossfläche oder Bauvolumen realisiert werden kann, um die Nutzungsziffer auszureizen.

Bei Betrachtung der Überbauungsziffer für die Ausnützungsarten *GFZ* und *BMZ* fällt auf, dass diese tendenziell abnimmt. Dies ist auf die Priorisierung des Bauens in die Höhe im Modell zurückzuführen. Wird bei ausgereizter Ausnützung die Geschosszahl erhöht, die restlichen Parameter aber beibehalten, so reduziert das Modell die Grundfläche des Gebäudes, um ein zusätzliches Stockwerk zu generieren. Die gleiche Geschossfläche wird also auf mehr Stockwerke verteilt. Das Volumen bleibt ebenfalls ähnlich. Die Reduktion der Grundfläche führt dann aber zu einer niedrigeren Überbauungsziffer. Es ist bei den höheren Nutzungsziffern aber ersichtlich, dass der Effekt der Abnahme bei der Überbauungsziffer abnimmt. Dies wird darauf zurückgeführt, dass hier bei der überwiegenden Zahl der Parzellen infolge der höheren Nutzungsziffer bereits die zur Verfügung stehende Grundfläche, nicht aber die erlaubte Ausnutzung ausgereizt ist. Während sich durch die Aufstockung zusätzliche Geschossfläche und Volumen realisieren lassen, ändert sich in diesen Fällen nichts an der Grundfläche.

Betrachtet man die Geschossflächenziffer für die Ausnützungsarten *ÜZ* und *Keine*, so sieht man, dass die Erhöhung der Geschosszahl einen Sprung sowohl bei der Spannbreite als auch dem Median der Dichte ergibt. Dies liegt daran, dass bei diesen Ausnützungsarten nur die Grundfläche für die Bemessung betrachtet wird und ein zusätzliches Geschoss tatsächlich zu einer Aufstockung bei gleichbleibender Grundfläche führt. Entsprechend steigen Geschossfläche und Bauvolumen. Gleichzeitig bleibt die Überbauungsziffer bei gleicher Nutzungsziffer aber konstant, wie in der Grafik gut sichtbar ist.

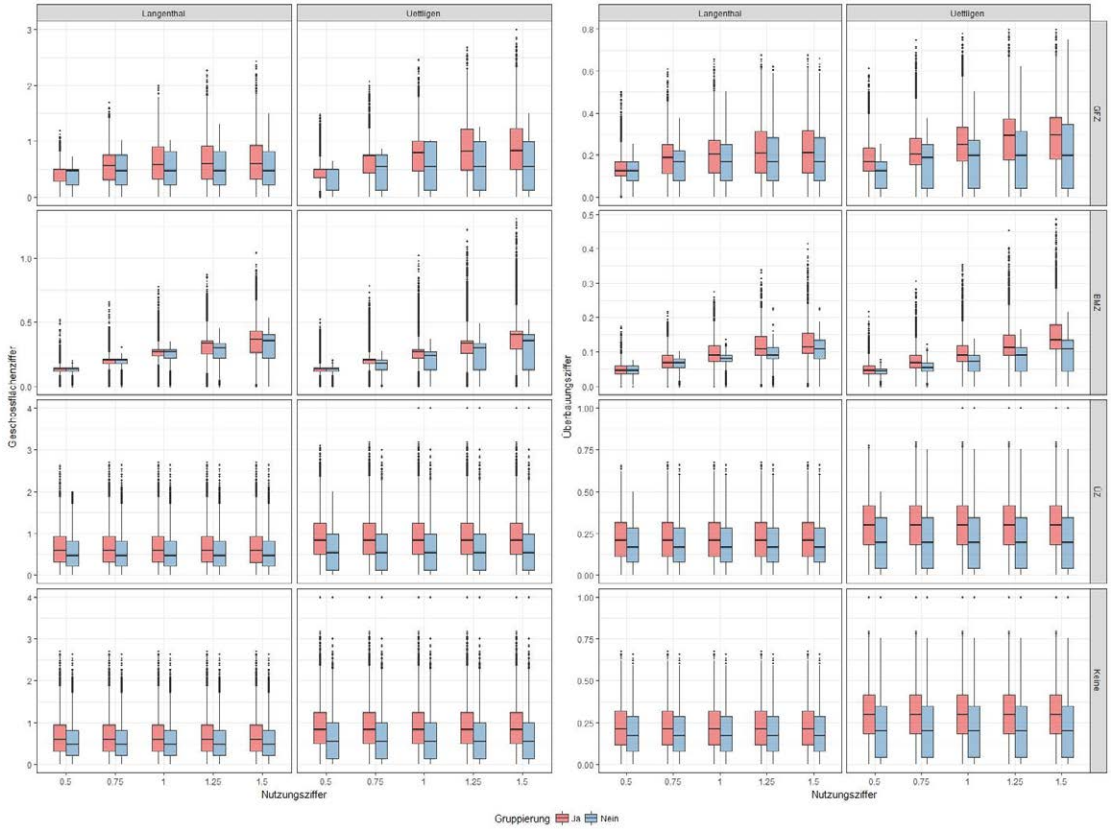


Quelle: eigene Darstellung

Abbildung 53: Verteilung von Geschossflächenziffer und Überbauungsziffer pro Parzelle nach Projekt, Ausnützungsart und Geschosszahl

5.4.8 Einfluss der Gruppierung

Abbildung 55 zeigt die Verteilung der Geschossflächenziffer und Überbauungsziffer pro Parzelle für die Situation mit und ohne Gruppierung von Parzellen als Box-Plots. Es ist ein klarer Unterschied zwischen Szenarien mit und ohne Gruppierung ersichtlich. Bei beiden Kennzahlen und allen Ausnützungsarten ist der Wertebereich grösser, wenn gruppiert wird. Ohne Gruppierung verschiebt sich die Bandbreite der Verteilung nach unten hin. Dieser Unterschied ist darin begründet, dass ohne Gruppierung die kleinen und oft schmalen Parzellen, die in der Realität normalerweise für Reihenhäuser genutzt werden, nicht oder nur wesentlich unterhalb der Nutzungsziffer ausgenutzt werden können. Dies insbesondere, weil die Abstandsbereiche die Bebauung grösserer Teile der Parzellen verhindern. Durch die Gruppierung werden diese Parzellen hingegen zusammengefasst und wie eine einzelne Parzelle behandelt, welche im Schnitt besser ausgenutzt werden kann. Durch die Gruppierung kann es zudem dazu kommen, dass einige Parzellen bei *GFZ* und *BMZ* als Ausnützungsart stärker ausgenutzt werden, als die Nutzungsziffer es eigentlich zulassen würde. Dies ist in Form der Ausreisser nach oben zu sehen. Im Austausch werden andere Parzellen der Gruppe weniger oder gar nicht ausgenutzt. Aus dem gleichen Grund ist für die beiden Ausnützungsarten ohne Gruppierung kein Wert oberhalb der Nutzungsziffer zu finden. Die etwas grössere Bandbreite der Werte bei Uettligen und die höhere Zahl an Ausreissern ist auf die grössere Zahl an Gruppen in diesem Perimeter zurückzuführen.



Quelle: eigene Darstellung

Abbildung 54: Verteilung von Geschossflächenziffer und Überbauungsziffer pro Parzelle nach Projekt, Ausnützungsart und Gruppierung

6 Schlussfolgerungen und Ausblick

Ziel dieser Master Thesis war die Entwicklung eines Vorgehensmodells zur Unterstützung von Testplanungen, welches auf dem Geodesign Framework nach Steinitz (2012) basiert und systematisch GIS mit Parametrischem Design integriert. Die zentrale Herausforderung für dieses Modell ist, eine räumliche Datenbasis sowie parametrische Modelle so zu verknüpfen, dass anhand der Outputs die durch Anpassung abstrakter Bauvorschriften verursachten Auswirkungen auf die innere Verdichtung quantifiziert und vermittelt werden können.

Das Thema innere Verdichtung hat in der Schweizer Raumplanung an Bedeutung gewonnen, seit das revidierte Raumplanungsgesetz eine qualitätsvolle und nachhaltige Innenverdichtung als Ziel definiert hat (RPG 2016). Dies bedingt in Planungsprozessen mehr Aufmerksamkeit von Planern und Architekten (Angélil et al., 2016). Durch den inhärenten Raumbezug spielt GIS bei deren Unterstützung und Bewertung eine zentrale Rolle (Goodchild, 2010, Wissen Hayek et al., 2016). Zudem bietet sich das Konzept des Geodesigns an, um die gestalterischen Entscheide auf Basis wissenschaftlich fundierter Methoden und Informationen zu treffen (Goodchild, 2010). Des Weiteren ist die in Geodesign sowie in Planungsprozessen vorgesehene Involvierung von vor Ort betroffenen Anspruchsgruppen integral, wenn es um die Schaffung eines gemeinsamen Verständnisses und breit abgestützter Lösungen geht (Steinitz 2012; Grams 2015; Moura 2015; Wissen Hayek *et al.* 2016). Im Rahmen von Planungsprozessen unterstützen Visualisierungen und Modelle die zielgruppengerechte Vermittlung der ansonsten komplexen und abstrakten Zusammenhänge sowie Fragestellungen in Bezug auf innere Verdichtung, wobei insbesondere der interaktiven Nutzung parametrischer Modelle im Rahmen von Workshops grosses Potential zuzuschreiben ist (Walz *et al.* 2008; Van Wezemael *et al.* 2014; Moura 2015; Wissen Hayek *et al.* 2016).

In der Arbeit wurde der von Steinitz beschriebene Ablauf zur Ausarbeitung eines Geodesign Prozesses durchlaufen, um das Vorgehensmodell zu entwickeln. Dadurch wurden in der Arbeit zwar weite Teile eines Geodesign Prozesses umgesetzt, aufgrund des Fehlens eines konkreten Planungsprojektes sowie der Partizipation betroffener Gruppen kann aber nicht von einem kompletten Geodesign Prozess gesprochen werden. Vielmehr wurde im Rahmen des Entwurfsprozesses eine auf die Fragestellung der inneren Verdichtung zugeschnittene Version des Geodesign Frameworks erarbeitet. Das resultierende Vorgehensmodell ist ein integriertes Gesamtmodell für Testplanungen zur inneren Verdichtung mit aufeinander abgestimmten Prozessschritten, Datenmodell und technischer Unterstützung. Das Vorgehen ist in Anlehnung an das Steinitzsche Framework in zwei Phasen zum Verständnis der Ist-Situation und zur Definition sowie Analyse von Verdichtungsszenarien mit je drei Schritten gegliedert, welche jeweils spezifische Teilaspekte und Zusammenhänge des Gesamtmodells beleuchten.

Die erste Phase des Vorgehens zielt darauf ab, die Ist-Situation im Projektgebiet zu verstehen und zu bewerten. In einem ersten Schritt wird im Rahmen des Representation Models das Projektgebiet abgebildet. Hierzu wurde eine räumliche Datenbasis aufgebaut, welche unter anderem Daten aus der Amtlichen Vermessung (u.a. Parzellen und Bodenbedeckung), harmonisierte Daten zu Bauzonen des Bundesamtes für Raumplanung, GEOSTAT Daten des Bundesamtes für Statistik, swissBUILDINGS3D 2.0 Daten mit 3D Modellen der Ist-Bebauung sowie Daten aus weiteren Quellen in eine PostGIS Datenbank integriert. Durch Verwendung von Daten, die schweizweit einheitlich oder dank standardisierten Basisdatenmodellen zumindest in ähnlicher Form verfügbar sind, wird die Übertragbarkeit des Modells sichergestellt. Die so aufgebaute Datenbasis dient als Input für die nachfolgenden Analysen und die Anwendung des parametrischen Modells in den weiteren Schritten des Vorgehensmodells.

Im zweiten Schritt wird mittels des Process Models eine Analyse der Ist-Situation vorgenommen, die ein Verständnis für die Dichte und (räumlichen) Zusammenhänge im Projektgebiet schaffen sollen. Die hierfür zusammengestellte Auswahl von Kennzahlen ist auf eine umfassende Betrachtung der Dichte aus verschiedenen Perspektiven ausgerichtet. Eine Gruppe von Kennzahlen zielt auf das Verständnis der Art und Struktur des aktuellen Gebäudebestandes anhand von Typologie, Gebäudehöhe, Geschosszahl und Geschossfläche ab. Diese Faktoren beeinflussen nicht nur die Wahrnehmung des Raumes und dessen Qualität in architektonischer und gestalterischer Hinsicht, sondern haben indirekt ebenso Einfluss auf die realisierbare Dichte (Kunze *et al.* 2012; Grams 2015; Angélil *et al.* 2016). Eine zweite Gruppe baulichen Dichtekennzahlen umfasst unter anderem die Dichte ausgedrückt über Geschossfläche, Bauvolumen, überbaute Fläche oder Freiraumanteil für verschiedene Bezugsflächen.

Der Freiraumanteil trägt zudem zur Qualität der Dichte bei, wobei Aneignung und Nutzung von Freiräumen Teil der (urbanen) Qualität ausmachen und durch vermehrte Interaktionen zur sozialen Dichte beitragen (Spiegel 2000; Angélil *et al.* 2016). Aspekte dieser sozialen Dichte werden von einer dritten Gruppe von Kennzahlen direkt beleuchtet, welche unter anderem Grössen wie Raumnutzerdichte, Beschäftigtendichte, Einwohnerdichte, Wohnungs- und Haushaltsdichte sowie durchschnittliche Wohnungsfläche pro Einwohner umfasst. Ergänzt werden diese Kennzahlen durch zusätzliche Kontextinformationen wie Bauzonen und der Erschliessungsgüte des öffentlichen Verkehrs.

Die Resultate der Analyse dienen im dritten Schritt im Rahmen des Evaluation Models als Grundlage für die Bewertung der Ist-Situation durch die Entscheidungsträger. Im Rahmen der Arbeit wurde der Schritt des Geodesign Prozesses weitgehend offengelassen, da dieser auf den individuellen Prozess abzustimmen ist und die Partizipation der betroffenen Entscheidungsträger bedingt. Zur Unterstützung dieses Prozesses wurden aber Auswertungen und Visualisierungen mittels R und QGIS entwickelt, welche die Analyseergebnisse und Kennzahlen in Form von interaktiv nutzbaren Projekten sowie Berichten zur Entscheidungsunterstützung aufbereiten.

Ziel des zweiten Teils des Vorgehensmodells ist die Auslotung und Vermittlung der Auswirkung der inneren Verdichtung im Projektgebiet. Zentrale Rolle spielt dabei das in der Arbeit entwickelte parametrische Modell zur Generierung von Verdichtungsvarianten, welches im vierten Schritt des Vorgehensmodells das Change Model umsetzt. Dieses ist von ähnlichen, vom Kompetenzbereich Dencity der Berner Fachhochschule eingesetzten Modellen inspiriert, wurde aber von Grund auf neu entwickelt. Es verfolgt in einigen Punkten wie der Integration von GIS, der Ableitung von Gebäudegrundrissen mittels Quadtree Zerlegung sowie der Generierung von Steildächern mittels angenähertem Medial Axis Transform andere Ansätze. Durch Definition von Szenarien mit unterschiedlichen Parameterwerten können mit Hilfe des parametrischen Modells mit wenig Aufwand verschiedene Verdichtungsvarianten erzeugt werden. Die Inputparameter sind an die Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe (IVHB 2005a) angelehnt, was die Übertragbarkeit des Modells auf unterschiedliche Teile der Schweiz erleichtert. Die Auswahl der Parameter umfasst u.a. die anrechenbare Grundstücksfläche, den kleinen und grossen Grenzabstand, die maximale Anzahl Geschosse, die Geschosshöhe, die maximale Gesamthöhe, die Art der Ausnützung sowie die zugehörige Nutzungsziffer. Das in Rhino Grasshopper umgesetzte parametrische Modell nimmt neben diesen Parametern räumliche Daten zum Projektgebiet in Form von Parzellenflächen und Abstandslinien als Input entgegen, um die Varianten zu generieren und zu visualisieren. Weitere Geodaten in Form eines Höhenmodells 3D Daten der Ist-Bebauung und Strassenflächen dienen der Ergänzung der Visualisierung. All diese räumlichen Daten werden aus der zentralen PostGIS Datenbank gelesen und umgekehrt werden die Resultate des Modells sowie Szenarien wiederum dort gespeichert. Dabei kommen GeoJSON als Schnittstellenformat und Python zur Umsetzung von Schnittstellen zum Einsatz. Mittels Python können aus dem parametrischen Modell heraus auch andere Systemteile wie der Export von R-Auswertungen als Grafik oder die Generierung von QGIS Projekten zur Berichtsgenerierung angesprochen werden. Die Kombination aus der Datenbank als zentralem Datenspeicher für alle Systemteile und Python für die Schnittstellenprogrammierung ermöglicht eine effiziente Integration zwischen GIS, parametrischen Designwerkzeugen und weiteren Systemkomponenten.

Im fünften Schritt des Vorgehensmodells werden die erzeugten Verdichtungsvarianten mittels des Impact Models auf deren Auswirkungen auf das Projektgebiet hin analysiert. Die dafür verwendeten Kennzahlen sind eine Untermenge der Kennzahlen des Process Models, was einen Vergleich der Varianten mit der Ist-Situation erleichtert. Das Schwergewicht liegt dabei auf baulichen Dichtekennzahlen, ergänzt durch Kennzahlen zu Raumnutzer-, Beschäftigten- und Einwohnerdichte.

Der sechste und letzte Schritt zielt im Rahmen des Decision Models auf die Bewertung und die Auswahl von Varianten, die als Grundlage für weitere Arbeiten dienen sollen. Analog zum Evaluation Model in Schritt drei wurde dieser Schritt im Rahmen der Arbeit offengelassen, da eine Abstimmung auf den konkreten Planungsprozess nötig ist. Wiederum wurden aber die Resultate aus dem Impact Model so aufbereitet, um Entscheidungsunterlagen zur Unterstützung des Prozessschrittes zur Verfügung zu stellen. Diese Unterlagen sind in ihrer Struktur an die Berichte und Visualisierungen zur Bewertung der Ist-Situation angelehnt, um einen direkten Vergleich zu ermöglichen.

Das Vorgehensmodell wurde anhand eines Fallbeispiels in der Gemeinde Langenthal entwickelt. Neben der technischen Implementierung diente der Perimeter in Langenthal zudem als erster Test für die Anwendung des Modells und für die Erzeugung zugehörigen Outputs. Durch Anwendung auf ein zweites Fallbeispiel in der Gemeinde Wohlen bei Bern konnte die grundsätzliche Übertragbarkeit des Modells gezeigt werden. Die beiden Fallbeispiele dienten anschliessend als Grundlage für weitere Verifizierungsschritte. Als Basis für eine Beurteilung aus Expertensicht wurden für die beiden Perimeter ausgewählte Szenarien definiert, die zugehörigen Varianten generiert, ausgewertet und als Input für einen Workshop mit Architekten und Raumplanern des Dencity aufbereitet. Zusätzlich wurde als Grundlage für eine Sensitivitätsanalyse des parametrischen Modells eine grössere Anzahl von Varianten für die beiden Projektgebiete generiert, indem die Permutationen innerhalb eines zuvor festgelegten Parameterraums mit ausgewählten Parametern und Wertebereichen systematisch durchgerechnet wurden.

Die Resultate der Sensitivitätsanalyse zeigen, dass sich das Verhalten des parametrischen Modells weitgehend mit den Erwartungen auf Basis der Modellannahmen deckt und die reale Zusammenhänge gut annähert. In den Ergebnissen sind allerdings Einschränkungen aufgrund von Eigenheiten und Schwächen des Modells ersichtlich, welche in den generierten Verdichtungsvarianten teilweise unrealistische Gebäudeformen sowie Dichten zur Folge haben, welche leicht zu niedrig sind. Diese Ergebnisse decken sich mit den Rückmeldungen aus dem Workshop mit Experten des Dencity. Deren überwiegend positive Bewertung des Vorgehensmodells im Allgemeinen und insbesondere des parametrischen Modells lassen den Schluss zu, dass diese eine geeignete Grundlage für weitere Arbeiten darstellen. Gleichzeitig wurde im Expertenfeedback auf einige Schwächen hingewiesen, die vor einem praktischen Einsatz adressiert werden müssen. Diese betreffen vorwiegend das zentrale parametrische Modell zur Umsetzung des Change Models im Geodesign Framework, aber auch das Vorgehensmodell generell:

1. **Integration mit bestehenden Prozessen und Verfahren**
   Das Vorgehensmodell und der unterliegende Geodesign Ansatz werden aufgrund ihrer ganzheitlichen Sicht und der partizipativen Aspekte zwar insgesamt positiv beurteilt, eine Aussage zur Eignung für konkrete Planungen kann aber ohne mehrere Tests in Praxisprojekten nicht gemacht werden. Für den Praxiseinsatz wird eine Anknüpfung und Anpassung des Modells an bestehende Verfahren und Prozesse als zwingend erachtet. Anknüpfungspunkte hierfür ergeben sich aus Parallelen zwischen bestehenden Planungsprozessen und der Strukturierung des Steinitzschen Geodesign Frameworks sowie aus dessen partizipativen Aspekten.
2. **Problematische Outputs des parametrischen Modells**
   Da das Modell keine Beschränkung durch Bauzonen kennt, können Gebäude ausserhalb dieser Zonen zu liegen kommen, was nicht mit Grundsätzen der Schweizer Raumplanung vereinbar ist. Die bebaubaren Flächen von Parzellen müssen entsprechend eingeschränkt werden, um dies zu vermeiden. Während der Ansatz zur Ableitung der Gebäudegrundrisse im Allgemeinen zu guten Resultaten führt, hat dieser auch Schwächen. Die Anwendung des grossen Grenzabstandes auf den gesamten, längsten Grenzabschnitt mit Ausrichtung nach Osten bis Südwesten hat eine zu starke Reduktion der potentiell bebaubaren Fläche zur Folge und trägt zu den zu tiefen Ausnützungen im Modelloutput bei. Bei schmalen Grundstücken resultieren aus dieser Art der Anwendung ausserdem schmale, langgezogene Gebäude, die in der Realität so nicht gebaut werden können. Für glaubhafte Resultate sind solche Gebäude zu vermeiden. Während die angenäherten Steildächer zu einem grösseren Realismus der Visualisierungen beitragen, berücksichtigt das Modell die potentiellen Geschossflächen im Dachgeschoss nicht. Dies ist ein weiterer Faktor für zu tiefe Ausnützungen im Modelloutput, welcher durch Berücksichtigung der Dachgeschosse korrigiert werden sollte.
3. **Fehlende Parameter**
   Das parametrische Modell berücksichtigt im Moment keine Strassenabstände. Die Einhaltung der gesetzlichen Mindestabstände ist für einen Einsatz des Modells in Planungen aber als notwendig anzusehen. Darüber hinaus wäre die Ergänzung weiterer Parameter aus der IVHB und der Planungspraxis wünschenswert. Hierzu gehört insbesondere die Festlegung von maximal Längen und Breiten zur Grössenbegrenzung von Gebäuden und Abstandslinien mit gestalterischer Wirkung (d.h. Gebäude müssen an die Linie heran gebaut werden und diese in ihre Fassadenflucht integrieren).

Zur Behebung dieser Schwachpunkte werden folgende Verbesserungen und Massnahmen im Rahmen von künftigen Arbeiten vorgeschlagen:

- **Verfeinerung des Vorgehensmodells durch Anwendung in Praxisprojekten:**
  In Kooperation mit dem Dencity sollte das Vorgehensmodell in Praxisprojekten angewendet und getestet werden. Die dabei vorgenommenen Anpassungen zur Anknüpfung an bestehende Prozesse und die gewonnenen Erfahrungen sollen als Grundlage für die Verfeinerung des Modells und dessen Schritte dienen. Als Stossrichtung wird eine Modularisierung und Flexibilisierung des Vorgehensmodells und seiner Teile empfohlen, damit dieses im Sinne einer Blaupause oder eines Werkzeugkastens einfach in verschiedenen Planungsverfahren als Grundlage genutzt werden kann.
- **Anpassung an der Datenaufbereitung für das parametrische Modell:**
  Die Nutzung des Modells ist dahingehend anzupassen, dass bei der Datenaufbereitung eine Intersection der Parzellenflächen mit Bauzonen vorgenommen wird. Die daraus resultierenden Teilflächen, deren Flächeninhalt der anrechenbaren Grundstücksfläche entspricht, werden anstelle der ursprünglichen Parzellen als geometrischer Input des parametrischen Modells verwendet. Infolge der Änderungen ist ausserdem die Zuordnung der Modellzonen so anzupassen, dass die Zugehörigkeit von Teilflächen einer Parzelle zu verschiedenen Bauzonen mitberücksichtigt wird. Zu prüfen ist zudem die Definition von Schritten zur manuellen Aufteilung grösserer Parzellen oder neu eingezonter Flächen bei der Datenaufbereitung, um entsprechende Situationen mit dem Modell unterstützen zu können.
- **Berücksichtigung von Geschossflächen in Dachgeschossen:**
  Um zu tiefe Ausnützungen zu vermeiden, ist das Modell so anzupassen, dass Geschossflächen im Dachgeschoss bei Gebäuden mit Steildach mitberücksichtigt werden. Hierzu sind die nutzbaren Flächen mit mindestens 1.5m Höhe im Dachgeschoss zu ermitteln. Diese sind mit in den Output zu integrieren. Bei der Ausnützungsart GFZ sind diese Flächen zudem bei der Dimensionierung zu berücksichtigen. Die zusätzliche Integration und Berücksichtigung von Attikageschossen ist eine weitere Option, die in diesem Zusammenhang zu prüfen ist.
- **Anpassung der Anwendung von Grenzabständen:**
  Die Anwendung der Grenzabstände ist anzupassen, um die in diesem Bereich identifizierten Schwachpunkte zu beseitigen. Einerseits ist die Anwendung des grossen Grenzabstandes so zu verfeinern, dass diese nur auf der Hauptwohnseite erfolgt. Damit wird der zu starken Einschränkung der potentiell bebaubaren Parzellenfläche entgegengewirkt. Bei Parzellen, bei denen zu schmale Baufelder und damit langgezogene Gebäude generiert oder gar die ganze Parzelle durch die Grenzabstände absorbiert werden, ist der grosse Grenzabstand statt auf der gesamten längsten Seite mit Ausrichtung nach Osten bis Südwesten nur auf der schmaleren Seite anzuwenden. Dies begünstigt die Generierung eines vorteilhafter geformten Baufeldes.
- **Integration von Strassenabständen:**
  Damit die Mindestabstände von Gebäuden zu Strassen eingehalten werden, ist das parametrische Modell um entsprechende Inputs zu erweitern. Strassenachsen oder -flächen sind als zusätzlicher geometrischer Input in das Datenmodell zu integrieren, wobei eine Kategorisierung nach verschiedenen Stufen (z.B. Kantonsstrasse, Gemeindestrasse) vorzunehmen ist. Zusätzlich sind Parameter für den Strassenabstand zu ergänzen, die zusammen mit den geometrischen Inputs der Ableitung zusätzlicher Abstandsbereiche dienen. Im Rahmen der Arbeiten sollte geprüft werden, ob die bereits für die Visualisierung eingebundenen Strassenflächen mit den neuen Inputs verknüpft werden können.
- **Verfeinerung der Ableitung von Gebäudegrundrissen:**
  Die Quadtree basierte Ableitung von Gebäudegrundrissen sollte weiterentwickelt werden. Der Algorithmus zur Zusammenstellung der Grundrisse ist so zu verfeinern, dass die Grundrisse ansprechende Formen aufweisen und ungünstige Formen vermeiden werden (z.B. lange, schmale Fortsätze, die nicht genutzt werden können). Dabei ist eine Vorgabe gestalterischer Parameter in Form einer Typologie (Einzelhaus, Doppelhaus, Reihenhäuser, geschlossene oder offene Blockränder) oder einzelner Kriterien (z.B. Seitenverhältnis, Einbau von Höfen) denkbar. Zu prüfen ist in diesem Zusammenhang ausserdem die Integration von maximalen Längen und Breiten sowie die Integration von Abstandslinien mit gestalterischer Wirkung.

- **Integration zusätzlicher Konfigurationsmöglichkeiten für manuelle Eingriffe:**
  Trotz den vorgeschlagenen Anpassungen und Erweiterungen des parametrischen Modells werden Situationen bestehen bleiben, in denen das Modell mit den Standardparametern für einzelne Parzellen keine zufriedenstellenden Gebäude generieren kann. Nachträgliche, manuelle Korrekturen sind aufgrund der Betonung der vorwiegend generativen Erzeugung von Varianten sowie der engen Integration mit GIS und Datenbanken nur schwer möglich und würden bei der nächsten Speicherung der Resultate wieder überschrieben. Als Kompromiss sollen zusätzliche Möglichkeiten zur manuellen Übersteuerung von Parametern für einzelne Parzellen mittels JSON basierten Konfigurationen ergänzt werden. Neben der Einflussnahme in Problemfällen eröffnet sich dadurch zudem die Möglichkeit, grundrechtliche Sonderbestimmungen (z.B. geringere Grenzabstände oder höhere Nutzungsziffern) für einzelne Parzellen abzubilden, welche im bisherigen Modell nicht unterstützt werden.
- **Anpassung von Auswertungen:**
  In Verbindung mit den Anpassungen sowohl am parametrischen Modell als auch der Verfeinerung des Vorgehensmodells im Allgemeinen, sind die zugehörigen Analysen und Auswertungen anzupassen und zu erweitern.

Glossar

Begriff / Abkürzung	Erklärung
AV	Amtliche Vermessung
BFS	Bundesamt für Statistik der Schweizer Eidgenossenschaft
BMZ	Baumassenziffer
BREP	Boundary Representation; Methode zur Beschreibung von Flächen- oder Volumenobjekten mittels deren begrenzenden Oberflächen
DB	Datenbank
GeoJSON	Geographic JavaScript Object Notation; ein auf JSON (JavaScript Object Notation) basierendes Austauschformat für räumliche Daten
GFZ	Geschossflächenziffer
GWS	Gebäude und Wohnungsstatistik des BFS
IVHB	Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe
NFP	Nationales Forschungsprogramm
ODBC	Open Database Connectivity; eine standardisierte Programmierschnittstelle für den Zugriff auf Datenbank Management Systeme
PNG	Portable Network Graphic
STATENT	Statistik der Unternehmensstruktur des BFS
STATPOP	Statistik der Bevölkerung und der Haushalte des BFS
ÜZ	Überbauungsziffer
WKT	Well Known Text

Literaturverzeichnis

AGI (2017a) *Amtliche Vermessung vereinfacht*, available: https://www.geo.apps.be.ch/de/geodaten/suche-nach-geodaten.html?view=sheet&guid=c71b255f-67cf-4e25-bc1b-ae4dc6e7d546&catalog=geocatalog&type=complete&preview=search_list [accessed 10.08.2017].

AGI (2017b) *Übersichtszonenplan 1:25'000*, available: https://www.geo.apps.be.ch/de/geodaten/suche-nach-geodaten.html?view=sheet&guid=8512e830-3ab7-ed84-0192-9856038327cc&catalog=geocatalog&type=complete&preview=search_list [accessed 16.06.2017].

AGR (2016) *Siedlungsentwicklung nach innen: Arbeitshilfe*, Bern: ecoptima ag, available: http://www.jgk.be.ch/jgk/de/index/raumplanung/raumplanung/kantonale_raumplanung/siedlungsentwicklungnachinnen/arbeitshilfe_seinfuerdieortsplanung.assetref/dam/documents/JGK/AGR/de/Raumplanung/SEin/agr_raumplanung_siedlungsentwicklung_nach_innen_arbeitshilfe_de.pdf [accessed 10.08.2017].

AGR (2017) *SEIN: Siedlungsentwicklung nach innen Kanton Bern*, available: https://www.geo.apps.be.ch/de/geodaten/suche-nach-geodaten.html?view=sheet&guid=f3e596f6-afcb-7254-ad8b-f21a06972529&catalog=geocatalog&type=complete&preview=search_list [accessed 02.03.2018].

Angélil, M. (2015) *Kommentar zu Behaupten und glauben (Gantenbein, 2015)*, Nachrichten - Planung & Städtebau, available: https://www.hochparterre.ch/nachrichten/planung-staedtebau/blog/post/detail/behaupten-und-glauben/1438849972/ [accessed 08.02.2018].

Angélil, M., Bornhauser, R., Christiaanse, K., Hömke, M., Kissling, T., Klaus, P., Kretz, S., Kueng, L., Magnago Lampugnani, V., Muri-Koller, G., Nüssli, R., Poloni Esquivié, V., Schmid, C., Ting, C. und Vogt, G. (2016) *Urbane Qualitäten: Ein Handbuch am Beispiel der Metropolitanregion Zürich*, 1st ed., Zürich: Edition Hochparterre.

Angélil, M.C., Kees; Magnago Lampugnani, Vittorio; Schmid, Christian; Vogt, Günther (2013) *Urbane Potenziale und Strategien in metropolitanen Territorien - am Beispiel des Metropolitanraums Zürich*, Zürich.

ARE (2007) *Nachhaltige Raumentwicklung Schweiz - Kriteriensystem*, available: https://www.are.admin.ch/dam/are/de/dokumente/raumplanung/dokumente/faktenblatt/nachhaltige_raumentwicklungkriteriensystem.pdf.download.pdf/nachhaltige_raumentwicklungkriteriensystem.pdf [accessed 15.02.2018].

ARE (2011) *Minimale Geodatenmodelle Bereich Nutzungsplanung, Modelldokumentation*, Ittigen, available: https://www.are.admin.ch/are/de/home/raumentwicklung-und-raumplanung/grundlagen-und-daten/minimale-geodatenmodelle/nutzungsplanung.html [accessed 08.11.2017].
ARE (2012) *Faktenblatt - Raumplanung in der Schweiz*, Bern: Bundesamt für Raumentwicklung ARE,, available: https://www.uvek.admin.ch/dam/uvek/de/dokumente/raumentwicklung/raumplanung_in_derschweiz.pdf.download.pdf/raumplanung_in_derschweiz.pdf [accessed 06.02.2018].
ARE (2017) *Geodaten Bauzonen Schweiz (harmonisiert)*, available: http://www.ikgeo.ch/geodatenangebot/geodaten-bauzonen-schweiz.html [accessed 08.02.2018].
ARE (2018a) *Fakten und Zahlen - Flächennutzung*, available: https://www.are.admin.ch/are/de/home/raumentwicklung-und-raumplanung/grundlagen-und-daten/fakten-und-zahlen/flaechennutzung.html [accessed 06.02.2018].
ARE (2018b) *Verkehrserschliessung in der Schweiz: ÖV-Güteklassen*, available: https://data.geo.admin.ch/ch.are.gueteklassen_oev/ [accessed 25.02.2018].
ARE ZH (2015) *Die Siedlungsentwicklung nach innen umsetzen*, Zürich: Kanton Zürich, Baudirektion, Amt für Raumentwicklung, available: https://are.zh.ch/internet/baudirektion/are/de/raumplanung/formulare_merkblaetter/_jcr_content/contentPar/form_2/formitems/kein_titel_gesetzt_/download.spooler.download.1430980298962.pdf/2015_04_Leitfaden_Siedlungsentwicklung.pdf [accessed 16.02.2018].
Arnaboldi, M. (2015) *Homepage Projekt Stadt- und Landschaftsgestaltung – öffentliche Räume in der "Città Ticino" von morgen*, available: https://search.usi.ch/en/projects/501/Public-Space-Design-in-the-Citta-Ticino-of-tomorrow [accessed 15.02.2018].
Batty, M. (2013) 'Defining Geodesign (= GIS + Design?)', *Environment and Planning B: Planning and Design*, 40(1), 1-2, available: http://dx.doi.org/doi:10.1068/b4001ed.
Beirão, J., Arrobas, P. und Duarte, J. (2012) 'Parametric Urban Design: Joining morphology and urban indicators in a single interactive model', in *Digital Physicality - 30th eCAADe Conference*, Digital Physicality - Proceedings of the 30th eCAADe Conference - 12.-14. September, Czech Technical University in Prague, Faculty of Architecture (Czech Republic), pp.167-175.
Beirão, J., Nourian Ghadi Kolaee, P. und Mashhoodi, B. (2011) 'Parametric urban design: An interactive sketching system for shaping neighborhoods', in *eCAADe 2011: Proceedings of the 29th conference on education and research in computer aided architectural design in Europe" Respecting Fragile Places", Ljubljana, Slovenia, 21-24 September 2011*, 21.09.2011, eCAADe, Faculty of Architecture, University of Ljubljana.
BFS (2005) *Arealstatistik Schweiz: Zahlen - Fakten - Analysen*, Neuchâtel: Bundesamt für Statistik, available: https://www.bfs.admin.ch/bfsstatic/dam/assets/344447/master [accessed 06.02.2018].
BFS (2016a) *Gebäude- und Wohnungsstatistik (seit 2009)*, available: https://www.bfs.admin.ch/bfs/de/home/statistiken/bau-wohnungswesen/erhebungen/gws2009.html [accessed 22.02.2018].
BFS (2016b) *Statistik der Bevölkerung und der Haushalte*, available: https://www.bfs.admin.ch/bfs/de/home/statistiken/bevoelkerung/erhebungen/statpop.html [accessed 22.02.2018].
BFS (2017a) *Gebäude- und Wohnungsstatistik (GWS): Geodaten 2015*, available: https://www.bfs.admin.ch/bfs/de/home/dienstleistungen/geostat/geodaten-bundesstatistik/gebaeude-wohnungen-haushalte-personen/ergebnisse-volkszaehlung-ab-2010.html [accessed 01.10.2017].
BFS (2017b) *Neue Volkszählung, Bevölkerung, Privathaushalte: Geodaten 2015 und 2016*, available: https://www.bfs.admin.ch/bfs/de/home/dienstleistungen/geostat/geodaten-bundesstatistik/gebaeude-wohnungen-haushalte-personen/ergebnisse-volkszaehlung-ab-2010.html [accessed 01.10.2017].
BFS (2017c) *Statistik der Unternehmensstruktur*, available: https://www.bfs.admin.ch/bfs/de/home/statistiken/industrie-dienstleistungen/erhebungen/statent.html [accessed 22.02.2018].
BFS (2017d) *Statistik der Unternehmensstruktur (STATENT), Beschäftigte und Arbeitsstätten: Geodaten 2015 (provisorisch)*, available: https://www.bfs.admin.ch/bfs/de/home/dienstleistungen/geostat/geodaten-bundesstatistik/arbeitsstaetten-beschaeftigung/statistik-unternehmensstruktur-statent-ab-2011.html [accessed 01.10.2017].

BFS (2018) *Alter, Zivilstand, Staatsangehörigkeit*, available: https://www.bfs.admin.ch/bfs/de/home/statistiken/bevoelkerung/stand-entwicklung/alter-zivilstand-staatsangehoerigkeit.html [accessed 13.05.2018].
BMBV (2011) 'Verordnung über die Begriffe und Messweisen im Bauwesen (BMBV) ', 25.
BSIG (2011) *Verordnung über die Begriffe und Messweisen im Bauwesen (BMBV; BSG 721.3); Erläuterungen und Empfehlungen zur Umsetzung in die kommunalen Baureglemente, Zonenpläne und Überbauungsordnungen* Bern: Justiz-, Gemeinde- und Kirchendirektion des Kantons Bern, available: http://www.bsig.jgk.be.ch/bsig-2010-web/bsig/fileDownload?documentId=814&LANGUAGE=de [accessed 26.01.2018].
Bundesrat (2017) *Verdichtetes Bauen in Ortszentren fördern, aber wie?*, Bern, available: https://www.are.admin.ch/dam/are/de/dokumente/recht/publikationen/verdichtetes-bauen-in-ortszentren-foerdern-aber-wie.PDF.download.PDF/verdichtetes-bauen-in-ortszentren-foerdern-aber-wie-de.PDF [accessed 16.02.2018].
Bürklin, T. und Peterek, M. (2008) *Stadtbausteine*, Basel: Birkhäuser.
Butler, H., Daly, M., Doyle, A., Gillies, S., Hagen, S. und Schaub, T. (2016) *No. RFC 7946: The GeoJSON Format*: Internet Engineering Task Force (IETF).
BV (1999) 'Bundesverfassung der Schweizerischen Eidgenossenschaft', 101, 96.
cadastre.ch (2018) *Datenmodell DM.01-AV-CH*, available: https://www.cadastre.ch/de/manual-av/method/modell.html [accessed 02.04.2018].
Campagna, M. und Di Cesare, E.A. (2016) 'Geodesign: Lost in Regulations (and in Practice)' in Papa, R. and Fistola, R., eds., *Smart Energy in the Smart City: Urban Planning for a Sustainable Future*, Cham: Springer International Publishing, 307-327.
Chatzikonstantinou, Y. (2014) 'Hoopsnake Add-on for Grasshopper', 0.6.7.
Dangermond, J. (2010) 'Geodesign and GIS–designing our futures', *Peer Reviewed Proceedings of Digital Landscape Architecture, Anhalt University of Applied Science, Germany*.
de Monchaux, N. (2010) 'Local Code: Real Estates', *Architectural Design*, 80(3), 88-93, available: http://dx.doi.org/10.1002/ad.1080.
de Monchaux, N., Patwa, S., Golder, B., Jensen, S. und Lung, D. (2010) 'Local code: The critical use of geographic information systems in parametric urban design', in *Life In:formation: On Responsive Information and Variations in Architecture - Proceedings of the 30th Annual Conference of the Association for Computer Aided Design in Architecture, ACADIA 2010*, 234-242.
Delrieu, L. (2016) 'bounding inner rectangle.gh (Grasshopper Definition)'.
Delrieu, L. (2018) 'sand dune using mesh.gh1 (Grasshopper Definition)'.
Efthymiou, D., Farooq, B., Bierlaire, M. und Antoniou, C. (2013) 'Agent-Based Indicators Analysis in the Context of Policy Evaluation', in *13st Swiss Transport Research Conference*, Monte Verità/Ascona, Switzerland.
Efthymiou, D., Farooq, B., Bierlaire, M. und Antoniou, C. (2014) 'Multidimensional indicator analysis for transport policy evaluation', *Transportation Research Record: Journal of the Transportation Research Board*, 2430(1), 83-94.
Flaxman, M. (2010) 'GeoDesign: Fundamental Principles', in *Paper presented at the GeoDesign Summit*, Redlands, Esri.
Foord, M. (2010) *Launching Sub-Processes*, available: http://www.ironpython.info/index.php?title=Launching_Sub-Processes [accessed 07.04.2018].
Foster, K. (2016) 'Geodesign parsed: Placing it within the rubric of recognized design theories', *Landscape and Urban Planning*, 156, 92-100, available: http://dx.doi.org/http://dx.doi.org/10.1016/j.landurbplan.2016.06.017.
Frazer, J. (2016) 'Parametric Computation: History and Future', *Architectural Design*, 86(2), 18-23, available: http://dx.doi.org/10.1002/ad.2019.
Gantenbein, K. (2015) *Behaupten und glauben*, Nachrichten - Planung & Städtebau, available: https://www.hochparterre.ch/nachrichten/planung-staedtebau/blog/post/detail/behaupten-und-glauben/1438849972/ [accessed 08.02.2018].
Gilgen, S. (2016) 'Fallstudie Langenthal', *MODULØR*, 1.11.2016(7), 54-55, available: https://dencityblog.files.wordpress.com/2016/11/mo-2016-07-055.pdf [accessed 10.08.2017].
Gilgen, S. und Walczak, M. (2017) *Innere Entwicklung Ringstrasse, 4900 Langenthal*, Burgdorf: Berner Fachhochschule Architektur, Holz und Bau, available: https://dencityblog.files.wordpress.com/2017/03/fallstudie-langenthal.pdf [accessed 10.08.2017].
Goebel, V. und Kohler, F. (2014) *Raum mit städtischem Charakter 2012. Erläuterungsbericht*, Neuchâtel, Schweiz, available: https://www.bfs.admin.ch/bfsstatic/dam/assets/349558/master [accessed 26.10.2017].

Golder, B. (2010) *Localcode RhinoPythonScripts Code Repository*, available: https://github.com/localcode/rhinopythonscripts [accessed 08.11.2017].
Goodchild, M.F. (1992) 'Geographical information science', *International Journal of Geographical Information Systems*, 6(1), 31-45, available: http://dx.doi.org/10.1080/02693799208901893.
Goodchild, M.F. (2010) 'Towards Geodesign: Repurposing Cartography and GIS?', *Cartographic Perspective*, 66(1), 7-22, available: http://dx.doi.org/10.14714/CP66.93.
Grams, A. (2015) *Spielräume für Dichte - Problemorientierter Verfahrensansatz für Verdichtung als Element der Innenentwicklung dargestellt am Beispiel kleiner und mittlerer Gemeinden im Schweizer Mittelland*, unpublished thesis, ETH-Zürich, available: https://doi.org/10.3929/ethz-a-010587273 [accessed 22.11.2017].
Greener, S. (2012) *Generating a Grid (fishnet) of points or polygons for PostGIS*, available: https://spatialdbadvisor.com/postgis_tips_tricks/300/generating-a-grid-fishnet-of-points-or-polygons-for-postgis [accessed 22.04.2018].
Grêt-Regamey, A., Celio, E., Klein, T.M. und Wissen Hayek, U. (2013) 'Understanding ecosystem services trade-offs with interactive procedural modeling for sustainable urban planning', *Landscape and Urban Planning*, 109(1), 107-116, available: http://dx.doi.org/http://dx.doi.org/10.1016/j.landurbplan.2012.10.011.
Gröger, G., Kolbe, T.H., Nagel, C. und Häfele, K.-H. (2012) *OGC City Geography Markup Language (CityGML) Encoding Standard, version 2.0.0*: Open Geospatial Consortium.
GS-UVEK (2013) *Faktenblatt - Raumplanung in der Schweiz*, Bern: Eidgenössisches Departement für Umwelt, Verkehr, Energie und Kommunikation UVEK, available: https://www.uvek.admin.ch/dam/uvek/de/dokumente/raumentwicklung/raumplanung_in_derschweiz.pdf.download.pdf/raumplanung_in_derschweiz.pdf [accessed 06.02.2018].
Hagmann, H. (2007) 'Städtische Dichte und Baugesetzgebung' in Lampugnani, V. M., Keller, T. and Buser, B., eds., *Städtische Dichte*, Zürich: NZZ Libro – Buchverlag der Neuen Zürcher Zeitung, 139-143.
Halatsch, J. (2013) '3D Zonenplan' in Wissen Hayek, U., Eisinger, A., Stauffacher, M. and Schmitt, G., eds., *Sustainable Urban Patterns / Nachhaltige urbane Muster*, Zürich, 32-34.
Häussermann, H. (2007) 'Phänomenologie und Struktur städtischer Dichte' in Lampugnani, V. M., Keller, T. and Buser, B., eds., *Städtische Dichte*, Zürich: NZZ Libro – Buchverlag der Neuen Zürcher Zeitung, 19-30.
Heumann, A. (2011) '2d_best_fit_bounding.ghx (Grasshopper Definition)'.
Honegger, U. (2015) *Der eigentliche Skandal*, Nachrichten - Planung & Städtebau, available: https://www.hochparterre.ch/nachrichten/planung-staedtebau/blog/post/detail/der-eigentliche-skandal/1439974779/ [accessed 15.02.2018].
Huber, J. und Walczak, M. (2016a) 'ADAM: Automatic Dencity Analysis Model'.
Huber, J. und Walczak, M. (2016b) 'SCCER Mobility - Erkenntnisse aus einem digitalen Modell'.
IVHB (2005a) *Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe (IVHB)*, Bern: Interkantonale Organ Harmonisierung Baubegriffe (IOHB), available: http://www.dtap.ch/bpuk/konkordate/ivhb/index.php?eID=tx_nawsecuredl&u=0&g=0&t=1517037378&hash=a566e8cd229a167155f99d200f9ac1541bfeebb3&file=/fileadmin/Dokumente/bpuk/public/de/konkordate/ivhb/Interkantonale_Vereinbarung_ueber_die_Harmonisierung_der_Baubegriffe.pdf [accessed 26.01.2018].
IVHB (2005b) *Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe (IVHB) - Anhang 1 Begriffe und Messweisen*, Bern: Interkantonale Organ Harmonisierung Baubegriffe (IOHB), available: http://www.dtap.ch/bpuk/konkordate/ivhb/index.php?eID=tx_nawsecuredl&u=0&g=0&t=1517037378&hash=99dc9995121b849c2880112463be13169af74c43&file=/fileadmin/Dokumente/bpuk/public/de/konkordate/ivhb/Begriffe%2Bund%2BMessweisen%2B%28Anhang%2B1%29.pdf [accessed 26.01.2018].
IVHB (2018) *Website Interkantonale Vereinbarung über die Harmonisierung der Baubegriffe IVHB*, available: http://www.dtap.ch/bpuk/konkordate/ivhb/ [accessed 26.01.2018].
Jabi, W. (2013) *Parametric Design for Architecture*, London: Laurence King Publishing.
Janssen, P. und Stouffs, R. (2015) 'Types of parametric modelling', in *Int. Conf. Computer-Aided Architectural Design Research in Asia (CAADRIA 2015)*, Daegu, South Korea, pp. 157-166.
Janssen, P., Stouffs, R., Mohanty, A., Tan, E. und Li, R. (2016) 'Parametric Modelling with GIS', in *Complexity & Simplicity - Proceedings of the 34th eCAADe Conference*, Oulu, Finland.
Kelly, T. (2014) *Unwritten procedural modeling with the straight skeleton* unpublished thesis (PhD), University of Glasgow, available: http://theses.gla.ac.uk/id/eprint/4975 [accessed 31.10.2017].

KKVA (2011) 'Richtlinie Detaillierungsgrad in der amtlichen Vermessung Informationsebene Bodenbedeckung', 62.
Koenig, R., Miao, Y., Knecht, K., Buš, P. und Mei-Chih, C. (2017) 'Interactive Urban Synthesis' in Çağdaş, G., Özkar, M., Gül, L. F. and Gürer, E., eds., *Computer-Aided Architectural Design. Future Trajectories: 17th International Conference, CAAD Futures 2017, Istanbul, Turkey, July 12-14, 2017, Selected Papers*, Singapore: Springer Singapore, 23-41.
Kunze, A., Dyllong, J., Halatsch, J., Waddell, P. und Schmitt, G. (2012) 'Parametric building typologies for San Francisco Bay Area: A conceptual framework for the implementation of design code building typologies towards a parametric procedural city model', in *Digital Physicality: Proceedings of the 30th eCAADe Conference*, Prague, Czech Republic, 09/2012, Czech Technical University in Prague, Faculty of Architecture, 187-193.
Lampugnani, V.M., Keller, T.K. und Buser, B.H. (2007) *Städtische Dichte*, Zürich: Verlag Neue Zürcher Zeitung.
McHarg, I.L. (1969) *Design with Nature*, Garden City, N.Y.: Published for the American Museum of Natural History [by] the Natural History Press.
Mehaffy, M.W. (2011) 'A city is not a rhinoceros: On the aims and opportunities of morphogenetic urban design', *Built Environment*, 37(4), 479-496.
Miller, N. (2015) 'Slingshot! Add-ons for Grasshopper', 2015.8.25.
Miller, W.R. (2012) 'Introducing Geodesign: the concept', *Esri*.
Molano, R., Rodríguez, P.G., Caro, A. und Durán, M.L. (2012) 'Finding the largest area rectangle of arbitrary orientation in a closed contour', *Applied Mathematics and Computation*, 218(19), 9866-9874, available: http://dx.doi.org/https://doi.org/10.1016/j.amc.2012.03.063.
Monedero, J. (2000) 'Parametric design: a review and some experiences', *Automation in Construction*, 9(4), 369-377, available: http://dx.doi.org/http://dx.doi.org/10.1016/S0926-5805(99)00020-5.
Moura, A.C.M. (2015) 'Geodesign in Parametric Modeling of urban landscape', *Cartography and Geographic Information Science*, 42(4), 323-332, available: http://dx.doi.org/10.1080/15230406.2015.1053527.
NASA (2008) *SRTM 1 Arc Second Global*, available: http://srtm.csi.cgiar.org/SRT-ZIP/SRTM_V41/SRTM_Data_GeoTiff/srtm_38_03.zip http://srtm.csi.cgiar.org/SRT-ZIP/SRTM_V41/SRTM_Data_GeoTiff/srtm_39_03.zip [accessed 29.11.2017].
Neuenschwander, N., Wissen Hayek, U. und Grêt-Regamey, A. (2014) 'Integrating an urban green space typology into procedural 3D visualization for collaborative planning', *Computers, Environment and Urban Systems*, 48, 99-110, available: http://dx.doi.org/http://dx.doi.org/10.1016/j.compenvurbsys.2014.07.010.
NFP48 (2008) *NFP 48 "Landschaften und Lebensräume der Alpen"*, available: http://www.snf.ch/de/fokusForschung/nationale-forschungsprogramme/nfp48-landschaften-lebensraeume-alpen/Seiten/default.aspx [accessed 02.01.2018].
NFP65 (2017) *Homepage Nationales Forschungsprogramm NFP 65 Neue Urbane Qualität*, available: http://www.nfp65.ch [accessed 29.03.2017].
Piacentino, G. (2009) 'Weaverbird Add-on for Grasshopper', 0.9.0.1.
PostgreSQL Global Development Group (2017) 'PostgreSQL 9.6.5'.
Qasem, S.W. und Touir, A.A. (2015) 'Cardinal Neighbor Quadtree: a New Quadtree-based Structure for Constant-Time Neighbor Finding', *International Journal of Computer Applications*, 132(8), 22-30, available: http://dx.doi.org/10.5120/ijca2015907501.
QGIS Development Team (2017) 'QGIS 2.18 LTR 'Las Palmas''.
QGIS Development Team (2018) 'QGIS 3.0.0 'Girona''.
R Core Team (2017) 'R: A Language and Environment for Statistical Computing, Version 3.4.0'.
Rainone, A. (2017) 'Region quadtrees in Go'.
Ramsden, J. (2015) *Instantiate a Value List Grasshopper component with C#*, available: http://james-ramsden.com/instantiate-a-value-list-grasshopper-component-with-c/ [accessed 05.01.2018].
Robert McNeel & Associates (2017a) 'GhPython Add-on for Grasshopper', 0.6.0.3.
Robert McNeel & Associates (2017b) 'Grasshopper 0.9 for Rhino 5.0', 0.9.76.
Robert McNeel & Associates (2017c) 'Rhinoceros 5.0', Version 5 SR14.
ROR (2012) *Siedlungsverdichtung und urbane Qualität - Positionspapier des Rates für Raumordnung*Rat für Raumordnung (ROR), available: https://www.are.admin.ch/dam/are/de/dokumente/raumplanung/dokumente/chronologie/2012/11/ror-siedlungsverdichtungundurbanequalitaet.pdf.download.pdf/ror-siedlungsverdichtungundurbanequalitaet.pdf [accessed 26.10.2017].
RPG (2016) 'Bundesgesetz über die Raumplanung', 700.

Samet, H. (1981) 'Connected Component Labeling Using Quadtrees', *Journal of the ACM (JACM)*, 28(3), 487-501, available: http://dx.doi.org/10.1145/322261.322267.
Schmid, C. (2007) 'Die Wiederentdeckung des Städtischen in der Schweiz' in Lampugnani, V. M., Keller, T. and Buser, B., eds., *Städtische Dichte*, Zürich: NZZ Libro – Buchverlag der Neuen Zürcher Zeitung, 31-38.
Scholl, B., Vinzens, M. und Staub, B. (2013) *Testplanung - Methode mit Zukunft*, Solothurn, available: https://www.are.admin.ch/are/de/home/medien-und-publikationen/publikationen/siedlung/testplanung---methode-mit-zukunft.html [accessed 26.01.2018].
Schumacher, P. (2008) 'Parametricism as style - Parametricist manifesto', *Dark Side Club, 11th Architecture Biennale*.
Schumacher, P. (2009) 'Parametricism: A New Global Style for Architecture and Urban Design', *Architectural Design*, 79(4), 14-23, available: http://dx.doi.org/10.1002/ad.912.
Schwalbach, G. (2009) *Stadtanalyse*, Birkhäuser.
SIA (2014) *SIA 112 Modell Bauplanung*, Zürich: Schweizerischer Ingenieurs- und Architektenverein.
Smilkov, D. (2014) *Largest rectangle in a polygon*, available: https://d3plus.org/blog/behind-the-scenes/2014/07/08/largest-rect/ [accessed 17.02.2018].
Smith, L. (2016) *Metrics Maven: Meet in the Middle - Median in PostgreSQL*, available: https://www.compose.com/articles/metrics-maven-meet-in-the-middle-median-in-postgresql/ [accessed 22.04.2018].
Speranza, P. (2016) 'Using parametric methods to understand place in urban design courses', *Journal of Urban Design*, 21(5), 661-689, available: http://dx.doi.org/10.1080/13574809.2015.1092378.
Spiegel, E. (2000) 'Dichte' in Häussermann, H. H., ed., *Grossstadt - Soziologische Stichworte* VS Verlag für Sozialwissenschaften.
Steinitz, C. (2012) *A Framework for Geodesign: Changing Geography by Design*, Redlands, CA: Esri Press.
Steinitz, C. (2016) 'On change and geodesign', *Landscape and Urban Planning*, 156, 23-25, available: http://dx.doi.org/http://doi.org/10.1016/j.landurbplan.2016.09.023.
Sulzer, J. und Desax, M. (2015) *Stadtwerdung der Agglomeration - Die Suche nach einer neuen urbanen Qualität.* , Zürich: Scheidegger & Spiess.
swisstopo (2010) 'SWISSIMAGE Das digitale Farborthophotomosaik der Schweiz - Produktinformation', 15.
swisstopo (2016) 'Produktinformation swissBUILDINGS3D 2.0'.
swisstopo (2017) *swissBUILDINGS3D 2.0*, available: https://shop.swisstopo.admin.ch/de/products/landscape/build3D2 [accessed 16.10.2017].
Taleb, H. und Musleh, M.A. (2015) 'Applying urban parametric design optimisation processes to a hot climate: Case study of the UAE', *Sustainable Cities and Society*, 14, 236-253, available: http://dx.doi.org/http://dx.doi.org/10.1016/j.scs.2014.09.001.
Tulloch, D. (2012) 'Geographic information systems and landscape architectural design and scholarship. A source of heritage and tension', *Revue Internationale de Géomatique*, 22(2), 169-184, available: http://dx.doi.org/https://doi.org/10.3166/rig.22.169-184.
Tulloch, D. (2017) 'Toward a working taxonomy of geodesign practice', *Transactions in GIS*, 21(4), 635–646, available: http://dx.doi.org/10.1111/tgis.12245.
Van Wezemael, J., Strebel, I., Schmidt, M., Devecchi, L., Loepfe, M., Kübler, D. und Eberle, D. (2014) *Prozess Städtebau - Strukturen, Dynamiken und Steuerungsmodi der Raumbildung in der Gegenwart*, Zürich, available: http://www.nfp65.ch/SiteCollectionDocuments/scientificreport_vanwezemael.pdf [accessed 15.02.2018].
Vanegas, C.A., Kelly, T., Weber, B., Halatsch, J., Aliaga, D.G. und Müller, P. (2012) 'Procedural Generation of Parcels in Urban Modeling', *Computer Graphics Forum*, 31(2pt3), 681-690, available: http://dx.doi.org/10.1111/j.1467-8659.2012.03047.x.
Verzone, C. (2015) *Homepage Projekt Food Urbanism*, available: http://www.foodurbanism.org/ [accessed 15.02.2018].
VLP-ASPAN (2014) *Raumplanung in der Schweiz: Eine kurze Einführung*, Bern: Schweizerische Vereinigung für Landesplanung, available: http://www.vlp-aspan.ch/sites/default/files/at_de_1.pdf [accessed 06.02.2018].
Walczak, M. (2017) *SCCER, Phase I - Automated Dencity Analysis Model (ADAM)*, Internal Berner Fachhochschule Architektur, Holz und Bau Report, unpublished.
Walz, A., Gloor, C., Bebi, P., Fischlin, A., Lange, E., Nagel, K. und Allgöwer, B. (2008) *Virtuelle Welten – reale Entscheide? Die Alpen im Modellbaukasten: thematische Synthese zum*

Forschungsschwerpunkt V «Virtuelle Repräsentation» des Nationalen Forschungsprogramms NFP 48 «Landschaften und Lebensräume der Alpen», Zürich: vdf Hochschulverlag.

Wehrli-Schindler, B. (2015) *Urbane Qualität für Stadt und Umland - Ein Wegweiser zur Stärkung einer nachhaltigen Raumentwicklung*, Zürich: Scheidegger & Spiess.

Wiles, F. (2006) *Automatically updating a timestamp column in PostgreSQL*, available: https://www.revsys.com/tidbits/automatically-updating-a-timestamp-column-in-postgresql/ [accessed 22.04.2018].

Wissen Hayek, U., Eisinger, A., von Wirth, T., Halatsch, J., Neuenschwander, N., Kunze, A., Koltsova, A., Efthymiou, D., Farooq, B., Kurath, S., Schaefer, M., Stauffacher, M., Grêt-Regamey, A. und Schmitt, G. (2013) *Sustainable Urban Patterns / Nachhaltige urbane Muster*, Zürich.

Wissen Hayek, U., Halatsch, J., Kunze, A., Schmitt, G. und Grêt-regamey, A. (2010) 'Integrating natural resource indicators into procedural visualisation for sustainable urban green space design', 339-347.

Wissen Hayek, U., Neuenschwander, N., Kunze, A., Halatsch, J., von Wirth, T., Grêt-Regamey, A. und Schmitt, G. (2012) 'Transdisciplinary collaboration platform based on GeoDesign for securing urban quality', in Buhmann, E., Ervin, S. and Pietsch, M., eds., *DLA 2012: Dialogue on GeoDesign, 3D-Modeling and Visualization in Landscape Architecture, Bernburg and Dessau, Germany, May 31 - June 2, 2012*, Dessau, Anhalt University of Applied Sciences.

Wissen Hayek, U., Neuenschwander, N., von Wirth, T., Kunze, A., Halatsch, J., Schmitt, G. und Grêt-Regamey, A. (2014) 'Combining Three Modelling and Visualization Tools for Collaborative Planning of Urban Transformation', in *Digital Landscape Architecture 2014 – Landscape Architecture and Planning*, Zürich, Peer Reviewed Proceedings of Digital Landscape Architecture 2014 at ETH Zurich.

Wissen Hayek, U., von Wirth, T., Neuenschwander, N. und Grêt-Regamey, A. (2016) 'Organizing and facilitating Geodesign processes: Integrating tools into collaborative design processes for urban transformation', *Landscape and Urban Planning*, 156, 59-70, available: http://dx.doi.org/10.1016/j.landurbplan.2016.05.015.

Zaha Hadid Architects (2018) *Homepage Zaha Hadid Architects*, available: http://www.zaha-hadid.com/ [accessed 02.02.2018].

Anhänge

Anhang A: Verwendete Software

Die nachfolgenden Programme sind für die Umsetzung der Arbeit verwendet worden:

- Datenbank
  - PosgreSQL 9.6.5
  - PostGIS 2.4.1
- GIS
  - QGIS 3.0.0 ‘Girona’
  - ArcGIS Desktop 10.3.1 mit Data Interoperability Extension
- Datenanalyse und Visualisierung:
  - R Language and Environment for Statistical Computing, Version 3.4.0
  - Packages
    - DBI
    - RODBC
    - ggplot2
    - reshape
    - plyr
    - dplyr
    - stringr
- Parametrisches Design und Visualisierung:
  - Rhinoceros Version 5 SR14
  - Grasshopper 0.9.76 Add-on
  - Grasshopper Add-ons:
    - GhPython 0.6.0.3
    - Slingshot! 2015.8.25
    - Weaverbird 0.9.0.1
    - Hoopsnake 0.6.7

Anhang B: SQL Code Listings

Dieser Abschnitt listet SQL Code für verschiedene Zwecke innerhalb des Projektes. Man beachte, dass der Code für die Darstellung formatiert und mit zusätzlichen Kommentaren versehen wurde. Dies gilt insbesondere für die in QGIS verwendeten Queries für Layer Definitionen, welche dort auf eine einzelne Zeile reduziert werden.

B.1. Datenimport und -analyse

B.1.1. Vorbereitung Datenbank

Die folgenden Befehle bereiten eine leere PostgreSQL Datenbank für die weiteren Schritte des Datenimports vor. Neben Erstellung der PostGIS Extension werden Funktionen für die Erstellung des Hektarrasters angelegt, welche auf einem Script aus einem Blog Post basiert (Greener 2012).

```
----------------------
--- Preparing Database
----------------------

CREATE EXTENSION postgis;
CREATE EXTENSION postgis_topology;
CREATE EXTENSION postgis_sfcgal;
CREATE EXTENSION pgrouting;

CREATE SCHEMA be
       AUTHORIZATION postgres;
CREATE SCHEMA ch
       AUTHORIZATION postgres;

--------------
--Create GRID Functions
--Basing on https://spatialdbadvisor.com/postgis_tips_tricks/?pg=2
--------------
-- Create required type
DROP   TYPE IF EXISTS T_Grid CASCADE;
CREATE TYPE T_Grid AS (
  gcol  int4,
  grow  int4,
  geom geometry
);


-- Drop function is exists
DROP FUNCTION IF EXISTS ST_RegularGrid(geometry, NUMERIC, NUMERIC, BOOLEAN);
-- Now create the function
CREATE OR REPLACE FUNCTION ST_RegularGrid(p_geometry   geometry,
                                          p_TileSizeX  NUMERIC,
                                          p_TileSizeY  NUMERIC,
                                          p_point      BOOLEAN DEFAULT TRUE)
  RETURNS SETOF T_Grid AS
$BODY$
DECLARE
   v_mbr   geometry;
   v_srid  int4;
   v_halfX NUMERIC := p_TileSizeX / 2.0;
   v_halfY NUMERIC := p_TileSizeY / 2.0;
   v_loCol int4;
   v_hiCol int4;
   v_loRow int4;
   v_hiRow int4;
   v_grid  T_Grid;
BEGIN
   IF ( p_geometry IS NULL ) THEN
      RETURN;
   END IF;
   v_srid  := ST_SRID(p_geometry);
   v_mbr   := ST_Envelope(p_geometry);
   v_loCol := trunc((ST_XMIN(v_mbr) / p_TileSizeX)::NUMERIC );
   v_hiCol := CEIL( (ST_XMAX(v_mbr) / p_TileSizeX)::NUMERIC ) - 1;
   v_loRow := trunc((ST_YMIN(v_mbr) / p_TileSizeY)::NUMERIC );
   v_hiRow := CEIL( (ST_YMAX(v_mbr) / p_TileSizeY)::NUMERIC ) - 1;
   FOR v_col IN v_loCol..v_hiCol Loop
     FOR v_row IN v_loRow..v_hiRow Loop
         v_grid.gcol := v_col;
         v_grid.grow := v_row;
```

```
        IF ( p_point ) THEN
          v_grid.geom := ST_SetSRID(
                         ST_MakePoint((v_col * p_TileSizeX) + v_halfX,
                                      (v_row * p_TileSizeY) + V_HalfY),
                         v_srid);
        ELSE
          v_grid.geom := ST_SetSRID(
                         ST_MakeEnvelope((v_col * p_TileSizeX),
                                         (v_row * p_TileSizeY),
                                         (v_col * p_TileSizeX) + p_TileSizeX,
                                         (v_row * p_TileSizeY) + p_TileSizeY),
                         v_srid);
        END IF;
        RETURN NEXT v_grid;
     END Loop;
   END Loop;
END;
$BODY$
  LANGUAGE plpgsql IMMUTABLE
  COST 100
  ROWS 1000;
-- Assign ownership
ALTER FUNCTION public.st_regulargrid(geometry, NUMERIC, NUMERIC, BOOLEAN)
  OWNER TO postgres;
```

B.1.2. Erstellung Tabellen für Importdaten

Die folgenden Befehle legen Tabellen für die nachfolgenden Schritte an.

```
--------------
--Create Tables
--------------
-- Create Table: ch.grid
DROP TABLE IF EXISTS ch.grid;

CREATE TABLE ch.grid
(
    gid serial NOT NULL,
    reli bigint,
    x_koord integer,
    y_koord integer,
    CONSTRAINT grid_pkey PRIMARY KEY (gid)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.grid
    OWNER to postgres;

SELECT AddGeometryColumn('ch', 'grid', 'geom', 2056, 'POLYGON', 2 );

-- Create Table ch.statpop2015
DROP TABLE IF EXISTS ch.statpop2015;

CREATE TABLE ch.statpop2015
(
    RELI int,
    X_KOORD int,
    Y_KOORD int,
    B15BTOT smallint, B15B11 smallint, B15B12 smallint, B15B13 smallint, B15B14 smallint, B15B15
smallint, B15B16 smallint, B15B21 smallint, B15B22 smallint, B15B23 smallint, B15B24 smallint,
B15B25 smallint, B15B26 smallint, B15B27 smallint, B15B28 smallint, B15B29 smallint, B15B30
smallint,
    B15BMTOT smallint, B15BM01 smallint, B15BM02 smallint, B15BM03 smallint, B15BM04 smallint,
B15BM05 smallint, B15BM06 smallint, B15BM07 smallint, B15BM08 smallint, B15BM09 smallint,
B15BM10 smallint, B15BM11 smallint, B15BM12 smallint, B15BM13 smallint, B15BM14 smallint,
B15BM15 smallint, B15BM16 smallint, B15BM17 smallint, B15BM18 smallint, B15BM19 smallint,
    B15BWTOT smallint, B15BW01 smallint, B15BW02 smallint, B15BW03 smallint, B15BW04 smallint,
B15BW05 smallint, B15BW06 smallint, B15BW07 smallint, B15BW08 smallint, B15BW09 smallint,
B15BW10 smallint, B15BW11 smallint, B15BW12 smallint, B15BW13 smallint, B15BW14 smallint,
B15BW15 smallint, B15BW16 smallint, B15BW17 smallint, B15BW18 smallint, B15BW19 smallint,
    B15B41 smallint, B15B42 smallint, B15B43 smallint, B15B44 smallint, B15B45 smallint, B15B46
smallint, B15B51 smallint, B15B52 smallint, B15B53 smallint, B15B54 smallint, B15B55 smallint,
B15B56 smallint,
```

```
    PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.statpop2015
    OWNER to postgres;

-- Create Table ch.statpop2016
DROP TABLE IF EXISTS ch.statpop2016;

CREATE TABLE ch.statpop2016
(
    RELI int,
    X_KOORD int,
    Y_KOORD int,
    E_KOORD int,
    N_KOORD int,
    B16BTOT smallint, B16B11 smallint, B16B12 smallint, B16B13 smallint, B16B14 smallint, B16B15
smallint, B16B16 smallint, B16B21 smallint, B16B22 smallint, B16B23 smallint, B16B24 smallint,
B16B25 smallint, B16B26 smallint, B16B27 smallint, B16B28 smallint, B16B29 smallint, B16B30
smallint,
    B16BMTOT smallint, B16BM01 smallint, B16BM02 smallint, B16BM03 smallint, B16BM04 smallint,
B16BM05 smallint, B16BM06 smallint, B16BM07 smallint, B16BM08 smallint, B16BM09 smallint,
B16BM10 smallint, B16BM11 smallint, B16BM12 smallint, B16BM13 smallint, B16BM14 smallint,
B16BM15 smallint, B16BM16 smallint, B16BM17 smallint, B16BM18 smallint, B16BM19 smallint,
    B16BWTOT smallint, B16BW01 smallint, B16BW02 smallint, B16BW03 smallint, B16BW04 smallint,
B16BW05 smallint, B16BW06 smallint, B16BW07 smallint, B16BW08 smallint, B16BW09 smallint,
B16BW10 smallint, B16BW11 smallint, B16BW12 smallint, B16BW13 smallint, B16BW14 smallint,
B16BW15 smallint, B16BW16 smallint, B16BW17 smallint, B16BW18 smallint, B16BW19 smallint,
    B16B41 smallint, B16B42 smallint, B16B43 smallint, B16B44 smallint, B16B45 smallint, B16B46
smallint, B16B51 smallint, B16B52 smallint, B16B53 smallint, B16B54 smallint, B16B55 smallint,
B16B56 smallint,
    PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.statpop2016
    OWNER to postgres;

-- Create Table ch.statpop2015
DROP TABLE IF EXISTS ch.statpophh2015;

CREATE TABLE ch.statpophh2015
(
    RELI int,
    X_KOORD int,
    Y_KOORD int,
    H15PTOT smallint,
    H15P01 smallint,
    H15P02 smallint,
    H15P03 smallint,
    H15P04 smallint,
    H15P05 smallint,
    H15P06 smallint,
    H15PI smallint,
    PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.statpophh2015
    OWNER to postgres;

-- Create Table ch.statent2015
DROP TABLE IF EXISTS ch.statent2015;

CREATE TABLE ch.statent2015
(
    E_KOORD int,
```

```
    N_KOORD int,
    X_KOORD int,
    Y_KOORD int,
    RELI int,
    B1508T smallint, B1508S1 smallint, B1508S2 smallint, B1508S3 smallint,
    B1508EMPT smallint, B1508EMPTS1 smallint, B1508EMPFS1 smallint, B1508EMPMS1 smallint,
B1508EMPTS2 smallint, B1508EMPFS2 smallint, B1508EMPMS2 smallint, B1508EMPTS3 smallint,
B1508EMPFS3 smallint, B1508EMPMS3 smallint,
    B1508VZAT real, B1508VZATS1 real, B1508VZAFS1 real, B1508VZAMS1 real, B1508VZATS2 real,
B1508VZAFS2 real, B1508VZAMS2 real, B1508VZATS3 real, B1508VZAFS3 real, B1508VZAMS3 real,
    B150801AS smallint, B150802AS smallint, B150803AS smallint, B150805AS smallint, B150806AS
smallint, B150807AS smallint, B150808AS smallint, B150809AS smallint, B150810AS smallint,
B150811AS smallint, B150812AS smallint, B150813AS smallint, B150814AS smallint, B150815AS
smallint, B150816AS smallint, B150817AS smallint, B150818AS smallint, B150819AS smallint,
    B150820AS smallint, B150821AS smallint, B150822AS smallint, B150823AS smallint, B150824AS
smallint, B150825AS smallint, B150826AS smallint, B150827AS smallint, B150828AS smallint,
B150829AS smallint, B150830AS smallint, B150831AS smallint, B150832AS smallint, B150833AS
smallint, B150835AS smallint,
    B150836AS smallint, B150837AS smallint, B150838AS smallint, B150839AS smallint, B150841AS
smallint, B150842AS smallint, B150843AS smallint, B150845AS smallint, B150846AS smallint,
B150847AS smallint, B150849AS smallint, B150850AS smallint, B150851AS smallint, B150852AS
smallint, B150853AS smallint,
    B150855AS smallint, B150856AS smallint, B150858AS smallint, B150859AS smallint, B150860AS
smallint, B150861AS smallint, B150862AS smallint, B150863AS smallint, B150864AS smallint,
B150865AS smallint, B150866AS smallint, B150868AS smallint, B150869AS smallint, B150870AS
smallint, B150871AS smallint,
    B150872AS smallint, B150873AS smallint, B150874AS smallint, B150875AS smallint, B150877AS
smallint, B150878AS smallint, B150879AS smallint, B150880AS smallint, B150881AS smallint,
B150882AS smallint, B150884AS smallint, B150885AS smallint, B150886AS smallint, B150887AS
smallint, B150888AS smallint, B150890AS smallint, B150891AS smallint, B150892AS smallint,
B150893AS smallint, B150894AS smallint, B150895AS smallint, B150896AS smallint,
    B150801EMP smallint, B150802EMP smallint, B150803EMP smallint, B150805EMP smallint,
B150806EMP smallint, B150807EMP smallint, B150808EMP smallint, B150809EMP smallint, B150810EMP
smallint, B150811EMP smallint, B150812EMP smallint, B150813EMP smallint, B150814EMP smallint,
B150815EMP smallint, B150816EMP smallint, B150817EMP smallint, B150818EMP smallint, B150819EMP
smallint, B150820EMP smallint, B150821EMP smallint, B150822EMP smallint, B150823EMP smallint,
B150824EMP smallint, B150825EMP smallint, B150826EMP smallint, B150827EMP smallint, B150828EMP
smallint, B150829EMP smallint, B150830EMP smallint, B150831EMP smallint, B150832EMP smallint,
B150833EMP smallint, B150835EMP smallint, B150836EMP smallint, B150837EMP smallint, B150838EMP
smallint, B150839EMP smallint, B150841EMP smallint, B150842EMP smallint, B150843EMP smallint,
B150845EMP smallint, B150846EMP smallint, B150847EMP smallint, B150849EMP smallint, B150850EMP
smallint, B150851EMP smallint, B150852EMP smallint, B150853EMP smallint, B150855EMP smallint,
B150856EMP smallint, B150858EMP smallint, B150859EMP smallint, B150860EMP smallint, B150861EMP
smallint, B150862EMP smallint, B150863EMP smallint, B150864EMP smallint, B150865EMP smallint,
B150866EMP smallint, B150868EMP smallint, B150869EMP smallint, B150870EMP smallint, B150871EMP
smallint, B150872EMP smallint, B150873EMP smallint, B150874EMP smallint, B150875EMP smallint,
B150877EMP smallint, B150878EMP smallint, B150879EMP smallint, B150880EMP smallint, B150881EMP
smallint, B150882EMP smallint, B150884EMP smallint, B150885EMP smallint, B150886EMP smallint,
B150887EMP smallint, B150888EMP smallint, B150890EMP smallint, B150891EMP smallint, B150892EMP
smallint, B150893EMP smallint, B150894EMP smallint, B150895EMP smallint, B150896EMP smallint,
    B150801VZA real, B150802VZA real, B150803VZA real, B150805VZA real, B150806VZA real,
B150807VZA real, B150808VZA real, B150809VZA real, B150810VZA real, B150811VZA real, B150812VZA
real, B150813VZA real, B150814VZA real, B150815VZA real, B150816VZA real, B150817VZA real,
B150818VZA real, B150819VZA real, B150820VZA real, B150821VZA real, B150822VZA real, B150823VZA
real, B150824VZA real, B150825VZA real, B150826VZA real, B150827VZA real, B150828VZA real,
B150829VZA real, B150830VZA real, B150831VZA real, B150832VZA real, B150833VZA real, B150835VZA
real, B150836VZA real, B150837VZA real, B150838VZA real, B150839VZA real, B150841VZA real,
B150842VZA real, B150843VZA real, B150845VZA real, B150846VZA real, B150847VZA real, B150849VZA
real, B150850VZA real, B150851VZA real, B150852VZA real, B150853VZA real, B150855VZA real,
B150856VZA real, B150858VZA real, B150859VZA real, B150860VZA real, B150861VZA real, B150862VZA
real, B150863VZA real, B150864VZA real, B150865VZA real, B150866VZA real, B150868VZA real,
B150869VZA real, B150870VZA real, B150871VZA real, B150872VZA real, B150873VZA real, B150874VZA
real, B150875VZA real, B150877VZA real, B150878VZA real, B150879VZA real, B150880VZA real,
B150881VZA real, B150882VZA real, B150884VZA real, B150885VZA real, B150886VZA real, B150887VZA
real, B150888VZA real, B150890VZA real, B150891VZA real, B150892VZA real, B150893VZA real,
B150894VZA real, B150895VZA real, B150896VZA real,
    B15080101 smallint, B15080102 smallint, B15080103 smallint, B15080104 smallint, B15080201
smallint, B15080202 smallint, B15080203 smallint, B15080204 smallint, B15080301 smallint,
B15080302 smallint, B15080303 smallint, B15080304 smallint, B15080501 smallint, B15080502
smallint, B15080503 smallint, B15080504 smallint, B15080601 smallint, B15080602 smallint,
B15080603 smallint, B15080604 smallint, B15080701 smallint, B15080702 smallint, B15080703
smallint, B15080704 smallint, B15080801 smallint, B15080802 smallint, B15080803 smallint,
B15080804 smallint, B15080901 smallint, B15080902 smallint, B15080903 smallint, B15080904
smallint, B15081001 smallint, B15081002 smallint, B15081003 smallint, B15081004 smallint,
B15081101 smallint, B15081102 smallint, B15081103 smallint, B15081104 smallint, B15081201
smallint, B15081202 smallint, B15081203 smallint, B15081204 smallint, B15081301 smallint,
B15081302 smallint, B15081303 smallint, B15081304 smallint, B15081401 smallint, B15081402
```

```
smallint,  B15081403  smallint,  B15081404  smallint,  B15081501  smallint,  B15081502  smallint,
B15081503  smallint,  B15081504  smallint,  B15081601  smallint,  B15081602  smallint,  B15081603
smallint,  B15081604  smallint,  B15081701  smallint,  B15081702  smallint,  B15081703  smallint,
B15081704  smallint,  B15081801  smallint,  B15081802  smallint,  B15081803  smallint,  B15081804
smallint,  B15081901  smallint,  B15081902  smallint,  B15081903  smallint,  B15081904  smallint,
B15082001  smallint,  B15082002  smallint,  B15082003  smallint,  B15082004  smallint,  B15082101
smallint,  B15082102  smallint,  B15082103  smallint,  B15082104  smallint,  B15082201  smallint,
B15082202  smallint,  B15082203  smallint,  B15082204  smallint,  B15082301  smallint,  B15082302
smallint,  B15082303  smallint,  B15082304  smallint,  B15082401  smallint,  B15082402  smallint,
B15082403  smallint,  B15082404  smallint,  B15082501  smallint,  B15082502  smallint,  B15082503
smallint,  B15082504  smallint,  B15082601  smallint,  B15082602  smallint,  B15082603  smallint,
B15082604  smallint,  B15082701  smallint,  B15082702  smallint,  B15082703  smallint,  B15082704
smallint,  B15082801  smallint,  B15082802  smallint,  B15082803  smallint,  B15082804  smallint,
B15082901  smallint,  B15082902  smallint,  B15082903  smallint,  B15082904  smallint,  B15083001
smallint,  B15083002  smallint,  B15083003  smallint,  B15083004  smallint,  B15083101  smallint,
B15083102  smallint,  B15083103  smallint,  B15083104  smallint,  B15083201  smallint,  B15083202
smallint,  B15083203  smallint,  B15083204  smallint,  B15083301  smallint,  B15083302  smallint,
B15083303  smallint,  B15083304  smallint,  B15083501  smallint,  B15083502  smallint,  B15083503
smallint,  B15083504  smallint,  B15083601  smallint,  B15083602  smallint,  B15083603  smallint,
B15083604  smallint,  B15083701  smallint,  B15083702  smallint,  B15083703  smallint,  B15083704
smallint,  B15083801  smallint,  B15083802  smallint,  B15083803  smallint,  B15083804  smallint,
B15083901  smallint,  B15083902  smallint,  B15083903  smallint,  B15083904  smallint,  B15084101
smallint,  B15084102  smallint,  B15084103  smallint,  B15084104  smallint,  B15084201  smallint,
B15084202  smallint,  B15084203  smallint,  B15084204  smallint,  B15084301  smallint,  B15084302
smallint,  B15084303  smallint,  B15084304  smallint,  B15084501  smallint,  B15084502  smallint,
B15084503  smallint,  B15084504  smallint,  B15084601  smallint,  B15084602  smallint,  B15084603
smallint,  B15084604  smallint,  B15084701  smallint,  B15084702  smallint,  B15084703  smallint,
B15084704  smallint,  B15084901  smallint,  B15084902  smallint,  B15084903  smallint,  B15084904
smallint,  B15085001  smallint,  B15085002  smallint,  B15085003  smallint,  B15085004  smallint,
B15085101  smallint,  B15085102  smallint,  B15085103  smallint,  B15085104  smallint,  B15085201
smallint,  B15085202  smallint,  B15085203  smallint,  B15085204  smallint,  B15085301  smallint,
B15085302  smallint,  B15085303  smallint,  B15085304  smallint,  B15085501  smallint,  B15085502
smallint,  B15085503  smallint,  B15085504  smallint,  B15085601  smallint,  B15085602  smallint,
B15085603  smallint,  B15085604  smallint,  B15085801  smallint,  B15085802  smallint,  B15085803
smallint,  B15085804  smallint,  B15085901  smallint,  B15085902  smallint,  B15085903  smallint,
B15085904  smallint,  B15086001  smallint,  B15086002  smallint,  B15086003  smallint,  B15086004
smallint,  B15086101  smallint,  B15086102  smallint,  B15086103  smallint,  B15086104  smallint,
B15086201  smallint,  B15086202  smallint,  B15086203  smallint,  B15086204  smallint,  B15086301
smallint,  B15086302  smallint,  B15086303  smallint,  B15086304  smallint,  B15086401  smallint,
B15086402  smallint,  B15086403  smallint,  B15086404  smallint,  B15086501  smallint,  B15086502
smallint,  B15086503  smallint,  B15086504  smallint,  B15086601  smallint,  B15086602  smallint,
B15086603  smallint,  B15086604  smallint,  B15086801  smallint,  B15086802  smallint,  B15086803
smallint,  B15086804  smallint,  B15086901  smallint,  B15086902  smallint,  B15086903  smallint,
B15086904  smallint,  B15087001  smallint,  B15087002  smallint,  B15087003  smallint,  B15087004
smallint,  B15087101  smallint,  B15087102  smallint,  B15087103  smallint,  B15087104  smallint,
B15087201  smallint,  B15087202  smallint,  B15087203  smallint,  B15087204  smallint,  B15087301
smallint,  B15087302  smallint,  B15087303  smallint,  B15087304  smallint,  B15087401  smallint,
B15087402  smallint,  B15087403  smallint,  B15087404  smallint,  B15087501  smallint,  B15087502
smallint,  B15087503  smallint,  B15087504  smallint,  B15087701  smallint,  B15087702  smallint,
B15087703  smallint,  B15087704  smallint,  B15087801  smallint,  B15087802  smallint,  B15087803
smallint,  B15087804  smallint,  B15087901  smallint,  B15087902  smallint,  B15087903  smallint,
B15087904  smallint,  B15088001  smallint,  B15088002  smallint,  B15088003  smallint,  B15088004
smallint,  B15088101  smallint,  B15088102  smallint,  B15088103  smallint,  B15088104  smallint,
B15088201  smallint,  B15088202  smallint,  B15088203  smallint,  B15088204  smallint,  B15088401
smallint,  B15088402  smallint,  B15088403  smallint,  B15088404  smallint,  B15088501  smallint,
B15088502  smallint,  B15088503  smallint,  B15088504  smallint,  B15088601  smallint,  B15088602
smallint,  B15088603  smallint,  B15088604  smallint,  B15088701  smallint,  B15088702  smallint,
B15088703  smallint,  B15088704  smallint,  B15088801  smallint,  B15088802  smallint,  B15088803
smallint,  B15088804  smallint,  B15089001  smallint,  B15089002  smallint,  B15089003  smallint,
B15089004  smallint,  B15089101  smallint,  B15089102  smallint,  B15089103  smallint,  B15089104
smallint,  B15089201  smallint,  B15089202  smallint,  B15089203  smallint,  B15089204  smallint,
B15089301  smallint,  B15089302  smallint,  B15089303  smallint,  B15089304  smallint,  B15089401
smallint,  B15089402  smallint,  B15089403  smallint,  B15089404  smallint,  B15089501  smallint,
B15089502  smallint,  B15089503  smallint,  B15089504  smallint,  B15089601  smallint,  B15089602
smallint, B15089603 smallint, B15089604 smallint,
    PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.statent2015
    OWNER to postgres;

-- Create Table ch.gws2015
DROP TABLE IF EXISTS ch.gws2015;
```

```
CREATE TABLE ch.gws2015
(
    RELI int,
    X_KOORD int,
    Y_KOORD int,
G15TOT smallint, G15A01 smallint, G15A02 smallint, G15A03 smallint, G15A05 smallint, G15A06
smallint, G15A07 smallint, G15A08 smallint, G15B01 smallint, G15B02 smallint, G15B03 smallint,
G15B04 smallint, G15B05 smallint, G15B06 smallint, G15B07 smallint, G15B08 smallint, G15B09
smallint, G15B10 smallint, G15B11 smallint, G15G01 smallint, G15G02 smallint, G15G03 smallint,
G15G04 smallint, G15G05 smallint, G15G06 smallint, G15G07 smallint, G15G08 smallint, G15G09
smallint, G15H01 smallint, G15H02 smallint, G15H03 smallint, G15H04 smallint, G15H05 smallint,
G15H17 smallint, G15H19 smallint, G15E01 smallint, G15E02 smallint, G15E03 smallint, G15E04
smallint, G15E05 smallint, G15E06 smallint, G15E07 smallint, G15E08 smallint, G15E17 smallint,
G15E19 smallint, G15EWW01 smallint, G15EWW02 smallint, G15EWW03 smallint, G15EWW04 smallint,
G15EWW05 smallint, G15EWW06 smallint, G15EWW07 smallint, G15EWW08 smallint, G15EWW17 smallint,
G15EWW19 smallint, G15W00 smallint, G15W01 smallint, G15W02 smallint, G15W03 smallint, G15W04
smallint, G15W05 smallint, G15W06 smallint, G15W07 smallint, G15W10 smallint,
    W15TOT smallint, W15TF int, W15T01 smallint, W15T02 smallint, W15T03 smallint, W15T04
smallint, W15T05 smallint, W15T06 smallint, W15T07 smallint, W15T08 smallint, W15T09 smallint,
W15T10 smallint, W15T11 smallint, W15T1 smallint, W15T1F smallint, W15T101 smallint, W15T102
smallint, W15T103 smallint, W15T104 smallint, W15T105 smallint, W15T106 smallint, W15T107
smallint, W15T108 smallint, W15T109 smallint, W15T110 smallint, W15T111 smallint, W15T2
smallint, W15T2F smallint, W15T201 smallint, W15T202 smallint, W15T203 smallint, W15T204
smallint, W15T205 smallint, W15T206 smallint, W15T207 smallint, W15T208 smallint, W15T209
smallint, W15T210 smallint, W15T211 smallint, W15T3 smallint, W15T3F smallint, W15T301 smallint,
W15T302 smallint, W15T303 smallint, W15T304 smallint, W15T305 smallint, W15T306 smallint,
W15T307 smallint, W15T308 smallint, W15T309 smallint, W15T310 smallint, W15T311 smallint, W15T4
smallint, W15T4F smallint, W15T401 smallint, W15T402 smallint, W15T403 smallint, W15T404
smallint, W15T405 smallint, W15T406 smallint, W15T407 smallint, W15T408 smallint, W15T409
smallint, W15T410 smallint, W15T411 smallint, W15T5 smallint, W15T5F smallint, W15T501 smallint,
W15T502 smallint, W15T503 smallint, W15T504 smallint, W15T505 smallint, W15T506 smallint,
W15T507 smallint, W15T508 smallint, W15T509 smallint, W15T510 smallint, W15T511 smallint, W15T6
smallint, W15T6F smallint, W15T601 smallint, W15T602 smallint, W15T603 smallint, W15T604
smallint, W15T605 smallint, W15T606 smallint, W15T607 smallint, W15T608 smallint, W15T609
smallint, W15T610 smallint, W15T611 smallint,
PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.gws2015
    OWNER to postgres;

-- Create Table ch.landuse2016
DROP TABLE IF EXISTS ch.landuse2016;

CREATE TABLE ch.landuse2016
(
    X int,
    Y int,
    RELI int,
    GMDE smallint,
    FJ85 smallint,
    FJ97 smallint,
    FJ09 smallint,
    LU85_46 smallint,
    LU97_46 smallint,
    LU09_46 smallint,
    LU85_10 smallint,
    LU97_10 smallint,
    LU09_10 smallint,
    LU85_4 smallint,
    LU97_4 smallint,
    LU09_4 smallint,
    PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.landuse2016
    OWNER to postgres;
```

```
-- Create Table ch.lancover2016
DROP TABLE IF EXISTS ch.landcover2016;

CREATE TABLE ch.landcover2016
(
    X int,
    Y int,
    RELI int,
    GMDE smallint,
    FJ85 smallint,
    FJ97 smallint,
    FJ09 smallint,
    LC85_27 smallint,
    LC97_27 smallint,
    LC09_27 smallint,
    LC85_6 smallint,
    LC97_6 smallint,
    LC09_6 smallint,
    PRIMARY KEY (RELI)
)
WITH (
    OIDS = FALSE
)
TABLESPACE pg_default;

ALTER TABLE ch.landcover2016
    OWNER to postgres;
```

B.1.3. Befehle Datenimport Kommandozeile

Die folgenden Befehle wurden auf der Kommandozeile ausgeführt, um Esri Shapefiles in die Datenbank zu importieren.

```
REM Datenimport Shapefiles
shp2pgsql -I -s 21781:2056
c:/geodaten/ch/ch.are.gueteklassen_oev_2018/ch.are.gueteklassen_oev/Oev_Gueteklassen_ARE.shp
ch.Oev_Gueteklassen_ARE |  psql -U postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/be/MOPUBE/LV95/data/MOPUBE_BBF.shp be.mopube_bodenbedeckung |
psql -U postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/be/MOPUBE/LV95/data/MOPUBE_LIF.shp be.mopube_grundstueck |
psql -U postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/be/MOPUBE/LV95/data/MOPUBE_HGF.shp be.mopube_gemeindegrenzen
|  psql -U postgres -d dencity

shp2pgsql -I -s 2056 C:/geodaten/be/UZP/LV95/data/UZP_BAU.shp be.uzp_bauzonen |  psql -U
postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/be/ADMGDE/LV95/data/ADMGDE_GDEDAT.shp be.admgde_gedat |  psql
-U postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/be/MOPUBE/LV95/data/MOPUBE_GAEINP.shp be.mopube_gaeinp |
psql -U postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/be/GEBADR/LV95/data/GEBADR_GADR.shp be.gebadr |  psql -U
postgres -d dencity

shp2pgsql -I -s 2056 c:/geodaten/ch/BauzonenSchweiz_ZonesabatirSuisse_2017/ch_are_bauzonen.shp
ch.are_bauzonen |  psql -U postgres -d dencity
```

Die folgenden Befehle wurden auf der Kommandozeile ausgeführt, um GeoTIFF Dateien in die Datenbank zu importieren.

```
REM Datenimport GeoTIFF
raster2pgsql -c -s 2056   -t 10x10   c:/geodaten/ch/srtm/srtm_38-39_05_lv95.tif ch.srtm_dhm |
psql -U postgres -d dencity
```

B.1.4. Befehle Datenimport PostgreSQL

Die folgenden Befehle werden in PostgreSQL ausgeführt, um CSV Daten in die zuvor vorbereiteten Tabellen zu importieren.

```
-------------
--Import Data
-------------

--Import STATPOP
COPY ch.statpop2015 FROM 'c:/geodaten/ch/geostat/Population/gd-b-00.03-10-
vz2015statpopb/STATPOP2015B.csv' (FORMAT csv,  HEADER,  DELIMITER ',');
COPY ch.statpophh2015 FROM 'c:/geodaten/ch/geostat/Population/gd-b-00.03-10-
vz2015statpopb/STATPOP2015_HH.csv' (FORMAT csv,  HEADER,  DELIMITER ',');
COPY ch.statpop2016 FROM 'c:/geodaten/ch/geostat/Population/gd-b-00.03-10-
vz2016statpopb/STATPOP2016B.csv' (FORMAT csv,  HEADER,  DELIMITER ',');
--Import Landuse
COPY ch.landuse2016 FROM
'c:/geodaten/ch/geostat/Arealstatistik/Entire_Switzerland_3_surveys/Land_Use/gd-b-00.03-37-
nolu04P/AREA_NOLU04_46_161114.csv' (FORMAT csv,  HEADER,  DELIMITER ',');
--Import landcover
COPY ch.landcover2016 FROM
'C:/geodaten/ch/geostat/Arealstatistik/Entire_Switzerland_3_surveys/Land_Cover/gd-b-00.03-37-
nolc04P/AREA_NOLC04_27_161114.csv' (FORMAT csv,  HEADER,  DELIMITER ',');
--Import GWS
COPY ch.gws2015 FROM 'c:/geodaten/ch/geostat/Buildings/gd-b-00.03-10-vz2015gwsb/GWS2015B.csv'
(FORMAT csv,  HEADER,  DELIMITER ',');
--Import Statent
COPY ch.statent2015 FROM 'c:/geodaten/ch/geostat/Enterrises/gd-b-00.03-22-
STATENT2015B/STATENT2015_N08_V170824B.csv' (FORMAT csv, HEADER, DELIMITER ',');
```

B.1.5. Datenaufbereitung

Die folgenden Befehle dienen der Erzeugung des Hektarrasters für die ganze Schweiz.

```
--
-- Filling Grid
--
-- Insert LV03 Geometries
INSERT INTO ch.grid(reli, x_koord, y_koord, geom)
         SELECT (gcol-20000)*10000+(grow-10000), (gcol-20000)*100, (grow-10000)*100, geom
           FROM
             public.ST_RegularGrid(
               ST_GeomFromText('LINESTRING(2485400 175200,2833900 1296000)',2056)
               ,100,100,false
             )
         WHERE  ((gcol-20000)*10000+(grow-1000)) IN (SELECT RELI FROM ch.landuse2016);

CREATE INDEX ch_grid_geom_idx ON ch.grid USING GIST ( geom);

CREATE INDEX ch_grid_reli_idx
   ON ch.grid USING btree (reli ASC NULLS LAST);


SELECT AddGeometryColumn('ch', 'grid', 'geom_lv03', 21781, 'POLYGON', 2 );

UPDATE ch.grid
  SET
    geom_lv03 = ST_Transform(geom,21781);

CREATE INDEX grid_geom_lv03_idx ON ch.grid USING GIST (geom_lv03);
```

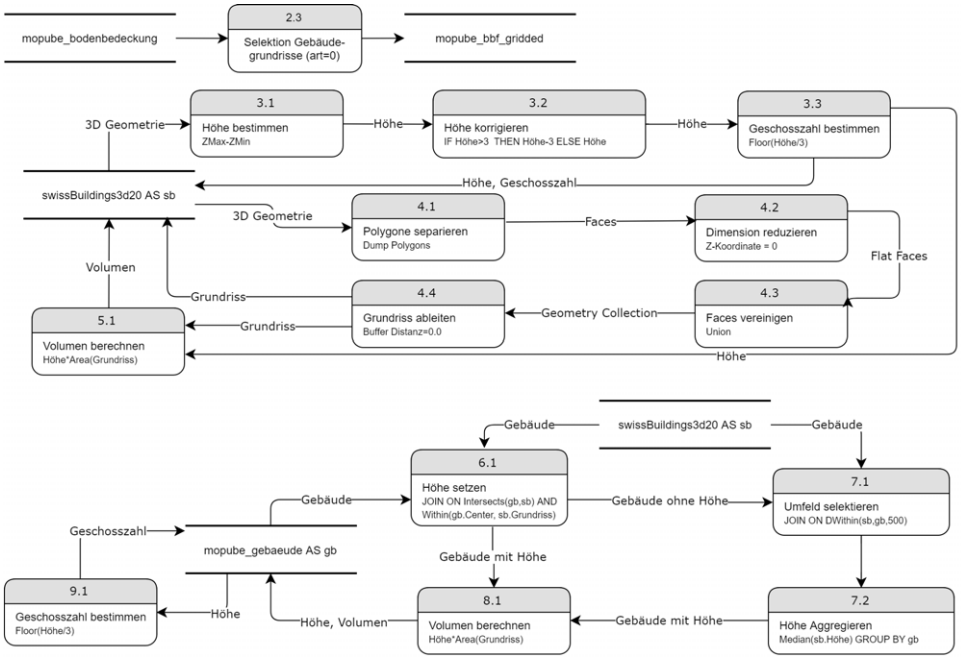
Die folgenden Befehle dienen der Aufbereitung der Daten zur Bodenbedeckung aus der Amtlichen Vermessung.

```
--Bodenbedeckung
--Populate metadata of geometry column
SELECT Populate_Geometry_Columns('be.mopube_bodenbedeckung'::regclass);

--Correct invalid geometries
UPDATE be.mopube_bodenbedeckung
  SET
    geom = St_Multi(ST_Buffer(ST_MakeValid(geom),0.0))
  WHERE NOT ST_IsValid(geom);
```

```
--Generate table with intersection of grid with mopube_bbf
DROP TABLE IF EXISTS be.mopube_bbf_gridded CASCADE;
SELECT bbf.gid, bbfid, art, qualitaet, gwr_egid, begid, guelteintr, bfsnr, shape_leng,
       shape_area, ST_Intersection(g.geom,bbf.geom) AS geom, reli, x_koord, y_koord,
       row_number() over() AS ngid
  INTO be.mopube_bbf_gridded
  FROM be.mopube_bodenbedeckung AS bbf INNER JOIN ch.grid g ON ST_Intersects(g.geom,bbf.geom)
  WHERE
        x_koord BETWEEN 556100 AND 677800 AND
        y_koord BETWEEN 130400 AND 243900
;

CREATE INDEX be_mopube_bbf_gridded_reli_idx
    ON be.mopube_bbf_gridded USING btree
    (reli ASC NULLS LAST)
    TABLESPACE pg_default;
```

B.1.6. Datenaufbereitung Gebäude

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Aufbereitung der gebäudebezogenen Daten auf.



Quelle: eigene Darstellung

Abbildung 55: Datenfluss Aufbereitung von gebäudebezogenen Daten

Die folgenden Befehle bereiten die swissBUILDINGS3D 2.0 Daten auf, wobei der Gebäudegrundriss abgeleitet sowie die Gebäudehöhe und das Bauvolumen abgeschätzt werden.

```
-- Baumasse
ALTER TABLE ch.swissbuildings3d20
  ADD COLUMN building_height real;

ALTER TABLE ch.swissbuildings3d20
  ADD COLUMN building_area real;

ALTER TABLE ch.swissbuildings3d20
  ADD COLUMN building_volume real;

ALTER TABLE ch.swissbuildings3d20
  ADD COLUMN building_storeys real;

SELECT AddGeometryColumn('ch', 'swissbuildings3d20', 'geom_footprint_base', 2056,
'MULTIPOLYGON', 3 );
CREATE INDEX ch_swissbuildings3d20_geom_footprint_base_idx ON ch.swissbuildings3d20 USING GIST
(geom_footprint_base);
```

```
SELECT AddGeometryColumn('ch', 'swissbuildings3d20', 'geom_footprint', 2056, 'MULTIPOLYGON', 2
);
CREATE INDEX ch_swissbuildings3d20_geom_footprint_idx ON ch.swissbuildings3d20 USING GIST
(geom_footprint);

UPDATE ch.swissbuildings3d20
  SET
    geom_footprint =
ST_Multi(ST_Buffer(ST_SetSRID(ST_Force2D(Replace(ST_AsText(geom),'POLYHEDRALSURFACE','MULTIPOL
YGON')),2056),0.0)),
    geom_footprint_base = ST_Multi((SELECT ST_Buffer(ST_Collect((i.poly).geom),0.0) AS geom
                            FROM (SELECT ST_Dump(geom) AS poly) AS i
                            WHERE ST_ZMax((i.poly).geom)=ST_ZMin(geom))),
    building_height = ST_ZMax(geom)-ST_ZMin(geom)-3;

UPDATE ch.swissbuildings3d20
  SET
       building_height = building_height+3
    WHERE building_height<0;

UPDATE ch.swissbuildings3d20
  SET
       building_area = ST_Area(geom_footprint),
       building_volume = ST_Area(geom_footprint)*building_height,
       building_storeys = Floor(building_height/3);

UPDATE ch.swissbuildings3d20
  SET
       building_storeys = 1
    WHERE  building_storeys=0;

SELECT Populate_Geometry_Columns('ch.swissbuildings3d20'::regclass);
```

Die folgenden Befehle dienen der Isolierung der Gebäudegrundrisse aus der Bodenbedeckung der Amtlichen Vermessung.

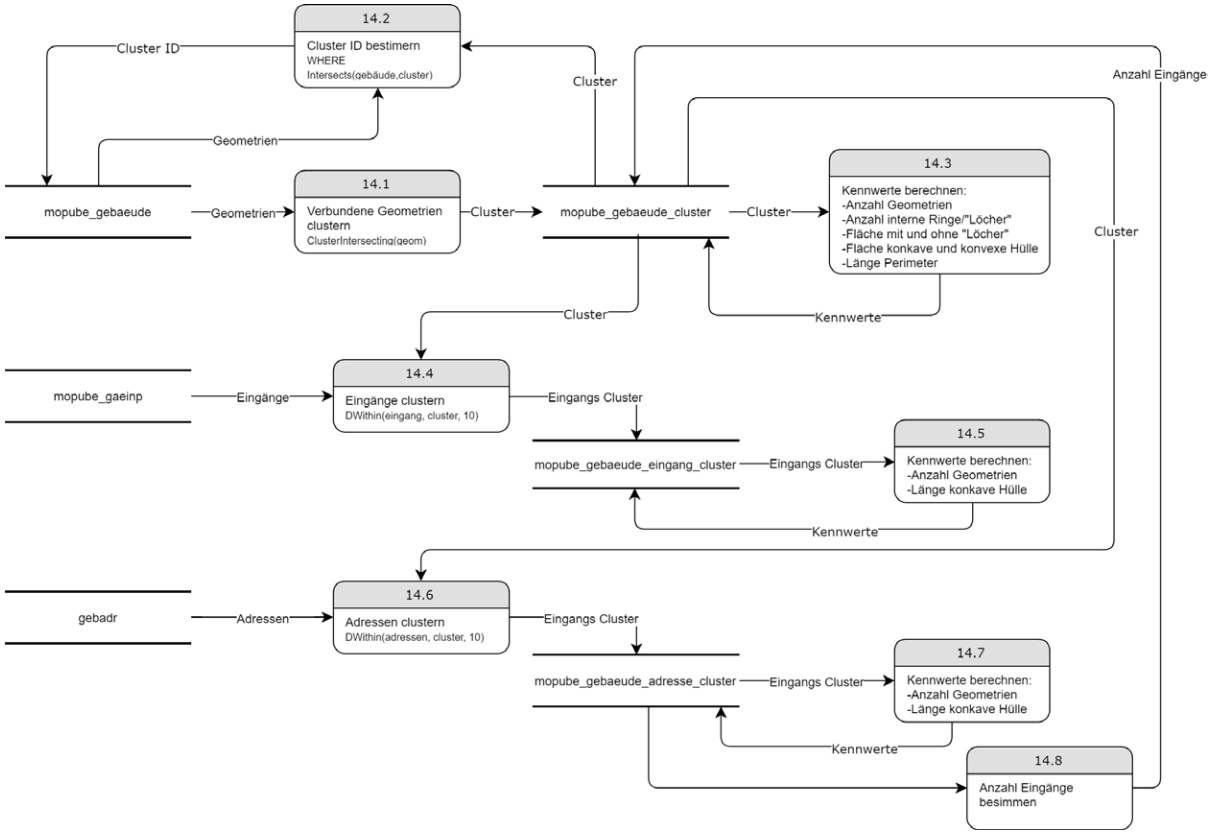
```
-- Create Table be.mopube_gebaeude
DROP TABLE IF EXISTS be.mopube_gebaeude;

CREATE TABLE be.mopube_gebaeude
(
  gid serial NOT NULL,
  bbf_gid integer NOT NULL,
  bbfid character varying(32),
  art integer,
  qualitaet integer,
  gwr_egid integer,
  begid integer,
  guelteintr date,
  bfsnr integer,
  cluster_gid integer,
  buildingtypology smallint,
  building_height real,
  building_area real,
  building_volume real,
  building_storeys real,
  shape_leng numeric,
  shape_area numeric,
  geom geometry(MultiPolygon,2056),
  CONSTRAINT mopube_gebaeude_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE be.mopube_gebaeude
  OWNER TO postgres;

CREATE INDEX mopube_gebaeude_geom_idx
  ON be.mopube_gebaeude
  USING gist
  (geom);

INSERT INTO be.mopube_gebaeude(
            bbf_gid, bbfid, art, qualitaet, gwr_egid, begid, guelteintr, bfsnr,
            shape_leng, shape_area, geom)
SELECT gid, bbfid, art, qualitaet, gwr_egid, begid, guelteintr, bfsnr,
```

```
      shape_leng, shape_area, geom
  FROM be.mopube_bodenbedeckung
  WHERE art=0;
```

Die folgenden Befehle dienen der Ergänzung der isolierten Gebäude um Höhe und Volumen basierend auf den swissBUILDINGS3D Daten. Wird keine direkte Übereinstimmung der Gebäudegrundrisse gefunden, so wird der Median der umliegenden Gebäude im Umkreis von 100 Meter verwendet. Die Medianberechnung basiert auf einem Blog Post zum Thema Medianberechnung in PostgreSQL (Smith 2016).

```
UPDATE be.mopube_gebaeude AS g
  SET
    building_area = ST_Area(geom);

UPDATE be.mopube_gebaeude AS g
  SET
    building_height = sb.building_height,
    building_storeys = sb.building_storeys
  FROM ch.swissbuildings3d20 AS sb WHERE ST_Intersects(g.geom,sb.geom_footprint) AND
ST_Within(ST_PointOnSurface(g.geom),sb.geom_footprint);

UPDATE be.mopube_gebaeude AS g
  SET
    building_height = sb.building_height,
    building_storeys = sb.building_storeys
  FROM ch.swissbuildings3d20 AS sb
  WHERE ST_DWithin(g.geom,sb.geom_footprint,1)
  AND sb.objektart IN (
          'Kapelle','Turm','Kuehlturm','Sakrales Gebaeude','Gebaeude
Einzelhaus','Hochhaus','Hochkamin','Im Bau','Sakraler Turm','Offenes Gebaeude','Treibhaus'
          --'Mauer gross','','Unterirdisches Gebaeude','Gebaeude unsichtbar','Bruecke
gedeckt','Lagertank','Verbindungsbruecke','Flugdach','Mauer gross gedeckt','Lueftungsschacht'
     )
AND g.building_height IS NULL;

UPDATE be.mopube_gebaeude AS g
  SET
    building_height = (
      --https://www.compose.com/articles/metrics-maven-meet-in-the-middle-median-in-
postgresql/
      SELECT
          ROUND(PERCENTILE_CONT(0.50) WITHIN GROUP (ORDER BY sb.building_height)::numeric, 2)
AS median_building_height
        FROM ch.swissbuildings3d20 AS sb WHERE ST_DWithin(g.geom,sb.geom_footprint,100)
AND objektart IN (
          'Kapelle','Turm','Kuehlturm','Sakrales Gebaeude','Gebaeude
Einzelhaus','Hochhaus','Hochkamin','Im Bau','Sakraler Turm','Offenes Gebaeude','Treibhaus'
          --'Mauer gross','','Unterirdisches Gebaeude','Gebaeude unsichtbar','Bruecke
gedeckt','Lagertank','Verbindungsbruecke','Flugdach','Mauer gross gedeckt','Lueftungsschacht'
     )
)
  WHERE g.building_height IS NULL;

UPDATE be.mopube_gebaeude AS g
  SET
    building_volume = building_height*building_area
  WHERE building_height IS NOT NULL;

UPDATE be.mopube_gebaeude
  SET
       building_storeys = Floor(building_height/3)
  WHERE building_storeys IS NULL AND building_height IS NOT NULL;

UPDATE be.mopube_gebaeude
  SET
       building_storeys = 1
    WHERE  building_storeys=0;
```

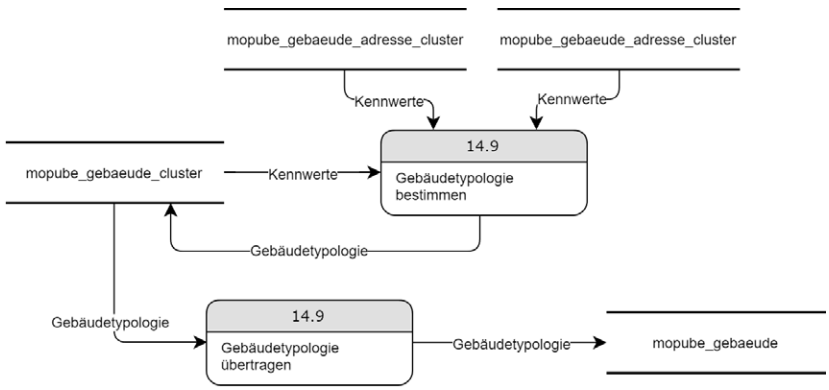
Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Aufbereitung der Daten, welche als Grundlage für die Zuordnung der Gebäudetypologie dienen.



Quelle: eigene Darstellung

Abbildung 56: Datenfluss vorbereitende Schritte zur Ableitung Gebäudetypologie

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Zuordnung der Gebäudetypologie.



Quelle: eigene Darstellung

Abbildung 57: Datenfluss Ableitung Gebäudetypologie

Die folgenden Befehle dienen der Aufbereitung und Bestimmung der Gebäudetypologie.

```
--Creat Table be.mopube_gebaeude_cluster
DROP TABLE IF EXISTS be.mopube_gebaeude_cluster;

CREATE TABLE be.mopube_gebaeude_cluster
(
  gid serial NOT NULL,
  count SMALLINT,
  count_entrances SMALLINT,
  count_interior_rings SMALLINT,
  count_interior_rings_union SMALLINT,
  area_interior numeric,
  area_filled numeric,
  area_interior_union numeric,
```

```
  area_filled_union numeric,
  area_concave numeric,
  area_convex numeric,
  area numeric,
  length numeric,
  buildingtypology smallint,
  geom geometry(MULTIPOLYGON,2056),
  CONSTRAINT mopube_gebaeude_cluster_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);

CREATE INDEX mopube_gebaeude_cluster_geom_idx
  ON be.mopube_gebaeude_cluster
  USING gist
  (geom);

INSERT INTO be.mopube_gebaeude_cluster (geom)
SELECT ST_SetSRID(ST_CollectionExtract(geom,3),2056) FROM
  (SELECT unnest(ST_ClusterIntersecting(ST_SetSRID(geom,2056))) AS geom
   FROM be.mopube_gebaeude) clusters
;

UPDATE be.mopube_gebaeude
  SET
    cluster_gid = c.gid
  FROM be.mopube_gebaeude_cluster AS c
    WHERE ST_Intersects(mopube_gebaeude.geom,ST_Buffer(c.geom,0.0));

UPDATE be.mopube_gebaeude_cluster
  SET
    count = ST_NumGeometries(geom),
    count_interior_rings = (SELECT SUM(ST_NumInteriorRings(geom))
                            FROM (SELECT (ST_Dump(geom)).geom) AS i),
    count_interior_rings_union = (SELECT SUM(ST_NumInteriorRings(geom))
                                  FROM (SELECT (ST_Dump(ST_Buffer(geom,0.0))).geom) AS i),
    area_filled = (SELECT ST_Area(ST_Collect(ST_MakePolygon(geom))) AS geom
                   FROM (SELECT  ST_ExteriorRing((ST_Dump(geom)).geom) AS geom ) AS i),
    area_filled_union =
                  (SELECT ST_Area(ST_Collect(ST_MakePolygon(geom))) AS geom
                   FROM (SELECT ST_ExteriorRing((ST_Dump(ST_Buffer(geom,0.0))).geom
                        ) AS geom ) AS i),
    area_concave = ST_Area(ST_ConcaveHull(geom,0.99)),
    area_convex = ST_Area(ST_ConvexHull(geom)),
    area = ST_Area(geom),
    length = ST_Perimeter(ST_Buffer(geom,0.0));

UPDATE be.mopube_gebaeude_cluster
  SET
    area_interior = area_filled-area,
    area_interior_union = area_filled_union-area;

--Create Table be.mopube_gebaeude_eingang
DROP TABLE IF EXISTS be.mopube_gebaeude_eingang;

CREATE TABLE be.mopube_gebaeude_eingang
(
  gebaeude_gid integer NOT NULL,
  eingang_gid integer NOT NULL,
  CONSTRAINT mopube_gebaeude_eingang_pkey PRIMARY KEY (gebaeude_gid,eingang_gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE be.mopube_gebaeude_eingang
  OWNER TO postgres;

CREATE INDEX be_mopube_gebaeude_eingang_eingang_gid_idx
       ON be.mopube_gebaeude_eingang
       USING btree (eingang_gid ASC NULLS LAST);
CREATE INDEX be_mopube_gebaeude_eingang_gebaeude_gid_idx
       ON be.mopube_gebaeude_eingang
       USING btree (gebaeude_gid ASC NULLS LAST);

INSERT INTO be.mopube_gebaeude_eingang (
            eingang_gid, gebaeude_gid)
```

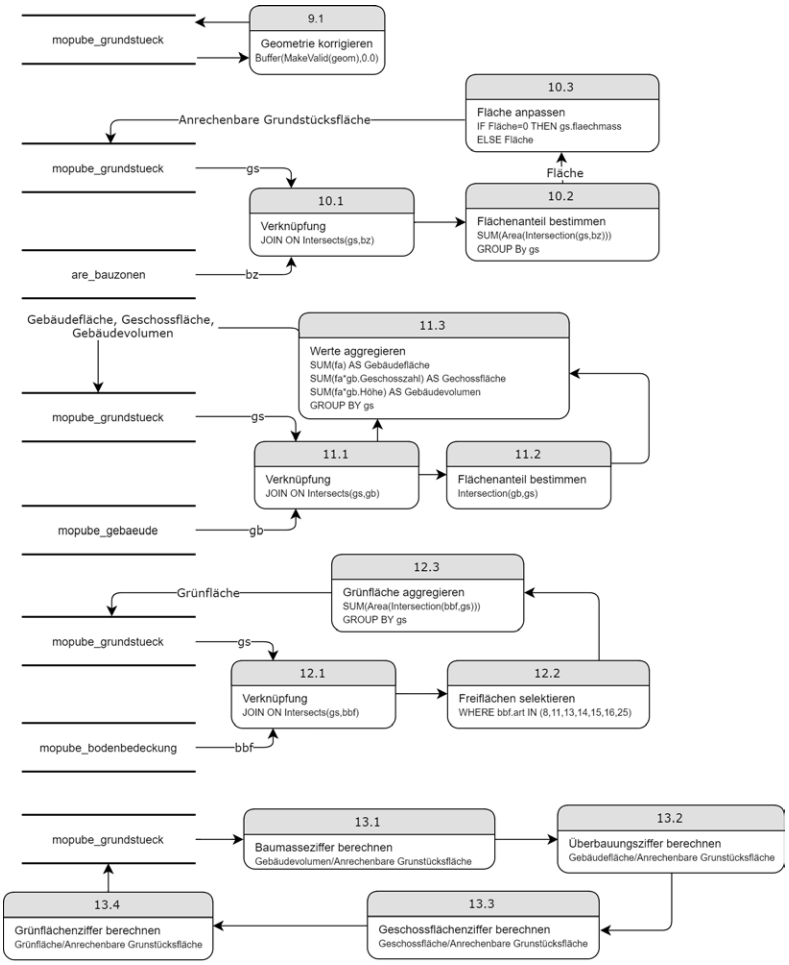
```
SELECT DISTINCT ON(eg.gid) eg.gid AS eingang_gid, g.gid AS gebaeude_gid
    FROM be.mopube_gaeinp AS eg, be.mopube_gebaeude AS g
    WHERE eg.gid <> g.gid AND ST_DWithin(eg.geom, g.geom, 20)
    ORDER BY eg.gid, ST_Distance(eg.geom,g.geom);

--Create Table be.mopube_gebaeude_eingang_cluster
DROP TABLE IF EXISTS be.mopube_gebaeude_eingang_cluster;

CREATE TABLE be.mopube_gebaeude_eingang_cluster
(
  gid serial NOT NULL,
  cluster_gid integer,
  count SMALLINT,
  area_concave numeric,
  area_convex numeric,
  length numeric,
  geom geometry(MULTIPOINT,2056),
  CONSTRAINT mopube_gebaeude_cluster_eingang_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE be.mopube_gebaeude_eingang_cluster
  OWNER TO postgres;

CREATE INDEX mopube_gebaeude_cluster_eingang_geom_idx
  ON be.mopube_gebaeude_eingang_cluster
  USING gist
  (geom);

INSERT INTO be.mopube_gebaeude_eingang_cluster (cluster_gid, geom)
SELECT cluster_gid, ST_Collect(geom) FROM
(SELECT DISTINCT ON(eg.gid) eg.gid AS eingang_gid, eg.geom, g.gid AS cluster_gid
    FROM be.mopube_gaeinp AS eg, be.mopube_gebaeude_cluster AS g
    WHERE eg.gid <> g.gid AND ST_DWithin(eg.geom, g.geom, 10)
    ORDER BY eg.gid, ST_Distance(eg.geom,g.geom)) AS nn
  WHERE cluster_gid IS NOT NULL
  GROUP BY cluster_gid;

INSERT INTO be.mopube_gebaeude_eingang_cluster (cluster_gid, geom)
SELECT gid AS cluster_gid, ST_Collect(geom) FROM
(
SELECT gid, ST_Centroid((ST_Dump(ST_CollectionExtract(geom,3))).geom) AS geom
    FROM be.mopube_gebaeude_cluster AS g
    WHERE NOT EXISTS (SELECT 1 FROM be.mopube_gebaeude_eingang_cluster
                      WHERE mopube_gebaeude_eingang_cluster.cluster_gid = g.gid)
) AS nn
  GROUP BY gid;


UPDATE be.mopube_gebaeude_eingang_cluster
  SET
    count = ST_NumGeometries(geom),
    area_concave = ST_Area(ST_ConcaveHull(ST_Buffer(geom,0.5),0.99)),
    area_convex = ST_Area(ST_ConvexHull(ST_Buffer(geom,0.5))),
    length = ST_Perimeter(ST_ConcaveHull(ST_Buffer(geom,0.5),0.99));

--Create Table be.mopube_gebaeude_adresse
DROP TABLE IF EXISTS be.mopube_gebaeude_adresse;

CREATE TABLE be.mopube_gebaeude_adresse
(
  gebaeude_gid integer NOT NULL,
  adresse_gid integer NOT NULL,
  CONSTRAINT mopube_gebaeude_adresse_pkey PRIMARY KEY (gebaeude_gid,adresse_gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE be.mopube_gebaeude_adresse
  OWNER TO postgres;

CREATE INDEX be_mopube_gebaeude_adresse_aderesse_gid_idx
  ON be.mopube_gebaeude_adresse
  USING btree (adresse_gid ASC NULLS LAST);
CREATE INDEX be_mopube_gebaeude_adresse_gebaeude_gid_idx
  ON be.mopube_gebaeude_adresse
```

```
  USING btree (gebaeude_gid ASC NULLS LAST);

INSERT INTO be.mopube_gebaeude_adresse (
            adresse_gid, gebaeude_gid)
SELECT DISTINCT ON(ad.gid) ad.gid AS eingang_gid, g.gid AS gebaeude_gid
    FROM be.gebadr AS ad, be.mopube_gebaeude AS g
    WHERE ad.gid <> g.gid AND ST_DWithin(ad.geom, g.geom, 1)
    ORDER BY ad.gid, ST_Distance(ad.geom,g.geom);

--Create Table: be.mopube_gebaeude_adresse_cluster
DROP TABLE IF EXISTS be.mopube_gebaeude_adresse_cluster;

CREATE TABLE be.mopube_gebaeude_adresse_cluster
(
  gid serial NOT NULL,
  cluster_gid integer,
  count SMALLINT,
  area_concave numeric,
  area_convex numeric,
  length numeric,
  geom geometry(MULTIPOINT,2056),
  CONSTRAINT mopube_gebaeude_cluster_adresse_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE be.mopube_gebaeude_adresse_cluster
  OWNER TO postgres;

CREATE INDEX mopube_gebaeude_cluster_adresse_geom_idx
  ON be.mopube_gebaeude_adresse_cluster
  USING gist
  (geom);

CREATE INDEX be_mopube_gebaeude_adresse_cluster_cluster_gid_idx
  ON be.mopube_gebaeude_adresse_cluster
  USING btree (cluster_gid ASC NULLS LAST);

INSERT INTO be.mopube_gebaeude_adresse_cluster (cluster_gid, geom)
SELECT cluster_gid, ST_Collect(geom) FROM
(SELECT DISTINCT ON(ad.gid) ad.gid AS adresse_gid, ad.geom, g.gid AS cluster_gid
    FROM be.gebadr AS ad, be.mopube_gebaeude_cluster AS g
    WHERE ad.gid <> g.gid AND ST_DWithin(ad.geom, g.geom, 10)
    ORDER BY ad.gid, ST_Distance(ad.geom,g.geom)) AS nn
  WHERE cluster_gid IS NOT NULL
  GROUP BY cluster_gid;

INSERT INTO be.mopube_gebaeude_adresse_cluster (cluster_gid, geom)
SELECT gid AS cluster_gid, ST_Collect(geom) FROM
(
SELECT gid, ST_Centroid((ST_Dump(ST_CollectionExtract(geom,3))).geom) AS geom
    FROM be.mopube_gebaeude_cluster AS g
    WHERE NOT EXISTS (SELECT 1 FROM be.mopube_gebaeude_adresse_cluster
                      WHERE mopube_gebaeude_adresse_cluster.cluster_gid = g.gid)
) AS nn
  GROUP BY gid;

UPDATE be.mopube_gebaeude_adresse_cluster
  SET
    count = ST_NumGeometries(geom),
    area_concave = ST_Area(ST_ConcaveHull(ST_Buffer(geom,0.5),0.99)),
    area_convex = ST_Area(ST_ConvexHull(ST_Buffer(geom,0.5))),
    length = ST_Perimeter(ST_ConcaveHull(ST_Buffer(geom,0.5),0.99));

UPDATE be.mopube_gebaeude_cluster AS c
  SET
    count_entrances = ac.count
  FROM be.mopube_gebaeude_adresse_cluster AS ac
  WHERE c.gid=ac.cluster_gid;

--Classifier v2, 6 Classes
WITH t AS (
        SELECT c.gid AS cluster_gid,
          ac.count AS ac_count,
          ac.area_concave AS ac_area_concave,
          ac.area_convex AS ac_area_convex,
          ac.length AS ac_length,
```

```
          ec.count AS ec_count,
          ec.area_concave AS ec_area_concave,
          ec.area_convex AS ec_area_convex,
          ec.length AS ec_length
        FROM be.mopube_gebaeude_cluster AS c
          LEFT JOIN be.mopube_gebaeude_adresse_cluster AS ac ON c.gid=ac.cluster_gid
          LEFT JOIN be.mopube_gebaeude_eingang_cluster AS ec ON c.gid=ec.cluster_gid
    )
    UPDATE be.mopube_gebaeude_cluster AS c
      SET
        buildingtypology =
            CASE
              WHEN count = 1 THEN
                CASE
                  WHEN (count_entrances > 1 OR count_interior_rings_union > 0 OR
area/area_concave < 0.85 OR area > 2500)
                    THEN 6
                    ELSE 1
                END
              WHEN count = 2 AND c.area_filled <= 250 THEN
                CASE
                  WHEN count_entrances = 2 AND c.area/c.area_concave > 0.8 THEN 2
                  ELSE 6
                END
              ELSE
                CASE
                  WHEN area/area_concave >= 0.9 AND area/area_convex >= 0.6 AND
count_interior_rings_union = 0 AND ac_area_concave/area_concave < 0.2 THEN 3
                  WHEN count_interior_rings_union > count_interior_rings AND
count_interior_rings_union =1 AND area_interior_union/area_filled_union > 0.20 THEN 4
                  WHEN area/area_concave < 0.9 AND count > 5 AND area/area_convex < 0.55 AND
count_interior_rings_union = 0 THEN 5
                  ELSE 6
                END
            END
      FROM t
      WHERE c.gid=t.cluster_gid;


UPDATE be.mopube_gebaeude AS g
  SET
    buildingtypology=gc.buildingtypology
  FROM be.mopube_gebaeude_cluster AS gc
  WHERE g.cluster_gid=gc.gid;
```

B.1.7. Datenaufbereitung Grundstücke

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Aufbereitung der grundstücksbezogenen Daten auf.



Quelle: eigene Darstellung

Abbildung 58: Datenfluss Aufbereitung grundstücksbezogene Daten

Die folgenden Befehle dienen der Aufbereitung der Grundstücksdaten.

```
ALTER TABLE be.mopube_grundstueck
  ADD COLUMN anrechenbare_gruendstuecksflaeche real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN building_area real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN building_volume real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN building_storey_area real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN gruenflaeche real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN baumasseziffer real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN ausnutzungsziffer real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN geschossflaechenziffer real;


ALTER TABLE be.mopube_grundstueck
```

```
  ADD COLUMN gruenflaechenziffer real;

ALTER TABLE be.mopube_grundstueck
  ADD COLUMN ueberbauungsziffer real;

UPDATE be.mopube_grundstueck
  SET
    geom = ST_Multi(ST_Buffer(ST_MakeValid(geom),0.0))
  WHERE NOT ST_IsValid(geom);

UPDATE be.mopube_grundstueck
  SET
    anrechenbare_gruendstuecksflaeche = t.anrechenbare_gruendstuecksflaeche
  FROM
    (SELECT p.gid
         ,SUM(ST_Area(ST_Intersection(bz.geom,p.geom))) AS anrechenbare_gruendstuecksflaeche
       FROM be.mopube_grundstueck AS p
         LEFT JOIN ch.are_bauzonen AS bz ON ST_Intersects(bz.geom,p.geom)
    GROUP BY p.gid
) AS t
WHERE t.gid=mopube_grundstueck.gid;

UPDATE be.mopube_grundstueck
  SET
    building_area = t.building_area,
    building_volume = t.building_volume,
    building_storey_area = t.building_storey_area
  FROM
    (SELECT p.gid
         ,SUM(ST_Area(ST_Intersection(g.geom,p.geom))*g.building_height) AS building_volume
         ,SUM(ST_Area(ST_Intersection(g.geom,p.geom))) AS building_area
         ,SUM(ST_Area(ST_Intersection(g.geom,p.geom))*building_storeys)                    AS
building_storey_area
       FROM be.mopube_grundstueck AS p
         LEFT JOIN be.mopube_gebaeude AS g ON ST_Intersects(g.geom,p.geom)
                                              --AND
ST_Within(ST_PointOnSurface(g.geom),p.geom)
    GROUP BY p.gid
) AS t
WHERE t.gid=mopube_grundstueck.gid;

UPDATE be.mopube_grundstueck
  SET
    gruenflaeche = COALESCE(t.gruenflaeche,0)
  FROM
(SELECT p.gid, SUM(ST_Area(ST_Intersection(bbf.geom,p.geom))) AS gruenflaeche
  FROM be.mopube_grundstueck AS p
  LEFT JOIN be.mopube_bodenbedeckung AS bbf ON ST_Intersects(bbf.geom,p.geom)
  AND bbf.art in (
  --0,--"Gebäude","bâtiment"
  --1,--"Strasse, Weg","route, chemin"
  --2,--"Trottoir","trottoir"
  --3,--"Verkehrsinsel","Îlot"
  --4,--"Bahn","chemin de fer"
  --5,--"Flugplatz","place aviation"
  --6,--"Wasserbecken","bassin"
  --7,--"übrige befestigte","autre revêtement dur"
  8,--"Acker, Wiese, Weide","champ, pré, pâturage"
  --9,--"Reben","vigne"
  --10,--"übrige Intensivkultur","autre culture intensive"
  11,--"Gartenanlage","jardin"
  --12,--"Hoch-, Flachmoor","tourbière"
  13,--"übrige humusierte","autre verte"
  14,--"stehendes Gewässer","eau stagnante"
  15,--"fliessendes Gewässer","cours d'eau"
  16,--"Schilfgürtel","roselière"
  --17,--"geschlossener Wald","forêt dense"
  --18,--"Wytweide dicht","pâturage boisé dense"
  --19,--"Wytweide offen","pâturage boisé ouvert"
  --20,--"übrige bestockte","autre boisée"
  --21,--"Fels","rocher"
  --22,--"Gletscher, Firn","glacier, névé"
  --23,--"Geröll, Sand","éboulis, sable"
  --24,--"Abbau, Deponie","gravière, décharge"
  25--"übrige vegetationslose","autre sans végétation"
    )
    GROUP BY p.gid
```

```
) AS t
WHERE t.gid=mopube_grundstueck.gid;

UPDATE be.mopube_grundstueck AS p
  SET
    baumasseziffer = building_volume/anrechenbare_gruendstuecksflaeche,
    geschossflaechenziffer= building_storey_area/anrechenbare_gruendstuecksflaeche,
    ueberbauungsziffer = building_area/anrechenbare_gruendstuecksflaeche,
    gruenflaechenziffer = gruenflaeche/anrechenbare_gruendstuecksflaeche
  WHERE anrechenbare_gruendstuecksflaeche > building_area
    ;

UPDATE be.mopube_grundstueck AS p
  SET
    gruenflaechenziffer = 1
  WHERE gruenflaechenziffer>1
    ;

UPDATE be.mopube_grundstueck AS p
  SET
    ueberbauungsziffer = 1
  WHERE ueberbauungsziffer>1
    ;
```

B.2. Erstellung Datenmodell parametrisches Modell

Der nachfolgende Code dient der Erstellung der Datenbanktabellen für das parametrische Modell. Für das automatische Update der Zeitstempel in den Tabellen wird eine aus einem Blog Post entlehnte Funktion verwendet (Wiles 2006).

```
--
--Function for updating timestamp column
--Basing on https://www.revsys.com/blog/2006/aug/04/automatically-updating-a-timestamp-column-in-postgresql/
--
CREATE OR REPLACE FUNCTION update_modified_column()
RETURNS TRIGGER AS $$
BEGIN
    NEW.modified = now();
    RETURN NEW;
END;
$$ language 'plpgsql';

--
-- Create Schema: dencity
--
--DROP SCHEMA IF EXISTS  dencity;
CREATE SCHEMA IF NOT EXISTS dencity
  AUTHORIZATION postgres;

--
--Drop Scenario tables
--
DROP TABLE IF EXISTS dencity.gebaeude;
DROP TABLE IF EXISTS dencity.parzelle;
DROP TABLE IF EXISTS dencity.abstandslinie;
DROP TABLE IF EXISTS dencity.szenario_parameter;
DROP TABLE IF EXISTS dencity.szenario;
DROP TABLE IF EXISTS dencity.projekt;

--
--Create Scenario tables
--

--
--Projekt table
--
CREATE TABLE dencity.projekt
(
  id serial NOT NULL,
  titel character varying(30),
  beschreibung character varying(512),
  modified timestamp DEFAULT CURRENT_TIMESTAMP NOT NULL,
  CONSTRAINT projekt_pkey PRIMARY KEY (id)
)
```

```
WITH (
  OIDS=FALSE
);
ALTER TABLE dencity.projekt
  OWNER TO postgres;

CREATE TRIGGER update_projekt_modtime BEFORE UPDATE ON dencity.projekt FOR EACH ROW EXECUTE
PROCEDURE update_modified_column();

--
--Base Szenario table
--
CREATE TABLE dencity.szenario
(
  id serial NOT NULL,
  titel character varying(50),
  beschreibung character varying(512),
  pid int NOT NULL REFERENCES dencity.projekt (id),
  modified timestamp DEFAULT CURRENT_TIMESTAMP NOT NULL,
  CONSTRAINT szenario_pkey PRIMARY KEY (id)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE dencity.szenario
  OWNER TO postgres;

CREATE TRIGGER update_szenario_modtime BEFORE UPDATE ON dencity.szenario FOR EACH ROW EXECUTE
PROCEDURE update_modified_column();

--
--Szenario Parameter table
--
CREATE TABLE dencity.szenario_parameter
(
  id serial NOT NULL,
  sid int NOT NULL REFERENCES dencity.szenario (id),
  zone_code character varying(100) NOT NULL,
  parameter_name character varying(100) NOT NULL,
  parameter_wert character varying(4096) NOT NULL,
  modified timestamp DEFAULT CURRENT_TIMESTAMP NOT NULL,
  CONSTRAINT szenario_parameter_unique UNIQUE (sid, zone_code, parameter_name),
  CONSTRAINT szenario_parameter_pkey PRIMARY KEY (id)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE dencity.szenario_parameter
  OWNER TO postgres;

CREATE TRIGGER update_szenario_parameter_modtime BEFORE UPDATE ON dencity.szenario_parameter
FOR EACH ROW EXECUTE PROCEDURE  update_modified_column();

--
--Parzelle tables
--
CREATE TABLE dencity.parzelle
(
  gid serial NOT NULL,
  pid int NOT NULL REFERENCES dencity.projekt (id),
  nummer character varying(15),
  zone_code character varying(100),
  anrechenbare_grundstuecksflaeche real NOT NULL,
  geom geometry(MultiPolygon,2056),
  CONSTRAINT parzelle_nummer_unique UNIQUE (pid, nummer),
  CONSTRAINT parzelle_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE dencity.parzelle
  OWNER TO postgres;

CREATE INDEX parzelle_geom_idx
  ON dencity.parzelle
  USING gist (geom);
```

```
SELECT Populate_Geometry_Columns('dencity.parzelle'::regclass);

--
--Abstandslinie table
--
CREATE TABLE dencity.abstandslinie
(
  gid serial NOT NULL,
  sid int NOT NULL REFERENCES dencity.szenario (id),
  gestaltend boolean DEFAULT false NOT NULL,
  geom geometry(Linestring,2056),
  CONSTRAINT abstandslinie_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE dencity.abstandslinie
  OWNER TO postgres;

CREATE INDEX abstandslinie_geom_idx
  ON dencity.abstandslinie
  USING gist (geom);

SELECT Populate_Geometry_Columns('dencity.abstandslinie'::regclass);



--
--Gebaeude tables
--
CREATE TABLE dencity.gebaeude
(
  gid serial NOT NULL,
  sid int NOT NULL REFERENCES dencity.szenario (id),
  hoehe real DEFAULT 0.0 NOT NULL,
  geschosszahl int DEFAULT 1 NOT NULL,
  geschossflaeche real NOT NULL,
  anteil_arbeitsnutzung real NOT NULL DEFAULT 0.0,
  nutzer_wohnen real NOT NULL,
  nutzer_arbeit real NOT NULL,
  geom geometry(MultiPolygon,2056),
  modified timestamp DEFAULT CURRENT_TIMESTAMP NOT NULL,
  CONSTRAINT gebaeude_pkey PRIMARY KEY (gid)
)
WITH (
  OIDS=FALSE
);
ALTER TABLE dencity.gebaeude
  OWNER TO postgres;

CREATE INDEX gebaeude_geom_idx
  ON dencity.gebaeude
  USING gist (geom);

CREATE TRIGGER update_gebaeude_modtime BEFORE UPDATE ON dencity.gebaeude FOR EACH ROW EXECUTE
PROCEDURE update_modified_column();

SELECT Populate_Geometry_Columns('dencity.gebaeude'::regclass);
```

B.3. Datenaufbereitung Szenarien

B.3.1. Projektgebiet Langenthal

```
--
--Prepare Projekt Langenthal
--

--Create project entry
INSERT INTO dencity.projekt(id, titel)
    VALUES (4, 'Langenthal');

--Create scenario entries
INSERT INTO dencity.szenario(id, titel, pid)
    VALUES (11, 'Szenario W3', 4);
INSERT INTO dencity.szenario(id, titel, pid)
    VALUES (12, 'Szenario W4', 4);
```

```
--Insert parcels for Langenthal
DELETE FROM dencity.parzelle WHERE pid=4;

INSERT INTO dencity.parzelle(pid, nummer, zone_code, anrechenbare_grundstuecksflaeche, geom)
SELECT pid
  ,nummer
  ,zone_code
  ,SUM(anrechenbare_grundstuecksflaeche) AS anrechenbare_grundstuecksflaeche
  ,MIN(geom) AS geom
FROM
(
  SELECT
    4 AS pid
    ,nummer AS nummer
    ,(SELECT j.zone_lo
      FROM (SELECT i.zone_lo
                  ,ST_Area(ST_Intersection(mopube_grundstueck.geom,ST_Force2D(i.geom))) AS
area
            FROM be.uzp_bauzonen AS i WHERE
ST_Intersects(mopube_grundstueck.geom,ST_Force2D(i.geom))) AS j
            ORDER BY j.area DESC LIMIT 1)  AS zone_code
    ,ST_Area(ST_Intersection(mopube_grundstueck.geom,ST_Force2D(uzp_bauzonen.geom))) AS
anrechenbare_grundstuecksflaeche
    ,mopube_grundstueck.geom
  FROM be.mopube_grundstueck, be.uzp_bauzonen
  WHERE bfsnr = 329
    AND ST_Intersects(mopube_grundstueck.geom,uzp_bauzonen.geom)
    AND bfs = 329
    AND ST_Within(mopube_grundstueck.geom,ST_SetSRID(ST_GeomFromText('Polygon
((2625763.06347824446856976 1228953.55572490207850933,
    2625855.84641563845798373 1228973.13377591199241579, 2625959.69520795112475753
1229008.88499949499964714,
    2626030.34643550775945187 1228744.58131086290813982, 2626211.23060244601219893
1228671.80203428328968585,
    2626165.69035335816442966 1228581.14714591205120087, 2626127.81108075240626931
1228401.96541819232515991,
    2626023.53667863551527262 1228396.43249073321931064, 2625977.14520993828773499
1228380.68492796458303928,
    2625864.7842215346172452 1228373.44956128695048392, 2625864.7842215346172452
1228373.44956128695048392,
    2625763.06347824446856976 1228953.55572490207850933))'),2056))
    AND nummer NOT IN
('1707','1713','4987','2333','197.04','197.03','2995','2055','4981','4983','1737','4982') --
Streets
) AS t
WHERE anrechenbare_grundstuecksflaeche>1
GROUP BY pid, nummer, zone_code
ORDER BY 2;

--Remap Bauzonen to Modellzonen
UPDATE dencity.parzelle
SET
  zone_code=
    CASE
      WHEN zone_code = 'Wohnzone W2/ B' THEN 'Z1'
      WHEN zone_code = 'Wohnzone W2/ C' THEN 'Z2'
      WHEN zone_code = 'Mischzone MZ 2' THEN 'Z3'
      WHEN zone_code = 'UeO Nr. 41 "Areal Anliker"' THEN 'Z4'
      WHEN zone_code = 'UeO Nr. 41 "Areal Anliker" Baubereich A'THEN 'Z4'
      WHEN zone_code = 'Zone für öffentliche Nutzung ZöN Art. 77 BauG' THEN 'Z5'
      ELSE 'Z5'
    END
WHERE pid = 4;

--Correct geometry of a specific parcel (removing overlapping slivers)
UPDATE dencity.parzelle
  SET geom = ST_Multi((SELECT geom
                       FROM (SELECT (ST_Dump(geom)).geom AS geom
                             FROM dencity.parzelle WHERE pid=4 AND nummer ='2166') AS t
                       ORDER BY ST_Area(geom) DESC LIMIT 1))
WHERE pid=4 AND nummer ='2166';
```

B.3.2. Projektgebiet Uettligen

```
--
```

```
--Prepare project Wohlen: Uettligen
--

--Create project entry
INSERT INTO dencity.projekt(id, titel)
    VALUES (3, 'Wohlen: Uettligen');

--Create scenario entries
INSERT INTO dencity.szenario(id, titel, pid)
    VALUES (13, 'Szenario W3', 3);
INSERT INTO dencity.szenario(id, titel, pid)
    VALUES (14, 'Szenario W4', 3);

--Insert Parcels for Uettligen
DELETE FROM dencity.parzelle WHERE pid=3;

INSERT INTO dencity.parzelle(pid, nummer, zone_code, anrechenbare_grundstuecksflaeche, geom)
SELECT pid, nummer, zone_code, SUM(anrechenbare_grundstuecksflaeche) AS
anrechenbare_grundstuecksflaeche, MIN(geom) AS geom
FROM
(
  SELECT
    3 AS pid
    ,nummer AS nummer
    ,(SELECT j.zone_lo
      FROM (SELECT i.zone_lo
                  ,ST_Area(ST_Intersection(mopube_grundstueck.geom,ST_Force2D(i.geom))) AS
area
            FROM be.uzp_bauzonen AS i
            WHERE ST_Intersects(mopube_grundstueck.geom,ST_Force2D(i.geom))
                  AND i.zone_lo NOT IN ('Landwirtschaftszone / ungezontes Gebiet')) AS j
            ORDER BY j.area DESC LIMIT 1)  AS zone_code
    ,ST_Area(ST_Intersection(mopube_grundstueck.geom,ST_Force2D(uzp_bauzonen.geom))) AS
anrechenbare_grundstuecksflaeche
    ,mopube_grundstueck.geom
  FROM be.mopube_grundstueck, be.uzp_bauzonen
  WHERE bfsnr = 360
    AND ST_Intersects(mopube_grundstueck.geom,uzp_bauzonen.geom)
    AND bfs = 360
    AND zone_lo NOT IN ('Landwirtschaftszone / ungezontes Gebiet')
    AND ST_Within(mopube_grundstueck.geom,ST_SetSRID(ST_GeomFromText('Polygon
((2594707.0389655982144177 1204297.62532898178324103,
2595275.660759671125561 1203316.75273420615121722, 2595881.24297035858035088
1203410.57533022807911038,
2596688.68591794185340405 1203910.9625090123154223, 2596103.00547004723921418
1204871.93334099510684609,
2596103.00547004723921418 1204871.93334099510684609, 2595568.50098361866548657
1204749.67965526948682964,
2594707.0389655982144177 1204297.62532898178324103))'),2056))
    AND nummer NOT IN ('6199','4251','4907','5825') --Streets
) AS t
WHERE anrechenbare_grundstuecksflaeche>1
GROUP BY pid, nummer, zone_code
ORDER BY 2;

--Remap Bauzonen to Modellzonen
UPDATE dencity.parzelle
SET
  zone_code=
    CASE
      WHEN zone_code = 'Wohnzone 1-geschossig' THEN 'Z1'
      WHEN zone_code = 'Dorfzone, 2 geschossig' THEN 'Z2'
      WHEN zone_code = 'Wohnzone 2-geschossig' THEN 'Z2'
      WHEN zone_code = 'Wohnzone 3-geschossig' THEN 'Z3'
      WHEN zone_code = 'Mischzone' THEN 'Z4'
      WHEN zone_code = 'Mischzone Landwirtschaft' THEN 'Z4'
      WHEN zone_code = 'Zone für öffentliche Nutzung' THEN 'Z5'
      ELSE 'Z5'
    END
WHERE pid = 3;
```

B.4. Code Rhino Grasshopper

B.4.1. Konversion von GeoJSON in Rhino interne Geometrien (GeoJson2Rhino.py)

Der folgende Code ist eine modifizierte und erweiterte Version des Scripts GeoJson2Rhino.py aus Golder (2010). Dieses wird bei der Konversion der GeoJSON Geometrien aus der PostGIS Datenbank in Rhino interne Geometrietypen verwendet.

```
"""
Allows for the translation of GeoJSON data to Rhino objects

GeoJSON _does_ support 3d, so this can take 3d coordinates for 3d GeoJSONs

The GeoJSON Format Specification can be found here:
    http://geojson.org/geojson-spec.html

The RhinoCommon SDK (where all the Rhino.Geometry objects are documented) is
here:
    http://www.rhino3d.com/5/rhinocommon/

I have decided to extend the GeoJSON specification by adding support for one
more type of geometry that would be really useful in Rhino (and elsewhere),
the Mesh. Here is an example of a json Mesh:

    {"type": "Feature",
     "geometry": {
                  "type": "Mesh",
                  "coordinates": [
                                  [3.43, 54.234, 2343.23],
                                  [...],
                                  [...],
                                  ...,
                                  ]
                  "faces": [
                            [0,3,2],
                            [5,32,1],
                            ...,
                            ]
                  }
      "properties": {"prop0": "value0"}
      }


Example of Use:
    >>> import GeoJson2Rhino as geoj
    >>> myGeoJson = '''
{ "type": "FeatureCollection",
  "features": [
    { "type": "Feature",
      "geometry": {"type": "Point", "coordinates": [102.0, 0.5]},
      "properties": {"prop0": "value0"}
      },
    { "type": "Feature",
      "geometry": {
        "type": "LineString",
        "coordinates": [
          [102.0, 0.0], [103.0, 1.0], [104.0, 0.0], [105.0, 1.0]
          ]
        },
      "properties": {
        "prop0": "value0",
        "prop1": 0.0
        }
      },
    { "type": "Feature",
       "geometry": {
         "type": "Polygon",
         "coordinates": [
           [ [100.0, 0.0], [101.0, 0.0], [101.0, 1.0],
             [100.0, 1.0], [100.0, 0.0] ]
           ]
       },
       "properties": {
         "prop0": "value0",
         "prop1": {"this": "that"}
         }
```

```
      }
    ]
  }'''
  >>> guidList = geoj.load(myGeoJson) #stores guids of new rhino objects

"""

# Import standard library modules
import json

# Import Rhino modules
import Rhino
from Rhino.Geometry import *
from scriptcontext import doc

# import .NET libraries
import System


def addRhinoLayer(layerName, layerColor=System.Drawing.Color.Black):
    """Creates a Layer in Rhino using a name and optional color. Returns the
    index of the layer requested. If the layer
    already exists, the color is updated and no new layer is created."""
    docLyrs = doc.Layers
    layerIndex = docLyrs.Find(layerName, True)
    if layerIndex == -1:
        layerIndex = docLyrs.Add(layerName,layerColor)
    else: # it exists
        layer = docLyrs[layerIndex] # so get it
        if layer.Color != layerColor: # if it has a different color
            layer.Color = layerColor # reset the color
    return layerIndex

def PointToRhinoPoint(coordinates, objectAttributes = None):
    if len(coordinates) > 2:
        z = coordinates[2]
    else:
        z = 0.0
    x, y = coordinates[0], coordinates[1]
    pnt =  Point3d(x, y, z)
    #if objectAttributes:
    #    setUserKeys(objectAttributes,pnt)
    return pnt

def MultiPointToRhinoPoint(coordinates, objectAttributes = None):
    rhPointList = []
    for point in coordinates:
        rhPointList.append(PointToRhinoPoint(point))
    return rhPointList

def MeshToRhinoMesh(coordinates, faces, objectAttributes = None):
    rhMesh = Mesh()
    for point in coordinates:
        rhPoint = PointToRhinoPoint(point)
        rhMesh.Vertices.Add(rhPoint)
    for face in faces:
        i, j, k = tuple(face)
        mFace = MeshFace(i, j, k)
        rhMesh.Faces.AddFace(mFace)
    rhMesh.Normals.ComputeNormals()
    rhMesh.Compact()
    return rhMesh

def LineStringToRhinoCurve(coordinates, objectAttributes = None):
    rhPoints = MultiPointToRhinoPoint(coordinates)
    crv = Curve.CreateControlPointCurve(rhPoints, 1)
    if objectAttributes:
        setUserKeys(objectAttributes,crv)
    return crv

def MultiLineStringToRhinoCurve(coordinates, objectAttributes = None):
    rhCurveList = []
    for lineString in coordinates:
        rhCurveList.append(LineStringToRhinoCurve(lineString,objectAttributes))
    return rhCurveList

def PolygonToRhinoCurve(coordinates, objectAttributes = None):
```

```
    # each ring is a separate list of coordinates
    ringList = []
    for ring in coordinates:
        ringList.append(LineStringToRhinoCurve(ring,objectAttributes))
    return ringList

def MultiPolygonToRhinoCurve(coordinates, objectAttributes = None):
    polygonList = []
    for polygon in coordinates:
        polygonList.append(PolygonToRhinoCurve(polygon, objectAttributes))
    return polygonList

def GeometryCollectionToParser(geometries, objectAttributes = None):
    #pass # I need to figure this one out still
    rhFeatures = []
    for geom in geometries:
        coordinates = geom['coordinates']
        type = geom['type']
        rhFeatures.append(geoJsonGeometryMap[type][0](coordinates))
    return rhFeatures

def addPoint(rhPoint, objAtt):
    return doc.Objects.AddPoint(rhPoint, objAtt)

def addPoints(rhPoints, objAtt):
    guidList = []
    for rhPoint in rhPoints:
        guidList.append(doc.Objects.AddPoint(rhPoint, objAtt))
    return guidList

def addCurve(rhCurve, objAtt):
    return doc.Objects.AddCurve(rhCurve, objAtt)

def addCurves(rhCurves, objAtt):
    guidList = []
    for curve in rhCurves:
        guidList.append(addCurve(curve, objAtt))
    return guidList

def addPolygon(ringList, objAtt):
    # for now this just makes curves
    # but maybe it should make TrimmedSrfs
    # or should group the rings
    return addCurves(ringList, objAtt)

def addPolygons(polygonList, objAtt):
    guidList = []
    for polygon in polygonList:
        # !! Extending the guid list !!!
        guidList.extend(addPolygon(polygon, objAtt))
    return guidList

def addMesh(rhMesh, objAtt):
    return doc.Objects.AddMesh(rhMesh, objAtt)

geoJsonGeometryMap = {
        'Point':(PointToRhinoPoint, addPoint),
        'MultiPoint':(MultiPointToRhinoPoint, addPoints),
        'LineString':(LineStringToRhinoCurve, addCurve),
        'MultiLineString':(MultiLineStringToRhinoCurve, addCurves),
        'Polygon':(PolygonToRhinoCurve, addPolygon),
        'MultiPolygon':(MultiPolygonToRhinoCurve, addPolygons),
        'Mesh':(MeshToRhinoMesh, addMesh),
        'GeometryCollection':(GeometryCollectionToParser),
        }

def setUserKeys(properties, objAttributes):
    for key in properties:
        objAttributes.SetUserString(key, str(properties[key]))
    return objAttributes

def jsonToRhinoCommon(jsonFeature):
        # deal with the geometry
        geom = jsonFeature['geometry']
        geomType = geom['type'] # this will return a mappable string
        if jsonFeature['properties']:
            prop = jsonFeature['properties']
```

```
        if geomType == 'GeometryCollection':
            # if this is a GeometryCollection, pass the geometries
            geometries = geom['geometries']
            func = geoJsonGeometryMap[geomType]
            rhFeature = geoJsonGeometryMap[geomType](geometries)
        else:
            coordinates = geom['coordinates']
            # if this is a mesh, pass the faces
            if geomType == 'Mesh':
                faces = geom['faces']
                rhFeature = geoJsonGeometryMap[geomType][0](coordinates, faces)
            # translate the coordinates to Rhino.Geometry objects
            else:
                rhFeature                                                                    =
geoJsonGeometryMap[geomType][0](coordinates,jsonFeature['properties'])
        return rhFeature

def addJsonFeature(jsonFeature, objAttributes):
        # deal with the properties
        if jsonFeature['properties']:
            objAttributes = setUserKeys(jsonFeature['properties'], objAttributes)
        geomType = jsonFeature['geometry']['type']
        rhFeature = jsonToRhinoCommon(jsonFeature)
        # return the GUID(s) for the feature
        return geoJsonGeometryMap[geomType][1](rhFeature, objAttributes)

def processGeoJson(parsedGeoJson,
         destinationLayer=None,
         destinationLayerColor=System.Drawing.Color.Black):
    # get the features
    jsonFeatures = parsedGeoJson['features']
    guidResults = []
    # set up object attributes
    for jsonFeature in jsonFeatures: # for each feature
        att = Rhino.DocObjects.ObjectAttributes()
        # setup layer if requested
        if destinationLayer != None:
            att.LayerIndex = addRhinoLayer(destinationLayer,
                                           destinationLayerColor)
        guidResults.append(addJsonFeature(jsonFeature, att))
    # return all the guids
    return guidResults

def load(rawJsonData,
         destinationLayer=None,
         destinationLayerColor=System.Drawing.Color.Black):
    # if the data already appears to be a dict literal ...
    if type(rawJsonData) == dict:
        jsonData = rawJsonData
    else: # otherwise, just try to load it
        jsonData = json.loads(rawJsonData)
    # if this is just a GeoJSON ...
    if jsonData["type"] == "FeatureCollection":
        # process the GeoJSON, pass the layer and color in
        return processGeoJson(jsonData, destinationLayer,
                              destinationLayerColor)
    # or if this is a set of layers from PostSites ...
    elif jsonData["type"] == "LayerCollection":
        # make a list for all the guids
        allResults = []
        layersList = jsonData['layers']
        for layer in layersList: # for each layer
            name = layer['name'] # get the name
            if 'color' in layer: # get the color if it exists
                color = layer['color']
            else:
                color = destinationLayerColor # or just make it black
            geoJson = layer['contents'] # get the GeoJSON for this layer
            # make it
            layerResults = processGeoJson( geoJson, name, color )
            allResults.append(layerResults)
        return allResults
    else:
        return "This doesn't look like correctly formatted GeoJSON data.\nI'm not sure what to
do with it, sorry."

def convert(rawJsonData):
```

```
        # if the data already appears to be a dict literal ...
    if type(rawJsonData) == dict:
        jsonData = rawJsonData
    else: # otherwise, just try to load it
        jsonData = json.loads(rawJsonData)
    # if this is just a GeoJSON ...

    return jsonToRhinoCommon(jsonData)
```

B.4.2. Generierung SQL Abfrage für Parzellen (Py Parcel Query)

Der folgende Code parametrisiert mit Hilfe von Python String Operationen das SQL SELECT Statement für die Abfrage der Parzellen, der zugehörigen Abstandslinien sowie der daraus resultierenden Abstandszonen aus der PostGIS Datenbank.

```
query = """
SELECT  '{ "type": "Feature", "geometry": ' || ST_AsGeoJSON(CASE WHEN geomLineNormal IS NULL
THEN geomparzelle ELSE ST_Union(geomparzelle,geomLineNormal) END) || ',
      "properties": {"gid": "'||parzelle.gid||'", "nummer": "'||nummer||'","zone_code":
"'||zone_code||'","anrechenbare_grundstuecksflaeche": "'||anrechenbare_grundstuecksflaeche||'"
}
      }' AS geom,
      COALESCE('{ "type": "Feature", "geometry": ' || ST_AsGeoJSON(geomLimitZone) || ',
      "properties": {"gid": "'||parzelle.gid||'", "nummer": "'||nummer||'","zone_code":
"'||zone_code||'"}
      }','') AS  geomLimitZone,
      COALESCE('{ "type": "Feature", "geometry": ' || ST_AsGeoJSON(geomLinegestaltend) || ',
      "properties": {"gid": "'||parzelle.gid||'", "nummer": "'||nummer||'","zone_code":
"'||zone_code||'","gestaltend": "true"}
      }','') AS  geomLineConstitutive,
      COALESCE('{ "type": "Feature", "geometry": ' || ST_AsGeoJSON(geomLineNormal) || ',
      "properties": {"gid": "'||parzelle.gid||'", "nummer": "'||nummer||'","zone_code":
"'||zone_code||'","gestaltend": "false"}
      }','') AS  geomLineNormal,
      parzelle.gid, parzelle.pid, parzelle.nummer, parzelle.zone_code,
parzelle.anrechenbare_grundstuecksflaeche
FROM
(SELECT gid, pid, nummer, zone_code, anrechenbare_grundstuecksflaeche, ST_Union(geomSp) AS
geomparzelle
FROM
(SELECT parzelle.gid, parzelle.pid, parzelle.nummer, parzelle.zone_code,
anrechenbare_grundstuecksflaeche,
        CASE WHEN abstandslinie.gid IS NOT NULL THEN
ST_CollectionExtract(ST_Union(parzelle.geom, abstandslinie.geom),3) ELSE parzelle.geom END AS
geomSp
  FROM dencity.parzelle
  LEFT JOIN dencity.abstandslinie ON abstandslinie.sid="""+str(scenarioId)+""" AND
ST_Intersects(parzelle.geom, abstandslinie.geom)
WHERE parzelle.pid="""+str(projectId)+""") AS w
GROUP BY gid, pid, nummer, zone_code, anrechenbare_grundstuecksflaeche
) AS parzelle
LEFT JOIN
(SELECT *
FROM
(SELECT ST_Collect(CASE WHEN ST_Intersects(geomSp,ST_OffsetCurve(geomL,-0.0001, 'quad_segs=4
join=round')) THEN geomSp ELSE NULL END) AS  geomLimitZone,
      gid, COUNT(*) AS SplitCount
FROM
(SELECT parzelle.gid, (ST_Dump(ST_Split(parzelle.geom, abstandslinie.geom))).geom AS geomSp,
parzelle.geom AS geomP, abstandslinie.geom AS geomL
  FROM dencity.parzelle, dencity.abstandslinie
WHERE abstandslinie.sid="""+str(scenarioId)+"""
AND parzelle.pid="""+str(projectId)+"""
AND ST_Intersects(parzelle.geom,abstandslinie.geom)) AS t
GROUP BY gid
) AS u
WHERE SplitCount>1) AS zones ON zones.gid=parzelle.gid
LEFT JOIN
(SELECT gid, ST_Collect(CASE WHEN gestaltend THEN geomInt ELSE NULL END) AS
geomLinegestaltend, ST_Collect(CASE WHEN NOT gestaltend THEN geomInt ELSE NULL END) AS
geomLineNormal
FROM
(SELECT parzelle.gid, parzelle.pid, parzelle.nummer, parzelle.zone_code,
```

```
(ST_Dump(ST_LineMerge(ST_CollectionExtract(ST_Intersection(abstandslinie.geom,
parzelle.geom),2)))).geom AS geomInt,

abstandslinie.gestaltend
  FROM dencity.parzelle, dencity.abstandslinie
WHERE abstandslinie.sid="""+str(scenarioId)+"""
AND parzelle.pid="""+str(projectId)+"""
AND ST_Intersects(parzelle.geom, abstandslinie.geom)
AND NOT ST_IsEmpty((ST_CollectionExtract(ST_Intersection(abstandslinie.geom,
parzelle.geom),2)))) AS v
GROUP BY gid
) AS lines ON lines.gid=parzelle.gid
;
""".format(projectId,scenarioId)
```

B.4.3. Konversion der Parzellen in Rhino Geometrien (Py GeoJson Parcels to Rhino)

Der nachfolgende Code dient der Konversion der von Slingshot ausgegebenen Daten zu den Parzellen, die aus der PostGIS Datenbank abgefragt wurden. Die GeoJSON Geometrien werden in Rhino interne Geometrien konvertiert und in separate DataTrees ausgegeben. Zusätzlich werden einige Attribute ebenfalls in DataTrees ausgegeben. Dieser Code wird in ähnlicher Form an mehreren Stellen eingesetzt, an denen Ergebnisse einer Datenbankabfrage konvertiert werden müssen. Der Unterschied liegt vor allem in den jeweils konvertierten und ausgegebenen Spalten.

```
import sys
modulepath = rhpsPath
#Add folder to sys.path if it is not yet contained
if modulepath not in sys.path:
    sys.path.append(modulepath)

#import and use custom module
import GeoJson2Rhino as geoj
import Rhino
import rhinoscriptsyntax  as rs
import scriptcontext as sc
import Grasshopper as gh
import random
import math
import json
import Rhino.Geometry as rg

#Functionfor generating a polygon surface from polylines.
def generate_surface_from_polyline(polyline):
    """Method for generating and polygon surface from polylines."""
    surface=(rg.Brep.CreatePlanarBreps(polyline)[0])#.Faces[0]

    return surface

#Function for collecting the values of a specified column,
#converting and outputting them as a separate tree. Requires a type
#specification for the tree content (i.e. the type of the collected values).
def itemsToTree (attributeName,type):
    #Create a new tree of the specified type
    res = gh.DataTree[type]()
    #Determine the column index based on the attribute name
    itemIndex = headings.index(attributeName)

    #Iterate all element of the column to be converted
    for i in range(len(data.Branches[itemIndex])):
        itemObjects = []
        item = data.Branches[itemIndex][i]
        itemObjects.append(item)

        #Create output path
        path = gh.Kernel.Data.GH_Path(i)

        if itemObjects:
            res.AddRange(itemObjects,path)
        else:
            #Add None value to ensure that the path exists
            res.Add(None,path)

    return res
```

```python
#Function for flattening a multi-level list to a single level
def flattenList(lst):
    fl = []
    for l in lst:
        if isinstance(l, list):
            fl.extend(flattenList(l))
        else:
            fl.append(l)
    return fl

#Function for collecting the GeoJSON values of a specified column,
#converting them to Rhino internal geometries and outputting them as
#a separate tree. Requires a type specification for the tree content
#(i.e. the type of the converted geometry) and optinally a conversion function.
def geometryToTree (attributeName,type,conversion = None):
    #Create a new tree of the specified type
    res = gh.DataTree[type]()
    #Determine the column index based on the attribute name
    geomIndex = headings.index(attributeName)

    #Iterate all element of the column to be converted
    for i in range(len(data.Branches[geomIndex])):
        geomObjects = []
        geom = data.Branches[geomIndex][i]
        if not geom=='':
            rhinoObj = geoj.convert(geom)

            #Apply conversion if a conversion function was specified
            if conversion:
                rhinoObj = conversion(rhinoObj)

            if isinstance(rhinoObj, list):
                #Extend instead of append to avoid sublists
                geomObjects.extend(rhinoObj)
            else:
                geomObjects.append(rhinoObj)

        #Create output path
        path = gh.Kernel.Data.GH_Path(i)

        if geomObjects:
            res.AddRange(geomObjects,path)
        else:
            #Add None value to ensure that the path exists
            res.Add(None,path)

    return res

#Column names of geometry attributes to be tranformed
_geomAttr = "geom"
_geomLimitZoneAttr = "geomlimitzone"
_geomLineConstitutiveAttr = "geomlineconstitutive"
_geomLineNormalAttr = "geomlinenormal"

#Conversion functions
lineCurveToPolylineCurve = lambda x: rg.PolylineCurve([x.PointAtStart,x.PointAtStart])
polylineCurveToBrep = lambda x: generate_surface_from_polyline(flattenList(x))

#Transformation of geometry columns
parcels = geometryToTree(_geomAttr,rg.Brep,polylineCurveToBrep)
limitZones = geometryToTree(_geomLimitZoneAttr,rg.Curve,flattenList)
limitLines = geometryToTree(_geomLineConstitutiveAttr,rg.Curve,flattenList)
constitutiveLines = geometryToTree(_geomLineNormalAttr,rg.Curve,flattenList)

#Column names of additional attributes to be tranformed
_zoneCodeAttr = "zone_code"
_parcelNumberAttr = "nummer"
_parcelUsableArea = "anrechenbare_grundstuecksflaeche"
_gidAttr = "gid"

#Transformation of additional columns
zoneCodes = itemsToTree(_zoneCodeAttr,str)
usableArea = itemsToTree(_parcelUsableArea,str)
parcelNumbers = itemsToTree(_parcelNumberAttr,str)
gid = itemsToTree(_gidAttr,str)
```

B.4.4. Verteilung von Parzellen auf Pfade nach Gruppierung (Py Dispatch Groups)

Der folgende Code dient der Verteilung der Parzellen auf verschiedene Branches in einem DataTrees basierend auf den konfigurierten Gruppierungen. Branch 0 umfasst die nicht gruppierten, Branch 1 die als Reihenhäuser gruppierten und in Branch 2 die als Blockrand gruppierten Parzellen.

```
import Rhino
import Grasshopper as gh
import json

#Output lists
parcelGroups = []
groupedParcelNumbers = []
parcelRoofTypes = {}

#Iterate all parameters to parse groups and roof types
for i in range(len(parameters)):
    paramsJson = parameters[i]
    params = json.loads(paramsJson)
    if params:
        if "gruppen" in params:
            groups = params["gruppen"]
            for grp in groups:
                parcelGroups.append(grp)
                groupedParcelNumbers.extend(grp["parzellen_nummern"])
        if "dachtypen" in params:
            roofs = params["dachtypen"]
            for rf in roofs:
                if rf["typ"]=="steildach":
                    roofType = 1
                if rf["typ"]=="flachdach":
                    roofType = 0

                for pn in rf["parzellen_nummern"]:
                    parcelRoofTypes[pn] = roofType

#Create output trees
res = gh.DataTree[object]()
res_rt = gh.DataTree[object]()

#Iterate parcel numbers for normal output
parcelIndexes = {}
j = 0
for i in range(parcelNumbers.BranchCount):
    branch = parcelNumbers.Branches[i]
    parcelNumber = branch[0]
    parcelIndexes[parcelNumber] = i
    if not (parcelNumber in groupedParcelNumbers):
        #Parcel is not grouped -> output on path 0
        path = gh.Kernel.Data.GH_Path(0,j)
        res.Add(i,path)
        print path
        j+=1
        #Output rooftype to corresponding path if a type is configured
        if parcelNumber in parcelRoofTypes:
            res_rt.Add(parcelRoofTypes[parcelNumber],path)
        else:
            res_rt.Add(None,path)

#Process grouped parcel numbers if they exist
if parcelGroups:
    print parcelGroups
    #Iterate parcel groups
    for i in range(len(parcelGroups)):
        group = parcelGroups[i]
        groupParcelNumbers = group["parzellen_nummern"]
        groupIndexes = []
        for k in range(len(groupParcelNumbers)):
            parcelNumber = groupParcelNumbers[k]
            if parcelIndexes.ContainsKey(parcelNumber):
                groupIndexes.append(parcelIndexes[parcelNumber])
        if groupIndexes:
            typeIndex = -1
            #Determine output path based on group type
```

```
            if group["typ"] == "reihe":
                typeIndex = 1
            elif group["typ"] == "block":
                typeIndex = 2
            path = gh.Kernel.Data.GH_Path(typeIndex,i)
            res.AddRange(groupIndexes,path)
            #Output rooftype to corresponding path if a type is configured
            if parcelNumber in parcelRoofTypes:
                res_rt.Add(parcelRoofTypes[parcelNumber],path)
            else:
                res_rt.Add(None,path)
else:
    #No groups -> ensure empty paths 1 and 2
    path = gh.Kernel.Data.GH_Path(1,0)
    res.EnsurePath(path)
    res_rt.EnsurePath(path)
    path = gh.Kernel.Data.GH_Path(2,0)
    res.EnsurePath(path)
    res_rt.EnsurePath(path)

paths = res
roofTypes = res_rt
```

B.4.5. Generierung SQL Abfrage für das Zurücksetzen der Parzellen (Py Parcel Geom to Query)

Der nachfolgende Code generiert ein SQL SELECT Statement, welches das Zurücksetzen der Parzellenflächen auf der PostGIS Datenbank umsetzt.

```
import Rhino.Geometry as rg
import json

jsonLines = []
jsonZones = []
limitZoneCount = 0

#Loop over all input branches
for b in range(parcel.BranchCount):
    #Get specific sub-branches
    parcelBranch = parcel.Branches[b]
    lineMajorBranch = lineMajor.Branches[b]
    setbackMajorBranch = setbackMajor.Branches[b]
    lineMinorBranch = lineMinor.Branches[b]
    setbackMinorBranch = setbackMinor.Branches[b]
    limitZoneBranch = limitZone.Branches[b]

    if parcelBranch and len(parcelBranch)>0:
        #there is a parcel for this sub-branch
        if lineMajorBranch and len(lineMajorBranch)>0:
            #convert all parcel sides with major setback to GeoJSON
            for l in lineMajorBranch:
                points =[]
                #collect and insert points in JSON fragment
                for i in range(l.Count):
                    p = l.Item[i]
                    points.append([p.X,p.Y])
                geomJSON = {"type":"LineString","coordinates":points}
                #loop all parcels in sub-branch (groups may have more than one)
                for par in parcelBranch:
                    jsonLines.append({"pid": project, "parcel":par,
                                      "geom":geomJSON,
                                      "setback": setbackMajorBranch[0]})
        if lineMinorBranch and len(lineMinorBranch)>0:
            #convert all parcel sides with minor setback to GeoJSON
            for l in lineMinorBranch:
                points =[]
                #collect and insert points in JSON fragment
                for i in range(l.Count):
                    p = l.Item[i]
                    points.append([p.X,p.Y])
                geomJSON = {"type":"LineString","coordinates":points}
                #loop all parcels in sub-branch (groups may have more than one)
                for par in parcelBranch:
                    jsonLines.append({"pid": project, "parcel":par,
                                      "geom":geomJSON,
```

```
                                    "setback": setbackMinorBranch[0]})
        #check if limit zones are applicable
        if limitZoneBranch and len(limitZoneBranch)>0:
            limitZoneCount += len(limitZoneBranch)

#convert list to GeoJSON sting
geoJSON = json.dumps(jsonLines)

sqlZones = ""

#if there are setbacks or limit zones, generate subquery for their geometries
#setbacks from lines are converted from GeoJSON
#limit zones are recreated on the server from the original limit lines
if len(jsonZones)>0 or limitZoneCount:
    sqlZones = """
    UNION ALL
    SELECT gid AS pid, nummer AS parcel, geomLimitZone AS setbackGeom
FROM
(SELECT ST_Collect(CASE WHEN ST_Intersects(geomSp,ST_OffsetCurve(geomL,-0.0001, 'quad_segs=4
join=round')) THEN geomSp ELSE NULL END) AS  geomLimitZone,
      gid, nummer, COUNT(*) AS SplitCount
FROM
(SELECT parzelle.gid, parzelle.nummer, (ST_Dump(ST_Split(parzelle.geom,
abstandslinie.geom))).geom AS geomSp, parzelle.geom AS geomP, abstandslinie.geom AS geomL
  FROM dencity.parzelle, dencity.abstandslinie
WHERE abstandslinie.sid="""+str(scenario)+"""
AND parzelle.pid="""+str(project)+"""
AND ST_Intersects(parzelle.geom,abstandslinie.geom)) AS t
GROUP BY gid, nummer
) AS u
WHERE SplitCount>1
    """

#piece together the actual query
if len(jsonLines)>0 or len(jsonZones)>0 or limitZoneCount:
    sqlCommand = """SELECT p.gid, p.nummer, '{ "type": "Feature", "geometry": ' ||
ST_AsGeoJSON(ST_Difference(p.geom,COALESCE(ST_Union(sb.setbackGeom),ST_GeomFromText('POLYGON
EMPTY')))) || ', "properties": {"gid": "'||p.gid||'", "nummer": "'||p.nummer||'"}}' AS geom
FROM dencity.parzelle AS p LEFT JOIN
    (SELECT pid, parcel, ST_Buffer(ST_SetSRID(ST_GeomFromGeoJSON(geom),"""+srid+"""),setback,
'endcap=square join=mitre') AS setbackGeom FROM
json_to_recordset('"""+json.dumps(jsonLines)+"""') AS  (pid int,  parcel character varying,
geom character varying, setback real)
    """+sqlZones+""") AS sb ON sb.parcel = p.nummer
    WHERE p.pid = """+str(project)+""" GROUP BY p.gid, p.nummer;"""
```

B.4.6. Berechnung Ausnutzung (Py Ausnutzung)

Der folgende Code berechnet auf Basis der Ausnützung und anderer Faktoren die Geschosszahl (adjustedStoreyCount), Fläche des Baufeldes (targetArea) und den Dach Typ (roofType) für die nachfolgende Generierung von Gebäuden.

```
import math

#Adjust storey count and roof type for Gesamthoehe
adjustedStoreyCount = storeyCount
if maxHeight:
    if roofType == 0 and storeyCount*storeyHeight > maxHeight: #Flat
        adjustedStoreyCount = math.floor(maxHeight/storeyHeight)
    elif roofType == None:
        if (storeyCount+roofHeightFactor)*storeyHeight > maxHeight:
            print [roofType, storeyCount*storeyHeight, maxHeight]
            if roofType == None and storeyCount*storeyHeight < maxHeight:
                roofType = 0 #Make Flat
            else:
                adjustedStoreyCount = math.floor(maxHeight/storeyHeight)
    storeyCount = adjustedStoreyCount

#Compute target area for Ausnuetzung
if utilizationType!=None:
    #Keine Ausnuetzung
    if utilizationType == 0:
        if aFootprint <= aUsable:
            targetArea = aFootprint
```

```
        else:
            targetArea = aUsable
    #Geschossflaechenziffer
    if utilizationType == 1:
        if (aFootprint*storeyCount) <= (aUsable*utilizationFactor):
            targetArea = aFootprint
        else:
            targetArea = (aUsable*utilizationFactor)/storeyCount
    #Baumasseziffer
    if utilizationType == 2:
        roofFactor = roofHeightFactor
        if roofType == 0:
            roofFactor = 0
        if              (aFootprint*((storeyCount+roofFactor)*storeyHeight))              <=
(aUsable*utilizationFactor):
            targetArea = aFootprint
        else:
            targetArea = (aUsable*utilizationFactor)/((storeyCount+roofFactor)*storeyHeight)
    #Ueberbauungsziffer
    if utilizationType == 3:
        if utilizationFactor > 1:
            utilizationFactor = 1
        if aFootprint <= (aUsable*utilizationFactor):
            targetArea = aFootprint
        else:
            targetArea = aUsable*utilizationFactor

##Adjust target area for Gruenflaechenziffer
if greenspaceRatio > 0:
    aNonGreenspace = aUsable*(1-greenspaceRatio)
    if targetArea > aNonGreenspace:
        targetArea = aNonGreenspace

#Roof type
if roofType == None and aUsable:
    #Adjst type on small and large buildings
    if targetArea > aMinSaddleRoof and targetArea < aMaxSaddleRoof:
        roofType = 1 #Saddle
    else:
        roofType = 0 #Flat
```

B.4.7. Quadtree Zerlegung für Ableitung Gebäudegrundrisse (C# CN Quadtree Footprints)

Der nachfolgende Code dient der Zerlegung der reduzierten Parzellenflächen und der Ableitung der Gebäudegrundrisse mit Hilfe eines Region Quadtrees, der einen Cardinal Neighbor Quadtrees als Datenstruktur verwendet. Die Implementation der Datenstruktur basiert in Teilen auf einer Implementierung in Go (Rainone 2017).

```
using Rhino;
using Rhino.Geometry;
using Rhino.DocObjects;
using Rhino.Collections;

using GH_IO;
using GH_IO.Serialization;
using Grasshopper;
using Grasshopper.Kernel;
using Grasshopper.Kernel.Data;
using Grasshopper.Kernel.Types;

using System;
using System.IO;
using System.Xml;
using System.Xml.Linq;
using System.Linq;
using System.Data;
using System.Drawing;
using System.Reflection;
using System.Collections;
using System.Windows.Forms;
using System.Collections.Generic;
using System.Runtime.InteropServices;

/// <summary>
```

```
/// This class will be instantiated on demand by the Script component.
/// </summary>
public class Script_Instance : GH_ScriptInstance
{
#region Utility functions
  /// <summary>Print a String to the [Out] Parameter of the Script component.</summary>
  /// <param name="text">String to print.</param>
  private void Print(string text) { /* Implementation hidden. */ }
  /// <summary>Print a formatted String to the [Out] Parameter of the Script component.</summary>
  /// <param name="format">String format.</param>
  /// <param name="args">Formatting parameters.</param>
  private void Print(string format, params object[] args) { /* Implementation hidden. */ }
  /// <summary>Print useful information about an object instance to the [Out] Parameter of the
Script component. </summary>
  /// <param name="obj">Object instance to parse.</param>
  private void Reflect(object obj) { /* Implementation hidden. */ }
  /// <summary>Print the signatures of all the overloads of a specific method to the [Out]
Parameter of the Script component. </summary>
  /// <param name="obj">Object instance to parse.</param>
  private void Reflect(object obj, string method_name) { /* Implementation hidden. */ }
#endregion

#region Members
  /// <summary>Gets the current Rhino document.</summary>
  private readonly RhinoDoc RhinoDocument;
  /// <summary>Gets the Grasshopper document that owns this script.</summary>
  private readonly GH_Document GrasshopperDocument;
  /// <summary>Gets the Grasshopper script component that owns this script.</summary>
  private readonly IGH_Component Component;
  /// <summary>
  /// Gets the current iteration count. The first call to RunScript() is associated with
Iteration==0.
  /// Any subsequent call within the same solution will increment the Iteration count.
  /// </summary>
  private readonly int Iteration;
#endregion

  /// <summary>
  /// This procedure contains the user code. Input parameters are provided as regular arguments,
  /// Output parameters as ref arguments. You don't have to assign output parameters,
  /// they will have a default value.
  /// </summary>
  private void RunScript(List<Brep> poly, List<double> ang, List<double> len, List<Line> edg,
double lim, int depthLimit, int maxDepth, double targetArea, ref object A, ref object B, ref
object C, ref object D, ref object E, ref object dW, ref object dH, ref object P)
  {
    //Return if no sufficient inputs are given
    if (poly == null || poly.Count == 0) {
      A = null;
      B = null;

      return;
    }

    //"upper left"
    Point3d corner = edg[2].To;
    Line lineWidth = edg[2];
    Line lineHeight = edg[1];
    double w = Math.Ceiling(len[0]);
    double h = Math.Ceiling(len[1]);
    double dim = Math.Max(w, h);

    //square subdivision
    w = dim;
    h = dim;

    //Direction vectors from bounding box sides
    Vector3d dirW = lineWidth.Direction;
    Vector3d dirH = lineHeight.Direction;

    dirW.Unitize();
    dirH.Unitize();
    dirW.Reverse();

    //Origin of subdivision square
    P = corner;
    dW = dirW;
```

```
    dH = dirH;

    //Construct quad tree
    double tolerance = this.RhinoDocument.ModelAbsoluteTolerance;
    CNQuadTree tree = new CNQuadTree(corner, w, h, dirW, dirH, poly, lim, depthLimit, maxDepth,
tolerance);

    //Prepare tree
    double treeArea = 0;
    tree.determineConnectedComponents();
    tree.determineFillRatio();

    //List for collecting geometries
    List<Curve> geoms = new List<Curve>();

    //Test if there are connected components
    if (tree.connectedComponents.Values.Count == 0) {
      //no connected components -> no quads inside input areas
      //=> fall back to scaled input areas
      double polyArea = 0;
      poly.ForEach(pol => polyArea += pol.GetArea());
      var p = ScaleBreps(poly, targetArea, polyArea);
      A = p;
      polyArea = 0;
      p.ForEach(pol => polyArea += pol.GetArea());
      B = polyArea;
      return;
    }

    //Eliminate small freestanding Quads
    if (tree.whiteNodes.Count > 1){
      //Limits depth limit for freestanding quads to be eliminated
      int singleCullDepth = tree.maxDepth;
      List<CNQNode> singles = new List<CNQNode>();

      //Test for all nodes whether they are "freestanding"
      foreach(CNQNode n in tree.whiteNodes.Values) {
        if(n.exposure < 0.3 && n.depth <= singleCullDepth) {
          singles.Add(n);
        }
      }

      //Turn exposed nodes to black
      if (singles.Count > 0){
        foreach(CNQNode n in singles) {
          tree.turnNodeBlack(n);
        }

        //recalculate fill ratio and exposure
        tree.determineFillRatio();
      }
    }

    if (tree.connectedComponents.Values.Count > 0) {
      //there are connected components (i.e. Quads inside the input areas)
      double maxArea = tree.connectedComponentAreas.Values.Max();

      List<Brep> brepGeom = new List<Brep>();

      //If more than one connected component
      //-> eliminate components smaller than 25% of largest component
      //(prefer larger patches for deriving footprints and avoid artifacts/small patches)
      HashSet<int> excludedComponents = new HashSet<int>();
      if (tree.connectedComponentAreas.Count > 1) {
        foreach(int key in tree.connectedComponents.Keys) {
          if (tree.connectedComponentAreas[key] < 0.25 * maxArea){
            excludedComponents.Add(key);
          }
        }
      }
      bool checkExclusion = excludedComponents.Count == 0;

      //Collect Quads using depth first search until target area is reached
      List<Curve> curveGeom = new List<Curve>();

      Stack<CNQNode> stack = new Stack<CNQNode>();
      CNQNode cNode = tree.root;
```

```
      stack.Push(cNode);
      //limits for inclusion (prefer medium to large blocks)
      int minDepthLimit = 2;
      int maxDepthLimit = maxDepth - 2;
      //double cullLimit = 0.3;
      while (!(stack.Count == 0)) {
        cNode = stack.Pop();
        if (cNode.value == 0) {
          //leaf oudside input -> no operation
          continue;
        } else if (cNode.value == 1
          && treeArea < targetArea
          && (!checkExclusion ||
          checkExclusion && !excludedComponents.Contains(cNode.connectedComponent))
        ) {
          //leaf inside input and not excluded -> collect Quad
          curveGeom.Add(cNode.rect);
          treeArea += cNode.area;
        } else {
          //internal Quad -> check inclusion
          if (cNode.depth < minDepthLimit
            || cNode.depth > maxDepthLimit
            || treeArea < targetArea
            //&& cNode.filledAreaRatio >= cullLimit
            //&& (cNode.filledAreaRatio == 1 || cNode.borderLength>2.5*cNode.w)
            //&&  !(cNode.filledAreaRatio  ==  0.5  &&  !(cNode.filledAreaRatio  ==  0.5  &&
(cNode.borderLength == 2 * cNode.w || cNode.borderLength == 2.5 * cNode.w))
          ) {
            //inclusin criteria are met -> process children
            if (cNode.children[quadrant.nw] != null) {
              stack.Push(cNode.children[quadrant.se]);
              stack.Push(cNode.children[quadrant.sw]);
              stack.Push(cNode.children[quadrant.ne]);
              stack.Push(cNode.children[quadrant.nw]);
            }
          }
        }
      }

      //Merge individual Quads to connected Breps
      brepGeom.AddRange(Brep.JoinBreps(Brep.CreatePlanarBreps(curveGeom), tolerance));

      //Prepare output
      if (treeArea > targetArea && brepGeom.Count > 0) {
        //Quads were collected and area is larger than target
        //-> scale down Breps
        List<Brep> scaledBrepGeom = new List<Brep>();

        double scaledArea = 0;
        double scaleFactor = targetArea / treeArea;
        foreach(Brep b in brepGeom) {
          scaledBrepGeom.Add(ScaleBrep(b, targetArea, treeArea));
          scaledArea += b.GetArea();
        }
        A = scaledBrepGeom;
        B = scaledArea;
        C = treeArea;
        D = targetArea / treeArea;

      } else if (brepGeom.Count > 0) {
        //Quads were collected and area is smaller than target
        //-> return Breps
        A = brepGeom;
        B = treeArea;
      } else {
        //No Quads were Collected -> fall back to scaled input areas
        var p = ScaleBreps(poly, targetArea, treeArea);
        A = p;
        double polyArea = 0;
        p.ForEach(pol => polyArea += pol.GetArea());
        B = polyArea;
      }

    }
  }

  // <Custom additional code>
```

```
/// <summary>
///Method for scaling down a Brep surface to a target area. No scaling is
///performed if area is already smaller.
/// </summary>
/// <param name="poly"></param>
/// <param name="targetArea"></param>
/// <param name="area"></param>
/// <returns></returns>
public Brep ScaleBrep (Brep poly, double targetArea, double area) {
  //Calculate AreaMassProperties to determine Brep area and centroid
  AreaMassProperties amp = AreaMassProperties.Compute(poly);
  if (amp.Area < targetArea){
    //area already smaller than target -> return original Brep
    return poly;
  } else {
    //area larger than target -> scale down Brep
    Point3d center = amp.Centroid;
    double factor = Math.Sqrt(targetArea / area);
    poly.Transform(Transform.Scale(center, factor));
    return poly;
  }
}

/// <summary>
///Method for scaling down a list of Brep surfaces such that their combined area
///is equal or smaller thn a target area. No scaling is performed if ths
///area is already smaller.
/// </summary>
/// <param name="poly"></param>
/// <param name="targetArea"></param>
/// <param name="area"></param>
/// <returns></returns>
public List<Brep> ScaleBreps (List<Brep> poly, double targetArea, double area) {
  //Calculate AreaMassProperties of all Breps
  List<AreaMassProperties> amps = new List<AreaMassProperties>();
  for (int i = 0; i < poly.Count; i++){
    amps.Add(AreaMassProperties.Compute(poly[i]));
  }
  //Sum the area
  double polyArea = 0;
  amps.ForEach(amp => polyArea += amp.Area);

  if (polyArea < targetArea){
    //summed area already smaller than target -> return original Breps
    return poly;
  } else {
    //area larger than target -> scale down individual Breps
    double factor = Math.Sqrt(targetArea / area);

    for (int i = 0; i < poly.Count;i++) {
      Point3d center = amps[0].Centroid;
      poly[i].Transform(Transform.Scale(center, factor));
    }
    return poly;
  }
}

///
///Cardinal Neighbor Quadtree
///

///Direction "constants"/indexes
public class quadrant { public static int nw = 0, ne = 1, sw = 2, se = 3;}
///Neighbor "constants"/indexes
public class neighbor { public static int w = 0, n = 1, e = 2, s = 3;}

/// <summary>
///Node class for Cardinal Neighbor Quad Tree.
///
///Partially based on "Region quadtrees in Go" by Aurelien Rainone 2017
///https://github.com/aurelien-rainone/go-rquad
/// </summary>
public class CNQNode {
  //Direction "constants"
  public static int[] opposite = new int[]{neighbor.e, neighbor.s, neighbor.w, neighbor.n};
  public static int[] traversal = new int[]{neighbor.s, neighbor.e, neighbor.n, neighbor.w};
```

```
    //Node Type: 0: outside/black leaf, 1: inside/white leaf, -1: internal
    public sbyte value;
    //Covered area
    public double area;
    //Width and Height, may differ in non square case
    public double w;
    public double h;

    //"Lower left" corner
    public Point3d p0;
    //Corner coordinates
    public double x, y;
    //Tree level
    public byte depth;

    //Quad geometry
    public Curve rect;
    //Parent node
    public CNQNode parent;

    //Child nodes
    public CNQNode[] children;
    //Cardinal Neighbors
    public CNQNode[] neighbors;
    //Z-Order ID
    public int quad;

    //Degree of exposure (-1: unset)
    //(i.e. ratio of black border to circumference)
    public double exposure = -1;
    //Lenght of borders to black nodes (-1: unset)
    public double borderLength = -1;
    public double borderLengthW = -1;
    public double borderLengthN = -1;
    public double borderLengthE = -1;
    public double borderLengthS = -1;
    //Filled (white) area (-1: unset)
    public double filledArea = -1;
    //Ratio of filled to total area (-1: unset)
    public double filledAreaRatio = -1;

    //ID of connected component
    public int connectedComponent;

    /// <summary>
    /// Create a new CNQNode
    /// </summary>
    /// <param name="parent"></param>
    /// <param name="quad">z-order quad ID</param>
    /// <param name="value">initial value</param>
    /// <param name="p0">"lower left" corner</param>
    /// <param name="x">x coordinate of corner</param>
    /// <param name="y">y coordinate of corner</param>
    /// <param name="w">numeric width of quad</param>
    /// <param name="h">numeric height of quad</param>
    /// <param name="rect">quad geometry</param>
    public CNQNode(CNQNode parent, int quad, sbyte value, Point3d p0, double x, double y, double
w, double h, Curve rect){
      this.value = value;
      this.children = new CNQNode[4];
      this.neighbors = new CNQNode[4];
      this.parent = parent;
      this.quad = quad;

      this.connectedComponent = -1;

      this.p0 = p0;
      this.x = x;
      this.y = y;
      this.w = w;
      this.h = h;
      this.area = this.w * this.h;
      this.rect = rect;
      //determine level
      if (this.parent != null) {
        this.depth = (byte) (this.parent.depth + 1);
```

```
  }
  else {
    this.depth = 0;
  }
}

/// <summary>
/// Determine cardinal neighbors of this node
///
/// Compare Steps in
/// QASEM, S. W. & TOUIR, A. A. 2015.
/// Cardinal Neighbor Quadtree:
/// a New Quadtree-based Structure for Constant-Time Neighbor Finding.
/// International Journal of Computer Applications, 132(8), 22-30.
/// </summary>
/// <returns></returns>
public void updateNeighbors() {
  // On each direction, a full traversal of the neighbors
  // should be performed.  In every quadrant where a reference
  // to the parent quadrant is stored as the Cardinal Neighbor,
  // it should be replaced by one of its children created after
  // the decomposition

  //Step 2.1: Updating Cardinal Neighbors of SW sub-Quadrant.
  if (!(this.parent == null || this.neighbors[neighbor.w] == null)) {
    // Quadrant not on west border
    if (this.neighbors[neighbor.n] != null) {
      if (this.neighbors[neighbor.n].w < this.w) {
        CNQNode c0 = this.children[quadrant.nw];
        c0.neighbors[neighbor.n] = this.neighbors[neighbor.n];
        // to update C2, we perform a north-south traversal
        // recording the cumulative size of traversed nodes
        CNQNode cNode = c0.neighbors[neighbor.w];
        double cSize = cNode.h;
        while (cSize < c0.h) {
          cNode = cNode.neighbors[neighbor.s];
          cSize += cNode.h;
        }
        this.children[quadrant.sw].neighbors[neighbor.w] = cNode;
      }
    }
  }

  // Step 2.2: Updating Cardinal Neighbors of NE sub-Quadrant.
  if (!(this.parent == null || this.neighbors[neighbor.n] == null)) {
    // Quadrant not on north border
    if (this.neighbors[neighbor.n] != null) {
      if (this.neighbors[neighbor.n].w < this.w) {
        CNQNode c0 = this.children[quadrant.nw];
        c0.neighbors[neighbor.n] = this.neighbors[neighbor.n];
        // to update C1, we perform a west-east traversal
        // recording the cumulative size of traversed nodes
        CNQNode cNode = c0.neighbors[neighbor.n];
        double cSize = cNode.w;
        while (cSize < c0.w) {
          cNode = cNode.neighbors[neighbor.e];
          cSize += cNode.w;
        }
        this.children[quadrant.ne].neighbors[neighbor.n] = cNode;
      }
    }
  }

  if (this.neighbors[neighbor.w] != null) {
    Action<CNQNode> opw = delegate (CNQNode node){
        CNQNode western = node;
        if (western.neighbors[neighbor.e] == this) {
          if (western.y + western.h > this.children[quadrant.sw].y) {
            // choose SW
            western.neighbors[neighbor.e] = this.children[quadrant.sw];
          } else {
            // choose NW
            western.neighbors[neighbor.e] = this.children[quadrant.nw];
          }
          if (western.neighbors[neighbor.e].y == western.y) {
            western.neighbors[neighbor.e].neighbors[neighbor.w] = western;
          }
```

```
            }
          };
        this.forEachNeighborInDirection(neighbor.w, opw);
      }

      if (this.neighbors[neighbor.n] != null) {
        Action<CNQNode> ops = delegate (CNQNode node){
            CNQNode northern = node;
            if (northern.neighbors[neighbor.s] == this) {
              if (northern.x + northern.w > this.children[quadrant.ne].x) {
                // choose NE
                northern.neighbors[neighbor.s] = this.children[quadrant.ne];
              } else {
                // choose NW
                northern.neighbors[neighbor.s] = this.children[quadrant.nw];
              }
              if (northern.neighbors[neighbor.s].x == northern.x) {
                northern.neighbors[neighbor.s].neighbors[neighbor.n] = northern;
              }
            }
          };
        this.forEachNeighborInDirection(neighbor.n, ops);
      }

      if (this.neighbors[neighbor.e] != null) {
        if (this.neighbors[neighbor.e] != null &&
this.neighbors[neighbor.e].neighbors[neighbor.w] == this) {
          // To update the eastern CN of a quadrant Q that is being
          // decomposed: Q.CN2.CN0=Q.Ch[NE]
          this.neighbors[neighbor.e].neighbors[neighbor.w] = this.children[quadrant.ne];
        }
      }

      if (this.neighbors[neighbor.s] != null) {
        if (this.neighbors[neighbor.s] != null &&
this.neighbors[neighbor.s].neighbors[neighbor.n] == this) {
          this.neighbors[neighbor.s].neighbors[neighbor.n] = this.children[quadrant.sw];
        }
      }
    }

    /// <summary>
    /// Perform an action for each neighbor in a specific direction
    /// </summary>
    /// <param name="dir"></param>
    /// <param name="operation"></param>
    /// <returns></returns>
    public void forEachNeighborInDirection(int dir, Action<CNQNode> operation) {
      // start from the cardinal neighbor on the given direction
      CNQNode node = this.neighbors[dir];
      if (node == null) {
        return;
      }
      operation(node);
      if (((dir == neighbor.n || dir == neighbor.s) && node.w >= this.w)
      || ((dir == neighbor.w || dir == neighbor.e) && node.h >= this.h)) {
        //no more neighbors -> end traversal
        return;
      }

      //traverse to other neighbors
      int travDir = traversal[dir];
      int oppDir = opposite[dir];

      CNQNode tNode = node.neighbors[travDir];
      while (tNode != null) {
        if (tNode != null && tNode.neighbors[oppDir] == this) {
          operation(tNode);
        } else {
          break;
        }
        tNode = tNode.neighbors[travDir];
      }
    }

    /// <summary>
    /// Perform an operation for each neighbor
```

```
    /// </summary>
    /// <param name="operation"></param>
    /// <returns></returns>
    public void forEachNeighbor(Action<CNQNode> operation) {
      this.forEachNeighborInDirection(neighbor.w, operation);
      this.forEachNeighborInDirection(neighbor.n, operation);
      this.forEachNeighborInDirection(neighbor.e, operation);
      this.forEachNeighborInDirection(neighbor.s, operation);
    }

    /// <summary>
    /// Update exposure values on this node. Only applies to leaf nodes.
    /// </summary>
    /// <returns></returns>
    public void updateExposure () {
      if (this.value != 1){
        //not a leaf
        this.borderLength = -1;
        this.exposure = -1;
        return;
      }

      //Value and action for calculating exposure
      double bLen = 0;
      Action<CNQNode> ops = delegate (CNQNode node){
          int val = node.value;
          if (val < 0){
            val = 0;
          }
          double mult = 0;
          if (node.y < this.y || node.y >= this.y + this.h){ //north or south
            if (node.w > this.w) {
              mult = this.w;
            } else {
              mult = node.w;
            }
          } else { //west or east
            if (node.h > this.h) {
              mult = this.h;
            } else {
              mult = node.h;
            }
          }
          bLen += val * mult;
        };
      //determine exposure by direction
      this.forEachNeighborInDirection(neighbor.w, ops);
      this.borderLengthW = bLen;
      bLen = 0;

      this.forEachNeighborInDirection(neighbor.n, ops);
      this.borderLengthN = bLen;
      bLen = 0;

      this.forEachNeighborInDirection(neighbor.e, ops);
      this.borderLengthE = bLen;
      bLen = 0;

      this.forEachNeighborInDirection(neighbor.s, ops);
      this.borderLengthS = bLen;

      //sum up border lenght and calculate exposure
      this.borderLength  =  this.borderLengthW  +  this.borderLengthN  +  this.borderLengthE  +
this.borderLengthS;

      this.exposure = (this.borderLength / (2 * this.w + 2 * this.h));
    }
  }

  /// <summary>
  ///Cardinal Neighbor Quad Tree data structure for subdividing parcel areas.
  ///
  ///Partially based on "Region quadtrees in Go" by Aurelien Rainone 2017
  ///https://github.com/aurelien-rainone/go-rquad
  /// </summary>
  public class CNQuadTree {
    //Origin ("lower left" corner) of subdivision area
```

```
    public Point3d p0;
    //Width and Height of subdivision area (may differ in non-square case)
    public double w;
    public double h;

    //area of white nodes
    public double area;

    //W direction vector (X-Axis direction)
    public Vector3d dirW;
    //H direction vector (Y-Axis direction)
    public Vector3d dirH;
    //Input area
    public List<Brep> poly;
    //Length limit for subdivision Quad side (w or h)
    public double limit = 1;
    //Overall subdivision depth limit of the tree
    public int maxDepth;
    //Relative subdivision limit (span actually occurring leaf sizes)
    public int depthLimit;
    //Minimal depth with a white node
    public int minDepth;

    //Dictionary of all Nodes
    public Dictionary<int,CNQNode> nodes;

    //List of white Quad geometries
    public List<Curve> geoms;
    //List of outer rings of the input area
    public List<Curve> polyOuter;
    //List of inner rings of the input area (i.e. holes)
    public List<Curve> polyInner;
    //Root node
    public CNQNode root;
    //Comparison tolerance
    public double tolerance;
    //White nodes by Z-Order ID
    public Dictionary<int,CNQNode> whiteNodes;
    //Equivalence list for determining connected components
    public Dictionary<int,int> connectedComponentEqiuvalence;
    //Connected components by ID
    public Dictionary<int,HashSet<CNQNode>> connectedComponents;
    //Area values of connected components by ID
    public Dictionary<int,double> connectedComponentAreas;

    /// <summary>
    /// Create a new CNQuadTree
    /// </summary>
    /// <param name="p0">Origin (lower left corner) of subdivision area</param>
    /// <param name="w">with of subdivision area (x in direction)</param>
    /// <param name="h">height of subdivision area (y in direction)</param>
    /// <param name="dirW">W direction vector (X-Axis direction)</param>
    /// <param name="dirH">H direction vector (Y-Axis direction)</param>
    /// <param name="poly">Brep representing the input area to be subdivided</param>
    /// <param name="lim">subdivision length limit for quads (w or h)</param>
    ///  <param name="depthLimit">relative subdivision limit (span actually occurring leaf
sizes)</param>
    /// <param name="maxDepth">maximum level of subdivision for the constructed tree</param>
    /// <param name="tolerance">comparison tolerance</param>
    public CNQuadTree(Point3d p0, double w, double h, Vector3d dirW, Vector3d dirH, List<Brep>
poly, double lim, int depthLimit, int maxDepth, double tolerance){
      this.nodes = new Dictionary<int,CNQNode>();
      this.geoms = new List<Curve>();
      this.whiteNodes = new Dictionary<int,CNQNode>();

      this.area = 0;

      this.p0 = p0;
      this.w = w;
      this.h = h;
      this.dirW = dirW;
      this.dirH = dirH;
      this.limit = lim;
      this.maxDepth = maxDepth;
      this.depthLimit = depthLimit;
      this.minDepth = System.Int16.MaxValue;
      this.poly = poly;
```

```
      this.tolerance = tolerance;

      //determine inner and outer rings from input area
      this.polyOuter = new List<Curve>();
      this.polyInner = new List<Curve>();
      foreach (Brep p in poly) {
        foreach (BrepLoop lp in p.Loops){
          var crv = lp.To3dCurve();
          if (lp.LoopType == BrepLoopType.Outer){
            this.polyOuter.Add(crv);
          } else if (lp.LoopType == BrepLoopType.Inner){
            this.polyInner.Add(crv);
          }
        }
      }

      //perform subdivision
      this.buildCNTree(p0, w, h);
    }

    /// <summary>
    /// Test whether the given geometry is completely inside the input area
    /// </summary>
    /// <param name="rect"></param>
    /// <returns></returns>
    public int testContainment(Curve rect) {
      int containment = 0;

      Plane plane = Plane.WorldXY;
      //Check if inside an outer loop
      for(int i = 0; i < this.polyOuter.Count;i++){
        Curve loop = this.polyOuter[i];

        RegionContainment relationship = Curve.PlanarClosedCurveRelationship(rect, loop, plane,
this.tolerance);
        if (relationship == RegionContainment.AInsideB){
          //completely within a outer boundary
          containment = 1;
          break;
        } else if (relationship == RegionContainment.MutualIntersection) {
          //intersecting an outer boundary -> children may be conteined
          containment = -1;
          break;
        } else if (relationship == RegionContainment.Disjoint) {
          //outside outer boundary
          containment = 0;
        }
      }

      if (containment != 0){
        //Check if outside all inner loops
        foreach(Curve loop in this.polyInner){
          RegionContainment  relationship  =  Curve.PlanarClosedCurveRelationship(rect,  loop,
plane, this.tolerance);
          if (!(relationship == RegionContainment.Disjoint)) {
            if (relationship == RegionContainment.AInsideB){
              //completely within a hole
              containment = 0;
              break;
            } else if (relationship == RegionContainment.MutualIntersection) {
              //intersecting a hole -> children may be contained
              containment = -1;
              break;
            }
            return containment;
          }
        }
      }

      return containment;
    }

    /// <summary>
    /// Constuct Quad rectangle geometry as PolylineCurve
    /// </summary>
    /// <param name="p0">corner point</param>
    /// <param name="w">width</param>
```

```
    /// <param name="h">height</param>
    /// <returns></returns>
    public Curve createRectGeom(Point3d p0, double w, double h) {
      Vector3d v1 = Vector3d.Multiply(dirW, w);
      Vector3d v3 = Vector3d.Multiply(dirH, h);
      Vector3d v2 = Vector3d.Add(v1, v3);

      Point3d p1 = Point3d.Add(p0, v1);
      Point3d p2 = Point3d.Add(p0, v2);
      Point3d p3 = Point3d.Add(p0, v3);

      Point3d[] corners = new Point3d[]{p0,p1,p2,p3,p0};

      return new PolylineCurve(corners);
    }

    /// <summary>
    /// Collect all geometries from the tree based in a condition.
    /// </summary>
    /// <param name="condition"></param>
    /// <returns></returns>
    public List<Curve> collectGeometries(Func<CNQNode,bool> condition) {
      List<Curve> geoms = new List<Curve>();
      foreach(CNQNode node in this.nodes.Values){
        if(condition(node)){
          geoms.Add(node.rect);
        }
      }
      return geoms;
    }

    /// <summary>
    /// Visit all nodes Post Order and collect the nodes in the specified
    /// List based on a condition.
    /// </summary>
    /// <param name="node"></param>
    /// <param name="condition"></param>
    /// <param name="nodes"></param>
    /// <returns></returns>
    public void visitPostOrder(CNQNode node, Func<CNQNode,bool> condition, List<CNQNode> nodes)
{
      //visit children first
      foreach(CNQNode child in node.children) {
        if (child != null) {
          visitPostOrder(child, condition, nodes);
        }
      }
      //visit parent after
      visitNode(node, condition, nodes);
    }

    /// <summary>
    /// Visit all nodes In Order and collect the nodes in the specified
    /// List based on a condition.
    /// </summary>
    /// <param name="node"></param>
    /// <param name="condition"></param>
    /// <param name="nodes"></param>
    /// <returns></returns>
    public void visitInOrder(CNQNode node, Func<CNQNode,bool> condition, List<CNQNode> nodes)
{
      if(node == null){
        return;
      }
      //visit parent first
      visitNode(node, condition, nodes);
      //visit childrren after
      foreach(CNQNode child in node.children) {
        visitInOrder(child, condition, nodes);
      }
    }

    /// <summary>
    /// Add a node to a list based on the specified condition
    /// </summary>
    /// <param name="node"></param>
    /// <param name="condition"></param>
```

```
    /// <param name="nodes"></param>
    /// <returns></returns>
    private void visitNode(CNQNode node, Func<CNQNode,bool> condition, List<CNQNode> nodes) {
      if (condition(node)){
        nodes.Add(node);
      }
    }

    /// <summary>
    /// Calculate the Z-Order ID of a Quad based in the parent and the quadrant
    /// </summary>
    /// <param name="parent"></param>
    /// <param name="quadrant"></param>
    /// <returns></returns>
    public int generateZOrder (CNQNode parent, int quadrant) {
      if (parent == null){
        return 0;
      }
      int shift = 2 * (this.maxDepth - parent.depth);
      return parent.quad | (quadrant << shift);
    }

    /// <summary>
    /// Initiate the tree construction
    /// </summary>
    /// <param name="p0">origin of the subdivision area</param>
    /// <param name="w">width of the subdivision area</param>
    /// <param name="h">height of the subdivision area</param>
    /// <returns></returns>
    public void buildCNTree(Point3d p0, double w, double h) {
      //Create root node spanning the entire subdivision area
      this.root = this.createAndInsertNode(null, quadrant.nw, p0, w, h, 0, 0);

      //Initiate the recursive subdivision process
      this.splitNode(this.root);
    }

    /// <summary>
    /// Create a new node and add it to the tree
    /// </summary>
    /// <param name="parent">parent  node</param>
    /// <param name="quadrant">quadrant inside the parent</param>
    /// <param name="p0">origin of the node quad</param>
    /// <param name="w">width of the quad</param>
    /// <param name="h">height of the quad</param>
    /// <param name="x">x-coordinate</param>
    /// <param name="y">y-coordinate</param>
    /// <returns></returns>
    public CNQNode createAndInsertNode(CNQNode parent, int quadrant, Point3d p0, double w,
double h, double x, double y) {
      Curve rect = createRectGeom(p0, w, h);
      int zOrder = this.generateZOrder(parent, quadrant);

      //determine value
      sbyte value = -1;
      if (parent == null){
        //root -> "grey" value
        value = -1;
      } else if(parent.value != -1){
        //parent is uniform -> inherit value
        //irregular case during intentional split in post processing
        value = parent.value;
      } else if((w > this.limit || h > this.limit) && parent.depth <= this.maxDepth){
        //determine value by containment
        int containment = this.testContainment(rect);

        if (containment == 1) {
          //inside
          this.geoms.Add(rect);
          value = 1;
        } else  if (containment == 0) {
          //outside
          value = 0;
        } else if ((w / 2 > this.limit || h / 2 > this.limit) && parent.depth + 1 <=
this.maxDepth) {
          //interior
          value = -1;
```

```
        } else {
          //declared outside due to limits
          value = 0;
        }
      } else {
        value = 0;
      }

      //actually create node and add to tree data structures
      CNQNode node = new CNQNode(parent, zOrder, value, p0, x, y, w, h, rect);
      this.nodes[zOrder] = node;
      if (value == 1){
        this.whiteNodes[node.quad] = node;
        this.area += node.area;
        if (node.depth < this.minDepth){
          this.minDepth = node.depth;
        }
      }
      return node;
    }

    /// <summary>
    /// Subdivide node
    ///
    /// Compare Steps in
    /// QASEM, S. W. & TOUIR, A. A. 2015.
    /// Cardinal Neighbor Quadtree:
    /// a New Quadtree-based Structure for Constant-Time Neighbor Finding.
    /// International Journal of Computer Applications, 132(8), 22-30.
    /// </summary>
    /// <param name="node"></param>
    /// <returns></returns>
    public void splitNode(CNQNode node) {
      if (node.depth == this.maxDepth) {
        return;
      }

      //Step 1
      //new width and height
      double w2 = node.w / 2;
      double h2 = node.h / 2;
      //Cectors for children origins
      Vector3d v1 = Vector3d.Multiply(this.dirW, w2);
      Vector3d v2 = Vector3d.Multiply(this.dirH, h2);
      Vector3d v3 = Vector3d.Add(v1, v2);

      //Generating child nodes
      //Rhino.RhinoApp.WriteLine("NW");
      CNQNode nw = this.createAndInsertNode(node, quadrant.nw, node.p0, w2, h2, node.x, node.y);
      //Rhino.RhinoApp.WriteLine("NE");
      CNQNode ne = this.createAndInsertNode(node, quadrant.ne, Point3d.Add(node.p0, v1), w2,
h2, node.x + w2, node.y);
      //Rhino.RhinoApp.WriteLine("SW");
      CNQNode sw = this.createAndInsertNode(node, quadrant.sw, Point3d.Add(node.p0, v2), w2,
h2, node.x, node.y + h2);
      //Rhino.RhinoApp.WriteLine("SE");
      CNQNode se = this.createAndInsertNode(node, quadrant.se, Point3d.Add(node.p0, v3), w2,
h2, node.x + w2, node.y + h2);

      //at creation, each sub-quadrant first inherits its parent external neighbors
      nw.neighbors[neighbor.w] = node.neighbors[neighbor.w];
      nw.neighbors[neighbor.n] = node.neighbors[neighbor.n];
      nw.neighbors[neighbor.e] = ne;
      nw.neighbors[neighbor.s] = sw;

      ne.neighbors[neighbor.w] = nw;
      ne.neighbors[neighbor.n] = node.neighbors[neighbor.n];
      ne.neighbors[neighbor.e] = node.neighbors[neighbor.e];
      ne.neighbors[neighbor.s] = se;

      sw.neighbors[neighbor.w] = node.neighbors[neighbor.w];
      sw.neighbors[neighbor.n] = nw;
      sw.neighbors[neighbor.e] = se;
      sw.neighbors[neighbor.s] = node.neighbors[neighbor.s];

      se.neighbors[neighbor.w] = sw;
      se.neighbors[neighbor.n] = ne;
```

```
      se.neighbors[neighbor.e] = node.neighbors[neighbor.e];
      se.neighbors[neighbor.s] = node.neighbors[neighbor.s];

      node.children[quadrant.nw] = nw;
      node.children[quadrant.ne] = ne;
      node.children[quadrant.sw] = sw;
      node.children[quadrant.se] = se;

      //Not a leaf anymore
      node.value = -1;

      //Step 2 and Step 3
      node.updateNeighbors();

      // subdivide non-leaf nodes
      if (node.depth + 1 < this.maxDepth
      && (this.minDepth == System.Int16.MaxValue || node.depth - this.minDepth + 1 <
this.depthLimit)) {
        if (nw.value == -1) {
          this.splitNode(nw);
        }
        if (ne.value == -1) {
          this.splitNode(ne);
        }
        if (sw.value == -1) {
          this.splitNode(sw);
        }
        if (se.value == -1) {
          this.splitNode(se);
        }
      } else {
        //declare external/blach due to limits

        if (nw.value == -1) {
          nw.value = 0;
        }
        if (ne.value == -1) {
          ne.value = 0;
        }
        if (sw.value == -1) {
          sw.value = 0;
        }
        if (se.value == -1) {
          se.value = 0;
        }
      }
    }

    /// <summary>
    /// Intentionally turn a node black/external. Intended for postprocessing.
    /// </summary>
    /// <param name="node"></param>
    /// <returns></returns>
    public void turnNodeBlack(CNQNode node) {
      if (node.value != 1){
        return;
      }
      node.value = 0;
      this.whiteNodes.Remove(node.quad);
    }

    /// <summary>
    /// Determine connected components of white nodes in the tree.
    ///
    /// Derived from
    /// SAMET, H. 1981.
    /// Connected Component Labeling Using Quadtrees.
    /// Journal of the ACM (JACM), 28(3), 487-501.
    /// </summary>
    /// <returns></returns>
    public void determineConnectedComponents () {
      //Data structures for labeling
      this.connectedComponentEqiuvalence = new Dictionary<int,int>();
      this.connectedComponents = new Dictionary<int,HashSet<CNQNode>>();
      this.connectedComponentAreas = new Dictionary<int,double>();

      //Function implementing the labeling steps when visiting the nodes
```

```
        Func < CNQNode,bool> visitLabel = delegate (CNQNode node){
            if (node.value != 1) {
              //black or internal
              return false;
            }

            //get component labels frim neighbors in N and W direction
            HashSet<int> components = new HashSet<int>();
            Action <CNQNode> getComponent = delegate(CNQNode vNode) {
                if(vNode.value == 1 && vNode.connectedComponent != -1){
                  components.Add(vNode.connectedComponent);
                }
              };

            node.forEachNeighborInDirection(neighbor.n, getComponent);
            node.forEachNeighborInDirection(neighbor.w, getComponent);

            //determine label
            if (components.Count == 0){ //new component
              node.connectedComponent = node.quad;
            } else if (components.Count == 1){ //one existing component
              node.connectedComponent = components.First();
            } else {
              int minComponent = components.Min();
              node.connectedComponent = minComponent;
              foreach(int c in components) {
                this.connectedComponentEqiuvalence[c] = minComponent;
              }
            }

            return true;
          };

        //label all nodes
        List<CNQNode> visitedNodes = new List<CNQNode>();
        this.visitPostOrder(this.root, visitLabel, visitedNodes);

        //determine equivalent components
        Dictionary<int,int>                ccEquivNormalized                =                new
Dictionary<int,int>(this.connectedComponentEqiuvalence);
        var sortedKeys = this.connectedComponentEqiuvalence.Keys.ToList();
        sortedKeys.Sort();
        foreach(int key in sortedKeys) {
          int prevKey = key;
          int cKey = this.connectedComponentEqiuvalence[key];
          while(prevKey != cKey) {
            prevKey = cKey;
            cKey = this.connectedComponentEqiuvalence[prevKey];
          }
          ccEquivNormalized[key] = cKey;
        }

        //merge equivalent components
        this.connectedComponentEqiuvalence = ccEquivNormalized;
        foreach(CNQNode node in visitedNodes) {
          //initial component label
          int equivalentComponent = node.connectedComponent;
          //determine equivalent label
          if(this.connectedComponentEqiuvalence.ContainsKey(node.connectedComponent)){
            equivalentComponent = this.connectedComponentEqiuvalence[node.connectedComponent];
          }
          node.connectedComponent = equivalentComponent;
          if (this.connectedComponents.ContainsKey(equivalentComponent)){
            //add to existing component
            this.connectedComponents[equivalentComponent].Add(node);
            this.connectedComponentAreas[equivalentComponent] += node.area;
          } else {
            //add new component for label
            HashSet<CNQNode> component = new HashSet<CNQNode>();
            component.Add(node);
            this.connectedComponents[equivalentComponent] = component;
            this.connectedComponentAreas[equivalentComponent] = node.area;
          }
        }
      }
      /// <summary>
      /// Calculate fill ratios (white ares to total area) for nodes in the tree.
```

```
    /// Also updates the exposure values of the nodes
    /// </summary>
    /// <returns></returns>
    public void determineFillRatio () {
      //Function for visiting nodes
      Func <CNQNode,bool> visitLabel = delegate (CNQNode node){
          if (node.value == 1) {
            //white leaf
            node.filledArea = node.area;
            node.updateExposure();
          } else if (node.value == 0) {
            //black leaf
            node.filledArea = 0;
            node.borderLength = 0;
          } else {
            //internal node
            node.borderLength = 0;
            node.filledArea = 0;
            //sum up values of children (possible since post order)
            foreach(CNQNode c in node.children) {
              if (c != null) {
                node.filledArea += c.filledArea;
                node.borderLength += c.borderLength;
              }
            }
          }

          //calculate actual ratio
          node.filledAreaRatio = node.filledArea / node.area;

          return true;
        };

      //Visit nodes post order
      List<CNQNode> visitedNodes = new List<CNQNode>();
      this.visitPostOrder(this.root, visitLabel, visitedNodes);
    }
  }

  // </Custom additional code>
}
```

B.4.8. Generierung SQL Abfrage für Höhenmodell (Py DEM Query from Extent)

Der nachfolgende Code generiert basierend aus den Eckpunkten der Bounding Box der Parzellen im Projektgebiet eine SQL Query für die Abfrage der Punkte des Höhenmodells.

```
import Rhino
import Grasshopper as gh
import Rhino.Geometry as rg

#Parametrize the bounding box tht defines the query extent
bbox = "ST_MakeBox2D(ST_Point({0}-50,{1}-50),ST_Point({2}+50,{3}+50))"
bbox = bbox.format(min_corner.X,min_corner.Y,max_corner.X,max_corner.Y)

#Parametrize the actual query
query = """
SELECT  '{ "type": "Feature", "geometry": ' ||
ST_AsGeoJSON(ST_Translate(ST_Force3D(geom),0.0,0.0,val)) || ',
      "properties": {"prop0": "value0"}
      }' AS geom, x, y, val FROM (
          SELECT (ST_PixelAsPoints(rast, 1)).*
            FROM ch.srtm_dhm
            WHERE ST_Intersects(rast,ST_SetSRID("""+bbox+""",2056))
) t;
"""

#Output the extents bounding box as Rhino PolylineCurve
border = 150
points = [rg.Point3d(min_corner.X-border, min_corner.Y-border, 0),
          rg.Point3d(max_corner.X+border, min_corner.Y-border, 0),
          rg.Point3d(max_corner.X+border, max_corner.Y+border, 0),
          rg.Point3d(min_corner.X-border, max_corner.Y+border, 0) ]
points.Add(points[0])
bounds = rg.PolylineCurve(points)
```

B.4.9. Generierung SQL Abfrage für Gebäude im Umfeld (Py Query Buildings from Extent)

Der nachfolgende Code generiert basierend aus den Eckpunkten der Bounding Box des Höhenmodells im Projektgebiet eine SQL Query für die Abfrage der swissBUILDINGS3D 2.0 Gebäudemodelle im Umfeld.

```
bbox = "ST_SetSRID(ST_MakeBox2D(ST_Point({0},{1}),ST_Point({2},{3})),2056)"
bbox = bbox.format(min_corner.X,min_corner.Y,max_corner.X,max_corner.Y)

query = """
SELECT
'{           "type":           "Feature",           "geometry":           '           ||
ST_AsGeoJSON(Replace(ST_AsText(geom),'POLYHEDRALSURFACE','MULTIPOLYGON')) || ',
      "properties": {"prop0": "value0"}
      }' AS geom,  quad
  FROM ch.swissbuildings3d20
  WHERE ST_Within(geom_footprint,"""+bbox+""")
"""

if excludeProject:
    query = query +   " AND NOT ST_Intersects(geom_footprint ,(SELECT ST_Union(geom) FROM
dencity.parzelle WHERE pid="+project+"))"
```

B.4.10. Generierung SQL Abfrage für Strassen im Umfeld (Py Query Streets from Extent)

Der nachfolgende Code generiert basierend aus den Eckpunkten der Bounding Box des Höhenmodells im Projektgebiet eine SQL Query für die Abfrage der Bodenbedeckung, welche als Strasse, Weg oder Verkehrsinsel klassifiziert sind. Um Geometrien zu erhalten, welche mit Grasshopper verarbeitet werden können, wird die Bodenbedeckung mit den Parzellen verschnitten.

```
bbox = "ST_SetSRID(ST_MakeBox2D(ST_Point({0},{1}),ST_Point({2},{3})),2056)"
bbox = bbox.format(min_corner.X,min_corner.Y,max_corner.X,max_corner.Y)

query = """
SELECT
'{ "type": "Feature", "geometry": ' ||
ST_AsGeoJSON(ST_Multi(ST_CollectionExtract(ST_Intersection(ST_Intersection(mopube_b
odenbedeckung.geom,mopube_grundstueck.geom),"""+bbox+"""),3))) || ',
      "properties": {"prop0": "value0"}
      }' AS geom, art
  FROM be.mopube_bodenbedeckung
  LEFT JOIN be.mopube_grundstueck  ON
ST_Intersects(mopube_grundstueck.geom,mopube_bodenbedeckung.geom)
  WHERE
       ST_Intersects(mopube_bodenbedeckung.geom,"""+bbox+""")
       AND be.mopube_bodenbedeckung.art IN (1,2,3);
"""
```

B.4.11. UI für Projekt und Szenarien Selektion (C# Update Project Selection und C# Update Scenario Selection)

Der nachfolgende Code dient der Erzeugung und Aktualisierung der Auswahlliste mit verfügbaren Projekten basierend auf dem Input einer SQL Query. Der Code basiert teilweise auf einem Beispiel in Ramsden (2015).

```
using Rhino;
using Rhino.Geometry;
using Rhino.DocObjects;
using Rhino.Collections;

using GH_IO;
using GH_IO.Serialization;
using Grasshopper;
using Grasshopper.Kernel;
using Grasshopper.Kernel.Data;
using Grasshopper.Kernel.Types;

using System;
using System.IO;
```

```
using System.Xml;
using System.Xml.Linq;
using System.Linq;
using System.Data;
using System.Drawing;
using System.Reflection;
using System.Collections;
using System.Windows.Forms;
using System.Collections.Generic;
using System.Runtime.InteropServices;

/// <summary>
/// This class will be instantiated on demand by the Script component.
/// </summary>
public class Script_Instance : GH_ScriptInstance
{
#region Utility functions
  /// <summary>Print a String to the [Out] Parameter of the Script component.</summary>
  /// <param name="text">String to print.</param>
  private void Print(string text) { /* Implementation hidden. */ }
  /// <summary>Print a formatted String to the [Out] Parameter of the Script component.</summary>
  /// <param name="format">String format.</param>
  /// <param name="args">Formatting parameters.</param>
  private void Print(string format, params object[] args) { /* Implementation hidden. */ }
  /// <summary>Print useful information about an object instance to the [Out] Parameter of the
Script component. </summary>
  /// <param name="obj">Object instance to parse.</param>
  private void Reflect(object obj) { /* Implementation hidden. */ }
  /// <summary>Print the signatures of all the overloads of a specific method to the [Out]
Parameter of the Script component. </summary>
  /// <param name="obj">Object instance to parse.</param>
  private void Reflect(object obj, string method_name) { /* Implementation hidden. */ }
#endregion

#region Members
  /// <summary>Gets the current Rhino document.</summary>
  private readonly RhinoDoc RhinoDocument;
  /// <summary>Gets the Grasshopper document that owns this script.</summary>
  private readonly GH_Document GrasshopperDocument;
  /// <summary>Gets the Grasshopper script component that owns this script.</summary>
  private readonly IGH_Component Component;
  /// <summary>
  /// Gets the current iteration count. The first call to RunScript() is associated with
Iteration==0.
  /// Any subsequent call within the same solution will increment the Iteration count.
  /// </summary>
  private readonly int Iteration;
#endregion

  /// <summary>
  /// This procedure contains the user code. Input parameters are provided as regular arguments,
  /// Output parameters as ref arguments. You don't have to assign output parameters,
  /// they will have a default value.
  /// </summary>
  private void RunScript(bool refresh, DataTree<object> data, object selectionList, ref object
updateDependent)
  {
    //Based on "Instantiate a Value List Grasshopper component with C#" by James Ramsden
    //http://james-ramsden.com/instantiate-a-value-list-grasshopper-component-with-c/

    if(refresh && Component.Params.Input[2].SourceCount == 0)
    {
      //Value list at input does not exist yet

      //instantiate  new value list
      var vallist = new Grasshopper.Kernel.Special.GH_ValueList();
      vallist.CreateAttributes();
      vallist.Name = "Projekt Auswahl";
      vallist.NickName = "Projekt:";
      vallist.ListMode = Grasshopper.Kernel.Special.GH_ValueListMode.DropDown;

      //customise value list position
      int inputcount = this.Component.Params.Input[2].SourceCount;
      vallist.Attributes.Pivot               =               new               PointF((float)
this.Component.Attributes.DocObject.Attributes.Bounds.Left - vallist.Attributes.Bounds.Width -
30, (float) this.Component.Params.Input[1].Attributes.Bounds.Y + inputcount * 30);
```

```
      //populate value list with our own data
      vallist.ListItems.Clear();

      //Until now, the slider is a hypothetical object.
      // This command makes it 'real' and adds it to the canvas.
      GrasshopperDocument.AddObject(vallist, false);

      //Connect the new slider to this component
      this.Component.Params.Input[2].AddSource(vallist);
    }
    if (refresh && data != null && data.Branches.Count == 4) {

      //Get value list and branches from DB input
      var            vallist            =            (Grasshopper.Kernel.Special.GH_ValueList)
this.Component.Params.Input[2].Sources[0];
      var idBranch = data.Branches[0];
      var labelBranch = data.Branches[1];

      //Empty value list
      vallist.ListItems.Clear();

      //Populate with project list from DB input
      for(int i = 0; i < idBranch.Count; i++){
        var id = idBranch[i];
        var label = labelBranch[i];
        vallist.ListItems.Add(new
Grasshopper.Kernel.Special.GH_ValueListItem(label.ToString(), id.ToString()));
      }

      //Initiate refresh of depending components and recompute of solution
      vallist.ExpireSolution(true);
      updateDependent = true;
    } else {
      updateDependent = false;
    }
  }

  // <Custom additional code>

  // </Custom additional code>
}
```

Der analoge Code für die Erzeugung und Aktualisierung der Auswahlliste mit verfügbaren Szenarien.

```
using Rhino;
using Rhino.Geometry;
using Rhino.DocObjects;
using Rhino.Collections;

using GH_IO;
using GH_IO.Serialization;
using Grasshopper;
using Grasshopper.Kernel;
using Grasshopper.Kernel.Data;
using Grasshopper.Kernel.Types;

using System;
using System.IO;
using System.Xml;
using System.Xml.Linq;
using System.Linq;
using System.Data;
using System.Drawing;
using System.Reflection;
using System.Collections;
using System.Windows.Forms;
using System.Collections.Generic;
using System.Runtime.InteropServices;

/// <summary>
/// This class will be instantiated on demand by the Script component.
/// </summary>
public class Script_Instance : GH_ScriptInstance
{
#region Utility functions
  /// <summary>Print a String to the [Out] Parameter of the Script component.</summary>
  /// <param name="text">String to print.</param>
```

```
  private void Print(string text) { /* Implementation hidden. */ }
  /// <summary>Print a formatted String to the [Out] Parameter of the Script component.</summary>
  /// <param name="format">String format.</param>
  /// <param name="args">Formatting parameters.</param>
  private void Print(string format, params object[] args) { /* Implementation hidden. */ }
  /// <summary>Print useful information about an object instance to the [Out] Parameter of the
Script component. </summary>
  /// <param name="obj">Object instance to parse.</param>
  private void Reflect(object obj) { /* Implementation hidden. */ }
  /// <summary>Print the signatures of all the overloads of a specific method to the [Out]
Parameter of the Script component. </summary>
  /// <param name="obj">Object instance to parse.</param>
  private void Reflect(object obj, string method_name) { /* Implementation hidden. */ }
#endregion

#region Members
  /// <summary>Gets the current Rhino document.</summary>
  private readonly RhinoDoc RhinoDocument;
  /// <summary>Gets the Grasshopper document that owns this script.</summary>
  private readonly GH_Document GrasshopperDocument;
  /// <summary>Gets the Grasshopper script component that owns this script.</summary>
  private readonly IGH_Component Component;
  /// <summary>
  /// Gets the current iteration count. The first call to RunScript() is associated with
Iteration==0.
  /// Any subsequent call within the same solution will increment the Iteration count.
  /// </summary>
  private readonly int Iteration;
#endregion

  /// <summary>
  /// This procedure contains the user code. Input parameters are provided as regular arguments,
  /// Output parameters as ref arguments. You don't have to assign output parameters,
  /// they will have a default value.
  /// </summary>
  private void RunScript(bool refresh, DataTree<object> data, object selectionList, ref object
updateDependent)
  {
    //Based on "Instantiate a Value List Grasshopper component with C#" by James Ramsden
    //http://james-ramsden.com/instantiate-a-value-list-grasshopper-component-with-c/

    if(refresh) {
      if(data != null && Component.Params.Input[2].SourceCount == 0)
      {
        //instantiate  new value list
        var vallist = new Grasshopper.Kernel.Special.GH_ValueList();
        vallist.CreateAttributes();
        vallist.Name = "Szenario Auswahl";
        vallist.NickName = "Szenario:";
        vallist.ListMode = Grasshopper.Kernel.Special.GH_ValueListMode.DropDown;

        //customise value list position
        int inputcount = this.Component.Params.Input[2].SourceCount;
        vallist.Attributes.Pivot                =                new                PointF((float)
this.Component.Attributes.DocObject.Attributes.Bounds.Left - vallist.Attributes.Bounds.Width -
30, (float) this.Component.Params.Input[1].Attributes.Bounds.Y + inputcount * 30);

        //populate value list with our own data
        vallist.ListItems.Clear();

        //Until now, the slider is a hypothetical object.
        // This command makes it 'real' and adds it to the canvas.
        GrasshopperDocument.AddObject(vallist, false);

        //Connect the new slider to this component
        this.Component.Params.Input[2].AddSource(vallist);
      }
      if (data != null && data.Branches.Count == 5) {

        //Get value list and branches from DB input
        var            vallist            =            (Grasshopper.Kernel.Special.GH_ValueList)
this.Component.Params.Input[2].Sources[0];
        var idBranch = data.Branches[0];
        var labelBranch = data.Branches[1];

        //Empty value list
        vallist.ListItems.Clear();
```

```
        //Populate with project list from DB input
        for(int i = 0; i < idBranch.Count; i++){
          var id = idBranch[i];
          var label = labelBranch[i];
          vallist.ListItems.Add(new
Grasshopper.Kernel.Special.GH_ValueListItem(label.ToString(), id.ToString()));
        }

        //Initiate refresh of depending components and recompute of solution
        vallist.ExpireSolution(true);
      }
    }
    updateDependent = true;
  }

  // <Custom additional code>

  // </Custom additional code>
}
```

B.4.12. Anwendung geladener Szenarien auf Parameter UI (Py Set Scenario Sliders)

Der folgende Code nimmt die aus der Datenbank abgefragten Parameterwerte eines Szenarios und wendet die Werte auf die UI Komponenten der entsprechenden Parameter an.

```
import Rhino
import Grasshopper as gh
import System

#Names of attribute columns
_zone = "zone_code"
_parameterName = "parameter_name"
_parameterValue = "parameter_wert"

doc = ghenv.Component.OnPingDocument()

#Method for converting a value to a specified type.
#Supports float,int, bool and str as target types.
#None is returned, if conversion fails.
def getStringAsType(val,valType):
    if not val or not valType:
        return None

    if valType.ToLower() == "float":
        try:
            return float(val)
        except:
            return None
    elif valType.ToLower() == "int":
        try:
            return int(val)
        except:
            return None
    elif valType.ToLower() == "bool":
        try:
            return val.ToLower() == "true"
        except:
            return None
    elif valType.ToLower() == "str":
        return str(val)
    else:
        return None

#Method for returning the input components for a specified model zone code
def getZoneParameterComponents(zoneCode):
    components = []

    if not zoneCode:
        return components
    #loop over all components and search for zone code in nick name
    for o in doc.Objects:
        if o.NickName.ToLower().find(zoneCode.ToLower()) == 0:
            components.append(o)
    return components
```

```
#Method for retreiving a input component based on its nick name
def getComponentByNickName(nickName):
    if not nickName:
        return Nothing
    #get components and loop through them to find the specified nick name
    for component in ghenv.Component.OnPingDocument().Objects:
        if component.NickName == nickName:
            return component


if refresh: #Update scenario parameters from database
    #Get components for project/scenario ID and label and set their value
    sidParameter = getComponentByNickName("SzenarioId")
    sidParameter.SetUserText(str(scenario))

    pidParameter = getComponentByNickName("ProjektId")
    pidParameter.SetUserText(str(project))

    scenarioParameter = getComponentByNickName("Szenario:")
    slabelParameter = getComponentByNickName("SzenarioLabel")
    slabelParameter.SetUserText(str(scenarioParameter.SelectedItems[0].Name))

    projectParameter = getComponentByNickName("Projekt:")
    plabelParameter = getComponentByNickName("ProjektLabel")
    plabelParameter.SetUserText(str(projectParameter.SelectedItems[0].Name))

    #Prepare to set zoning parameters
    defaults = False
    defaultZones = []
    zoneCodeIndex = headings.index(_zone)
    paramNameIndex = headings.index(_parameterName)
    paramValueIndex = headings.index(_parameterValue)

    paramNames = []
    for n in parameters:
        paramNames.append(n.ToLower())

    #Collect parameter values in dictionary
    paramValues = {}
    if data.BranchCount>0:
        zoneCodeBranch = data.Branches[zoneCodeIndex]
        paramNameBranch = data.Branches[paramNameIndex]
        paramValueBranch = data.Branches[paramValueIndex]
        for i in range(len(data.Branches[zoneCodeIndex])):
            code = zoneCodeBranch[i]
            name = paramNameBranch[i]
            value = paramValueBranch[i]
            #key coreesponding to component nickname (convention)
            key = code.ToLower() + "_" + name.ToLower()
            paramValues[key] = value

    #Set values for each zone
    for zone in zoneCodes:
        zoneComponents = getZoneParameterComponents(zone)
        addToDefaultZones = False
        zoneIndex = None
        for component in zoneComponents:
            value = None
            key = component.NickName.ToLower()
            name = key[key.index("_")+1:]
            paramIndex = paramNames.IndexOf(name)

            #determine parameter value
            if paramValues.ContainsKey(key): #has value
                value = paramValues[key]
            else: #has no value -> determine default
                value = parameterDefaults[paramIndex]
                addToDefaultZones = True
            valueCast = getStringAsType(value, parameterTypes[paramIndex])


            #set parameter value on component
            if isinstance(component, gh.Kernel.Special.GH_NumberSlider):
                component.Slider.Value = System.Decimal(float(valueCast))
            elif isinstance(component, gh.Kernel.Special.GH_Panel):
                component.SetUserText(valueCast)
```

```
            elif isinstance(component, gh.Kernel.Special.GH_ValueList):
                component.SelectItem(valueCast)
            else:
                continue

            #expire component
            component.ExpireSolution(True)

        if addToDefaultZones:
            defaults = True
            defaultZones.append(zone)

    if defaults:
        print("At least some values were laoded for the following zones:
{0}".format(defaultZones))
        #Message box UI crashes Grasshopper!
        #icon = System.Windows.Forms.MessageBoxIcon.Information
        #btn = System.Windows.Forms.MessageBoxButtons.OK
        #Rhino.UI.Dialogs.ShowMessageBox("Default values were laoded for the following zones:
{0}".format(defaultZones),"Information",btn,icon)
        #rs.MessageBox("Default values were laoded for the following zones:
{0}".format(defaultZones))
```

B.4.13. Generierung SQL für Speicherung von Szenario Parametern (Py Save Scenario)

Der folgende Code liest die Parameterwerte der UI Komponenten für die Zonenparameter aus und generiert eine Reihe INSERT/UPDATE Statements für deren Speicherung.

```
import Rhino
import Grasshopper as gh

#Names of attribute columns
_zone = "zone_code"
_parameterName = "parameter_name"
_parameterValue = "parameter_wert"

doc = ghenv.Component.OnPingDocument()

#Method for returning the input components for a specified model zone code
def getZoneParameterComponents(zoneCode):
    components = []

    if not zoneCode:
        return components
    #loop over all components and search for zone code in nick name
    for o in doc.Objects:
        if o.NickName.ToLower().find(zoneCode.ToLower()) == 0:
            components.append(o)
    return components

#Method for retreiving a input component based on its nick name
def getComponentByNickName(nickName):
    if not nickName:
        return Nothing
    #get components and loop through them to find the specified nick name
    for component in ghenv.Component.OnPingDocument().Objects:
        if component.NickName == nickName:
            return component

sqlCommand = "";
if save:
    #Names of parameters to be saved
    paramNames = []
    for n in parameters:
        paramNames.append(n.ToLower())

    #For all zones present in the model
    for zone in zoneCodes:
        #Get and loop parameter UI components
        zoneComponents = getZoneParameterComponents(zone)

        for component in zoneComponents:
            parameterValues = [scenario,zone] #value array for this parameter
            value = None
```

```
            key = component.NickName.ToLower()
            name = key[key.index("_")+1:]
            paramIndex = paramNames.IndexOf(name)

            #Get Value from UI component basing on its type
            if isinstance(component, gh.Kernel.Special.GH_NumberSlider):
                value = component.Slider.Value
            elif isinstance(component, gh.Kernel.Special.GH_Panel):
                value = component.UserText
            elif isinstance(component, gh.Kernel.Special.GH_ValueList):
                selectedItems = component.SelectedItems
                if selectedItems.Count:
                    value = selectedItems[0].Value
            else:
                print type(component)
            parameterValues.append(parameters[paramIndex])
            parameterValues.append(value)
            #Append INSERT/UPDATE Statement to SQL statement
            query = ("INSERT INTO dencity.szenario_parameter(sid, zone_code, parameter_name,
parameter_wert) VALUES ({0}, '{1}', '{2}', '{3}') ON CONFLICT ON CONSTRAINT
szenario_parameter_unique DO UPDATE SET parameter_wert=
'{3}';").format(parameterValues[0],parameterValues[1],parameterValues[2],parameterValues[3])

            sqlCommand = sqlCommand + query
    execute = True
else:
    execute = False
sql = sqlCommand
```

B.4.14. Konversion Gebäudegrundrisse in GeoJSON (Py Footprint to GeoJSON)

Der folgende Code konvertiert die Grundriss Geometrien der generierten Gebäude in MultiPolygon GeoJSON Geometrien, die anschliessend für die Speicherung der Szenario Resultate in der Datenbank als Input verwendet werden.

```
import Rhino.Geometry as rg
import pprint
import json

res = {}
mpoly = {}
mpoly["type"] = "MultiPolygon"
poly = []
if not geometry == None:
    #for all loops/rings in the input geometry
    for lp in geometry.Loops:
        ring = []
        segment = None
        crv = lp.To3dCurve()
        if lp.LoopType == rg.BrepLoopType.Outer:
            #outer loop -> clockwise
            for i in range(0,crv.SegmentCount,1):
                segment = crv.SegmentCurve(i)
                ring.append([segment.PointAtStart[0],segment.PointAtStart[1]])
            #close ring
            ring.append([segment.PointAtEnd[0],segment.PointAtEnd[1]])
        elif lp.LoopType == rg.BrepLoopType.Inner:
            #inner loop ("hole") -> counter clockwise
            for i in range(crv.SegmentCount-1,-1,-1):
                segment = crv.SegmentCurve(i)
                ring.append([segment.PointAtEnd[0],segment.PointAtEnd[1]])
            #close ring
            ring.append([segment.PointAtStart[0],segment.PointAtStart[1]])
        poly.append(ring)
    #convert the collection to a GeoJSON string
    mpoly["coordinates"]=[poly]
    geoJSON = json.dumps(mpoly)
else:
    geoJSON = None
```

B.4.15. Generierung SQL für Speicherung Szenario Resultate (Py Save Results)

Der folgende Code generiert auf Basis der Inputs die SQL Statements für die Speicherung der Szenario Resultate in der PostGIS Datenbank.

```
import Rhino
import Grasshopper as gh

sqlCommand = "";
if save:
    #statement to delete existing results for the scenario
    sqlCommand += "DELETE FROM dencity.gebaeude WHERE sid={0};".format(scenario)
    #loop all inputs
    for p in geom.Paths:
        geomBranch = geom.Branch(p)
        #one path/parcel may contain more than one building
        for j in range(geomBranch.Count):
            g = geomBranch[j]

            #if a geometry exists, parametrize an INSERT statement
            if g:
                h = 0
                h = height.Item[p,0]
                sc = 0
                sc = storeyCount.Item[p,0]
                sa = 0
                sa = storeyArea.Item[p,j]
                wur = 0
                wur = workUseRatio.Item[p,0]
                ul = 0
                ul = usersLiving.Item[p,j]
                uw = 0
                uw = usersWork.Item[p,j]
                query = ("INSERT INTO dencity.gebaeude(sid, hoehe, geschosszahl,
geschossflaeche, anteil_arbeitsnutzung, nutzer_wohnen, nutzer_arbeit, geom) VALUES ({0}, {1},
{2}, {3}, (NFP48), (NFP65), (NFP65),
ST_SetSRID(ST_GeomFromGeoJSON('{7}'),(NFP48)));").format(scenario,h,sc,sa,wur,ul,uw,g,srid)
                print query
                sqlCommand += query
    #execute subsequent SQL Command component
    execute = True
else:
    #do not execute subsequent SQL Command component
    execute = False
sql = sqlCommand
```

B.4.16. Generierung QGIS Projekte (Py Create Analysis Projects)

Der folgende Code dient der Generierung von QGIS Projekten durch Parametrisierung von Templates basierend auf den Inputs.

```
import Rhino
import Rhino.Geometry as rg
import Grasshopper as gh

import os
import System

doc = ghenv.Component.OnPingDocument()

#Remove illegal characters from filename
def sanitizeFilename(filename):
    for c in r'[]/\;,><&*:%=+@!#^()|?^ ':
        filename = filename.replace(c,'_')
    return filename

#do not execute dependent components for generating visuals
updateVisuals = False

#Prepare paths
sanitizedProject = sanitizeFilename(projectLabel)
projectPath = os.path.join(baseProjectPath,sanitizedProject)
projectPath = projectPath + "\\"
```

```
sanitizedScenario = sanitizeFilename(scenarioLabel)
scenarioPath = os.path.join(projectPath,sanitizedScenario)
scenarioPath = scenarioPath + "\\"

scenarioFileName = sanitizedScenario

#Create folders if they do not exist
if not os.path.exists(projectPath):
    print ""
    os.makedirs(projectPath)

if not os.path.exists(scenarioPath):
    os.makedirs(scenarioPath)

#Calculate map layout bounds
ratio = 0.818181818 #ratio of layout maps
pMin = projectBounds.PointAt(0)
xMin = pMin.X
yMin = pMin.Y

pMax = projectBounds.PointAt(2)
xMax = pMax.X
yMax = pMax.Y

dx = xMax-xMin
dy = yMax-yMin

xCenter = xMin+dx/2
yCenter = yMin+dy/2

if dx>dy:
    print "wide"
    dyMod = dx*ratio
    yMin = yCenter-dyMod/2
    yMax = yCenter+dyMod/2
else:
    print "high"
    dxMod = dy/ratio
    xMin = xCenter-dxMod/2
    xMax = xCenter+dxMod/2

#Create analysis QGIS project from template
if createAnalysis:
    #open and read template file
    templateFileName = "Analyse_Template_QGIS_3.qgs"
    templatePath = os.path.join(baseProjectPath,templateFileName)
    with open(templatePath, "r") as in_file:
        text = in_file.read()

        #replace placeholders in template
        text = text.replace("{{xMin}}",str(xMin))
        text = text.replace("{{yMin}}",str(yMin))
        text = text.replace("{{xMax}}",str(xMax))
        text = text.replace("{{yMax}}",str(yMax))
        text = text.replace("{{project}}",project)
        text = text.replace("{{scenario}}",scenario)
        text = text.replace("{{scenarioFileName}}",scenarioFileName)
        text = text.replace("{{scenarioLabel}}",scenarioLabel)
        text = text.replace("{{projectLabel}}",projectLabel)

        text = text.replace("{{connectionParameters}}",qgisConnection)

        #save modified template to new file
        outFileName = "Analyse_{0}_{1}.qgs".format(sanitizedProject,sanitizedScenario)
        outFilePath = os.path.join(scenarioPath,outFileName)
        with open(outFilePath, "w") as out_file:
            out_file.write(text)

#Create edit QGIS project from template
if createEdit:
    #open and read template file
    templateFileName = "Projekt_Template_QGIS_3.qgs"
    templatePath = os.path.join(baseProjectPath,templateFileName)
    with open(templatePath, "r") as in_file:
        text = in_file.read()

        #replace placeholders in template
```

```
        text = text.replace("{{xMin}}",str(xMin))
        text = text.replace("{{yMin}}",str(yMin))
        text = text.replace("{{xMax}}",str(xMax))
        text = text.replace("{{yMax}}",str(yMax))
        text = text.replace("{{project}}",project)

        text = text.replace("{{connectionParameters}}",qgisConnection)

        #save modified template to new file
        outFileName = "Projekt_{0}.qgs".format(sanitizedProject)
        outFilePath = os.path.join(projectPath,outFileName)
        with open(outFilePath, "w") as out_file:
            out_file.write(text)

if updateAnalysis or createAnalysis:
    #execute dependent components for generating visuals
    updateVisuals = True
else:
    #do not execute dependent components for generating visuals
    updateVisuals = False
```

B.4.17. Ausführung R Scripts für Auswertungen (Py R Integration)

Der folgende Code führt zwei R Scripts für die Analyse der Ist-Situation und des aktuellen Szenarios aus. Dies geschieht durch das Starten von Subprozessen, die mittels rscript.exe die eigentlichen Scripts ausführen. Der Code für das Starten der Subprozesse basiert auf dem IronPython Cookbook (Foord 2010).

```
import sys
from System.Diagnostics import Process

#Basing on IronPython Cookbook
#http://www.ironpython.info/index.php?title=Launching_Sub-Processes

#"C:/Program Files/R/R-3.4.0/bin"
rPath = rExecutablePath
#"C:/school/UniGIS/Master_Thesis/git/src/r/gebaeude_histogramme.R"
rScriptPath = projectBasePath+"\\gebaeude_histogramme.R"
#"C:/school/UniGIS/Master_Thesis/git/src/r/gebaeude_histogramme_szenario.R"
rScriptPathScenario = projectBasePath+"\\gebaeude_histogramme_szenario.R"

#Function for executing an R Script with arguments
def executeRScript(rScriptPath, arguments):
    proc = Process()

    #set up environment for pricess
    proc.StartInfo.FileName = rPath+"\\rscript.exe"
    proc.StartInfo.Arguments = rScriptPath + " " + arguments
    proc.StartInfo.UseShellExecute = False
    proc.StartInfo.RedirectStandardOutput = True
    proc.StartInfo.RedirectStandardError = True

    #start process and wait for the script to finish
    proc.Start()

    stdout = proc.StandardOutput.ReadToEnd()
    stderr = proc.StandardError.ReadToEnd()
    proc.WaitForExit()

    print stdout
    print stderr

#do not execute depending components
updateDependents = False

if refresh:
    #compile arguments and run script for current situation
    arguments =  "{0} \"{1}\" ".format(project, scenarioPath)
    print arguments
    executeRScript(rScriptPath, arguments)

    #compile arguments and run script for scenario
    argumentsScenario =   "{0} {1} \"{2}\" \"{3}\"   \"(NFP48)\" ".format(project,  scenario,
scenarioPath, scenarioLabel, scenarioFileName)
```

```
    print argumentsScenario
    executeRScript(rScriptPathScenario, argumentsScenario)

    #execute depending components
    updateDependents = True
```

B.4.18. Export 3D Viewport Screenshots (Py Viewport Screenshots)

Der folgende Code exportiert vier Screenshots des Rhino 3D Viewports als PNG Bilddateien.

```
import sys

import Rhino
import Grasshopper as gh
import System.Drawing
import Rhino.Geometry as rg
import os

#do not execute dependent components
updateDependents = False

if refresh:
    #determine target point for camera
    boundingBox = box.BoundingBox
    target = boundingBox.Center

    #set up view directions
    directions = {"n":rg.Vector3d(0.0,0.6,-0.5),
                  "w":rg.Vector3d(-0.6,0.0,-0.5),
                  "s":rg.Vector3d(0.0,-0.6,-0.5),
                  "o":rg.Vector3d(0.6,0.0,-0.5)}

    #parametrize current viewport
    rhDoc = Rhino.RhinoDoc.ActiveDoc
    view = rhDoc.Views.ActiveView.ActiveViewport

    Rhino.RhinoDoc.ActiveDoc.Views.ActiveView.ActiveViewport.WorldAxesVisible = False

    #set a shading type with shadows
    activeViewPortDm                                                                    =
Rhino.RhinoDoc.ActiveDoc.Views.ActiveView.ActiveViewport.DisplayMode.LocalName
    shaded =Rhino.Display.DisplayModeDescription.FindByName("Ghosted")
    Rhino.RhinoDoc.ActiveDoc.Views.ActiveView.ActiveViewport.DisplayMode = shaded

    #prepare paths
    path = scenarioPath

    if exportCurrent:
        fileBaseName = "3D_Ist_"
    else:
        fileBaseName = "3D_Szenario_"+scenarioFileName+"_"

    #save vurrent camera direction for restore
    cameraDirectionOri = rhDoc.Views.ActiveView.ActiveViewport.CameraDirection

    #export a screenshot for each view direction
    for d in directions.Keys:
        fileName = os.path.join(path,fileBaseName+d+".png")
        print fileName
        direction = directions[d]

        #adjust camera direction z-value with parameter input
        direction.Z=z

        #set camera target and direction
        view.SetCameraTarget(target, True)
        view.CameraUp = rg.Vector3d.ZAxis
        view.SetCameraDirection(direction,True)

        #zoom to bounding box
        view.ZoomBoundingBox(boundingBox)

        #save viewport content to disk
```

```
        image                                                                                                =
Rhino.RhinoDoc.ActiveDoc.Views.ActiveView.CaptureToBitmap(System.Drawing.Size(1680,2100),False
, False, False)

        image.Save(fileName)
        image.Dispose()

    #restore former view
    view.SetCameraTarget(target, True)
    view.CameraUp = rg.Vector3d.ZAxis
    view.SetCameraDirection(cameraDirectionOri,True)

    view.ZoomBoundingBox(box.BoundingBox)

    Rhino.RhinoDoc.ActiveDoc.Views.ActiveView.ActiveViewport.WorldAxesVisible = True

    #execute dependent components
    updateDependents = True
```

B.4.19. Integration QGIS Python Script (Py QGIS 3 Integration)

Der nachfolgende Code, der aktuell nicht im eigentlichen Prozess verwendet wird, setzt prototypisch die Integration der Ausführung von QGIS basierten Python Scripts um. Dies erfolgt analog zur R Integration durch Ausführung des Scripts in einem Subprozess. Damit die Integration funktioniert, müssen die Umgebungsvariablen des Subprozesses analog zur Umgebung in der OSGeo4W Umgebung zur Ausführung von QGIS Python Scripts erfolgen. Der gezeigte Code ist auf QGIS 3 ausgerichtet, die Integration von QGIS 2.x erfolgt aber analog.

```
import sys

#Basing on IronPython Cookbook
#http://www.ironpython.info/index.php?title=Launching_Sub-Processes

qpyPath = "C:/Program Files/QGIS 3.0/apps/qgis/python"
scriptPath = "C:/school/UniGIS/Master_Thesis/git/src/qgis/qgis_project_3.py"

from System.Diagnostics import Process

qgisPath="C:/Program Files/QGIS 3.0"
qgisRoot = "C:/Program Files/QGIS 3.0"

proc = Process()

qgisRoot = "C:/Program Files/QGIS 3.0"
qgisName = "qgis"
qgis = qgisRoot+"/apps/"+qgisName
qgisPrefixPath = qgis
pyQgisPath = qgisRoot + "/bin/python3.exe"
proc.StartInfo.FileName = pyQgisPath
proc.StartInfo.Arguments = '"%s"' % scriptPath
proc.StartInfo.UseShellExecute = False
proc.StartInfo.RedirectStandardOutput = True
proc.StartInfo.RedirectStandardError = True

proc.StartInfo.EnvironmentVariables['QGIS_PREFIX_PATH'] = qgisPrefixPath
proc.StartInfo.EnvironmentVariables['GDAL_DATA'] =  qgisRoot + "/share/gdal/"
proc.StartInfo.EnvironmentVariables['PYTHONHOME'] = qgisRoot+"/apps/Python36"
proc.StartInfo.EnvironmentVariables['PYTHONPATH']                                                    =
qgis+"/python;"+qgisRoot+"/apps/Python36/Lib/site-packages"

proc.StartInfo.EnvironmentVariables['PATH']                                                          =
qgisRoot+"/bin;"+qgisRoot+"/apps/Python36/Scripts;"+"C:/WINDOWS/system32;C:/WINDOWS;C:/WINDOWS
/system32/WBem;"+qgisRoot+"/apps/qgis/bin;"+qgisRoot+"/apps/Qt5/bin"
proc.StartInfo.EnvironmentVariables['GDAL_DRIVER_PATH'] =  qgisRoot + "/bin/gdalplugins"
proc.StartInfo.EnvironmentVariables['GEOTIFF_CSV'] =  qgisRoot + "/share/epsg_csv"
proc.StartInfo.EnvironmentVariables['JPEGMEM'] =  "1000000"
proc.StartInfo.EnvironmentVariables['PROJ_LIB'] = qgisRoot + "/share/proj"
proc.StartInfo.EnvironmentVariables['QT_PLUGIN_PATH'] = qgisRoot+"/apps/Qt5/plugins"
proc.StartInfo.EnvironmentVariables['QT_RASTER_CLIP_LIMIT'] = "4096"

proc.Start()

stdout = proc.StandardOutput.ReadToEnd()
```

```
stderr = proc.StandardError.ReadToEnd()
proc.WaitForExit()

print stdout
print stderr
```

B.5. Analyse R

B.5.1. Histogramme Ist-Situation (gebaeude_histogramme.r)

```
##Load Libraries
library(DBI)
library(RODBC)

library(ggplot2)
library(reshape)
library(plyr)

library(grid)
library(gridExtra)

library(stringr)

##General arguments and plot settings
#Arguments (defaults apply when no argument is present)
args = commandArgs(trailingOnly=TRUE)
if (!is.na(args[1])) {
  projectId <- args[1]
} else {
  projectId <- "4"
}
if (!is.na(args[2])) {
  plotPath <- args[2]
} else {
  plotPath <- "C:/school/UniGIS/Master_Thesis/git/src/qgis/"
}
if(grepl("/",plotPath)) {
  if(substr(plotPath, nchar(plotPath),nchar(plotPath))!="/") {
    plotPath = paste(plotPath,"/",sep="")
  }
} else {
  if(substr(plotPath, nchar(plotPath),nchar(plotPath))!="\\") {
    plotPath = paste(plotPath,"\\",sep="")
  }
}

#Plot Settings
plotWidth <- 840
plotHeight <- 1050

#Output of values to screen
projectId
plotPath

##Zoning classes
#Harmonized building zones
bauzonenLabels <- c("11 Wohnzonen","12 Arbeitszonen",
                    "13 Mischzonen","14 Zentrumszonen",
                    "15 Zonen für öffentliche Nutzungen","16 eingeschränkte Bauzonen",
                    "17  Tourismus-  und  Freizeitzonen","18  Verkehrszonen  innerhalb  der
Bauzonen",
                    "19 weitere Bauzonen"
)
bauzonenCodes <- c("Wohnzonen","Arbeitszonen",
                   "Mischzonen","Zentrumszonen",
                   "Zonen für öffentliche Nutzungen","eingeschränkte Bauzonen",
                   "Tourismus-    und    Freizeitzonen","Verkehrszonen    innerhalb    der
Bauzonen","weitere Bauzonen"
)
bauzonenColors <- c(rgb(255,166,0,255,maxColorValue=255),
```

```
                    rgb(204,153,255,255,maxColorValue=255),
                    rgb(244,164,159,255,maxColorValue=255),
                    rgb(217,179,153,255,maxColorValue=255),
                    rgb(140,140,140,255,maxColorValue=255),
                    rgb(128,255,51,255,maxColorValue=255),
                    rgb(255,153,255,255,maxColorValue=255),
                    rgb(204,204,204,255,maxColorValue=255),
                    rgb(224,212,255,255,maxColorValue=255)
)

##Common functions
#Function exporting a plot to PNG and SVG
save_graph_to_disk <- function(graph, path, filename, w, h){
  #Export to PNG
  png(paste(path,filename,".png",sep=""), width = w, height = h)
  print(graph)
  dev.off()

  #Export to SVG
  svg(paste(path,filename,".svg",sep=""), width = w/100, height = h/100, pointsize=12)
  print(graph)
  dev.off()
}

#Function for plotting multiple histograms divided by groups (building zones)
print_bauzonen_histogram <- function(df, var, grp, cds, clrs,  binwidth, scales, title,
xTitle, yTitle, path, filename, w, h) {
  #Calculating summary for plotted variable
  dfsummary <-ddply(df
                    , grp
                    , function(x){
                      mean = mean(x[[var]],na.rm=TRUE)
                      median = median(x[[var]],na.rm=TRUE)
                      sd = sd(x[[var]],na.rm=TRUE)
                      max = max(x[[var]],na.rm=TRUE)
                      min = min(x[[var]],na.rm=TRUE)
                      data.frame(mean, median, sd, min, max)
                    }
  )

  #Plotting the actual graph
  graph <- ggplot(df, aes_string(x=var, fill = grp, color = grp))+
    geom_histogram(alpha=0.8, position="identity", binwidth=binwidth)+
    scale_color_manual(values=clrs[match(levels(df[[grp]]),cds)], guide=FALSE)+
    scale_fill_manual(values=clrs[match(levels(df[[grp]]),cds)], guide=FALSE)+
    facet_wrap( grp, scales = scales, ncol=2)+
    geom_vline(data=dfsummary, aes(xintercept=mean), color="darkgrey")+
    geom_vline(data=dfsummary, aes(xintercept=mean-sd),linetype="dashed", color="darkgrey")+
    geom_vline(data=dfsummary, aes(xintercept=mean+sd),linetype="dashed", color="darkgrey")+
    labs(title = title)+
    xlab(xTitle)+
    ylab(yTitle)

  #Print graph on screen
  print(graph)

  #Export graph to disk
  save_graph_to_disk(graph, path, filename, w, h)
}

##Establish DB Connection
#Local ODBC DSN of the specified name needs to be configured in system level!
ch <- odbcConnect("pg_localhost_dencity")

##Query building data for entire municipality
#Parametrize query string with project ID
queryBuildingsMunicipality <- stringr::str_interp("
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche, sb.building_height,
sb.building_storeys, sb.building_area
FROM ch.are_bauzonen AS bz
LEFT JOIN ch.swissbuildings3d20 AS sb ON ST_Intersects(bz.geom,sb.geom_footprint)
```

```
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
AND sb.building_height<100
--AND sb.objektart IN (
--  'Kapelle','Turm','Kuehlturm','Sakrales Gebaeude','Gebaeude
Einzelhaus','Hochhaus','Hochkamin','Im Bau','Sakraler Turm','Offenes Gebaeude','Treibhaus'
----'Mauer gross','','Unterirdisches Gebaeude','Gebaeude unsichtbar','Bruecke
gedeckt','Lagertank','Verbindungsbruecke','Flugdach','Mauer gross gedeckt','Lueftungsschacht'
--)
")
#Execute query
bdgDataMunicipality <- sqlQuery(ch, queryBuildingsMunicipality)
#Calculate Geschossfläche
bdgDataMunicipality$building_storeyarea <-
bdgDataMunicipality$building_storeys*bdgDataMunicipality$building_area

##Query building data for project area
#Parametrize query string with project ID
queryBuildingsProject <-stringr::str_interp( "
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche, sb.building_height,
sb.building_storeys, sb.building_area
FROM ch.grid
LEFT JOIN ch.are_bauzonen AS bz ON ST_Intersects(grid.geom,bz.geom)
LEFT JOIN ch.swissbuildings3d20 AS sb ON ST_Intersects(bz.geom,sb.geom_footprint)
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
AND sb.building_height<100
--AND sb.objektart IN (
--'Kapelle','Turm','Kuehlturm','Sakrales Gebaeude','Gebaeude
Einzelhaus','Hochhaus','Hochkamin','Im Bau','Sakraler Turm','Offenes Gebaeude','Treibhaus'
----'Mauer gross','','Unterirdisches Gebaeude','Gebaeude unsichtbar','Bruecke
gedeckt','Lagertank','Verbindungsbruecke','Flugdach','Mauer gross gedeckt','Lueftungsschacht'
--)
AND grid.x_koord BETWEEN (SELECT MIN(ST_XMin(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-2000000-200 AND (SELECT MAX(ST_XMAX(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-2000000+150
AND grid.y_koord BETWEEN (SELECT MIN(ST_YMin(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-1000000 -200 AND (SELECT MAX(ST_YMAX(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-1000000+150
")
#Execute query
bdgDataProject <- sqlQuery(ch, queryBuildingsProject)
bdgDataProject$building_storeyarea <-
bdgDataProject$building_storeys*bdgDataProject$building_area

##Query parcel data for entire municipality
#Parametrize query string with project ID
queryParcelsMunicipality <- stringr::str_interp("
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche,
     flaechmass, anrechenbare_gruendstuecksflaeche, gruenflaeche,
building_area, building_volume, building_storey_area,
baumasseziffer, geschossflaechenziffer, gruenflaechenziffer, ueberbauungsziffer
FROM ch.are_bauzonen AS bz
LEFT JOIN be.mopube_grundstueck AS g ON ST_Intersects(bz.geom,g.geom)
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
")
#Execute query
prclDataMunicipality <- sqlQuery(ch, queryParcelsMunicipality)

##Query parcel data for project area
#Parametrize query string with project ID
queryParcelsProject <- stringr::str_interp("
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche,
flaechmass, anrechenbare_gruendstuecksflaeche, gruenflaeche,
building_area, building_volume, building_storey_area,
baumasseziffer, geschossflaechenziffer, gruenflaechenziffer, ueberbauungsziffer
FROM ch.grid
LEFT JOIN ch.are_bauzonen AS bz ON ST_Intersects(grid.geom,bz.geom)
LEFT JOIN be.mopube_grundstueck AS g ON ST_Intersects(bz.geom,g.geom)
```

```
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
AND grid.x_koord BETWEEN (SELECT MIN(ST_XMin(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-2000000-200 AND (SELECT MAX(ST_XMAX(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-2000000+150
AND grid.y_koord BETWEEN (SELECT MIN(ST_YMin(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-1000000 -200 AND (SELECT MAX(ST_YMAX(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-1000000+150
")
#Execute query
prclDataProject <- sqlQuery(ch, queryParcelsProject)


##Modify factors for zoning classes
bdgDataMunicipality$ch_bez_d <-
bauzonenLabels[match(bdgDataMunicipality$ch_bez_d,bauzonenCodes)]
bdgDataMunicipality$ch_bez_d <- factor(bdgDataMunicipality$ch_bez_d,levels=bauzonenLabels)

bdgDataProject$ch_bez_d <- bauzonenLabels[match(bdgDataProject$ch_bez_d,bauzonenCodes)]
bdgDataProject$ch_bez_d <- factor(bdgDataProject$ch_bez_d,levels=bauzonenLabels)

prclDataMunicipality$ch_bez_d <-
bauzonenLabels[match(prclDataMunicipality$ch_bez_d,bauzonenCodes)]
prclDataMunicipality$ch_bez_d <- factor(prclDataMunicipality$ch_bez_d,levels=bauzonenLabels)

prclDataProject$ch_bez_d <- bauzonenLabels[match(prclDataProject$ch_bez_d,bauzonenCodes)]
prclDataProject$ch_bez_d <- factor(prclDataProject$ch_bez_d,levels=bauzonenLabels)

##Plots for building height
print_bauzonen_histogram(bdgDataMunicipality, "building_height", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                         "Ist-Situation: Gebäudehöhe im ganzen Gemeindegebiet nach
Harmonisierten Bauzonen",
                         "Gebäudehöhe in m", "Anzahl Gebäude",
                         plotPath, "Verteilung_Gebäudehöhe_Gemeindegebiet", plotWidth,
plotHeight)
print_bauzonen_histogram(bdgDataProject, "building_height", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                         "Ist-Situation: Gebäudehöhe im Projektgebiet nach Harmonisierten
Bauzonen",
                         "Gebäudehöhe in m", "Anzahl Gebäude",
                         plotPath, "Verteilung_Gebäudehöhe_Projektgebiet", plotWidth,
plotHeight)

##Plots for building storey count
print_bauzonen_histogram(bdgDataMunicipality, "building_storeys", "ch_bez_d", bauzonenLabels,
bauzonenColors, 1, "free_y",
                         "Ist-Situation: Geschosszahl (geschätzt) im ganzen Gemeindegebiet
nach Harmonisierten Bauzonen",
                         "Geschosszahl", "Anzahl Gebäude",
                         plotPath, "Verteilung_Geschosszahl_Gemeindegebiet", plotWidth,
plotHeight)
print_bauzonen_histogram(bdgDataProject, "building_storeys", "ch_bez_d", bauzonenLabels,
bauzonenColors, 1, "free_y",
                         "Ist-Situation: Geschosszahl (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                         "Geschosszahl", "Anzahl Gebäude",
                         plotPath, "Verteilung_Geschosszahl_Projektgebiet", plotWidth,
plotHeight)

##Plots for building storey area
print_bauzonen_histogram(bdgDataMunicipality, "building_storeyarea", "ch_bez_d",
bauzonenLabels, bauzonenColors, 100, "free",
                         "Ist-Situation: Geschossfläche (geschätzt)  im ganzen Gemeindegebiet
nach Harmonisierten Bauzonen",
                         "Geschossfläche in m²", "Anzahl Gebäude",
                         plotPath, "Verteilung_Geschossfläche_Gemeindegebiet", plotWidth,
plotHeight)
print_bauzonen_histogram(bdgDataProject, "building_storeyarea", "ch_bez_d", bauzonenLabels,
bauzonenColors, 100, "free",
```

```
                        "Ist-Situation: Geschossfläche (geschätzt)  im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Geschossfläche in m²", "Anzahl Gebäude",
                        plotPath, "Verteilung_Geschossfläche_Projektgebiet", plotWidth,
plotHeight)

##Plots for parcel ÜZ
print_bauzonen_histogram(prclDataMunicipality, "ueberbauungsziffer", "ch_bez_d",
bauzonenLabels, bauzonenColors, 0.1, "free_y",
                        "Ist-Situation: Überbauungsziffer (geschätzt) im ganzen
Gemeindegebiet nach Harmonisierten Bauzonen",
                        "Überbauungsziffer", "Anzahl Gebäude",
                        plotPath, "Verteilung_Überbauungsziffer_Gemeindegebiet", plotWidth,
plotHeight)
print_bauzonen_histogram(prclDataProject, "ueberbauungsziffer", "ch_bez_d", bauzonenLabels,
bauzonenColors, 0.1, "free_y",
                        "Ist-Situation: Überbauungsziffer (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Überbauungsziffer", "Anzahl Gebäude",
                        plotPath, "Verteilung_Überbauungsziffer_Projektgebiet", plotWidth,
plotHeight)

##Plots for parcel GFZ
print_bauzonen_histogram(prclDataMunicipality, "geschossflaechenziffer", "ch_bez_d",
bauzonenLabels, bauzonenColors, 0.5, "free_y",
                        "Ist-Situation: Geschossflächenziffer (geschätzt) im ganzen
Gemeindegebiet nach Harmonisierten Bauzonen",
                        "Geschossflächenziffer", "Anzahl Gebäude",
                        plotPath, "Verteilung_Geschossflächenziffer_Gemeindegebiet",
plotWidth, plotHeight)
print_bauzonen_histogram(prclDataProject, "geschossflaechenziffer", "ch_bez_d",
bauzonenLabels, bauzonenColors, 0.5, "free_y",
                        "Ist-Situation: Geschossflächenziffer (geschätzt) im Projektgebiet
nach Harmonisierten Bauzonen",
                        "Geschossflächenziffer", "Anzahl Gebäude",
                        plotPath, "Verteilung_Geschossflächenziffer_Projektgebiet",
plotWidth, plotHeight)

##Plots for parcel BMZ
print_bauzonen_histogram(prclDataMunicipality, "baumasseziffer", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                        "Ist-Situation: Baumasseziffer (geschätzt) im ganzen Gemeindegebiet
nach Harmonisierten Bauzonen",
                        "Baumasseziffer", "Anzahl Gebäude",
                        plotPath, "Verteilung_Baumasseziffer_Gemeindegebiet", plotWidth,
plotHeight)
print_bauzonen_histogram(prclDataProject, "baumasseziffer", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                        "Ist-Situation: Baumasseziffer (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Baumasseziffer", "Anzahl Gebäude",
                        plotPath, "Verteilung_Baumasseziffer_Projektgebiet", plotWidth,
plotHeight)

##Closing DB connection
close(ch)
```

B.5.2. Histogramme Szenario (gebaeude_histogramme_szenario.r)

```
##Load Libraries
library(DBI)
library(RODBC)

library(ggplot2)
library(reshape)
library(plyr)

library(grid)
library(gridExtra)

library(stringr)
```

```
##General arguments and plot settings
#Arguments (defaults apply when no argument is present)
args = commandArgs(trailingOnly=TRUE)
if (!is.na(args[1])) {
  projectId <- args[1]
} else {
  projectId <- "4"
}
if (!is.na(args[2])) {
  scenarioId <- args[2]
} else {
  scenarioId <- "7"
}
if (!is.na(args[3])) {
  plotPath <- args[3]
} else {
  plotPath <- "C:/school/UniGIS/Master_Thesis/git/src/qgis/"
}
if (!is.na(args[4])) {
  scenarioName <- args[4]
} else {
  scenarioName <- "7"
}
if (!is.na(args[5])) {
  scenarioFileName <- args[5]
} else {
  scenarioFileName <- scenarioName
}
if(grepl("/",plotPath)) {
  if(substr(plotPath, nchar(plotPath),nchar(plotPath))!="/") {
    plotPath = paste(plotPath,"/",sep="")
  }
} else {
  if(substr(plotPath, nchar(plotPath),nchar(plotPath))!="\\") {
    plotPath = paste(plotPath,"\\",sep="")
  }
}

#Plot settings
plotWidth <- 840
plotHeight <- 1050

#Output of values to screen
projectId
scenarioId
plotPath
scenarioName
scenarioFileName

##Zoning classes
#Model zones
modellLabels <- c("Z1","Z2",
                  "Z3","Z4",
                  "Z5"
)
modellCodes <- c("Z1","Z2",
                 "Z3","Z4",
                 "Z5"
)
modellColors <- c(rgb(27,158,119,255,maxColorValue=255),
                  rgb(217,95,2,255,maxColorValue=255),
                  rgb(117,112,179,255,maxColorValue=255),
                  rgb(231,41,138,255,maxColorValue=255),
                  rgb(102,166,30,255,maxColorValue=255)
)
#Harmonized building zones
bauzonenLabels <- c("11 Wohnzonen","12 Arbeitszonen",
                  "13 Mischzonen","14 Zentrumszonen",
                  "15 Zonen für öffentliche Nutzungen","16 eingeschränkte Bauzonen",
```

```
                    "17 Tourismus- und Freizeitzonen","18 Verkehrszonen innerhalb der
Bauzonen",
                    "19 weitere Bauzonen"
)
bauzonenCodes <- c("Wohnzonen","Arbeitszonen",
                   "Mischzonen","Zentrumszonen",
                   "Zonen für öffentliche Nutzungen","eingeschränkte Bauzonen",
                   "Tourismus- und Freizeitzonen","Verkehrszonen innerhalb der
Bauzonen","weitere Bauzonen"
)
bauzonenColors <- c(rgb(255,166,0,255,maxColorValue=255),
                    rgb(204,153,255,255,maxColorValue=255),
                    rgb(244,164,159,255,maxColorValue=255),
                    rgb(217,179,153,255,maxColorValue=255),
                    rgb(140,140,140,255,maxColorValue=255),
                    rgb(128,255,51,255,maxColorValue=255),
                    rgb(255,153,255,255,maxColorValue=255),
                    rgb(204,204,204,255,maxColorValue=255),
                    rgb(224,212,255,255,maxColorValue=255)
)

#Function exporting a plot to PNG and SVG
save_graph_to_disk <- function(graph, path, filename, w, h){
  #Export to PNG
  png(paste(path,filename,".png",sep=""), width = w, height = h)
  print(graph)
  dev.off()

  #Export to SVG
  svg(paste(path,filename,".svg",sep=""), width = w/100, height = h/100, pointsize=12)
  print(graph)
  dev.off()
}

#Function for plotting multiple histograms divided by groups (zones)
print_bauzonen_histogram <- function(df, var, grp, cds, clrs,  binwidth, scales, title,
xTitle, yTitle, path, filename, w, h) {
  #Calculating summary for plotted variable
  dfsummary <-ddply(df
                    , grp
                    , function(x){
                      mean = mean(x[[var]],na.rm=TRUE)
                      median = median(x[[var]],na.rm=TRUE)
                      sd = sd(x[[var]],na.rm=TRUE)
                      max = max(x[[var]],na.rm=TRUE)
                      min = min(x[[var]],na.rm=TRUE)
                      data.frame(mean, median, sd, min, max)
                    }
  )

  #Plotting the actual graph
  graph <- ggplot(df, aes_string(x=var, fill = grp, color = grp))+
    geom_histogram(alpha=0.8, position="identity", binwidth=binwidth)+
    scale_color_manual(values=clrs[match(levels(df[[grp]]),cds)], guide=FALSE)+
    scale_fill_manual(values=clrs[match(levels(df[[grp]]),cds)], guide=FALSE)+
    facet_wrap( grp, scales = scales, ncol=2)+
    geom_vline(data=dfsummary, aes(xintercept=mean), color="darkgrey")+
    geom_vline(data=dfsummary, aes(xintercept=mean-sd),linetype="dashed", color="darkgrey")+
    geom_vline(data=dfsummary, aes(xintercept=mean+sd),linetype="dashed", color="darkgrey")+
    labs(title = title)+
    xlab(xTitle)+
    ylab(yTitle)

  #Print graph on screen
  print(graph)

  #Export graph to disk
  save_graph_to_disk(graph, path, filename, w, h)
}

##Establish DB Connection
```

```
#Local ODBC DSN of the specified name needs to be configured in system level!
ch <- odbcConnect("pg_localhost_dencity")

##Query building data for entire municipality
#Parametrize query string with project ID
queryBuildingsMunicipality <- stringr::str_interp("
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche, sb.building_height,
sb.building_storeys, sb.building_area
FROM ch.are_bauzonen AS bz
LEFT JOIN ch.swissbuildings3d20 AS sb ON ST_Intersects(bz.geom,sb.geom_footprint)
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
AND sb.building_height<100
AND sb.objektart IN (
'Kapelle','Turm','Kuehlturm','Sakrales Gebaeude','Gebaeude
Einzelhaus','Hochhaus','Hochkamin','Im Bau','Sakraler Turm','Offenes Gebaeude','Treibhaus'
--'Mauer gross','','Unterirdisches Gebaeude','Gebaeude unsichtbar','Bruecke
gedeckt','Lagertank','Verbindungsbruecke','Flugdach','Mauer gross gedeckt','Lueftungsschacht'
)
")
#Execute query
bdgDataMunicipality <- sqlQuery(ch, queryBuildingsMunicipality)
#Calculate Geschossfläche
bdgDataMunicipality$building_storeyarea <-
bdgDataMunicipality$building_storeys*bdgDataMunicipality$building_area

##Query building data for project area
#Parametrize query string with project ID and scenario ID
#Scenario only (not used)
queryBuildingsProject <- stringr::str_interp("
SELECT g.gid, zone_code, anrechenbare_grundstuecksflaeche, hoehe, geschosszahl,
geschossflaeche, nutzer_wohnen,
nutzer_arbeit, anteil_arbeitsnutzung
FROM dencity.parzelle AS p
LEFT JOIN dencity.gebaeude AS g ON ST_Covers(p.geom, g.geom)
WHERE p.pid=${projectId} AND g.sid=${scenarioId}
")
#Parametrize query string with project ID
#Scenario and current combined
queryBuildingsProject <- stringr::str_interp("
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche, sb.building_height,
sb.building_storeys, sb.building_area, sb.building_area*sb.building_storeys AS
building_storeyarea
FROM ch.grid
LEFT JOIN ch.are_bauzonen AS bz ON ST_Intersects(grid.geom,bz.geom)
LEFT JOIN ch.swissbuildings3d20 AS sb ON ST_Intersects(bz.geom,sb.geom_footprint)
LEFT JOIN dencity.parzelle AS p ON ST_Intersects(p.geom,sb.geom_footprint) AND
p.pid=${projectId}
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
AND sb.building_height<100
--AND sb.objektart IN (
--'Kapelle','Turm','Kuehlturm','Sakrales Gebaeude','Gebaeude
Einzelhaus','Hochhaus','Hochkamin','Im Bau','Sakraler Turm','Offenes Gebaeude','Treibhaus'
----'Mauer gross','','Unterirdisches Gebaeude','Gebaeude unsichtbar','Bruecke
gedeckt','Lagertank','Verbindungsbruecke','Flugdach','Mauer gross gedeckt','Lueftungsschacht'
--)
AND grid.x_koord BETWEEN (SELECT MIN(ST_XMin(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-2000000-200 AND (SELECT MAX(ST_XMAX(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-2000000+150
AND grid.y_koord BETWEEN (SELECT MIN(ST_YMin(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-1000000 -200 AND (SELECT MAX(ST_YMAX(geom)) FROM dencity.parzelle WHERE
pid=${projectId})-1000000+150
AND p.gid IS NULL
UNION ALL
SELECT  bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche, hoehe AS building_height,
geschosszahl AS building_storeys, ST_Area(g.geom) AS building_area, geschossflaeche AS
building_storeyarea
FROM ch.are_bauzonen AS bz
LEFT JOIN dencity.gebaeude AS g ON ST_Intersects(bz.geom, g.geom) AND
ST_Within(ST_PointOnSurface(g.geom),bz.geom)
```

```
WHERE g.sid=${scenarioId}
AND bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))
")
#Execute query
bdgDataProject <- sqlQuery(ch, queryBuildingsProject)

##Query parcel data for entire municipality
#Parametrize query string with project ID
queryParcelsMunicipality <- stringr::str_interp("
SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche,
flaechmass, anrechenbare_gruendstuecksflaeche, gruenflaeche,
building_area, building_volume, building_storey_area,
baumasseziffer, geschossflaechenziffer, gruenflaechenziffer, ueberbauungsziffer
FROM ch.are_bauzonen AS bz
LEFT JOIN be.mopube_grundstueck AS g ON ST_Intersects(bz.geom,g.geom)
WHERE bfs_no IN (SELECT bfsnr FROM be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT
ST_Union(geom) FROM dencity.parzelle WHERE pid=${projectId})))"
)
#Execute query
prclDataMunicipality <- sqlQuery(ch, queryParcelsMunicipality)

##Query parcel data for project area
#Parametrize query string with project ID and scenario ID
#Scenario only (not used)
queryParcelsProject <- stringr::str_interp("
SELECT p.gid, p.zone_code, ST_Area(p.geom) AS area, SUM(p.anrechenbare_grundstuecksflaeche) AS
anrechenbare_grundstuecksflaeche

,SUM(g.hoehe*ST_Area(g.geom))/SUM(p.anrechenbare_grundstuecksflaeche) AS baumasseziffer
,SUM(ST_Area(g.geom))/SUM(p.anrechenbare_grundstuecksflaeche) AS ueberbauungsziffer
,SUM(geschossflaeche)/SUM(p.anrechenbare_grundstuecksflaeche) AS geschossflaechenziffer
,(SUM(p.anrechenbare_grundstuecksflaeche)-
SUM(ST_Area(g.geom)))/SUM(p.anrechenbare_grundstuecksflaeche) AS gruenflaechenziffer
FROM dencity.parzelle AS p
LEFT JOIN dencity.gebaeude AS g ON ST_Covers(p.geom, g.geom)
WHERE p.pid=${projectId} AND COALESCE(g.sid,${scenarioId})=${scenarioId}
GROUP BY p.gid, p.zone_code
")
#Parametrize query string with project ID and scenario ID
#Scenario and current combined
queryParcelsProject <- stringr::str_interp("
SELECT
bz.gid, bz.ch_code_hn, bz.ch_bez_d, bz.ch_bez_f, bz.flaeche, p.gid AS ggid
                                        ,ST_Area(p.geom) AS flaechmass,
MIN(p.anrechenbare_grundstuecksflaeche) AS anrechenbare_grundstuecksflaeche
                                        ,(MIN(p.anrechenbare_grundstuecksflaeche)-
SUM(ST_Area(ST_Intersection(g.geom,p.geom)))) AS gruenflaeche
                                        ,SUM(ST_Area(g.geom)) AS building_area,
SUM(g.hoehe* ST_Area(ST_Intersection(g.geom,p.geom))) AS building_volume, SUM(g.hoehe *
ST_Area(ST_Intersection(g.geom,p.geom))) AS building_storey_area
                                        ,SUM(g.hoehe *
ST_Area(ST_Intersection(g.geom,p.geom))) / MIN(p.anrechenbare_grundstuecksflaeche) AS
baumasseziffer
                                        ,SUM(g.geschosszahl *
ST_Area(ST_Intersection(g.geom,p.geom))) / MIN(p.anrechenbare_grundstuecksflaeche) AS
geschossflaechenziffer
                                        ,(MIN(p.anrechenbare_grundstuecksflaeche) -
SUM(ST_Area(ST_Intersection(g.geom,p.geom)))) / MIN(p.anrechenbare_grundstuecksflaeche) AS
gruenflaechenziffer
                                        ,SUM(ST_Area(ST_Intersection(g.geom,p.geom))) /
MIN(p.anrechenbare_grundstuecksflaeche) AS ueberbauungsziffer FROM dencity.parzelle AS p
                                        LEFT JOIN dencity.gebaeude AS g ON
ST_Covers(p.geom, g.geom)
                                        LEFT JOIN ch.are_bauzonen AS bz ON
ST_Intersects(p.geom,bz.geom) AND ST_Within(ST_PointOnSurface(p.geom),bz.geom)
                                        WHERE p.pid=${projectId} AND
COALESCE(g.sid,${scenarioId})=${scenarioId}
                                        GROUP BY p.gid, p.geom,bz.gid, bz.ch_code_hn,
bz.ch_bez_d, bz.ch_bez_f, bz.flaeche
                                        UNION ALL
```

```
                                        SELECT bz.gid, bz.ch_code_hn, bz.ch_bez_d,
bz.ch_bez_f, bz.flaeche, g.gid AS ggid,
                                        flaechmass, anrechenbare_gruendstuecksflaeche,
gruenflaeche,
                                        building_area, building_volume,
building_storey_area,
                                        baumasseziffer, geschossflaechenziffer,
gruenflaechenziffer, ueberbauungsziffer
                                        FROM ch.grid
                                        LEFT JOIN ch.are_bauzonen AS bz ON
ST_Intersects(grid.geom,bz.geom)
                                        LEFT JOIN be.mopube_grundstueck AS g ON
ST_Intersects(bz.geom,g.geom)
                                        LEFT JOIN dencity.parzelle AS p ON
ST_Intersects(p.geom, g.geom) AND p.pid = ${projectId}
                                        WHERE bfs_no IN (SELECT bfsnr FROM
be.mopube_gemeindegrenzen WHERE ST_Intersects(geom,(SELECT ST_Union(geom) FROM
dencity.parzelle WHERE pid=${projectId})))
                                        AND grid.x_koord BETWEEN (SELECT MIN(ST_XMin(geom))
FROM dencity.parzelle WHERE pid=${projectId})-2000000-200 AND (SELECT MAX(ST_XMAX(geom)) FROM
dencity.parzelle WHERE pid=${projectId})-2000000+150
                                        AND grid.y_koord BETWEEN (SELECT MIN(ST_YMin(geom))
FROM dencity.parzelle WHERE pid=${projectId})-1000000 -200 AND (SELECT MAX(ST_YMAX(geom)) FROM
dencity.parzelle WHERE pid=${projectId})-1000000+150
                                        AND p.gid IS NULL

")
#Execute query
prclDataProject <- sqlQuery(ch, queryParcelsProject)

##Modify factors for zoning classes
bdgDataMunicipality$ch_bez_d <-
bauzonenLabels[match(bdgDataMunicipality$ch_bez_d,bauzonenCodes)]
bdgDataMunicipality$ch_bez_d <- factor(bdgDataMunicipality$ch_bez_d,levels=bauzonenLabels)

bdgDataProject$ch_bez_d <- bauzonenLabels[match(bdgDataProject$ch_bez_d,bauzonenCodes)]
bdgDataProject$ch_bez_d <- factor(bdgDataProject$ch_bez_d,levels=bauzonenLabels)

prclDataMunicipality$ch_bez_d <-
bauzonenLabels[match(prclDataMunicipality$ch_bez_d,bauzonenCodes)]
prclDataMunicipality$ch_bez_d <- factor(prclDataMunicipality$ch_bez_d,levels=bauzonenLabels)

prclDataProject$ch_bez_d <- bauzonenLabels[match(prclDataProject$ch_bez_d,bauzonenCodes)]
prclDataProject$ch_bez_d <- factor(prclDataProject$ch_bez_d,levels=bauzonenLabels)


##Plots for building height
print_bauzonen_histogram(bdgDataMunicipality, "building_height", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                         "Ist-Situation: Gebäudehöhe im ganzen Gemeindegebiet nach
Harmonisierten Bauzonen",
                         "Gebäudehöhe in m", "Anzahl Gebäude",
                         plotPath,
paste("Verteilung_Gebäudehöhe_Gemeindegebiet_Szenario_",scenarioFileName,sep=""), plotWidth,
plotHeight)
print_bauzonen_histogram(bdgDataProject, "building_height", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                         "Szenario: Gebäudehöhe im Projektgebiet nach Harmonisierten
Bauzonen",
                         "Gebäudehöhe in m", "Anzahl Gebäude",
                         plotPath,
paste("Verteilung_Gebäudehöhe_Projektgebiet_Szenario_",scenarioFileName,sep=""), plotWidth,
plotHeight)

##Plots for building storey count
print_bauzonen_histogram(bdgDataMunicipality, "building_storeys", "ch_bez_d", bauzonenLabels,
bauzonenColors, 1, "free_y",
                         "Ist-Situation: Geschosszahl (geschätzt) im ganzen Gemeindegebiet
nach Harmonisierten Bauzonen",
                         "Geschosszahl", "Anzahl Gebäude",
```

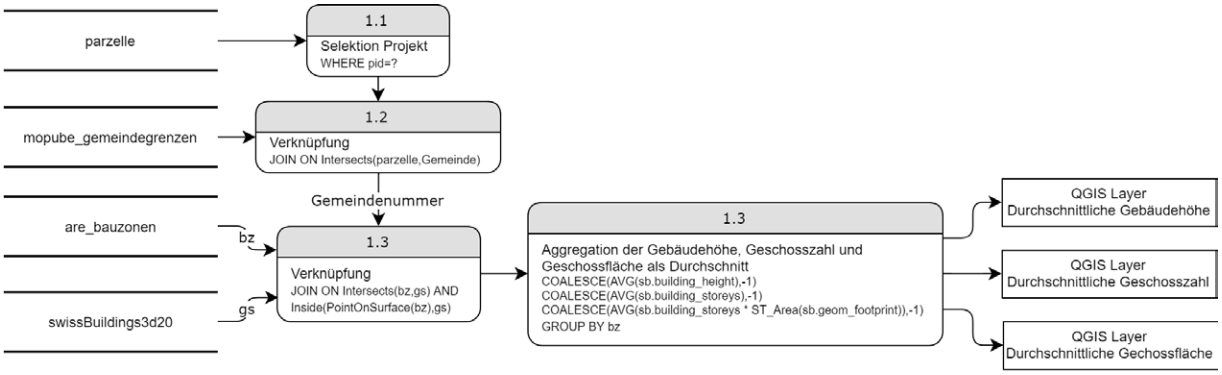
```
                        plotPath,
paste("Verteilung_Geschosszahl_Gemeindegebiet_Szenario_",scenarioFileName,sep=""), plotWidth,
plotHeight)
print_bauzonen_histogram(bdgDataProject, "building_storeys", "ch_bez_d", bauzonenLabels,
bauzonenColors, 1, "free_y",
                        "Szenario: Geschosszahl (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Geschosszahl", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Geschosszahl_Projektgebiet_Szenario_",scenarioFileName,sep=""), plotWidth,
plotHeight)

##Plots for building storey area
print_bauzonen_histogram(bdgDataMunicipality, "building_storeyarea", "ch_bez_d",
bauzonenLabels, bauzonenColors, 100, "free",
                        "Ist-Situation: Geschossfläche (geschätzt)  im ganzen Gemeindegebiet
nach Harmonisierten Bauzonen",
                        "Geschossfläche in m²", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Geschossfläche_Gemeindegebiet_Szenario_",scenarioFileName,sep=""),
plotWidth, plotHeight)
print_bauzonen_histogram(bdgDataProject, "building_storeyarea", "ch_bez_d", bauzonenLabels,
bauzonenColors, 100, "free",
                        "Szenario: Geschossfläche (geschätzt)  im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Geschossfläche in m²", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Geschossfläche_Projektgebiet_Szenario_",scenarioFileName,sep=""), plotWidth,
plotHeight)

##Plots for parcel ÜZ
print_bauzonen_histogram(prclDataMunicipality, "ueberbauungsziffer", "ch_bez_d",
bauzonenLabels, bauzonenColors, 0.1, "free_y",
                        "Ist-Situation: Überbauungsziffer (geschätzt) im ganzen
Gemeindegebiet nach Harmonisierten Bauzonen",
                        "Überbauungsziffer", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Überbauungsziffer_Gemeindegebiet_Szenario_",scenarioFileName,sep=""),
plotWidth, plotHeight)
print_bauzonen_histogram(prclDataProject, "ueberbauungsziffer", "ch_bez_d", bauzonenLabels,
bauzonenColors, 0.1, "free_y",
                        "Szenario: Überbauungsziffer (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Überbauungsziffer", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Überbauungsziffer_Projektgebiet_Szenario_",scenarioFileName,sep=""),
plotWidth, plotHeight)

##Plots for parcel GFZ
print_bauzonen_histogram(prclDataMunicipality, "geschossflaechenziffer", "ch_bez_d",
bauzonenLabels, bauzonenColors, 0.5, "free_y",
                        "Ist-Situation: Geschossflächenziffer (geschätzt) im ganzen
Gemeindegebiet nach Harmonisierten Bauzonen",
                        "Geschossflächenziffer", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Geschossflächenziffer_Gemeindegebiet_Szenario_",scenarioFileName,sep=""),
plotWidth, plotHeight)
print_bauzonen_histogram(prclDataProject, "geschossflaechenziffer", "ch_bez_d",
bauzonenLabels, bauzonenColors, 0.5, "free_y",
                        "Szenario: Geschossflächenziffer (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Geschossflächenziffer", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Geschossflächenziffer_Projektgebiet_Szenario_",scenarioFileName,sep=""),
plotWidth, plotHeight)

##Plots for parcel BMZ
print_bauzonen_histogram(prclDataMunicipality, "baumasseziffer", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                        "Ist-Situation: Baumasseziffer (geschätzt) im ganzen Gemeindegebiet
nach Harmonisierten Bauzonen",
```

```
                        "Baumasseziffer", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Baumasseziffer_Gemeindegebiet_Szenario_",scenarioFileName,sep=""),
plotWidth, plotHeight)
print_bauzonen_histogram(prclDataProject, "baumasseziffer", "ch_bez_d", bauzonenLabels,
bauzonenColors, 2, "free_y",
                        "Szenario: Baumasseziffer (geschätzt) im Projektgebiet nach
Harmonisierten Bauzonen",
                        "Baumasseziffer", "Anzahl Gebäude",
                        plotPath,
paste("Verteilung_Baumasseziffer_Projektgebiet_Szenario_",scenarioFileName,sep=""), plotWidth,
plotHeight)

##Closing DB connection
close(ch)
```

B.6. Analyse QGIS

B.6.1. Ist-Situation: Durchschnittliche Gebäudehöhe, Geschosszahl und Geschossfläche

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der durchschnittlichen Gebäudehöhe, Geschosszahl und Geschossfläche.

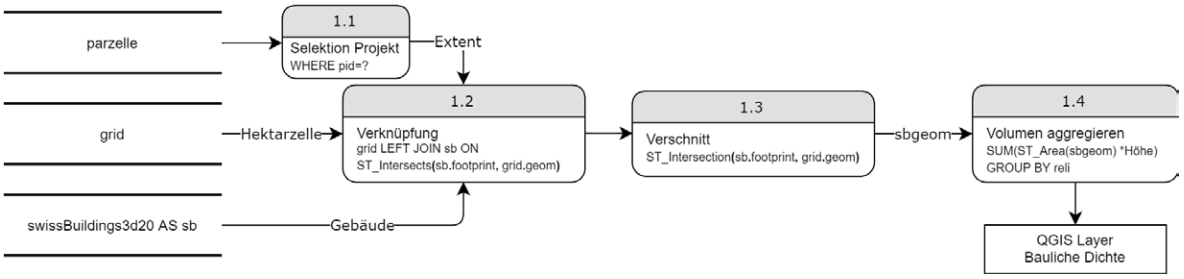


Quelle: eigene Darstellung

Abbildung 59: Datenfluss Ist-Analyse Durchschnittliche Gebäudehöhe, Geschosszahl und Geschossfläche aggregiert auf Bauzonen

B.6.2. Ist-Situation: Bauliche Dichte

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der Baumasse bezogen auf das Hektarraster.

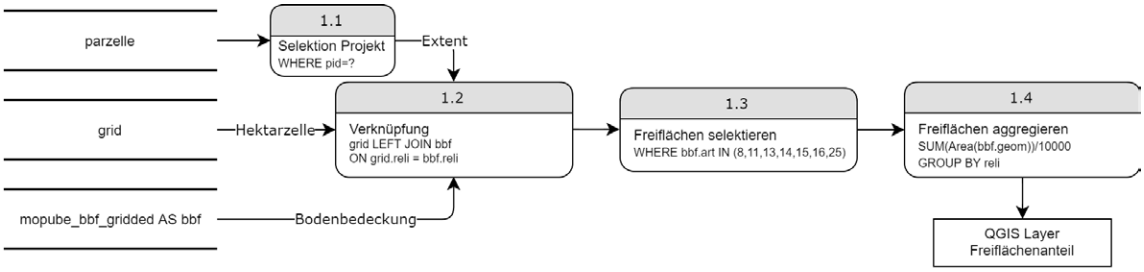


Quelle: eigene Darstellung

Abbildung 60: Datenfluss Ist-Analyse Bauliche Dichte bezogen auf Hektarraster

B.6.3. Ist-Situation: Freiflächenanteil

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung des Freiflächenanteils bezogen auf das Hektarraster.

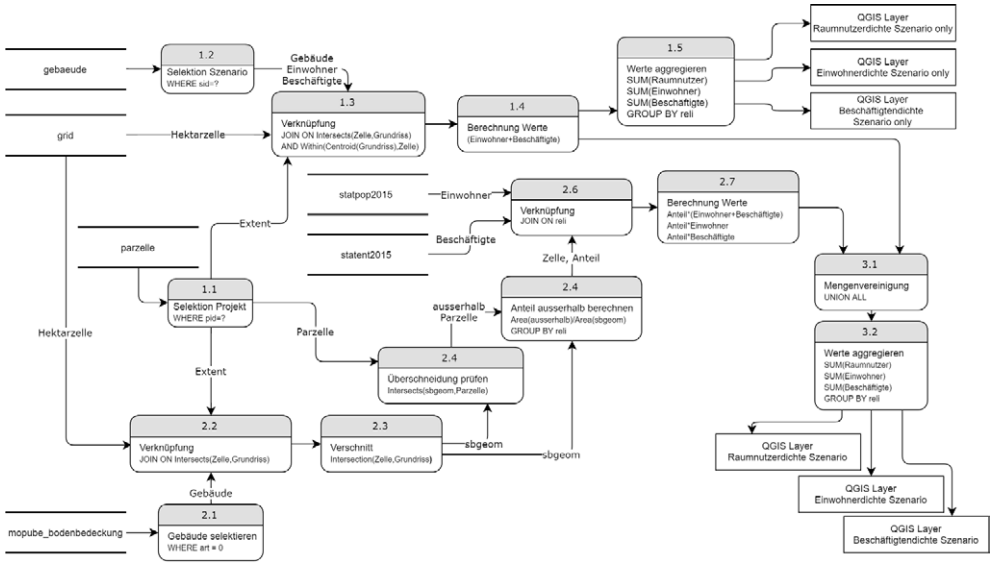


Quelle: eigene Darstellung

Abbildung 61: Datenfluss Ist-Analyse Freiflächenanteil bezogen auf Hektarraster

B.6.4. Ist-Situation: Raumnutzerdichte, Einwohnerdichte, Beschäftigtendichte

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte Freiflächenanteils bezogen auf das Hektarraster.

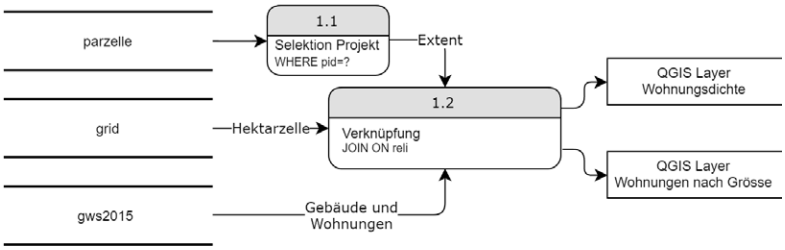


Quelle: eigene Darstellung

Abbildung 62: Datenfluss Ist-Analyse Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte bezogen auf Hektarraster

B.6.5. Ist-Situation: Wohnungsdichte (inkl. Zimmerzahl)

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der Wohnungsdichte und der Verteilung der Wohnungen nach Zimmerzahl bezogen auf das Hektarraster.

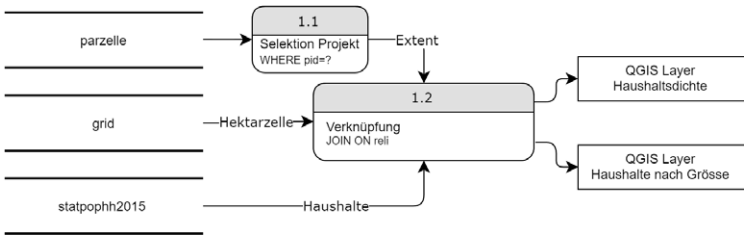


Quelle: eigene Darstellung

Abbildung 63: Datenfluss Ist-Analyse Wohnungsdichte bezogen auf Hektarraster

B.6.6. Ist-Situation: Haushaltsdichte (inkl. Haushaltsgrösse)

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der Haushaltsdichte und der Verteilung der Haushalte nach Grösse bezogen auf das Hektarraster.

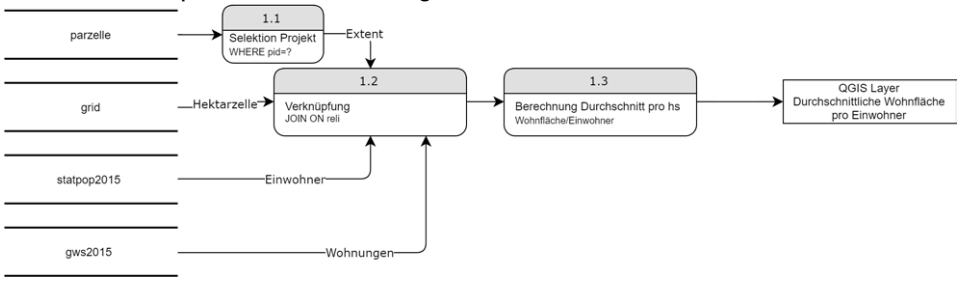


Quelle: eigene Darstellung

Abbildung 64: Datenfluss Ist-Analyse Haushaltsdichte bezogen auf Hektarraster

B.6.7. Ist-Situation: Durchschnittliche Wohnfläche pro Einwohner

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der durchschnittlichen Wohnfläche pro Einwohner bezogen auf das Hektarraster.

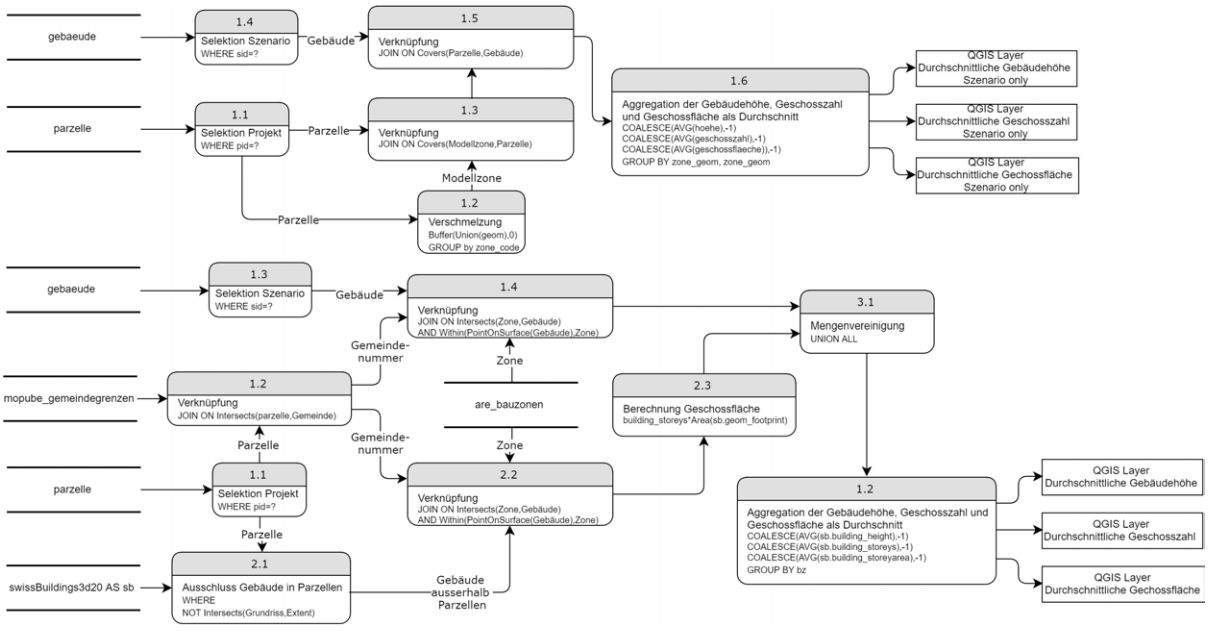


Quelle: eigene Darstellung

Abbildung 65: Datenfluss Ist-Analyse durchschnittliche Wohnfläche pro Einwohner bezogen auf Hektarraster

B.6.8. Szenario: Durchschnittliche Gebäudehöhe, Geschosszahl und Geschossfläche

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der durchschnittlichen Gebäudehöhe, Geschosszahl und Geschossfläche, welche zusammen bestimmt werden. Es ist jeweils ein Datenfluss für die Parzellen des Projektgebietes allein sowie für die Kombination von Projektgebiet und die Ist-Situation vorhanden.

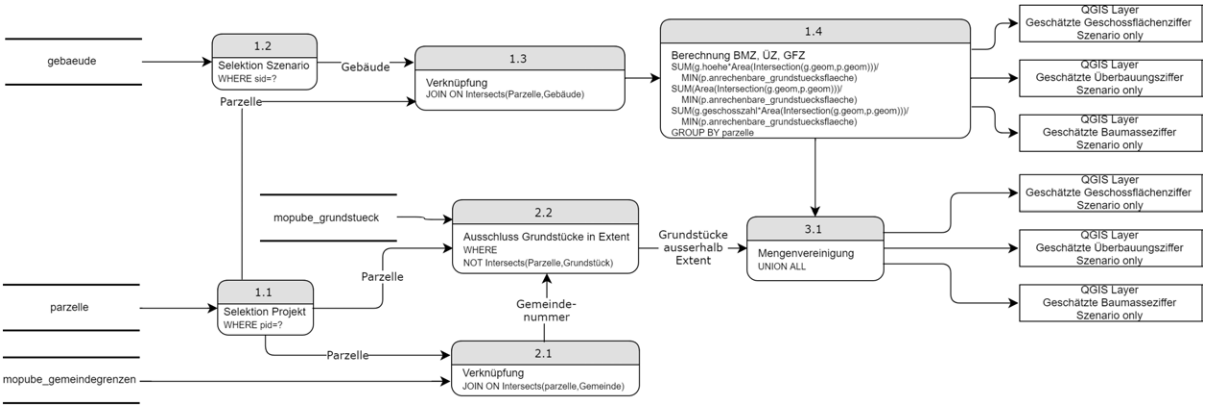


Quelle: eigene Darstellung

Abbildung 66: Datenfluss Szenario Analyse Durchschnittliche Gebäudehöhe, Geschosszahl und Geschossfläche aggregiert auf Bauzonen

B.6.9. Szenario: Geschätzte Geschossflächenziffer, Überbauungsziffer und Baumassenziffer

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der geschätzten Geschossflächenziffer, Überbauungsziffer und Baumassenziffer, welche zusammen bestimmt werden. Es ist jeweils ein Datenfluss für die Parzellen des Projektgebietes allein sowie für die Kombination von Projektgebiet und die Ist-Situation vorhanden.

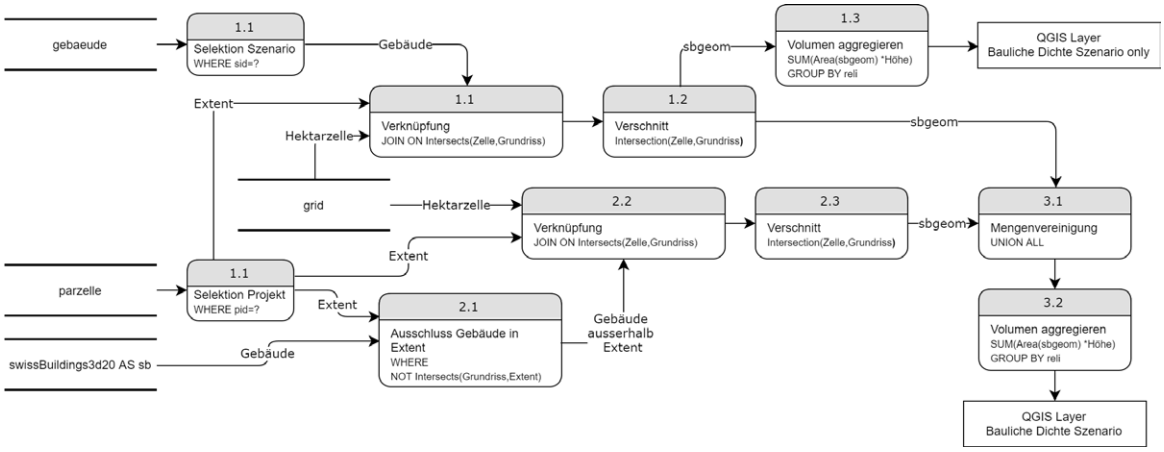


Quelle: eigene Darstellung

Abbildung 67: Datenfluss Szenario Analyse für geschätzte Geschossflächenziffer, Überbauungsziffer und Baumassenziffer bezogen auf Grundstücke

B.6.10. Szenario: Bauliche Dichte

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung des überbauten Volumens bezogen auf die Hektare. Es ist jeweils ein Datenfluss für die Parzellen des Projektgebietes allein sowie für die Kombination von Projektgebiet und die Ist-Situation vorhanden.

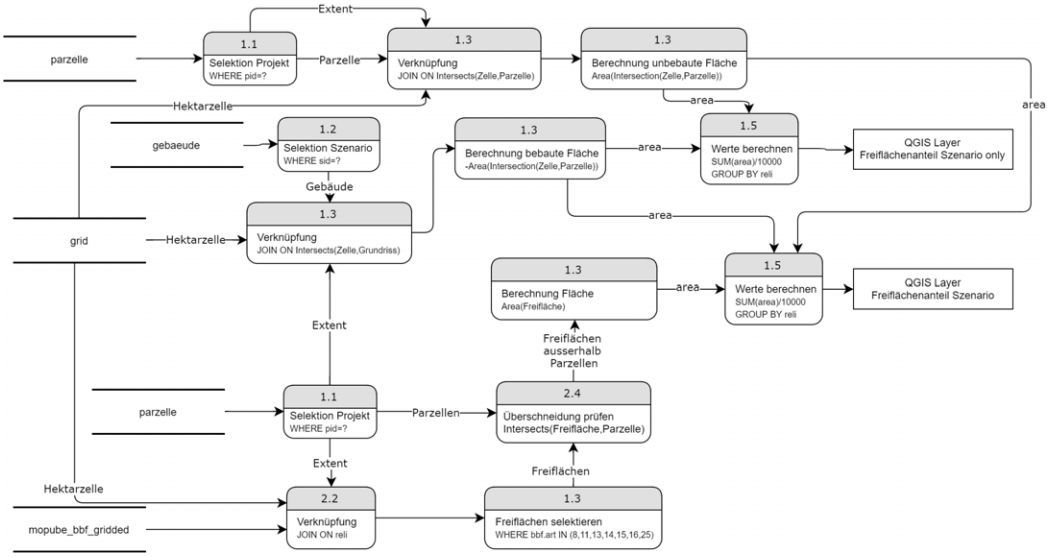


Quelle: eigene Darstellung

Abbildung 68: Datenfluss Szenario Analyse Bauliche Dichte bezogen auf Hektarraster

B.6.11. Szenario: Freiflächenanteil

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung des Freiflächenanteils bezogen auf die Hektare. Es ist jeweils ein Datenfluss für die Parzellen des Projektgebietes allein sowie für die Kombination von Projektgebiet und die Ist-Situation vorhanden.

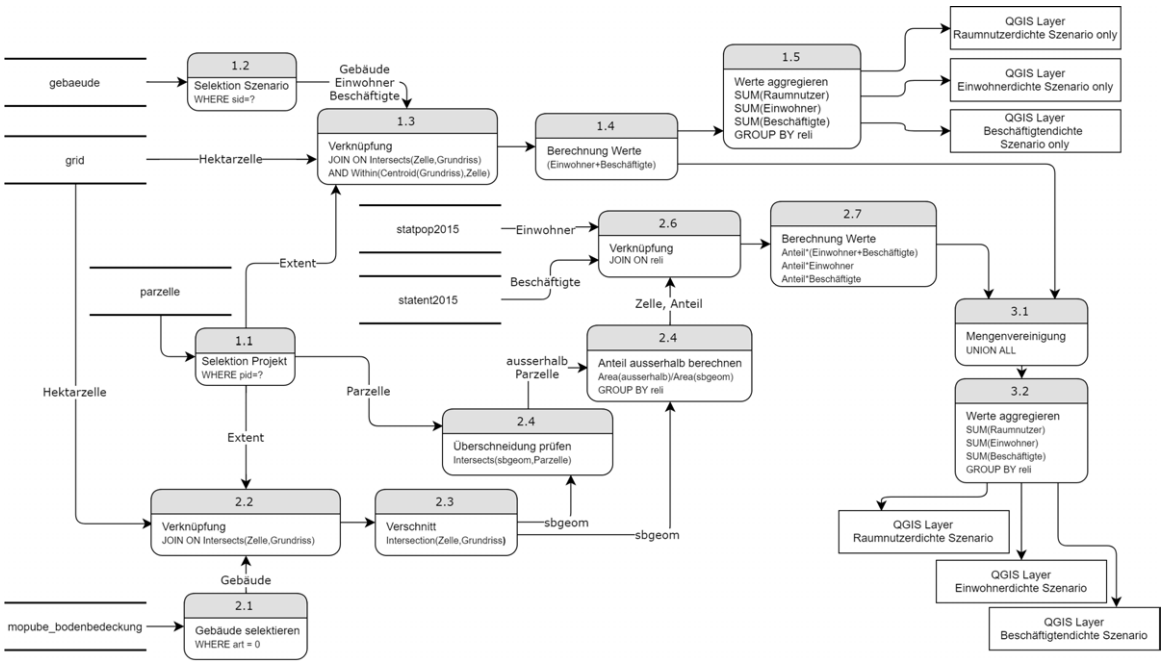


Quelle: eigene Darstellung

Abbildung 69: Datenfluss Szenario Analyse Freiflächenanteil bezogen auf Hektarraster

B.6.12. Szenario: Raumnutzerdichte, Einwohnerdichte, Beschäftigtendichte

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte bezogen auf die Hektare. Es ist jeweils ein Datenfluss für die Parzellen des Projektgebietes allein sowie für die Kombination von Projektgebiet und die Ist-Situation vorhanden.

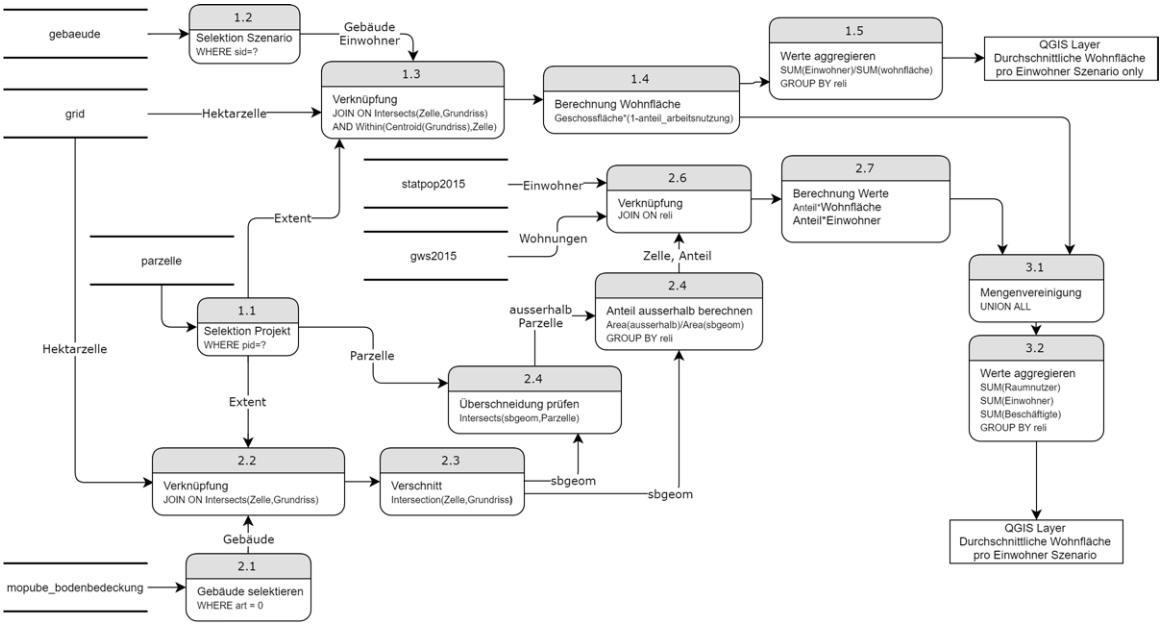


Quelle: eigene Darstellung

Abbildung 70: Datenfluss Szenario Analyse Raumnutzerdichte, Einwohnerdichte und Beschäftigtendichte bezogen auf Hektarraster

B.6.13. Szenario: Durchschnittliche Wohnfläche pro Einwohner

Das folgende Datenflussdiagramm zeigt vereinfacht den Datenfluss für die Auswertung der durchschnittlichen Wohnfläche pro Einwohner bezogen auf die Hektare. Es ist jeweils ein Datenfluss für die Parzellen des Projektgebietes allein sowie für die Kombination von Projektgebiet und die Ist-Situation vorhanden.



Quelle: eigene Darstellung

Abbildung 71: Datenfluss Szenario Analyse durchschnittliche Wohnfläche pro Einwohner bezogen auf Hektarraster