Summary Report

1. Research Question and Aims [250 words]

2. Context and contribution[1000 words]

3. Case Studies and Thesis structure and [500 words]

4. Methodology [750 words]

5. Timetable

6. Diagram of thesis structure

1. Research Question and Aims

The ability to generate complex computational models of architectural design proposals is rapidly increasing, but critical reflection on the consequences of such advances on architectural practice are not keeping pace.

Geometry and data are produced in huge quantities during the design life of a project and the digital models that such data constitute are used to simulate a building's performance, across a range of criteria, and to decide whether it is suitable to be built. Commercial design and analysis software mediates design processes and policy in the built environment as never before. But by what means are such models and simulations created and in what ways does the nature of the software itself mediate the process and contribute to the outcomes?

My research takes, as its starting point, the view that it is the closed nature of the software itself that creates this unacknowledged mediation and in this project I will explore methods of open source in order to address the following questions;

- How might the processes of computational design and simulation differ through being focused on the means of open production rather than digital products?

- How can such an open source methodology be implemented?

- What would be the tools of such a methodology and how would they operate?

- How is scientific knowledge created through spatial simulations and how can this process be 'open-sourced'?

2. Context and contribution- *1000 words*

Many forms of contemporary architectural practice are increasingly characterised by the use of complex, sophisticated software for the design, analysis and production of projects. Previously, software such as CAD has been skeumorphic – mimicking non-computational techniques such as hand drawing – and predominantly providing improvements in speed of production and organisation of design material. However, current developments in the field of architectural software, such as BIM, represent a significant change in the relationship between the architect and the computational tools he/ she employs.

Single design models can contain full geometric information - at a 'virtual' 1:1 scale – as well as massive amounts of data related to construction cost, environmental performance, energy use, structural stability. The tools for creating all of this data lie within a limited selection of proprietary, commercial software packages. The ability to access the raw data contained within such models, as well as the means to analyse and interpret it is limited to those with access to the software itself. Furthermore, the extent to which such technology mediates design processes is poorly understood.

Simulation in spatial design

The use of simulation in spatial design includes 'well understood' applications such as structural design packages (for sizing steelwork members) and environmental deign software (used for predicting internal temperatures and energy use). In both cases, they are used to provide an evaluation of the predicted performance of the proposed design and in order to do this, the spatial deign model is placed within a simulated world, based on abstractions and simplifications of things such as gravity, wind, weather. In one sense, these kinds of approaches are a development of 'traditional' approaches to the technical aspects of design, where the traditional, by-hand calculations have been replaced with rapid and multiple iterations of complex design scenarios.

There also is another strand of simulation emerging in architectural design. Approaches such as *Space Syntax*' [1] represent a break from simulation as augmentation to a pre-existing design operation. Space-syntax analysis investigates characteristics of space such as 'depth of view' – that is how far can someone see – or axial connectivity [REF] within geometric models of buildings and urban environments. The results of this analysis are used to evaluate the performance of a design proposal. This approach to spatial analysis is not an augmentation of existing design processes through increased computer processing power, it exists as a direct consequence of that computing power.

1 Space Syntax – write something here

These two strands of simulation – augmented processes and invented processes – appear very different and on the surface their only real similarity seems to be that they are both reliant on the availability of computing power to support their use. However, this reliance on computing power goes deeper than a question of 'horse-power'. In deploying such power to create complex simulation models both approaches create a difficulty in evaluating and challenging their outcomes.

In the recent past, research into building performance was translated into rules of thumb guidance, that could be applied 'manually', or laborious hand calculations of daylight may be required. Rapid increases in computing power and software sophistication have changed the way in which these calculations are made and conclusions are drawn. It influences not just the types of space that may be constructed, but it shapes the regulatory framework in which the designers operate.

The carbon buzz [REF] initiative creates a platform for building designers and operators to compare simulated and actual environmental performance. While it is currently far from a comprehensive database [the project is still in its early stages], it highlights very clearly three crucial issues. The first, and most obvious, is the discrepancy between the 'designed' and the 'actual' performance and actual energy use is shown to be significantly higher than predicted. In some cases the difference is so significant that if the 'true' figures were presented at the design phase, the project would not have met the relevant regulatory requirements. Much valuable research has been and continues to be carried out identifying the reasons for such discrepancies, examining things such as end user familiarity with the building's control systems. But there is little focus on designers familiarity with issues of adopting such computational simulation in the first place. The second aspect is linked to the ease with which this discrepancy can be identified. Both the predicted and measured performance of building is reduced to a single 'grade' - A, B, C etc. The complexity of the simulated model - thousands of triangular faces of geometry, lists of materials, calculations made for every hour of every day - has been simplified to a single character. The final observation is that each design proposal is evaluated and presented in isolation and is not situated in any way. There is no link to its wider 'real-world' context and the simplified environment of the simulation is transposed into the world 'as is'.

[IMAGES OF CARBON BUZZ WEBSITE]

Although an invented process such as space syntax, does not [at least yet] have the same impact on policy, it can certainly be described as returning the results of simulation as reality.

Space syntax, although far from being a common approach in architectural and urban design, has certainly 'left the lab', and has been used on many high profile projects. Even though it is not a widespread approach it is still very difficult to challenge the findings of a study like space syntax. It is of course, supported by significant and 'robust' research but is also the nature of the simulation itself that makes contesting it difficult. If I wished to challenge the 'accuracy' of the simulation I would be, by implication, accepting the ontological validity of the approach. I would need a kind of anti 'space syntax' device in order to fully challenge it's usefulness. In returning this idealised virtual model directly into the 'real world', Space syntax creates its own essentialist ontological position. That is to say that it makes a claim that that there are characteristics of spatial assemblies that can be found, understood and harnessed. The complexity of the computer technology used to generate this position is the means by which it is reinforced.

[IMAGES OF SPACE SYNTAX]

Both of these examples sit on spectrum of computational simulation that is currently employed in architectural practice. At one end, computational augmented simulations of environmental performance directly inform the policies than regulate construction. At the other end, computationally-driven simulations of abstracted spatial concepts provide seemingly incontestable analysis of 'good' design. In each case, the ability to engage with the way in which they are produced and the findings they produce are limited to a very narrow professional field.

The contributions of this research project in the field of digital simulation for spatial design are;

1. ???

2. ???

3.???

4. ???

3. Methodological approach

open source methodologies

My research adopts a novel methodological approach concerning opens source to direct both the analytical and design-based case studies. This involves going beyond conventional 'convergent' methods of open source and employing more contemporary 'divergent' methods.

Convergent models of open source

'Open source' is now a widely understood concept, at least on a very broad level. In terms of software development, at its basic level open source involves the release of a software's 'source code' allowing the user the freedom to;

1. use the software for any purpose

2. study how it works

3. redistribute copies

4. make modifications

[REF]

The difficulty with the most basic use of these freedoms, in terms of software development has been twofold. Firstly, an expert community of coders needs to coalesce in order to share and contribute complex source code which has been made freely available. The release of the source code does not in itself widen the pool of potential contributors beyond the expert knowledge that would have been required whether or not the code was open source [REF]. The second, common criticism of such simple models of open source software is that it merely replicates commercial programs rather than providing creative alternatives.

In architecture the software tools employed in architectural design practice are typically commercial and closed, therefore placing certain limits on the ability of non-professionals to adopt or adapt them. The focus has instead been on the release of architectural design as 'source code'. The 'wiki' has been commonly adopted as the method for the release of this 'source code'. The 'open source' manifesto for architecture is a multi-authored page of wikipedia itself. The manifesto identifies the need for various 'standards' including software and hardware interoperability. Initiatives such as 'wiki house' [REF] offer 3D models of designed and verified component and assemblies for users to download, manufacture and build their own project.. The project Wikiplaza [REF], by Hackitectura, offers a more diverse resource within it's 'source code' and asks the question 'what would it be like to build a wiki?'. Their proposal, conceived as 'copy left' concept, aims to take the methodological approach of the digital wiki and apply it, almost directly, to the physical world.

Divergent models of open source

Although both of the above approaches have contributed to and supported the idea of an open source methodology they display an underlying characteristic that is problematic. Both strands are based on 'convergent' models of open source, that tend towards a single, stable outcome.

More recently, there has been a resurgence in 'divergent' methods of open source based on 'versioning' technology. 'Versioning' is a technological solution to the problem of managing multiple copies of software programs (and parts of programs) that are at various stages of development and stability. The core functionality of versioning is the ability for multiple uses to 'branch' individual components of a piece of software assembly, whilst retaining the functionality of the full assembly. This feature allows for multiple, co-authored versions to exist simultaneously. Users can then modify files and 'commit' them back to the assembly, 'merging' them with other branches of the project. This process creates a complete, genealogical trace of each step in the development of a project that can be accessed at any moment during a project's life.

The wider significance of versioning, beyond its use in software development has most interestingly been explored in the manifesto for 'Urban Versioning' [Haque and Fuller]. They propose a methodology of versioning for architectural and urban design that challenges conventional approaches to design, making and authorship.

Tools for a divergent open source methodology

I will adopt methods associated with an emerging field in social science research - Software Studies. In

proposing a 'Manifesto for Software Studies', Kitchin and Dodge identify 'genealogies' as a suitable method for research into how code emerges. Whereas, Latour follows 'materiality' through the scientific processes, a genealogy of software traces the code, the iterations and the unfolding versions. This method will be employed to construct a genealogy of existing software as well as informing the means by which future software should be created.

Github is a versioning technology and platform for 'social coding', based on the popular Git versioning software. As with much versioning software, Github creates structured databases of changes to code within shared projects, in the form of 'branches', 'merges' and 'commits'. While Github itself provides some analysis tools, the structured data sets that it produces can be extracted for custom analysis and visualisation.

code traces mapped [goarse, gephi, my openPHD]

3. Thesis structure and Case Studies [500 words]

The thesis is structured around the following three, thematic strands;

1. Science and SIMULATION

Simulations represents the potentials and dangers of computational technology. They risk taking the simplified abstraction of the complexity of the world and return it as it's underlying structure.

This section will explore the following:

- A comparison of a constructive and realist ontological position with regard to scientific knowledge (Latour & De Landa) and the use of simulation as a means of producing scientific knowledge

- The role of narrative in an autopoeitic understanding of simulation (Hayles & Varela)

Virtual INTERFACES - post humanisms, digital materialism and the myth of the virtual

SUMMARY

Digital traces – ALGORITMS and CODE

SUMMARY

Thesis Structure

Introduction – Background, Context and Contribution

Methodology

Towards a new form of open source in architecture

Section 1. Science and SIMULATION

Chapter 1 Ontologies Of Science - Constructing Knowledge

Chapter 2 Simulation and Narrative

Case Study 1

Section 2. Digital traces – ALGORITMS and CODE

Chapter 6 Software As Code

Chapter 7 Code And Space

Chapter 8 Algorithms As Stable Code

Case Study 2

Section 3. Virtual INTERFACES - post humanisms, digital materialism and the myth of the virtual

Chapter 3 - Post Humanist futures

Chapter 4 - Digital Materialism

Chapter 5 - Myth Of The Virtual

Case Study 3

Case Studies

Each of these three case studies sits within one of the thematic strands described above;

SIMULATION - CASE STUDY 1.

GENEALOGICAL SOFTWARE MAPPING OF 'SPACE SYNTAX' SIMULATION TECHNIQUES

- 'Genealogical' analysis of 'Space Syntax' software tools

This case study will create a 'genealogical trace' of the development process that Space Syntax techniques of 'axial mapping' and 'visibility' mapping have undergone since their inception in the late 1970s. This case study will use as its source material the extensive record of peer reviewed material that supports the Space Syntax research project. By adopting a genealogical approach, and 'following the code', the intention is to map through translation and codification the capabilities and functionality of each technique and therefore provide a critical narrative around its construction and contemporary use.

CODE/ ALGORITHM - CASE STUDY 2.

DYNAMIC DATA STREAMS FOR DATA DRIVEN DESIGN

- Creating 'open', publicly accessible streams of computational design data geometry and simulation data, specifically environmental data

This case study is about the design and creation of 'open streams' of design data for public access. It will explore the issues around interoperability through the use of novel file types and forms of data and will use twitter as the means of aggregation and distribution. One of the outputs of this will be a software plug in for a 3D modelling package. This new technological innovation will be tested and demonstrated through a short series of student workshops.

Technical aspects of this case study have been developed as part of my @simBot pilot project. This was used and tested as part of a workshop in environmental design in July 2013

IMAGES FROM @simBot *PROTOTYPE*

INTERFACE - CASE STUDY 3.

@SIMBOT AS COMPANION SOFTWARE

- *A personal, customisable, companion 'app' capable of 'reading' to dynamic data streams*

This case study creates ways in which the design data streams developed in case study 2 can be used as design and teaching tools. It will focus on the design and use of a prototype mobile phone 'app' that is capable of navigating and 'interpreting' the data feeds, as described above. The app will be designed in such a way as to allow customisation and for the user to create their own 'companion' app. This new technological innovation will be tested and demonstrated through a short series of student workshops.

Technical aspects of this case study have been developed as part of my @simBot pilot project. This was and tested as part of a workshop in environmental design in July 2013

IMAGES FROM @simBot PROTOTYPE