The Green Grasshopper: Approaches to Teaching Computational Design Methods in Landscape Architecture

Philip Belesky[1]

[1]RMIT University, Melbourne/Australia · philip.belesky@rmit.edu.au

Abstract: As computational design methods grow more common in landscape architectural education and practice there is a growing need for pedagogical approaches that teach these new methods in the terms of our own discipline. In contrast to established approaches, that largely cater to the tasks involved in design buildings, a landscape context requires a re-framing of how computational methods are used; often involving a shift away from form and towards landscape dynamics as the prime driver of design development. This adaptation is discussed here in terms of several key approaches developed when using parametric modelling as part of a teaching practice across a range of contexts.

Keywords: Parametric modelling, parametric design, computational design, teaching, education

1 Introduction

The use of computational design methods is increasingly common in landscape architectural education − whether introduced as part of the core curriculum, a studio brief, or explored through a student's self-guided learning. Yet, in contrast to architectural education, instruction in computational methods is less widespread, typically less-supported, and has fewer resources that cater to the distinct characteristics of landscapes (WALLIS et al. 2014). As a result, the pedagogy of computational design in landscape architectural education remains challenging − even as experience in this area is becoming more widespread in practice and increasingly expected of graduates (FRICKER et al. 2013).

This paper discusses a number of approaches for teaching computational design in the specific context of landscape architectural education and contrasts them with the de facto approaches developed by other disciplines. These approaches draw from a diverse range of experience in teaching computational methods across both landscape architecture and architectural programs; in a variety of settings that range from design studios to representation-focused courses to specialised electives; and across experience levels that span from 1st-year undergraduates through to a postgraduate level. While the learning outcomes of each context varies dramatically, each case employed parametric[1] approaches (using the *Grasshopper* plugin[2] for the *Rhinoceros* modeller) to explore these broader goals using computational methods. Put briefly the parametric modeling of geometry allows users to define shapes (and

[1] Parametric modeling is but one particular technique within the broader range of computational techniques available. That said, Grasshopper allows for advanced students to explore methods such as scripting, agent-based modeling, and evolutionary solvers that go beyond a parametric paradigm.

[2] Grasshopper is a common tool used across education and practice settings for 'computational' approaches to design development; a factor driven by its (relatively) intuitive interface and the high amount of extensibility offered by its plugin system and active online community.

Journal of Digital Landscape Architecture, 3-2018, pp. 406-413. © Wichmann Verlag, VDE VERLAG GMBH · Berlin · Offenbach. ISBN 978-3-87907-642-0, ISSN 2367-4253, e-ISSN 2511-624X, doi:10.14627/537642043.

other data) according to specific *parameters* (say a number, or a coordinate) rather than by manually drawing or sculpting a mimetic representation using media such as paper/pencil or the sculptural tools of 3D modellers. As a simple example, a user might create a circle parametrically by defining explicit numeric values (i. e. parameters) for its origin point and radius. The resulting circle could then become a parameter for a successive operation (such as creating a column via a vertical). Modifications to the original circle's parameters would then automatically affect subsequent operations that depend on them (such as the width of a column). *Grasshopper* provides a 'visual scripting' interface that makes these approaches (relatively) accessible to novice designers through a diagram-like methodology for developing a parametric model whereby users place small 'nodes' onto a canvas (that represent functions, such as creating a point in space) and 'wire' them together to create chains of successive computational operations (such as creating a point with specific XYZ coordinates or using a series of points to draw a line). This is essentially a visual representation of a *data-flow* programming paradigm (DAVIS 2013) whereby information transfers and transforms in a linear manner to cumulatively develop a more complex model.

2 Form and Landform

As with any other area of instruction, educators should tailor even a highly-specific teaching goal to the wider curricula and to the abilities and interests of the student cohort. This is especially true for heavily technique-based instruction, such as computational design, which itself has no single clear purpose but can instead accelerate or alter how designers approach a wide variety of design tasks. Nevertheless, there are common threads to the various approaches for introducing computational design methodologies. These span across the enabling software tools used, the design techniques they afford, their relationship to established disciplinary concepts, and how they play into related developments in design technologies such as digital fabrication processes or new methods of site survey. Importantly, in each of these cases, there are distinct differences to the approaches commonly used in architectural education − the typical point of reference for how to teach computational design techniques in the built environment disciplines. Many of these standard approaches become counterproductive when deployed within a landscape architectural context.

For example, in many landscape projects the design process typically begins with conceptual moves at larger scales and only later proceed to smaller-scale form-making. In contrast, a typical approach to parametric design in architecture encourages beginning design development at a medium-scale (say that of a façade or building section) and progressively developing more detail 'upwards' (by combining the forms that define a building's envelope or by refining forms in relation to the site context) or 'downwards' (by modularising a medium-scale form through smaller levels of formal detail that refine aesthetic or tectonic concerns). Such design moves (when performed parametrically) are more valuable when they can create explicit relationships to the formal aspects of a design that are still in flux. A designer could run a set of shading analytics prior to massing development; but they would become much more valuable when enacted as series of parametric rules that use that very analysis to (for instance) quantitatively evaluate different massing options or automatically optimise formal details such as louvre orientations. Thus, the task of defining a building's envelope, or some other mid-scale attribute, acts as a linchpin design task that can be informed by initial site analysis and inform subsequent analysis.

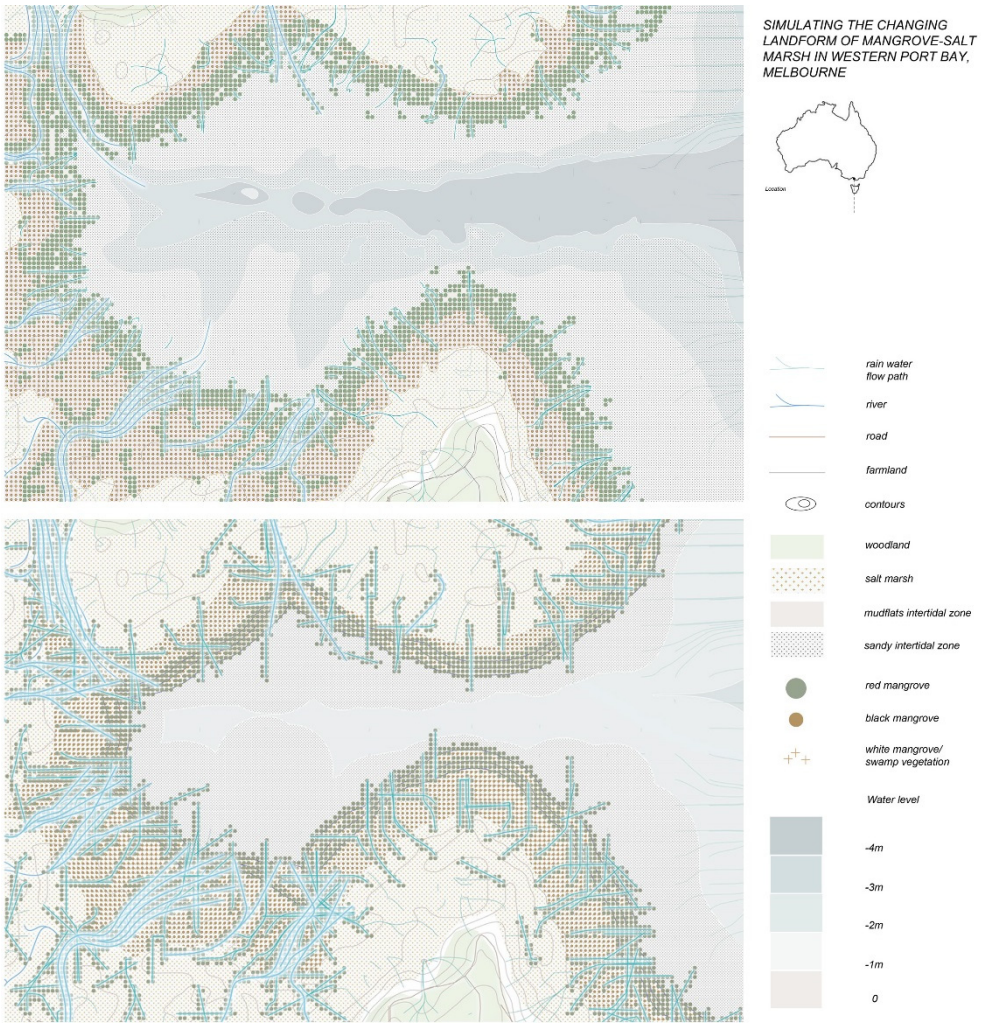

Fig. 1: Extracts from the work of Sharon Wu completed as part of the first year of the RMIT's Master of Landscape Architecture program. The maps show how generalizable parametric techniques (in this case a method for simulating surface water flows and vegetation spread) can be used to model the evolving conditions of a mangrove wetland system. After developing this parametric model Sharon then used it to test design interventions using the parameters available for simulating different time periods (the top map is +0 years; the bottom +50 years) and the effects of design proposals (such as upstream improvements to sediment flows and water quality or improved maintenance regimes that promote revegetation).

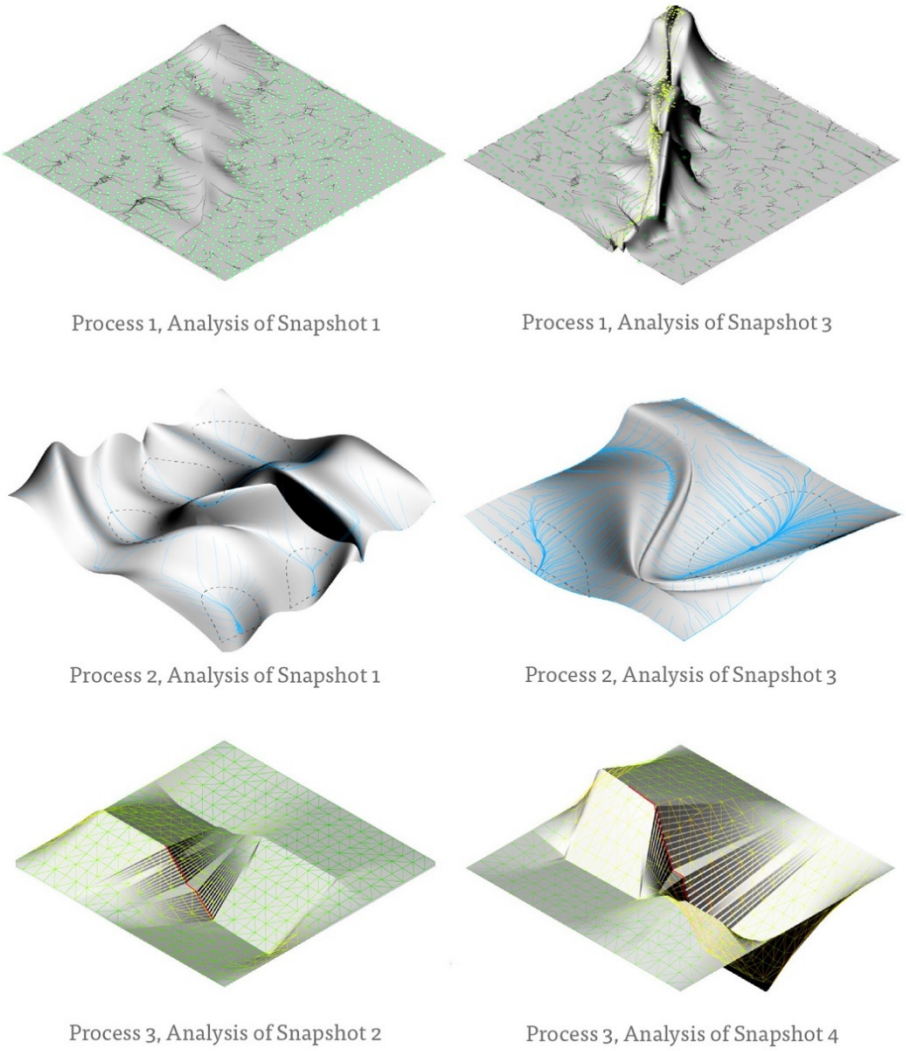

Fig. 2: Extracts from the work of Louella Exton for the Communications 2 course that is completed as part of the second year of RMIT's Bachelor of Landscape Architecture program. Each of the two sets of 'process snapshots' shown here are renders showing a particular parametric model developed by the student (the 'process') with a given set of parameters (the 'snapshot'). Each parametric model had two stages: a process of surface generation and a subsequent process of analysing or simulating how that surface performs according to a particular landscape characteristic. Here the surface was generated to emulate geological process of uplift, erosion, and fracturing respectively (top-to-bottom) while an overlay visualises a projection of vegetation growth, surface water flows, and grade respectively according to how that surface form of the landscape is changing at different time periods (left-to-right).

Within landscape projects there is usually less of an obvious 'first-step' in formal terms that initial analysis and concept design work can pivot towards. Instead, that initial step is something that needs to be itself constructed from working with a given site's conditions. As a result, when beginning landscape architectural design development *parametrically,* most larger scale site analysis or concept-stage work either precedes the shift to the parametric environment or is itself the paramount design task to resolve before a designer proceeds with more 'formal' moves that manipulate the existing site conditions.

This lack of a formal entry point into design development is seemingly problematic in a parametric modeling environment because most common resources for using Grasshopper and similar tools focus on developing Non-Uniform Rational B-Spline geometries (NURBS) as a key means of design development due to their capacity for quickly creating and manipulating complex forms. However, the reduction of landscape form to a plastic NURBS geometry encourages designers to see landscape form and material as a homogeneous and inert entity (RAXWORTHY 2013) in a manner that runs contrary to the disciplines' emphasis on dynamism and differentiation. Thus, to the extent that a CAD environment represents a site model − say as a NURBS surface in Rhinoceros − the manipulation of that model becomes the obvious 'first step' for design development even as those manipulations are largely limited to highly sculptural modifications to a landform whose 'model' lacks detailed information about ground cover, soil texture, and innumerable other contextual details that could normally inform or create design choices.

3 Reframing Parameters

In addition to these specific concerns, the broader (apparent) framing of computational design within an architectural context means that it is easily seen as just a method for driving complex form-making and/or performance-driven approaches that simulate the effects of building phenomena. This perspective, left unchallenged, portrays computational methods as alien to a landscape context where these tasks are less prominent. This minimises the potential of computational methods by rendering their applications as responding only to niche concerns. However, alternative framings can provide perspectives that see computational methods, and their results, through a landscape architectural lens that portrays their value in much more expansive terms. Doing so allows students to locate the use of computation in relation to their pre-existing interests within the discipline, which in turn offers them more agency in how they approach the task of learning and adopting new design methods (ROUDAVASKI 2012). That teaching new − and in many ways unusual − design techniques would benefit from being strongly located within a disciplinary context is of course not surprising. Yet, exactly how to do so in the particular case of computational design methods is not intuitive nor widely supported by accessible education resources.

Successful approaches in most cases seem to involve a re-orientation of the implicit purpose of a typical parametric model: to develop or test geometric details at levels of complexity that are difficult to manage with traditional approaches to analogue or digital modelling. The dynamism inherent to parametric modeling means that while the end result of a model may be geometric, the process by which that geometry is constructed can be rationalised in a number of different ways. When modeling forms or phenomena in a landscape context, the inputs or outputs of the model may still be form-dependant but the manner in which those forms are

constructed can be framed as a task of understanding the operations and effects of landscape processes and *then* translating those into geometric representations that explore tendencies rather than fixed outcomes. This contrasts with the standard approach to geometric development where the construction of formal outcome is designed by an instrumental goal of the designer's choosing − such as sculpting a pleasing form or rationalising a form into constructible components. For example, processes of erosion can be understood through geometric processes of division and warping whereby the input (initial site topography) and output (projected site topography) can be mediated between by actively constructing parametric relationships that quantify exactly how erosive action impacts landform. In cases such as these, parametric models offer a means to actively explore landscape processes (as is often emphasised in contemporary theory as per WALLIS) in a manner that goes beyond general tendencies, and instead becomes an instrument for specific design exploration due to the explicit yet flexible nature of the parametric model.

In a similar manner, the parameters that drive the expression of parametric geometries are not necessarily just capturing design intent (whereby users might 'dial in' whatever values create an interesting outcome) but can be a means to quantitatively assess site conditions and dynamics by uncovering their *specific* relationships. Hydrological conditions, such as surface water pooling or flooding, can be examined as a general effect but also refined to specific values derived from site research and observation that can then help to test the designed outcome. Even better, parametric approaches offer the chance to go beyond purely 'layer cake' approaches to site analysis and instead begin to examine the relationships *between* diverse landscape conditions − such as linking rainfall levels to specific spatial flooding extents − and then using this to correlation to extrapolate a more general model.

This capacity for parametric malleability to be understood as not just a means for design exploration but as a model of landscape operations can then be used to better investigate a fundamental aspect of designing landscapes: time. While modifying parameters is typically seen as an excise in choosing between design variations or specifying performance attributes, they also provide the means to easily enable cross-temporal representations of how landscapes evolve. This highlights a relatively unique capability of parametric modelling, whereby a model that projects how a site changes over time can be readily constructed by re-organising established geometric relationships in terms of temporal change. As a result, aspects such as plant growth, programmatic intensity, or cut/fill manipulations can be examined not as a single intervention but as a staged and successional series of operations.

4 Affinities and Overlaps

At the same time there are many affinities to how computational methods are taught in both an architectural and landscape architectural context. These are worth highlighting here though to identify where these can overlap with standard landscape architectural curricula and how they can reinforce or re-frame concepts that are explored in other settings that might not seem like natural contexts for introducing computational design techniques. These include:

- *A relational approach to design development*. The dynamic nature of parametric models ties closely to existing landscape architectural discourse that emphases systems thinking as an approach to understanding site conditions through interrelationships and feedback cycles. Computational techniques allow for students to explore this approach in a rigor-

ous and explicit manner by not only understanding a system in abstract terms but to begin to quantify those interactions and employ them within design development (WALLIS et al. 2014).

- *A pluralistic approach to design tooling*. In the particular context of Grasshopper, there is a thriving software ecosystem of free plugins that allows students to look beyond the monolithic software programs common to design education and instead start 'looking for the right tool for the problem' (DAVIS & PETERS 2013). As per BUSON et al. 2017, given our discipline co-ordinates between a diverse range of actors, having flexibility in adopting new methods can help better understand related fields and encourage collaboration.
- *An emphasis on explicit geometric relationships*. While not all designs should exhibit formal complexity, the task of defining complex forms through parametric models is often a rigorous test of how well students understand geometry and tectonics; particularly when explored in conjunction with fabrication technologies that introduce physical constraints.

5 Conclusion

Teaching that heavily incorporates digital design methods and software workflows are often characterised as exercises in 'skilling' (FRICKER et al. 2013) or as more vocational aspects of instruction better confined to external resources such as online videos. While these external resources can be valuable for learning specific concrete tasks, they remove much of the spaced interaction and ad hoc collaboration that drive effective design outcomes (ZEHNER et al. 2010). They may provide helpful instruction in the mechanical tasks of executing software programs but then fail (particularly in a landscape architectural context) to detail and develop the distinct conceptual approaches that underpin computational design and that are key to understanding its value to the design process beyond a specific task.

Developing this higher-level understanding typically involves the difficult task of first giving students enough of an introduction to this new form of design that they can then locate its value within the discipline and the wider goals of the course (PAAR & REKITTKE 2012). Many of the pedagogical approaches discussed above can help aid and accelerate this process, while also providing a more critical angle on how many de-facto practices and resources for teaching computational design techniques can be ill-suited to landscape architectural purposes. This"[3] project: a Grasshopper plugin and set of online documentation that provides resources (similar to options such as Lynda[4] or Designalyze[5]) explicitly catering to landscape architectural uses and the approaches discussed here. While online resources don't provide the kinds of highly-contextual instruction discussed above, having resources that more explicitly focus

[3] The site and plugin are currently in a public beta phase with new content being rolled out over the course of 2018. Both are accessible at the http://groundhog.la URL.

[4] Lynda (http://www.lynda.com) is a professional skill-training video company with many courses catering to CAD and other design software.

[5] Designalyse (http://designalyze.com) offers a series of instructional videos focused on Rhinoceros and Grasshopper developed by two architects who work across academia and practice.

on a landscape context will likely serve as useful aids in developing instruction or for students' own self-directed learning.

That said any approach to teaching computational design should first reflect critically on any provided resource or tool and look to adapt both its use and framing in terms of a broader learning intent (FRICKER et al. 2013). Adapting emerging technologies and approaches in this manner can be difficult, as even with established examples of pedagogical and professional practice it is difficult to distinguish between 'soft' and 'hard' limitations that stem from de-facto practices and the tools themselves respectively. But doing so allows us to characterise tools such as computational methods as a way further *in* to our discipline instead of as a curiosity or instance of avant-gardism.

References

BUSÓN, I. L. et al. (2017), A Computational Approach to Methodologies of Landscape Design. In: De RYCKE, K. et al. (Eds.), Humanizing Digital Reality. Springer Singapore, Singapore, 657-670.

DAVIS, D. (2013), Modelled on Software. PhD thesis, RMIT University. http://www.danieldavis.com/thesis/.

DAVIS, D. & PETERS, B. (2013), Design Ecosystems: Customising the Architectural Design Environment with Software Plug-ins. Architectural Design, 83 (2), 124-131.

FRICKER, P., GIROT, C. & GOERG, M. (2013), How to Teach 'New Tools' in Landscape Architecture in the Digital Overload. In: Computation and Performance Proceedings of the 31st eCAADe Conference. Delft, The Netherlands: Delft University of Technology, 545-554.

RAXWORTHY, J. (2013), Novelty in the Entropic Landscape. PhD Thesis, The University of Queensland.

REKITTKE, J. & PAAR, P. (2012), Wheeling a Trojan Horse to Teach MLA Students Geoinformation Methods. In: Proceedings of the 13th Digital Landscape Architecture Conference. Bernburg and Dessau, Germany. Wichmann, Berlin/Offenbach, 32-40.

ROUDAVSKI, S. (2012), Selective Jamming: Digital Architectural Design in Foundation Courses. International Journal of Architectural Computing, 9 (4), 437-462.

WALLISS, J. et al. (2014), Pedagogical foundations. Journal of Landscape Architecture, 9 (3), 72-83.

ZEHNER, R. et al. (2010), Optimising Studio Outcomes. In: Connected 2010 − 2nd International Conference on Design Education. University of New South Wales, Sydney, Australia, 1-5.