Essential Education for Computational Design in Architecture

"A meaningful building of the digital age is not just any building that was designed and built using digital tools: it is one that could not have been either designed or built without them"

- Mario Carpo [1]

RAJAA ISSA
Robert McNeel & Associates
Seattle, WA
NewSchool of Architecture and Design
San Diego, CA

ABSTRACT

Computation methods have been introduced in architectural education out of impressive precedence, increased demand, and the need for diversity. While excellent programs have emerged at a few pioneering schools, digital integration is costly and uncertain, and can challenge established programs and bureaucracies. As a result, many schools stand unable or unwilling to incorporate the technology. This paper presents a rigorous approach to providing an essential education for computational design that addresses many of the challenges facing computational methods in education. It includes a case study for the implementation of this approach at the NewSchool of Architecture and Design.

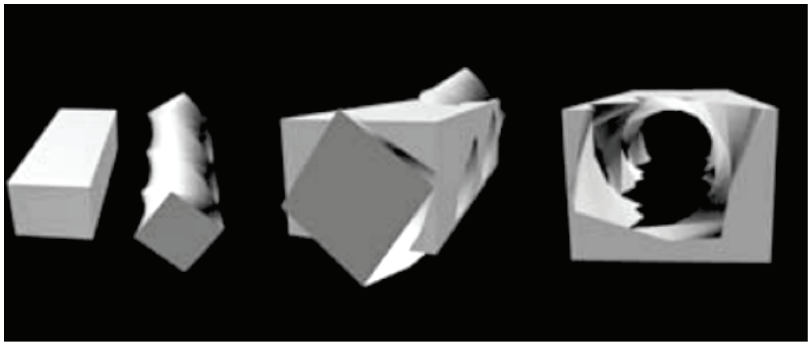
BACKGROUND

The utility of computers as an efficient way to produce architectural drawings was the basis behind early commercial computer-aided design (CAD) applications. Documentation continues to be the most dominant use of computers in architectural practice today. However, a different view that focuses on the potential of using digital processes as an integral part of design and construction started long before commercial CAD.

Early ideas included *Pattern Language* [2], *Shape Grammers* [3], and *Evolutionary Architecture* [4]. New architectural theories such as Blob Architecture were directly influenced by the capabilities of computer graphics [5]. Those ideas remained largely confined to research, and found very limited application in practice.

In education, computers were mostly introduced in courses centered around commercial CAD applications. These courses are typically isolated from the design studio and other classes. This approach meant that students were given the means to produce complex forms, without the knowledge of how to analyze these forms in the context of design and building [6]. The poor results were blamed on computers that allegedly encouraged complexity without meaning (figure 1). As a result, many educators distanced themselves from computers and discouraged their use in design. Many of these views linger among educators to this day.

While schools were busy ignoring computers, a small section of



1

architectural practices approached the new technology quite differently. They saw an opportunity to use computers to realize complex forms that had never been built before. These architects started to use specialized computer programs uncommon to architecture such as Catia and Rhinoceros to accurately model form. They teamed up with clients and invested massive amounts of effort and resources. Many players across the whole building industry were involved in their research and innovation. The results were breathtaking. Architecture witnessed the emergence of new fascinating forms that captured the imagination of the architectural community and the public alike (figure 2). More importantly, it revolutionized how computers were used in architecture. The interest in computational methods as an active part of building

Figure 1: Student designs directly influenced by 3D computer operations of Twist and Boolean. Image by Lynn [5].

2

architecture was resurrected.

This mounting investment in computational methods motivated the creation of new intuitive algorithmic tools that are accessible through the new generation of computer aided architectural design applications (CAAD). New programs such as Generative Components for Microstation and Digital Project for Catia were introduced, both relying on text-based programming. They found some success, but it was Grasshopper for Rhinoceros with its visual scripting environment that defined a turning point in the popularity of algorithmic methods amongst designers. The community devoured parametric methods with tremendous interest at an unprecedented scale. New architectural theories such as Parametricism started to gain momentum [7]. The newfound popularity of parametric design supported by the intuitive, accessible and open nature of Grasshopper started to attract an increasing number of professionals to build a variety of new functionality on top

Figure 2: Complex forms in built architecture. Images by (clockwise from top left): Wilth [12], Ardfern [13], Taxiarchos228 [14], Guichard [15].

of Grasshopper. The community involvement transformed Grasshopper from a tool into a rich environment supporting activities such as site studies, structural evaluation, environmental analysis, prototyping, robotics, evolutionary methods of optimization, and interfacing with people and devices [8]. Parametrics started to offer a viable integrated solution to support every aspect of the design process. What made this new environment even more valuable was the rapid development of digital fabrication methods that tied the digital world to physical reality. Computational methods became embedded in a wide range of design, engineering, and building activities in contemporary architecture [9].

THE CHALLENGES OF COMPUTATIONAL METHODS IN ARCHITECTURAL EDUCATION

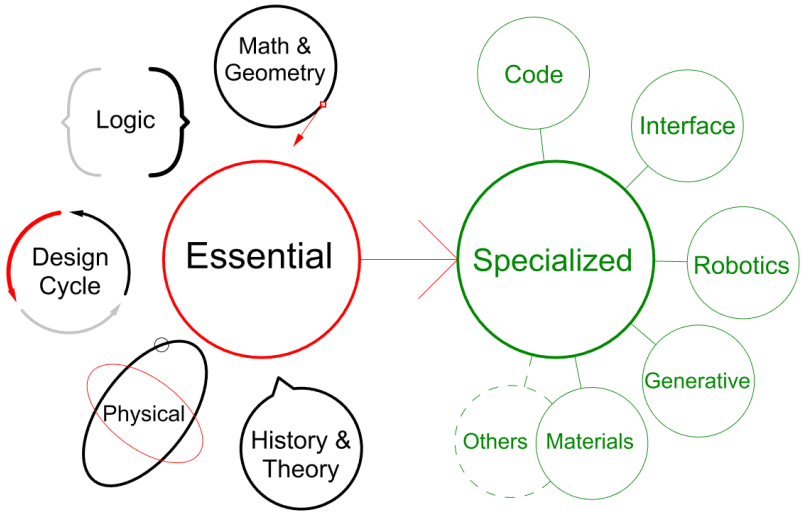
While the prospect of computational design methods is fascinating and its utility has become evident in practice, schools are still unsure about how to support them in their curricula. The challenges can be summarized as the following:

1. Many educators are not convinced of the value of computational methods in architectural education. Hence, they are unable or unwilling to embrace computational methods as part of their teaching.
2. Algorithmic thinking and computational methods are not trivial to teach and require specialized resources and careful planning.
3. Software and fabrication equipment is changing rapidly, which poses a financial and planning challenge.
4. While there are many courses, studios and workshops that teach computation in many architectural schools, most are not documented and their effectiveness is not evaluated. There are no recommended guidelines or standards that can help institutions decide ‘what’ and ‘how’ to include about computational methods in their programs.

ESSENTIAL EDUCATION FOR COMPUTATIONAL DESIGN

This paper proposes an essential education for computational design (EECD); one that attempts to address these challenges and provide meaningful education at the same time. It has been based on an

extensive research study by the author through analysis and construction of computational tools and supporting professionals in the building industry. EECD identifies five core components that are key for any comprehensive understanding of computational methods in architecture. There are many other areas of specialization within computation that have great application in architecture, but it would be unwise to introduce them without providing the essential core subjects shown in figure 3.



3

With EECD, students are expected to develop:

- A strong background in vector mathematics and geometry
- Skills in algorithmic thinking
- An understanding about computation as an integral part of all stages of the design process
- Hands-on experience with different fabrication methods and the ability to transition back and forth between the digital to the physical
- Critical understanding of the theory of computational methods

There are a few considerations that should be mentioned when designing an implementation of EECD, depending on the fluency of faculty members in computation and available resources. For example, schools new to computation may want to introduce EECD as a self-contained sequence of classes so that it does not disrupt existing curriculum, concurrently educating the faculty and limiting the amount of up-front commitment. Another important consideration is to try to develop teaching material that is relevant, engaging and adaptable to the fast

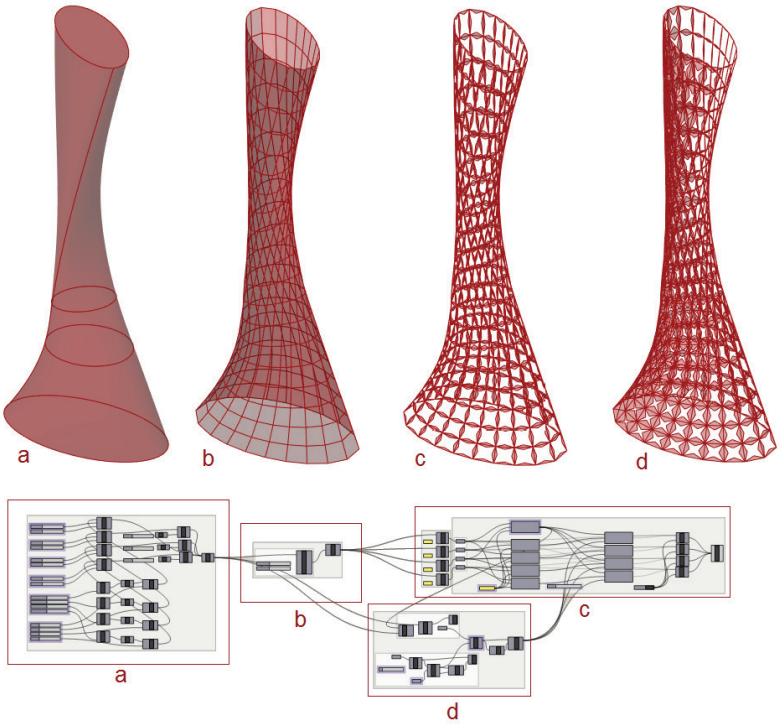
Figure 3: The components of the essential education in computational design (EECD). Image by author.

pace of the digital technology.

INSTRUCTIONAL PHILOSOPHY

It is important to learn 'how' to use computational methods, but it is even more important to actively engage in questioning 'why' and 'when' they should be used. Developing the ability to recognize the context in which computational methods add value is key to successful utilization of any design method. Involving students in critical discussions about precedent helps them develop a better understanding of meaning and context.

Algorithmic thinking skills develop over an extended period of time. Also, the concepts of mathematics and geometry may be challenging, especially to design students. Therefore, it is best to develop knowledge in algorithms, mathematics and geometry early in the program before involving any creative design activity. Designs should start as simple, and gradually increase in complexity over time (figure 4).

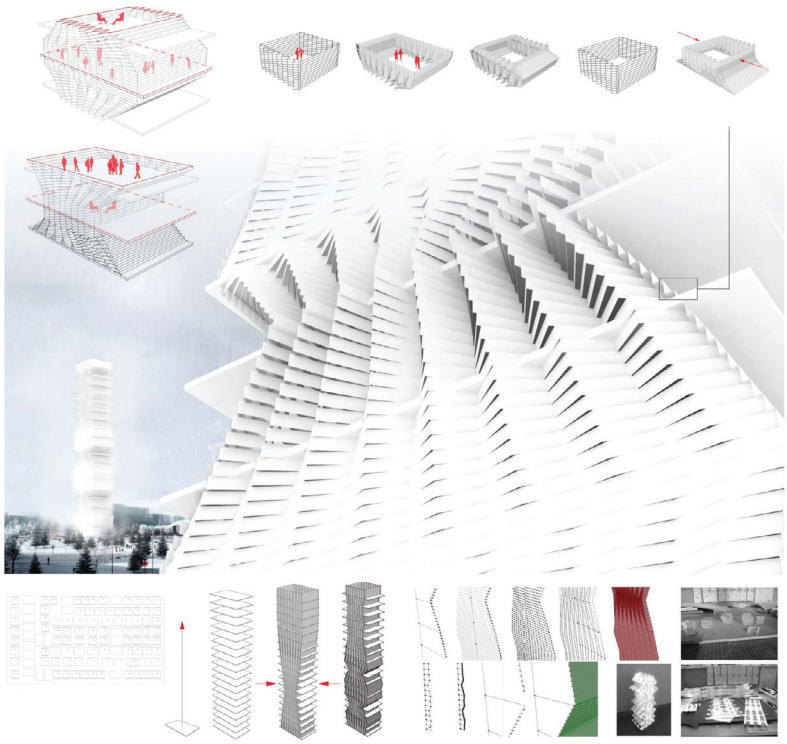


4

Figure 4: Progress of algorithmic complexity in the 'Adaptive Skin' project of NewSchool student Julio Medina. Image by author.

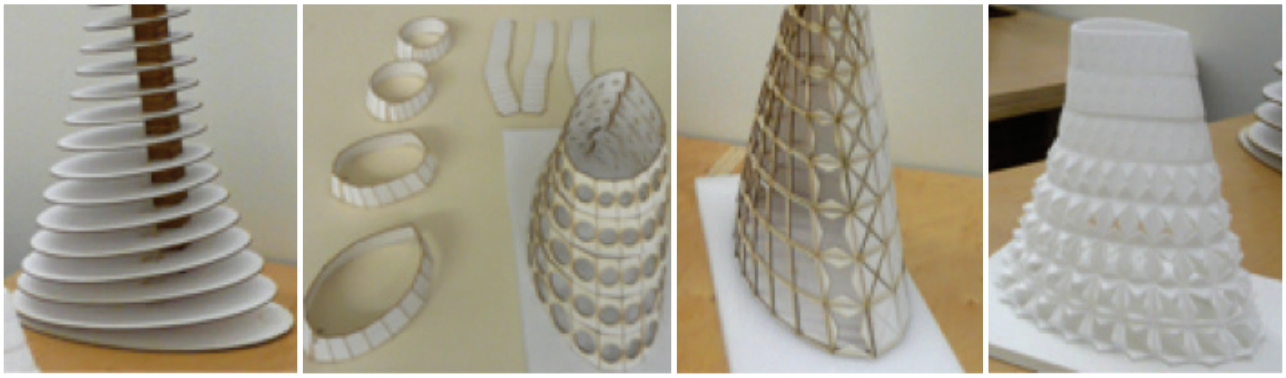
The material needs to be relevant and engaging in order to appeal to creative and visual thinkers. The program should utilize intuitive computational tools with instant visual feedback when possible to help facilitate quick cycles of synthesis and reflection [10]. It is also important

to use computational methods to solve problems that are harder to resolve when using other design methods (figure 5).



5

Using digital tools in conjunction with other design media helps students improve their representational and communication skills. When appropriate, the use of sketching and physical modeling should be



6

encouraged alongside computation. Digital fabrication and the physical realization of digital models should be at the center of the program. The transition between the digital and the physical is a very powerful tool to help students appreciate both the implications of decisions made with the digital tools,WW and the feasibility to realize them in physical form

Figure 5: The use of computational method to investigate responsive facades, student work. Image by Christian Garcia.

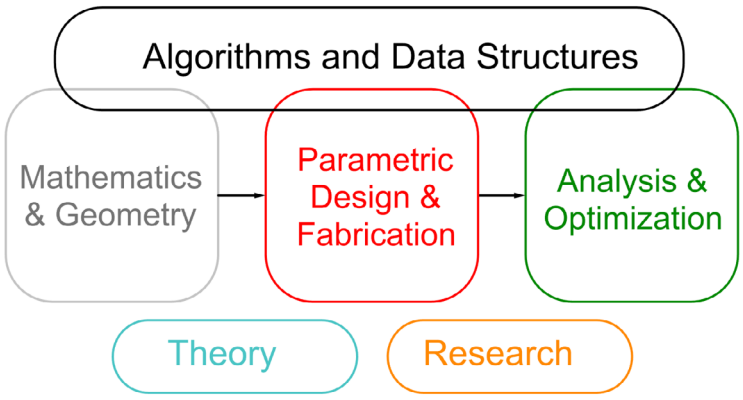
Figure 6: Different digital fabrication methods explore different aspects of design. Student work: Julio Medina. Images by author.

(figure 6).

The EECD program should include study of the theory of computation. This theory component includes studying digital processes employed successfully in real projects and discusses various limitations and potentials. Students should be encouraged to develop critical understanding of the field and research new ways to enhance and develop current practices and technologies.

CASE STUDY: EECD AT NEWSCHOOL OF ARCHITECTURE AND DESIGN

NewSchool of Architecture and Design in San Diego, California was established in 1980. The school has a strong focus on practitioners pursuing education in architecture and design. While faculty members are accomplished educators and practitioners, they generally have little expertise in computational methods. The first computer lab opened in the early 1990's, and started with AutoCAD in terms of software. Revit and FormZ followed in 2007, with Rhinoceros 3D and Grasshopper introduced in 2008. The materials lab of the school is equipped with laser cutters, computerized numerical control (CNC) routing machines, wood working tools, and an advanced 3D printer. In the summer of 2013, the author joined the school with the vision to introduce comprehensive education in computational design without disrupting the existing curriculum. She started a new series of classes in 'computational design methodology'. These classes evolved into a sequence of three three-credit elective classes that stretched across one academic year.



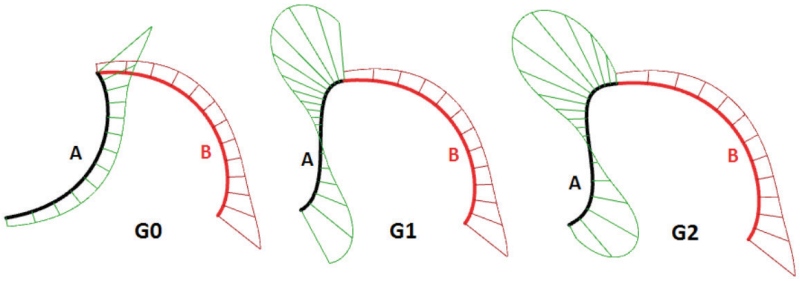
7

Figure 7: Essential Education in Computational Design (EECD) course structure as implemented at NewSchool of Architecture and Design. Image by author.

Classes met four hours a week for 33 weeks. The sequence was open to all students; from second year undergraduate to final year graduate, with

no prerequisites to enter the first course. Figure 7 illustrates how the program is structured.

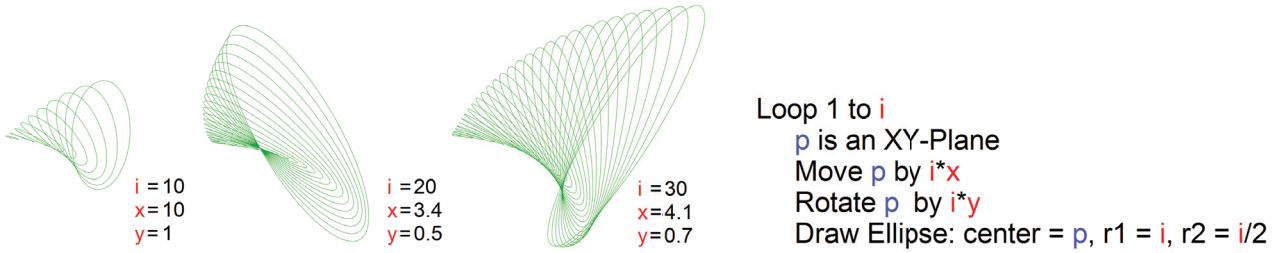
The following is a description of the subjects taught in the program.



8

Mathematics and Geometry

While some students came to the program with good analytical skills and some facility with mathematics, most were very poor in this area. The foundation unit was chosen to be the first in the sequence. Essential vector mathematics, transformations and NURBS geometry concepts were introduced in a design-related context. A visual approach to teaching mathematics through Grasshopper was adopted to help with the visualization of abstract concepts [11].



9

Algorithms and Data Structures

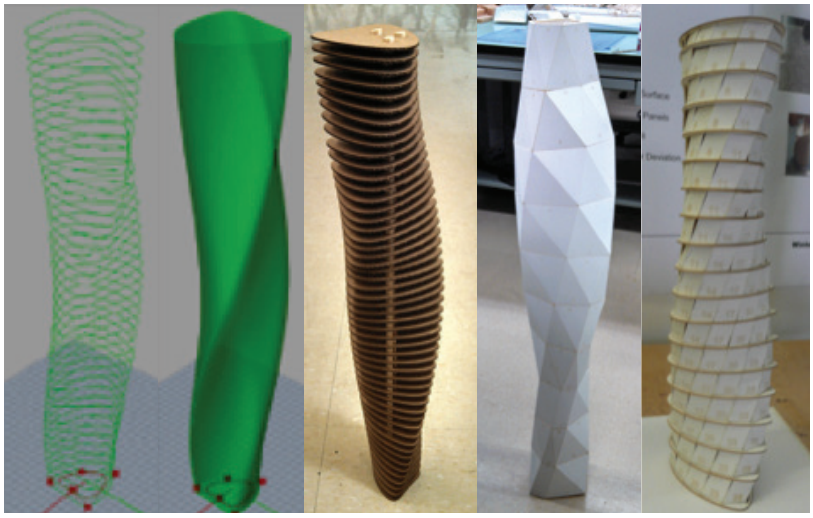
As mentioned before, developing skills in logical thinking and algorithmic problem solving takes significant time. Therefore, this subject was included in all classes within the sequence. Students started with simple problems that gradually increased in complexity. Weekly problems were given to analyze and solve algorithmically under time constraints, aiming to developed fluency. They also studied various ways to store and manipulate digital data.

Parametric Design and Digital Fabrication

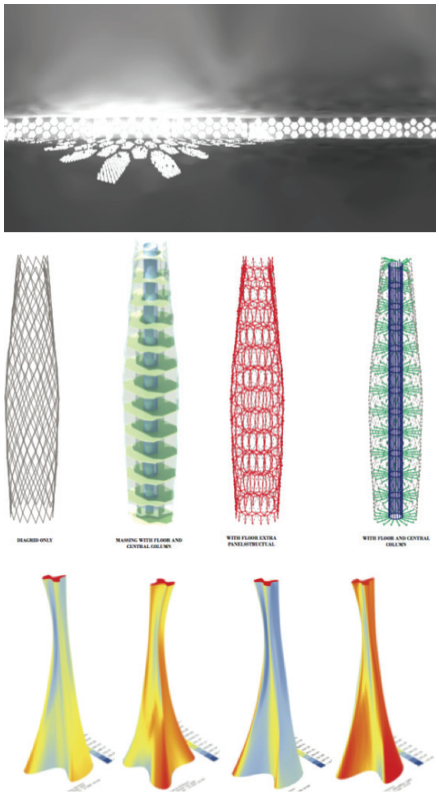
During the second unit, the students designed an adaptive skin system

Figure 8: Graphic display of continuity for various NURBS (Non-uniform rational basis spline) curves. Image by author.

Figure 9: Design of algorithms involves analyzing desired output into well-defined steps. Image by author.



10



11

Figure 10: Sequence: abstract digital form to a fully rationalized physical model. Student work: Anthony Rodriguez. Image by author.

Figure 11: Glare, structural, and thermal analysis followed by optimization using the Grasshopper environment. Images of student work (top to bottom): Ryan Stangl, Yangyi Situ, Heiarii Li Cheng.

for a skyscraper using parametric methods. The skins were required to respond to at least one constraint that was environmental, functional or aesthetic. Students completed three cycles of parametric design and fabrication: building mass, paneling geometry, and ultimately a fully responsive façade system. The 'adaptive skin' project not only helped students understand the process of rationalizing design forms, it also enabled both experimentation with different fabrication techniques and the development of abilities for building fully parametric design solutions.

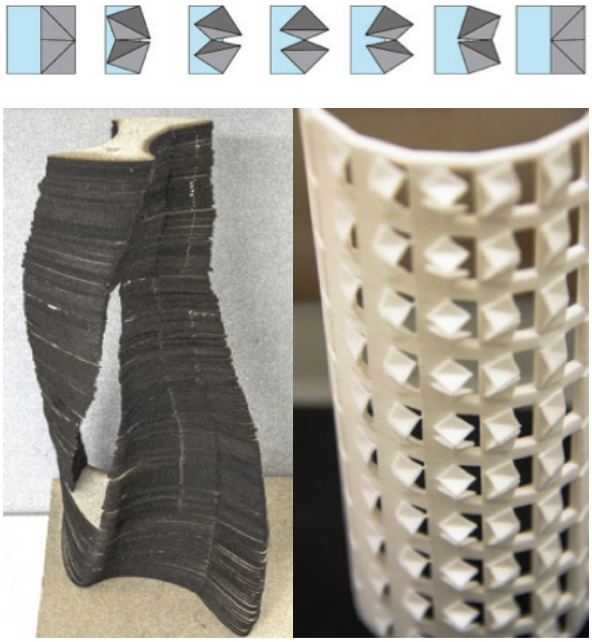
Digital Analysis and Optimization

In the third unit, the performance of the adaptive facades developed during the second unit was evaluated and further developed using a variety of digital analysis and optimization tools. The students were deliberately exposed to a variety of tools to help them develop the ability to quickly evaluate, learn, and utilize digital tools. They were encouraged to critically analyze their solutions and produce performative designs. By the end of this final unit, they started to appreciate the full cycle of design using computational methods.

Theory and Research

Simultaneously, students were required to research one of the prevailing questions in the field of computational design, tackling issues such as representation, communication, digital processes and the role of the architect in the digital age. They shared research through a series of seminars. This theory and research component exposed students to

current debates in the field, and gave them a better understanding of the technology, processes, and issues involved.



12

The influence of EECD at NewSchool

Many of the students who participated in the program incorporated their newfound skills of algorithmic thinking into their design studio projects and research. Other students and faculty members were exposed to the new methods, and started to gradually appreciate the utility and impact of computational methods in design. After only two years, the course enrollments are at full capacity.

CONCLUSIONS

Essential education in computational design (EECD) identifies five core subjects that need to be included in any program. It aims to provide a comprehensive education with strong focus on the foundational knowledge of geometry and algorithmic thinking. EECD helps students reach a level of fluency with digital design tools and fabrication methods that they can utilize appropriately at any stage of the design process.

Figure 12: An abstract building mass ideas (left image) evolves into fully rationalized solutions (right image). Images of student work: Christopher Voltl.

The implementation at NewSchool has shown very promising results on both an individual level, as well as influencing the school's culture in general. Most students continued their pursuit of the field through their design studio projects and research. Their work provided a precedent for other students and faculty members and introduced them to the value of computational methods in architecture. The school is also seeking to establish a new certification program in computational design based on the EECD.

ACKNOWLEDGEMENTS

I would like to acknowledge the great support of Kurt Hunker, the graduate architecture chair at NewSchool, who trusted my vision and gave me the opportunity to develop the program at NewSchool. I would also like to thank our materials lab manager, Erik Luhtala, whose expertise in fabrication has been an invaluable resource for the program. I am very grateful to my colleague, Michael Riggin, who did not spare any efforts to support and push the program forward. Last, but not least, I am indebted to my very bright and faithful students. Their ideas and enthusiasm inspire me everyday. Without all these people, my thoughts and aspirations would have never materialized.

REFERENCES / IMAGE CREDITS

1. Carpo, Mario. *The Digital Turn in Architecture 1992-2012*. Chichester: Wiley, 2013. Print.
2. Alexander, Christopher, Sara Ishikawa, and Murray Silverstein. *A Pattern Language: Towns, Buildings, Construction*. New York: Oxford UP, 1977. Print.
3. Stiny, George. "Introduction to Shape and Shape Grammars." *Environment and Planning B: Planning and Design Environ. Plann. B* 7.3 (1980): 343-51. Print.
4. Frazer, John. *An Evolutionary Architecture*. London: Architectural Association, 1995. Print.
5. Lynn, Greg. *Folds, Bodies & Blobs: Collected Essays.* Bruxelles: La Lettre Volée, 1998. Print.
6. Issa, Rajaa, and Jules Moloney. "The Potential of Computer Modeling Software to Support a Consideration of Building Materials in Architectural Design Education." *Connecting the Real and the Virtual - design e-ducation: 20th eCAADe Conference Proceedings* (2002): 440-447. Print.
7. Schumacher, Patrik. "Parametricism: A New Global Style for Architecture and Urban Design." *Architectural Design* 79.4 (2009): 14-23. Print.
8. Peters, Brady. "Computation Works: The Building of Algorithmic Thought." *Architectural Design* 83.2 (2013). Print.
9. Marble, Scott. *Digital Workflows in Architecture: Designing Design -- Designing Assembly -- Designing Industry*. Basel: Birkhäuser, 2012. Print.
10. Schon, Donald. *The Reflective Practitioner: How Professionals Think in Action*. San Francisco: Jossey Bass, 1987. Print.
11. Issa, Rajaa. *The Essential Mathematics for Computational Design, Third Edition*. Seattle: Robert McNeel & Associates, 2013. Print.
12. Wilth. (2013). Heydar Aliyev Cultural Center Baku [Online image]. Retrieved fromhttps://flickr.com/photos/wilthnet/8999193744
13. Ardfern. (2010). Guggenheim Museum, Bilbao, July 2010 [Online image]. Retrieved from https://upload.wikimedia.org/wikipedia/commons/e/e8/Guggenheim_Museum%2C_Bilbao%2C_July_2010_%2811%29.JPG
14. Taxiarchos228. (2011). Graz-Kunsthaus1 [Online image]. Retrieved from https://upload.wikimedia.org/wikipedia/commons/4/40/Graz_-_Kunsthaus1.jpg
15. Guichard, A. (2010). 30 St Mary Axe from Leadenhall Street [Online image]. Retrieved from https://upload.wikimedia.org/wikipedia/commons/4/4c/30_St_Mary_Axe_from_Leadenhall_Street.jpg