ROBOTIC WORKFLOW

An Architectural Pedagogical Approach

KEVIN SWEET
1 Victoria University of Wellington, Wellington, New Zealand
kevin.sweet@vuw.ac.nz

Abstract. While new software interfaces are making the interaction between humans and robots more pedestrian, there is still an extremely complex workflow from the conception of data on the part of humans to the final action of the arm. In order to continue to promote and advance the use of these versatile tools in architecture, pedagogical strategies are needed to better enable users to engage with them quickly and obtain results while minimising frustration. This paper will outline a pedagogical strategy for introducing the multi-layered levels of knowledge and understanding required to operate a 6-axis robotic arm as developed in undergraduate architectural coursework. It will highlight the various learning modules created in order to deliver the necessary information for understanding the complex operational pipeline required to interact with and operate the robotic arm successfully.

Keywords. Robots; fabrication; parametric; parametric modelling; simulation.

1. Introduction

Every few decades or centuries, a new set of skills and intellectual activities become crucial for work, conviviality, and citizenship—often democratising tasks and skills previously only accessible to experts. (Walter-Herrmann and Büching, 2014)

Robots are stupid. They require explicit directives to move from one point in space to another, let alone complete complicated instructions like the process entailed in building a wall. Yet we are still fascinated with them because they offer nearly limitless possibilities. Much like the human arm, any tool
can be attached to the end of the robot and it can be programmed or directed to perform nearly any task imaginable – or at least any task that can be mathematically described. Because they are such a close analogue to the human arm, they tend to be anthropomorphised. There is a sense of wonder, empathy, and even romanticism when they perform. They have been called upon by architects and designers to stack bricks, fold metal, cut foam, 3D print and even plaster walls. It seems that these incredibly flexible machines can perform whatever is asked of them without complaint. What is often misunderstood, or not represented in what we see from the outside, is the extremely complex and underestimated amount of time and knowledge that is required to get the robots to undertake even the most menial task.

At first glance, using a Computer Numerically Controlled (CNC) industrial robot is similar to using most other CNC machines. You create the initial data, translate that data into code that the machine understands, then let the machine produce the object. However, unlike most CNC machines, the tooling and the environment of a robot is far more complicated. The advantage of the robot lies in its complexity: the tool and work surface are not predetermined by the limitations or functionality of the machine. The robot can interact with any work surface or fixture within its reach and use any tool that is attached to the end of it. This subtle, but exponentially large difference requires a greater level of understanding of the entire workflow in order to operate the robot successfully.

The production workflow for the robot is not something that can be learned from a single software application or owner’s manual. Many layers of potentially unrelated knowledge have to be acquired in order to synthesize them into a cohesive set of instructions that allow the robot to perform in an expected manner. This paper outlines a pedagogical strategy for the introduction of the multi-layered levels of knowledge and understanding required to operate a 6-axis industrial robotic arm as it was developed in an undergraduate architectural course. Through the application of a series of three learning modules, students gained an understanding of the multifaceted systems necessary to successfully operate the robot.

The first learning module engaged students in a physical to digital feedback loop that introduced them to parametric "thinking" and parametric modelling. Parametric thinking – the ability to change a defined parameter in a system – is ideal for a machine like the robot that can perform a programmed task repeatedly. The change in parameter allows for the physical manifestation of a design with minimal effort. The second module provided a link between the digital, parametric model and conventional building practices. The third and final learning module combined the tools from the first two modules and introduced the complexities of the physical work environ-
ment including the tooling of the robot and the design of its work area. The work of the final module resulted in a complex, material output with extremely varied results that were tested quickly, easily and recursively with the robot. Engagement in these learning strategies demonstrated to students how it is now possible to regard computer programming and architectural construction as conditional upon each other, and to see their reciprocity as fundamental to architecture in the digital age. As a consequence, the digital becomes concrete and tangible. Hereby the robot is both a symbol of, and a primary tool for, a profound reformation of the discipline. Through the robot, architecture develops a form of materialisation adequate for the information age.(Gramazio et al., 2014)

2. Physical to Digital Feedback Loop

It may appear counterintuitive to begin a course about an industrial robot with physical modelling but sometimes the simplest approach is the most effective. The students in this course were undergraduate students in their third year of school with little to no knowledge of construction practices or even tool use. Given this, combined with the fact that they had never used any parametric software previously, it was an obvious point of departure to introduce students to the principles of parametric design. Requiring students to begin with physical modelling that asked them to perform repetitious tasks with only slight variations to the model each time, allowed them to get a tactile sense of how small changes in design parameters could have large ramifications on the final output. It also allowed them to understand how digitally simulating models parametrically would allow them to explore variations quickly and how the robot could physically build them. Students were taught that parametric design is not a software application, but a way of thinking about design: a computational way of thinking based on “processes [that] start with elemental properties and generative rules to end with information which derives form as a dynamic system.”(Menges and Ahlquist, 2011)

2.1. CODIFYING THE AGGREGATION

In the first exercise, students were initially given three, predefined planar elements and asked to physically reproduce them via the laser cutter. Once they had cut a large number of the elements, the students were then asked to assemble them into three-dimensional forms. They were not given a set of predetermined parameters regarding how the elements were to be assembled, but were required to design the form so that the process of construction was repeatable. As part of this process students were asked to carefully document
the steps involved in the assembly of the form, inclusive of explicit notations regarding the directionality and orientation of each element, emulating how they might have to "program" the form into software at a later stage.

The intention for this portion of the project was to introduce them to "parametric thinking" by requiring them to codify the relationship of the elements to each other. An important facet of the learning process, this approach allowed them to focus solely on the system of logic and relationships that bound the individual pieces together, rather than the aesthetics, functionality or variants of the element itself.

Parametric design depends on defining relationships and the willingness (and ability) of the designer to consider the relationship-definition phase as an integral part of the broader design process. It initially requires the designer to take one step back from the direct activity of design and focus on the logic that binds the design together. This process of relationship creation requires a formal notation and introduces additional concepts that have not previously been considered as part of ‘design thinking’. (Woodbury, 2010)

2.2 IDENTIFYING BEHAVIOURS

The second exercise asked students to begin to observe patterns or behaviours that resulted from the assembly of forms produced in the first exercise. When students connected enough of the elements together in a controlled manner, certain formal patterns emerged and the pieces themselves began to form larger modules that could then be further codified as a collection of el-

![Diagram of predefined element and examples of students' coding system.](image-url)
ements. These newly found modules could then be reassembled into a new, more complex form. This discovery was key to the introduction of the potential of computational thinking and the concept of emergence.

Emergence is generally understood to be a process that leads to the appearance of structure not directly described by the defining constraints and instantaneous forces that control a system. Over time ‘something new’ appears at scales not directly specified by the equations of motion. An emergent feature also cannot be explicitly represented in the initial and boundary conditions. In short, a feature emerges when the underlying system puts some effort into its creation. (Crutchfield, 1994)

If the behaviours they witnessed in the physical models could be codified and translated into an algorithm, students could begin to explore “form as a dynamic system” (Menges and Ahlquist, 2011) through the parametric software. Students were not be limited by the number of pieces they could physically make and assemble into forms, but rather they could explore the myriad opportunities virtually through the software. This connection between the haptic experience of the physical model and the virtual interpretation was a key to understanding the how programming and construction are conditional upon each other: that “reciprocity is fundamental to architecture in the digital age”. (Gramazio et al., 2014)

Figure 2. Physical aggregated forms and simulated digital forms.

2.3 VIRTUAL SIMULATION

The final exercise of the first module introduced the parametric software Grasshopper as a tool to explore the behaviours that students witnessed and described in the previous exercise. Utilizing the coding system they developed, students were given the tools to replicate the individual elements and, ultimately, the modules or behaviours observed in the physical environment in the virtual environment. Compared to previous courses where there was
no physical component, the high learning curve of comprehending and programming Grasshopper was made easier as a result of the introduction provided in the previous exercise. By explicitly describing and producing a notational language for the relationships of each element to the module (see Figure 1), students were able to make the transition from physical to virtual space and understand what was required by the computer to digitally replicate the model. Parametric software requires very explicit instructions to locate an object in space and perform an operational task on it. For example, to rotate an object, the software needs to know a point of rotation, a direction of rotation and an amount to rotate. Because students used their physical objects and notational system to describe these parameters, they were better able to understand and code the software with required instructions.

Once programmed with a set of rules from the observation of the physical model, the virtual model could then be used to create simulations of larger forms comprised of an aggregation of modules (collections of the original elements) rather than the individual elements themselves. This was a key tool and discovery to the notion of emergence described above.

3. Virtual, Traditional Simulation

While the first learning module was developed to encourage a computational or programming way of thinking, the second learning module provided a link between the virtual parametric environment and a traditional constructional methodology, one that employed the robot as a tool to physically reproduce a design. This learning module offered a simplified method that used the student’s foundational knowledge of parametric modelling to introduce how to integrate the requirements of the robot into the workflow in order to produce a physical output. Rather than being distracted by complex algorithms or forms, students were asked to reproduce a relatively basic and traditional method of construction: laying a brick wall. The brick wall was first conceived virtually and was then constructed physically with the robot. The ultimate goal of the project was to demonstrate how robots are now connecting technology and knowhow, as well as imagination and materialization, like never before, and have the potential to reveal a radically new way of thinking about materialising architecture. This takes away the abstract and forced artificial character of the digital in architecture and imbues it with a totally distinct material significance and identity.”(Gramazio and Kohler, 2014)
3.1 TRADITIONAL WORKFLOW

In the first exercise of the second learning module, students were asked to create a parametric brick wall using traditional bond patterns; Running, Flemish, English, etc. This allowed students to engage in a focused exercise of programming the geometric rules of the pattern into Grasshopper and dealing with sequencing the data (bricks) for output. Since, in this case, the robot was programmed to emulate a construction process, the data had to be ordered so that the robot would build the wall by laying the bricks from the bottom to top.

Seemingly a simple task, the students were confronted by the challenges entailed in navigating the organisation of data that is contrary to how computers typically organise data. Data within the computer is generally organized based on the rules of the software designer and not based on rules of construction. Students must therefore understand and predict the rules of construction in order to organise the data correctly for output. Other rules such as the end condition of the wall (half-bricks), the mortar gap, and building tolerances also had to be considered and added to the complexity of the output data.

3.2 ROBOTIC SIMULATION

The second exercise introduced the plugin for Grasshopper called *Kuka|prc*. *Kuka|prc* allowed for a virtual, visual simulation that provided a safe environment for students to think through every step of the construction process, no matter how small, with the recognition that the robot will only do what it is told. Commands to pick up bricks, move to a location, lower the brick in place, clear any already placed bricks and return for another brick, all had to be programmed into the sequence of the robot. If any step in the process was eliminated or forgotten, the simulation would show that the robot was not able to perform the task as expected. This was a key pedagogical tool that helped students understand the entirety of a workflow and gain confidence that they could command the robot to perform as expected.

Without this plugin, the task of formatting the output data for the robot from Grasshopper or any other program would be impossible to teach within an undergraduate program. Due to its simplicity and seamless integration with Grasshopper, students were quickly and easily able to create the correctly formatted data for the robot.
3.3 PHYSICAL OUTPUT

The end of the second module required students to physically build the wall they previously programmed and simulated in the first two exercises. The use of traditional bond patterns allowed them to concentrate on the programming of the data and the resultant output. By building the wall, the connection between the programmed and simulated output could easily be compared to the physical manifestation of the wall. Errors in spatial location, clearances, sequencing, brick feeding, work surface, and gripper errors could be identified and rectified quickly because the output of the wall was a known quantity.

This step in the learning process was vital for students to become familiar with and understand the connection between the programmed data and the physical movement and reaction of the robot. It was analogous to the first learning module in that there was a physical result from a programmed set of actions: a key learning tool for understanding complex topics.

4. Robotic Workflow: Dynamic Wall

In the final learning module, students were asked to create a more complex version of the brick wall by generating a new formal bond pattern reinforced by computational methods as well as a secondary pattern, delineated by bricks of different colours. The small addition of coloured bricks increased the complexity of the programming as the students now had to insure that the appropriate coloured bricks were being fed to the robot according to the predetermined pattern. Both full and half-bricks were required to be managed and, with all variables in place, students were required to navigate the task of programming up to six different bricks types.

Along with the added complexity of programming the software to organise all of the new brick types, students also had to address tangible factors within the physical environment. Notable factors included the engineering required in the design of the end effector (gripper), as well as the bricks and the brick-feeding mechanism. The functionality of the robot and the accuracy of the work it performs, is dependent upon the fine-tuned relationship between the programme, robot, tool and material. Addressing these physical considerations entailed in the robotic workflow was an invaluable exercise that had a cascading effect throughout the entire learning process. Once these components were adequately designed, they had to be modelled and programmed into the digital simulation. The reciprocal relationship between each part of the system reinforced how a well-designed system is greater than the sum of its parts. It was through this exercise that students began to
understand how computational methods and robots were interconnected and what they could achieve – a sense of digital materiality.

*Digital materiality* leads to a new expression and – surprisingly enough, given the technical associations of the term “digital” – to a new sensuality in architecture. Digital and material orders enter into a dialogue, in the course of which each is enriched by the other. *Digital materiality* is thereby able to address different levels of our perception. It is characterized by an unusually large number of precisely arranged elements, a sophisticated level of detail, and the simultaneous presence of different scales of information. (Gramazio and Kohler, 2008)

5. Conclusion

The pedagogical approach outlined in this paper was successful in that the outcome of the course was an identifiable artefact that the students had a sense of pride and accomplishment about. The efficacy of the first module is questionable. The relevance of the modules and their aggregation may not have been a strong connection or transition to the next module. It is suggested that future courses either continue the physical learning module parameters (the physical connections and their variations) into the second phase or that the brick be introduced through the first module.

The real value of the course was the greater emphasis on the physical, material and spatial possibilities of fabrication with a robot – a condition de-
fined as ‘design robotics’ – and these are arguably the attributes that are most important to a contemporary architectural education.

Design robotics describes creative, material-focused automation approaches from conceptual digital design to robotic fabrication and construction. (Bechthold, 2014)

Design robotics has the added benefit of supporting a systems based way of thinking where the sum is greater than the parts.

The holistic nature cannot be seen in the individual part, nor can it be seen with the addition of its parts. The system behaviour emerges only in the dynamics of the interactions of the parts. This is not a cumulative linear effect but rather a cyclical causal effect in which the complexity of the level and amount of interaction cannot be directly deciphered.” (Menges and Ahlquist, 2011)

This, in the end, is what one would hope to achieve in a typical architectural design process: a result that is greater than the sum of its individual parts. An academic exercise that utilises a robotic arm emulates this type of design process but on a smaller, appropriate and achievable scale for architectural students. The entirety of the process – from digital conception, to virtual simulation, to physical manifestation – clearly demonstrates that the “reciprocity between programming and construction is fundamental in the digital age”. (Gramazio et al., 2014) It is not just about the tool or the immediate outcome of the tool but it is about a learning process that prepares students for an architectural profession that continually relies on these tools and processes to evolve the field.

References
Walter-Herrmann, J. and Büching, C. (eds.): 2014, *Fablab: Of Machines, Makers and Inventors*, Transcript Verlag,