Building Technology Educator's Society

Volume 2019 Article 46

6-2019

Comprehensive BIM Integration for Architectural Education Using Computational Design Visual Programming Environments

Roger Schroeder
SUNY Alfred State College of Technology, schroera@alfredstate.edu

William Dean
SUNY Alfred State College, deanwc@alfredstate.edu

Comprehensive BIM Integration for Architectural Education Using Computational Design Visual Programming Environments

Roger Schroeder and William Dean
SUNY Alfred State College of Technology

Abstract

It is well established that Building Information Modeling (BIM) has had a significant impact on the way Architects and their firms view and approach design projects, and, to a larger extent, how this evolution is influencing and advancing workflow in the Architecture, Engineering and Construction (AEC) industry as a whole. What is not as clear is how the academy is responding to these changes, and the challenges associated with integrating the complexities of a fast-emerging technology into the architectural curriculum. Recent developments in visual programming environments that function as integrated components of the BIM software point to new ways to interact with the emerging and evolving BIM paradigm now common to most educators. Presenting this material comprehensively across the program curriculum is difficult since the expertise to develop course-specific computational content may not yet exist.

This idea of exploring computational visual programming across the curriculum can be envisioned as a tool for reimagining aspects of the process of design education and learning methodologies. Core evolving features of using this technology are the ability to include new collaborators, create real-time collaboration across web interfaces, provide design participants interactive design tools, compress design development cycles, and create more efficient designs that enhance beauty and functionality.

This examination explores a methodology of applying computational frameworks into coursework throughout the curriculum. It is proposed that building learning components in a modular form will allow both educators and students improved accessibility to the concepts. Underpinning this modular approach is the availability of using node based parametric modeling tools to extend BIM software, notably Dynamo for Revit, and the nodes being developed for it by the design community.

Course materials are envisioned around a pre-constructed BIM model (or one developed for the course) and will use guide documents and step-by-step video instructional support to simplify inclusion into the course. Conceived to be one class period in length with the exercise completed outside of class, a few examples will be explored and developed for this initial survey. Instructors would use the material directly or as boilerplate for customizing it to varying instructional situations. Specifically this learning framework uses visual programming experiences applied to parametric elements (glazing, wall type, room shape etc.) to achieve insight into specific course concepts. For example, in a course section on life safety codes, egress distances can be explored using parametric tools that calculate distance

dynamically as rooms and exit doors are moved during design development of a mockup model. Students will work from a model and then add code and components allowing for conformance reporting. In another example for energy analysis, a parametric louver system might be adjusted in real time in simulations to optimize orientation.

Analysis will also include a deeper exploration of the BIM component, and exploiting it using advanced expression of embedded data using visual code and computational concepts. In addition to looking inward at BIM components, possibly more impactful will be looking outward to model sharing across the browser using WebGL (web graphic software) and code for web and other new functionality now part of standard browser features. Both learning experiences prepare us for the future of integrating AI and machine learning into design tasks.

These individual experience “modules” embedded throughout the curriculum have the ability to connect information to design, allow interactivity and facilitate collaboration with potentially new outcomes. This might create new ways of sustaining creativity and interest in this evolving aspect of design education that might otherwise appear as a “dry” subject.

The Relevance of Visual Programming Throughout the Curriculum

Just as the BIM paradigm has become fundamental in architectural education and shifted thinking on many aspects of design education, it is likely that computational design and visual programming will have similar profession-wide effects.

It is not surprising that many AEC conferences feature presentations by leading firms illustrating the use of visual programming and computational design as a strategic component of the design process. These presentations, by the nature of their sophistication and complexity, are also uniquely interesting. Typically, they describe the coordination between a team of experienced users as a significant element of successful implementation. It is natural with this level of inherent complexity that computational design and visual programming are treated as specialized technologies and presented in stand-alone advanced courses.

To address the idea that this is a complex “high level” concept reserved for advanced courses, the initial program development focuses on learning experience modules that have the potential for being most impactful and accessibility. A design or analysis task that would be difficult to accomplish any other way or with great difficulty has the potential demonstrate usefulness and encourage engagement especially if it can be demonstrated to be implemented with relatively ease.

The best known computational design visual programming environment (CDVPE) for architectural use is Grasshopper, first released in 2007 as a tool for McNeel Associate’s Rhino modeling software. Rhino was quickly adopted by students and architects interested in developing complex but very accurate NURBS defined geometries and Grasshopper allowed for detailed component development on these surfaces. This positioned both the software and the resulting architectural forms at the advanced level in terms of perception. Knowing the history of the development of these tools helps us understand how it is positioned and perceived today.

It is useful to start with an examination about the breadth of the utility of Grasshopper specifically because of the expansive plugins and application this software is used for. A list of plugins to accomplish tasks (some are just in early stages of development while others are mature products) include:

- Daylight and Energy Simulation (Ladybug Tools)
- Structural Analysis (Karamba 3D)
- Acoustic Analysis (Pachyderm)
- Behavioral Space Planning (Space Syntax)
- Evolutionary Problem Solving (Galapagos)
- Form Finding (Kangaroo)

Since the tools that augment the visual programming software extend its utility to many aspects of architectural design and learning, it is an opportunity to look for ways integrate them throughout established courses.

Already, computational design techniques are being applied in industry to achieve objectives that are also taught in architectural course, but without the advantages of these techniques. And while CDVPE's are not currently fundamental to the process it may be informative, creating an important foundation for the evolution of these techniques and better preparing students for the technological changes happening in practice.

Still, having applicable technologies (task specific nodes in the case of CDVPE) is only one element of making it a useful and meaningful educationally. For lessons to work in a proposed modular fashion, they must be both manageable in scope (fit in the time allotted) and be able to expand on the concept in a way that advances thinking. Building these modules successfully will require effort, coordination and desire by all stakeholders. Fortunately we have many examples of computational tools that are creating academic interest because of their usefulness.

For example, for environmental design courses (i.e., Environmental Controls 1&2) "Some of the world's leading architectural practices have developed teams with a special focus on sustainability have implemented the use of building simulation, parametric design techniques and customized computational tools."[1] (Emanuele Naboni 2013).

In courses that focus on building codes and compliance (i.e., Municipal Codes and Regulations) research is being done on automated techniques using visual programming to check BIM models for code compliance.[2] (Preidela and Borrmanna 2015)

Space planning components of studio work could anticipate exploring the mapping of space utilization in the BIM model by using data sets of occupant behavior integrated into a visual representation with Dynamo.[3] (McGinley and Fong 2015)

Delivering Visual Programming Learning Objectives Throughout the Curriculum With Insertions Into Existing Courses

We are already familiar with the fragmented (but improving) application of BIM being applied throughout design education curriculum in the last decade. The BIM model can provide the framework on which to build visual computational concepts. The idea of bridging curricula is not new. "For AEC education to set the pace for industry, the siloing of curricula must be broken down through integration of the disciplines, similarly to what is seen in the industry."[4] (Becerik-Gerber, et al. 2011)

Making curriculum space for the inclusion of new ideas and technologies presents unique challenges. Adding courses seems a cumbersome and fragmented way to introduce ideas that are basic and comprehensive. Our work with developing modular learning components for

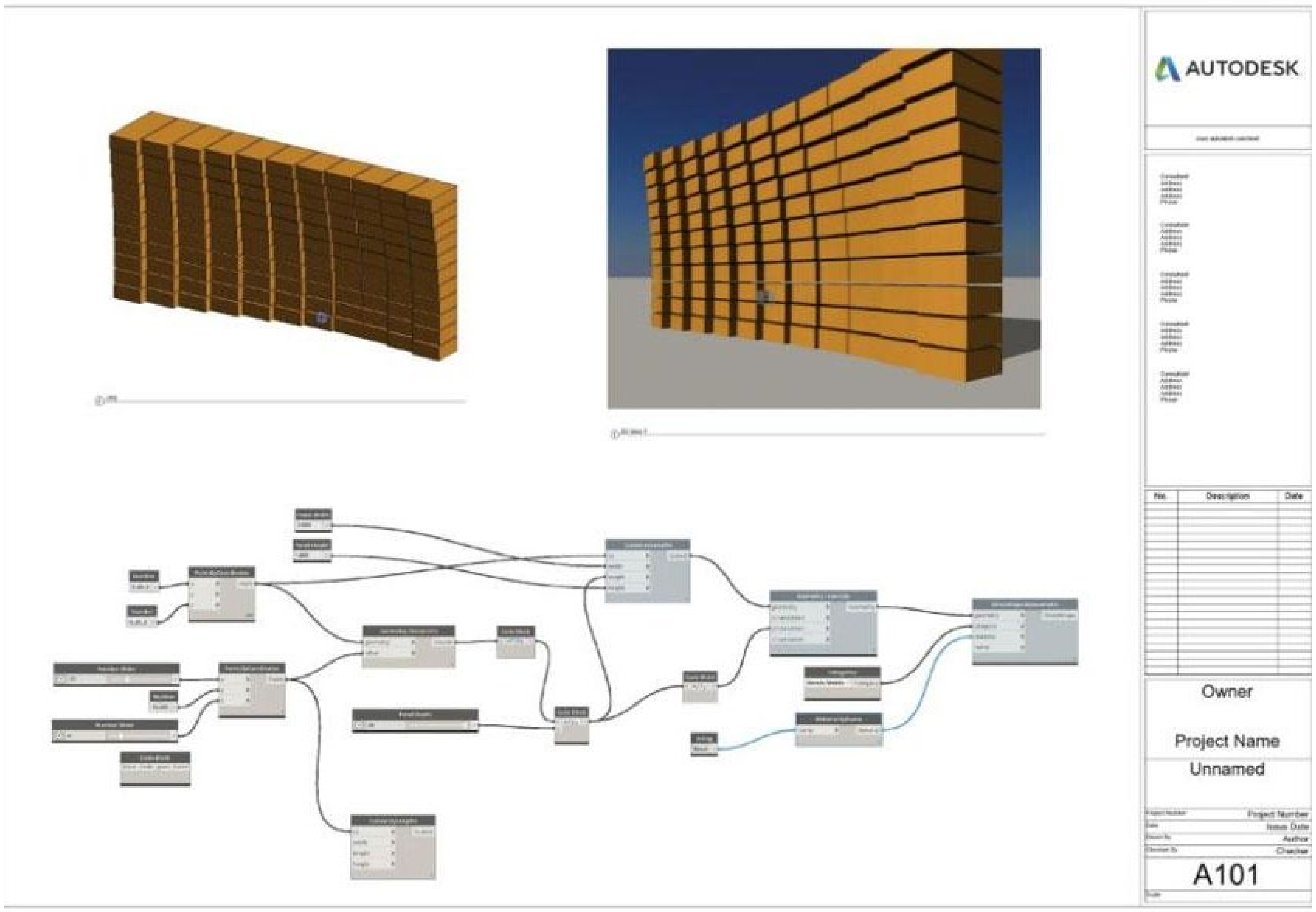

Fig. 1 Attractor Point to Control Panel Depth.

both BIM and Visual programming course work at SUNY Alfred State indicates a potential methodology.

CDVPE Concept Accessibility

Before considering the introduction of computational design concepts to a pre-existing curriculum at the undergraduate level, it is important to gain insight on how early in the sequence students can grasp the concepts well enough to create meaningful learning experiences.

To help us answer this question, in the spring of 2017 faculty introduced a computational design exercise using Dynamo into the Computer Visualization course, an introduction to BIM. This first-year, second-semester course is the first experience students have with BIM software. This project is structured around creating an array of “facade” panels and varying panel depth computationally according to the distance from the panel to a point in space (attractor point). The position of the attractor point is controlled in 2D space using adjustable slider inputs (Fig.1). Students performed as well with this task as other modeling tasks assigned in the course with 72 out of 76 completing the task. This module is now a permanent component of the syllabus.

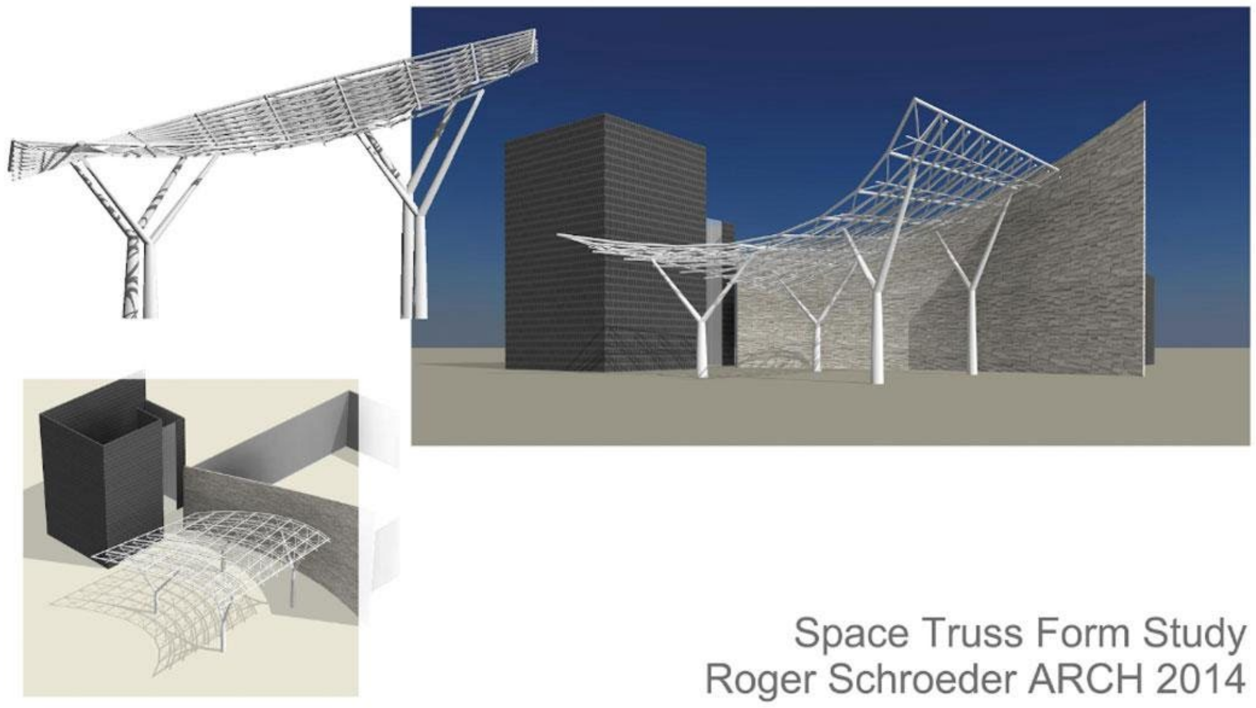

Fig. 2 First Year Space Frame Truss Project for ARCH 2014 Computer Visualization.

The fifth-year Advanced Structural Concepts course contained many CDVPE components in project assignments, and a survey of students (response 13 out of 24) at the end of the semester indicated that students felt that CDVPE material could be introduced by at least the third-year with 60% indicating an introduction even earlier in the second-year.

Software accessibility has also improved with the availability of CDVPE software to students and faculty. Visual programming environments were until recently more complicated to access and had the potential for added student expense. In the case of using Rhino and Grasshopper there was a cost and the need to learn a new software. Within the last few years Autodesk Revit has simplified the task with the inclusion of their CDVPE in the taskbar. This allows quick access by anyone using Revit. Outside developers that have produced nodes for Grasshopper are now making similar if not identical ones for Dynamo, likely in part to the significance of being on the Revit platform.

A Basis for Considering the Integration of Computational Design and Visual Programming into the Architectural Curriculum

The intention of an across-the-program insertion of visual programming, computational design and data driven design use is not to “train” specific solutions, but to create the idea of these tools becoming “natural” in use and application, not unlike sketching is used to inform the creative process.

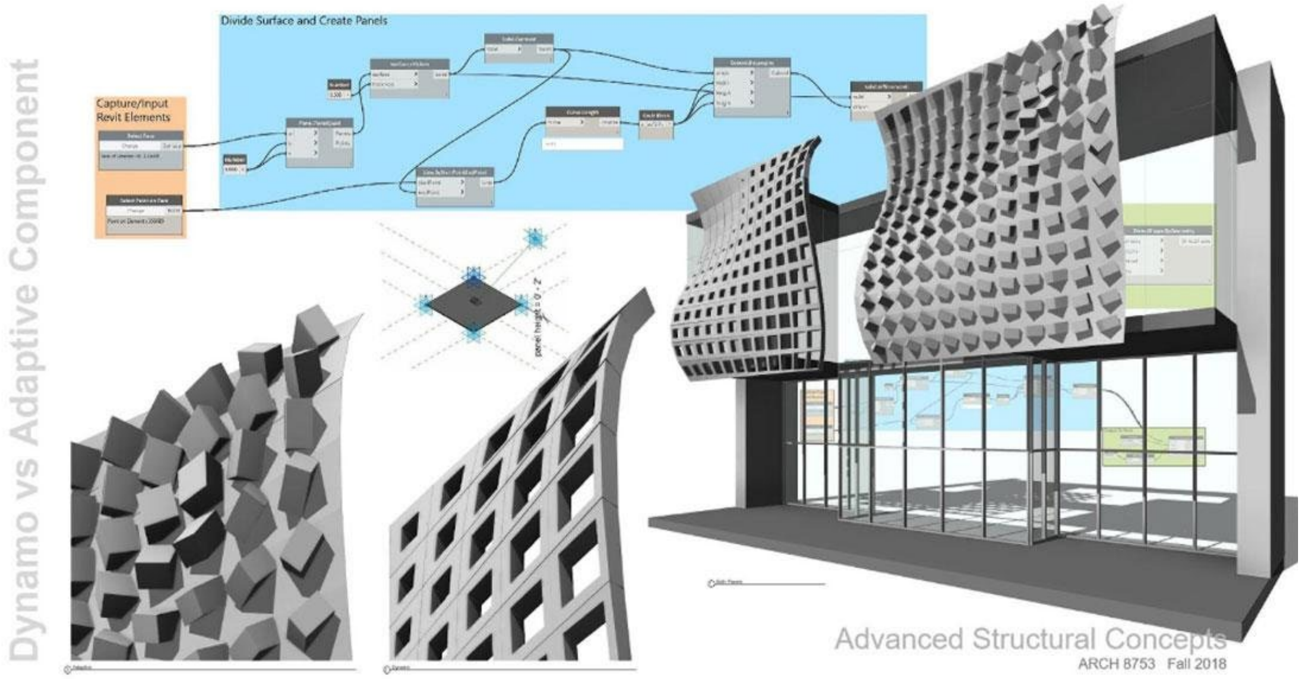
Dynamo vs Adaptive Component
Divide Surface and Create Panels
Capture/Input Revit Elements
Advanced Structural Concepts
ARCH 8753 Fall 2018

Fig. 3 Project to Contrast Adaptive Components to Dynamo Elements.

Just as BIM is now a commonly accepted component of architectural education programs, visual programming and computational design might then be considered the second-generation evolution of this platform. More concise CDVPE may be considered an augmentation to BIM process techniques.

Fragmentation of technologies resulting from the need to fit into course structure may be an impediment to advancing integrated learning of CDVPE. Just as with early BIM introduction, “Previous studies suggested that offering stand-alone BIM courses without any follow-ups in other courses do not support student long-term learning because students rarely find an opportunity to reuse BIM skills in different courses”[5] (Hu, M. 2018). The intentional injection of CDVPE modules into coursework is intended to address this concern.

Visual Programming vs Parametric BIM Components

The earliest advantages of visual programming were the ability to parametrize, subdivide and apply panels and other components to complex surfaces. Many BIM modelers today address this with features incorporating similar functionality. Pattern based surface division with adaptive component instances is a popular way to create complex geometry in Revit (Fig.2).

It might be argued that many of these types of higher-level tools obviate the need for visual programming knowledge.

While the evolving feature set of BIM software functionality can be viewed as a natural response to improving products in response to customers/industry demands, many BIM features are first developed in a CDVPE. Both the academy and profession are challenged to consider whether to wait for the new feature to be added to software or view CDVPE as an opportunity to innovate and use it as a driver for innovation in both toolset creation and design innovation.

Revit also anticipates this use of visual programming as an "innovation tool" with the development of a Dynamo Script Player. This plugin allows anyone the ability to create scripts that help automate common tasks, for example, calculating room occupancy capacities or automating rebar placement in concrete. This use of visual programming to develop workflow task automation makes learning it a practical as well as creative tool.

Still, there will exist a question about whether CDVPE will lose importance with increasingly sophisticated BIM functionality. To address this question, a project module has been created that solves a parametric problem using both adaptive BIM components and the CDVPE (Dynamo). This allows students to see both methods and the advantages and disadvantages of the alternate techniques (Fig.3).

Inserting Complex Ideas and Techniques in Introductory Courses as Modules

An advantage of exploring ideas in a digital software environment is the ability to easily describe ideas and process across the computer screen. The faculty took advantage of the fact that most concepts can be explained and demonstrated in the BIM screen interface in development of the Computer Visualization course (ARCH 2014). In the spring of 2017 video instructions for each project were created to enhance instruction. Students expressed almost unanimous preference to watching video instruction as an alternative to direct instruction. Reasons stated by students included the ability to individually pace the activity and review problem elements of the process (i.e., rewind the video). This experience is reinforced by a Harris Poll sponsored by Pearson "Beyond Millennials: The Next Generation of Learners" with 59% reporting YouTube as the #1 preferred learning method with 47% spending 3 or more hours on YouTube[6] (Harris Poll 2018).

The general positive response to this type of instruction set the stage for development of visual programming projects/experiences in the Advanced Structural Concepts course (ARCH 8753). In that course students were able to execute complex visual programming tasks without step-by-step classroom instruction. Using the knowledge gained from that experience, the authors are encouraged to attempt to create trial stand-alone visual programming modules that can be made available as components of existing courses. The intention of the module is that instructors need only the conceptual knowledge of the learning objective and no expert (just a working) knowledge of the technical visual programming process. This knowledge can be gained by pre-working the module.

Surveying Student Experience for Understanding

At the conclusion of the Advanced Structural Concepts course, the faculty surveyed students specifically about the visual programming component of the curriculum. This first survey is preparation for developing the ongoing survey instrument for gathering data.

The response rate for the survey was 13 out of 24 students. The survey was presented as a link to a Google form after course grading was completed to remove bias over grading and evaluation concerns. Students were encouraged with a follow up email to respond. In the future the bias concern may be outweighed by the need to improve the response rate. Surveys may be a course

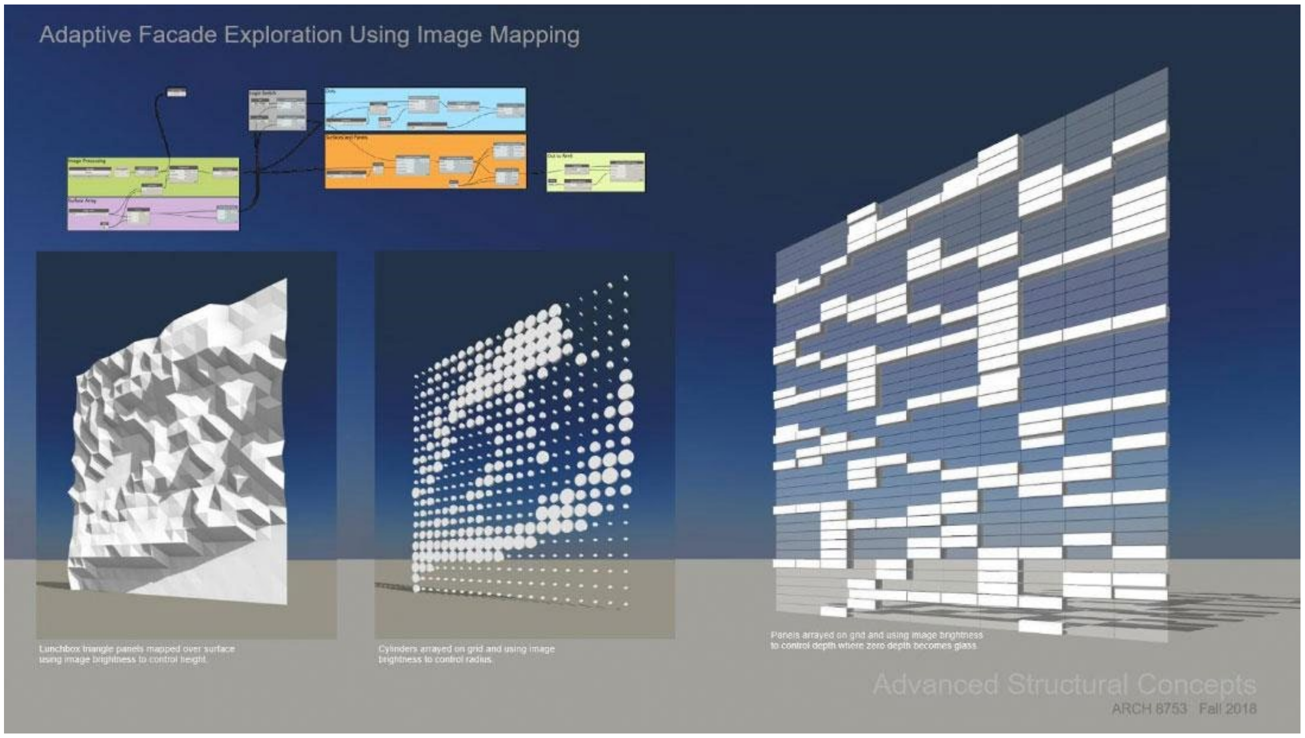
Adaptive Facade Exploration Using Image Mapping
Lunchbox triangle panels mapped over surface using image brightness to control height.
Cylinders arrayed on grid and using image brightness to control radius.
Panels arrayed on grid and using image brightness to control depth where zero depth becomes glass.
Advanced Structural Concepts
ARCH 8753 Fall 2018

Fig. 4 Adaptive Facade Using Image Mapping.

requirement next semester with users remaining anonymous ensuring a larger survey cohort.

Noteworthy in this first survey was that 76% responded that visual programming will allow them to be more creative designers. The consensus on the difficulty of learning visual programming when compared to other architectural skills was reported to be the same or just slightly higher than other skills.

Important going forward is the development of methods for validating any findings and conclusions with survey and testing instruments.

Formulating Visual Programming Modules

Inserting visual programming components into current course curriculum begins by looking at the existing courses, pre-existing visual programming conceptual projects then finding reasonable first fits. Visual programming content development for the Advanced Structural Concepts course provided good initial candidates for project modules.

Module: Relating Random Data to Form (Facade)

An introductory module was developed to relate data to an architectural form that involves using a photographic image to alter one characteristic of a facade. Specifically in this exercise, students link an image to an array and use that array as the basis for a facade surface. Data is extracted from the image at array points and numerical values are assigned according to image density at that location. The data is adjusted to a useful range and then used to drive facade component parameters to create variations in surface topography, opening diameter, and

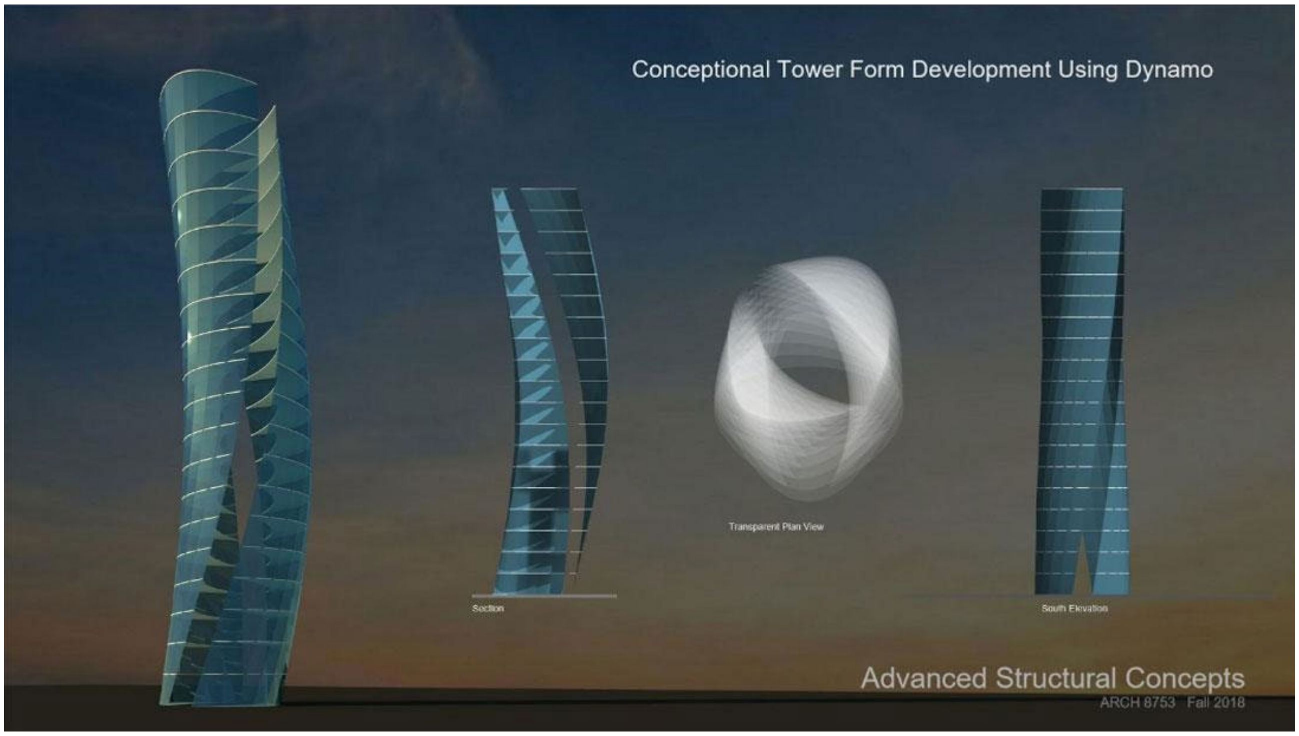
Conceptional Tower Form Development Using Dynamo
Transparent Plan View
Section
South Elevation
Advanced Structural Concepts
ARCH 8753 Fall 2018

Fig. 5 Conceptual Tower Form Using Dynamo.

panel depth in a series of digital mockups. Finally, for the final digital mockup, a threshold is set and Boolean logic is used to substitute solid for glazed components (Fig.4).

The abstraction of data allows students to see data in freeform ways by removing conceptual barriers to both creating and using data.

This project can likely be introduced as early as the first or second design studio with very little dependence on the BIM modeler and a focus on the visual programming interface.

Module: Data as Form (Tower)

High-rise form development using visual programming, while cliché, is also a useful instructional technique. In this module the building floor plate is defined by 4 points joined into a surface using NURBS curves and manipulated by rotating, translating and offsetting the initial surface. Students see how complexity can come from relatively simple math. In this project the sine function is used to create building curvature (Fig.5). Variation is created using slider inputs to vary parameters for height, twist, and curvature, etc. Example applications of this project as a module might be a component of an urban planning design studio or building massing project.

Module: Form from Mesh (T-Splines)

T-Splines can be thought of as the three-dimensional equivalent of a two-dimensional NURBS curve. In this module, students create a form and divide it into a panelized form, and apply T-Spline forms with variable inputs for shape values of radius and fillet, etc. (Fig.6).

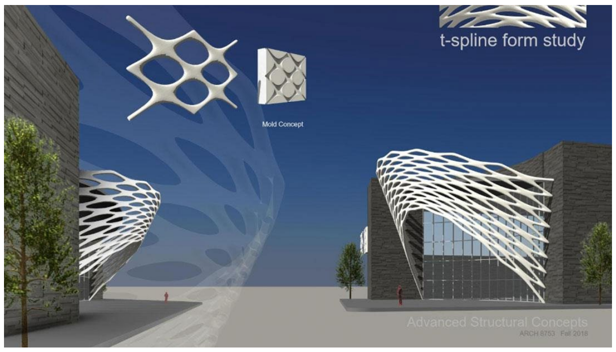

Fig. 7 T-Spline Form Study Project.

It is interesting to note that T-Spline geometry was a component of one of Autodesk's products (Inventor) and is now "exposed" in Dynamo as an experimental function with approximately 150 nodes.

T-Spline geometry is new and evolving but evident in some of Zaha Hadid Architects projects [7] (Schwerdtfeger, E. 2018). Because the resulting forms are organic and interesting, a module could be part of a first- or second-year design fundamentals studio in the program.

Module: Visual Programming and Making

Translating complex forms into buildable components is one of the most powerful applications of CDVPE. In order to build complex forms, one typically needs to be able to manipulate from a position in three-dimensional space to a two-dimensional plane for cutting sheet goods or creating construction drawings. This module takes a curvilinear form and panelizes it before rotating the panels onto a fabrication plane (Fig.7). Students get both the basic ideas of translating geometry and an understanding of flatness as it relates to panelization. When panels are oriented to a plane, the out-of-plane shape of the panel becomes apparent. The techniques

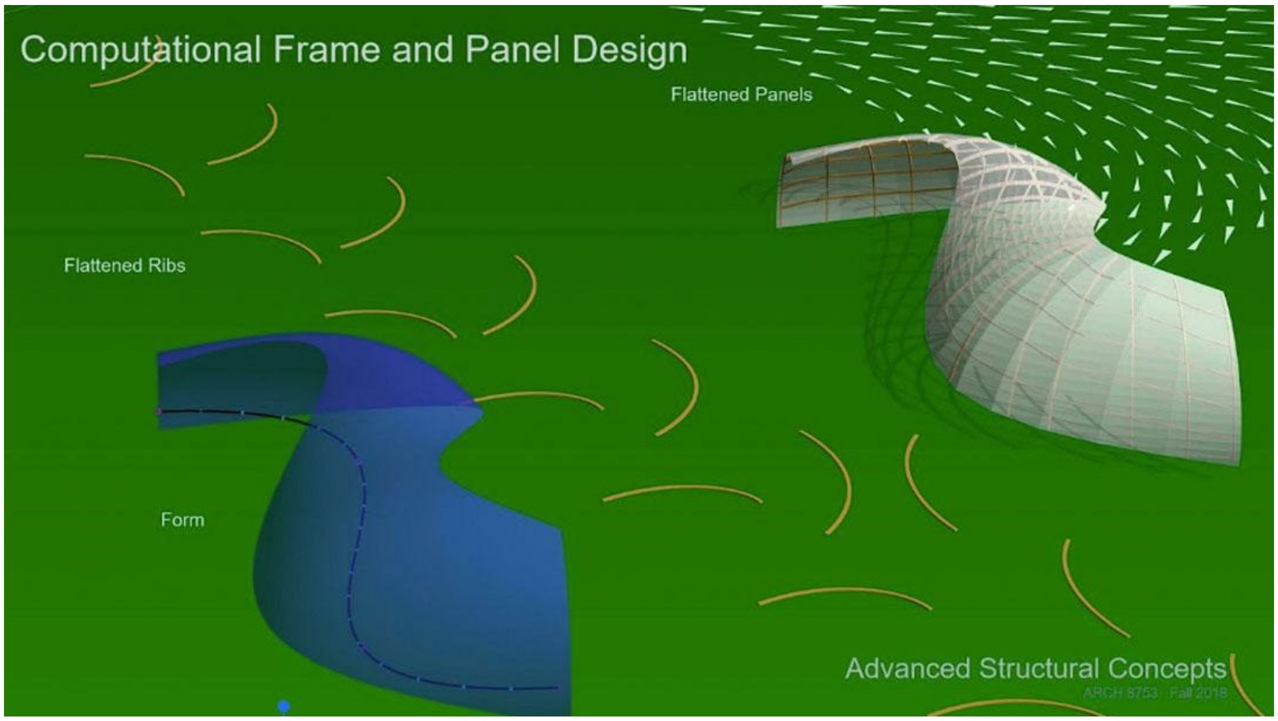
Computational Frame and Panel Design
Flattened Panels
Flattened Ribs
Form
Advanced Structural Concepts
ARCH 8753 Fall 2018

Fig. 6 Project to Design and Flatten Components.

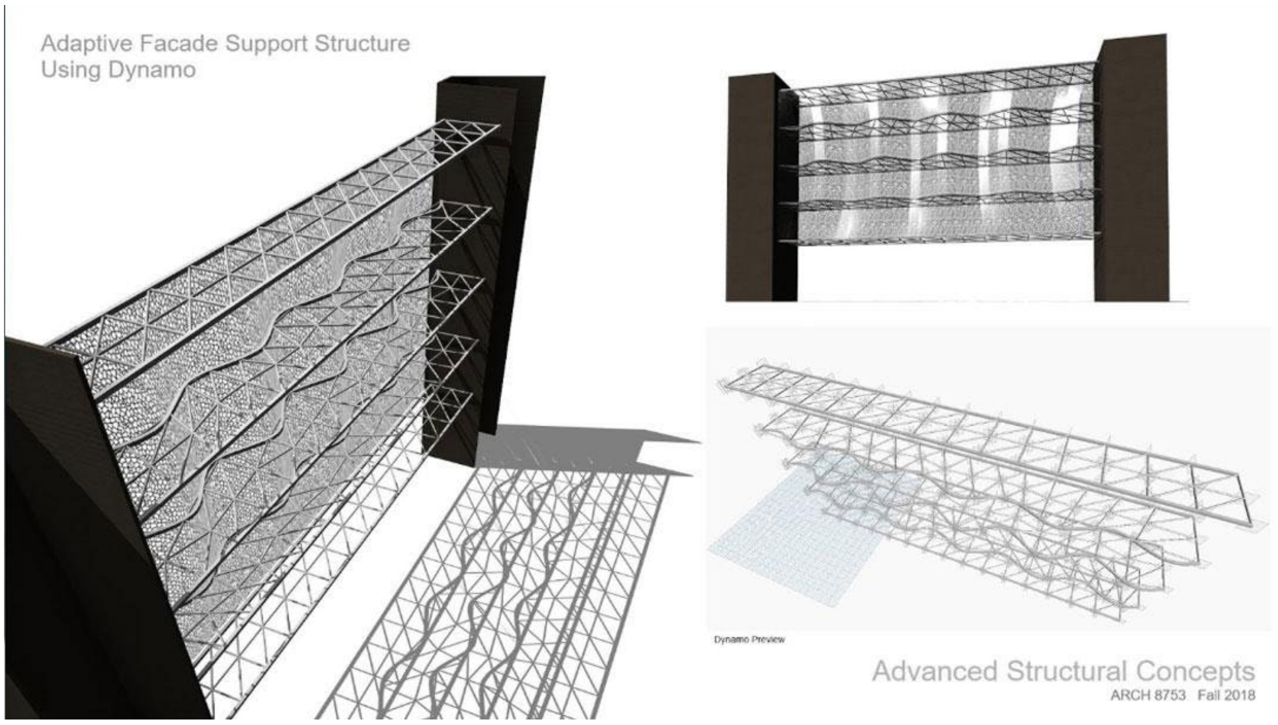
Adaptive Facade Support Structure
Using Dynamo
Dynamo Preview
Advanced Structural Concepts
ARCH 8753 Fall 2018

Fig. 8 Computational Truss

developed in this module would be a good fit for studio work involving physical model making and fabrication

Module: Complex Structural Elements

In a previous project, students constructed a wave facade and were then directed to develop structural trusses to adapt to the facade form. Top and bottom truss flanges are identified by intersecting surfaces to find the contour. Flanges are then formed by extruding a profile along the contours. Web bracing is developed by creating a frame in the surface and finally completing the web with the extrusion of a cylindrical profile on web brace elements (Fig.8).

This module could be part of the truss component of a structure's class.

Frame on Form

Visual programming is used to define a form to which an adaptive structural family (BIM component) such as structural ribs can be applied. Inspired by the forms of architect Santiago Calatrava [8] (Calatrava, S. 2014) (World Trade Center Transportation Hub), the visual programming element allows all the shape parameters to be adjustable including the number of ribs (Fig.9). Ribs can be constructed of a variety of structural materials allowing this to work in structures courses or studios which explore materials and their application and expression.

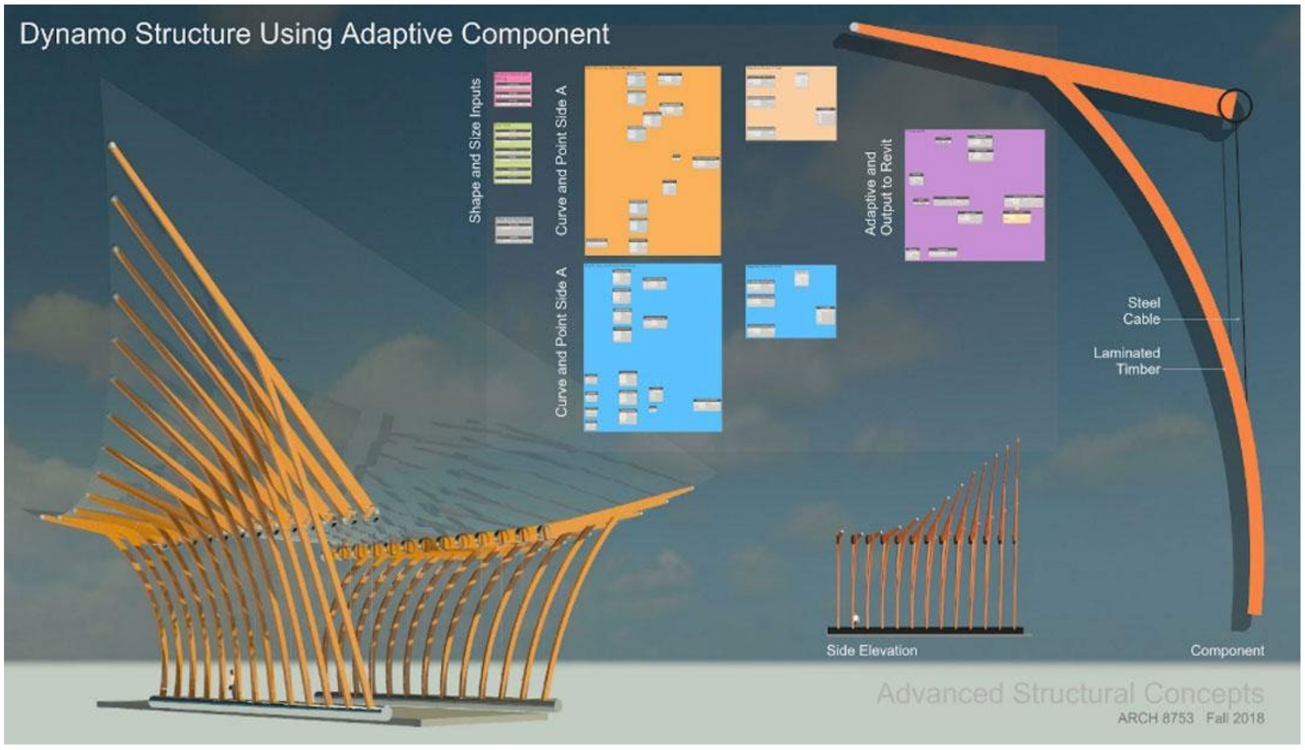

Fig. 9 Adaptive Component Applied to Computational Form

Future Modules: Automated Reiterative / Optimization Modeling (In Development)

Once BIM models are set up parametrically with accessible parameters, they can be uploaded into software that can automate reiterative modeling. Autodesk offered a web version that would allow uploading Dynamo files and provide variable input values (according to user setting) to run the model and save the output (Fig10). Users could review the output for those achieving desired criteria, and because it was web based, it could share the outputs remotely with collaborators. Discontinued in January of 2019, the company is working on an alternative version (Refinery) that is in development with a preview version available. Other companies have similar tools in various stages of development.

Sharing Modules Throughout the Web

Dynamo's visual programming environment can be uploaded to the web, and the ability to share across the web interface opens up opportunities for collaboration. It is unclear where the development of this web product is moving, but as it takes new forms it shows promise as an experimentation tool for collaboration in the classroom (Fig.11).

Methods for Sharing Modules Throughout Courses

Most colleges and universities use course management software that provides a platform for content delivery. Blackboard (software tradename) is used by SUNY Alfred State and is effective for delivering high definition (1920x1080) quality instructional video through high-speed internet connections. The format also allows for multiple instructors, important for responsive technical

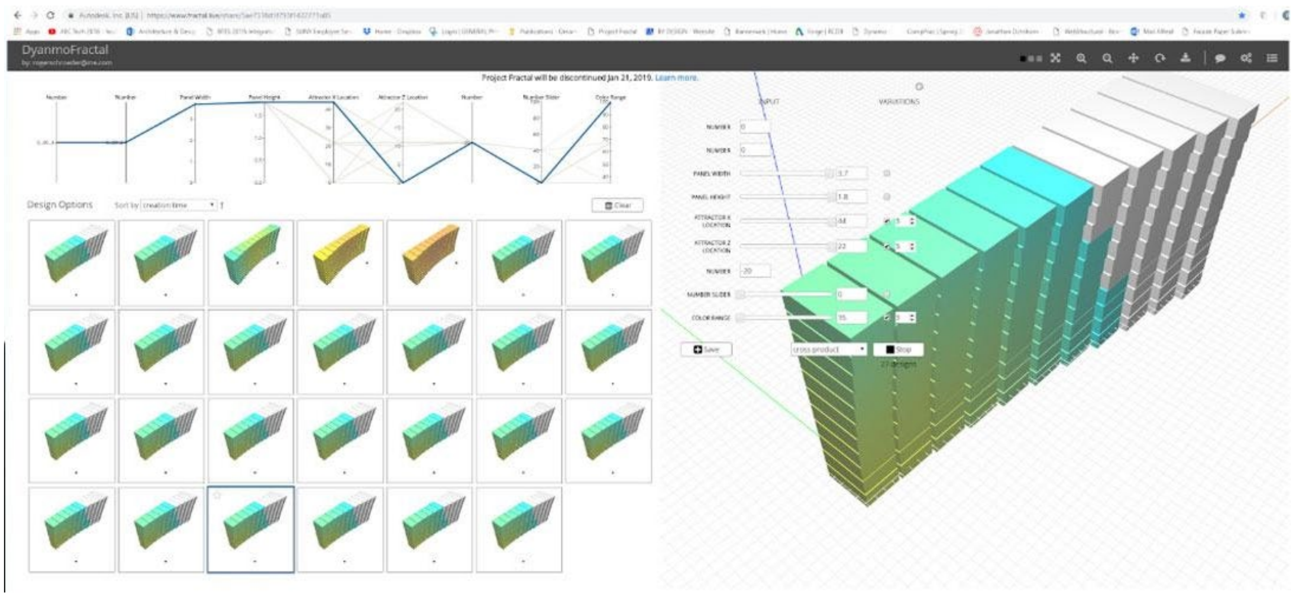
DynamoFractal
Project Fractal will be discontinued Jan 21, 2019. Learn more.
Design Options
INPUT
VARIATIONS

Fig. 10 Autodesk Fractal Live

support for primary course instructors unfamiliar with specifics of visual programming software. The faculty have set up a development "dummy course" to host modules in development.

Developing an Implementation Plan

The concept of introducing shared content between courses might need to be envisioned as part of a larger college experience based on familiar concepts like a lecture series where participation is expected school wide. The depth of an initial trial might involve three courses in a single semester with the related modules built to be a two-hour online tutorial/project. Course faculty would participate in crafting or modifying content and determining scheduling and grading assessments that are consistent with the specific course requirements. Modules would be developed and faculty would be trained on the techniques prior to the semester implementation. Surveys would be added to existing course evaluation tools to provide specific assessment data.

Discussion

All of the projects presented for the initial development of the modules were given to fifth-year Bachelor of Architecture students with no visual programming experience. These students were mature and motivated to complete projects. Students many times expressed frustration with Dynamo issues creating time-consuming functional problems when running. This will be an important consideration going forward when planning for implementation with students in early parts of the undergraduate program.

Faculty interest and enthusiasm to adopt new elements into establish courses will determine the effectiveness of the investigation of these modules.

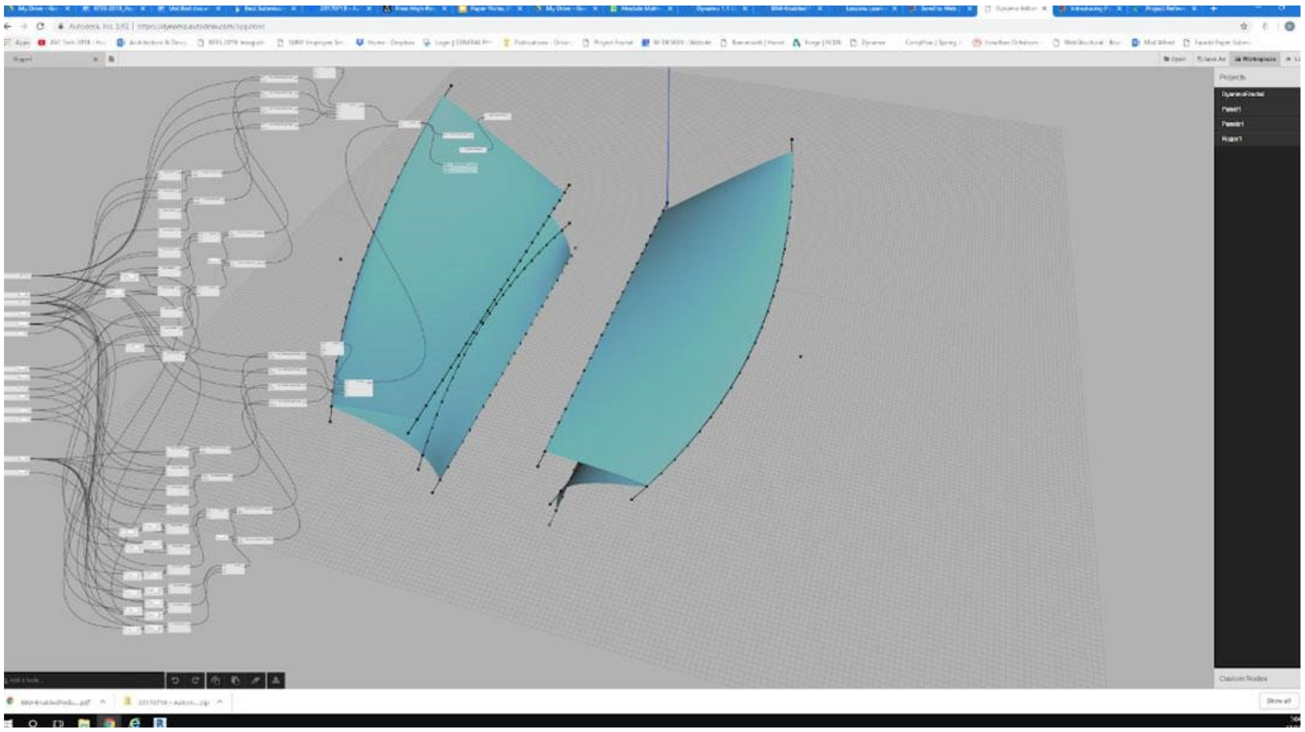

Fig. 11 Dynamo Editor (web based interactive interface)

Conclusions

While Computational Design and Visual Programming Environments appear now to be an advanced curriculum content that may be become accessible to undergraduates with well-developed projects and exercises throughout modules introduced into the curriculum.

Initial success with visual programming at the first-year level indicates that early introduction of these concepts may be effective in other courses encouraging the next steps in initial trials.

Further experience with fourth year students in using CDVPE's demonstrates the possible introduction of these tools earlier in the program and into a variety of courses.

References:

1 Naboni, E. "Environmental Simulation Tools in Architectural Practice." PLEA2013 - 29th Conference, Sustainable Architecture for a Renewable Future, Munich, Germany. September 2013.

2 Preidela, C. and Borrmanna A. "Automated Code Compliance Checking Based on a Visual Language and Building Information Modeling." Conference: International Symposium on Automation and Robotics in Construction and Mining (ISARC 2015). June 2015

3 McGinley, T. & Fong, D. "DESIGNGHOSTS Mapping occupant behavior in BIM". Proceedings of the 20th International Conference of the Association for Computer-Aided Architectural Design Research in Asia CAADRIA 2015. May 2015.

4 Becerik-Gerber, Burcin & Gerber, David & Ku, Kihong. "The Pace of Technological Innovation in Architecture, Engineering, and Construction Education: Integrating Recent Trends Into the

Curricula." Electronic Journal of Information Technology in Construction. 16. February 2011.

5 Hu, M. "BIM-Enabled Pedagogy Approach: Using BIM as an Instructional Tool in Technology Courses." Journal of Professional Issues in Engineering Education and Practice, October 2018.

6 Harris Poll Retrieved from web: "The Next Generation of Learners". Retrieved from https://pearson.com/content/dam/one-dot-com/one-dot-com/global/Files/news/news-annoucements/2018/The-Next-Generation-of-Learners_final.pdf

7 Schwerdtgeger, E. 2018:"Custom Computational Workflows for BIM Design Implementation" Retrieved from: https://www.autodesk.com/autodesk-university/class/Custom-Computational-Workflows-BIM-Design-Implementation-2018 November 2018,

8 Santiago Calatrava – Architects & Engineers. (n.d.). "World Trade Center Transportation Hub: New York." Retrieved from https://calatrava.com/projects/world-trade-center-transportation-hub-new-york.html