Assemblage Form: An ontology of the urban generic with regard to architecture, computation, and design.

THÈSE N° 6954 (2016)
PRÉSENTÉE LE 13 MAI 2016
À LA FACULTÉ INFORMATIQUE ET COMMUNICATIONS
LABORATOIRE DE DESIGN ET MEDIA (IC/ENAC)
PROGRAMME DOCTORAL EN ARCHITECTURE ET SCIENCES DE LA VILLE

ÉCOLE POLYTECHNIQUE FÉDÉRALE DE LAUSANNE
POUR L'OBTENTION DU GRADE DE DOCTEUR ÈS SCIENCES

PAR

Trevor PATT

acceptée sur proposition du jury:
Prof. F. Kaplan, président du jury
Prof. J. Huang, directeur de thèse
Prof. C. Najle, rapporteur
Prof. M. Carpo, rapporteur
Prof. L. Stalder, rapporteur
ASSEMBLAGE FORM:
An ontology of the urban generic with regard to architecture, computation, and design

Prof. Jeffrey Huang
Laboratoire de Design et Media × Design Lab
École polytechnique fédérale de Lausanne
06 January, 2015

Trevor Ryan Patt
Acknowledgements

I am deeply thankful for having had the opportunity to conduct this work and to pursue such a deep and wide-ranging research through to completion. For this I am especially grateful to Professor Jeffrey Huang for his ongoing support and his encouragement to challenge the conventional form of a thesis and to shape it according to the research instead. As a researcher, the Media x Design Lab has been an outstanding home, and I am especially appreciative of the chance to have integrated my research interests within the design studios that we have taught. I owe a great deal to the students who participated in these studios and would like to thank them for their engagement, their effort, and their insight as I worked through many of the initial stages of this thesis. I would like to also thank my colleagues at the lab who have created a diverse and stimulating context to work in. I am especially indebted to Nathaniel Zuelzke who first introduced me to this program, but also for his unmeasurable impact on my work through our collaboration at MxD and through countless conversations in which he was always deeply engaged, critical, and inspiring.

I would like to thank my jury committee, Profs. Mario Carpo, Ciro Najle, Laurent Stalder, and Frédéric Kaplan for their participation in the defense and the very stimulating discussion that followed. I would also like to thank Profs. Kees Christiaanse, Peter Trummer, and Dieter Dietz for their input prior to the defense. I would not have been able to carry out this research without the financial support of the SNSF and Food Urbanism Initiative and I greatly appreciate the interdisciplinary challenge that the FUI research team provided as an early testing ground for ideas that would eventually find their way into this document. The earliest traces of this thesis took shape in an essay written in 2010, thank you to Esther Choi and Marrikka Trotter instigating that essay and for the zeal you brought to editing it. Thanks also to Prof. Kongjian Yu and Lu Xiaoxuan for facilitating the research visits in China and for providing site data. This work feels, for me, to be intimately linked to every point of my educational history and there are dozens and dozens of others who have shaped or influenced this thesis directly or indirectly; I hope to thank you all personally in the near future.

Finally, I would like to thank my family and especially my parents for always encouraging my studies and never making me feel anything but supported in my work though it has taken me so far from them. And thank you to Rebecca John for all your support, especially in the chaos of the last moments of this thesis' conclusion.
Agencemet des formes:
Une ontologie décrivant l’urbanisme par rapport à l’architecture, le calcul, et design

Résumé

Ce manuscript introduit une théorie de l’urbanisme contemporain qui met en parallèle le caractère transformatif de la ville avec une nouvelle perspective computationnelle de la conception urbaine dans une description intégratrice du concept d’une organisation générique. Cette caractérisation décrit la condition urbaine en tant que question ouverte comme décrite par les théoriciens de l’urbanisme d’agencement (“assemblage urbansim”). Cette thèse étend les travaux existants dans le domaine de l’urbanisme basés en empirisme vers une théorie opérative qui fusionne ces résultats connus avec une ontologie d’une agence basée-objet qui peut être appliqué en pratique pour la conception. Mes travaux défend l’idée que les processus computationnels sont unique dans la façon dont ils facilitent le design urbain de fonctionner de la même manière qu’une ville réagit à son propre dynamisme et sa représentation pendant qu’ils mettentwhile l’accent sur la dimension réthorique de systèmes interactifs comme l’urbanisme et les modèles procéduraux.

Les similitudes entre les modèles computationnels et la dynamisme de la ville seront explorés par une série de sujets d’étude qui s’étendent entre l’étude urbaine et la conception urbaine tout en liant des questions sur l’histoire de la dimension numérique dans l’urbanisme. Les quatre thématiques — Interaction, Génération, Réflexion, et Entropique — convergent autour un schéma théorique intégré. Dans ce cadre-là je défends l’influence croissante de l’architecture en tant qu’agent dans le processus de conception urbaine. Je présente des exemples potentiels à ce sujet obtenus à modèles numeriques à côté des arguments théoriques pour renforcer le fait que les contraintes liées à la codification en langage machine sont des determinants significatifs pour la conception tout comme les réalités matérielles le sont pour l’expérience urbaine.

Enfin les leçons sont ramenés vers l’urbanisme de telle sorte que l’architecture agit en tant qu’interface pour mettre en prise la ville, permettant l’architecture de participer par le biais d’une interaction qui est mutuellement contingent. Dans ces cas, la ville provoque en continu l’architecture d’interroger a son tour le status quo de la situation urbaine et son potentiel changeant. En permettant des résultats indéfinis, l’urbanisme et la conception computationnelle peuvent produire le même genre alterité et de irréalité productif.

Mot de clés
agencement, le design computationnelle, inaesthetiques, la philosophie centrée sur l’objet, la monadologie, l’urbanisme procédural
Abstract

The thesis proposes a projective theory for contemporary urbanism that equates the active processes of the city and a new orientation for procedural urban design within a single line of thought that delineates the concept of a generic organization—forming heterogeneous assemblages, but resisting the tendency to totalizing systematization. Such a description characterizes the urban condition as an open phenomenon, such as that described by theorists of assemblage urbanism. This thesis extends this work from an analytical or empirical theory to an operative one that can be applied to design practice by combining it with an ontology of encapsulation and object-based agency. I will argue that computational processes are unique in the way that they enable urban design to operate in the same manner as the city with regard to enaction and representation while drawing attention to the rhetorical dimension of interactive systems like urbanism and procedural models.

These parallels are further explored through a series of themes that bridge between urban studies and urban design and that connect to historical concerns in computation and urban design. The four themes—Interactive, Generative, Reflexive, Entropic—coalesce around an integrated theoretical schema. Within this framework, I also argue for an increasingly involved role for architecture as an active agent in the urban design process. Illustrations of how this might occur (as functional code and software screenshots) are presented alongside the arguments to underscore the fact that the material basis of the computational model is as significant a determinant of design practice as the material realities of the city are to the urban experience.

Finally, these lessons are imported back into urbanism with architecture acting as an interface to the city. Procedural engagement allows architecture to participate in urbanism through a mutually contingent interaction. In cases where this occurs, the city continuously prompts architecture to carry out new inquiries into the changing potential of the urban situation. Rather, by allowing the the outcome of the situation to remain undecided, both urbanism and computational modeling can be seen to offer the same productive irreality and alterity: an urban generic.

Keywords

assemblage urbanism, computational design, inaesthetics, object-oriented philosophy, monadology, procedural urbanism
CONTENTS:

Acknowledgements
Résumé / Abstract

0.1 Introduction
Influences / Goal / Motivations

0.2 Structure
Schematic / Chapter summary

19

ASSEMBLAGE URBANISM

1.1 Challenges and Shortcomings of Critical Urban Theory
Critical theory as urban theory / The object of critical urbanism is not the city / Reduction of urbanism to epiphenomena / Restriction of agency within structuralist framework

1.2 Aims and Characteristics of Assemblage Urbanism
Assemblage urbanism’s response / The dynamic city / Instantaneously emergent

1.3 Addressing the Organization and Agency of the City without Structure
Against scalar hierarchies / The elements of urbanism / New assemblages

1.4 To Account for a Wider Field of Agency
Alterity / Difference / Enaction and agency / New characteristics

1.5 Tracing the Origins of Assemblage
Deleuze and Guattari / Manuel DeLanda / Bruno Latour

1.6 Common Assemblage Foundations
Inorganic agency / Immaterial objects / Flat ontology

1.7 Responsivity, Assemblage Urbanism, and Engagement with the City
Individual engagement and constituency / Technological agency / Interface for new interactions

33

OBJECT ONTOLOGY

2.1 A Concept for Objects
The expanded object / Assemblage urbanism’s contribution

2.2 Object-Oriented Philosophy
Overview / Being as difference

2.3 Internal Existence
Definitively withdrawn / Characteristics and qualities / Organization and endo-relations

2.4 Exterior Relations
Information passes and is translated between objects / Exo-relations form new assemblages / Causation

2.5 Monadology
Pure Interiority / Extension / Objectile

2.6 Environment
Inclusion / Incompossibility / Construction / Cultivation

43

PROCEDURAL RHETORIC

3.1 Synthesis
Constituent characteristics / Model autonomy / Nonlinear interaction

3.2 Unique Traits of Computation
Formalization and automation / Representation and enaction

3.3 Authorship
Split agency / Manifold agency

3.4 Unit Operations
Configurative systems / Distributed units

3.5 Rhetoric of Simulation
Engagement / Representation and semiotics / Rhetoric of simulation

3.6 Model and Agency
Matters of concern / Agency
INTERACTIVE

4.1 Introduction to Part II
Operative criticism / Design illustrations

4.2 Introduction to Interactive
Rhetorical models / Processes as values / Interactive concerns epistemology / Conclusion

4.3 Significance of Information Interaction
Prepositional mode of being / State space

4.4 Modes of Input
Flatwriter / Individual input / Molar input / Derived values

4.5 Forms of Output
Metadata and datastructure / User feedback / Dataascapes / Customizing the display / Archive

Project 1: MESHES

4.6 Positioning
Differentiation and specification / Adaptive pointcloud / Accumulative information

4.7 Localization and Spatialization
Developing locality / Inclusion

4.8 Instantiation
Procedural identities / Categorizing, distributing, filtering, and sorting

4.9 Concept of Model
Introduction / Types of models / Badiou's purposes regarding models / An inventive praxis / Representation and formalization / Participation

4.10 Conclusion
Projective models / Architecture as urban interface

GENERATIVE

5.1 Generative Ontology
Definition of objects and forms / Means of interaction

5.2 Multiagent Systems
Encapsulation / SEEK / Learning from multiagent systems

5.3 Diffuse agency
Multiagent approximations / Hybrid and distributed agency

Project 2: PARCELS

5.4 Discretizing the ground
Half-edge mesh implementation / Agency of the ground

5.5 Extension
Agent definition / Adjacency behaviors

5.6 Assemblage Emergence
Clustering behaviors / Intensional and extensional sets

5.7 Ecology of Interaction
Observing monads / Recording temporal data

5.8 Enacting Encounters
Virtual / Complexity of urban encounters
REFLEXIVE
6.1 Intensifying Actualization
Actual–virtual axis / Intensities / Reflexity

6.2 Feedback Urbanism
Ville Cybernetique / Participating in urbanism

6.3 Mobile Agents
Tropism and stigmergy / Determinant and emergent planning

6.4 Multi-scalar and Bidirectional Influence
Internal complication / Types and populations

6.5 Architecture as Effectual Urban Input
Architecture as urban effector / Collective form

6.6 Reflexive Model Tendencies
Differentiation engines / Risk of equilibrium

ENTROPIC
7.1 Resisting Terminality
Nondeterministic models / Generator

7.2 Informal Structures
Novel through network excess / Urban morphology / Graph agents

7.3 Graph Functions
Network depth and properties / Nodal properties / Urban networks as design ground

7.4 Nonlinear Dynamics
Agents and embedded mapping / Duration and persistence / Metabolism and catastrophe

7.5 Contingent Identity
Disassembling / Pluripotential
INAESTHETICS
8.1 Urban Inaesthetics
The city thinks itself

8.2 The Open City
The heterogeneity of the urban / The multiplicity of architecture

8.3 Material Enaction
Enaction supersedes the script

8.4 Lines of Flight
Content and action

GENERIC
9.1 The Generic Multiple
Accounting for novelty / Relating the generic to formalization

9.2 The Urban Generic
Locating the generic / Mutual contingency of architecture and urbanism

9.3 Conclusion
Summary of the thesis
§0.1 Introduction

Influences

This thesis gathers up a number of my long-standing interests and preoccupations with the intention to link them together within a single line of thought. Thus while many doctoral theses are hermeneutic in character, developing a new understanding through closely reading and parsing a source, this one has a synthetic nature, bringing together a number of sources and constructing a new category from their assemblage. In particular, I am interested in the agency of architecture on urban design, the extension of its effects over the shaping of the city; the interrelationship of emergent networks and the realist irreducibility of the actors involved, the multiplication of localized behaviors and of informed sites; the rhetorical limits of formalization in such systems; projections of the future potential of the city; and the representation of these potentials. In sum, the thesis concerns how architecture and design can relevantly interact with the complex production of contemporary urbanity.

Goal

Under the name of the ‘urban generic’, I will come to define a theory for urbanism constructed around an integrated practice of computational design and planning. I will argue that computational thinking is uniquely suited to the challenge of a contemporary theory of urbanism. This assertion is not put forward with the intention of reconstructing the urban so that it follows the recognizable forms of facile algorithmic processes, but to rewrite approaches to design computation and urbanism toward their common ground: a procedural network of countless actors and organizations enmeshed in an open-ended interrelationship.

The use of the term ‘generic’ here is not meant to evoke a convergence of sameness as in Koolhaas’ “The Generic City,” but rather the philosophical category of generic as invoked by Alain Badiou to designate the relationship to an untotalisable multiplicity, one that cannot be fully circumscribed within the formal definition of a situation and that “will proceed to reconstruct—locally, to begin with—the whole set of rules by which things appear.” This definition of the generic simultaneously affirms the value of formalization in the approach to urbanism while insisting on the irreducibility of the urban to such formalizations.

Motivations

As the actual experience and management of cities are drawn closer and closer toward the computational in their own logics, this thesis contemplates ways in which computational thinking might also be incorporated into the design of cities—not only as a source of formal language or to satisfy the exigencies of the engineer, but in the production of urban planning and design—as an environment of creative becoming. I am especially interested in

1 Patt, “Taipei 2.0.2: Computation and the Urban Generic.”
2 Zuecke, Patt, and Huang, “Computation as an Ideological Practice.”
3 Maas and La, Skyar City.
4 Patt, “The Collective Image: Form, Figure, and the Future.”
5 Koolhaas, S, M, L, XL. p1248-1265
Badiou, Being and Event (London: Contin, 2006). Meditation 31, p327ff
7 Badiou and Hallward, “Beyond Formalisation: An Interview.” p131
8 Graham, “Software-Sorted Geographies.”
Amin and Thrift, Cities: Reimagining the Urban.
Graham and Marvin, Splintering Urbanism: Networked Infrastructures, Technological Mobilities, and the Urban Condition.
delineating a conceptual framework that acknowledges that the urban environment is acutely receptive to bidirectional influence between the small-scale of architectural objects and users and the large-scale urban concerns of policy and planning. Using procedurality as a medium, I propose a model that is inherently dynamic and interactive, and therefore also emphasizes the mutability of plans. Such dynamic enactment allows the urban designer to make the claim for emergent phenomena that exceed the definition of the design, however a committed engagement with processual planning also requires consideration of processes by which future development is indirectly steered and how those processes can be engaged with by designer and citizen alike. To address these concerns it is also necessary to introduce an ontology of urban plenitude into the model of computational design.

§0.2 Structure

Schematic

The structure of the argument is fourfold. Each section will be addressed twice: once, primarily from the theoretical sources, and a second time along a theme related to technical deployment in a computational environment. In addition, there will be four design examples accompanying each technical theme to illustrate aspects of potential applications. The sections are:

1. Procedural Rhetoric – Interactive (Meshes)
2. Object Ontology – Generative (Parcels)
3. Assemblage Urbanism – Reflexive (Buildings)
4. Inaesthetics – Entropic (Paths)

where the second term names the theme (and the object of the design illustration) and the first term a theoretical source of explication or supporting context. The isolation of themes as well as the separation of teoria from techné is a somewhat artificial one, but one which will allow more focused attention to each aspect of the argument given the diverse set of sources being brought together. When intertwined with the others, each contributes to the understanding of the urban generic (Figure 0.1).

Chapter summary

As a point of entry, chapter 1 will review a current within urban studies, Assemblage Urbanism, which deemphasizes structures of social power in favor of sociomaterial explanations of agency that exist equally for human and non-human actors. In contrast to political–economic structures, assemblages are considered to be especially temporal, even ad hoc, for the purpose of dispensing with the hierarchical framework of structuralist critical urban theories. This chapter argues that assemblage urbanism's method for approaching the city as an entity dynamically constructed from its (socio-)material elements emphasizes the responsivity of the city and enables an expanded role for the agency of the inorganic matter of the city independently at various scales. This actant dynamism is a needed addition to urban design practice for both pragmatic reasons of engagement with temporal changes and ideologically to locate changing urban quality as a more significant aspect of urban design.

Chapter 2 recognizes that the efficacy of agency must be supported by an expanded concept of the object (and assemblages as objects) exploring the loose category of Object-Oriented Philosophy (OOP), a recent trend in philosophical thought which seeks a realist ontology independent of human causation—that is, an eminent object agency—and shares a number of concerns with assemblage urbanism. The object properties developed here will be supplemented by a new reading of monadology, to highlight the mode of existence of discrete units, their internal multiplicity, and their engagement with the exterior (or detachment therefore). Additionally, this section will reference Latour’s reintroduction of Tarde with regard to digital profiles and databases and Deleuze’s explication of Leibniz and baroque complexity.
In order to combine the situational contingency of assemblage urbanism and the internalization of object properties within object oriented philosophy, it is necessary that the urban situation also initiate autonomous behaviors. Ian Bogost’s twin concepts of Unit Operations and Procedural Rhetoric show how such a set of behaviors can produce a system as complex as the city in a non-reductive way, and how procedural representations, like simulation models, stipulate new terms of engagement, drawing on the unique possibilities of autonomous reactions given by computation. As rhetoric, the primary interest of modeling is shifted away from the application of formula toward solving well-delimited scientific problems and refocused on the framing of the cultural potentials of procedurality—that is, to make arguments, to initiate discussions, to persuade constituencies, to structure interactions, to foster engagement, or to suggest possibilities.

In this framework, architecture is situated as an interface of the city, an information threshold that both acts and is acted upon. As an interface, it influences the directions in which identity and information are pointed, though only indirectly. This requires a commitment from the design to the multiplicity of possibilities and the acknowledgment that inputs, whether they are contextual or user generated, are not directly actionable, but always received and translated. Chapter 4 will describe the ‘Interactive’ as an application of procedurality to the task of incorporating external and extrinsic information (into a system, an assemblage, or an object), the ways that information is managed, and what possibilities its dissemination enables or disables. This section will discuss technical methods for data input and user interaction and will use ‘Project 1: Meshes’ as an example of interactive methods for organizing contextual inputs into a surface of urban orientations and oriented building volumes as interfaces for types of urban space on a topographic site in Sichuan.

Occupying a position based on such contingent foundations, motivates a substantial reevaluation of authorship. Chapter 5 proposes a transference of design decisions into objects, giving them agent-like autonomous behaviors. Operating in the absence of centralized coordination, the ‘Generative’ focuses on the encapsulation of internal traits as the identity of an object and the reflection in other units of these external presences. How objects produce extensive series over and across one other leading to emergent orders (including the ability to form larger assemblages which encapsulate new behaviours) and co-implicated networks points the way toward an urban design that does not rely on a single, actor at its head. This chapter will address challenges and potentials that arise from multi-agent systems and will use ‘Project 2: Parcels’ as an example of generative behaviors applied to the elements of an urban mesh and which develop their virtual identities following a complex extension of traits over a site in peri-urban Beijing.

9 Chatterton, “The Urban Impossible: A Eulogy for the Unfinished City.”
10 Complete files will be available to access and download at http://ahtehha.net/assemblage-form/
11 Also sometimes, ‘Speculative Realism’ (SR).
The lack of centralized coordination means that a multi-agent system will frequently encounter conflicts and contradictions among its components. The ‘Reactive’ function negotiates these incompatibilities while maintaining an agonistic environment. Reflexivity concerns the actualization of the urban field into persistent urban actors and emphasizes the diffuse impact of effects that ripple through an interrelated network. These contingent effects are a source of feedback causality that can act as regulating restrictions or spontaneous catalysts. This chapter will touch on how feedback loops can be incorporated within associative geometry using ‘Project 3: Buildings’ as an example of assemblages reconfiguring component elements to form collective ensembles between high-rise towers in urban Shanghai.

Iteration of the reflexive function tends toward solutions in static equilibrium, abandoning the definitive characteristic of the city as an open system. It is necessary to introduce a remedial operation which destabilizes identity and returns the focus to procedures of expansion and adaptation. The ‘Entropic’ is a deterritorializing function which preserves the flexibility and dynamic potential of the system by forgetting or discarding established patterns, structures, or assemblages. Chapter 7 will propose tactics for subverting the tendency of formally explicit computational models toward closed systems and will use ‘Project 4: Paths’ as an example of productive entropy in networked structures to negotiate the metabolism of an urban village in Guangzhou and the disposition of its public spaces.

The idea of forgetting as de-differentiation is treated above as an important condition for preserving the multiplicity within urban design. The proposed model is an environment that would formulate the contested status of its own inherent complexity as a condition to be extended rather than resolved, that is, to insistently define the urban as the condition of the possibility of the City and not as a solution to the city or a statement on the city. Badiou defines ‘Inaesthetics’ as the relationship of philosophy to aesthetics such that philosophy does not dictate the agenda of aesthetics, but reflects on its revelations. In particular, he discusses the possibility of aesthetic media being dedicated not to the singularity of Art, but toward providing the conditions for the possibility of an art, through a process of forgetting that supersedes knowledge or information via enactment. Chapter 8 considers this analogous to the relationship of urbanism and architecture. In this way, the enactment of urban processes can become the means through which they exceed and are not limited to a strict, technocratic definition.

Finally, chapter 9 concludes with a meditation on the ‘Generic.’ As mentioned above, the generic follows Badiou’s adoption of the set-theoretical definition of the generic set by Paul Cohen: the set which cannot be defined through reference to its encompassing set, an untotalizable set, a set which is indiscernible by the situation, which exceeds definition. For Badiou, the generic provides the basis for the possibility of the new. The generic allows one to conceive of formal systems which are not limited to the conditions of their definition but which are capable of open-ended potentiality without compromising their formalization. This thesis draws much of its argument from Badiou’s equation of set theory with ontology, but introduces a positive alternative to the ‘void’ by drawing from the previous discussions of assemblage theory and object oriented ontology forming a new synthesis tuned to the plurality of the urban condition.
This diagram illustrates the schema that relates the various parts of the thesis. The four themes at the center, Interactive, Generative, Reflexive, and Entropic, are related to one another as oppositional terms along the diagonal axes and by translations around the perimeter. Interactivity is primarily concerned with extensities: quantitative metrics that will become the component parts for the objects that follow as they are included in the subsequent objects. Generativity addresses the identity of unique objects, withdrawn and yet overlapping and extending their influence over one another in the formation of assemblages. Reflexivity defines these assemblages through the intensional convergence of various objects into a consistent assemblage whose internal relations settle at a stable limit, at which point Entropy is necessary to disrupt these limits and to force the individual to re-engage the virtual.

In this schematic, we have distinguished emphasis on individual, as a monad, and the multiple as an assemblage as well as the internal constitution of the object and its external relations. However, the ontological equivalence between objects and assemblages (all objects are multiple and each assemblage can be taken as an individual) means that this diagram can be recursively nested: any quadrant can be taken as a new origin point and the diagram repositioned at that anchor (that is to say that the perimeter sequence is not a necessary order, in fact any entity is always acting in many roles at once).

Each theme is linked to a particular object of urbanism (Meshes, Parcels, Buildings, Paths) that will constitute the basis of the design illustrations and that are intended to ground such a metaphysical discussion in the concerns of urban design. Together, the thematic categories and the design illustrations are informed by specific theoretical sources that provide a critical context in urban studies and media presentation, though of course, the text does not support such a discrete, one-to-one relationship as is shown here, but the separations become blurred and overlapping.

The sequence of the text itself is traced by the dashed line beginning with Assemblage Urbanism (chapter 1) in the lower left and concluding with Generic (chapter 9) at the bottom center.

The terminology is largely drawn from: Deleuze, The Fold: Leibniz and the Baroque. Chapter 4: p41-60, especially p 49, 57.
Bibliography:


CHAPTER 1
ASSEMBLAGE URBANISM
§1.1 Challenges and Shortcomings of Critical Urban Theory

Critical theory as urban theory

The tradition of critical urban theory can be defined as an abstraction of the urban organization that is “enabled by and oriented towards” its specific context and the normative practices surrounding it while emphasizing “the disjuncture between the actual and the possible.” The scope of urban studies has expanded in recent years as urbanization has accelerated and cities themselves have come to be defined more and more by their actions in the global sphere. This has stretched the definition of urban studies “across the entire world economy … at all spatial scales and across the entire surface of planetary space.” In particular, it has challenged the relationship of urban studies to critical theory: Brenner argues that urban studies can no longer be treated as a subtopic of applied critical theory, but, because urbanism is the ubiquitous condition in which social, political, and economic relations are organized and enacted, that the two are definitively connected, “that critical theory must necessarily be a critical urban theory.”

The object of critical urbanism is not the city

Already by 1970 Lefebvre had forecast the ubiquity of urbanization and the displacement of industrial production by the production of urban space as a dominant mode of social development. However, by casting the urban as an abstraction of pure form with “no specific content,” Lefebvre devalues the role of the city, itself, in the final analysis. Ignacio Farías argues that “in the case of critical urban studies, the focus on cities and space is only contingent. What is ultimately at stake in those discussions is the organization of contemporary capitalism.” That is to say that critical urban theory is always practiced in service of and bounded by critical theory’s project “to investigate the forms of domination associated with modern capitalism” and “to excavate the emancipatory possibilities that are embedded within, yet simultaneously suppressed by, this very system.” The division between these two targets has only gotten more complicated in the wake of the dematerialization of the city.

Reduction of urbanism to epiphenomena

Thus while urban matters have taken on greater significance within critical theory, any real effects or processes of urbanism are still held as intermediate outcomes that represent tendencies in global political economy. This results in a discourse that ignores or badly investigates a multitude of questions about the city which cannot be reduced to fundamentally political–economic terms. Though Brenner calls for “a much more systematic integration of urban questions into the analytical framework of critical social theory as a whole,” critical urban theory continues to frame these urban questions under familiar categories and structures. While this offers the benefit of established and tested methodology, “the prevailing spatial imaginary behind this tradition of work has been that of

4. Ibid. p206
6. Lefebvre, The Urban Revolution. p1
7. Ibid. p139
8. Prigge, “Reading ‘The Urban Revolution’: Space and Representation.” p48
9. Lefebvre, The Urban Revolution. p118
10. For Lefebvre, “the form of space is defined as the possibility of encounter, assembly, and simultaneous gathering regardless of what—or who—is gathered.”
a theoretical framework that has done little to connect urban theory to ‘grounded’ accounts of everyday occupation of the city drawn from ethnographic or other empirically motivated approaches. Within such a framework the microscale specificities of urban space, public/private interfaces, pedestrian networks and everyday urban experience are often reduced to epiphenomena of larger scale processes and structures. From the perspective of urban specificity, the critical urban approach seems to reverse the order of inquiry, with a fascination for pre-established scales and contexts that overrules the details of individual urban situations: “space and scale as products that somehow become independent from the practices and processes originating them … in the sense of taking for ontologically autonomous something which is rather a quality of actual networks of practices.” At this point, critical urban theory falls back on a weak structuralist ontology to ensure an inroad for political economy in the explanation rather than fully committing to an immanent urbanism.

Restriction of agency within structuralist framework

For the critical urban theorist, the inscription within larger structures defines the roles and agency of actants in the city. These “scalar and spatial fixes” are responsible for providing an agent with its capacity to act or similarly for restricting its action. Two difficulties arise from this model: firstly, the implicit separation of the actant from the motive source of its action; the motivation is now held within the structure and the particular enacting individual is of secondary importance; secondly, the inherent reductionism of such an operation and the limitation of these categorical structures to existing concepts. Not only does this suppress the recognition of unique properties from situation to situation, it discourages the advancement of new configurations. Such reliance on pre-given structures and categories results in relatively narrow definitions of participation and modes of representation that are largely at odds with the contemporary theoretical environment. At the extreme end, this approach assumes “having a privileged access to the real facts, structures and contradictions of urban life,” and suggests “that by unveiling these hidden structures, the strength of the powerful will be combated.” rather than looking for developing alternatives among the undercurrents.

§1.2 Aims and Characteristics of Assemblage Urbanism

Assemblage urbanism’s response

In response to the shortcomings of critical urbanism, there has been a growing trend under the name of ‘assemblage urbanism’, with three primary objectives. The first goal is to work directly with the dynamic variability of contemporary urbanism and to account for the creation of new entities and ad hoc organizations within the city. The second objective is to address these organizations without reducing them to pre-established structures but rather to describe them in ways which follow their contingent formation. Finally, assemblage urbanism aims through these means to identify a broader field of agency for transforming or engaging with the city. This section will review how these goals are characterized as practices among urban theorists and sociologists before then looking at their origins in assemblage theory and defining a particular usage for this study.

The dynamic city

In the last 30 years, cities have demonstrated an accelerated pattern of development that has brought forward a number of new formations to complicate the theorization of the city: incredibly rapid change and growth, massive informal urbanization, divergence between economic models formal and practiced, buy-to-leave property investors, automated technological controls, interdependent service networks—to name a few. Such changes have reiterated the importance of a processual conception of the urban as a mode of becoming. “Rather than focusing on cities as resultant formations, assemblage thinking is interested in emergence and process, and in multiple temporalities and possibilities.” For assemblage urbanists, this means “focusing on the dynamic and transactional unit formed by an organism-in-its-environment” in
which the acts of dwelling make and unmake the city. In place of a well-defined, bounded totality, the city is redefined as a locus of high connectivity between a multitude of different entities, one which is not without its own historical and spatial contingencies, but that is continuously redefined by processes of interrelation.

In highlighting the dynamism of the city, assemblage urbanism emphasizes the fact that the persistencies of the city do not possess privileged control over the trajectory of development, but are simply co-participants, allowing more open projection concerning the potential of the future of the city to be imagined differently.

Instantaneously emergent

In fact, the temporality of assemblages is specifically tuned to recognize “the capacity of events to disrupt patterns, generate new encounters with people and objects, and invent new connections and ways of inhabiting everyday urban life.” These moments of emergence engage existing organizations in a continuous process of renegotiation, territorialization, and adaptation. Thus when one speaks of an assemblage it is always in the sense of an entity in an ongoing act of assembling itself. “Such territorialization is as much an alignment of connections as a hardening of boundaries,” writes Dovey, pointing to the action of aligning rather than the end product of a boundary. The territory of an assemblage is not a given property around which walls can be drawn, but a topological domain that waxes and wanes. Territorialization (and deterritorialization) thus become interesting indicators of situational activity within an assemblage.

§1.3 Addressing the Organization and Agency of the City without Structure

Against scalar hierarchies

One of the more controversial aspects of assemblage urbanism is the prioritization of a topological space and the perception that this dispenses with scalar differences. We have just seen how the assemblage concept of context is radically differentiated compared to critical urban theory in order to eliminate the reductivism of top-down definition. The introduction of scale as a categorical qualifier is another place where structuralism creeps back in. Rather than follow this sort of tree-like thinking, which is anyway antithetical to urban organization, the intention is to look for “tactics and strategies of power embedded in the morphology of the city and the ways that an assemblage of small-scale adaptations can produce synergistic emergent effects at higher levels.” Horizontalité, however, does not preclude the existence of different scales, nor does it force all influence to come from the bottom-up. Much more simply, it “provides a progressive basis for a critical revaluation of spatial categories and scalar dynamics” by diffusing powerful top-down assemblages within the same kind of imminent assemblages as the everyday encounter.
The elements of urbanism

While we can speak of the city as a vast assemblage then, this does not confer a simple, organic totality to its persistent parts. Various aspects of urbanism (bureaucratic planning organizations, climatic patterns, aspects of the city’s built form, etc…) may follow individual paths into multiple assemblages with all manner of agendas. Because the elements of an assemblage maintain their individuality separate from the larger networks they form, the theorist must account “for all actual entities involved in such processes of construction, whether human or nonhuman, their interactions and transformations.” Each element is itself an assemblage and is defined by its emergence and not from outside. The emergence of an urban environment is neither exclusive nor reductive and the assemblage urbanist assays to convey the complexity of this inclusiveness. Ash Amin describes the urban spatiality as “a subtle folding together of the distant and the proximate, the virtual and the material, presence and absence, flow and stasis, into a single ontological plane upon which location—a place on the map—has come to be relationally and topologically defined.”

New assemblages

Two points in Amin’s description are particularly significant. First, the localization of the urban—each assemblage actively situates itself through its interactions, locality is not an inherent property given by a global structure but a relational construction. Secondly, the plurality of participants and the many modes of engagement allow the preservation of subtle differences within the assemblage, differences that individualize the assemblage. Though not all such idiosyncrasies will be significant, they leave openings for new, ad hoc engagements in the future or wrinkles in the assemblage’s own development. Unpredictabilities like this preserve the potentiality of the urban assemblage preventing it from collapsing into a set of pre-established possibilities.

§1.4 To Account for a Wider Field of Agency

Alterity

Although new alternatives are a significant feature of critical urbanism’s attention—one of the key elements of critical theory (according to Brenner) is to emphasize “the disjuncture between the actual and the possible”—the inscription of alterity within the existing political-economic structures restricts the theorist to the realm of given possibilities. “We still find in much critical theory the negative use of the term power as oppression (power over) rather than power as capacity (power to),” writes Dovey, underscoring how assemblage urbanism focuses on a broader conception of alterity as an active, rather than reactive, process. This follows directly from the act of tracing the actual processes and actions, investigating which are significant and what would have occurred if they had been enacted otherwise. The assemblage method is one of progressive differentiation, not constrained to a particular model but promoting a rich multiplicity.

Difference

Assemblage urbanism does not only operate by differentiation of one state into another, but also through the openings created by the heterogeneous differences embodied within an assemblage. These differences characterize the behaviors of assemblages as nonlinear interactions illustrative of “the transformative potential of multiplicity and experimentation emerging through often irresolvable differences.” In this construction, both the agency of the individual elements and the interactive whole are preserved as distinct, “where the agency of both can change over time and through interactions.” It is significant that although the formation of the assemblage is “a form of integration, where different elements become aligned in the production of particular effects,” it does not subsume the identity of the part to the will of the new whole, but is set in motion by the tensions that arise between them.
Enaction and agency

In fact, the transformation itself becomes the object of attention in assemblage urbanism’s redefinition of power from consolidated resource to the distributed effects of agency. “Agency is thus an emergent capacity of assemblages … it is the action or the force that leads to one particular enactment of the city.” Even more radically, one can say that “agency in this reading is less an attribute or property and more a name for the ongoing reconfiguring of the world.”

This definition bears some similarity to de Certeau’s conception of the pedestrian acts of urbanism that have replaced the need for representation with action, however the assemblage position goes further by relocating all agentic acts, whether quotidian or operative, in the realm of enactment. Thus there are not two kinds of agency (imposed power and enacted resistance), but one, albeit a heterogeneous one, insinuated across the entire field of operation.

New characteristics

As a result of the twofold displacement—casting assemblage causality as nonlinear interaction and distributing agency everywhere—the assemblage urbanist is forced to adopt new approaches with more emphasis on inquiry than critique. “In most cases, it is practically impossible to know in advance the definitive list of human and nonhuman actors involved, affected or concerned, the scope of their networks or their actual relationships,” and as a result, the urbanist must also be open to a wider array of possible objects and the new potential forms of agency they imply. Of particular interest is the suggestion of an increased compatibility and engagement with urban design and planning as projective, inquisitive practices. These practices can be profoundly enriched by considering the way that agency is constructed and transformed through connections between people and their environment or between urban processes and constructed space in ways that exceed a simple subject–object relationship. Although there is considerable debate concerning the extent of the repercussions of assemblage theory on urban thinking, the reformulation of these relationships away from any external structure impel not only a methodological break from critical urbanism but an ontological break as well.

§1.5 Tracing the Origins of Assemblage

Deleuze and Guattari

The theoretical foundations of assemblage theory originate from two primary sources, the early writings of Deleuze and Guattari, and the development of Actor-Network Theory following Bruno Latour, though neither used the term in the way it is employed now. The Deleuze–Guattarian current comes by way of the English translation of ‘agencement’ as ‘assemblage’ in Foss and Patton’s translation and the subsequent adoption of this term by later translators. The assemblage for Deleuze and Guattari offered an alternative to the dialectical method, which holds apart content-

38 Ibid. p369
39 Amin, “Re-Thinking the Urban Social.” p103
40 “new and unpredictable directions develop when assemblages encounter novel perturbations. This is a conception of causality that seeks to depart with linearity and to make room for novelty and randomness in emergence. Here, randomness may emerge from multiple sources, such as the volatility of initial conditions, unexpected changes in external environments or the chance relations that emerge as differential properties of existing parts are brought into the assemblage”
42 “Possibility is a variation implicit in what a thing can be said to be when it is on target. Potential is the immanence of a thing to its still indeterminate variation, under way” Massumi, Parables for the Virtual: p9, see also p134-137
44 Dovey, “Uprooting Critical Urbanism.”
45 McFarlane, “Assemblage and Critical Urbanism.” p211
47 “these multiple enactments or multiple becomings are not understood as discontinuous, even contradictory and mutually exclusive.”
48 McFarlane, “Decentering the Object of Urban Studies.” p14
49 McFarlane, “Assemblage and Critical Urbanism.” p211
50 Ibid. p208
52 McFarlane, “Assemblage and Critical Urbanism.” p218
53 “neither author nor spectator, shaped out of fragments of trajectories and alterations of spaces: in relation to representations, it remains daily and indefinitely other.” de Certeau, The Practice of Everyday Life. p93
54 Dovey, “Uprooting Critical Urbanism.” p349
56 “Such thinking connects disparate threads of current urban theory as it opens new modes of multi-scalar and multi-disciplinary research geared to urban design and planning practices and therefore to potentials for urban transformation.” Dovey, “Uprooting Critical Urbanism.” p347
matter from form-expression through a bizarre structural causation. Rather, for Deleuze and Guattari, content and expression “can be abstracted from each other only in a very relative way because they are two sides of a single assemblage” and neither can one be subordinated as the object of another for “these relations between forces take place ... within the very tissue of the assemblages they produce.” This is a significant detail because it points to two defining aspects in the interpretation of *agencement* that risk being lost in the translation. The first is that the assemblage corresponds to notions of becoming insofar as the assemblage cannot be reduced to the elements composing it but rather exists as the event of their co-participation. In light of this, Phillips warns against the tendency to describe assemblage states as statements, disjoining the subject from its enunciation as though the assemblage is separate from the temporal sense of its formation. The second is the act of arrangement that comes from within the assemblage, “an active force of becoming or a will expressed equally by and through individuals,” which, in addition to the arrangement of entities, also instills the assemblage with an agency. These twinned concepts are summarized concisely in Braun’s gloss of *agencement* as “capacity to act with the coming together of things.”

**Manuel DeLanda**

The most rigorous development of assemblage theory in this direction comes from Manuel DeLanda who delves into the framework of assemblage interactions. For DeLanda “the ontology of assemblages is flat since it contains nothing but differently scaled individual singularities.” Thus all relations of belonging, whether of an individual to a population, or an entity to an organization, are considered equally to produce new individuals and not new types. A flat ontology is supported by the Deleuzian position that entities are not defined by the assemblages they participate in. Such “relations of interiority,” where the relations constitute the defining properties of elements—characteristic of organic holism—are contrasted to “relations of exteriority,” in which defining properties and “capacity to interact” are separable, allowing individuals to enter or exit assemblages “without the terms changing.” This leaves the formation of assemblages contingent on development embedded in the temporal dimension rather than dictated by abstract necessity. The historical generation of the assemblage prompts empirical investigation because “there is no way to tell in advance in what way a given entity may affect or be affected by innumerable other entities.” The full capacities of an assemblage may go unexercised and are ultimately unknowable, suggesting a redundancy of causality. Graham Harman sees a difficulty in the fact that assemblages “withhold themselves from their relations with the outer world insofar as they are never fully actualized, and withhold themselves from their own pieces by exceeding those parts and forming a new reality,” arguing that such withdrawal also separates the assemblage from its generative process. His concern is that as an assemblage crystallizes into a particular pattern of existence, the accidents of its formation are no longer differentiating features. Harman forgets, though, the potential for undiscovered capacities to distinguish the assembly in ways that reanimate previously redundant history.

DeLanda is careful to distinguish assemblage ontology from an atomism that only enables causation from the bottom up. He insists on the coexistence of differently scaled assemblages and points out that, though always composed of smaller entities, assemblages are most often composed by larger ones often as effects of other expressive or territorializing actions. Naturally, assemblages are also capable of interacting with one another as well. The implication of this (because the interaction must revolve around some relation of exteriority, and because relations of exteriority constitute assemblages by definition) is that relations always produce new assemblages, however briefly. This excessive access to the novel is, perhaps, the most revelatory strength of DeLanda’s assemblage ontology and it helps to return the active, agentic capacity to a description that at times risks becoming overly rigid and schematic. Furthermore, the ties to the material–expressive axis of the assemblage further cement the social character of the assemblage among all those entities which in reality come together to effect the staging or execution of an event.
Figures 1.1; 1.2; 1.3

The construction process in Xiaozhoucun, Guangzhou. First the site is cleared, the existing building is dismantled and the bricks cleaned of mortar. On a cleared plot, new foundations are poured. Because of the narrowness of the passageways, construction equipment must be compact, often a cement truck will wait at the village gate and material will be carried to the site by hand in wheelbarrows. Concrete frame construction enables the site to be rebuilt at higher densities than the previously existing structures.
Bruno Latour

The other primary strand of thinking in assemblage theory follows Bruno Latour’s description of Actor-Network Theory. Latour is similarly motivated by a desire to remove the over-arching frameworks that reduce the specificity of the case at hand in favor of situated non-linear interactions between actors. “when a force manipulates another, it does not mean that it is a cause generating effects; it can also be an occasion for other things to start acting.” The actor is induced to act, but the diverse ways in which the agency of the action is figured (in connection with many actors) is highly contingent. Though the actor-network branch has produced some of the more ontologically radical assemblage urbanists, Latour’s development of Actor-Network Theory is much more methodological—primarily emphasizing how a researcher or theorist can remain committed to a flat ontological grounding—while remaining intentionally quiet about existence within this ontology. He gives almost no description of groups themselves, offering only the performative definition that the assembling (or disassembling) of groups is itself the mapping of social context and that the groups do not exist outside of this action. Latour sees no inherent difference between this action and the acts of a researcher, concluding that “the network does not designate a thing out there… It is nothing more than an indicator of the quality of a text about the topics at hand.” What does exist for Latour is nothing if not concrete, however these actants “are fully relational in character, with no distinction between object and accident, object and relation or object and quality … to change one’s relations is to change one’s reality.” Again assemblage theory promotes a prodigious new production of entities, but through the complete inverse of what we find in DeLanda. Furthermore, it means that there is no place for withheld properties, all actants are, in this case, defined by their efficacy.

Despite these contradictions, actor-network theory does illuminate some relational aspects which can be incorporated with the approach adopted in this text, in particular, the production of scale through the construction of linkages or the action of an individual. Even more promising is the potential reversibility of scale relations, by which an individual can incorporate larger assemblages that it may even belong to itself. These topics will return in the next chapter to establish how the reserve complexity of objects prevents a reduction to merely relations and how the effectuality of relations establishes the social interplay that generates difference and preserves the agency of actors.

§1.6 Common Assemblage Foundations

Inorganic agency

From the various interpretations or applications of assemblage thinking discussed above, there are a few common elements which provide a foundation for the following chapters. The first is a recognition of inorganic assemblages and the increased emphasis given to them in the mapping of urban milieux. As Amin argues, “technology, things, infrastructure, matter in general, should be seen as intrinsic elements of human being, part and parcel of the urban ‘social’, rather than as a domain apart with negligible or extrinsic influence on the modes of being human.” Thus, inquiry cannot be limited to or explained away by interpersonal interaction alone, but is distributed across the social and the material. One description “would summarize ‘sociomateriality’ as things in their mediating role.” In fact, this is not necessarily a project which breaks from the critical urban tradition, conventional critical urbanism can operate through this lens and has occasionally done so well. Assemblage urbanism pushes this concept further, suggesting that things mediate among themselves in addition to mediating human experience. In fact, inorganic material assemblages constitute an intersubjective field by virtue of “the efficacy of objects in excess of the human meanings, designs, or purposes they express or serve.” Acknowledging and including this excess enables the consideration of the urban from a more ecological sensibility and elevates the significance of the built environment in its particular configurations.

78 Latour, "Irreductions."

79 Latour, *Reassembling the Social: An Introduction to Actor-Network-Theory.* p60

80 Ibid. p53

81 Latour provides the best description of this much later in a summary passage: "not by transporting a force that would remain the same throughout as some sort of faithful intermediary, but by generating transformations manifested by the many unexpected events triggered in the other mediators that follow them along the line."

Ibid. p107

82 Farías, "The Politics of Urban Assemblages."

83 "the word 'group' is so empty that it sets neither the size nor the content. It could be applied to a planet as well as to an individual."

Latour, *Reassembling the Social: An Introduction to Actor-Network-Theory.* p29

84 Ibid. p32, 35

85 Ibid. p129

86 Harman, *Prince of Networks.* p104

87 Latour, "Irreductions." p158

88 "Scale is the actor’s own achievement"

Latour, *Reassembling the Social: An Introduction to Actor-Network-Theory.* p185

89 Law, "And If the Global Were Small and Non-Coherent? Method, Complexity and the Baroque."

90 cf §2.6 'Environment'

91 McFarlane, "Assemblage and Critical Urbanism." p216-217

92 Amin, "Collective Culture and Urban Public Space." p8

93 Angelo stipulates three components of this definition: "(1) physically different material referents with different properties; (2) as containing and constitutive of both subjective (mental) and objective (material) dimensions of social life; and (3) as physical mediation between individuals and large scale structures in the urban everyday."

Angelo, "Hard-Wired Experience: Sociomateriality and the Urban Everyday." p570

94 Reeh, *Ornaments of the Metropolis: Siegfried Kracauer and Modern Urban Culture.*

95 Bennett, *Vibrant Matter: A Political Ecology of Things.* p20

Figures 1.4; 1.5

New concrete frame construction receives infill of bricks like those reclaimed in Figure 1.1

Other *ad hoc* constructions on the city—whether electrical, plumbing, or carpentry—make equally apparent the role of material agency in the informal urbanism of the Village-in-the-City.
Immaterial objects

Additionally, the inclusion of object excess can be extended to immaterial objects—the virtual, the potential, the anticipatory. This element can be credited in part to the potential within DeLanda’s assemblage theory for capacities (and therefore possibly entire assemblages) to go unexercised if they do not encounter the right interactions or detection without being any less real. That assemblages operate “between the possible—the unstable flows of materials and substances—and the prescribed—the imposition of functional, stable structures,” allows assemblage urbanists to propose new points of articulation that support new ways for “urban histories and everyday life to be imagined and put to work differently” before these forms have fully materialized, as a generative critique that to some extent prefigures the emergence of its object.

On a more prosaic level, the acknowledgment of immaterial objects also extends to the shaping role of the technological strata of the city, the software protocols, which, of course depend on material assemblages to be enacted, but which can be changed and reoriented without modification of their material expression.

Flat ontology

Finally, as mentioned above, the idealization of a flat ontology in which no object is more fundamental than another is a common concept shared by all forms of assemblage urbanism. Characteristics of flat ontologies, such as the distinct multiplicity of different layers and scales, the exclusion of expressions that would reduce one to a side-effect of another, and the replacement of inherited, or unchanging, structural relationships for ones that are emergent and constructed have been discussed already. In the context of the last paragraphs, it is also worth mentioning that these characteristics apply equally to the relationships between humans and the world. In a flat ontology this relation is neither “a form of metaphysical relation different in kind from other relations between objects” nor does it treat the “subject-object relation as implicitly included in every form of object–object relation.” The flatness in such an approach thus enables one to speak of human–object and object–object (as well as object–human) relations without having to impose a global hierarchy between them as separate modes.

§1.7 Responsivity, Assemblage Urbanism, and Engagement with the City

Individual engagement and constituency

The first motivation for enlisting an assemblage urbanism approach is to begin from a model that encourages and valorizes engagement with the city by individuals or groups, by emphasizing the new forms of participation enacted on a continuous basis, and by “countering politically paralyzing pictures of the unity of capital with notions of a more open social field.” So far, we have considered how assemblage urbanism recasts spatial and scalar contexts as effects of assemblage production, how distributed agency destabilizes linear causality and the identity of belonging, and how a flat ontology supports an ecological understanding of the urban through both human and non-human actants. Ultimately, this culminates in a model which cannot discount any assembling action as a source of potential change, but complements these actions with the ability to trace or project possible effects through pragmatic means. Assemblage urbanism pushes us toward an agonistic pluralism expressed by diverse constituencies, but also constructs diverse public places.
Technological agency

In addition, there is a new technological agency, which enforces boundaries, regulates rhythms, and supports infrastructures within the city. Assemblage urbanism enables the city to be thought processually alongside them and to encounter "other powerful social effects of the urban technological unconscious." Thus, we increase the number of connections between large technical systems while also opening up their functions from within black boxes to illustrate points of articulation and precariousness. Rather than monolithic, controlling entities, technological agency can be located in a prepositional mode of being which "set up what comes next without impinging in the least on what is actually said." Though this study will not go into topics of real-time sensing and controls, in later sections we will develop this parallel between the procedural enactment of the city and the procedural enactment of code.

Interface for new interactions

As an actively constructive theory, assemblage urbanism, helps mitigate the danger of translating an analytic method into a generative one. The projective nature implicit in assemblage theory, of "linking the actual with potential," makes it more welcoming to design input and the very real impacts which small-scale modifications can bring about. Finally, assemblage urbanism establishes the beginnings of a framework that perpetuates new urban interactions: a reflexive operation by which such impacts feed back into the urban field as immanent causes. Within the urban field, each individual assemblage also functions as an interface for potential engagement.
Bibliography:


§2.1 A Concept for Objects

The expanded object

The conceptual advantages of utilizing assemblage urbanism are strongest and most apparent when the urban situation under observation involves surprising or unexpected combinations of objects, actions, and scales. As argued above, preserving the complexity of these situations requires an approach that examines the reality of their entanglements instead of reducing difficult concerns to explanations that conform with established structures. The second motivation introduced was to account for the activity of the city, or the agency to act upon it that issues from non-human sources or without human intervention—automatic technological protocols as well as the more quotidian interactions of matter and objects—in order to further advance an ecological definition of the city.

In order to fully describe the city in this way, one also requires a more detailed theory of objects that expands on what assemblage theory provides. Assemblage urbanism is always able to compare against empirical observation to fill in any conceptual gaps, and so tends to produce definitions of assemblages which do not impinge on their objects of observation. This thesis is interested in projective or speculative design practice, anticipating that the designer’s empirical research will be less thorough and that objects or object properties may also need to be invented or assumed. In these cases, a stronger conceptual model of how assemblages function and behave as objects is necessary.

Assemblage urbanism’s contribution

In the first chapter, we spoke mostly of how assemblage urbanism reframes the primary aspects of urban studies without spending much time on the description of assemblages themselves. The use of assemblage theory by urbanists primarily as a methodological (and not an ontological) explanation has allowed the notion of the assemblage to remain indistinct. In fact, assemblage urbanists actively promote a less circumscribed invocation as a resistance of pluralism against a dogmatic enforcement of ‘proper’ use.

From assemblage theory, objects can be characterized as heterogeneous assemblages that cannot be reduced to their parts and whose component elements similarly are not subsumed within the totality of the object. Furthermore, being an element in a larger assemblage does not render an object inferior to the larger assemblage or prevent it from interacting with it directly and independently nor is the larger assemblage considered derivative of its parts. All objects are considered ontologically equal on a flat plane of existence regardless of their size or scale, their complexity or their simplicity.
Expanding on this base, there are three aspects to consider about the assemblage as an object which will be explored through some of the recent philosophical work on object-oriented ontology and developed further by a new reading of monadology that it enables. These three points are the interior existence of objects, the external relations of objects to others and to their environments, and the connection between objects and causation.

§2.2 Object-Oriented Philosophy

Overview

As the most active branch of what has been named ‘Speculative Realism’, object-oriented philosophy shares the realist7 and anti-reductionist8 positions of assemblage urbanism. Perhaps the best summary of the objective of object-oriented philosophy is the desire “to think a subjectless object, or an object that is for-itself rather than an object that is an opposing pole before or in front of a subject … an object for-itself that isn’t an object for the gaze of a subject, representation, or a cultural discourse.”9 Naturally, this attitude of “a universe made up of objects wrapped in objects wrapped in objects wrapped in objects”10 lends itself well to a flat ontology like the one proposed by Manuel DeLanda11 and covered in chapter one.12 This can be seen in one of the core problematics of object-oriented philosophy, the question of access to objects:

“On the other hand, where the anti-realists have obsessively focused on a single gap between humans and objects, endlessly revolving around the manner in which objects are inaccessible to representation, object-oriented philosophy allows us to pluralize this gap, treating it not as a unique or privileged peculiarity of humans, but as true of all relations between objects whether they involve humans or not. In short, the difference between humans and other objects is not a difference in kind, but a difference in degree.”13

Here, Levi Bryant lays out two establishing operations. First, the multiplication of significant relations from only those that involve human interpretation to include the entire field of inter-object relations,14 and second, the leveling of all these relations into the same register in order to preserve their specific characters.15 Though aimed at the nature–culture divide in particular, this excerpt combats the argument of a world constructed by human experience and intentionality, more generally. By widening the scope of relations to include every sort of relata, object-oriented philosophy calls attention to the huge array of contingent relations that accompany an assemblage. It is not that objects are defined by how they appear to humans, but even strictly cultural objects rely on and involve inorganic objects: “collectives involving humans are always entangled with all sorts of nonhumans without which such collectives could not exist.”16

Being as difference

Nor are objects defined by their relations with one another as a general case, rather object-oriented ontology asserts the more pragmaticist definition that “to be is to make or produce differences” or that “there is no difference that does not make a difference.”17 Though this seems at first glance to produce a contradiction—being-as-difference must surely be relational, mustn’t it?—we draw here from Deleuze’s description of difference-in-itself18 that distinguishes itself from the ground yet without the ground also performing a reciprocal distinction. If “all things equally exist, yet they do not exist equally,”19 then the question one must answer is how to describe the asymmetry between difference-in-itself and the extrinsic differences between two objects. Deleuze holds that “extensity does not account for the individuations which occur within it.”20 Thus in the following we sketch an outline of the object: first addressing the characteristic of extensive qualities; then defining the role of intensities, or endo-relations, distinct from actualization; and finally unpacking the implications for relations with external objects and the environment. In all this, the goal is to preserve the equal ontological status of individuals and to avoid reductionist arguments that would smooth away the tension between assemblages and their (equally individual) parts.21
§2.3 Internal Existence

Definitively withdrawn

From the principle of redundant causality we know that “within open systems or entanglements of objects, the powers of discrete objects are often veiled or inactive.”22 If these extensive entanglements were given the power of defining the identity of objects, they would never be capable of asserting clear independence, always remaining muddled, unable to identify whether an object exists or not, only able to suggest possible objects.23 Furthermore, this would be akin to the ground differentiating from the object and would contradict the difference-in-itself. Thus, “objects must also be thought in terms of their endo-relations or their inter-ontic structure as radically independent of their exo-relations or their inter-ontic relations.”24 The endo-relations of an assemblage encapsulated within the object form an interior existence which is never entirely accessible to an external object. This is considered a definitive property of objects: “there are no objects characterized by full presence or actuality. Withdrawal is not an accidental feature of objects … but is a constitutive feature of all objects regardless of whether they relate to other objects.”25

Characteristics and qualities

Those aspects of an object which are not withdrawn but are accessible and relate to other objects—its qualities—are freed from carrying the responsibly to define identity. Classical concepts of substance had difficulty splitting qualities from objects because there weren’t any additional differentiations beyond the object’s qualities to individuate it,26 but in an object-oriented ontology, “objects are not identical to their qualities but are rather the ground of qualities.”27 Qualities are no longer the building blocks or quanta of being, but actualizations of the object. “Objects can be fully concrete without locally manifesting themselves or actualizing themselves in qualities … Local manifestation is something that objects can do, but an object that does not locally manifest itself is not lacking in some way, nor is it somehow incomplete.”28 As such, extensities are not constrained to formal or necessary roles in the object, but can follow diverse potential behaviors.29 It is more appropriate, therefore, to think of an object’s extensive qualities not “as something an object possesses, has, or is, but rather as acts, verbs, or something that an object does.”30 Qualities can be responsive to the idiosyncrasies of their contexts—both internal and external—in ways that properties of identity would resist.31 Perceived from the point of view of two separate external relations, an object can even enact contradictory or incomposable qualities based on the properties “that emerge as a result of the manner in which the object relates to other objects.”22
Organization and endo-relations

Having peeled away the qualitative dimension, there remains the internal structure of the object. Object-oriented ontology contends that “objects are not merely aggregates of other objects, but have an irreducible internal structure of their own.” This topological organization is not common for all or even a group of assemblages, which would suggest an organization based on a shared predicate, nor is it fixed and immutable. In fact, the intensive is caught up in a transitional immediacy of a relation to its own indeterminancy. “Withdrawn into an all-encompassing relation with what it will be. It is in becoming, absorbed in occupying its field of potential” That is, while any transition within an assemblage’s internal relations transforms the field of potential emergent properties, it still remains “an operationally closed object that relates to the sub-multiples of which it is composed or the multiples that it composes only in terms of its own internal organization” and “cannot be determinately indexed to anything outside itself.” Mereologically, the object’s internal being still remains independent from any assemblages it might be a part of and even those out of which it is composed. It can be properly said that this independence exceeds everything that can be known about the object through its relations.

This nonqualitative structure follows Deleuze’s concept of the virtual; however, Bryant critiques Deleuze’s insistence that the virtual is pre-individual, arguing that “the virtual is not something that produces the individual, but rather must strictly be a dimension of the individual.” This is done to preserve the agency of the object in causal interactions and to locate production as an act of the individual rather than the individual as the residue of production, harking back to the assertion earlier that to be is to make or produce difference. If objects are to exist, they do so as differentiation engines. Bryant refers to this virtuality of the individual as the ‘virtual proper being’.

§2.4 Exterior Relations

Because objects cannot be reduced upwards into controlling structures nor downward atomistically into their parts, neither is there an a priori global container, “There is no world … that connects things together. All such connections must be emergent properties of the objects themselves.” The internal withdrawal of objects and the location of their potency within the virtual clearly complicates the ways by which such connections are able to form. To remain consistent with the ontological formation advanced thus far, any possible relation forgoes direct contact: it must derive from the individual object and issue from its own agency. Bryant proposes that exo-relations can be characterized as translations of information—with the understanding that “information is thus not something that exists in the world independent of the systems that ‘experience’ it, but is rather constituted by the systems that ‘experience’ it … Information is, as it were, a genuine event that befalls a substance or happens to a substance.”

A few significant aspects of this concept are worth detailing. First, the information of a relation does not have its own, separate being, but is enacted by the emitting object as a property, a quality, or an event and received by the second object in an act of sensing or perceiving: “information is object-specific, whereas the same perturbation can affect a variety of different objects while producing very different information for each object.” Relations are highly sensitive to the affective capability of objects. Second, because these information-events are translated into being only through the apperception of various objects, there is no ‘original meaning’ or ‘pure interpretation’. Third, in a nicely symmetrical moment, this fact holds true even for the originating object, from which the quality is a self-othering event. We can confirm this by following Deleuze’s argument that the virtual does not in any way resemble the actual. Taken together, these points prevent the the relation-as-information from devolving into mere simulacra, thinking it instead “as force-signs of deterritorialization and of reterritorialization.”
Exo-relations form new assemblages

Meanwhile, objects are always joining together to form larger assemblages. In fact, Harman has written that “when two objects enter into genuine relation, even if they do not permanently fuse together, they generate a reality that has all of the features that we require of an object … they create something that has not existed before, and which is truly one.”\(^{34}\) However, this would effectively reduce all relations to endo-relations and the assertion that “there are properties of objects that emerge as a result of the manner in which the object relates to other objects”\(^{35}\) would have to be modified to acknowledge that those properties occur only as the result of the top-down influence of an encompassing assemblage. Though we want to facilitate the production of new objects as much as possible, we will hold off from extending objecthood to such an extent, preferring to leave open the possibility of horizontal relations between objects that remain merely relations. Even stable patterns of relations should be permitted without automatically conflating the relationship with a new object.\(^{36}\) Rather, a new object occurs “when exo-relations among other objects manage to attain operational closure such that their aggregate or multiple composition becomes capable of encountering perturbations as information in terms of their own endo-consistency.”\(^{57}\) While this qualification risks being misinterpreted as saying that all objects are strictly defined intentionally, by a shared property or predicate, the earlier specification of the virtual proper being as a dimension subsequent to the individual, allows objects to be defined extensionally by naming or enumeration as well.

Causation

The problems of external relation and the formation of new assemblages both raise questions about causality, or, the efficacy of assemblages. Timothy Morton seemingly implies that the withdrawal of objects away from one another produces a “disturbing illusory play of causality,”\(^{58}\) “This would be an understandable position if one focuses on the interior being within an object as a split that divides ‘vertically from the implicate to the explicate.’”\(^{59}\) Through this lens it would appear that the virtual within an object that is acted upon constructs effects that are detached from their source. Rather, returning to the dictum that “difference is an activity … existence is thought as a sort of doing or movement,”\(^{60}\) it is apparent that such a reading confuses the agency of the object with its reception. “No object can transfer a force to another object without that force being transformed in some way or another,”\(^{61}\) but this does not mean that the force is not exerted or that the transference is only an illusion. The agency of an object is measured by its effecting of the world, a process that is always messy and complexly negotiated, not by distilling the legibility of intent away from any interferences.

---

30 Bryant, The Democracy of Objects. p89
31 Ibid. p69
32 Bryant, “The Ontic Principle: Outline of an Object-Oriented Ontology.” p272
33 Bryant, The Democracy of Objects. p214
34 Badoua, Being and Event. p44
35 Massumi, Parables for the Virtual. p5
36 Bryant, The Democracy of Objects. p218
37 Massumi, Parables for the Virtual. p7
38 Bryant, The Democracy of Objects. p31
39 Bogost, Alien Phenomenology: Or, What It’s Like to Be a Thing. p30
40 Deleuze, Difference and Repetition. p208
41 Bryant, The Democracy of Objects. p103
42 Ibid. p100-103
43 Ibid. p88
44 Morton, “Objects as Temporary Autonomous Zones.” p151
45 “systems or substances only relate to themselves. Put differently, while substances can enter into exo-relations with other substances, they only do so on their own terms and with respect to their own organization.”
46 Bryant, The Democracy of Objects. p147
47 Ibid. p155
48 Bryant, “The Ontic Principle: Outline of an Object-Oriented Ontology.” p274
49 “no object is capable of representing another object or of functioning as a pure carrier of the perturbations issued from another object. This is because objects always transform or translate perturbations.”
50 “information differentially links an object to itself in a relation between the withdrawn virtual proper being of the object and its local manifestations … information affects a self-othering in the object whereby the virtual dimension of the object simultaneously withdraws and a quality is produced.”
51 Ibid. p156
52 Contra Badoua’s argument about the actual as simulacrum
53 Badoua, Deleuze: The Clasour of Being. p43-53
54 Allou, Diagram 3000 ['Wordl]/Diagramm 3000 ['Worte]. p13
55 Harman, Guerilla Metaphysics: Phenomenology and the Carpentry of Things. p85
56 Bryant, “The Ontic Principle: Outline of an Object-Oriented Ontology.” p272
57 Bryant, The Democracy of Objects. p218
58 Ibid. p273
59 Morton, “Objects as Temporary Autonomous Zones.” p153
60 Bryant, The Democracy of Objects. p102
61 Bryant, The Democracy of Objects. p175
In fact, the opposite is true. If there were a medium or metalanguage by which information were transferred without alteration, there would be no action—that is, no difference—remaining in the act itself but only within the mediator. Just as all connections emerge from the objects themselves, so too are communications produced through patterns of encounters, based on “the records of actions antecedent in the production of consequents” and are perpetually challenged by new divergences. Neither is causality effected by the machinery of an underlying structure, but it manifests as a phenomenon that emanates from and repositions objects in new spatio-temporal contexts.

§2.5 Monadology

Pure interiority

Deleuze describes Leibniz’s concept of the monad in similar language as “the autonomy of the inside” and “a unity that envelops a multiplicity, this multiplicity developing the One in the manner of a series.” As a pure interiority, the monad is withdrawn from direct connections, which “must retain the distinction of its details and its own individuality in the hierarchy in which it enters.” Again, there is agreement that local manifestations, or ‘inclinations’, of the monad are non-necessary traits that occur as “an act, a movement, a change, and not the state.” Inclinations are qualitative but not attributable, “the predicate is above all a relation and an event, and not an attribute,” therefore predicates do not ground the monad, they are included in the monad. More than anything else, Deleuze forcefully emphasizes this point. It is established in the very first sentence of his text on monadology referring “not to an essence but rather to an operative function,” a detail adamantly repeated throughout his analysis: “Inflection is the event that happens to the line or to the point;” “not defined by an attribute, but by predicates-as-events;” “the spontaneity of manners replaces the essentiality of the attribute.”

Extension

As these events proliferate, they gain series of entanglements with other objects whereby they extend into or over one another. The way the monad includes its predicates determines a harmonization of the monad with the adjacent and component objects, with regard to the production of the world around it. The monad’s withdrawn virtual being—or its intrinsic singularities, to use Deleuze’s term—generates the events that include relations within the object. Meanwhile there is a second, reflexive operation which directs these series of inclinations inward toward convergence as intensities. As such, the propagation of relations is not a constriction of the object but a continuous prolongation with regard to the world it engages. “Even compressed, folded, and enveloped, elements are powers that enlarge and distend the world.” In the same way, architecture extends into a frame that “itself becomes detached from the inside, and establishes relations with the surroundings so as to realize architecture in city planning.”

Objectile

The extension of monads and the inclusion of additional inflections gives the monad a changing ‘texture’ of qualities and potentials. “Extensions effectively are forever moving, gaining and losing parts carried away in movement; things are endlessly being altered; even prehensions are ceaselessly entering and leaving variable components.” The example of architecture’s extension into urbanism cited above illustrates how the object gains new arenas of influence as it attunes to new inflections. “This area of interindividual, interactive clustering is quite agitated, because it is an area of temporary appurtenances or of provisional possessions.” Continuous differentiation of the developing assemblage feeds back into the withdrawn being of the object such that the object “no longer refers … to a relation of form–matter—but to a temporal modulation that implies as much the beginnings of a continuous variation of matter as a continuous development of form.” Michael Guggenheim has demonstrated how architecture is incapable of being restricted to a single domain, but is always an object acting in multiplicity of associations, making it uniquely suited to the model of the objectile, where “fluctuation of the norm replaces the permanence of a law.” Following this concept,
we will show how architecture activates and intensifies the urban dynamic by including its perception of the unlocalizable rhythms of the city as inflected predicates.

§2.6 Environment

Inclusion

Inclusion, according to Deleuze, is formed by the monad’s apperception. Inclusion carries events into the monad,\textsuperscript{88} enabling exo-relations and prompting individual manifestations. Because identity for the objectile is not a reciprocil definition but always a vector,\textsuperscript{89} perceptions advance differentially: infinitesimal variations of perceptions that develop inclusion.\textsuperscript{90} Of the different types of inclusion,\textsuperscript{91} we are interested here in how the monad includes the world within itself. Consistent with the object-oriented position, there is no object which functions as a universal world, one that can contain all others. Despite the fact that, for Leibniz, every monad includes the whole world, the “reason of the series … is not. The limit remains \textit{extrinsic} and appears only in a harmony \textit{preestablished} among the monads.”\textsuperscript{92} In this formulation, every monad is a singular subject; however, these subjects are themselves without objects, “these are minute perceptions lacking an object, that is, hallucinatory microperceptions.”\textsuperscript{93} Despite the inversion of intent, the result retains a correspondence with the object-oriented goal of “subjectless objects” because no monad is thrown under another subject as its correlate, but each exists only for itself.\textsuperscript{94} What is changed in an object-oriented ontology is that there is no longer a guarantee nor a necessity of overall harmonious convergence across the totality of monads.

Incompossibility

For Leibniz, the convergent harmonization of these series was required by the imperative of a single compossible world that is sharply delineated from all others. Deleuze, drawing from Riemannian manifolds, introduces “a fibered conception according to which ‘monads’ test the paths in the universe and enter in syntheses associated with each path… a world of captures instead of closures.”\textsuperscript{95} In this model “bifurcations, divergences, incompossibilities, and discord belong to the same motley world,”\textsuperscript{96} or rather, a plurality of non-exclusive worlds.
Construction

The relation of object to world is a complex one: “there is always a double antecedence: the world is virtually first, but the monad is actually first.” To clarify, we would say that the point of view of the monad precedes the individual object as a potential series of interaction between the monad and surrounding objects, but that the world, or environment that it occupies does not pre-exist as such. Part of the generative ability of objects includes their “active role in constructing their environment, both through determining relevancies in the environment and through actively changing their environment.” For each and every object, therefore, there is a unique environment, which it includes. However, in the same way that objects that become components of an assemblage do not give up their distinct identity or agency to become mere docile parts, so do the environmental conditions exceed the object, they are equally the conditions involved in other existing objects, and that cannot therefore be specified as belonging to that object alone, nor as terminating in it. In particular one can say that “while objects construct their openness to their environment, they do not construct the events that take place in their environment” but relate to it in a feedback cycle of construction and constraint. The need for objects to form “contingent strategies for contending with the environment” constitutes the ground of exo-relations.

Cultivation

Though Deleuze employs the metaphor of the fold to convey the complexly implicated interior of the monad, the virtual dimension from which objects are unfolded is not a preindividual stratum that is continuous like a sheet of fabric. Instead it is like an entangled knot or rhizome: not everywhere continuous but, through a complex selection, continuously interconnected. “This genesis is a genesis from other objects or discrete individuals, and in many instances is productive of new individual entities.” It is perhaps better to use Leibniz’s own images of every portion of matter as teeming with individuals “like a garden full of plants and like a pond full of fishes” in order to remind ourselves of the complex plenitude of components at every scale. As an alternative to assembling or constructing, then, we might also speak of objects as ‘cultivating’ their environments.
Bibliography:


———. "The Road to Objects." *Continent* 1, no. 3 (2011): 171–79.


§3.1 Introduction

Constituent characteristics

The urban model that we put forward, then, must proceed from assemblages that satisfy the parameters set out in the previous chapters: sensitive to various contingencies and able to develop alongside the surrounding environment through a mutual production of the city but yet not simply relational entities defined entirely from without. Rather they must possess an internal consistency through which external pressures are absorbed, translated, and made sense of. Unlike some recent applications of object-oriented philosophy within urban design, we are not especially concerned with classifying the type of object that best represents the city, nor do we advance a naïve formal figure of objects based on a kind of metaphorical materiality. Instead the imperative is to develop an urban model whose components possess degrees of individual autonomy from the urban plan as a whole.

Model autonomy

In particular, it is necessary that the constituent urban assemblages are able to react freely to situational conditions so as to enable the kind of spontaneous formation that is the goal of Assemble Urbanism. “The indeterminacies of the formative processes of urbanism require methods which specify its propositions provisionally.” For this reason, while the city or masterplan as a whole does constitute an assemblage of its own, such a frame is less appealing to this thesis, as it renders all of its responses subject to a single endo-consistency. The aim, then, is not only that the assemblages afford some possibilities, but that the internalization of behaviors and affects ground an object agency that registers its own apperception and exercises a decision-making capacity with significant impact both on its own development and as an action or force “that leads to one particular enactment of the city.” Additionally, the elements of the model must be discrete individuals independent from one another, in order to establish their identities through the selective inclusion or cultivation of their environment, their parts, their relations, and their characteristics.

Nonlinear interaction

The effect of inconsistent assemblages engaged with one another is the ecology of nonlinear interaction needed to provide our urban model with an analogue of the dynamic behaviors of actual, lived urbanism. The key is in enacting meaningful, responsive interaction, that is, responses that “make a difference” in their execution. These interactions stand in sharp contrast to the perfectly coordinated responses of linear effects that do not sufficiently distinguish between individuals but are more typical of the internal actions of a highly regulated assemblage. As we said earlier, urbanism operates also through the openings created by the heterogeneous differences embodied within an assemblage and the potential for transformation present when assemblages

---

2. Amin, “Re-Thinking the Urban Social.” p112
3. Peter Trummer calls the model of the contemporary city the “assembled object” and characterizes it by the state of architecture providing the ground for further architecture rather than exporting that role to territory or a larger scalar figure (such as the grid) which we do agree with.
4. Trummer, “The City as an Object: Thoughts on the Form of the City.”
5. Wiscombe, “Discreteness, or Towards a Flat Ontology of Architecture.”
6. The implication of this is that the model maintains an autonomy from the designer as well.
8. Procedurality is the principal value of the core practice of software authorship.
9. “Procedurality in this sense refers to the procedure of software authorship. Software is composed of algorithms that model the way things behave...Procedural systems generate behaviors based on rule-based models; they are machines capable of producing many outcomes, each conforming to the same overall guidelines. Procedurality is the principal value of the computer, which creates meaning through the interaction of algorithms.
10. “This ability to execute a series of rules fundamentally separates computers from other media”
13. “More specifically, procedurality refers to the practice of encapsulating specific real-world behaviors into programmatic representations.”
14. Ibid. p4
16. Ibid. p14
encounter unexpected or conflictual reactions. Apart from conflicts and effects of translation, the interaction between model elements become nonlinear when they do not follow a single generative timeline or scalar progression, but operate at multiple levels simultaneously or feed impacts back onto the process prompting a readjustment of the current state.

§3.2 Unique Traits of Computation

Formalization and automation

While certainly not the only solution to these criteria, computational modeling is uniquely suited to provide insight to the task of a complex, autonomous, urban design. As an inherently, even uniquely, procedural medium, it is well-suited to simulating behaviors. Furthermore, in the automation and wide variety of possible responses, computation is more capable of producing formal, yet complex, systems than other methods (for example participatory or game-based planning). Lorenzo-Eiroa argues that computation has “shifted the mapping of extrinsic content to the coding of emergent content or agency.” That is, in place of translating external forces from the environmental, technological, or political realm into notational representations that inform design, these forces are being explicitly formalized as starting points that are increasingly absent from the representational aspect of computational modeling, which in turn has shifted its attention to the result of playing out these forces (and others).

Representation and enaction

Procedural representation, as a medium, is not quite that simple, however. “Procedural systems like computer software actually represent process with process. This is where the particular power of procedural authorship lies, in its native ability to depict processes.” In this sense, the representational mode of computation is not constrained to the outcome of a simulation but is at the same time, an enaction. This allows a representation to more closely parallel its objects and produces a more tangibly comprehensible image of the logics depicted, while it also stands on its own as an active environment that can be engaged with independently of its role as a representamen. While representation is always a creative process, putting forth an “independent object” that “defines an agency,” computation hones this point by establishing a self-reflexive representation: aspects of the program are represented within the code to other elements or objects within the program. In the same way, the computational model “represents a formal logic which governs the formation of a category or type,” and simultaneously “a unique solution.” In contrast to instances that become “problematic when there is no agency at a representational level, such as when the content represented is extrinsic to the performance of its medium,” computational agency is located precisely in the representational register. This fact motivates the close focus “on the emergent quality of code” and the particular attributes and constraints it engenders as a form and as an “autonomous logical system.”

§3.3 Authorship

Split agency

One of the defining features of computational design is that it is inherently marked by a divided or displaced authorship. “To write procedurally, one authors code that enforces rules to generate some kind of representation, rather than authoring the representation itself.” The rules then generate a specific instance through the “intervention of some additional agency that may be other than, and even unrelated to, the … designer.” Mario Carpo defines this relationship as the ‘split agency’ of computational design and, though he typifies the product as an “an open-ended algorithm, or a generative, incomplete notation,” he confers a hierarchical blessing on this objectile naming the programmer as the “primary author” or the “real digital author.” Meanwhile the user or operator of the system is relegated to becoming an ‘interactor’ who “exerts only a limited and ancillary form of agency” in adjusting the scenario to a specific case or to personal taste.
Manifold agency

Immediately, however, we recognize that this simple division does not hold in any practical sense. Carpo admits that “in practice, these two stations of agency are often merged into one, as a single agent often does both jobs – first designing the general program, then finalizing one or more specific objects designed and made with it. This is normal and to some extent inevitable.”

Yet, this still ignores the fact that the programs that he portrays as determining are themselves always designed within the constraints of other software or coding languages and are often functionally extended into unintended directions through independently developed plug-ins or libraries, or more prosaically through user scripts. Moreover, there is no room in the two-level split agency for cases where the computer plays out its own scenarios through programmed automation or a generative solver. These cases highlight the fact that in all events, the model itself exercises an agency that does not properly belong to either of Carpo’s two agents. Neither the programming nor the application able to detach from the other as an entirely inclusive, self-controlled activity, but both are conducted as inter-active acts.

§3.4 Unit Operations

Configurative systems

By considering computational models as complex manifold of actions and agencies, we position the model closer to the conceptual idea of the city as an endlessly reconfigurable assemblage (of assemblages) not structured by an overarching law. This does, however, create some difficulty in assessing the success or weaknesses of an urban design. Ian Bogost introduces the concept of ‘unit operations’ as an interpretive tool that engages with procedural media on their own terms. As is the case in assemblage theory, the unit in unit operations is very loosely defined: “In essence, a unit is a material element, a thing. It can be constitutive or contingent, like a building block that makes up a system, or it can be autonomous, like a system itself. Often systems become units in other systems.” More significant is the way that multiple units relate to one another within a work. In contrast to hierarchical systems that “regulate meaning for their constituents,” these units aggregate into “a configurative system, an arrangement of discrete, interlocking units of expressive meaning.” The important detail here is that the level of operational control remains in the individual unit and though these units may be said to form systems, the systems themselves do not become structuring forces but are “the spontaneous and complex result of multitudes.”

20 Ibid. p7
21 One which may be materially identical to the software-sorted agency in operation controlling the actual city. Graham, “Software-Sorted Geographies.”
23 Hays and Trotter, “Re-enchanted Architecture.”
24 Lorenzo-Eiroa, “From Coding to Representation, to Formal Autonomy, to Media Representation, Four Levels of Architecture Agency.” p212
25 simultaneously both the diagram of a procedural method and the quantitative record of an individual instance” or, following Peirce, both a legisign and a signum.
26 Zuecke, Patt, and Huang, “Computation as an Ideological Practice.” p193
27 Ibid
28 Ibid. p213
31 Ibid.
32 cf §2.5 “Objectile”
33 Carpo, The Alphabet and the Algorithm, p126
34 Carpo, “Authors, Agents, Agencies, and the Digital Public.”
35 This terminology is attributed to Janet Murray, whose separation seems more apt as she was writing about digital narrative where the author/audience divide is distinguished by very different tasks in a way that might not translate to a designer/designer divide. Murray, Hamlet on the Holodeck: The Future of Narrative in Cyberspace.
36 Carpo, The Alphabet and the Algorithm, p126
37 Carpo, “Authors, Agents, Agencies, and the Digital Public.”
38 See, for example, the various add-ons to Grasshopper at Food4Rhino: http://www.food4rhino.com/grasshopper-addons or the libraries for Processing: https://processing.org/reference/libraries/ the majority of which are user-contributed in both cases.
39 Like those in this text.
41 Chapter 4 will develop the concept of the Interactive with regard to both the prepositional influence that the designer has over the end products of a computational model, and also the reciprocal influence which the constructed model has in interpreting the inputs of the designer.
Distributed units

Thus “rather than attempting to construct or affirm a universalizing principle, unit operations move according to a broad range of diverse logics, from maximizing profit to creating new functional capacity.” The diversity of logics in play requires close attention to the ways that individual units are positioned within the system as well as how their influence spreads through the network. In the first case, rather than trying to interpret the overall meaning of a network, one might analyze how a particular point of view manifests itself within the context of the network. In the second case, a study of the indirect effects that arise from an isolated action may be called for, or tracking of how the coherence of specific information changes over time. In either case, the emphasis shifts to an exploratory or interpretive response of the situation and away from the “attitudes or values that inform the approaches that created the systems in the first place.”

§3.5 Procedural Rhetoric
Engagement

Following from this logic, we observe that unit analysis produces a unique mode of engagement and representation. The lack of holistic or consensus meaning in the system suggests a complex interaction for the critic, author, or user, but this is not necessarily new. The combination of such open-ended meaning with the dynamic aspect of computational models, however, is. Procedural media “have to be operated. They are not static objects, but active devices, machines rather than texts.” As such, the understanding of the model must occur through interaction and occur “with an eye toward identifying and interpreting the rules that drive that system” not just an assessment of the end results. This is a particularly salient point in design fields, which are accustomed to separating process from product and to seeing products as singular artifacts rather than series of multiples. Bogost argues that scenario modeling constitutes “more abstract representations about the way the world does or should function” than do conventional verbal or visual discourses because the familiarity of those modes of communication have made their tropes more deeply ingrained. At the same time the active response of a computational model can create a more encompassing engagement that prompts further response and deeper attention than a conventional image.

Representation and semiotics

The goal of procedural representation, then, is to maintain the active dimension of the simulation, directing the user’s attention from the product, which is just one contingent state of many, and toward a consideration of the logic behind the scenario being played out, and the simulation that supports it. That is, to encourage computational thinking. “Computation is representation, and procedurality in the computational sense is a means to produce that expression … computer processes are representational, and thus procedurality is fundamental to computational expression.” Computation represents itself best through processes and aims to likewise be interpreted as process rather than images or words.

This immediately calls to mind the work of Charles Sanders Peirce, whose semiotic model was not based on a dialectic between the signified and signer but on a triadic interplay between the representamen, the object, and the interpretant in which the image of the sign brings forth a new image in a process that extends into an infinite series. In this model the interpretation of the sign is not given as meaning but as a continuation of representation. This process, which Peirce called “pure rhetoric,” was not seen as a degradation of, nor affront to, the real, but as a creative process of reproduction in kind.
Bogost points out that the unit operations mode of critique is not solely applicable to digital media, but can be applied to literature, film, or any other work that might struggle to fit within a holist interpretation.

Bogost, Unit Operations: An Approach to Videogame Criticism.

Ibid. p5

That is, the system is not structural with regard to its components. Systems may act as units (encapsulated, black-boxed, counted-as-one) in their relations with other units; unit analysis is not an atomistic theory.

Ibid. p4

cf. §7 4 'Nonlinear Dynamics'

Ibid. p6

This is not to discount the interactive potential of images:

Ibid. ix

Patt, "The Collective Image: Form, Figure, and the Future."

Ibid. p8

Carpo, The Alphabet and the Algorithm. p103

Ibid. p9

"The interpretation of the sign is not, for Peirce, a meaning but another sign; it is a reading, not a decodage, and this reading has, in its turn, to be interpreted into another sign, and so on ad infinitum … Only if the signs engendered meaning in the same way that the object engenders the sign, that is, by representation, would there be no need to distinguish between grammar and rhetoric."


"The object of representation can be nothing but a representation of which the first representation is the interpretant. But an endless series of representations, each representing the one behind it, may be conceived to have an absolute object at its limit. The meaning of a representation can be nothing but a representation. In fact, it is nothing but the representation itself conceived as stripped of irrelevant clothing. But this clothing never can be completely stripped off, it is only changed for something more diaphanous. So there is an infinite regression here. Finally, the interpretant is nothing but another representation to which the torch of truth is handed along; and as representation, it has its interpretant again. Lo, another infinite series."


Baudrillard, Simulacra and Simulation.


McCollough, "4k Formalism: An Interview with Ian Bogost."


Patt, "The Collective Image: Form, Figure, and the Future."

Baudrillard, Simulacra and Simulation.


McCollough, "4k Formalism: An Interview with Ian Bogost."

Rhetoric of simulation

Suggestively, Ian Bogost, has proposed the name “procedural rhetoric” to cover techniques “for making arguments with computational systems and for unpacking computational arguments others have created.” Following the category of “expression that represents processes or systems with processes or systems,” (and Peirce’s thesis that the interpretant further produces new representations), procedural rhetoric covers both how “simulation authors … think about their objects as systems and consider which are the laws that rule their behaviors,” and the ways in which “people who interpret simulations create a mental model of it by inferring the rules that govern it,” neatly bridging the problematic split between the designer and user discussed above. Ultimately, the confrontation between the authorship of the creator against the application by the user is flattened out in the jump to infinite series enabled by the authority of the simulation itself.

Bogost frames procedural rhetoric as a persuasive tool, with a primary interest in how videogames can be employed as a medium of critique or political statement. From this perspective, he writes, “Persuasion is related to the player’s ability to see and understand the simulation author’s implicit or explicit claims about the logic of the situation represented.” Interacting with a simulation, requires one to make attempts to understand the logic and “to analyze, contest and revise the model’s rules according to his personal ideas and beliefs.” Though many procedural media restrict the possibility of rewriting the logical rules of the simulation itself, such calibration of procedures, responses, and degrees of freedom is often possible within—even typical of—a computational design process. “The iterative, reductive blending of the model’s system of transition functions over the course of the design process produces an explicitly structured and strategically searchable solution space.” This ‘intention space’ formalizes the model’s ideological position, and reframes the persuasive role of procedural rhetoric “from the simple achievement of desired ends to the effective arrangement of a work so as to create a desirable possibility space for interpretation.” These interpretations (or interpretamen) reflect the Peircian rhetorical mode, and can themselves be related to one another diagrammatically as a non-Euclidean spatial figure, which can illustrate trends or groupings of potential parameter states. A better understanding of the parameter space can be used in a more traditional argumentative role to construct a more explicitly comprehensible parameter space and “to address the logic of a situation in general, and the point at which it breaks down and gives way to a new situation in particular.”

§3.6 Model and Agency

Matters of concern

Although we earlier referred to the model’s authority, this wrongly suggests that the model is the ultimate arbiter of design decisions. In fact, “a computational engine is not a conclusion but an evolving document which formalizes, refines, and clarifies its authors’ intents.” Moreover, the potential stimuli reactions and invariant relations built into the model are activated to highlight significant concerns and direct the attention of subsequent users toward particular issues, what one might call the ‘bias’ of the simulation. When the computational model assembles a heterogeneous array of unit operations “that simultaneously embed material, functional, and discursive modes of representation,” it is capable of transitioning from purely quantitative matters of fact into matters of concern that also incorporate associations and intentions.

In contrast to the ‘parametricist’ approach, the goal is not a correlation of diverse data into a single communicative platform, but to only to frame this information such that it can be positioned within a single conversation. The imperative on the designer is thus to afford multiple types of interaction simultaneously against a changing background. In contrast to a conventional design process, the computational model is not structured as stages with distinct solutions but as an integrated model where different kinds of information engage the designer to address from many angles a single, complex situation.
Agency

Finally, we argue that the expanded field of rhetoric provides a more substantial support for design agency, as “a purposeful inclusion of critical practice and the architectural project but also of the more specific use of artificial intelligence techniques in a design setting.” Just as a rhetorician first establishes an argument and then engages in a discussion or debate, so design agency is exercised first as authorial intent asserted through the construction of a configurative model, and then again through an interactive engagement of that system that develops a relational understanding of the situation from a particular point of view. In the chapters immediately following we will examine various way by which a designer might work in this mode. Following that, we will return the question of why this is especially needed in urban design and argue that this thesis is not limited to computational design but can be extended to a general theory of urbanism.

66 Frasca, "Videogames of the Oppressed.”
67 “The player or critic could make appeals to authorship or origin, but such an act isn’t necessary—it’s equally satisfactory to reflect on the role of a strange, unfamiliar machine.”
68 “Procedural rhetoric . . . has possible use well outside of games . . . certainly in computation more generally, but also in domains that use modeling as their representational mode. That can include physical models, for example, or demonstrations, or perhaps even scientific experimentation.”
70 cf §4.2 ‘Interactive concerns epistemology’ where we argue that interaction is primarily concerned with understanding.
71 Frasca, “Videogames of the Oppressed.”
75 Ibid. p99
77 Zuelzke, Patt, and Huang, “Computation as an Ideological Practice.” p194
78 Bogost, Unit Operations: An Approach to Videogame Criticism. p97
79 Ibid. p105
80 Latour, “Why Has Critique Run Out of Steam? From Matters of Fact To Matters of Concern.”
81 Schumacher, “Parametricism: A New Global Style for Architecture and Urban Design.”
82 Schumacher claims: “Employing associative logics correlates the the different urban and architectural subsystems in ways that make them representations of each other. Everything communicates with everything. This is not a metaphysical assertion about the world, but a heuristic principle for parametric design under the auspices of parametricism.”Schumacher, “Parametric Semiology: The Design of Information Rich Environments.” p178
83 cf §4.5 ‘Forms of Output’
84 “Architectural design, practiced computationally, possesses a unique temporality which escalates the traditionally iterative process of design by drawing together initial premises, processes, and effects produced simultaneously. Such integrated, self-informing feedback gives the impression of automation, but in fact allows (even sometimes requires) the architect to reexamine, reassemble, and elaborate upon the early assumptions rather than accepting them as values given.”
85 Zuelzke, Patt, and Huang, “Computation as an Ideological Practice.” p195
86 Gerber, “Parametric Tendencies and Design Agencies.”
87 Zuelzke, Patt, and Huang, “Computation as an Ideological Practice.” p187
Bibliography:


§4.1 Introduction to Part II

Operative criticism

In the following sections we will trace out a project for urban design that leverages procedurality as a medium for discarding the boundary between macro- and micro-scales or between urban and architectural design stages. This project will be the product of bundling together the theoretical sources that have just been covered into a framework in which they relate to one another as complementary moments of an attitude toward the city. This framework constructs a correspondence between the processes and developments of the contemporary urban condition and the processes and enactment of a computational design model, bridging the gap between analytic and operative theory.

We will exposée this attitude through four thematic aspects of computation, building up in complexity of dynamism and temporal interrelation. This chapter continues the discussion of procedural rhetoric from chapter 3 into a discussion of the interactive nature of computational models and the ways in which interactivity both expands and constrains the possibilities of the computational model. Following this, we will examine the generative potential of locally situated behaviors as encapsulated in agent-based models as an extension of the analysis of monadology from chapter 2 and the emergence of virtual orders that result from a reflexive feedback of these behaviors as theorized by assemblage urbanism in chapter 1. Finally, we will introduce an entropic function to counter the overdetermined, teleological tendency of computation and orient toward the goal of the open city.

Design illustrations

Borrowing a phrase from Gilbert Simondon, one could say that "this mentality is developing, and therefore incomplete and at risk of being prematurely considered as monstrous and unbalanced. It requires a preliminary attitude of generosity towards the order of reality that it seeks to manifest." For this reason, each theme is accompanied by a design example that will be used to illustrate and develop key aspects of each theme. These exercises are themselves also incomplete: fragments that emphasize one particular aspect of a computational urban design attitude, though they could not avoid participating in each theme to some degree. These design exercises will also enable the discussion of actual techniques for producing computational models through real, working code examples.

When possible, these will be presented within the text though some supporting code will be relegated to appendices or omitted in order to preserve legibility and a consistent thematic flow.

---

1. “What is normally meant by operative criticism is an analysis of architecture (or the arts in general) that, instead of an abstract survey, has as its objective the planning of a precise poetical tendency, anticipated in its structures and derived from historical analyses programmatically distorted and finalized. By this definition operative criticism represents the meeting point of history and planning. We could say in fact that operative criticism plans past history by projecting it towards the future. Its verifiability does not require abstractions of principle, it measures itself, each time, against the results obtained, while its theoretical horizon is the pragmatist and instrumental tradition.”


3. The projects described have all been produced using Grasshopper, a visual programming environment and plugin for Rhinoceros. The computational excerpts presented here have all been written as GhPython modules, which allow for customized textual coding within the node of a Grasshopper component. An ellipsis (...) denotes that a single line of code had been continued onto the next line because of length. Note that GhPython can process broken lines implicitly and that the ellipsis is only employed here as a graphic aid.

4. The projects will not be presented in their entirety, rather individual GhPython code will be shown and their significance and role within the overall definition will be commented on. A map that situates each code fragment within the project definition will be included at the beginning of each of the appendices. The entire Grasshopper definitions will also be available for download at http://labebha.net/asembe/forge/

5. Code examples will be identified in the text by the symbol # and references to individual lines will be identified in parentheses with a colon thus (10).


7. Ibid. p11


9. Ibid. p105

10. Ibid. p105

11. Ibid. p95

12. Ibid. p104

13. Ibid. p104

14. “there is an urge to literally reground the environment with an intelligence of place—interpreted not so much in the conservative sense of Martin Heidegger’s and Christian Norberg-Schulz’s genius loci but more in Elia Zenghelis’s contemporary interpretation of uncovering existing logics of reality and finding a site’s capacity by distinguishing the junk from the potentials.”
§4.2 Introduction to Interactive Rhetorical models

In recounting a brief history of urban modeling, Michael Batty argues that the trend of computational urbanism from the mid-1980s turned toward “building models that informed, extended our understanding, focused us on key issues,” that is, toward the development of rhetorical models that enfranchised bottom-up, decentralized urban formation. By focusing on “how spatial structures might emerge” through agent-based interactions driven by individuals rather than aggregates, urban modeling found a way to rigorously approach the dynamic and unexpected changes observed in cities in a speculative way if not yet operationally.

Already in 1969, Ian McHarg made a similar point about his use of overlay mapping for land use planning: that although the empirical and overt methods he used were significant, so too was the fact that this process opened up space for the community to insert their own wishes and aims. Inasmuch as his clearly delineated and categorical maps effect a persuasive rhetoric in themselves, McHarg’s methods are mobilized to make the “obscure and covert” criteria of the planner or designer explicit so that they can be engaged with directly. More precisely, his methods allowed the designer to present a model which could lead to various outcomes following discussion. McHarg repeatedly comments that none of the case studies presented in Design With Nature can be considered plans in themselves. “A plan includes the entire question of demand and the resolution of demand relative to supply, incorporating the capacity of the society or institution to realize its objectives,” but are at most “an expression of physical, social and economic goals.” In fact, this is the element of McHarg’s work that we are most interested in here: over, and in place of, a plan, the collection of empirical facts from sited processes and the translation of them into a framework that engages with adaptable or nondeterministic goals.

Processes as values

Rather than a plan, which gives only the image of an expected end, the goal is to construct a new definition of the site that can be engaged with in explicit terms. For McHarg this meant that “the place is a sum of natural processes and that these processes constitute … values.” That is, to aggregate the significant processes occurring at each location in a way that enables comparison between them and between different locations on multiple criteria. McHarg called this the “intrinsic suitability” of a site, what today may be more often referred to as the “local intelligence” of the site. The significance of local intelligence is twofold. First, it establishes a datum of objectivity to the site definition that deftly moves the conversation beyond foundational matters and toward operational responses instead (while refraining from fixing these responses—changes can still be made or supplements introduced to the source data). Second, it puts forward an alternative to ‘geometrical’ planning based on a priori principles.

Still, it would not be much of an improvement to simply substitute local intelligences for geometric principles within a conventional plan. It is important that this process also enacts a translation of processes into values, of empirical data to operational logics, of matters of fact into “matters of concern.” Thus, it is important to frame these values relationally as contingent on the initial site definition as well as the proposed application, and as reactive to changes in the model over time. “For certain land uses the maximum condition will be preferable, for others it will be the minimum that has the highest value … In addition, in certain cases some factors will be conducive to specific land uses while others are restrictive.” Similarly, there will be sites that are equally suited for multiple uses that may only be decided through the influence of tertiary processes. These sites of multiplicity are also interesting for suggesting more complex considerations of hybridized uses or occupation by phased progression through multiple states. Rather than absolute values then, the site is really defined as a set of tendencies, each of which implies certain costs or benefits to pursue.
Interactive concerns epistemology

Interactivity, in this case, is the exploration of potential tendencies and a testing of their possible realizations. The interactive is an experimental moment and it is equally an experiential moment. The experience of the action and response separates interactivity from merely reading an analysis. As Bogost described, the definitive feature of interactivity occurs when one engages with a model as a means of comprehending its inner workings. Thus we link the interactive to epistemological concerns.

In the following, we will examine three aspects of this mechanism. First, there is a particular interest in how information is made and presented as a concern—the identification of relevant data and the interpretation of its significance within the structure of the design question. Second, there is a question of how information can be added to the model—whether as an organizational method by the designer or as a configurational input from other external sources. Third, is the need for information to be shared or translated from one domain to another for example, formatting the output of the model so that information can be passed between different operations or to further resonate in the social world where it can be viewed and assessed.

Conclusion

The interactive dimension arises through these three aspects and through an arena of mixed impulses: the internal dispositions of the logical units of the model, tendencies of the environment, and individual actions; temporary states and provisional behaviors; expressions and translations within the model and beyond. One of the most significant aspects will be the selection and recombination of different data sources to produce new, synthetic interpretations. McHarg had already seen the potential of that computational methods could bring to this problem, though the challenge today is not with technological capability, but grappling with the question of how the model shapes our understanding of the city.

§4.3 Significance of Information Interaction

Prepositional mode of being

In defining the model, special attention must be given to the means of interaction and the combinatory modes that they support. If the variability of the model is reduced to simply providing templates or dictating a narrow set of options, the concept of interaction as a means of understanding or making expressing concerns is significantly compromised by the reduced role of procedurality and the short-circuiting of emergent results.

The mechanisms by which the interactive operates—that is, the interfaces between different datasets—do not supply information or have meaning in themselves, but establish the modes of translation between objects. They “set up what comes
next without impinging in the least on what is actually said,”26 in a pre-positioning that is “light but also
decisive.”27 The significance of this point is again to establish the product of the interaction as an object
created through the information process. “The finished work is always a novelty, discovery, or surprise,”28 it
is not a type given in advance or imposed by the designer, but as argued earlier,29 the designer “welcomes,
gathers, prepares, explores, and invents the form of the work”30 through the procedural medium.

Such a ‘work’ can be said to always be in the process of formation, which means that the
instantaneous state of the design is equally acting as a source of information, highlighting the fact that
the initial conditions themselves have been taken up *in media res*, at an arbitrary point.31 Rather than
being troubled by the arbitrariness of starting or stopping points,32 we note that this gives a consistency
to our objects: there is no separate species of *a priori* conditions that are free from historical or temporal
contingency. The design “is arbitrary and contingent, but from that moment on it becomes a part of the
contrasts that we will have to make use of in order to sort things out.”33 For as long as the model itself
remains continuously reactive, the status of the situation, its tendencies, and trajectory are traced out by
the receptivity to influence of the assembled objects and the character of their responses. The capacity of
each model to produce a unique ontological assemblage of objects is established through the prepositional
mode of its interactive dimension.34 To fully understand the agency of the model requires a commitment to
the indirect diffusion of actions and the complex multiplicity of influences that result from continuous and
ongoing processing to information.

State space

In contrast to the typical parametric model that is defined by a quite rigid and hierarchical structure,35
we are calling for a much more radically open definition of the relations that allows transformation of the
organizational logic of the urban model. Parametric models have long been characterized through the
concept of ‘state space’, a topological map of all values of a system’s degrees of freedom comprising a diagram
every possible outcome.36 This operation reveals how deterministic parametric modeling can be inasmuch as
it accounts for all possibilities in advance. However, as an analytical—rather than definitive—tool for
differentiating the current state of a model from others, or for reading and comprehending the intricacies
of internal relations, state space can be quite useful. Though the global configuration of the state space may
yet be subject to dramatic changes, the current state’s local neighborhood of possibilities can reveal how the
model might be positioned to change in the immediate future or if it is subject to recurring behaviors.37
Representing the fluctuations of model state is an important aspect of instrumentalizing complex models.38

§4.4 Modes of Input

Flatwriter

Contemporary with the publication of *Design with Nature* was one of the first projects to
explicitly link digital, computational methods to an open-ended, interactive design product.39 Yona
Friedman’s *Flatwriter*,40 a computer model, “which follows certain rules of composition inspired by the
urban regulations. This computer shows visually (in a master plan) the city which will be composed little by
little according to the visitors of the exhibition handling *Flatwriter*. This visualization will be made by means
of cathode ray monitors placed above *Flatwriter*.”41

Friedman ran up against the computational limitations of his day—the project was never realized,
even after the ’70 Osaka World Fair it had been proposed for had passed42—but it was described in sufficient
detail that it could be reproduced.43 Given Friedman’s preference for a “nonpaternalist”44 system (one that
does not interject its own judgments, but defers to the user input) that enables not-specialist individuals
to design their own dwellings, *Flatwriter* was developed almost entirely through interactive elements with
minimal predefined logic. Friedman envisioned it as “an application of a new information process between
the future user and the object.”45 The kernel of the project is in the implementation of a set of simplified
symbols representing elements and transformations that can be understood immediately46 and the input
of the user’s desired apartment configuration through a keyboard that corresponds to these symbols. Any additions or changes to the master plan can be shared with all users through analytic charts that relate quantitative and qualitative aspects of the plan’s current state. Here we clearly observe the three aspects of interactivity that we highlighted earlier: that of understanding concerns, of adding data to the model, and of sharing or translating information.
Individual input

The influence of Flatwriter is still apparent in the computational design landscape of today. For example, VillageMaker, by MVRDV and The Why Factory, proceeds much like Flatwriter: values are added by a user one-by-one through a series of inputs selected from a menu of options, and as individual users accumulate, the master plan emerges as an aggregation. The results are produced by a very algorithmic process whereby a change in the values requires a return to the beginning to re-enter the new values when prompted. The rise of purpose-built parametric modeling softwares has decreased the need for such a linear process, while also increasing the number of available input mechanisms and their complexity.

As Mario Carpo has warned, reliance on the set of available mechanisms can be somewhat restrictive to the designer, however most mature software platforms include the ability to customize the software’s functionality to various degrees or to add or write new functions. For example, the GraphMapper component in Grasshopper allows the direct manipulation of a law curve, but the interface only allows automated inputs for the independent variable, requiring manual editing for any of the curve parameters or graph dimensions. The following example, uses a Python script to access the active components from the Grasshopper environment and identify the desired component by type and by a .Nickname attribute. Once identified and saved, this component’s properties can be modified as an active parameter as in .

\[ \text{4.4.2} \]

Molar input

Often it will be inefficient or impossible to enter all values directly and large sets of data will need to be imported into the model at once, particularly on urban-scale projects. David Gerber recounts how, working on the proposal for the One North masterplan, “we were confronted with the need for managing vast data bases which required the fast visualization of the modifications.” The invention of a planning tool was considered an essential aspect of the deliverable product as well as an active ingredient in the design process. The result was a software which allowed input and editing of planning tables within a spreadsheet and translated that data to models of three-dimensional form. The direct relationships between data and form a establishes One North as one of the first instances of parametric urbanism in practice, and it demonstrates the need for a control mechanism to coordinate or parameterize those relationships. Projects 2, 3, and 4 will all involve importing large external data sets from different file formats.
INPUTS: nickNames As List of string
#FIND OBJECTS AND METHODS AT:
#  gh.Kernel.Graphs.GH_GraphContainer.xxx
#CALL BY:
#  obj.Container.xxx
import
import Grasshopper as gh

graphObj= [False for x in xrange(len(nickNames))]
for obj in ghenv.Component.OnPingDocument().Objects:
  if type(obj) == gh.Kernel.Special.GH_GraphMapper:
    if nickNames.Contains(obj.NickName):
      ndx=nickNames.IndexOf(obj.NickName)
      graphObj[ndx] = obj

scriptcontext.sticky["graphObjects"] = graphObj

OUTPUTS: None

---

**4.4.1 GraphObjects**

This code creates a 'sticky' python list of Grasshopper Graph objects from the canvas:
https://github.com/mcneel/rhinopython/blob/master/samples/sticky.py

Note: Grasshopper components' NickNames are accessible as the first line in the right-click menu. By default, GraphMapper components are named 'Graph'.

Reference: Marcus & Hannes Leschke:

---

**4.4.2 ModifyGraphX**

Modify the internal parameters of GraphMapper objects on the canvas, (namely their X1 value) remotely from a separate script component:
Note: boolean button input refresh triggers recalculation if script component has not properly updated.

input scriptcontext

graphObj = scriptcontext.sticky["graphObjects"]
for i, graph in enumerate(graphObj):
  graph.Container.X1= 2*X1[i]

OUTPUTS: None

---

Figure 4.3
View of the Grasshopper canvas with four GraphMappers and the GhPython components for 4.4.1 and 4.4.2.
Create a new class that contains a point location and any additional localized metadata:

07 This class is located within the code for the TopoFlow component (4.6.1:07-32) and many of its methods refer to this script.
14 The initiation operation takes multiple arguments. Because Python is dynamically typed, these and the methods below can be filled with default values (i.e. False) then changed on the fly throughout the code, to suit needed purposes. For example, a variable might be initialized with a Boolean as a check to test whether the expected value has been assigned yet.

The typical uses of these methods are as follows:

- _xyz_ contains the point location.
- _src_ records the indices of the previous point along the flowline from this point.
- _trgt_ records the indices of the next point along the flowline from this point.
- _tan_ contains the tangent vector along the topographic curve at the point.
- _aggr_ contains an aggregate value that increases as a flowline lengthens or merges with another.
- _slp_ contains a vector that points from the point at _src_ and the point at _trgt_.
- _pln_ contains a plane oriented to the slope and tangent of the topography.
- _sol_ records a value for solar incidence calculated in a separate script component as the degree between the normal vector and the average daytime sun angle.
- _solHrs_ records the number of hours of direct sunlight at the location.
- _attr_ processes the proximity to various features of the site.
- _fit_ converts site values into fitness values according to input parameters.
- _typ_ records the indices of the two highest fitness values.
- _ornt_ stores a plane that blends various site orientations based on the point type.
- _vvv_ holds an area polygon calculated from the voronoi diagram.
- _area_ evaluates the area of the voronoi cell.
- _ntwrk_ stores lines that connect each point to its adjacent points.

Figure 4.2
Visualizations of various site data that is stored in the TopoPoint class.

a) slope (21)

b) areas of high runoff (19)

c) average solar incidence (23)

d) cell area (32)

e) attractor vectors colored by type (26)

f) type (28)
Derived values

Finally, within the parametric model, processes will accept input data that is computed automatically by other processes. The graphical interface of Grasshopper is built around the concept of explicitly visualizing these events and structures the flow of data from left to right. One way of extending the possibilities for receiving input data is to create object classes with attributes to store various information. Using GhPython, these object attributes can be modified by any script component and the changes are reflected in any other component that accesses that object,61 effectively short-circuiting the linear flow of the Grasshopper graph. In 4.4.3, we create a class for storing a list of properties about a topographic location its xyz-coordinates (16), its position relative to other points (17,18), and a number of analytic measurements (slope (22), solar incidence (24), proximity to external influences (26)) as well as values that will be used later to define its role in the masterplan (27).

As the project develops complexity, new attributes and methods can be added by the designer at will. Conversely, some of this metadata may never be accessed or even assigned a value, but the object is still prepared to receive certain information.

§4.5 Forms of Output

Metadata and datastructures

For data that is going to continue playing an active role in the model, it is clearly necessary to make this information easy to retrieve and also to make sense of. This is especially important if one anticipates frequent reference from diverse sources within the code or at sporadic intervals as one can expect from complex and nonlinear processes of urban design. Saving location-based analyses as attribute data is convenient because one typically compares or synthesizes many features of a single location with one another. Keeping this information as metadata of a class ensures that the data for one location will not be mixed with that of another as a result of, for example, differently sized lists. Another strategy is to use the data’s position within a datastructure to convey information or identify a set of data. Frequently, we will use an object’s position within a list as an identifier such that other processes can access by means of the list index.62 More explicit, is the dictionary datastructure63 in which each value is entered alongside a key that can later be used to recall it. This key can take any value (provided it is unique within the dictionary). Constructing a class to extend the dictionary class allows Python scripts in Grasshopper to transfer data from one component to another without the need of reformatting in a native Grasshopper format. In the following examples we show how to construct such an extension of the defaultdict class64 (=4.5.1) and then how that class can be used to sort a list of topographic curves using their elevation as the key value (=4.5.2).

57 “The planning tool was developed in parallel with the project and was meant to have acted as a design participant” Gerber, “Towards a Parametric Urbanism.” p159
58 Gerber categorizes the project as “pseudo-parametric” because the planning tool was uni-directional. Gerber, “Parametric Practices.” p102-103
59 Ibid. p104
60 Appendix.5.1 importing from a .csv text file, Appendix.6.1 importing from an image map, and Appendix.6.2 importing from an .osm text file.
61 The change will occur when the component is prompted to update by its explicit input parameters, thus requiring some coordination in the definition to prevent contradictory values.
62 cf. =5.4.1 Half-Edge Mesh
63 https://docs.python.org/2/tutorial/datastructures.html
64 The main distinction between dict and defaultdict is that the latter creates a key value automatically if one attempts to reference a key that doesn’t yet exist in the dictionary. This is convenient when the values are lists as they are in =4.5.2 because we can move straight to the list functions (append in (13) for example) without first needing to check that the list exists.

http://docs.python.org/2/library/collections.html#collections.defaultdict
**4.5.1 ghDefaultDict**

- **INPUTS:** None
- **Imports:**
  ```python
  from collections import defaultdict
  import scriptcontext
  ```
- **Class ghDefaultDict:**
  ```python
  class ghDefaultDict():
      """custom class:ghDefaultDict
      EX: myDict= scriptcontext.sticky['ghDefaultDict'](defaultdict(list))
      ""
      def __init__(self, defD):
          self.d= defD
      def __str__(self):
          strLen= str(len(self.d.keys()))
          return ('ghDefaultDict with ' + strLen + ' keys')
  ```
- **Function:**
  ```python
  scriptcontext.sticky['ghDefaultDict']= ghDefaultDict
  print (ghDefaultDict.__doc__)
  ```
- **OUTPUTS:** None

**4.5.2 dictTopoCurves**

- **INPUTS:** topoCrvs as List of Curve
- **Imports:**
  ```python
  import Rhino.Geometry as rhG
  from collections import defaultdict
  import scriptcontext
  ```
- **Function SortCrvsToDictionary:*
  ```python
  def SortCrvsToDictionary(crvList):
      crvDict= scriptcontext.sticky['ghDefaultDict']...
      (defaultdict(list))
      for i, icrv in enumerate(crvList):
          pt= rhG.Point3d(icrv.Curve.PointAt(iCrv,.5))
          crvDict.d[round(pt.z)].append(iCrv)
      return crvDict
  ```
- **Function:**
  ```python
  srtdCrvs= SortCrvsToDictionary(topoCrvs)
  print(srtdCrvs.ToString())
  print("KEYS")
  print(srtdCrvs.d.keys())
  pyDict= srtdCrvs
  ```
- **OUTPUTS:** pyDict

**Example:**

- **Function ghDefaultDict used to sort topography curves.** The benefit of using defaultdict is that lists of curves can be saved to a single key—in this case, the curves’ elevations, allowing convenient organization of topocurve sets with multiple peaks.

**Additional information:**

- **Reference:** Benjamin Golder & Giulio Piacentino:
  http://www.grasshopper3d.com/forum/topics/exchanging-basic-python-types-between-separate-python-components

- **Additional thanks to Jason Lim:**
  http://www.grasshopper3d.com/forum/topics/trying-to-exchange-a-defaultdict-python-component

**01** import Python’s defaultdict.

**05** Begin new class definition, this must be placed in the code above the first instance of it being called.

**06** Text to print when the __doc__ method is called (as in :18). Here, an identification of the class and a reminder of how to instantiate a new ghDefaultDict.

**10** Instantiation operation includes a defaultdict in its arguments. This is accessed through the method .d [:11].

**13** Method to print number of keys in a ghDefaultDict.

**18** Save this class to sticky dictionary to be accessible to other script components.
Within the design process there is also the need for “some mediating series of mechanisms which operate on the messages sent between model and author.” Without a means for the model to communicate detailed information about its state or processes, the designer has limited ability to interact with it in a meaningful way. The visual display of quantitative data is a necessary part of making informed decisions when confronted with the amount of data which an urban model is likely to contain and the fact that active processes might be continuously altering this data only compounds the problem further.

The Grasshopper environment is predisposed to visual representation of geometric forms in one window while displaying more technical information on the editing canvas. The divide between geometry preview and process operation sometimes makes it difficult to intuit the connection between the two and hinders the performance of the model as an interactive medium. Though a great deal of urban modeling can exist in a purely quantitative mode, the scale of urban design which is under discussion here involves spatial and formal data as an integral, even motivating influence on the success or failure of the model. The projects which we present will emphasize the presentation of quantitative measures as spatially differentiated as the properties of geometries or superimposed on them.

Datascapes

Such mapping of quantitative data over actual spatial information calls to mind the datascapes pioneered by MVRDV in a method that seemed to elide the distinction between the forces shaping the project and its ultimate form or organization. Bart Lootsma describes datascapes as “visualizations of laws, rules, norms, and statistical probabilities, and as such they constitute representations of . . . bureaucratic systems where the trust in the system as well as the people, institutions and machines that represent it, lies in one’s confidence in certain specialized expertise.” Data visualizations are therefore ultimately rhetorical statements, “in that they image data in knowingly selective ways. They are designed not only to reveal the spatial effects of various shaping (e.g. regulatory, zoning, legal, economic, and logistical rules and conditions), but also to construct a particular eidetic argument.” As we have argued, the rhetorical dimension is a key aspect of a procedural model, when this is expressed through processes and behaviors, interaction and response rather than simply an image. This task requires going beyond the limits of the datascape, not simply to ground an authority based in sublimated pragmatics, but as a constructed framework that enables the procedurality of the model to be expressed. An interactive model cannot pretend to embody the ‘correct’ results in itself, but must be enacted in coordination with a series of parameters and inputs that explore and invent the final form.

---

63 User feedback
67 The adoption of HTML5 has widened the field of experimentation with data visualization, prompting the adaptation of graphical methods to interactive processes. The biggest transformation being the possibility to present large datasets in their macro context while allowing individual information to be selected and isolated to highlight specific details.
68 The Diagram Debate, or the Schizoid Architect.” p22
69 MVRDV, FARMAX: Excursions on Density. p102
Customizing the display

The following code examples define methods for interrupting Grasshopper’s default display pipeline and modifying how the program renders geometry previews. The first, `customLineDisplay`, addresses the display of curves (§4.5.3) or line segments. It enables a lineweight function in addition to the display color, giving more expressive capability, useful in the creation of bar graphs, or simply for visual emphasis (Figure 4.5).\(^7\) As a complement to this, `customMeshDisplay` (§4.5.4) makes it possible to display texture and transparency maps on a mesh surface. This allows more visual differentiation of surfaces, or the use of notational textures such as a hatch pattern to denote material patterns or a gradient of intensity. The texture maps used in Figure 4.6 are tileable patterns registered to a global position, giving them a consistent continuity independent of the mesh position. In contrast, the pop-up screens, which are drawn in front of the geometry and oriented toward the view camera, have their image maps registered to a corner of the mesh so that the ideogram remains centered when the camera moves and the geometry is redrawn. The creation and positioning of these screens is detailed in (§4.5.5).

Archive

It will also occasionally be necessary to export data to a static form for archiving or to exchange with another program, for example to export geometry compatible with rendering software. The method of ‘baking’ geometry is a fundamental feature of Grasshopper and even the more customized baking operations are easily managed with available plug-ins, so this will not be covered here. Sometimes, in order to produce computationally intensive analyses, it is more convenient to record the data to an external file so that it can be analyzed in a separate operation. An example of this process will be covered in Project 4 where a text file is created to save calculated values that will later be brought back into Grasshopper for visualization purposes and as active data reassociated with the model.\(^7\)

\(^{72}\) Beyond the preview visualization, this method is also helpful for final documentation at high-resolutions where the default lineweights can become too faint to read clearly.

\(^{73}\) cf §7.3a ‘Network Depth’

Figure 4.5

This map assigns different colors to separate categories and a gradient of lineweights (in a range from 1-6) to intensities.
Replaces the default display pipeline for given curves allowing customization of colors and lineweight.

Reference: Mostapha Sadeghipour Roudsari: [link to forum thread]
based on the example 'starMaker' by Steve Baer: [link to PythonRhino entry]
(Note: this link no longer exists.)


This class manages the events of drawing geometry within the display pipeline and overriding the default display. This script will only work on Curves, other geometry included in `crvs` will be ignored. If the geometry, color, and lineweight lists (`crvs`, `c`, `w`) are not of the same length, the script will use the 'longest list' matching method by repeating the last entry of the shorter list (this allows, for example, a single color or lineweight to be input, in place of a long list of identical values).

The boolean input `custDraw` must be `True` for the display to preview. Similarly, it must be `False` to hide the display or the component must be disabled. Simply setting the Python script component to 'Preview Off' will not remove the preview. Neither will switching to a different definition or closing (without unloading) the Grasshopper window.

```python
import Rhino as rh
import Rhino.Display.DisplayPipeline as rhDisP

class CustomObjectDraw:
    def __init__(self, drawCrvs):
        self.geometries = []
        for crv in drawCrvs:
            # GET GEOMETRY FROM ID
            geoID = ghdoc.Objects.Find(crv).Geometry
            if (geoID != None):
                self.geometries.append(geoID)

        # CALCULATE BOUNDING BOX FOR UPDATED OBJECTS
        self.UpdateGeometry(self, drawCrvs)

        # DRAW OBJECTS
        self.PostDrawObjects += self.MyDisplayPostDrawObjects

    def UpdateGeometry(self, object):
        temp_object = object
        if (temp_object != None):
            # UPDATE THE BOUNDING BOX
            temp_object = object.GetBoundingBox(False)
            self.temp_objects_bbox = temp_object.GetBoundingBox(False)

    def RemoveHandler(self, sender, e):
        # REMOVE THE EXISTING CONDUIT
        self.RemoveHandler -= self.MyDisplayPostDrawObjects

    def MyDisplayPostDrawObjects(self, e):
        try:
            e.Display.DrawCurve(geometry, c[min(len(c)-1,1)], ...)
            e.IncludeBoundingBox(self)
        except: pass

    def main():
        if len(crvs) > 0:
            CustomObjectDraw(crvs)

def main():
    if custDraw:
        print("Custom Display")
    else:
        print("Typical Display")
OUTPUTS: None
```
import Rhino as rh
import Rhino.Geometry as rhG
import Rhino.RhinoDoc as doc
import rh.Display.DisplayPipeline as rhDisP

#=================================#
class CustomObjectDraw():
    def __init__(self, drawMesh):
        self.geometries= []
        # ADD GEOMETRY INTO SELF
        self.geometries.append(mesh)
        # CALCULATE BOUNDING BOX FOR UPDATED OBJECTS
        self.UpdateGeometries(self, drawMesh)
        # CALCULATE NEW DISPLAY BOUNDING BOX
        rhDisP.CalculateBoundingBox += self.MyDisplayCalBBox
        # DRAW OBJECTS
        rhDisP.PostDrawObjects += self.MyDisplayPostDrawObjects
        # THIS SHOULD REMOVE PREVIOUS CONDUITS
        ghev.Component.PingDocument += self.RemoveHandler

def UpdateGeometries(self, objects, Arg):
    self.temp_objects= []
    self.temp_objects_bbox= None
    for object in self.geometries:
        # UPDATE THE BOUNDING BOX
        temp_obj= object
        if temp_obj != None:
            if self.temp_objects_bbox == None: # FIRST OBJECT
                self.temp_objects_bbox= temp_obj.GetBoundingBox(False)
            else:
                self.temp_objects_bbox= rhG.BoundingBox.Union...
        self.temp_objects.append(object)

def RemoveHandler(self, sender, e):
    # REMOVE THE CONDUIT
    rhDisP.PostDrawObjects -= self.MyDisplayPostDrawObjects
    rhDisP.CalculateBoundingBox -= self.MyDisplayCalBBox
    # MY DISPLAY POST DRAW OBJECTS
    for i, geometry in enumerate(self.temp_objects):,
        newmat= rh.Display DISPLAY MATERIAL()
        newmat.Diffuse= c[MIN(len(c)-1,1)]
        newmat.Transparency= t[MIN(len(t)-1,1)]
        # ASSIGN TEXTURE MAPS TO SELECT MESHES
        if i in select:
            newmat.SetBitmapTexture(fD[MIN(len(fD)-1,1)],True)
            newmat.SetTransparencyTexture(fA[MIN(len(fA)-1,1)],True)
        try:
            s.Display.DrawMeshShaded(geometry,newmat)
        except:
            pass
    rh.Display.DISPLAY MATERIAL.Dispose(newmat)

def MyDisplayCalBBox(self, sender, e):
    if (self.temp_objects_bbox != None):
        e.IncludeBoundingBox(self)
#=================================#
def main():
    if len(geo) > 0:
        # CUSTOM DISPLAY DRAW
        CustomObjectDraw(geo, select)
    else:
        print("Typical Display")

OUTPUTS: None

Replaces the default display pipeline for given meshes allowing customization of colors and texture maps (transparency and bitmap).

44 Calls the CustomObjectCreate class to create the pop-up screen, cf:4.5.5.
83 Create a new display material.
84 Assign the diffuse and the transparency (85) values to the material from input lists c and t.
89 For select meshes (full size screens, the rest are small arrows) assign bitmap and transparency (90) maps from input lists fD and fA.
93 Mesh faces and edges are drawn separately. Edges can be drawn with variable lineweights by e.DisplayDrawMeshWires() or omitted (95).
### 4.5.5 createPopUpScreen

```
#======================================================
class CustomObjectCreate():
    def __init__(self, anchors, size, sel):
        vwpt = doc.ActiveDoc.Views.ActiveView.ActiveViewport
        oPln = vwpt.GetFrustumNearPlane()[1]
        self.meshes = []
        for i, pt in enumerate(anchors):
            dim = 6
            if i in sel: dim = size
            xVal = (dim/2)*1.5
            rectX = (0,10,xVal,xVal,-xVal,-xVal,-10)
            rectY = (-10,-20,-20,-20,-dim,-20,-dim,-20)
            pl = rh.GPolyline(4)
            scrPt = vwpt.WorldToClient(pt)
            for x,y in zip(rectX,rectY):
                diagPt = vwpt.ClientToWorld(scrPt+rh.GPoint2d(x,y))
                pl.Add(diagPt.PointAt(.99))
                pL.Add(pt[0])
                m = rh.Mesh.CreateFromClosedPolyline(pL)
                rh.GPolygon.Interval(0,7.5), rh.GPolygon.Interval(0,7.5), rh.GPolygon.Interval(0,7.5)
                if i in sel:
                    w = pt[2].DistanceTo[pl[5]]
                    h = pt[2].DistanceTo[pl[3]]
                    mapping = rh.GPolygon.Interval(0,x), rh.GPolygon.Interval(0,h), rh.GPolygon.Interval(0,y)
                    oPln, rh.Interval(0, w, rh.Interval(0, h), rh.Interval(0, y))
                m.TextureCoordinates.SetTextureCoordinates(mapping)
                self.meshes.append(m)
    #======================================================
```

---

**Draws meshes to the camera frustum as an information graphic display.**

**Reference:** Human plugin, Render Mesh to Screen

[http://www.food4rhino.com/project/human](http://www.food4rhino.com/project/human)

---

05 This code exists in the same GhPython component as 4.5.4.

10 Get the plane of the near face of the frustum from the active viewport.

14 The vertical dimension of the pop-up screen, *dim*, has a default size 6, which will draw an arrow.

18 The relative x- and y-coordinates (19) of the pop-up screen perimeter points in pixels. Note that screen y-coordinates count downward from the top so lower y-values are higher on the screen.

22 Collect the screen position of the anchor point pt.

25 To define the mesh perimeter points, find their screen position (relative to scrPt) and project back into 3d space. This returns a diagonal line along the ray of the camera projection that will be seen as a point.

26 Select a point on that line very, very close to the near frustum and save it to the polyline that will define the perimeter.

29 Reassign the origin of oPln to the corner of the pop-up screen so that the texture maps can be calibrated to display properly.

32 Calculate the width and height (33) of the pop-up screen.

34 Create a TextureMapping using oPln for orientation and the mesh width and height for scale. Non-selected meshes (i not in sel) will not have a texture map, so the scale is irrelevant and filled with a default value.

36 Assign the TextureMapping to the mesh.

---

A gradient of hatches combined with color value displaying the hours of solar incidence *solHrs* (**4.4.3**, 24). The popup screens are located at random, selecting one of them, draws a large screen showing the hours of direct sunlight at that location.
Project 1: Meshes (Leshan)

This project is located to the northwest of Leshan in Sichuan province, approximately 140 km south of Chengdu. The site is currently covered with small, but very steep hills averaging between 40 and 50 meters tall.24 Between the hills are extensive rice paddies. The intent of this design exercise is to inscribe a mesh that organizes contextual inputs into a guide that defines urban orientations and the location of building volumes in alignment with ecological forces as an alternative to the imposition of a modernist grid. Special attention is given to the hydrology25 of the site and the attempt to integrate new construction alongside productive agricultural land. This motivates the use of the hillsides as the primary building zones, leaving the rice paddies below intact.26

The result is a multilayered meshwork that responds to different concerns. One layer addresses building orientation, another erosion control, another access and circulation. Each layer is derived from the same environmental data, however it is filtered through differing criteria and assessed by different values. The sensitivity to one input or another can be adjusted and a new set of meshes produced through various interactive methods, many of which will reappear in the following projects as well.

74 Leshan site data courtesy of Turenscape.
75 Base vector map of China above (and in subsequent chapters) via WikiMedia (user Whangpi).
Source: http://commons.wikimedia.org/wiki/File:China_Blank_Map_with_Province_Names.svg
76 Yu, “China’s Water Crisis.”
76 Yu, “Beautiful Big Feet: Toward a New Landscape Aesthetic.”

Figure 4.7 Detail of Project 1: localized orientation vectors
Figure 4.8 Site condition and boundary
Figure 4.9 Orthophoto of Leshan from Google Maps
§4.6 Positioning

The foundation of the project is to construct a geometric armature of points, anchors from which subsequent operations can be launched. As we established earlier, the initial move may be arbitrarily chosen, but, once incorporated into the sequence, will resonate through contingently implicated relationships. Logically, the first operation should identify a distinguishing feature of the available data and make explicit the many ways which that data is differentiated. The distinguishing feature of this site in Leshan is the topography, represented in the file as topographic curves cut at one meter increments.

Adaptive pointcloud

The irregularity of the ground, however, does pose a challenge. The slope descends in every direction, with saddle points, plateaus, and concave recesses. The topographic curves are wildly different lengths, with no common alignment, so subdivision of these curves into points produces a very poor coverage of the site. To account for the complexity of this ground and to attempt to incorporate such complexity in the results, this project begins by calculating lines of flow down the steepest paths of the hillsides. This now familiar method is commonly used in landscape urbanism projects as a regulating line in much the same way as Peter Eisenman once used superimposed traces, as a dense set of lines whose intersections and orientations contribute to generative geometries. This use was largely limited to a visual application, as the convergence of lines was not in any sense controlled or interrelated. The method in §4.6.1 improves on this by giving the code the ability to collapse traces into a single path when they grow too close and to insert new traces when they diverge too far from their neighbors, yielding a much more complete site coverage (Figure 4.10). Additionally, each point is sorted among adjacent points on the same level and related to the points above and below it on the same trace through the .src and .trgt pointers in the TopoPoint class (§4.4.3). The result is a very adaptive pointcloud with control over density averages and extrema that is highly indexed. The illustrations that follow feature an iteration with around 5,600 points, stepping down the hillsides in four meter increments initially spaced 10 meters apart horizontally. The traces are adjusted whenever the spacing expands to more than twice or compresses to less than half this measurement.
4.6.1 TopoFlow

Calculates the path continuously perpendicular to a series of topographic curves: uphill, this gives the steepest climb; downhill, the path of water runoff over a landscape.

01 Import Python modules

07 TopoPoint class: code elided, see \( \approx \) 4.4.3

37 A function to calculate the angle between the current slope tangent and the next possible point. Angles near \( 90^\circ \) occur on a constant slope; extremely acute or obtuse angles suggest landscape anomalies that must often be removed or adjusted for.

44 A function that saves points to the dictionary as TopoPoints, while setting preliminary associated values.

45 Because the \textit{trgt} value has not yet been calculated, the TopoPoint is instantiated with a boolean value. After rows are added, the previous rows \textit{trgt} values are updated accordingly (\( \approx \) 48).

47 This function also creates lines in the dataTree of geometry, \textit{segTree} for preview display.

53 This function handles exceptions where new source points need to be added into the list because adjacent paths have diverged too far from one another (relative to \texttt{maxSpan}).

57 Rhino’s handling of curve geometry introduces discontinuities in the parameter \((s)\) space of curves at the endpoints, even on closed curves. The simplest condition is when the interval needing new source points is located entirely on the interior of the curve with no discontinuities. Complex conditions occur when the interval crosses the endpoints, for which separate strategies are needed for closed (\( \approx \) 70) and open (\( \approx \) 84) curves.
def main(jList, ptD):
    for j, jCrv in enumerate(jList):
        ptD[jCrv].append((j, jCrv, jList[jCrv].TangentAt(jList[jCrv].t[1])))
        ptD[jCrv].sort()
        for i in range(len(jList), 1, -1):
            for k in range(len(jList), i):
                if kCt < 3:
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # LOOP FOR ELEVATION (i) TO NEXT ELEVATION (i+topoInt)
    for i in range(len(jList), 1, -1):
        for m, nCrv in enumerate(jList):
            if nCrv in enumerate(jList):
                for k in range(len(jList), i):
                    if kCt < 3:
                        tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                        tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # LOOP THROUGH CURVES ON NEXT LAYER AND PREPARE EMPTY LISTS
    for j, jLat in enumerate(jList):
        if jLat < 3:
            for k in range(len(jList), i):
                if kCt < 3:
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # SEARCH FOR NEXT POINT IN PATH
    for i, j in enumerate(jList):
        if jLat < 3:
            for k in range(len(jList), i):
                if kCt < 3:
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # LOOP THROUGH CURVES AT CURRENT ELEVATION
    for j, jCrv in enumerate(jList):
        kCt = len(jList)
        for m, nCrv in enumerate(jList):
            if nCrv in enumerate(jList):
                for k in range(len(jList), i):
                    if kCt < 3:
                        tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                        tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # SAVE AS TUPLE (DISTANCE, INDEX, POINTS, tVAL)
    kCt = len(jList)
    for j, jLat in enumerate(jList):
        if jLat < 3:
            for k in range(len(jList), i):
                if kCt < 3:
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # FIND CLOSEST FROM SET OF kCpts
    kCpts.sort()
    for i, kCpt in enumerate(kCpts):
        if kCpt.trgt:
            # CHECK VECTOR ANGLE
            if abs(ang - (math.pi / 2)) < (math.pi * maxAng) or kCt < 3:
                tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # LASTLY WITHIN i-LOOP, CHECK PROXIMITY BETWEEN NEIGHBORS
    for j, jLat in enumerate(tmpPts):
        for k in range(len(jList), i):
            if kCpt.trgt:
                # CHECK SPACING AND SAVE tmpPts
                if i < jLat:
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                    tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

    # LOOP THROUGH POINTS IN tmpPts
    for j, k in enumerate(jList):
        if jLat < 3:
            for k in range(len(jList), i):
                if kCpt.trgt:
                    # CHECK SPACING AND SAVE tmpPts
                    if i < jLat:
                        tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))
                        tmpPts.append((jList[jCrv].TangentAt(jList[jCrv].t[1])))

The function main() executes the bulk of the analysis. The pBDefDataDist of TopoPoint goes by the variable name ptD within this function. The dictionary is at the method as it was defined in the sticky class.

The next layer could have more than one topography curve, so the script must calculate the closest point from each (loop at :131) and then check that the nearest in this set is within the distance threshold (%142) and the angle threshold (%146).

If no point fits the criteria, change the TopoPoint's tVal value to False to signal the end of a path.

This is a complicated line of code for those not familiar with the condensed style of Python. The map() tool in Python runs a function—here ClosestPoint()—on a list (or other sequence) without having to invoke loop syntax.

Because this function takes a Point3D and the jList contains a different datatype, TopoPoint, the sequence field is filled with a generator expression that extracts the Point3D at x3 and from each TopoPoint (a generator expression operates like a list comprehension but returns each element individually instead of a list of all elements and is denoted by parentheses rather than brackets).

The list returned by the map() operation is then itself used as the basis of a list comprehension that makes a tuple of the t-Value, tVal of the calculated closest point and its index within the list, n (which corresponds to its position in jList).

By sorting the resultant list of tuples, the points can all be referenced in sequential order along the curve regardless of the order they were added (%196, %197).
#ENOUGH SPACE ON BOTH SIDES: ADD PT AS CONTINUOUS

if (distCk[0] > divDist*adjRto[0]) and ... 
SetPoints ptD.d[i+topoInt,j,kPt,trgtCt,True) 
trgtCt += 1

#ONLY PREVIOUS POINT TOO CLOSE

elif (distCk[1] > divDist*adjRto[0]): 
if not skipDbl == srtdPts[k][1]: 
halfPt= kPt.xyz+ jLst[kPrev].xyz/2 
transF= kPt.tan+jLst[kPrev].tan/2 
ptD.d[i+topoInt][j].append( ... 
trgtCtt += 1

#ONLY NEXT POINT TOO CLOSE

elif (distCk[0] > divDist*adjRto[0]): 
if not skipDbl == srtdPts[k][1]: 
halfPt= kPt.xyz+ jLst[kPrev].xyz/2 
transF= kPt.tan+jLst[kPrev].tan/2 
ptD.d[i+topoInt][j].append( ... 
trgtCtt += 1

#BOTH POINTS TOO CLOSE

else: 
kPt.aggr = .01 
ptD.d[i+topoInt][j][trgtCt-2].aggr= .01 
skipDbl= kNext 

#TOO MUCH SPACE UNTIL NEXT POINT

if (distCk[1] > divDist*adjRto[1]) and adjBool[1]: 
newPts= [] 
for PP3W in enumerate(newPts): 
trgtCtt += 1

#DIRECTION OF COUNT WILL ALWAYS BE IN DIRECTION OF CURVE

t1= 0; t2= 0 
t1= dupSrtdCrvs[j].ClosestPoint(kPt.xyz) 
t2= dupSrtdCrvs[j].ClosestPoint(jLst[kNext].xyz) 

#FIX SIMPLE tVal==0/1 CONFUSION

if t2[1] == 0: t2= (True, 1) 
if t1[1]ct2[1]: 
newPts=InsertPoints(dupSrtdCrvs[j].True, t1[1], t2[1])); 

def newPts= [] 
for n, nPt in enumerate(newPts): 
ptD.d[i+topoInt][j].append(TopPoint(nPt, ... 
trgtCtt += 1

#CHECK FOR CURVES WHICH HAVEN'T BEEN INTEGRATED YET

for j, jCrv in enumerate(artdCrvs[i+topoInt]): 
if len(ptD.d[i+topoInt][j]) == 0: 
KCT= max(jCrv.GetLength()//divDist,2) 
rdivV= jCrv.DivideByCount(KCT, False) 
for k, kPt in enumerate(rdivV): 
ptD.d[i+topoInt][j].append(TopPoint(kPt, ... 
return ptD
# SET DIRECTION OF FLOW FROM topoInt
zSt = int(topoRng[0]); zEn=int(topoRng[1]+1)
if (topoInt<0):
zSt=int(topoRng[1])
zEn=int(topoRng[0]-1)

# DICTIONARY (key=ELEVATION, value=LIST BY CURVE (points nested))
ptDict=scriptcontext.sticky['ghDefaultDict']().defaultdict(list)
segTree=gh.DataTree[gh.GLine]()

# TEST THAT topoDict IMPORTS CORRECTLY
print("Landscape spans " + str(len(topoDict.d.keys())) + " meters")

# MAIN FUNCTION TAKES DICTIONARY OF SORTED TOPO CURVES
mainReturnDict= main(topoDict.d, ptDict)

# PYTHON FORMATTED OUTPUT
pyDict= mainReturnDict

# GH FORMATTED OUTPUTS
agg= DataTreeFromJKDict(mainReturnDict.d, 0)
trgt= DataTreeFromJKDict(mainReturnDict.d, 2)

OUTPUTS: pyDict, agg, trgt, segs

This sets the direction of analysis (uphill or downhill).
Initializing the ghDefaultDict from the sticky class.
Call the function main().
For the sake of brevity the trivial function DataTreeFromJKDict() has been omitted from this text; 4.7.1 (41) includes a similar function using list comprehensions.

Figure 4.10
The above image shows of the TopoFlow script run without adding or collapsing lines. The inflection of the topography quickly directs most traces together while ignoring the majority of the site. Below, with these functions added, the coverage is much more complete and even.

Figure 4.11
The radius of the circles illustrates the agg values at each TopoPoint. When the flow direction is set downhill this approximates water runoff. The zoom-in shows how agg values combine when paths are joined together as well as the introduction of new paths at lower levels on the slope.
Accumulative information

A sophisticated base of information provides not only internal consistency and easier representation, but also allows the datapoints to develop additional information—and thus differentiation—for themselves. For example, the control of internal relationships and adjacencies in the TopoFlow code allows for cumulative runoff values to be calculated along the length of the trace, and combined when two traces are collapsed together. Naturally, a consistent and thorough datastructure (in combination with a receptive object class) also makes it possible to synthesize analyses that take place at different moments in the model.

§4.7 Localization and Spatialization
Developing locality

At each point, a number of quantitative or geometric analyses are calculated—the average solar incidence; slope magnitude, direction, and normal vector; total runoff; distance and direction to nearby roads, the river, or rice paddies; an area polygon defined by a planar voronoi diagram—and applied to the point in its metadata. Some of these quantities are calculated as byproducts of §4.6.1, while other are added through later operations. In §4.7.1 we show how the proximity of various site features and infrastructures are recorded in the point. Here, a set of polyline curves denotes the edge of rivers, streets, and agricultural fields. A vector is created from each point to the nearest curve in each category. The amplitude of these vectors is remapped according to the graphs from §4.4.1 and saved in the .attr attribute allowing us to consider mediated influence—the radius of influence of a highway is not necessarily equal to than that of a small road—and to control how that influence manifests itself in a given location.

Inclusion

Referring back to the object ontology developed in chapter 2, we should note that such localization involves the association of extensities to the the landscape object such that this information is included within the apperceptions of the object. Again, the individual object is not identically defined by its qualities. Rather than necessary components, the qualities themselves are actualizations of the point in the environment through which the landscape also enacts itself procedurally “as a relation of knowing, perception, and apprehension.” More precisely this differentiation is a process of interaction, because it elides the distinction between subject and object or between figure and ground. It is for this reason that we link the interactive dimension to questions of epistemology. And yet, this is not to say that localization is an empirically driven process. The landscape is not at all a pre-existing, continuous, objective space but an assemblage of discrete monadological objects. As each individual comes to occupy a point of view so also does the cultivated assemblage that is the environment.
This function measures the distance from each point to geographic features such as roads, rivers, rice paddies, and highways and maps them onto functions of influence.

10 List comprehensions make it simple to evaluate the test point against the entire set of curves.

13 Then the found points are zipped together with their distance measure and sorted (14) to determine the nearest. This sequence is similar to the one used in 4.6.1 (16).

22 The distance to the closest point is mapped onto a value by the GraphMapper component from 4.4.1.

37 The attrVec values are attached to an object method of the TopoPoint (29) so there is no need to save the result of the map() function into a list variable.

42 Here the list comprehension is used to output the attractor vectors to a DataTree for visual confirmation.

49 Reducing the number of global variables by placing operations within functions that limit variable scope is one way of making functions more memory efficient.
§4.8 Instantiation

Procedural identities

The development of object qualities is thus also a movement toward the question of identities. Or rather, to avoid suggesting the introduction of any essential definitions, it may be better to say that it introduces a process of identification. Identification in place of identity places the emphasis on the acts of receiving, interpreting, and translating information in ways that are also contingent on the actors that are identifying and their means of querying. Identifying objects is an extensive process of naming or categorizing, separating a gradient of similar objects into discrete selections which can be associated with other roles, functions, or behaviors as instances of a given type. These instantiations reveal themselves as "exploratory or interpretative responses" to inquiry, and are projected back onto the object as situated and contingent tendencies rather than permanent, static properties. As such, they can help the designer align aspects of the site to unit operations within the model.

Categorizing distributing, filtering, and sorting

Like McHarg’s method of overlapping maps to filter out unsuited territory and to reveal regions of potential, this project constructs a series of filters to sieve through the data and identify certain patterns of spatial relations and quantitative values. In the case of the Leshan site, we want to instantiate a set of situated orientations that each follow the same evaluation process, but with adjustments to the parameters that define different attitudes toward the ground. Each operation will produce a set of connected territories with a unique form and configuration. Five open categories are proposed to correspond to the different types of attractor geometry and differing environmental requirements. For each point created in \(4.6.1\) a fitness value is calculated from the input parameters. The code in \(4.8.1\) details how values for attractor strength, accumulation of water runoff, solar incidence, and overall slope are given unique weighted significance for each category and the two highest values recorded. Following this (\(4.8.2\)), an orientation plane is calculated, again from a list of weighted inputs according to the dominant fitness type and the disposition of these planes compared to interlink the mesh when appropriate. The explicitness of these processes requires the designer to anticipate certain formal or spatial decisions—a slab building may imply an orientation that follows the topography at a constant level, while steep slopes suggest terraced massings—in the early stages of the design process and to test the repercussions of these decisions in the model.
class CalcTypeFitness(pt, t, kSol, kHyd, kSlp):
    tVec, thy, tsol, tSlp = t

    # INCREASE FITNESS VALUE FOR HIGHEST ATTRACTOR VALUE
    vLng= [v.Length for v in pt.attr]
    pt.fit[vLng.index(max(vLng))/-2.5]

    # ATTRACTOR MAGNITUDES AS TYPE FITNESS FACTOR
    thrChk= 0
    for pyHF in enumerate(pt.attr):
        # ADD TO FITNESS VALUE FOR HIGH-VALUE ATTR
        if vLng[m] > tVec:
            thrChk+= 1
    # SUM CUMULATIVE .attr VECTORS AND SAVE TO .ornt AFTER LOOP
    if vLng[m] > 0:
        pt.ornt[0]= vecOrnt

    # IF NOT TWO HIGH ATTR; ADD FITNESS FOR EROSION CONTROL
    if WKU&KNSW¿W>@ WKU&KNSW¿W=
    # SOLAR INCIDENCE AS TYPE FITNESS FACTOR
    if pt.sol!=180:
        for P¿W in HQXPHUDWHSW¿W
            # RUNOFF AS TYPE FITNESS FACTOR
            for P¿W in HQXPHUDWHSW¿W
                # SLOPE ANGLE AS TYPE FITNESS FACTOR
                if pt.slp:
                    slpAng=-math.degrees(math.asin(pt.slp.Z/2))
    # SLOPES AS QUARTILES OF THRESHOLD
    else:
        pt.typ[0]=- 10

    ptOut.append(kPt.xyz)
    vecOut.append(kPt.ornt[0])

# LOOP THROUGH ALL POINTS IN ptDict
for i, jKey in enumerate(ptDict.d):
    for k, kPt in enumerate(ptDict.d[jKey]):
        vLng= [v.Length for v in kPt.attr]
        maxNdx= vLng.index(max(vLng))
        for n, fit in enumerate(kPt.fit):
            fit[m]= kHyd[m] * (kPt.aggr-thydy)
            if pt.slp:
                slpAng=-math.degrees(math.asin(pt.slp.Z/2))
                if slpAng>tSlp*1.5:
                    kPt.type[0]= kPt.type[1]= 10
            else:
                # SLOPES AS QUARTILES OF THRESHOLD
                ptOut.append(kPt.xyz)
                vecOut.append(kPt.ornt[0])

        pyDict= ptDict
        pt= ptOut
        vecOut= vecOut

4.8.1 calcFitnessValues

Fitness values are calculated for different types of use. The list inputs are coefficients attached to sliders allowing the user to customize how sensitive each category is to environmental inputs (below, Figure 4.12).

The t variable is brought into the class as a list of all four threshold values. For clarity and legibility it is immediately split into four individual named variables rather than referred to by list indices.

The fitness values, fit, are calculated by summing a number of incremental adjustments.

After the fitness values have been calculated, the maximum values are saved as the points primary and secondry types in .typ.
def MergePoints(pt, nchr, thr, mLst):
    for n, nPt in enumerate(nchr):
        if nPt.DistanceTo(mPt) < thr:
            nchr[n] = mPt
    mLst.append(mPt)

# FOR EACH POINT
for iKey in enumerate(ptDict.keys()):
    for j in enumerate(ptDict[iKey]):
        for k in enumerate(ptDict[iKey][j]):
            kPt = ptDict[iKey][j][k]
            kPt.nchr[]
            typ = kPt.typ[0]

            # COMPUTE ORIENTATION AND LINK ANCHORS
            if kPt.alp and typ > 0:
                vecTemp = rhG.Vector3d(kDir[typ][0]*kPt.alp) + (kDir[typ][1]*kPt.tan) + (kDir[typ][2]*kPt.attr[typ]) + (kDir[typ][3]*vecSol)
            X = nDim[typ].X; Y = nDim[typ].Y; Z = nDim[typ].Z
            X = nDim[typ].X; Y = nDim[typ].Y; Z = nDim[typ].Z
            kPt.nchr[0].PointAt(X, Y, Z)
            kPt.nchr[1].PointAt(X, Y, Z)
            kPt.nchr[2].PointAt(X, Y, Z)
            kPt.nchr[3].PointAt(X, Y, Z)

            # CHECK PREV
            if x[typ] == "Mirror":
                kPt.nchr[0].PointAt(X, Y, Z)
                kPt.nchr[1].PointAt(X, Y, Z)
                kPt.nchr[2].PointAt(X, Y, Z)
                kPt.nchr[3].PointAt(X, Y, Z)

            # CHECK DOWNHILL
            if x[typ] == "Cross":
                kPt.nchr[0].PointAt(X, Y, Z)
                kPt.nchr[1].PointAt(X, Y, Z)
                kPt.nchr[2].PointAt(X, Y, Z)
                kPt.nchr[3].PointAt(X, Y, Z)

        # FOR EACH POINT
        for iKey in enumerate(ptDict.keys()):
            for j in enumerate(ptDict[iKey]):
                for k in enumerate(ptDict[iKey][j]):
                    mPt = ptDict[iKey][j][k]
                    nchrOut.append(mPt)
                    typ0.append(kPt.typ[0])
                    ptOut.append(kPt.xyz)

pyDict = ptDict
pt = ptOut
t0 = typ0
n = nchrOut

corner = pyDict, pt, t0, n

Each point is given a primary orientation based on its currently assigned type (4.8.1), some anchor points are placed on an oriented plane, and anchors that overlap with neighbors are compressed to the same location to create an interlinked mesh.

These operations would normally be repeated for the secondary orientation, but have been omitted here for space.

Each influencing Vector is multiplied by an input coefficient and combined into an aggregate orientation. Vectors have all been unitized elsewhere, so their length coming into this script is 1.

The coordinates seem to be mixed up here because the Plane ornt[1] is aligned vertically with the World ZAxis as its YAxis (30). This eliminates the need to calculate an orthogonal Vector which this Plane will produce anyway.

"Mirror" and "Cross" (39) are options for adding four anchor points. The attempts here is to limit the number of inputs required, so each set of anchor points can be built from a single vector input, nDim, rather than separately for each anchor.
§4.9 Concept of Model

Introduction

At this point it would be beneficial to reflect more critically on the specific meaning of ‘model’ that we want to advance and its relation to the category of the interactive. Alain Badiou writes that "the model is that which allows us to think through participation," providing a useful entry point to the analysis he presents in *The Concept of Model*. Here, Badiou is specifically discussing (logico-)mathematical models so some caution is called for before applying his words to the much less rigorously circumscribed use of model in urban design or architecture. However, there are enough similarities that we can absorb these arguments as commentary, if not as definitive statements.

To precisely follow Badiou’s usage, the model would designate the way of presenting an urbanism’s organization as such, the collection of ways by which architecture might participate in the urban. Thus, a model would comprise the framework for resolving urbanism and computation (that is, it would cover the scope of this entire thesis). Such criteria would seem to be far too expansive to be applicable to individual urban design models, but we will argue that a thoroughly consequent computational model constitutes these concerns as in a microcosm, making this approach appropriate to individual models as well.

Types of models

On a more limited level, Badiou identifies two primary groups in the use of models: abstract models and material assemblages. The first are scriptural objects, assemblages of hypotheses held together by a common code; the second includes graphs, diagrams, physical models and automata. It is apparent that the computational model includes aspects of all of these modes and thus participates in a general concept of modelness. As we have discussed in this chapter, the necessity to “spatially present non-spatial processes in a synthetic fashion” through graphs, datascapes, or other means is significant aspect of engagement with computational models, and while physical models may not always be employed, the spatial visualizations on-screen fall within Badiou’s definition of “realiza[ing] formal structures, that is to transfer scriptural materiality to another ‘region’ of experimental inscription.” Procedurality as a medium also encompasses the definition of automata as the class of models that “aims to imitate behaviours.” Finally, the entire model is formally recorded in code, each command of which operates as an individual hypothetical unit.

Badiou’s purposes regarding models

Badiou’s goal throughout *The Concept of Model* is "to isolate the scientific – i.e., logico-mathematical – concept of model from its notional envelopment by the categories of bourgeois epistemology" or, put another way, to interrogate the relationship between empiricism and formalism in order to establish that the modeling function "is not an *a priori* formal science grounding the empirical sciences’ access to reality but rather the paradigmatic instance of a productive experimental praxis." This insight is born of a subtle distinction made in order to prevent the conflation of the production of a model and “the technical regulation of concrete processes” in such a way that would obscure the materialist history to which the model relates. For these reasons, Badiou disapproves of the epistemological modeling of economic (which propounds a contingent, “integrated technical image” as though it were an “atemporal necessity”) as well as cybernetics’ naturalized epistemology (which becomes lost in an idealist “structure of structures”).
An inventive praxis

Instead, Badiou emphasizes the artificial or “irreal” character of the model as an experimental enchainment,109 “wholly assembled”111 in such a way that its mechanics are rendered more transparent. Doing so, one is able to decouple the model from empiricist concerns with the result of emphasizing its productive and inventive dimension. In this way, the model is freed from the scrutiny of proof, as such, given over to the “inventive freedom of artifice”112 whereby the model bestows objects their universality and their limits.113 Although the standards of assessing the model are changed the standard is no less high: resemblance alone can no longer be assumed to be sufficient, but the conditions of correspondence between the model and the real must be created and explicitly defined alongside the model.114

Representation and formalization

This brings us to the second, related point of emphasis: to establish formalization as an operation that distances the model from a representational function.115 Badiou’s equation of formalization in mathematical modeling with the tracing out of an ontology strengthens this argument: if an ontology were only assessed for its ability to analogize a situation, it would be “subservient to some prior concept of what is at stake in analysis”116 and lack the force of thought that is inherent to the category of ontology. For Badiou, then, the relationship of the model to the real must not be representational, but rather the presentation of a unity that exists only within the formal system of the model.117 This is to say that the defined correspondences between model and object do not proceed from predefined empirical existence, as such, but are actively created. The representational mode is not productive, but exhausts itself,118 whereas formalization invents new identities and unities. This is consistent with the position advanced in the previous chapter that procedural simulation could not be reduced to conventional representational categories but was instead a creative presentation of particular scenarios and modes of engagement.119

Participation

According to Badiou, models have traditionally (and naively) been interpreted to flow alongside scientific inquiry in a way that, at its worst, can obscure the facts of their construction and disguise an ideological framework as a natural process.120 The separation of the model from representational purposes and toward rhetorical ones21 avoids this obscurantism and embraces the idea of “thinking through participation.” At the same time, this moves the model into a much more interesting position as a product of inquiry as well as a means of production.122 “The productive value of formalization lies in its double inscription … that of using and reproducing certain knowledges … at the same time as constructing specific models to produce new knowledge”123 While an overall
model of urbanism is what we endeavor to define, individual, experimental models can, in practice, anticipate these developments, leveraging the gap between implicit understanding of urban processes and their eventual formalization.\textsuperscript{124} Brassier emphasizes that it is through this combination of “retrospective causality” and “anticipatory intelligibility” that the model becomes an engine of differentiation.\textsuperscript{125} Similarly, within the individual model, parameters are anticipated or put forward before a fixed value can be entirely decided on and these conditions can be recursively modified based on their subsequent interactions. Badiou’s definition of a model requires this progressive differentiation because “no formalisation can claim to encompass the totality of the consequences of the event it draws upon”\textsuperscript{126} and so the initial conditions must be revisited again and again.

§4.10 Conclusion

Projective models

Finally this brings us to two conclusions. The first is the need to acknowledge and even emphasize the fact that the model “cannot be mistaken for its empiricist representation or conflated with an ambient scientific worldview, a diffuse ideological distillate synthesized from various scientific disciplines,”\textsuperscript{127} but is a constructed, formal system that occupies a particular point-of-view. At the same time, the model includes a level of flexibility and indeterminacy such that, despite its imminent autonomy, it is also contingently situated and inclusive. This necessitates that a computational model must not be merely representative of empirical data, but projective in a way that exceeds systematization and is open to the effects of competing contingencies.\textsuperscript{128} To the extent that interaction involves the application of procedurality to the task of incorporating external information and localized details within an aggregate organization, it must be a model that enables interaction

Architecture as urban interface

Secondly, we posit that in lieu of representational similarity, the model and the real are linked through a correspondence of analogous operations. In this conceptualization of the city, the urban realm operates as an untotalizable, but structured, organization: one which is itself highly responsive to local perturbations and which enables individual participation via certain prepositional modes. This pushes architecture into the position as the medium through which we can singularly act on\textsuperscript{129} and participate in the urban realm. Architecture then can be conceptualized as an interface onto the urban realm that anticipates and helps bring into being a collective urban form, and that is also retroactively acted on by its environment and context.
Bibliography:


§5.1 Generative Ontology

Definition of objects and forms

Where the interactive addressed epistemology and how interaction reveals procedural logics, the generative concerns the ontological dimension of objects themselves and their internal definition. In chapter 2 we described an object-oriented ontology featuring an irreducible withdrawn interior\(^1\) that achieves objecthood when individuals assemble together and their inter-relationships “attain operational closure … capable of encountering perturbations as information in terms of their own endo-consistency.”\(^2\) Additionally, the monadic interior of their own object is a constantly changing texture of “temporary appurtenances and provisional possessions.”\(^3\) In contrast to purely relational ontologies,\(^4\) this “continuous development of form”\(^5\) is defined by the limits of the individual\(^6\) and as its inflections. However developments can be instigated by interactions between objects as well as intra-actions. Graham Harman describes this occurrence as a phenomenon where the encounter between two objects draws out aspects that might have previously been entirely interiorized\(^7\) or which were only dimly perceived. This potential for encapsulated behaviors to manifest surprising reactions and the capability of objects’ extensive qualities to provoke reactions combine as a generating function that catalyzes new situations.

Means of interaction

The extension of one object over its neighbors will be a major topic of investigation in this chapter,\(^8\) in particular the implication of diffusely redistributing the design agency within the situation, as well as the reception of exterior stimulations, and the preparation of the object to receive these influences through encapsulated behaviors. The reciprocal relationship that forms between encapsulation and extension forms an interesting inside-out tension alluded to by Deleuze’s reference to a “double antecedence”\(^9\) that exists between the monad and its world. A similar occurrence can be observed among digital entities as Latour makes clear in his description of the monadic qualities of digital profiles. In this example, the individual’s contexts—organizations, institutions, associations—are included in that individual as features or qualities,\(^10\) while at the same time, the context, as an object itself, also contains the individual as a component member.\(^11\) In this example, neither object is subsumed into the other nor can either be functionally replaced by the other. This exact situation will arise later in the code examples (\(^=5.6.1, 5.6.2\)). The argument behind Latour’s digital monad is that the reversibility of the monad\(^12\) prevents analysis from drifting to generalizations when dealing with large groups but forces it to move “from particular to more particulars.”\(^13\) As a result, context or networks do not belong to “a second level added to that of the individual, but exactly the same level differently deployed,”\(^14\) which closes off the possibility of an all-encompassing totality.\(^15\) In place of

---

1. Bryant, The Democracy of Objects. p32
2. Ibid. p273
3. Deleuze, The Fold: Leibniz and the Baroque. p79
5. Deleuze often uses the terms Form or Identity for what we are calling objects.
7. Ibid. p48
8. cf §5.5 ‘Extension’
9. cf §2.6 ‘Environment’ note 97
10. “The set of attributes—the network—may now be grasped as an envelope—the actor—that encapsulates its content in one shorthand notation.”
11. Ibid. p593
12. Ibid. p599
13. Ibid. p599
14. Ibid. p593
15. In fact, Latour goes so far as to suggest that organizations are simply side effects of interpreting datasets: “the notion of ‘context’ might be as much an artifact of navigational tools” and “institutions … are just a trajectory through data starting from a different entry point in the database.”
16. “Every time an entity is associated with a new monad, it’s individualized through the previous associations gathered by that monad”
17. “In proposing such a navigation we move away from the dream of simulation and prediction and explore another path, that of description where the added value is no longer the power of prediction, but the progressive shift from confusing overlaps to successive clarifications of provisional wholes.”
a centralized control, each overlap (or extension) of an individual acts on the other to further individualize it.\textsuperscript{16} Attention to the ecology of interactions gives us both a better understanding of each individual object and of the emergent complexities of the plan as extensive series.\textsuperscript{17}

\section*{§5.2 Multiagent systems}

\subsection*{Encapsulation}

The operative character of the digital monad is established by the definition of an interior realm that may contain a set of qualities or behaviors. By encapsulating these properties, the object closes them off somewhat, detaching or veiling their operation from the external world.\textsuperscript{18} The obvious benefit of encapsulation is that it removes the need for oversight of these operations and for an overseer to guarantee them, which simplifies the external relationships and frees up the designer to consider other concerns.\textsuperscript{19} Behaviors can be addressed through a design stance rather than requiring a technical description of every interaction.\textsuperscript{20} Furthermore, “the ability of the monads themselves to assess and to calculate” on their own establishes a “background of ‘calculable forces’”\textsuperscript{21} that confers a rich multiplicity and heterogeneity that simply cannot be achieved through direct control.\textsuperscript{22} From a programming perspective, encapsulation is a common feature of object-oriented programming as we have seen in the previous chapter,\textsuperscript{23} but the concept can be extended further if we consider each computational object as an autonomous agent.\textsuperscript{24}

\textbf{SEEK}

In pure cases of multiagent systems, the autonomy of agents would be complete—each agent an entirely distinct computer system\textsuperscript{25}—necessitating mechanisms for communication, synchronization, and interpretation between agents.\textsuperscript{26} An intriguing take on this scenario was proposed by Nicholas Negroponte and Leon Groisser’s Architecture Machine Group as a continuation of the microworld research of \textit{URBAN5},\textsuperscript{27} and exhibited at the SOFTWARE exhibition in 1969.\textsuperscript{28} This project consisted of a grid of five hundred cubic blocks\textsuperscript{29} on a 5×8 foot tabletop. The blocks could be moved or repositioned one at a time by a robotic arm mounted on a gantry above the tabletop.\textsuperscript{30} Constantly disrupting the configuration of this environment were a number of agents in the form of “a small colony of gerbils.”\textsuperscript{31} The central conceit of \textit{SEEK} was that the gerbils’ actions stemmed from an encapsulated intentionality, and though these intentions were inaccessible to the programmer, the computer could learn the generative pattern behind the gerbils actions by interpreting and reacting to the transformations that occurred.\textsuperscript{32} The arm was equipped with sensors to read the current position of the blocks,\textsuperscript{33} which it compared to its internal model. Small disturbances could be realigned to the grid, while larger discrepancies were noted and reacted to more deliberately. In addition to accepting the alterations wrought by the gerbils, the software could also propose new configurations and generate layouts,\textsuperscript{34} either to “purposely correct or amplify gerbil-provoked dislocations.”\textsuperscript{35} The interesting aspect of this setup is that there was no direct communication between the agents, but instead each communicated solely by acting on the environment and interpreting the traces of actions left by others.

\subsection*{Learning from multiagent systems}

The animal/machine division in \textit{SEEK} effectively illustrates many of the characteristic traits of multiagent systems. Because discrete agents are limited to local knowledge and agency,\textsuperscript{36} perceptions, interpretations, and models of the environment can vary widely: “the fact that agents may observe different things makes the world partially observable to each agent, which has various consequences in the decision making of the agents.”\textsuperscript{37} Such multiplicity and incompleteness can be an advantage especially when data or expertise is already unevenly distributed or when the environment is “open, or at least highly dynamic, uncertain or complex.”\textsuperscript{38} However, in design contexts, it is quite unlikely to find scenarios that truly require multiagent systems. Much more common is a complex object-oriented system that has been constructed to simulate a multiagent system in order to model a problem on societal metaphor. These simulations can avoid many of the technical challenges of implementing multiagent systems by having some common centralized
An agent is a computer system that is situated in some environment, and that is capable of autonomous action in this environment in order to meet its design objectives.

Wooldridge, An Introduction to Multiagent Systems. p15

"It ran six programs: Generate, Degenerate, Fix It, Straighten, Find, and Error Detect, used to randomly lay out, reconfigure, align, and correct the blocks environment, using its arm and plastic attachments to stack, move, and vibrate the blocks into place"


"In most domains of reasonable complexity, an agent will not have complete control over its environment. It will have at best partial control, in that it can influence it."

Wooldridge, An Introduction to Multiagent Systems. p16

"agents may observe data that differ spatially (appear at different locations), temporally (arrive at different times), or semantically (require different interpretations)."

Vlassis, A Concise Introduction to Multiagent Systems and Distributed Artificial Intelligence. p2


Figures 5.1; 5.2
SEEK, 1969.
Burnham, SOFTWARE - Information Technology: Its New Meaning for Art. cover, p9
control system for basal functions such as synchronizing actions on a common run-time clock or ensuring that communications between agents are broadcast successfully. These approximations of multiagent systems are natural given that object-oriented methodologies are among the common starting points for the analysis and design of multiagent systems and so many of the lessons of multiagent systems are still applicable. For example, agent behavior can be separated into reactive and proactive capabilities, the combination of which can prove difficult to successfully balance. The advances made in multiagent systems, where this balance cannot be effected by with a central control intelligence but must be tested and negotiated, can push object-oriented models to incorporate additional flexibility and autonomy into their definitions.

§5.3 Diffuse agency

Multiagent approximations

Among the most applicable agent-based methods are models that use entities that move across the environment analyzing, marking, or modifying the ground to produce traces of reactive organization. At their simplest, these may simply be purely reactive path-finding elements. A more complex scenario instrumentalizes the agents’ paths to modify the environment, allowing agents to communicate with one another indirectly, transforming a naïvely aggregative logic into a stigmergic one. With the use of large agent populations, these models can be categorized as swarm intelligences. According to Neil Leach, swarm urbanism involves seeding design intent into a set of autonomous design agents which are capable of self-organizing into emergent urban forms and “to encode intelligence into urban elements and topologies.” Swarm intelligence is often seen as a useful way to replicate or work within informal urban settlements because it reproduces the localized logics and adaptation that occur in the absence of a guiding masterplan.

A similar but somewhat simplified modeling technique is based on the use of cellular automata. Cellular automata models “consist of an array of cells, each with a finite set of possible states … The state of the cells evolves synchronously in discrete time steps as a function of its state and a set of rules which relates the cell to other cells in the system.” Each cell then acts as a fixed actor, which is typically distributed within an orthogonal grid and whose state responds to the state of its neighbors following a set of transition rules. In the simplest systems, the cell states are boolean values, but the number of state values can be increased to give more nuance to the simulation, for example, to project multiple land-use classes. In the same way, other facets of the ‘conventional’ model can be changed to introduce more complex conditions or to model conditions in a more intuitive way. Among these modifications: the definition of neighbor can be expanded beyond directly adjacent cells to effect more dispersed influence; transition rules can be layered as multi-pass operations. To modulate the model granularity, multiple cellular automata operating at different spatial or temporal scales can be linked, or the cellular automata can be embedded within an agent-based model or have a multiagent system overlaid on itself. The many variations on the model illustrate the fuzzy boundary between a multiagent system and a cellular automata, but even in its simplest forms cellular automata are parallel, discrete and dynamic computational frameworks, where “the rules built in to the model are replicated in all the discrete components of the model” and thus a potential approximation of a multiagent system.

Hybrid and distributed agency

In both cases described above, the desire is to leverage the operative and instrumental nature of the diagram and to multiply its mechanisms—not as a plan générateur, but “as a modulator of synthesis” in a generative environment. Instead of a centralized control of power, and a singular structure, “their internal relationships are transposed: moved part by part into the new organizational context resulting in distributed agency and “emergence exploration” as Negroponte put it. This exploratory attitude toward emergence is significant, because it points beyond the configuration of new objects to the structure of the parameter space that is assembled and which can support the formation of other objects within itself. It is this extension and weaving together of internal identities into an environment that cultivates new assemblages and enables new actualizations that will constitute the generative function.
Couclelis mentions many of the directions which urban models may find it productive to deviate from conventional cellular automata practices.


Typically with a weighted decay at greater distances, this is implemented both within the cellular lattice.

Slager, de Vries, and Jessurun, “Methodology to Generate Landscape Configurations for Use in Multi-Actor Plan-Making Processes.” p10

or by absolute distance:

Li and Yeh, “Urban Simulation Using Principal Components Analysis and Cellular Automata for Land-Use Planning.” p345

Coates and Derix, “Parsimonious Models of Urban Space.” p338


Dijkstra and Timmermans, “Towards a Multi-Agent Model for Visualizing Simulated User Behavior to Support the Assessment of Design Performance.”

Popov, “Generative Sub-Division Morphogenesis with Cellular Automata and Agent-Based Modelling.” p168

Coates and Derix, “Parsimonious Models of Urban Space.” p339

“the mediating ingredient of the diagram derives not from the strategies that inform the diagram, but from its actual format, its material configuration. The diagram is not a metaphor or paradigm, but an ‘abstract machine’ that is both content and expression. This distinguishes diagrams from indexes, icons and symbols. The meanings of diagrams are not fixed. The diagrammatic or abstract machine is not representational. It does not represent an existing object or situation, but it is instrumental in the production of new ones.”

van Berkel and Bos, More. p324-325

Le Corbusier, Towards a New Architecture. p45

Allen’s entire essay on the diagram is relevant here, but will be treated at more length later, cf §8.4 ‘Lines of Flight’

Allen, Diagram 3000 (Words). p9

Allen, “Mapping the Unmappable: On Notation.” p22


as cited in:


DeLand, Philosophy and Simulation: The Emergence of Synthetic Reason, p5
Project 2: Parcels (Beijing)

Qinglonghu is located on the remote periphery of Beijing just outside the sixth ring road around the reservoir for which it is named. The area is characterized by its location in the foothills of the western mountains and this project proposes to occupy these foothills,\textsuperscript{70} densifying the village there rather than converting farmland to developable land.\textsuperscript{71} The specific site of focus, above the small village of Xinkaikoucun, is approximately 1000×500m in plan on a sharply rising terrain with an elevation change of 80m.\textsuperscript{72}

The project begins with a mesh already derived from a topographic analysis (Figure 5.6) and will investigate the generative potential in addressing individual parcels of land as discrete agents with associated traits and behaviors. By staging interactions between neighboring parcels, including the construction of larger assemblages, the project will shed light on how the extension of objects over one another can negotiate a bottom-up urban strategy. In particular, we are interested in drawing parallels between landscape and architectural uses, rather than separating the two disciplines, and how strategies linking the development of new housing to the ground might establish resistances to the complete engulfing of urban villages by the formal city while still increasing density and quality of living.\textsuperscript{73}

\textsuperscript{71} Lai et al., “Rethinking Property Rights and Industrial Development.” p62-63  
\textsuperscript{72} Qinglonghu site data courtesy of Turenscape  
\textsuperscript{73} Shannon et al., “Reconsidering Village in the Expanding City – Taihu.” p121
Methods which return integers can be nested together in chains. Thus:

- `myMesh.eNE[myMesh.eNE[p]]` returns the index of the edge forward two
  in the case of a triangular face this equals `myMesh.eNE[p]`.

- `myMesh.splitCtri[f]` adds a new vertex `v` above the centroid
  of face `f`, then redefines `f` using this new vertex and one edge `eNE[p]`.
  New triangular faces are added on the remaining edges and all other
  methods (their edge and vertex data) are updated accordingly.
§5.4 Discretizing the ground

Half-edge mesh implementation

We will not introduce mobile agents until the next chapter; in order to focus on the aspects of extension and encapsulation, this project will restrict itself to stationary, location-based agents, defined as parcels of land. A mesh of polylines derived from a topographic analysis, similar to those created in Project 1, was drawn on the site. Certain segments of this mesh were coded as roads and pathways, but the rest of the site was left undefined. In order to convert this mesh into discrete units, we will first translate the various segments into a half-edge mesh datastructure (Figure 5.6).\textsuperscript{74} The half-edge mesh is so called because each segment, or edge, in the mesh is represented by two elements oriented in facing directions along the edge. The half-edge mesh datastructure contains a list of all of the half-edges, vertices, and faces (or enclosed loops) in the mesh and uses pointers to identify relations between them. Half-edges, for example, record the indices of the vertex at their origin, the face which they belong to, the half-edges before and after them in the sequence of that face, and their twin half-edge.\textsuperscript{75} Each vertex records the index of one edge that has that vertex as its origin, while each face also records one of its constituent half-edges.\textsuperscript{76}

First among the advantages of this datastructure is the ability to forgo a predefined geometric structure and adopt to irregular and unordered configurations while still maintaining excellent command of adjacencies and network relationships.\textsuperscript{77} Additional benefits include the low memory levels required, the ability to quickly locate a point with regard to the mesh, and the flexibility to travel across it either along edges, vertex-to-vertex, or across faces using the half-edges’ twins to identify the bordering face.\textsuperscript{78}

Agency of the ground

As a datastructure, the half-edge mesh doesn’t exhibit any particular agency in itself, it is simply a record of existing geometry. However, we will elaborate on this substrate to construct a multiagent system with agents that over time will calculate a tendency toward and fitness for potential land-use from their geographic and geometric properties, will seek to influence their neighboring parcels according to local behavior diagrams, and will form assemblages with their neighbors when they are in agreement. This project will effectively satisfy the definition of a cellular automata: the faces are discrete entities with state values that change over time in response to itself and adjacent faces.\textsuperscript{79} However, in most aspects of its implementation, the procedures chosen will diverge substantially from the properties of a classical cellular automata, generating a more complex ecology as we will detail in the following sections.
This code organizes a list of line segments into a Half-Edge Mesh Datastructure, with edges, vertices, faces, and the necessary pointers saved as methods of the HEM class.

Note: This version anticipates a mesh which is roughly parallel with the world XY-plane, which it uses for sorting the edges radiating out from a vertex into anticlockwise order. Intersections which do not occur at an endpoint will not be included in the half-edge mesh.

Reference: de Berg et al., Computational Geometry Algorithms and Applications. ch.2

\[5.4.1\] HalfEdgeMesh

```python
def _init_(self, *):
    self.e=*
    self.eOV=[]
    self.ePE=[]
    self.eNE=[]
    self.eIF=[]
    self.f=[]
    self.fIE=[]
    self.eDep=[]

def EdgeDirection(eCurr,m):
    #FIND DIRECTION OF EDGE
    deg=0
    if eCurr.Direction.X>0:
        deg= 90* eCurr.Direction.Y/abs(eCurr.Direction.Y)) % 360
    else:
        if eCurr.Direction.X>0 and deg >= 180:
            deg = (360+deg) % 360
        m.nDep.append(deg)
        m.nDep.append((deg + 180) % 360)

def CheckSortVertex(eCurr,t,dom,div,m,ct):
    #VERTEX SPATIAL KEY AND CHECK DUPLICATES
    ptCk= rh.Point3d(eCurr.PointAt(t))
    #POINT KEY
    keyX= math.Floor((ptCk.X-dom.00) / (dom.01-dom.00)/div))
    keyY= math.Floor((ptCk.Y-dom.00) / (dom.01-dom.00)/div))
    keyZ= min(max(keyX,0), div-1)
    keyY= min(max(keyY,0), div-1)
    #CHECK ALREADY ADDED VERTICES
    for val in dict.slkeyx,keyY):
        if isinstance(val,tuple):
            vta,div=tval
            if ptCk.DistanceTo(vtx)<.01:
                break
            m.nV.append(ptCk)
            dict.update({keyX,keyY}.append((ptCk,ct[0])))
            ct=ct[0]+1, ct[1])
        return ct

def LoopFace(m,n,init,ct,tree):
    if n!=init:
        m.n[ct].append(n.e[n].PointAt(0))
        tree.Add(m.n[n].PointAt(0), GH_Path[ct])
        n.e[m]=ct
        LoopFace(m,n,sNE[n],init,ct,tree)
        #===================================================================

INPUTS: seg as List of Curve, bnd as UVInterval, sbdv as Int

# IMPORT MODULES
import Rhino as rh
import Rhino.Geometry as rg
import Grasshopper as gh
from Grasshopper.Kernel.Data import GH_Path
import scriptcontext
from collections import defaultdict

#============================================================

5.4.1 HalfEdgeMesh

class HEM HALF-EDGE MESH DATASTRUCTURE AND POINTERS

class HEM:
    """Half-Edge Mesh""

    def _init_(self): ...

    def EdgeDirection(eCurr,m):
        ... # FIND DIRECTION OF EDGE ...

    def CheckSortVertex(eCurr,t,dom,div,m,ct):
        ... # VERTEX SPATIAL KEY AND CHECK DUPLICATES ...

    def LoopFace(m,n,init,ct,tree):
        ... # LOOP FACE ...
```

This code organizes a list of line segments into a Half-Edge Mesh Datastructure, with edges, vertices, faces, and the necessary pointers saved as methods of the HEM class.

Note: This version anticipates a mesh which is roughly parallel with the world XY-plane, which it uses for sorting the edges radiating out from a vertex into anticlockwise order. Intersections which do not occur at an endpoint will not be included in the half-edge mesh.

Reference: de Berg et al., Computational Geometry Algorithms and Applications. ch.2

\[5.4.1\] HalfEdgeMesh

```python
def _init_(self, *):...
    self.e=*
    self.eOV=[]
    self.ePE=[]
    self.eNE=[]
    self.eIF=[]
    self.f=[]
    self.fIE=[]
    self.eDep=[]

def EdgeDirection(eCurr,m):
    #FIND DIRECTION OF EDGE
    deg=0
    if eCurr.Direction.X>0:
        deg= 90* eCurr.Direction.Y/abs(eCurr.Direction.Y)) % 360
    else:
        if eCurr.Direction.X>0 and deg >= 180:
            deg = (360+deg) % 360
        m.nDep.append(deg)
        m.nDep.append((deg + 180) % 360)

def CheckSortVertex(eCurr,t,dom,div,m,ct):
    #VERTEX SPATIAL KEY AND CHECK DUPLICATES
    ptCk= rh.Point3d(eCurr.PointAt(t))
    #POINT KEY
    keyX= math.Floor((ptCk.X-dom.00) / (dom.01-dom.00)/div))
    keyY= math.Floor((ptCk.Y-dom.00) / (dom.01-dom.00)/div))
    keyZ= min(max(keyX,0), div-1)
    keyY= min(max(keyY,0), div-1)
    #CHECK ALREADY ADDED VERTICES
    for val in dict.slkeyx,keyY):
        if isinstance(val,tuple):
            vta,div=tval
            if ptCk.DistanceTo(vtx)<.01:
                break
            m.nV.append(ptCk)
            dict.update({keyX,keyY}.append((ptCk,ct[0])))
            ct=ct[0]+1, ct[1])
        return ct

def LoopFace(m,n,init,ct,tree):
    if n!=init:
        m.n[ct].append(n.e[n].PointAt(0))
        tree.Add(m.n[n].PointAt(0), GH_Path[ct])
        n.e[m]=ct
        LoopFace(m,n,sNE[n],init,ct,tree)
        #===================================================================

INPUTS: seg as List of Curve, bnd as UVInterval, sbdv as Int

# IMPORT MODULES
import Rhino as rh
import Rhino.Geometry as rg
import Grasshopper as gh
from Grasshopper.Kernel.Data import GH_Path
import scriptcontext
from collections import defaultdict

#============================================================

5.4.1 HalfEdgeMesh

class HEM HALF-EDGE MESH DATASTRUCTURE AND POINTERS

class HEM:
    """Half-Edge Mesh""

    def _init_(self): ...

    def EdgeDirection(eCurr,m):
        ... # FIND DIRECTION OF EDGE ...

    def CheckSortVertex(eCurr,t,dom,div,m,ct):
        ... # VERTEX SPATIAL KEY AND CHECK DUPLICATES ...

    def LoopFace(m,n,init,ct,tree):
        ... # LOOP FACE ...
```
**BODY OF CODE**

```python
segCt=0
hem=HEM()
ef=[]

#****FOR EACH SEGMENT, ADD EDGE AND ITS REVERSE (twins) TO edgeList****
for i, iSeg in enumerate(seg):
    if iSeg.PointAtStart.DistanceTo(iSeg.PointAtEnd)>0:
        #ADD EACH SEGMENT AND ITS REVERSE
        hem.e.append(rh.Line(iSeg.PointAtStart, iSeg.PointAtEnd))
        hem.e.append(rh.Line(iSeg.PointAtEnd, iSeg.PointAtStart))
        #ADD TWIN EDGE INDEX
        hem.eTE.append(segCt*2)
        hem.eTE.append(segCt*2+1)
        segCt+=1

#NEGATIVE INDEX VALUES INDICATE NOT YET ASSIGNED VALUES
hem.eIF.append(-10) for x in xrange(2)
hem.eNE.append(-10) for x in xrange(2)
hem.eVE.append(-10) for x in xrange(2)

EdgeDirection = hem.e[10] == hem.e[-2]

/*
NEGATIVE INDEX VALUES INDICATE NOT YET ASSIGNED VALUES
ADD TWIN EDGE INDEX TO edgeList
for i in range(len(edgeList))
    if not (edgeList[i].i == 0 or edgeList[i].i == len(edgeList)-1)
        #ADD TO end(enumerate(edgeList))
        edgeList.append(edgeList[i])

/*
NEGATIVE INDEX VALUES INDICATE NOT YET ASSIGNED VALUES
ADD TWIN EDGE INDEX TO edgeList
for i in xrange(len(edgeList)-1)
    edgeList[i].append(edgeList[i])
*/

**CONSTRUCT VERTEX LIST FROM LIST OF HALF-EDGES**********
for i, iSeg in enumerate(hem.e):
    vCt=h.DataTree[h.face(iSeg,0,-1)]
    vTree=hem.DataTree[h.face(iSeg,0,shb)]

**FROM EACH VERTEX, SORT EDGES RADIALY*********
for i, iSeg in enumerate(vTree):
    nxran=len(vTree)
    nAtStart=0
    nAtEnd=0
    list.sort()
    for j, (deg,ndx) in enumerate(list):
        if deg==0:
            if nAtStart+1<nxran:
                list.sort()
            else:
                list.sort()

**FROM EACH HALF-EDGE, LOOP AROUND FACE**********
for i, iSeg in enumerate(hem.e):
    if hem.e[i].i<0:
        hem.f.append(i)
    hem.f.append(i)
    hem.f.append(hem.e[i].i.PointAt(0))
    fTree.Add(hem.e[i].i.PointAt(0), GH_Path(hem.f))
    hem.f.append(i)
    hem.f=sbdv
    vTree=hem.DataTree[h.face(iSeg,0,-1), faceCt]...}
```

- `segCt=0`
- `hem=HEM()`
- `ef=[]`
- 

#****FOR EACH SEGMENT, ADD EDGE AND ITS REVERSE (twins) TO edgeList****
for i, iSeg in enumerate(seg):
    if iSeg.PointAtStart.DistanceTo(iSeg.PointAtEnd)>0:
        #ADD EACH SEGMENT AND ITS REVERSE
        hem.e.append(rh.Line(iSeg.PointAtStart, iSeg.PointAtEnd))
        hem.e.append(rh.Line(iSeg.PointAtEnd, iSeg.PointAtStart))
        #ADD TWIN EDGE INDEX
        hem.eTE.append(segCt*2)
        hem.eTE.append(segCt*2+1)
        segCt+=1

#NEGATIVE INDEX VALUES INDICATE NOT YET ASSIGNED VALUES
hem.eIF.append(-10) for x in xrange(2)
hem.eNE.append(-10) for x in xrange(2)
hem.eVE.append(-10) for x in xrange(2)

EdgeDirection = hem.e[10] == hem.e[-2]

/*
NEGATIVE INDEX VALUES INDICATE NOT YET ASSIGNED VALUES
ADD TWIN EDGE INDEX TO edgeList
for i in range(len(edgeList))
    if not (edgeList[i].i == 0 or edgeList[i].i == len(edgeList)-1)
        #ADD TO end(enumerate(edgeList))
        edgeList.append(edgeList[i])

/*
NEGATIVE INDEX VALUES INDICATE NOT YET ASSIGNED VALUES
ADD TWIN EDGE INDEX TO edgeList
for i in xrange(len(edgeList)-1)
    edgeList[i].append(edgeList[i])
*/

**CONSTRUCT VERTEX LIST FROM LIST OF HALF-EDGES**********
for i, iSeg in enumerate(hem.e):
    vCt=h.DataTree[h.face(iSeg,0,-1)]
    vTree=hem.DataTree[h.face(iSeg,0,shb)]

**FROM EACH VERTEX, SORT EDGES RADIALY*********
for i, iSeg in enumerate(vTree):
    nxran=len(vTree)
    nAtStart=0
    nAtEnd=0
    list.sort()
    for j, (deg,ndx) in enumerate(list):
        if deg==0:
            if nAtStart+1<nxran:
                list.sort()
            else:
                list.sort()

**FROM EACH HALF-EDGE, LOOP AROUND FACE**********
for i, iSeg in enumerate(hem.e):
    if hem.e[i].i<0:
        hem.f.append(i)
    hem.f.append(i)
    hem.f.append(hem.e[i].i.PointAt(0))
    fTree.Add(hem.e[i].i.PointAt(0), GH_Path(hem.f))
    hem.f.append(i)
    hem.f=sbdv
    vTree=hem.DataTree[h.face(iSeg,0,-1), faceCt]...}

- This and the following lines use generator expressions to add placeholder values into the eIF, eNE, and eVE lists two at a time (in xrange(2)). These values will be assigned later and out of order as the script uncovers the various relationships.
- Generator expression to populate the bin sort dictionary with a square array of keys as lists.
- A variable for storing the face geometry from f in a grasshopper datastructure for output.

This counter saves the number of vertices in its first term and the second term denotes (by a negative number) if a new vertex is added or (by a positive number) the index of the existing vertex if a point at that location has already been saved (59, 53).

sEdge is a temporary list that saves the indices of all the edges leaving each vertex and their degree so that they can be sorted radially (:116) allowing the code to determine their succession (:118, 119).
§5.5 Extension

Agent definition

Like the TopoPoint (§4.4.3), the FaceAgent class includes a list of attributes for storing metadata values related to geometric properties and topographic analysis.\(^{80}\) In contrast to that class, which mostly served a recording function, the FaceAgent also includes within itself the functions that define its behaviors. The result is that the functional code that directs the flow of the script becomes very short, but following the script now means reading from two different locations: first the function call in the sequential code §5.5.1 (§40) and then moving to the object definition §5.5.2 (§64ff). As objects accumulate multiple functions, some additional attention to how these functions interact with one another is required given their dispersion within the code. Wooldridge describes a layered architecture for agents wherein functions are given an order of priority, with the initial functions also acting as gates to the subsequent functions.\(^{81}\)

The FaceAgent class has three main behaviors. It first calculates a series of values that indicate fitness for various uses based on its environment.\(^{82}\) When one of the fitness values reaches a sufficiently high value, the face takes on the designated use and begins to influence its neighbors. Finally, if the face has maintained a use for a certain duration, it can look to its neighbors and attempt to form clusters with them. The transition from one function to the next is regulated by the value .\textit{lyr} attribute which is evaluated at the end of each function (§113ff).\(^{83}\)

The .\textit{use} value in this example is comparable to the state variable in a cellular automata, it gives the physical outcome and the overall pattern. Though cellular automata states are most often defined by a simple patterns of adjacent cells, there is a precedent for defining the transition rules through a synthesis of multiple criteria.\(^{84}\) Here, we calculate the fitness for each face by adding up incremental adjustments throughout §5.5.2 in the \textit{ftAdjst} variable and then applying the sum of those adjustment values to the face’s \textit{ftns} values. In this process, the initial frames are more sensitive to the static site properties like the slope, solar incidence, and elevation (§76), while the later frames respond more to dynamic inputs such as the position of attractor curves that can be controlled by the user while the simulation is running (§92ff). The matrix of the adjustment values and the way they impact uses differently can also be controlled by the user through a spreadsheet as illustrated in Figure 5.7.\(^{85}\)

---

80 These have been elided in the presentation of §5.5.2 for considerations of space, but will be identified in the code comments when they appear if they have not been explained yet.

81 Wooldridge, An Introduction to Multiagent Systems. p97

82 These uses are simple volumetric or surface categories:
1) high-rise of around 8 stories;
2) mid-rise of 3-5 stories;
3) low-rise of 1-2 stories;
4) open green space that is potentially a courtyard or garden space; and
5) open void space that remains unused.

83 While the transition criteria is determined within the object functions, this implementation is unlike Wooldridge’s example in that the actual layer transition is evaluated outside the object domain (36).

84 An example of multiple criteria:
Katoshevski, Arentze, and Timmermans, “Simulating Urban Dynamics Using a Combination of Cellular Automata and Activity-Based Models” and multi-step transition rules:
Coates and Derix, “Parsimonious Models of Urban Space.”

85 Importing the spreadsheet data from a .csv file is described in detail in Appendix.5.1.
5.5.1 enactFaceAgents

```python
# 5.5.1 enactFaceAgents

0...4  INPUTS: mf, coeff as List; dnnAttr, openAttr as List of Curve;

   hardReset, stepClock, run as Boolean
   import scriptcontext as sc

5  #-------------------------------T R N 0  5  SETUP-------------------------------#
   if "ftnCounter" not in sc.sticky:
      sc.sticky["ftnCounter"] = 0

10  # SINGLE FRAME STEP COUNTER IF stepClock
   if stepClock:
      print("step")

15  # RESET VALUES IF hardReset
   if hardReset:
      sc.sticky["ftnCounter"] = 0
   for f in mf:
      f.dur = 0
      f.fitSee = [0, 0, 0, 0, 0]
      t.use = -10
      t.sens = 1
      f.clime = -10
      f.lyr = 0
      run = False

25  #-----------------------------------------------------------------------------
   # graphObj= sc.sticky["sensitivitiesGraphs"]
   # INCREMENT WHEN run
   if run:
      sc.sticky["ftnCounter"] += 1

45  #-----------------------------------------------------------------------------
   # CHECK BEST FIT, UPDATE USE AND LAYER
   fN.CheckUse(graphObj[1].Container.X)
   psyMF =

OUTPUTS: psyMF
```

---

**A frame counter is set up to allow incremental iteration of the code with a timer, slider animation, or through manual prompts.**


05 The frame counter variable ftnCounter is saved to sticky. Note that if more than one counter is being used in a Grasshopper definition, they must take different variable names.

10 The stepClock input is a boolean Button. When triggered, the code will run once.

13 hardReset is also a boolean Button, which resets the frame counter when pushed. The aggregate object properties must also be reset (16-21) or the next running of the script will start from a biased position.

27 run is a boolean Toggle, which allows better stop/start control when timers are attached.

36 Every face begins the simulation with a lyr value of 0. While they remain at this layer, they calculate adjustments to their fitness values each frame (40) through a call to an object function, CalcFitness() (5.5.2).

44 CheckUse() is another object function, not reprinted here, that identifies whether the maximum unfit value is high enough to indicate a use value and to effect a transition to the next lyr.

---

**Figure 5.7**

In order to facilitate user input and adjustment of large quantities of related data, this project uses formatted spreadsheets (LibreOffice .ods files) where values tables of data can be quickly set and compared to one another. The spreadsheet manager can automatically color the cells by a value gradient to allow visual comparison. This replaces an unwieldy set of text panels and sliders as in 4.8.1 and Figure 4.12.

This spreadsheet lists a number of environmental conditions in the first column and asks the user to judge how these properties impact a parcel’s fitness for five different use cases (cf. note 82 above). The FaceAgent CalcFitness() in 5.5.2 will apply these coefficients to the parcel’s unfit values as incremental incentives (positive values) or disincentives (negative values) each frame of the simulation until a suitable use is determined.

Importing and formatting these data files in GchPython is described in Appendix 5.
```
5.5.2 FaceAgent class, CalcFitness

The FaceAgent class is instantiated for each face in the HalfEdgeMesh (=5.4.1) with geometric and topographic metadata (=10.49). This class encapsulates the behaviors the faces will use to analyze their fitness and determine their state transitions. The first function is included here, others will follow (=5.5.4).

66 The coefficient values from Figure 5.7 are imported as a single, nested list and here split into separate (sometimes still nested) lists for improved legibility.

71 The slope coefficients have three gradations (=15, 15-30, and >30). The slopeCh function, all three coefficient lists can be summed up and applied to the ftAdj function, all three coefficient lists can be summed up and applied to the ftAdj list. The 4 value comes from an evaluation of the graphMapper component (=5.5.1 .25) exactly like the one in \(\text{list.78}\). This allows the impact of different properties to change over time.

96 Curve attractors can be introduced into the model during the simulation runtime to produce targeted densification (attr[0]) or voids (attr[1].103). The attr variable is defined as the inverse of \(k\) (=93) so that the attractors begin with low impact allowing the fixed environmental conditions to act and then increase in effect as the others wane. This allows the initial site conditions to influence the placement of attractor curves.

111 When all the fitness adjustments have been added up they are added to the FaceAgent’s ft values.

```
Adjacency behaviors

As the landscape of fitness values develops and parcels with `.use` values emerge, these faces continue to the next behavior layer, `.CalcAdjacency (=5.5.2)` in which their `.use` value influences the cells neighboring it. At this point, the active faces are analogous to the seed of a cellular automata with the added distinction that new seed cells may emerge independent of the adjacency behaviors since the fitness functions continue to run on faces that have not yet established a `.use` value.

One significant departure that distinguishes this model from a classical cellular automata is that the cells do not have a regular shape or distribution. This fact contributes, of course, to the differentiation of parcels in the calculation of fitness values but it also has an effect here. Earlier, in the `FaceAgent` class definition, we compiled a list of data about each face’s direct neighbors in the `.fDeg` attribute (153). Looping through this list, we again apply incremental adjustments to the `.ftns` values of the neighboring cells, following a matrix of patterns (Figure 5.9) that define desirable or undesirable adjacencies for the current face and its current projected use. The spreadsheet for inputting this matrix is formatted to allow the direction of the neighboring face to influence the response, biasing certain orientations, for example to avoid placing a low building in the shadow of a taller one. Other location-based data can also be considered, for example, opening up a view corridor from taller buildings or altering behaviors based on relative elevation. In this way, the abstract behaviors of a cellular automata are translated into more concrete conditions that will likely be more immediately useful to the designer while maintaining the generative, bottom-up complexity of localized behaviors.

---

**Figure 5.9**

Another spreadsheet is used to input the magnitudes of adjacency behaviors. The same process of incremental adjustment is used, but here there is also a spatial condition.

For each use case that a parcel may take (vertical columns) there is a set of adjustments that it applies to its neighbors’ `.ftns` values (horizontal rows) to try to effect a change. The adjustment values will vary depending on the orientation of the neighbor parcel from the current one. These values are arrayed in a compass rose around each `'+'.

If, for example, the current parcel has a `.use` value of 0, it is tending toward a high-rise massing (`HI`) and will read from the first column. Any parcel to the north of it will receive an adjustment to its `.ftns` values of `[-.5, -.25, -.25, -1, 1]`. This is motivated by a desire to discourage mid-rise (`>MI`) or low-rise (`LO`) buildings from being placed in its shadow but to allow the next parcel to also be a high-rise (`>HI`) or a non-garden open space (`>0`).

The `VIEW` value refers to an orientation parallel to an input vector that defines a desirable view, in this case downhill toward the village.

86 König also describes a process where initial seed cells emerge from a generative process, though he appears to maintain a strict division between the two stages.


87 That is, those faces that share an edge with the face in question, in CA terms, the von Neuman neighborhood.

Dijkstra and Timmermans, “Towards a Multi-Agent Model for Visualizing Simulated User Behavior to Support the Assessment of Design Performance.” p224

Batty, “A Digital Breeder for Designing Cities.”
This script calls into action the first layer behaviors, the extension of traits from one face over its adjacent.

2. This script references GraphMapper components like those in 14.4.1 to gradually dampen the effect of adjacency perturbations.

11. For each face in the loop, if the fitness value has crossed the threshold to trigger the behaviors for the first layer (58), calculate the adjacency effects, 5.5.4.

Each face extends a set of modifications to its neighbors' fitness values and checks its own lyr value.

153. The quadrant qd in which the neighboring face is located is calculated from the angle, \( \theta \) (this is the measure in radians from north)

154. Values from Figure 5.9 (coeff) are selected by the current face’s use and the direction of the neighboring face (qd) and added to the adjacent face’s ftv values.

167. Built uses react to view corridors with a radius of two faces (173, 181).

188. If the current face’s use value has persisted, the dur value will raise to signal the second layer behaviors of Cluster formation, 5.6.1.
§5.6 Assemblage Emergence

Cluster coefficients

Finally, we introduce the final layer, which matches adjacent faces against a template to see if they follow the desired pattern of behavior. Where such Clusters form, they are then able to further direct their neighboring faces toward completing the set. Here we have input a target pattern with a central green space surrounded by buildings of various height, taller toward the north and terracing downward to the south (Figure 5.10). The Cluster introduces a new scalar layer of tendencies into the model. Thus, while the method of influence remains incremental adjustment of the \( \text{ftns} \) value (0.21-2.20), the number of relationships that might occur between two faces multiples. Neighboring faces may both belong to the same Cluster, to two different ones, or to none. A face may be added to a Cluster (0.56.2.24), drawn away by a larger Cluster (0.56.1.242), or two Clusters merged together (0.56.2.29). The script can extract their perimeter boundaries (0.56.2.42) or links which could have design uses, such as the placement of garden walls or circulation paths.

Intensional and extensional sets

The inclusion of formal assemblages in the definition of the model raises additional questions about how such assemblages are to be defined. Badiou points out that modern set theory defines inclusion as an extensional function—"that is, the result of a simple collecting together of previously existent elements, which may or may not share any unifying properties." In such a case, an object can be included in a set, any set whatsoever, simply by the willful act of creating the set, rather than belonging to a set because of some inherent property. We should be inclined to favor the extensionally defined set then, in order to preserve the distinction between the assemblage’s internal definition and its external manifestation of qualities. Furthermore, given that each element of the assemblage is itself an assemblage, the concept of a necessary intensionality falls apart when confronted with the complexity of an assemblage’s internal multiplicity. In this example, the Clusters are formed and unformed extensively by the script out of an intensional set, the set of all neighboring faces, in a way that sprawls and extends beyond that passive definition. In this way, set formation is a creative act that acknowledges the agency of naming that "writes a new type of reality," and better catalyzes the generative force action of the virtual.
5.6.1 ClusterFormation

This script enacts the second layer behaviors, the formation of clusters, reinforcing groups of adjacent faces as assembled units.

205 There is not need to form a Cluster if the cluster already exists. Clusters are also interrupted if the shared edge is a road (207).

216 The adjacent use is checked against the template in the first column of (Figure 5.10), if it does not match, the adjustment values are applied to the neighbor (219, 220).

223 If the two faces match the Cluster template, the face must determine whether to add itself to its neighbor’s Cluster (226), annexing a face into a Cluster (238), or merging Clusters (242).
5.6.2 Cluster class

The Cluster class has meta data to identify itself and its component faces and to effect and manage the assemblage of Clusters.

04 The face index is a property of the Cluster just as the Cluster index is a property of the face (31)

07 The Cluster requires an number to ID itself because as Clusters are formed, uniformed, merged, and split, the list structure is too volatile to keep track of them.

15 The Cluster keeps track of its perimeter, the ‘naked edges’. This can be used for visualizing Cluster boundaries, but also for creating garden walls or introducing paths in the site. Recording the naked edges during other transformations occurs in (42ff).

Figure 5.11
Clusters forming and surface uses (rooftop and ground level) begin to be assigned at the last layer of behaviors.
§5.7 Ecology of Interaction

Observing monads

Of course, it is not only the final state that is interesting, but also the process of formation that leads there, particularly as urbanization is not the kind of project where the time of construction and implementation can be ignored. “‘Good’ macroscopic form always depends on microscopic processes,” however behaviors that are truly beyond observation are difficult if not impossible to control, direct, or mediate. For this reason Latour reintroduces Tarde to urge increased study of the “background of ‘calculable forces’” and inter-subjective decisions, and the “fabric of vectors and tensors which defines the attachments of people and assets” in order to erode the division between macro- and micro- actions, and to erase the separation between the levels of the individual and of societies. Neither individual nor assemblage is ever wholly singular, but form only as “temporary aggregates, partial stabilizers, nodes in networks,” which, if they can be tracked, can inform the interpretation of results or the design decisions.

Recording temporal data

For example, Project 2 eventually converges to a stable, static solution and there is a great deal of morphological data to pore over just in this last configuration. However, as in any complex system, the difficulty of correlating this last state to the initial parameters frustrates attempts to explore alternate scenarios. By recording the state values of the model as it goes from frame to frame, formatting it, and graphing the data, we can observe trends in the population of faces and compare different parameter sets asynchronously. This histomap (Figure 5.12) presents a timeline of every face’s use state on the x-axis, and is sorted with the faces whose state has persisted the longest toward the top. The green band at the top of this graph reveals that open green spaces tend to find their final use earlier and the blue built uses are not settled until later in the process, suggesting that a study that compared this result with models that change the proportionate adjacency or fitness adjustment values might be a revealing comparison (Figure 5.13).

Figure 5.12

The histomap displayed in the geometry preview window below the site shows a record of every face state (use) since the beginning of the simulation. Each horizontal line represents the history of one face while a vertical slice through the graph reveals the proportions between different uses across the site at a single point in time. The lineweight increases the longer a face state persists at a single use value, which gives a density effect to the graph.
## 5.7.1 RecordState

**INPUTS:** mf as List; reset as Boolean  
from collections import defaultdict; import scriptcontext as sc  
histData = sc.sticky['gDefaultDict']().defaultdict(list))  
for i, fn in enumerate(mf);  
histData[i].append(fn)  
pyHist=histData  
OUTPUTS: pyHist

## 5.7.2 HistomapData

**INPUTS:** mf as List; histData  
from collections import defaultdict; import scriptcontext as sc  
import Grasshopper as gh; from Grasshopper.Kernel.Data import GH_Path  
import System.Int32  

histoTree = gh.DataTree(System.Int32)()  
for i, fn in enumerate(mf):  
  histData[i].append(fn)  
else:  
  histData[i][i] = 0  
  histData[i][i][i] = IN.use:  
  histData[i][i] = 1  
else:  
  histData[i][i][i] = 0  
  histData[i][i][i][i] = IN.use:  
  histData[i][i][i][i] = 0  

age.append((histData[i][i][i][i]))  

age.sort()  

age.reverse()  
for i, (val, ndx) in enumerate(age):  
  histoTree,AddRange(histData,nds[i], gh_Path(i))  
pyHist=histData  
val=histoTree  
age= [val for val,ndx in age]  
OUTPUTS: pyHist, vals, ages

## 5.7.3 HistomapFormat

**INPUTS:** crv as Curve; vals as DataTree  
import math; import Rhino.Geometry as rhG  
import Grasshopper as gh; from Grasshopper.Kernel.Data import GH_Path  

stp= crv.PointAtStart.DistanceTo(crv.PointAtEnd)//len(vals.Branch(0))  
dir= stp crim TangentAtStart

display=[];  
collat=[];  
wtat=[]  
for j, lat in enumerate(vals.Branches):  
  pt= crv.PointAtStart+(1.25*ct)*rhG.Vector3d.YAxis  
  persist= 1  
  len= len(lat)  
  if lat|num-1|>0 and lat|num-1|<4 and y|<3:  
    for i, use in enumerate(lat):  
      display.append(rhG.Line(pt,pt+dir))  

  use < 0:  
    collat.append(6)  
  else:  
    collat.append(use)  
    if use == lat|num-1| and use!=0:  
      persist= .1  
    else:  
      persist=1  
    wtat.append(math.floor(persist))  
  pt=dir  
  ct+=1  
l= dispLat  
col= collat  
wt= wtat  
OUTPUTS: ln, col, wt

While the frames are advancing, this script saves the values and creates a format for displaying the graph.

After the first frame has passed, the script checks how the current frame compares to the previous frame. If they match, the age resets (12), otherwise it resets to zero (14).

The age list is created with the age values for each key, and the index values, this is then sorted (19) from highest to lowest (20) and is also used to ensure that the data is output in the correct order (22).

The data is prepared to be drawn in the viewport as an array of Curves with lineweight and color values.

A base Curve is used to set the dimension, orientation (05), and position (12) of the graph.

The color is determined by the .use value. Here, only the integer is saved. It will later be used to select a color from a list.

The lineweight corresponds to the persistence of the use. The persistence increments slowly (24) while the lineweight must always be an integer (27).

Figure 5.13  
Graphs comparing values three different simulations: the first runs only the fitness values and a simple version of adjacency behaviors; the second runs the fitness values and the adjacency behaviors shown in Figure 5.9; the third adds to that the clustering behaviors in Figure 5.10.

Figure 5.14 (cover)  
Perspective view on the site after the model has reached a terminal state.
Distribution of Active Cell States
HI: 102  4.1%  
MI: 792  31.6%
LO: 120   5.2%  
  40.9% Σ BUILT
GR: 997  39.8%
Ω: 483   19.3%  
  59.1% Σ OPEN
Total: 2504

Cluster Links from Garden to:
HI: 97   
MI: 1014  
LO: 136   
Total: 1247

Distribution of Active Cell States
HI: 241  8.3%  
MI: 700  24.1%
LO: 400   13.8%  
  46.2% Σ BUILT
GR: 837  28.8%
Ω: 721   24.9%  
  53.8% Σ OPEN
Total: 2899

Cluster Links from Garden to:
HI: 209  
MI: 874   
LO: 361   
Total: 1444

Distribution of Active Cell States
HI: 197  6.6%  
MI: 810  27.3%
LO: 339   11.4%  
  45.3% Σ BUILT
GR: 991  33.4%
Ω: 630   21.2%  
  54.6% Σ OPEN
Total: 2967

Cluster Links from Garden to:
HI: 201  
MI: 1049  
LO: 336   
Total: 1586
§5.8 Enacting Encounters

Virtual

In this effort we endeavor to progress from the evaluation of possible outcomes to the potential for emergent order, and even to approach an image of the virtual being of the model.\textsuperscript{108} The application of generative behaviors to the design elements and the complex extension of their traits over on another develops the virtual being of the objects\textsuperscript{109} by exercising the agency it has encapsulated and unfolding the individuality of these objects through their interactions.\textsuperscript{110} This community or ecology of objects depends on the fact that objects are, on the one hand, discrete individuals, but also are continuously implicated “as a system in which all the parts simultaneously cause and affect one another.”\textsuperscript{111} As an aggregated model of urbanism, there is a connection to Stan Allen’s seminal essay on “field conditions”, wherein “overall shape and extent are highly fluid and less important than the internal relationships of parts, which determine the behaviour of the field.”\textsuperscript{112} However, in his account, the field tends to become a unifying force that overwhelms the objects within it, to the extent that the field “establishes the conditions within which the material will be deployed”\textsuperscript{113} while the material only reacts passively to “register the complexity of the given.”\textsuperscript{114} In contrast, we would prefer to invert the proportionate influence of field and material, with more substantial entities such that each object acts as the ground for the next.\textsuperscript{115} The formal or material properties of these objects would construct a field in a way that may not be continuous, fluid, or smoothly gradated. This parallels the earlier statement that the virtual is a dimension of the individual rather than a pre-individual plenum.\textsuperscript{116}

Complexity of urban encounters

Moreover, the essay’s characterization that field conditions “smoothly accommodated” exceptions and inconsistencies “with the overall order”\textsuperscript{117} is at odds with an approach to urbanism that emphasizes discontinuities, ruptures, and breaks from the actual order of the city—discontinuities that the temporality of assemblages is specifically tuned to recognize.\textsuperscript{118} The open city is not achieved through the avoidance of juxtaposition or in the correlation of differentiation,\textsuperscript{119} but through “the capacity of events to disrupt patterns, generate new encounters with people and objects, and invent new connections and ways of inhabiting everyday urban life.”\textsuperscript{120} The city is not a space of smooth transitions, but frequently consists of sharp demarcations even while it equivocally supports pluripotential uses.\textsuperscript{121} The material of the city is not a self-similar passive register, but is a diverse set of assemblages and societies\textsuperscript{122} with contradictory aims and our models should reflect that. In the next chapter we will look at how reflexivity intensifies the generative behaviors explored here into sociomaterial agency\textsuperscript{123} and the negotiated actualization of urban assemblages.
“Whatever medium you are operating in, you miss the virtual unless you carry the images constructed in that medium to the point of topological transformation. If you fall short of the topological, you will still grasp the possible (the differences in content and form considered as organizable alternatives). You might even grasp the potential (the tension between materially superposed possibilities and the advent of the new). But never will you come close to the virtual.”

Massumi, *Parables for the Virtual*. p134

“the only way an image can approach it [the virtual] alone is to twist and fold on itself, to multiply itself internally … The virtual can perhaps best be imagined by superposing these deformational moments of repetition rather than sampling differences in form and content.”

*Ibid.* p133

Bryant, *The Democracy of Objects*. p104

Allen, “From Object to Field: Field Conditions in Architecture and Urbanism.” p120

*Ibid.* p128

Trumm, “The City as an Object.” p57

Allen, “From Object to Field: Field Conditions in Architecture and Urbanism.” p132

As is advocated by Parametricist Urbanism:


For example, the oscillation between open or built space in Project 2.


cf. §1.6a ‘Inorganic Agency’


§6.1 Intensiﬁying Actualization

Actual–virtual axis

In opposition to the withdrawn virtual being of the object,¹ actualization deals with the qualities and properties that manifest externally. Left as simple expression, the generative process laid out in the previous chapter risks becoming merely an unfolding of potentials and a brief resolution of conﬂicting tendencies until an identity is settled on and a form produced.² This process would not need to invoke the virtual dimension of the object: because there is only a singular situation, the end state is never really in question of developing otherwise, so the complexity of the model is undercut.³ The object’s internal disposition only really becomes integral when changing conditions disrupt the orderly harmonization of the model and the object internalizes these perturbations. This is because the internal workings of the object determine how environment is brought in and translated into information, “any information value the perturbation takes on is constituted strictly by the distinctions belonging to the organization of the [assemblage] itself.”⁴ When the environment is dynamic or rapidly changing, the assemblage is characterized by “strategies of selection or continuance within an environment that they are unable to completely anticipate and which they are certainly unable to dominate or master.”⁵

Intensities

In such cases, there develops a bidirectional inﬂuence between the internal and external, the virtual and actual, the encapsulated and the environment. Deleuze writes that “intensity is the determinant in the process of actualisation”⁶ Intensive properties are often cursorily deﬁned as indivisible properties,⁷ though a more operative deﬁnition is that “differences in intensity, though not in quality, can drive ﬂuxes of matter or energy.”⁸ Because a “key concept in the deﬁnition of the intensive is productive difference.”⁹ For example, the difference between the desired state of an agent and its current condition drives the dynamics of a simulation until the desired condition is achieved. If, however, achieving this goal disturbs other agents there arises a new differential between the satisfaction of one agent and all the others, a condition which the various agents will attempt to equilibrate¹⁰—though only if the disruption of the ﬁrst agent is intensively registered. In this way, intensiﬁcation powers the differential engine that enacts spatio-temporal dynamism¹¹ and differentiation.¹² While extensive processes revealed the identities of singular objects as they related to one another discretely, intensive processes are concerned with the relations within a complex assemblage and the ways that the assemblage’s entities come into convergence in an actual thing.¹³ “We may expand the meaning of the term ‘intensive’ to include the properties of assemblages, or more exactly, of the processes which give rise to them”¹⁴

¹ Deleuze, Difference and Repetition. p208
² “The playing out of those potentials requires an unfolding in three-dimensional space and linear time—extension as actualization; actualization as expression. It is in expression that the fade-out occurs. The limits of the ﬁeld of emergence are in its actual expression.”
³ Massumi, Parables for the Virtual. p35
⁴ DeLanda, Intensive Science and Virtual Philosophy. p65
⁵ cf. §2.4a ‘Information passes and is translated between objects’
⁶ Bryant, The Democracy of Object. p141
⁷ “by virtue of the greater complexity that each environment possesses when compared to the complexity of systems”
⁸ Deleuze, Difference and Repetition. p245
⁹ DeLanda, Intensive Science and Virtual Philosophy. p45
¹⁰ Ibid. p60
¹¹ Ibid. p63
¹² Ibid. p60
¹³ “Spatio-temporal dynamisms, that is, morphogenetic processes exhibiting intensive properties, are processes of individuation”
¹⁴ Protevi, “Out of This World: Deleuze and the Philosophy of Creation.”
¹⁵ cf. §2.2b ‘Being as difference’
¹⁶ “Parts or wholes do not exist any more; they are replaced by degrees for each character.”
¹⁷ Deleuze, The Fold: Leibniz and the Baroque. p47
¹⁸ DeLanda, Intensive Science and Virtual Philosophy. p64
Reflexivity

Therefore we can define the reflexive as the translation between virtual and actual. Reflection “goes the other way from production. It is a matter of … moving from extensity through intensity to virtuality.” The necessity to form this loop emerges because the virtual is composed of multiplicities without qualities but still “the qualities differentiated by virtue of the relations they actualize impose their own requirements, as do the extensities differentiated by virtue of the distinctive points they incarnate.” Intensities straddle the divide as “intensities are implicated multiplicities … which direct the course of the actualization” Protevi argues that the intensive “mediates the virtual and actual,” a mediation “between unexercised power and actualized quality within an individual.”

§6.2 Feedback Urbanism

Ville Cybernetique

In the realm of urban speculation, Nicholas Schöffer was an early experimenter and advocate of applying principles of cybernetics to the question of urban planning and management. Inspired by Norbert Weiner’s cybernetic theories, he envisioned a city integrated with three central computers “constantly receiving data, processing it, and answering its requirements with new information and directives for various urban systems and services.” One of these computers was responsible for “governmental mandates and regulations” but the other two handled more diffuse actions and information, with one monitoring “information and behavior” and the other “modifications and perturbations.” The computer system was to coordinate a self-regulation of every aspect of society from the production and availability of goods and services and affordances for leisure to the sanctioning of new social regulations to head off revolutions. Schöffer’s primary attention, however, was on how these computers could influence the public life through spectacles that filled urban space. The Ville Cybernetique featured plazas located throughout the city, where he proposed large, frame-like towers “that supported moving sculptural elements, light projectors, and speakers,” activating the city with kinetic light and sound performances as directed by the central computer. The towers were also information gathering points that would sense local activity such as sound and light levels or other urban flows, and could vary their projections in response to the current conditions. In this way the regulating impulses of the city would be kept in flux, constantly adjusting itself to the conditions of the urban environment.

Participating in urbanism

Furthermore, while the projections were typically spectacles to be taken in passively, the inhabitants of the city were “sometimes actively programming them and creating new ones.” For Schöffer, the participation of the inhabitants was an integral aspect of the feedback mechanism. His emphasis on ambient experiences rather than more explicit messages points to the significance of the experience of space over communication as the primary motivation in the Ville Cybernetique: “the cybernetic city was a place in which space had been activated as a palpable substance, filled with the transmission of aesthetic and informational ‘matter’. The forms of the structures mattered less than the ‘psychophenomenological’ effects of this ambience, which would serve to immerse the inhabitant in a vast field of perceptual space.” In spite of the centralized processing hub, the Ville Cybernetique valorized these kinds of bottom-up acts of urbanism that also characterized de Certeau’s pedestrians, substituting localized actions in place of global representations.
Brian Massumi, in a suggestive paragraph, describes affect in very similar terms: “What is being termed affect in this essay is precisely this two-sidedness, the simultaneous participation of the virtual in the actual and the actual in the virtual, as one arises from and returns to the other. Affect is this two-sidedness as seen from the side of the actual thing, as couched in its perceptions and cognitions.”

Massumi, *Parables for the Virtual*. p.35

Protevi, “Water.”

Bryant, *The Democracy of Objects*. p.109

Deleuze, *Difference and Repetition*. p.245

Ibid. p.244

Protevi also argues that “we should consider the intensive as an independent ontological register” in part because the intensive disappears in the individualization process but I do not think this is necessary in our case where the virtual is always already individual.

Protevi, “Out of This World: Deleuze and the Philosophy of Creation.”


My translation, the original reads: « gouvernement ordres d’action et de régulation », « traitement des informations contrôles des comportements », « centre de modification et de perturbations »

Schöffer, *La Tour Lumière Cybernétique*. p.97


For example, the tower constructed in Lyon (Figure 6.2) reacts to the coming and going of the metro trains below it.


Ibid. p.55

cf. §1.4c ‘Enaction and Agency’
Project 3: Buildings (Shanghai)

The site in Shanghai is a large area of 143ha spanning both banks of the Wusong River. It includes the Zhongtan and Zhenping metro stations on the #4 and #2 lines of the subway and the elevated track of the #3 line, which passes along the north edge of the site. Directly to the east is the Shanghai Railway Station. This site is currently occupied by high-rise apartment towers densely packed in a semi-gated neighborhood and modeled on typical floor-plan types that are repeated with minimal variation.

This project proposes an alternate plan that activates the buildings themselves in the determination of siting and formal variation. Each building is conceived as an individual agent seeking a desirable location and adjusting its position, and its formal and geometrical properties in negotiation with the other agents. Of particular interest is the motivation of masterplanning decisions by architectural details—both technical limits of building systems and design logics—that demonstrate the freedom of cross-scalar influence in early stages of the design process.

---

30 Site data collected from openstreetmap.org. Importing geometry from an .osm file is described in Appendix 6.2.

31 cf. §1.3a 'Against scalar hierarchies'

32 Zuelzke, Patt, and Huang, "Computation as an Ideological Practice."
§6.3 Mobile Agents
Tropism and stigmergy

Chapter 5 introduced agent-based models with an emphasis on the encapsulation of operational behaviors and data within object classes that activated themselves as opposed to having an external functional control. Mobile agents introduce additional requirements such as determining their position within their environment, sensing their changing surroundings, or communicating with other agents in dynamic, unstructured interactions. In the simplest model, agents decide their movements based on a tropistic response to some property of the environment that varies in density or distribution. Tropism is “a compulsive movement,” a strict rule of cause-and-effect. Yet when subjected to a changing gradient of external stimuli, the agent does not follow a single-minded goal but is continuously redirected, tracing out nonlinear paths through the environment. A purely tropistic agent operates indexically, making legible the field of forces that it reacts to. A slightly more complex agent model is one that is able to act on, add to, or modify the environment either to reinforce its decisions or to communicate indirectly with other agents. A common reference for such agents is found in the stigmergic behavior of insect colonies that build up pheromone trails over well-traveled routes. The stigmergic process displays a positive feedback loop that reinforces behaviors over time.

Determinant and emergent planning

In a more controlled setting, this offers the attractive combination of combining given conditions of a global environmental with a responsive augmentation of that environment. In the context of masterplanning, it is easy to imagine general data, such as overall density plots constituting one facet of an environmental dataset within which individual massing volumes could emerge. In §6.3.1, a simple mobile agent is defined that moves over the site through a combination of its own momentum and external forces which are modulated by a site-derived coefficient. The agent’s position is mapped to a grid of 50×50m cells that have been projected density values assigned to them from a color-coded image (Figure 6.6). The higher the correspondence between the agent’s height value and the desired density the lower the coefficient, suppressing the movement vector and keeping the agent in place. In contrast, a low correlation will magnify the movement vector, moving the agent to a new location more quickly. During the searching process, the agents also subtly modifying the density diagram, reinforcing their position and, over time, increasing the chance of gathering similar agents near them.

33 Or approximations thereof.
34 Wooldridge, An Introduction to Multiagent Systems. p33
35 Verstegen, Tropisms: Metaphoric Animatino and Architecture. p9
36 As previously mentioned this is an effect of localizing the diagram but now the impact of spatial diffusion is multiplied by each incremental step in time.
37 Bonabeau, Theraulaz, and Dorigo, Swarm Intelligence: From Natural to Artificial Systems. p26ff
38 Wooldridge describes an example that demonstrates how controlled stigmergic signals can communicate more specific instructions even without direct contact. Wooldridge, An Introduction to Multiagent Systems. p92-94
39 The external forces are a combination of resistance from site boundaries and the attraction or repulsion of nearby agents.
40 Importing pixel data from a bitmap image is covered in Appendix 6.
6.3.1 BaseAgent

```python
class BaseAgent:
    def __init__(self, pt, i, j):
        self.ndx = [i, j]
        self.tra = pt
        self.trace = rhG.Polyline()
        self.trace.End = (self.x, self.y, self.z)

        self.pln = rhG.Plane(pt, -rhG.Vector3d.YAxis, rhG.Vector3d.ZAxis)
        self.pln = rhG.Vector3d(self.pln.YAxis)

        def Move(self, grid, src, hi, ch):
            grid = self.grid(n0)
            grid = self.grid(n1)

            # TIME BASED COEFFICIENTS
            tmp = GaussFunct(1.25, 0, 100, 0)

            tmp = GaussFunct(1.25, 0, 100, 0)

            # MOVE POINT
            self.x += (tmp*src)*self.force

            # RESET MOMENTUM VECTOR AND EXTERNAL FORCES
            tmpRe = (tmp*src*self.force)

            self.x += tmpRe

            # IDEAL TRACES
            self.trace.Add(self.x, self.y, self.z)

            # UPDATE DENSITY ATTRACTION VALUES
            for i in xrange(1, 4):
                if i == hi:
                    if grid[grid[grid[grid[grid[i]]] < 253]:
                        grid[grid[grid[grid[grid[i]]] == 2]
                    else:
                        for j in xrange(1, 4):
                            if grid[grid[grid[grid[grid[i]]] > 1]:
                                grid[grid[grid[grid[grid[i]]] > 1]
                                grid[grid[grid[grid[grid[i]]] > 1]
                                grid[grid[grid[grid[grid[i]]] > 1]
                                grid[grid[grid[grid[grid[i]]] > 1]
                                grid[grid[grid[grid[grid[i]]] > 1]

A simple mobile agent, this will be the base of the tower (6.4.1) and the main force of movement.

19 The agent is located by a grid cell to make position and adjacency comparisons more efficient.
21 A polyline of the sequential positions (22) traces the path that the BaseAgent has taken (71).
27 BaseAgents can be frozen in place by setting their state to False.
32 The agent’s .vec value is similar to momentum, it records the direction of the last movement.
34 The .force vector gathers up all the external forces pushing the agent in other directions.
37 Links form when an agent is the realm of influence of another.
41 Since the floorplate of the tower will be rather large relative to the distance between agents, it’s helpful to have points for comparison off the agent’s central point to formulate more accurate distance and angle measurements.
56 The movement of the agent will be a combination of its momentum (.vec) and external forces (.force).
64 The base grid (50, 51) is coordinated to an image density map (Figure 6.6, below) that modifies (via the coefficient attrX) the magnitude of the movement vector. As the agent crosses the site it also leaves an impact on this data (73).

A fraction of that movement vector is saved as the .vec value for the next frame, and the .force vector is reset to zero (68).
```
Each building is itself an assemblage of multiple agents, bundled together in a single class.

These levels are saved in a list as an central axis, etc.

Each level exerts a horizontal force on the ones above and below it.
§6.4 Multi-scalar and bidirectional influence

Internal complication

In addition to the interplay between top-down and bottom-up, the reflexive also oscillates between the singular and the plural, and between the interior and the exterior depending on whether we count the assemblage as a multiple of many parts, as an individual monad, or as a component in an assemblage that annexes it.\(^{41}\) Because the assemblage does not erase the individuality of its component elements,\(^{42}\) we should expect a heterogeneous mixture of impulses, trajectories, and motivations within an assemblage. Complex agents will often find themselves caught between contradictory tendencies, to say nothing of multi-agent systems.

In this project, each building is modeled by a series of agents (§6.4.1) linked together at intervals of about 10 storeys (149-152) in a vertical axis. Each makes connections with other agents at the same level that impart horizontal and rotational forces on the tower up and down the axis (159, 164, 189, 191). The inevitable conflicts are resolved through iterative, non-binding actions that express the goals of the individual agents while maintaining the intensive and topological relationships within the tower assemblage. The TowerSkeleton cluster itself is minimally defined. Most of the action has been moved into subcomponents—the BaseAgent or LevelAgent classes—as have most of the metadata variables.\(^{43}\) While .DragAxis (155) preserves a proximity and orientation affinity between layers, .UpdateNdx (205) is restricted to maintaining a position in the defaultDict datastructure that reflects the location of the tower for easier recall while .Move (236) and .AdjAgents (§6.4.2) are simply coordinated prompts to component actions.

\(^{41}\) cf. Figure 0.1  
\(^{42}\) cf. §1.3b ‘The elements of urbanism’  
\(^{43}\) This recalls Bruno Latour’s digital monad cf. §5.1b ‘Means of interaction’  
\(^{44}\) Lee and Jacoby, “Typological Urbanism and the Idea of the City.”  
\(^{45}\) Trummer, “Associative Design: From Type to Population.” p182
Type and populations

It is possible to pursue this approach further and to further delineate the architectural object into a multiplicity of parts. This emphasizes the formation of architecture as a procedural event that also informs the urban configuration, and opposes a typological ideal or forces a reconsideration of typology that does not stand apart as a preexisting ideal. These changes are no longer defined by the identity or essence of an object, but rather would allow us to understand the object as possible forms of appearances through the interaction of the population of its parts. Deleuze describes such aggregates as an “area that can be called mixed, or rather, intermediary, between statistical collections and individual distributions … still more interindividual and interactive than it is collective.” Emphasizing that the identity of the assemblage does not dictate the configuration of the assemblage. Rather, “it is the real in matter, the thing, that has inner characters whose determination enters each time into a series of magnitudes converging toward a limit, the relation between these limits being that of a new type.” Here the idea of a type is replaced by a contingent definition. The assemblage does not conform to some law, but a type is simply the result as a statistical description—each time and for each case a new ‘type’ is formed. The replacement of definitions with limit conditions enables a shift from given forms to open-ended becoming.

In the same way that assemblage urbanism was given to emphasize the agency of the socio-material of the city, population thinking also emphasizes the agency of the associative material of the design model. In the associative model “each of the architectural primitives defines the metrical constraint components of an architectural object.” The challenge, then, is to preserve the individual agency of the components and not to fall back into a predefined type. Parametric modeling supports a classical interpretation of typology (the necessity of explicit formalization in some ways predisposes it to such an interpretation) just as readily as it constitutes a challenge to such thinking. In this project, we extend the configurational intelligence into smaller and smaller parts of the design, first with key floorplates that structure the overall shape of the tower and react to the presence of neighboring towers, and then again into individual units (Figure 6.4.1). Units are flexible within their own limits and limits relative to the floorplates, while floorplates are limited by the building’s surroundings and vertical cores. The tower converges to a final form through the feedback that occurs between all of these individuals equally, not bounded by scalar categories, resulting in a widening of the gap between the scale of architectural objects and the scale of its potential theater of effects.
This area of interindividual, interactive clustering is quite agitated, because it is an area of temporary appurtenances or of provisional possessions ... everywhere there are places available for mutations, explosions, abrupt associations and dissociations, or reconcatenations.

DeLanda, A New Philosophy of Society: Assemblage Theory and Social Complexity. p21

Because the limits cannot be known from a single case, the conditions that thoroughly describe an assemblage have to converge from a large sample that covers "the latitude of their variation and the relation of their limits"

DeLanda, A New Philosophy of Society: Assemblage Theory and Social Complexity. p21

"Principles as such will be put to a reflective use. A case being given, we shall invent its principle."

Deleuze, The Fold: Leibniz and the Baroque. p67

Deleuze connects definables with extensities, magnitudes (Figure 0.1), and similitude.

Trimmer, "Associative Design: From Type to Population." p182
§6.5 Architecture as Effectual Urban Input
Architecture as urban effector

For Schöffer, the particular form of the Ville Cybernetique, plays a very minor role, even less developed than in Yona Friedman’s Ville Spatiale, “rather, he focuses on the city’s complex system of functions and topologies, so that the whole remains at a very abstract level in terms of its design and planning.”53 Schöffer’s ambitions are really to extend the means of information feedback, while the architectural objects in the city surface are simply “a way to literally cover the surfaces of the city and fill its spaces with a structured grammatical system of patterns and ambient effects.”54 The conception of architecture as a mediating link between intent and spatial effect,55 as well as the role of architecture to facilitate further creative acts of urbanism,56 are both aspects of that we would like to advocate as features of reflexivity that draw attention to the active role architecture has shaping urban organization and not merely responding to the urban plan.

Collective form

However, this leaves out a significant and necessary element, which is the influence of architecture as an actual object and material presence in the city. The importance of the formal details of architecture is too often omitted from discussions of urbanism as though their significance was limited to the sphere of immediate experience or an indifferent symbolic function.57 In a lecture from 1965, Fumihiko Maki drew a distinction between mega-structures and group-form,58 two approaches to open-ended and temporal processes of collective forms. Mega-structures were those constructed of primarily independent systems that were bound together within a frame.59 While the expression of a physical, infrastructural frame would come to be a stylistically dominant feature of the mega-structure,60 Maki’s early definition accurately describes the Ville Cybernetique, whose tower array constitutes a conceptual and experiential framework that bundles together Schöffer’s five topological systems.61 Group-form, in contrast, “evolves from a system of generative elements in space,”62 where the units actively give shape to a mutable overall form.63 The formal details are significant, even operative, in the formation of collectives. “Forms in group-form have their own built-in link, whether expressed or latent, so that they may grow in a system. They define basic environmental space which also partakes of the quality of systemic linkage.”64 In this way, each element is reciprocally linked in a feedback relationship with the larger pattern65 as well as the local environmental conditions.66

Increasingly, the acceleration of large-scale urbanization in Asia points to the need to formulate a new relationship between architecture and urbanism.67 In the case of urban developments that are built immediately rather than incrementally, there is a unique opportunity for hybrid or unprecedented typologies that exceed the received forms of urban architecture.68 Nor is this only a negative case, an acquiescence to a narrow range wherein the architect’s influence can be exercised,69 but rather an exemplary case that points to a new general link between architecture and urbanism, one that can be exported from the large-scale development to other implementations.70 I would like to argue that, in fact, the potentiality of architecture as a reflexive urban interface, underscores urban architectural pursuits in general.71

§6.6 Reflexive Model Tendencies
Differentiation engines

With regard to computational modeling, reflexivity prioritizes pluralistic models that develop through iterative differentiation.72 This development could be called ‘agonistic’ to the extent that it stages agents with conflicting motivations but shared common ground in an environment that does not resolve the conflict but works through solutions that construct a consensus without removing the intensive differences that set agents against one another.73 With enough active variables in the model, the process begins to exhibits nonlinear complexity. Still, the actualized results are not completely divergent, but will begin to fall into recurring states. Any given outcome “can be activated in a variety of different ways, actualizing objects in a variety of different ways at the level of local manifestations.”74 These attractor states can only be
Schöffer’s designs for various buildings within the city are composed in a style of architecture parlante that appears to disregard the cybernetic network entirely. Darò argues that the metaphoric expressionism of these buildings is at least consistent with the scientific metaphor that Schöffer employs at the expense of “application and its material constitution.”

Darò, “Nicolas Schöffer and the Cybernetic City.” p9, 11


Ibid. p180

« Le rôle de l’artiste n’est plus de créer un œuvre, mais de créer la création. »

Schöffer, La Ville Cybernétique. p5

Jencks, The Iconic Building.

Maki, Investigations in Collective Form. p4

Ibid. p8, 12

Banham cites Ralph Wilcoxson’s four point definition of megastructures, two of which insist on a prominent structural frame. Banham’s definition is looser, but he dismisses attempts to retroactively deem the medieval urban matrix of Urbino equivalent to the frame.

Banham, Megastructure: Urban Futures of the Recent Past. p8, 76

In fact, Banham does mention Schöffer among the “Beginners and Begetters” of megastructuralism.

Ibid. p57

The five topologies are: time (rhythms), light, sound, climate, and space.

Schöffer, La Ville Cybernétique. pXX

Maki, Investigations in Collective Form. p14

of §5.8 ‘Enacting Encounters’ for a parallel with ‘field conditions’.

“a unit can be added without changing the basic structure of the village. The depth and frontage of the unit, or the size … may differ from unit to unit.”

Ibid. p18

Ibid. p19

Ibid. p19

“It may be easy for someone to invent a geometric form and call it group-form because such forms have characteristics of being multiplied in a sequential manner. This is, however, meaningless, unless the form derives from environmental needs”

Ibid. p21

Keeton, Rising in the East: Contemporary New Towns in Asia. p23ff, esp.31-33

“The split between architecture and urbanism that was collaged and amalgamated by the skyscraper was newly rebuilt within a larger superseding process, which uses the skyscraper as a means … If the skyscraper apparently obliterated urban traditions in order to develop itself as an anti-urban tradition, the overskyscraper uses its typology as an organizational primitive and as a thematic platform, from where to engineer new and empowered forms of discipline.”

Najle, “The Overskyscraper: Ubiquitous Tower Collectives.”

“we may conclude that architects can only intervene urbanistically in an increasingly remedial manner and that one effective instrument for this is the large building program that may be rendered as a megaform”

Frampton, Megaform as Urban Landscape. p39-40

Note that Frampton differentiates his use of ‘megaform’ from the megastucture through the relative continuity of the megaform, its topographic orientation, and its civic setting in the megalopolitan landscape, though he acknowledges that the two terms are not necessarily exclusive.

Ibid. p16

In more limited cases, the anticipatory potential “for linking newly established parts with parts not yet conceived” is particularly present.

Maki, Investigations in Collective Form. p35

of §9.2 ‘The Urban Generic’

An assembly process may be said to be characterized by intensive properties when it articulates heterogeneous elements as such … a process is intensive if it relates difference to difference … but also endows the process with the capacity of divergent evolution, that is, the capacity to further differentiate differences.”

DeLanda, Intensive Science and Virtual Philosophy p64

Mouffe, “Deliberative Democracy or Agonistic Pluralism.” p15

Bryant, The Democracy of Objects. P114

cf. §1.5b ‘Manuel DeLanda’ on redundant causality
described after comparing a number of simulations to one another through statistical measures, that is, these categories are populations that emerge from the model rather than being preexisting templates or types.\textsuperscript{75}

\textbf{Risk of Equilibrium}

Of course the notion of patterns of attractors implies that the model tends toward a state of equilibrium at which point the passage of time becomes immaterial.\textsuperscript{76} Counteracting the tendency toward stasis requires that “appropriately large differences in intensity need to be maintained by external constraints and not allowed to get canceled or be made too small.”\textsuperscript{77} The key point here is not simply the disruption of static states with additional energy or displacement but the preservation of intensities. After all, Schöffer himself considered the disruptions that issued from the central computer of the \textit{Ville Cybernetique} to tend not toward eccentricity but “towards efficiency, by modifying a situation in such a way as to maintain a supple balance between its components.”\textsuperscript{78} The next chapter will describe the movement following intension to the entropic function as a means of countering inflexible, teleological models.\textsuperscript{79}
Bibliography:


Frampton, Kenneth. *Megaform as Urban Landscape*. Ann Arbor: University of Michigan, 199AD.

Frampton, Kenneth. *Megaform as Urban Landscape*. Ann Arbor: University of Michigan, 199AD.


§7.1 Resisting Terminality

Nondeterministic models

Current masterplanning practices, display an “overreliance on narrow, singular, inflexible pictures of the future,”1 whether through two-dimensional plans or in rendered images. Even when designated for phased implementation, these plans project themselves years into the future, attempting to offer concrete targets for the future of the city.2 Unfortunately, because of the complexity of the city and the impossibility of producing a truly complete model, these targets will often never be achieved. This is true even of plans that employ dynamic or procedural modeling in their formulation.3 This chapter seeks to put forward the possibility of a method of urban design that is projective, but not teleological, one that remains open-ended and enacted in time through selective introduction of entropy into the model.4 This entropy could come in the form of updates to the model that allow unforeseen changes in the city to be incorporated (effectively resetting the given conditions), or as a programmed process of forgetting5 that would disrupt configurations that have reached static equilibrium. In addition to providing more responsibility to real contingencies,6 this would also shift the emphasis of urban design toward more strategic goals like encapsulating responses as unit operations7 and how actions might be assembled together in networks across the city.8

Generator

One project that explicitly embraced this idea was Cedric Price’s Generator. Unlike the other projects referenced in this text, the various studies for Generator never approached the urban or masterplanning scale, however it was designed with that potential in mind8 and its use and ultimate form was left to the whims of the occupants.9 Originally worked out through menus and physical games, the project was redeveloped by John and Julia Frazier as an intelligent computational system composed of programs to “manage the rules for Generator’s layout and the use of its parts,”10 to engage with the users, prompting changes, and “allowing users to prototype and visualize the outcomes”11 through physical models and computer plots. A final program, a boredom function12 was added to correct for inaction by the users or a history of monotonous uses. Thus while the program could “learn from the alterations made to its own organization and coach itself to make better suggestions,”13 it was also capable of challenging its users’ directions14 even when reconfiguration required unlearning all the patterns it had developed. Molly Wright Steenson argues that Price’s attraction to organizational tools such as computer programming was not motivated by harnessing methods of control, like it was for Schöffer, but “as a means to destructure the experience of the architectural project”15 in service of flexibility and indeterminacy. Following this, Price viewed the structural function of the cubes themselves as less important than the circulation network of paths and catwalks that constituted a ‘free-space ’ not intended to make a straightforward directive information flow; it represents possibilities and distribution, or different flows over time.”16

1 Verebes, Masterplanning the Adaptive City. p93
2 On predictions of the future: Part, “The Collective Image: Form, Figure, and the Future.”
3 For example ZHA’s Kartel-Pendik masterplan or GroundLab’s Deep Ground plan for Longgang.
4 Bullivant, Masterplanning Futures. p.22, 252-258
5 “the cycle of decay can become a linking force in our cities. If recognized, it provides an opportunity to replace old structures in an old environment with new structures still in an old environment.”
6 Maki, Investigations in Collective Form. p.94
7 cf. §3.4 ‘Unit Operations’
8 cf. §1.7 ‘Responsivity, Assemblage Urbanism, and Engagement with the City’
10 cf. §3.4 ‘Unit Operations’
11 “[A]s a forest facility for between one and one thousand visions,” though it was never explored in more than 150 room-sized cubes.
13 In fact, “one of the initial reasons why computers were required was to assist in handling the overlapping parameters involved in the Generator’s performance.”
14 Furtado, “Cedric Price’s Generator and the Frazers’ Systems Research.” p.58
16 Ibid. p.150
17 Frazier, An Evolutionary Architecture: Themes VII. p.41
18 “boredom would have to be the character of the shift in agency”
20 Ibid. p.123
21 Ibid. p.147
22 Ibid. p.146
§7.2 Informal Structures

Novel through network excess

Price envisioned these paths “as a puzzle or a maze” that could only be understood by walking through it and experiencing the connections or disconnections that were produced. Such experiences would then prompt the visitors to envision changes and to imagine the complex differently. This reflected his belief that “uncommitted or free-space must be seen not merely as the canvas for a new piece of architecture but as a continuing resource.” Philosophical accounts of how new forms come into being tend to favor a negative cause that derives from a void or lack in the current situation, however this position “is thoroughly informed by implicit structuralist assumptions.” Bryant suggests instead to locate the novel in the excess latent within the network, looking at the existing system not as a deterministic structure but as the dynamic formation of “ever shifting elements in networks or assemblages. In fact, it is already misleading to speak of networks or assemblages as this implies fixed and static beings. Rather, we should speak of assembling and networking, where elements brought together evoke action in one another, producing unforeseeable results and configurations.” Understanding and responding to such an environment calls for an immanent process of making sense of these processes, “a superior empiricism, a practice of cartography, capable of tracing networks and assemblages, or ever shifting relations among actors.” In order to engage in this sort of mapping, it is important to avoid totalizing and aloof images that would detach from “a more procedural sequence of intermittent coupling.”

Urban morphology

Historically, the dominant practices of urban morphology have been concerned with defining morphological regions through identification of historical forms of development (that is, with similar relationships between ground plan, built fabric, and land use) and identification of typological processes. Although addressed to the processes of city formation, the dynamism of urban morphogenesis is stunted by placing it within a catalog of fixed types. A trend toward techniques that could enable a more procedural cartography of the city can be discerned in some of the newer methods of urban morphology that merge the analytical power of computational processing with the metrical precision of GIS software and typically combine topological network analysis with metrical and spatial data. While still frequently employed for quite traditional mapping purposes, in the best cases these maps approach what Kwinter called “procedural maps … protocols or formulas for negotiating local situations and their fluctuating conditions.” An adoption of these methods by designers suggests the potential for many new metrics to be developed (since the design process opens direct access to morphological data that are not available to the researcher looking backward in time) as well as an activated concern for projecting alternative futures within particular, concrete urban assemblages.

The most well-defined and wide-spread of these new practices is the ‘space syntax’ approach, which developed from morphological studies of architecture into a series of “consistent techniques for the representation and analysis of spatial patterns,” particularly those that characterize “the cognitive dimension of architectural and urban space.” A unique aspect of space syntax is the reconstruction of the urban network “illustrating street segments as nodes and junctions as edges, known as the dual representation,” which better corresponds to the subjective, experiential position of space syntax, but which creates formal difficulties in correlating the results with other models. In our opinion, the attachment of space syntax’s analytic categories to vaguely defined cognitive categories also introduces problematic epistemological issues between analysis and interpretation. Alternatively, one can apply an analysis of the urban network that uses the primal graph (representing streets with segments and intersections with nodes), which implies a less subjective and more materialist focus.
Figure 7.1; Tangible interface for Generator. 1978-80

Figure 7.2; Detail of ground floor and roof plans with walkways.

Figure 7.3; Computer generated perspectives created by John Frazer, 1979

20 Bryant, “Towards a Critique of the Politics of the Void; Notes Towards a Politics of Assemblages.” p4
21 Ibid. p9
22 “it is the insertion of the dimension of time into the field that establishes a relation of continuity between subject and object, figure and ground, observer and event.”
Kwinter, Architectures of Time: Toward a Theory of Event in Modernist Culture. p98
23 Bryant, “Towards a Critique of the Politics of the Void; Notes Towards a Politics of Assemblages.” p14
24 de Certeau, The Practice of Everyday Life. p92-93
25 Kwinter, Architectures of Time: Toward a Theory of Event in Modernist Culture. p80
26 The main schools of urban morphology follow Conzen, primarily in the realm of urban geography; or Caniggia, who takes a more architectural approach. Integrated or comparative studies are increasingly popular.
Sima and Zhang, “Comparative Precedents on the Study of Urban Morphology.”
27 “For M.R.G. Conzen the climax of the exploration of the physical development of an urban area was the division of that area into morphological regions … an area that has a unity in respect of its form that distinguishes from surrounding areas.”
28 “The typological process is a succession of types in the same cultural area – diachronic changes – or in several cultural areas in the same space of time – synchronic changes … For Caniggia and Maffei, the type is a cultural entity rooted in, and specific to, the local process of cultural development.”
Oliveira, Monteiro, and Partanen, “A Comparative Study of Urban Form.”
29 Jiang, “Extending Space Syntax towards an Alternative Model of Space within GIS.”
30 Marcus, Westin, and Liebst, “Network Buzz: Conception and Geometry of Networks in Geography, Architecture, and Sociology.”
31 Kwinter, Architectures of Time: Toward a Theory of Event in Modernist Culture. p98
For example, Eric Fischer’s map of bus movement during the Occupy Oakland protests, compiled and uploaded the same day.
Fischer, “AC Transit Bus Service so Far Today.”
32 “it is a process of assembling possibilities out of actualities. Design connects us with vision, image and imagination; it produces hope and is productive of desire.”
Dovey, “Uprooting Critical Urbanism.” p350
33 “politics must be seen as a response to a particular problem inhabiting an assemblage and not as an eternal and unchanging set of questions”
Bryant, “Towards a Critique of the Politics of the Void; Notes Towards a Politics of Assemblies.” p12
34 Hillier and Hanson, “The Reasoning Art: Or, The Need for an Analytical Theory of Architecture.” p1
35 Marcus, Westin, and Liebst, “Network Buzz: Conception and Geometry of Networks in Geography, Architecture, and Sociology.” p68-6
Graph agents

One criticism of street network graphs is that one segment may actually have many different qualities over its length, limiting the resolution of an analysis that uses street segments and resulting in a granularity that varies unevenly depending on the network morphology. Sevtsuk addresses this concern by adding building entrances to the network as terminal points that branch off from the street segments, enabling the use of building data such as floor height, use, or occupant load to add weighted values to the urban analysis. In this project, we address this concern by adding additional segments at the location of the party walls that separate buildings or at locations where the street width changes significantly (Figure 7.5). This gives a much finer (and relatively even) resolution and renders the building volumes a more integral part of the network itself.

The additional benefit of using the primal graph is that it is compatible with the half-edge mesh datastructure that we introduced in §5.4.1. We will now add a dynamic component to this framework in the form of agents capable of movement through the network and capable of altering its topology. These agents are constructed in a similar manner as the mobile agents in §6.3.1 but instead of an xyz position and free trajectory vectors, they will be located at vertex points of the mesh (30, 33) and take their direction from the incident mesh edges (29, 32). Movement of the agents can be initially defined as a random walk—or, in this case a weighted randomization that factors in the width of the path (33, 40)—and then grow more motivated as a record of information is added to and collected from the environment.
7.2.1 GraphAgent

class GraphAgent:
    def __init__(self, ndx, m):
        self.ndx = ndx
        self.m = m
        self.vid = m.vid
        self.vids = set(m.vid)
        self.vids.add(m.vid)
        self.vids.remove(m.vid)
        self.vids = set(m.vid)

    def step(self, ndx, m, edgData, nde, boolCalc):
        zeroCt = []
        if nde:
            self.vids = self.vids - set([self.vid])
        else:
            self.vids = self.vids - set([self.vid])
        self.vids = set(m.vid)
        self.vids.add(m.vid)
        self.vids.remove(m.vid)
        self.vids = set(m.vid)

        if len(self.vids) > 0:
            traceRem = self.vids.RemoveAt[0]
            self.vids = self.vids - set([traceRem])
            self.vids = set(m.vid)
            self.vids.add(m.vid)
            self.vids.remove(m.vid)
            self.vids = set(m.vid)
        else:
            self.vids = self.vids - set([self.vid])
            self.vids = set(m.vid)
            self.vids.add(m.vid)
            self.vids.remove(m.vid)

        edgData[self.vid].tracMC = 1
        edgData[m.vid][self.vid].tracMC = 1
        self.vids.add(m.vid)
        self.vids.remove(m.vid)
        self.vids = set(m.vid)

        totalWidth = edgData[self.vid].w
        edgData[m.vid][self.vid].w = edgData[m.vid][self.vid].w + totalWidth

        edgData[self.vid].dnn = edgData[self.vid].tracMC /
        edgData[self.vid].dnn = edgData[self.vid].dnn

        edgData[m.vid][self.vid].dnn = edgData[m.vid][self.vid].dnn

        # CALCULATE THE LOCAL DEPTH MAP

        if len(zeroCt) == 0 and boolCalc:
            CalcDepthList(edgData)
        else:
            self.vids.append(m.vid)
The village-in-the-city phenomenon in Chinese cities, which has resulted from rigid restrictions on migrant mobility and lack of effective urban planning restrictions in what are legally considered rural lands, has been well documented. However, the majority of research focuses on those villages-in-the-city most embedded in the urban fabric. The villages of Haizhu island in Guangzhou are unique in that they are still surrounded by agricultural (and park) land, the Wanmu Orchard, despite occupying a geographically central position within Guangzhou. This fact isolates them somewhat from the pressures of urbanization and renders them more autonomous units than their counterparts.

This project is sited within Xiahzhoucun, one of the larger villages in Haizhu, located on the banks of a forked canal that connects the irrigation network of the orchard to the Zhujiang River. Unlike many villages-in-the-city, the population growth of Xiahzhoucun is not predominantly driven by migrant workers, but by students who attend school in the nearby University Town and artists associated with the Guangzhou Academy of Fine Arts drawn to the quietness and tranquility of the village. This project focuses on the network of public space (in a 15ha area at the center of the village) as dynamic material for enacting situated, loosely coordinated, incremental renovation and densification.

---

42 『城中村』‘chêngzhōngcūn,’ also sometimes called ‘urban village.’


46 Zacharias, Hu, and Huang, “Morphology and Spatial Dynamics of Urban Villages in Guangzhou’s CBD.”

47 “Yingzhou Ecological Park, located in Xiaozhou Village in southeast of Haizhu District, together with the neighbouring Wanmu Orchard, form the largest agro-ecological park in Guangzhou … Wanmu Orchard is a large green land rarely seen in large-scale cities in China.”

§7.3 Graph Functions
Network depth and properties

Before initializing the agents, it will be helpful to analyze the existing urban fabric in its current condition. Whereas the previous examples have used the half-edge mesh structure solely as a means of data organization and recall, here the properties of the mesh network are of direct interest. We will first look at properties of the street connections (as edges), then the intersections (as nodes), and finally the plots (as faces).

In constructing these analyses, there is a great deal of research to draw upon from the field of graph theory that seeks to explain properties of a network based on its morphological properties. From each node, the minimum distance required to reach the most distant node in the network is referred to as the eccentricity at that node. The largest eccentricity is called the diameter of the graph and the smallest, the radius. In 7.3.1 we set up a script that calculates eccentricity of a given node by recursively (21) tracing along the path network until the shortest distance to each path has been calculated, using a depth-first search. By iterating this analysis over the entire network, we can calculate the diameter and radius and identify central and peripheral vertices by comparing the distribution of eccentricity over the graph (Figure 7.9b). These metrics on their own may not tell us much about the structure, however they can give some suggestions about the urban network when plotted alongside the graph geometry (Figure 7.9a). For example, the path of the radius is a winding route that occasionally turns back on itself, highlighting detours or blocked connections, while the diameter alternates between stair-stepping and continuous spines (these are usually following geographic boundaries such as the river edges), illustrating that the meshwork has aspects of both a grid and a ladder structure. Similarly, the central and peripheral nodes are not arrayed radially but along the diagonal that stretches from the southeast to the northwest. Interestingly, two of the main entrances (Figure 7.10) are among the most peripheral points.

Beyond these graph-specific properties, we must also remember that the urban network is insistently place-based, adding an additional spatial dimension of fixed distances and locations. Thus where graph analysis may be interested in the simple depth, urban network analysis can extend that analysis by considering not just the number of links, but also the physical length of the segments and a comparison of the embedded graph space to continuous global space (Figure 7.12).
isocurves have been drawn over the map afterwards in the style of Friedman’s Flatwriter to emphasize the irregularity of the urban network.

Calculating these values over the entire network is slow and not efficient to try during the dynamic operation.

Figure 7.9
Plotting the radius (397.46m) and diameter (767.96m) of the graph in the heavy yellow lines. Additionally, each vertex is labeled with a hexagon whose size and color is defined by the value of the vertex’s eccentricity. The 10% most peripheral and 10% most central nodes are highlighted with more opacity.

The zoom-in displays the eccentricites without the other data overlaid.

Figure 7.10
A level structure map of the distance to each segment of the circulation network from the four primary entrances to the village in network lengths. Edges are colored based on their distance to the nearest of the four entrances. Isocurves have been drawn over the map afterwards in the style of Friedman’s Flatwriter to emphasize the irregularity of the urban network.
7.3.1 DepthMap

```python
def TracePaths(vHdx, vOrig, mEd, dist, dList):  # Main Code. Initialize output lists
    distOut = []
    ntwkSum = {}
    rtoLat = {}
    for i, pt in enumerate(vHdx):
        distOut[i].extend([(m*1.2**j) for j in range(1, len(vHdx[i]) + 1)])
        if j%2 == 0:
            if jk2 == 0:
                valCk = {}
                for list in distOut:
                    valCk.append(list[i][0])
        newVal = valCk.min()  # 0 ≤ lim distOut[newVal][j][i] < lim
        distOut.append(newVal, j, i)  # 0 ≤ lim
        dList.append((newVal, j, i))
        edgOut.append((newV, j, i))
        dList.append((newVal, j, i))
        edgOut.append((newV, j, i))
        gD = dpthList
        dD = distLat
        pR = rtoLat
        nt = ntwkSum
        edHdx = distLat.index(eco)
        edOut
```

The script measures the distance from given vertex points to every point in the graph (or up to a given limit).

07 The recursion of this subroutine (21) follows the next edge (0, 0) in the HEM datastructure until it encounters an already measured lower value or a dead end. The while loop ensures that the subroutine picks up where it left off with other branches at the given intersection (24).

08 Check that the edge has a width greater than zero (i.e. it is currently a path).

11 Check that the edge has a width greater than zero (i.e. it is currently a path)

15 Write a tuple of the geodesic and Euclidian distances to the output list.

17 Exit while loop if a dead end is reached.

36 For each edge, a default value greater than the limit is added to the distance list. These values are replaced whenever a lower measured value is found.

38 Call the recursive function TracePaths.

40 The remainder of the code processes the distance values assigned to each segment into individual metric values, such as the proportional ratio between the geodesic and Euclidian distances (49), the total length of the network reachable within the limit (50), and the number of segments reached (51).

44 Only every other value (1%2) is used because segments are not directed and thus half-edges and their twins will always have the same values. Also in (57).

49 When all the fitness adjustments have been added up they are added to the FaceAgent's fitness values.

62 At each segment, find the lowest value of all the distances measured from the various starting points.
7.3.2 TracebackShortestPath

```
import Rhino.Geometry as rg; import math

def EdgesOut(m, wCurr, eOrig, d, lst):
    eHypt = m.eNE[eCurr]
    if eHypt == m.eTE[eOrig]:
        return
    else:
        if d[1][math.floor(d[1]/2)] > 0:
            lst.append( (d[1][math.floor(d[1]/2)], eHypt) )
        else:
            lst.append( (10000, eHypt) )
    EdgesOut(m, m.eTE[eHypt], eOrig, d, lst)
    return(lst)

def TrackBack(m, sCurr, d, lst):
    lstOut = []
    EdgesOut(m, sCurr, eCurr, d, lstOut)
    if len(lstOut) > 0:
        eHypt = min(lstOut, key=lstOut)
        if m.wOV[1][eHypt] == atOdd or m.wOV[eHypt][eHypt] == atOdd:
            lst.append(m.e[1][eHypt])
        return
    else:
        TrackBack(m, eHypt, d, lst)

# MAIN CODE. INITIALISE LISTS
edgLat = []
estEven = []
estOdd = []
EdgesOut(bem, (eOdd*2), (edgOdd*2), gD, lstEven)
EdgesOut(bem, (edgOdd*2)+1, (edgOdd*2)+1, gD, lstOdd)

# COMPARE VALUE IN EITHER DIRECTION, CHOOSE WHICH HALF-EDGE TO FOLLOW
if lstEven and lstOdd:
    if min(lstEven) < min(lstOdd):
        edgLat.append(bem, eOdd*2, gD, edgLat)
        TrackBack(bem, edgOdd*2, gD, edgLat)
    else:
        edgLat.append(bem, eOdd*2+1)
        TrackBack(bem, (edgOdd*2)+1, gD, edgLat)
elif lstEven:
    edgLat.append(bem, eOdd*2)
    TrackBack(bem, edgOdd*2, gD, edgLat)
else:
    edgLat.append(bem, eOdd*2+1)
    TrackBack(bem, (edgOdd*2)+1, gD, edgLat)
```

OUTPUTS: e

This script uses the geodesic distance values computed in §7.3.1 and traces back from any given ending segment to the starting point following the lowest values. This will only work with a single starting point.

3. The EdgesOut subroutine runs recursively to loop through all the edges radiating outward from an intersection given an incoming half-edge.

9. Return the list of indices and the geodesic distances associated with them in a tuple so that they lowest value can be determined.

16. The TrackBack subroutine calls EdgesOut, saving the edge with the lowest value and calling itself recursively (§26) if it has not yet reached the start point (§21).

33. To begin the code, run the EdgesOut subroutine on either side of the given edge and proceed in the direction with the lower value (§38, §41) or in the direction that is continuous if the other does not continue (§44, §47).

Figure 7.11
An analysis of the network depth from node #1900, and the traceback of longest path across the network using the scripts on these pages. This contributes one datapoint (of 2035) to Figure 7.9.
Nodal properties

Similarly, the physical nature of urban networks affect which properties of the network nodes are interesting. In certain cases, mapping the angle of intersections can reveal intersections or blocks of regularity, as can the plotting the degree of the nodes. Unlike social networks, urban networks are rarely scale-free networks, but have a relatively normal distribution of low-degree intersections (Figure 7.13). While certain central hubs may seem to dominate urban space, this is typically a result of use—for example, carrying disproportionate traffic—rather than the number of physical connections. Urban networks are not as discrete as social networks, and actually emphasize the continuity from one link to another. For this reason, among others, the project will use dynamic, mobile agents as the basis of an analysis that accumulates over time.

Urban networks as design foundation

Much of the attractive urban quality of the village of Xiaozhoucun derives from the experience of walking through the warren-like alleys of the village and the small pockets of open space, especially along the river. In this project, the existing urban fabric will be preserved to the greatest extent possible while rehabilitating the building stock through selective rebuilding. As mentioned earlier (Figures 1.1-1.4), there is a rich, dynamic pulse to the village that is driven by a material cycle of demolition and rebuilding. As opposed to the current methods of clearing and rebuilding tabula rasa or updating superficial stylistic features this project proposes a framework for planning the redevelopment of informal urbanism temporally and punctually, from within. The plan will not be defined, nor will it have a targeted end product, but will remain open to continually changing in response to collected information. For this project, that information will be generated by a swarm of dynamic, mobile agents as the basis of an analysis that accumulates over time.

Figure 7.12
This map compares the connectivity of the circulation network. The radius of the hexagon at each node is determined by the size of the network that can be reached within a 150m walk. The color reflects a ratio between the distance required to walk to each of those points and the Euclidean distance measured directly.

The zoom-in shows one such local network with a maximum depth of 150m and the effect that discontinuities (like the canal) have on accessibility.

Figure 7.13
The node degree represented with larger and brighter spots for higher degrees. The distribution is roughly normal.
mobile agents and we will search for opportunities to increase the overall connectivity of the circulation network, to open up additional small pockets of public space, and to increase the built density when existing buildings fall into obsolescence.

§7.4 Nonlinear Dynamics

Agents and embedded mapping

In making the transition from deterministic to open-ended masterplanning, we will adapt our static, global analyses into dynamic, embedded procedural maps. Rather than trying to characterize features of the urban network, such as clusters, bottlenecks, or betweenness based on morphological patterns, we are using a small swarm of mobile agents. Random walks on a network, and particularly random walks by multi-agent systems have been shown to be an effective way of determining clustering patterns in networks.

The first metric we will use is simply the record of the agents’ movement history, recorded as an incremental value stored in each edge that increases every time an agent passes over that edge. As the agents are instantiated at public spaces throughout the village, those areas that are conveniently connected to these public spaces will record high traffic and modifications can be made that reinforce their status as places of public gathering. Other segments that will record high traffic values are dead-ends and cul-de-sacs (i.e. small clusters that are not well integrated in the network) where agents will get trapped for multiple frames. These are equally important to address because they are points where the typical connectivity is disrupted and small changes to the network can have more intense impact. During their random walk, the graph agents are also executing local analyses of the graph proximities. These functions operate like the global analysis shown in Figure 7.11, except that the scope of exploration is capped at a lower limit, 75 meters. In this case, however, the agent’s are not interested in calculating specific properties of the local network, but of finding opportunities for modifying it. Thus, they process the depth map twice: first as usual, restricted to the circulation network; and then again including the edges that denote party walls or untraversable edges. By comparing the two sets, the agents can determine more interesting data than simply the eccentricity of their surroundings. They identify instead, the location which exhibits the greatest expansion of distance in the current network state compared to a potentially open network. The more efficient path is traced, and the closed segments that it crosses are recorded with an incremental value representing shortcut potential. The traffic and shortcut values are merged into a single number that can be displayed on the network while the model is running.

Duration and persistence

Meanwhile, data about the existing buildings have been imported through a bitmap image, and each face has been assigned a value that defines its persistence, a value that incrementally decreases each frame. Presently, the value is created as a factor of of the building height, but this value would also allow differentiation of buildings by ownership, physical condition, age, construction type, or some top-down redevelopment plan. The face’s persistence value is compared to the pressures exerted on it by the demand from its perimeter edges and enters contention for replacement when the pressure exceeds persistence. When a building is selected for redevelopment, the edges that triggered the event are widened, any adjacent edges that had high shortcut potential are considered a part of the circulation network proper, and the plot is rebuilt with the appropriate setbacks and additional height as appropriate. If a setback is deep enough, that spot will be added to the list of public spaces that can instantiate a new agent. Existing agents are cleared away, and the cycle repeats, now operating on a different ground condition.
Metabolism and Catastrophe

Fumihiko Maki wrote that urban metabolism “gives morphological demonstration of the ever-changing and diverse character of city life.” specifically referencing a cycle of decay that allowed old structures to be replaced with new ones within an old environment. For assemblage urbanists, metabolism is a lens that brings into focus sociomaterial interactions: revealing “process geographies and wherever they lead.” In this chapter we would like to bring these two perspectives, retaining both the significance of the evolving form of the city as well as maintaining a social role for material to play in the development process. In this sense, the recycling of brick that occurs in Haizhu villages (Figures 1.1-1.5) is not simply a curiosity, but a parameter of the village form. Properly approached, even the material properties of standardised construction can be incorporated into the project metabolically—that is, not as a criteria that requires a certain plan, but as a filter that helps regulate or distribute particular processes. For example, the dimensions required for an elevator can be used in conjunction with the process outlined above to determine the height of new construction: plots that are sizable enough, or can pair with an adjacent plot, are able to build higher than those that must be reliant on staircases alone. This filter would encourage assemblages or group form, without necessitating an erasure of the older context.

Typically, the dynamics of social space are not characterized by a smoothly continuous change, but by sudden changes and mutations in character. In dynamics, such bifurcations in the trajectory of a system are termed catastrophe events. Catastrophes plunge the system out of equilibrium disproportionately to the change of parameters that triggered them. Generator’s boredom routine instigated a catastrophe, all the learned patterns of occupation were thrown out, with the hope of catalysing a reaction that would drive it into a new set of order. The opening up of shortcuts into a dense urban fabric may also spark a catastrophe that diverts users, uses, and materials into new actions and new forms of organization. The increased entropy of a dynamical system out of equilibrium disrupts settled identities and leaves it susceptible to influence by incidental perturbations. While even the more tightly controlled masterplans are always susceptible to catastrophes from outside, thus far they have not risen to the challenge of working with and through catastrophe.
7.4.1 CalcDepthList

def CalcDepthList(edgData):
    ini:
    self.depthList.clear()
    self.CircDepthNetwork(self.vtx, edgData, ini, 0)
    self.FullDepthNetwork(self.vtx, edgData, ini, 0)
    self.netwLength=0
    self.depthTo = (0, 0)
    for key, val in self.depthTo.items():
        if k2 == 0 and val[1] < ini and val[1] > 0:
            self.netwLength = edgData[key].length
            self.depthTo = ((self.depthTo[0] + (val[1] / val[2]), ...)
            self.depthTo[1])

    """Find biggest discrepancy"
    nullVal = math.ceil((lim - vals[1]) / lim)
    deltLat = [(vals[1] - vals[0]) * nullVal, key]...
    for key, vals in self.depth.items():
        deltLat = sort()
        eNode = int(2 * math.floor(deltLat[1] / 2))

    """Initialize trackback"
    evenLat = []
    oddLat = []
    self.VetEdges(m, endSeg, endSeg, evenLat)
    self.VetEdges(m, endSeg, endSeg, oddLat)

    if evenLat and constLat:
        if min(evenLat) < min(oddLat):
            pthList.append(m.eNode)
        else:
            pthList.append(m.eNode)
    else:
        pthList.append(m.eNode)

    """Traverse back"
    self.TraverseBack(m, endSeg, pthList, edgData)

7.4.2 VertexEdges

def VetEdges(self, m, eOrig, lst):
    if m.eOrig:
        return
    else:
        if (math.floor(eNode)) in self.vDepth:
            lst.append((self.vDepthList[int(eNode)], eNode, ))
        self.vDepthList[m, eOrig, eOrig, lst]

return(lst)
§7.5 Contingent Identity

Disassembling

Thus entropy is invoked here not as a gradual evening out of energy or a dissolution of order, but in the sense of a process that emphasizes the actions of making and unmaking rather than ultimate state of the finished work,\(^7\) in the same way that it was adopted within 'process art.'\(^7\) Entropy is also interesting because it "contradicts the usual notion of a mechanistic world view" through its irreversibility. In an entropic system, serial repetition does not produce identical replication, but is each time acting in a context that has been incrementally differentiated.\(^7\) Because entropy tends toward disorder, this should not be interpreted as a teleological drive, but as potentially leading to a catastrophe event.\(^7\)

The reason that we can avoid an eventual static equilibrium is that we do not define the city or the plan as a closed system\(^8\) but as one with a constant injection of energy from without. In the model, the agency of the diagram is added to the model at each cycle to both disassemble existing order while also creating new, imminant organizations.\(^8\) The diagram acts to entropically to remove structure, but also negentropically by inserting new forms: "it makes history by unmaking its past realities and significations, constituting so many cutting edges of emergence or of creationism, of unexpected conjunctions, of improbable continua."\(^8\) Furthermore, this model permits the introduction of other external events—the uncoordinated and naturally occurring development of the village—without them disrupting or derailing the usefulness of the model.

Pluripotential

Rather, this variability is assumed as a natural behavior for the model, which anticipates the complete mutability of the individual entities. Alliez writes that the diagram is a "surface of experimentation … flush with the real that writes a new type of reality, carried into the very fabric of the most concrete of assemblages by the joint deterritorialization of expression and content."\(^8\) Stripping away the intensities of the assemblage, this forces the object back into confrontation with its informal substance and the possibility of new forms (Figure 0.1). For this reason, the informality of the village-in-the-city is a productive testing ground where the identities assigned by building and land-use codes are either absent or only apply ironically,\(^8\) allowing material traits to draw out new alternate potentials.\(^8\) The next section will expand on how this surface of experimentation works through material enaction to supercede simple description.
Arheim points out that entropy cannot be measured in an instantaneous state. "The particular nature of any one such state does not matter. Its structural uniqueness, orderliness or disorderliness does not count, and its entropy cannot be measured. What does matter is the totality of these innumerable complexities, adding up to a global macrostate... only by adding up a sufficient number of momentary complexities over a sufficient length of time can we tell something about the macroscopic state."

Arheim, Entropy and Art: An Essay on Disorder and Order. p17

Lee, Object to Be Destroyed: The Work of Gordon Matta-Clark p39

Smithson, "Entropy Made Viable: Interview with Alton Sky (1973)." p301

Deleuze, Difference and Repetition. p2

As in Smithson’s ‘Partially Buried Woodshed’ (1970).

Lee, Object to Be Destroyed: The Work of Gordon Matta-Clark p46

As it was in Chapter 5

"the diagram and abstract machine have lines of flight that are primary, which are not phenomena of resistance or counterattack in an assemblage, but cutting edges of creation and deterritorialization."

Deleuze and Guattari, A Thousand Plateaus: Capitalism and Schizophrenia. p585 as quoted (and translated) in Alliez, Diagram 3000 (Words). p11
Bibliography:


§8.1 Urban Inaesthetics

The city thinks itself

Badiou introduces inaesthetics as an alternative to historical modes of thinking where the relationship of philosophy to art is such that philosophy does not dictate an aesthetic agenda, but reflects on the revelations of art and "maintaining that art is itself a producer of truths, makes no claim to turn art into an object for philosophy. Against aesthetic speculation, inaesthetics describes the strictly intraphilosophical effects produced by the independent existence of some works of art." Key to this relationship is a position that art is itself a unique form of thought and that it is irreducible to other forms of thinking (including philosophy). He elaborates that individual works of art constitute a "local instance" or "differential point" of an artistic configuration "which is a generic multiple, possesses neither a proper name nor a proper contour, not even a possible totalization in terms of a single predicate. It cannot be exhausted, only imperfectly described." In this light, inaesthetics offer a productive analogue to the relationship between the city and architecture. The city, as we know can never be contained by the theory that reflects on it; it proceeds through a momentum and metabolism proper to itself. At the same time, works of architecture swarm around, assembling the city, not by any single plan but participating in the prefiguring of the city through "inventive inquiry into the configuration." Even large masterplans can only be implemented as discrete elements of the city, the whole can be approached "only by the chance of their successive occurrences."^

§8.2 The Open City

The heterogeneity of the urban

At the same time, we must be careful not to accept this formulation uncritically. One particular problematic point of divergence is that the city cannot be defined so narrowly as Badiou defines artistic configurations—selectively assembled out of material that is all of a kind—but must acknowledge (at least temporarily) all of the objects that inhabit in, interface with, or pass through itself. The city is continuously changing its alignment, being made and unmade in ways that exceed the variation that Badiou's configurations undergo.^

---

1 Didactic, romantic, and classical are the three historical aesthetic schemata to which Badiou contrasts inaesthetics. In the text they are represented by the philosophies of Plato, Heidegger, and Aristotle, respectively. Badiou, *Handbook of Inaesthetics*. p5, 7
2 Ibid. p0
3 “‘Immanence’ refers to the following question: Is truth really internal to the artistic effect of works of art? Or is the artwork instead nothing but the instrument of an external truth? ‘Singularity’ points us to another question: Does the truth testified by art belong to it absolutely? Or can this truth circulate among other registers of work-producing thought We will therefore affirm this simultaneity… Art is rigorously coextensive with the truths that it generates… These truths are given nowhere else than in art.” Ibid. p9
4 Ibid. p12
5 Ibid. p12
6 “we must above all not conclude that it is philosophy’s task to think art. Instead, a configuration thinks itself in the works that compose it.” Ibid. p14
7 Ibid. p14
8 Ibid. p12
9 Ibid. p12
10 cf. §2.4b ‘Exo-relations form new assemblages’
11 “What Leibniz calls metamorphosis or metaschematism not only involves the initial property of bodies—in other words, their capacity to envelop infinitely and, up to a certain point, develop their specific parts—but also the second property, the fluxion that causes parts endlessly to leave their specified aggregate in order to enter into entirely different aggregates that are differently specified.” Deleuze, *The Fold: Leibniz and the Baroque*. p115
12 cf. §1.2 ‘The dynamic city’
13 Strictly speaking, these configurations are “intrinsically infinite” truths, and thus not truly variable. However, as their description is always imperfect (cf. note 5, above), the description and understanding of what these configurations ‘will have here’ can and does change as new works emerge. This is covered under the concept of ‘foicing,’ cf. §9.1b ‘Relating the generic to formalization’.
The multiplicity of architecture

Nor can the parallel between architecture and art be made so simply. For Badiou, the work of art is essentially finite, it is self-defined by its own finitude.\(^{14}\) We have already argued, however, that architecture “is incapable of being restricted to a single domain”\(^ {15}\) but is always an object acting in a multiplicity of associations and that it should be understood as an interface\(^ {16}\) that extends out beyond its limits. Michael Guggenheim does an excellent job of showing how a building is often simultaneously a work of art, a dwelling, a technical system, or any other number of uses;\(^ {17}\) “the manifold interfaces simply allow too many starting points for different uses by different people at the same time.”\(^ {18}\) This is not simply a property of buildings as such, but is also intrinsically linked to how buildings are necessarily situated in their environment, and their disposition within the urban assemblage.\(^ {19}\) The meaning and definition of architecture changes from moment to moment depending on the networks that engage it or the material assemblages to which it is enlisted. Being finite neither in time nor in space, evading principles of autonomous completion, architecture cannot be made equivalent to art in this comparison.\(^ {20}\) We will say then that architecture does not define the urban nor does it exist as finite inquiries into the urban condition, but instead that “it is the sign of the possibility” of an urbanism.\(^ {21}\)

§8.3 Material Enaction

**Enaction supersedes the script**

What does inaesthetics have to say about architecture, then? In fact, Badiou writes at length about the possibility of aesthetic media dedicated not to the singularity of art, but that remain open to the infinite; a condition he uses to illustrate the native act of thinking that occurs within media.\(^ {22}\) The deferral, or refusal, of the limits of finitude points “toward thought as event, but before this thought has received a name”\(^ {23}\) The event is in the act of being worked out, not predetermined, though neither is it entirely indeterminate.\(^ {24}\) Rather, architecture acts as a ground that organizes an undecided event, setting up the next steps, without impinging in the least on what they may be.\(^ {25}\)

We have touched on this topic already, but Badiou elaborates on its mechanics. Firstly, there is a complication of the internal potential state that prevents a simple, direct realization by requiring a continual adjustment to the situation.\(^ {26}\) This occurs when the contingencies of the city interact with the diagram of architecture. Architecture in its enaction supersedes its own diagram, replacing a scripted definition with something more improvisational.\(^ {27}\) “What one sees is at no point the realization of a preexisting knowledge, even though knowledge is, through and through, its matter or support.”\(^ {28}\) Badiou characterizes this as an act of forgetting, which we can also compare to the caesura of self-othering that takes place in information exchange or recall.\(^ {29}\) When Badiou writes that this pre-existing quality yields to emergence\(^ {30}\) he is reiterating that to remain as a mode of thinking means to maintain a state of unfixed uncertainty despite actively pressing forward.\(^ {31}\) A work of art is a presentation, or even a representation, of “the persuasive procedure of its own finitude,”\(^ {32}\) but architecture manifests as “a false totality. It does not possess the closed duration of a spectacle, but is instead the permanent showing of an event in its flight, caught in the undecided.”\(^ {33}\) All of this adds up to the banishment of a panoptic view of urbanism, of plans that could stand outside of time, “what there are instead are disparate truths, an aleatory multiple of events of thought”\(^ {34}\) that must be played out as a temporal process, embedded in a particular urban space.\(^ {35}\)
“art is the creation of an intrinsically finite multiple, a multiple that exposes its own organization in and by the finite framing of its presentation and that turns this border into the stakes of its existence.”

Ibid. p11

“Whereas an artwork is the product of the system of art, and without the functional system of art the artwork would often not even exist, this cannot be said of buildings. They are first of all buildings and can be made into objects of art or science, but never exclusively. Even those buildings that are monofunctional building types … are not controlled by these functional systems.”

Guggenheim, "Building Memory: Architecture, Networks and Users." p46 (also p48)

Guggenheim uses the description ‘quasi-technology’: “objects that are sometimes real technologies, functioning as black boxes, but at other times they lose this quality. They are turned from technologies, in the sense of blackboxed procedures, into mere ‘masses’ of materials. They become materialized as I would like to call it. To materialize in this sense means that an object is freed from its actor-network and reduced to its material qualities.” Thereby enabling undirected or opportunistic uses.

Guggenheim, “Mutable Immobiles: Change of Use of Buildings as a Problem of Quasi-Technologies.” p7

“A work of art is essentially finite. It is trebly finite. First of all, it exposes itself as finite objectivity in space and/or in time. Second, it is always regulated by a Greek principle of completion: It moves within the fulfillment of its own limit. It signals its display of all the perfection of which it is capable. Finally, and most importantly, it sets itself up as an inquiry into the question of its own finality. It is the persuasive procedure of its own finitude.”

Badiou, Handbook of Inaesthetics. p10-11

The full quotation reads, “Dance is not an art, because it is the sign of the possibility of art as inscribed in the body.”

Ibid. p69

In the text, Badiou singles out dance as an exemplary practice. For more on the connection between architecture and dance see my forthcoming essay.

Badiou, Handbook of Inaesthetics. p61

“the spatialization of imminence would thus be the metaphor for what every thinking grounds and organizes.”

Ibid. p62

cf. §4.3a ‘Prepositional mode of being,’ especially note 26

In fact, this addresses one of Badiou’s disputes with Deleuze, eti-a-vivi chance and the “eternal Return.” We will avoid this debate, but point instead to the explication of “virtual proper being” in §2.3c ‘Organisation and endo-relations’.

Badiou, Deleuze: The Clamor of Being. p75

Patt, “Performance Review: In Praise of the Possibility of Architecture.”

Badiou, Handbook of Inaesthetics. p66
§8.4 Lines of Flight

Content and action

Thus the concept of forgetting as de-differentiation plays an important role in preserving the multiplicity inherent in urban design and in re-orienting urban design from the production of a plan to an environment for continual production and reconfiguration.\(^\text{36}\) an environment that would formulate the contested status of urbanism’s own inherent complexity as a condition to be extended rather than resolved.\(^\text{37}\) Just as the material enactment of the city overrides the motive force behind it, the production of urban designs should also avoid the direct execution of their underlying support data, instead striving to make productive use the slippages and discontinuity of information by leveraging urbanism’s discrete constellations,\(^\text{38}\) instrumentalizing emergence and erasure to evolve relevance of definition and diagrams of design. In contrast to those who argue that the assemblage of urban systems from architecture “can be most adequately grasped if it is analyzed as an … autopoietic system of communications,”\(^\text{39}\) we would argue that to the extent that architecture is concerned with communication, it does so “not as carriers of something Other,” semiotic content, for example, “but as forces-signs of deterritorialization and of reterritorialization.”\(^\text{40}\) The parametricist position seeks to maintain the singularity of architecture—“as a sui generis system,”\(^\text{41}\)—but gives up its immanence by locating the content entirely in external meanings and abstractions.\(^\text{42}\) Ultimately, this also removes architecture from the generation or transformation of ideas, offering instead “nothing but the consequence of playing out an act of naming.”\(^\text{43}\) Inaesthetics demonstrates an alternative axis between content and action that invests the reality of material assemblages with an agency to extend and support the potentiality of new organizations that exceed strict, technocratic definitions and reimagine the city otherwise than it is.\(^\text{44}\)
Bibliography:


§9.1 The Generic Multiple

Accounting for novelty

The capacity for transformation is surely one of the most essential points of a theory of contemporary urbanism. For a thorough, formal model this requirement is made more difficult by the fact that a true change cannot be given in advance by the initial state of the urban situation. Modeling an urban situation (whether computationally or mentally) requires constructing some sort of formal definition or representation of the specific conditions; whereas true change must exceed this formalization not by negating it but by instantiating a generic alternative. Such an alternative is called a ‘generic’ multiplicity because, in contrast to the specific form of the known model, this part cannot initially be described in full by the model. It is “something lying beneath, or something at work of the known model, this part cannot initially be described in full by the model. The generic leverages its own indiscernibility to bridge the gap between presentation and the vast number of possible forms of representation.” Badiou goes through a lengthy explication of how the generic multiplicity is defined in set theory in order to prove the rigorous possibility of the logical operations that have to occur and the correspondence of the mathematical proofs to his own ontology.

Relating the generic to formalization

It is unnecessary to recount these details here, instead we want to highlight the usefulness of the generic as a conceptual tool. In the same way that the construction of assemblages benefited from an external definition, we do not need to completely know or understand the generic multiplicity. Knowing for certain only that it cannot be entirely defined by the previous situation, we can include it in the current situation extensionally and investigate the consequences under a projective hypothesis of what the generic might entail. This leads to “a praxis consisting of a series of enquiries into the situation … to work out how to transform the situation in line with what is revealed by the event’s belonging to the situation.” Such inquiries can be used to reveal the contour of the generic subset. The process of inquiry, which Badiou (following Cohen) calls ‘forcing’, is an anticipatory modeling that incrementally reveals aspects of the generic and ultimately “renders the indiscernible immanent.”

1 “The fact that the procedure is generic entails the noncoincidence of this part with anything classified by an encyclopaedic determinant. Consequently, this part is unnameable by the resources of the language of the situation alone. It is subtracted from any knowledge

2 cf §8.3a ‘Material enaction’ especially note 31

3 Badiou, Being and Event, p338

4 “The orientation of constructivist thought … is the one which naturally prevails in established situations because, it measures being to language such as it is.”

5 Ibid, p328

6 Feltham, Alain Badiou: Live Theory, p108

7 In this passage the generic is contrasted with the constructivist position: “The constructivist orientation of thought proceeds by restricting the multiples admitted at the level of representation to those multiples that correspond to a strictly defined formula, and while this might hold true in the early set theory of Frege, it does not characterize Leibniz’s ontology (as we have employed it in chapter 2), which is equally permissive about the production of new multiples.”

8 Feltham, Alain Badiou: Live Theory, p95-96

9 “The Baroque solution is the following: we shall multiply principles … and in this way we will change their use. We will not have to ask what available object corresponds to a given luminous principle, but what hidden principle responds to whatever object is given, that is to say, to this or that ‘perplexing case.’ Principles as such will be put to a reflective use. A case being given, we shall invent its principle.”

10 Deleuze, The Fold: Leibniz and the Baroque, p67

11 “to assert the existence of a generic multiple is to assert that a procedure can hold together in the absence of objective and known guarantees.”

12 Ibid, p41

13 §8.1 ‘Urban Inaesthetics’

14 “In set theory, one can have ‘models’ of set theory which are interpretations that flesh out the bare bones of sets and elements by giving values to the variables … The model itself, as a structured multiplicity, can be treated itself as a set. Cohen takes as his starting point what he terms a ‘ground model’ of set theory. Badiou takes this model as the schema of a historical situation. Each subset of this model satisfies a property which can be expressed in the language used in the model. That is, every multiple found in the model can be discerned using the tools of language. A generic set, on the other hand, is a subset that is ‘new’ insofar as it cannot be discerned by that language. For every property that one formulates, even the most general … the generic set has at least one element which does not share that property.”

§9.2 The Urban Generic

Locating the generic

The generic dimension of urbanism will therefore be located within architecture,16 which provides the kind of infinite multiplicity required17 and is also capable, when introduced into an existing urban situation, of provoking a reconfiguration of the urban order.18 While other subsets of urbanism—programmatic organizations, functional systems, infrastructures, policy and regulatory dictates—have the capacity to effect this realignment, none belong to or participate in the urban as insistently as architecture.19 The generic subset “is a multiple that intersects—contains some elements of—every domination. The generic multiple thus contains at least one element corresponding to every property whatsoever; it contains a little bit of everything.”20 In this sense, although architecture is uniquely suited to the role of urbanism’s generic multiplicity, we cannot indiscriminately confer the category of generic on any and all architecture. Only architecture that operates as an interface to the city,21 cutting across every category22 can occupy this role. What we have argued here is that it must be an active participant in the processes of the urban ecology (as an integrator of different systems and flows; as a reconfigurable environment; as a model that adapts to multiple locations through out the city; as a material support for diverse urban practices; etc…) because the generic is never simply given, but always assembled or “gathered together.”23

Procedural engagement also allows architecture to continuously prompt new inquiries into the changing potential of the urban situation. Because “a concrete situation is an interplay of different situations in the ontological sense of the term,”24 there may be multiple ways of conceiving the generic extension within one or many different assemblages.25 As mentioned above,26 the indeterminacy of the generic subset allows this redundancy to exist while the generic remains undecided.27 This also prevents the role of architecture from becoming a deterministic influence on the city.28 The urban situation is always developing, never complete, even if it lies dormant for a prolonged duration.

Mutual contingency of architecture and urbanism

All this having been said, care must be taken not to reduce the relationship between architecture and urbanism to a simple part-to-whole relationship, the two are inextricably and mutually contingent. While the urban precedes the architecture that develops it—urbanism is virtually first—it is not given material definition until the architecture enacts it—architecture is actually first. This is an identical relationship to the one between the monad and the environment and the same relation of bidirectional feedback applies here.29 Thus, we should also be wary of drawing too firm a divide between the two, urban assemblages—neighborhoods, quarters, ensembles, masterplans, communities—exist on the same plane in our flat ontology30 and are subject to the same oscillation between their components and their environments.31

§9.3 Conclusion

Summary of the thesis

Through the course of this text I have continuously endeavored to draw a line between a series of texts and sources (that may have appeared disparate at first) and to synthesize them into a single body of thought. My goal has been twofold: first, to define a contemporary theory of urbanism that is active, agile, and responsive; and second, to delineate a project for design that preserves the radical openness of urbanity in contrast to the often closed tendencies of masterplanning. Inspired by the concept of the Generic as described in this chapter—an untotalizable multiplicity, one that cannot be defined or known in advance, that exceeds any given formal system—I have sought to detail how such a concept would appear as an approach to these questions around urbanism.

My hypothesis has been that these two positions could be brought together by framing them in terms of themes that are shared by both urban processes and procedural modeling. The Interactive theme described the application of procedurality and the ways that it shapes an epistemological understanding of the situation.
To show that a generic set actually exists, Cohen develops a procedure whereby one adds it to the existing ground model as a type of supplement, thereby forming a new set. Within this new set, the generic multiple will exist at the level of belonging, or in meta-ontological terms, presentation. The new supplemented set provides the ontological schema of a historical situation which has undergone wholesale change.

Ibid. Introduction by Feltham and Clemens, p30

Badiou calls this "the 'generic extension' and it results from the initial situation being supplemented by its own generic subset as one of its elements ... (it) produces an anticipatory knowledge of the new situation, a knowledge that is under condition"

Feltham, Alain Badiou: Live Theory. p111

In Badiou's account "the subject falls outside the purview of ontology. Thus in Badiou's terms, ontology cannot think the being of the subject, but it can think its operation, which is forcing." The object-oriented ontology we have put forward, does not advance the same privileged position of the subject, but this point can be compared, with some substitution, to the point that local manifestations of the monad are acts and not definitive states and that the 'point of view' of a monad is a preceding condition that the monad comes to apprehend.

cf. §2.5a 'Pure interiority' and §2.6c 'Construction' especially note 98
Ibid. p111

Ibid. p64, 65

cf. §4.9d 'An inventive praxis'

Badiou, Infinite Thought: Truth and the Return to Philosophy. Introduction by Feltham and Clemens, p28

"Architecture" being interpreted broadly to include landscape design and open public spaces, not only buildings.

cf. §8.2b 'The multiplicity of architecture'

"a suitably subtractive or generic art, which is always an effort to devise new kinds of form at the very edge of what the situation considers monstrous or devoid of form."

Badiou and Hallward, "Beyond Formalisation: An Interview." p113

cf. §1.1 'Challenges and Shortcomings of Critical Urban Theory'

Feltham, Alain Badiou: Live Theory. p108-109

cf. §4.10b 'Architecture as urban interface'

Feltham, Alain Badiou: Live Theory. p109

Ibid. p108

Badiou, Infinite Thought: Truth and the Return to Philosophy. p174

cf. §2.6b 'Incompossibility'

cf. §9.1, note 4

"there is thus an unassignable gap between presentation and representation: there are incalculably more ways of representing presented multiples than there are such multiples," the spanning of this gap is never unilaterally completed, this is why architecture is only the sign of the possibility of architecture. (cf. §8.2b 'The multiplicity of architecture')

Feltham, Alain Badiou: Live Theory. p95-96

cf. §2.6 'Environment'
and includes the problematic concerns and possible reactions of a model. Ontological questions were addressed by the Generative, particularly the definition of objects along monadological lines: the internalization of virtual identities and extension of traits across the environment. The Reflexive function guided the actualization of intensifying urban dynamics into assemblages that self-organize in a network of heterogeneous agency. Finally, Entropic behaviors immanent to real assemblages worked to uniform assemblages. Though they frustrate the attempt of the theorist to provide concrete identities, they enact a necessary process of continual regrounding to prevent terminal stasis. The selection of these four themes is not meant to dictate the absolute terms of urbanism or urban design, however I believe that they do form a complete schema whose coherence and level of integration truly constitute a total theory of urbanism. This theory is certainly not complete, in particular, the design examples are probably too elementary to thoroughly explore the analogues between (computational) model and the city—an especially interesting field of research, I think.\textsuperscript{33} Much is being done to produce urban simulations that reproduce empirically measured phenomena, but much more needs to be done to explore the realm of the productive unreality and alterity that models can offer to the designer and to the city-dweller.\textsuperscript{33}

Ultimately, the generic itself gives the model for research: a series of inquiries that "cuts through and intersects with each category,"\textsuperscript{34} little by little drawing out the contour of a new future, unknowable but approached at each moment with new anticipation.

\begin{itemize}
\item \textsuperscript{32} "Modelling alone that renders … ontology ‘concrete’ by producing new knowledge of what is concrete in a particular domain … modelling itself, through its procedures of assigning specific semantic values to elements of a syntax, produces consistent procedures of change … this knowledge is valuable since it can pass, via philosophy’s encounter with generic procedures, into other practices."
\begin{flushright}
Feltham, Alain Badiou: Live Theory. p.133-134
\end{flushright}
\item \textsuperscript{33} Modeling that "attempts to enlarge its boundaries and disrupt any supposed completion" that "de-totalizes it and extends it".
\begin{flushright}
Ibid. p.132
\end{flushright}
\item \textsuperscript{34} Ibid. p.109
\end{itemize}
Bibliography:


Bibliography:


http://eumcad.wisc.edu/ge-koworks/ShowEvent.

http://www.euwiplantonline.net/edreader/Reader.aspx?token=vxr5BBthEX5m9sMyW25w%3d%3d.


——. "The Road to Objects." Continent 1, no. 3 (2011): 171–79.


The script reads values from a .csv file and formats them as a list of lists to be used as coefficients in assigning fitness values to a landscape.

05 Concatenate the folder path p and the filename.
06 Check that this file exists.
08 readline() loads the entire document into memory, on larger files this could be dangerous and using the iterative .readline() could be preferable (cf. A.6.2). It could also be substituted here, since we only examine one line at a time (:11), but it is possible we may want to compare values from different lines at some point.
12 Dividing the .csv using a comma delineator returns a list of the cell values.
17 We attempt to cast the third value of the line to a float to check if it is numeric. Comparing to the spreadsheet (Figure A.4.1) if there is not a numeric value, the line is either the header or a blank space.
19 If a data line, save the five values into the last list.
24 At the end of the loop, all values have been saved, with empty lists in the place of non-data lines. In the following, we reassign the relevant lines to new variables to increase readability in the code.
34 The coefficients are then returned to the main body of the code formatted and ordered.

The formatted spreadsheet to be read (cf. §5.5 "Extension").
The script retrieves the pixel data of an image file (.bmp, .gif, .jpg, .png, .tif) that maps values onto a plan of the site using a grid of reference points in the 3d model as sampling locations within the image.

Reference (including other methods):
http://www.grasshopper3d.com/forum/topic/dynamic-input-for-image

08 Concatenate the folder path and the image name to create the file location as a string.
10 `GH_MemoryBitmap()` is the most straightforward means of reading image data in Grasshopper. For other methods see the reference note above.
14 The images used have been scaled so that one pixel is equal to 1 meter, so the only conversion necessary is to subtract the coordinates of the upper-left corner from the sample points.
15 Image coordinates start with $y=0$ at the top row and increase as the rows count down, inverse of the geometry’s coordinate system.
17 The `Sample()` method returns a boolean and a color. Here, check that the operation was successful by reading the boolean returned.
18 In the definition, the reference images are .png files with a transparent background, where only the site bounds are opaque. This line checks whether the alpha channel of the current pixel is transparent, meaning it is not located within the bounds of the site.
21 The `Colour()` method can also be used to replace a pixel value by giving a color input as a third input value.
24 Always release the `MemoryBitmap` or the file will remain locked. Change the boolean field to `True` if changes were made that you want to be saved.

Figure A.6.1
The image map of the site coded with user-defined attraction values (cf. §6.3 ‘Mobile Agents’).
```python
from collections import defaultdict
import math
import os.path as osp
import Rhino.Geometry as rg
import scriptcontext as sc

# Appendix 6.2 OSMpoints

# Define the function that reads the OSM file
def PlotLatLonOnXY(lat, lng, val):
    xVal = val[2] + ((lng - map[0]) * map[4])
    yVal = val[3] + ((lat - map[1]) * map[4])
    pt = rg.Point3d(xVal, yVal, 0)
    return pt

# Define the function that checks if a file exists
def readFile(fileName):
    if not os.path.exists(fileName):
        print('File does not exist')
        return False
    else:
        file = open(fileName, 'r')
        lineStr = file.readline()
        while lineStr:
            lineSpl = lineStr.split('"
')
            if lineSpl[0] == "<node id=":
                inNode = True
            if inNode:
                lineStr = file.readline()
                userNode = lineStr
                latVal = float(lineSpl[1])
                lngVal = float(lineSpl[2])
                inNode = False
            elif inNode:
                if latVal < brckt.01 or lngVal > brckt.01 or ...:
                    latVal < brckt.01 or lngVal > brckt.01:
                    inNode = False
                elif latVal < brckt.01 or lngVal > brckt.01:
                    print(lineSpl[1])
                    break
                else:
                    lineStr = file.readline()
                    lineCnt = 1
                    ptID = [val[0] for val in sortNode]
                    pt = [val[1] for val in sortNode]

# Main function
if __name__ == '__main__':
    readFile(fileName)
```

The script reads an `.osm` file from OpenStreetMap and creates points for each "node." Then polygons are created from the "way" data and saved with their tags in a dictionary, so that they can be selected or sorted through this metadata. Note that the .osm formatting changes from time to time and the precise data position or identification may need to be adjusted.

---

16 This function maps the longitude and latitude values to a rectangular plot.
32 The file is opened in read-only mode with the "r" tag.
33 `readline()` returns the first line of the file and iterates to the next line everytime thereafter.
35 `lineStr` will only be empty at the end of the file or if the file is completely empty.
36 Use quote delimiter to separate values.
38 Check the first value in the line for the tag that identifies a new point location.
42 Only three values are of interest for the node: its index number, and its latitude and longitude.
47 An optional UVInterval input, `brckt`, can be input to limit nodes to only those within a certain boundary.
53 The index value of the node and a `Point3d` are saved as a tuple in list to be sorted. This will help with recall later.
56 If the line begins with the tag that identifies a new way, the `while` loop is broken and the `while` loop exited.
62 The node index values and point locations are converted to lists with list comprehension.

idTemp = 0
wayBool = False

# MAKE WAYS
while lineStr:
    lineSpl = lineStr.split('"
    
    # BEGIN WAY
    if lineSpl[0] == "<way id=";
        idTemp = int(lineSpl[1])
        wayBool = True
        pl = rhPolyline()
        keyList = []
        valList = []

    # WAY IS ALREADY BEGIN
    if wayBool:
        # BEGIN WAY
        if lineSpl[0][2] == ":":
            if len(pl) > 0:
                sortWay, idTemp = append(pl)
                sortWay, idTemp, append(keyList)
                sortWay, idTemp, append(valList)
                plOut, append(pl)
                wayBool = False
        else:
            pass
        
            # END TAG TO WAY
            else:
                keyList, append(lineSpl[1])
                valList, append(lineSpl[3])

    # RELATIONS BEGIN
    if lineSpl[0] == "<relation id=":
        print (lineCnt)
        break

    lineStr = FSM.readline()
    lineCnt += 1

# always close()
pt = OSMpath
crv = plOut
pyWay = sortWay
OUTPUTS: ptb, ptID, pt, crv, pyWay

Figure A.6.3
Nodes and ways as they are represented in the .osm file.

Figure A.6.2
Ways as they appear in the geometry preview.
Curriculum Vitae

EDUCATION

2016  Doctor ès Sciences (Ph.D.)  
École Polytechnique Fédérale de Lausanne  
‘Assemblage Form: An ontology of the urban generic with regard to architecture, computation, and design’  
Funded research: Food Urbanism Initiative, 2011-2014

2009  Master of Architecture (MArch I AP)  
Harvard Graduate School of Design  
‘Taipei 2.0.2: Computation and the urban generic’  
Digital Design Award, 2009

2006  Bachelor of Science in Architectural Studies (BSAS)  
University of Wisconsin–Milwaukee SARUP  
Magna cum laude  
Honors College  

TEACHING EXPERIENCE

2014  Singapore University of Technology and Design  
Adjunct Assistant Professor  
Undergraduate Design Studio:  
Fall 2014  ‘Core Studio I’

2009-13  École Polytechnique Fédérale de Lausanne  
‘Assistant Scientifique’ / ‘Assistant Doctorant’

Masters Design Studios:  
Fall 2012 and Spring 2013  ‘Morphogenesis: Parametric Typologies.’

Spring 2012  ‘Morphogenesis: Growth Typologies.’

Fall 2011  ‘Organicités: Food Urbanism Lausanne.’

Fall 2010 and Spring 2011  ‘Organicités: Ras-al-Khaimah Ecological Campus.’

Spring 2010  ‘Organicités: Tectonic Differentiation.’

Fall 2009  ‘Organicités: (Un)Natural Selection.’

Masters theses supervised as ‘Mâitre’:

2013-14  
Christina Haas, « La mer d’Aral, d’une zone sinistrée à un parc du patrimoine. »
Charles Sarasin, « Métahutong: une couture entre quartiers traditionnels. »
Alexandre Sadeghi, « Anamnesis. Plan urbain évolutif. »

2012-13  
Anne-Catherine Gay, « Integrative Network. Ecoles techniques et parcs urbains pour l’intégration des “barrios” et de ses jeunes dans Caracas. »
Johann Watzke, « Hybridation d’infrastructure et d’architecture dans un quartier de Lausanne en pleine mutation. »

2010-11  
Andrea Pellacini, « La mort dans la vil(l)e: Création d’un complexe funéraire au sein de Sara D. Roosevelt Park. »

Undergraduate Design Seminar:  
May 2010  ‘HOME 2.0: Parametric Quality’

2012  Universität für Angewandte Kunst, Wien  
Workshop Instructor  
Design and Fabrication Workshop:  
February 2012  ‘IoA Spring Challenge.’
2007-09 Harvard Graduate School of Design
Teaching Assistant

Design Studio and Seminar TA:

Spring 2009
‘Verticalism.’ critics Ilhak Abalos and Daniel Lopez-Perez.

Fall 2008

Spring 2008

Fall 2007 and Spring 2007

Digital Media Workshop Instructor:

Spring 2008-Spring 2009
Rhino3d, Rhino VScript, Grasshopper3d
MELscript for Maya

Spring 2009
Workshop for ‘Mat Ecologies.’ critic: Chris Reed
Workshop for Fourth Semester ‘Core Studio.’ critic: John Hong

CAD/CAM Technician:

2008-09
6-axis robotic waterjet and milling machines

PUBLICATIONS

Research Articles as Primary Author:

2016 “Performance Review: In Praise of the Possibility of Architecture.”

2015 “Generative Masterplanning Inspired by Cellular Automata with Context-specific Tessellations”

2015 “Visualizing Data: Qinglonghuizhen, Peri-Urban Beijing.”

Trevor Patt. in e-publicspace.net 2014.

2014 “Scenario Modeling for Agonistic Urban Design.”

2013 “The Surface of Borromini.”

2013 “Food Urbanism Modeling.”

2012 “Computation as an Ideological Practice.”
Nathaniel Zueleke, Trevor Patt, Jeffrey Huang, in ACSA 100th Annual Meeting: Digital Aptitudes + Other Openings, Marc Goulthorpe and Amy Murphy, eds. ACSA, 2012.

2011 “Taipei 2.0.2: Computation and the Urban Generic.”

2010 “The Collective Image: Form, Figure, and the Future.”

2008 “Surfacing Stone: Digital Stereotomy and Material Effect.”
2007  *Skyecar City.*  
Winy Maas and Grace La eds. Actar. 2007.

Research Articles as Secondary Author:

Mark Meagher, Jeffrey Huang, Nathaniel Zuebske, Trevor Patt, Guillaume Labelle, Julien Heimbreni, Simon Peter, Thomas Favre-Bulle. in *Digital Creativity,* v.26, 2015.

Jeffrey Huang, Trevor Patt, and Peter Ortner; in Foreign Designers Venture into China’ Landscape Architecture Frontiers, v.1n5, October 2013.

2013  “Architecture Challenge 2012.”  

2012  “Architecture Spring Challenge.”  

Non-Authored Publications Featuring Design Work / Media Exposure

2015  “Growth Typologies, Localities and Defamiliarisation: Experiments with Artificial Urbanism in Sichuan, Guangzhou and Beijing.”  

2013  “Topographie des Senses.”  
in Pabellones experimentales, EPFL Lausanne’ in AV Proyectos, n057, 2013.

2010  “Chinatown Storefront Library.”  

2010  “Chinatown Storefront Library.”  


2010  “SkyCar City: Metropolis 2.0.”  

2009  “Taipei 2.0.2: Computation and the Urban Generic.”  

2009  “Lending a Library.”  
Joan Wickersham. in Architecture Boston. v.1n4 Winter. 2009.

2009  “Argos Tower.”  

2009  “More Bang for the Buck.”  

2010  “The Return of the Future: A Second Go at Robotic Construction.”  

2008  “Surfacing Stone: Digital Explorations in Masonry Curtain Wall Design.”  
PRESENTATIONS AND LECTURES

2015  “City and Model: Commingling of Formal and Informal Urban Agency”

2015  “The self-othering event of object qualities and architecture as a mutable interface.”

2014  “Notes on Assemblage Form.”

2014  “Scenario Modeling for Agonistic Urban Design.”

2012  “Negotiation between aesthetics and execution in architecture.”

2012  “Computation as an Ideological Practice.”

2011  “Procedures: Designing from the bottom up.”
Invited lecture: the Gerald D. Hines College of Architecture, University of Houston. 9 September, 2011.

2011  “Taipei 2.0.2: Computation and the Urban Generic.”

2011  “Closing Performance: In Praise of the Possibility of Architecture.”
Symposium speaker: Architecture is All Over: organized by Work Books with UT, OCAD, TIFF, Toronto, Canada. 11-12 February, 2011.


AWARDS AND RESEARCH GRANTS

2011-14  “Food Urbanism Initiative”
3-year cooperative grant; project lead: Craig Verzone.

2009  “Digital Design Award”
Awarded by Harvard Graduate School of Design.
EXHIBITIONS

Curated:

2015 “Land-/Sea-Scape”  
Organized and designed at Galerie Antenne, Lausanne. 27 June, 2015.

Included in:

2016 “Tell me about a Rhino command: architectural history through change logs”  

2013-14 “Totally Lost”  

2012 “Cosandey Pavilions”  

2012 “AAG Video Panorama”  

2010 “HOME 2.0”  

2010 “Post-Oil-City – The History of the Future of the City”  

2010 “Emerging Professionals”  

2010 “Futures in the Present”  

2009 “GSD Platform 2”  

2009 “BEYOND MEDIA Festival”  
Florence. 2009.

2008 “(Im)material Processes”  

2008 “Venice Biennale”  

2007 “Studioworks”  
PROFESSIONAL EMPLOYMENT

2012- Ahtehha
   Principal: SUTD DMaD Laboratory, sg50 Innovation Landscape with Immanuel Koh, et al.

2009-15 Convergeo
   Designer: Pavillons Cosandey, Rolex Experience, Rolex Flagship

2009 New World Design
   Intern Architect: Nasinyyah Truck Stop

2006-09 Freelance Design/Computation Consultant
   MOS
      2008-09
      Computational Designer: Necklace Dome, Desert Islands #2, Museum Boijman, Warsaw
   Preston Scott Cohen Inc.
      2008
      Scripting Consultant: Tel Aviv Museum of Art
   SEED Magazine
      2008
      Visualization Consultant: 2008 WEF
   MVRDV
      2006
      Visualization Consultant: Istanbul Waterfront

2007 SsD Architects
   Intern Architect: Hidden Fortune House

2006-07 La Dallman Architects
   Intern Architect: Levy House, Discovery World

NON-PROFIT WORK

2009 Chinatown Storefront Library
   Design and Fabrication with Department of Micro-Urbanism, Boston Street Lab, Harvard Community Service Fellows

EDITORIAL POSITIONS

2016- Frontiers Journal
   Associate Editor, Digital Architecture (Specialty section in Digital Humanities)