Integrating Physical Computing with Making

Why can't a building be as smart as a car? Currently automobiles of all varieties contain numerous sensors—close to 400 sensors for high-end models.[1] For a long time cars have displayed speed, engine temperature, distance traveled, or the amount of fuel consumed, but these don't come close to accounting for the amount of data your car collects.

JOSHUA VERMILLION
University of Nevada Las Vegas

Newer vehicles can sense when you are wearing your seatbelt (and when you aren't), monitor air pressure and engine temperature, help you answer your phone without using your hands, and detect when your tires are spinning too fast, or when your tires have locked up. Some cars are equipped with cameras and distance sensors that prevent the driver from getting too close to traffic or from hitting objects while in reverse. Others have features that are activated by voice command. Car problems are diagnosed by plugging the vehicle into a computer, and, in the not so distant future, they might even become driverless, autonomous vessels that sense, communicate with, and react to neighboring cars in real-time. As time marches on, cars sense and collect more and more data and use this information to respond to various driving scenarios in order to enhance performance.

Suffice to say, the built environment is also rich with opportunities for embedding and integrating digital technologies and sensors to create responsive and adaptable systems. Rather than speculating on how these systems might transform our environment, this paper outlines efforts to integrate the prototyping of responsive systems with course design projects that already focus on making, primarily using digital fabrication tools and methods.

PHYSICAL COMPUTING

Tom Igoe and Dan O'Sullivan of New York University use the term "physical computing" to refer to active systems that can sense, interpret the sensed data computationally and, in response, physically change.[2] These systems typically use a small computer, called a microcontroller that needs to be programmed. Microcontrollers are typically tethered (via electrical circuits) to any variety of sensors for information input, as well as to actuators to output responses. The technical implementation of these systems at scale requires a broad range of skills that span multiple knowledge domains—design, engineering, mechanics, programming and computer science, robotics, mathematics, electronics—just to name some. There are examples of built design projects that successfully negotiate these multi-disciplinary challenges and deploy them to create responsive prototypes to create spatial effects and experiences. For instance, recently exhibited projects include ceilings that move and change color,[3] walls that sense and emit light,[4] panels that fold in shape according to sound and acoustic qualities[5], or even completely immersive environments with parts that change, move, and reconfigure both locally and systemically.[6]

TINKERING AS AN ETHOS FOR DESIGN INVESTIGATIONS

Assuming that the design of the built environment will increasingly integrate physical computing systems, does our architectural repertoire of skills and knowledge need to be adjusted to meet these challenges? In particular, how do we educate and prepare architecture professionals for a future of physically active and interactive environments? Of course, it seems impractical to propose that professional architects will also have to be professional programmers, engineers, and electricians. Fortunately, it is not necessary to have a comprehensive understanding of these topics to begin prototyping a physical computing system. At small scale, results can be achieved by borrowing and repurposing snippets of programming, hacking widely available hardware, learning from any number of web resources posted by a very active global community of makers, and simply experimenting with components—tinkering substitutes for expertise.

Most are probably familiar with the traditional dictionary meanings of tinkering, but defined with more rigor, tinkering can be combined with engaged or constructivist learning frameworks. In fact, this combination is currently being applied in many schools in the U.S. to foster STEM (Science, Technology, Engineering, and Math) learning for K-12 students.[7, 8] These frameworks are also helpful when introducing students to physical computing in design and fabrication courses. The sophistication of results changes with timeline, scope, scale, and complexity, yet the following methods are adaptable in application:

1. Experimentation and play are important parts of learning, particularly when confronted with new concepts or new tools to explore concepts. Students should be prompted to ask questions during structured "play" and these questions lead to more experiments.

2. Iterative prototyping and testing create feedback loops for incremental improvements. When performed rigorously, they yield successes and surprises as well as failures (a natural and necessary part of learning).

3. Repurposing involves taking an idea, method, or widget out of its situational context and applying it elsewhere or in a new way. Often, disparate concepts or techniques can be repurposed and combined to create something novel.

Prototyping physical computing systems, particularly at first, presents a steep learning curve in implementing the technical aspects necessary to sense, compute, and actuate. The choice of an appropriate platform helps alleviate technical difficulties. Robust platforms can provide more intuitive methods and interfaces for beginners and typically have a large web-based community of users, documentation, and other helpful resources. Two platforms—Lego Robotic NXT kits and Arduino microcontroller kits—are used in the following student projects. Both of these platforms encourage novices of all age groups to tinker, prototype, and create.

APPLICATION

The following projects were created in courses co-taught by the author. They demonstrate, to various degrees, each of these aspects of tinkering. While the students were learning about physical computing systems and practicing many skills related to their technical implementation, they were also asked to focus on design and fabrication, detailing, human interaction, and spatial effects.

1

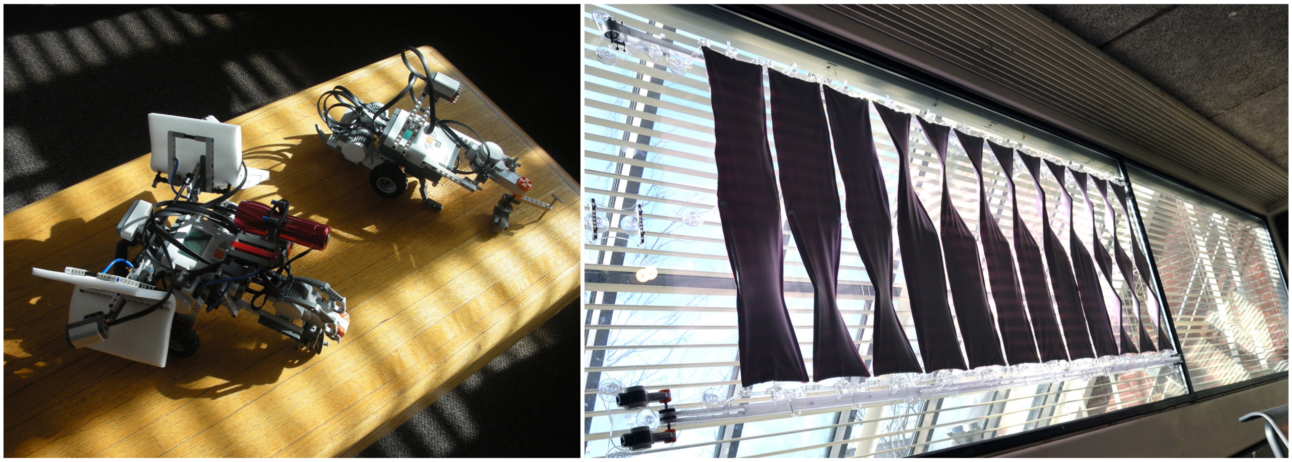


Figure 1: Example Legobots (left) and Twist installation (right).

LEGOBOTS

“Legobots” was an initial “structured play” project in a design studio co-taught with Mahesh Daas at Ball State University. The studio was tasked to explore the creation of prototypes or “Legobots” that could behave according to a small set of stimuli and rules. Tasks were given for the organism to perform without prescribing how the tasks were to be accomplished. LEGO NXT kits were useful for developing these prototypes quickly—the kits have predesigned connection systems, are easily assembled, modular, re-configurable, and packaged with sensors, microcontrollers, and actuators all driven with a visual programming interface. Failures were abundant as the students quickly found the limitations of these kits. For instance, sensing ranges for light, sound, and proximity had to be discovered and carefully controlled by physical location and direction of sensors, as well as by calibrating and fine-tuning the programming.

The small student teams quickly customized the LEGO kits, including modifying and integrating mobile phones (with blue-tooth technologies), affixing lights and drawing instruments, or creating several Legobots that worked in tandem to accomplish particular tasks or behaviors. The LEGO NXT software allowed the students to program the robotic behaviors through a visual interface without the burden of learning a particular scripting language. An important point to note is that most of the students had little or no programming experience when they started the studio. This interface provided a robust framework for beginning students to program and test complex behaviors while introducing the fundamentals of scripting, such as linking numeric parameters to functions, creating conditional statements, and looping—all enabled by quickly testing the outcomes. In parallel, the students attended several workshops to learn how to use Grasshopper, a visual programming plug-in for Rhino software. Similar to the LEGO programming, Grasshopper modeling was used as an introduction or precursor to slightly more advanced procedural modeling using Rhino Script (introduced later in the semester). The parametric capabilities of Grasshopper helped some of the students design custom, laser-cut components to extend the capacity of the LEGO NXT kits for specific behaviors such as “aiming” the directionality of sound sensors.

With a thorough working knowledge of the LEGO systems, the teams realigned to create scaled-up, reactive prototypes able to inhabit specific sites and engage human behavior. Scaling up involved many more difficulties for the student teams such as amplifying forces and movements, while minimizing weight. Initial ideas were drawn or modeled in parallel with drafting statements about each team’s

2

project intentions. While parametric modeling was initially used by some of the teams to model a first design iteration and simulate its behaviors, the students quickly met the limitations of their software under strict time constraints. In these cases, physical prototyping proved to be the critical method of driving the design and innovation process. Most of the design changes and development occurred through the building, testing, and modifying of full-working prototypes. This iterative feedback loop greatly enhanced the students' awareness of each project's performance related to materials, weight, scale, forces, and movement. The projects that underwent the most prototyping, from very early in the process, were the most successful in terms of negotiating site, kinetics, detailing, and experience.

An example installation from this project, called "Twist", used custom-made drive belts to twist stretched-cloth panels in patterns. This project attached to a linear expanse of windows and sensed passers-by in an adjacent hallway, twisting and opening sequences of panels to reveal sunlight and views to the surrounding campus. The project consisted of a modular, expandable kit of parts that were laser cut from acrylic. All connections were achieved without traditional hardware, underscoring the importance of tolerances and details. This modular, "plug-in" design and assembly logic was key to testing and improving the installation's performance. Sets of components formed modular assembly systems such as framing systems, stretching systems, pivoting systems, twisting systems, etc. If one of these systems failed to perform, particular system components could be redesigned and fabricated quickly, while ready-made to plug back in to the larger whole. This partitioning of functions and systems enabled adaptations to particular component designs with minimal interference or redesign of the entire prototype.

ARCUS ANIMUS

Jumping in scale, "Arcus Animus" was an installation constructed over a long weekend by students in a design studio co-taught by the author as part of an immersive workshop led by Philip Beesley from the University of Waterloo. Arcus Animus was a hanging installation composed of several layered mesh works consisting of acrylic, bamboo, and mylar components. The installation reacted to human occupation interpreted by arrayed proximity sensors. These physical reactions consisted of "shaking" movements actuated pneumatically using solenoid valves and custom air muscles. The workshop accelerated the students' learning and application of many technical skills related to digital fabrication, electronics, and micro-controller programming. Beyond this, much was learned about teamwork and group dynamics particularly as all of the work occurred over a short, accelerated time line,

Figure 2: Arcus Animus installation photos.


3

Figure 3: Morpholuminescence installation photos.

necessitating the efficient delegation of many overlapping tasks.

The installation was designed in advance of the workshop, and this led many of the students to question the importance of their roles in the project. Rather than claiming sole “authorship” of a project by starting from scratch, the students were confronted with more team-oriented roles and a blurred, collaborative authorship. The project, in fact, could be seen as being part of a long lineage of collaborations and inventions from various students and professors over many iterations. After the workshop, many of the students realized that they had made meaningful contributions to the project, and had been able to design or “author” changes in many different ways--including site selection, designing or modifying new components for last second problem-solving, or manipulating programming code.

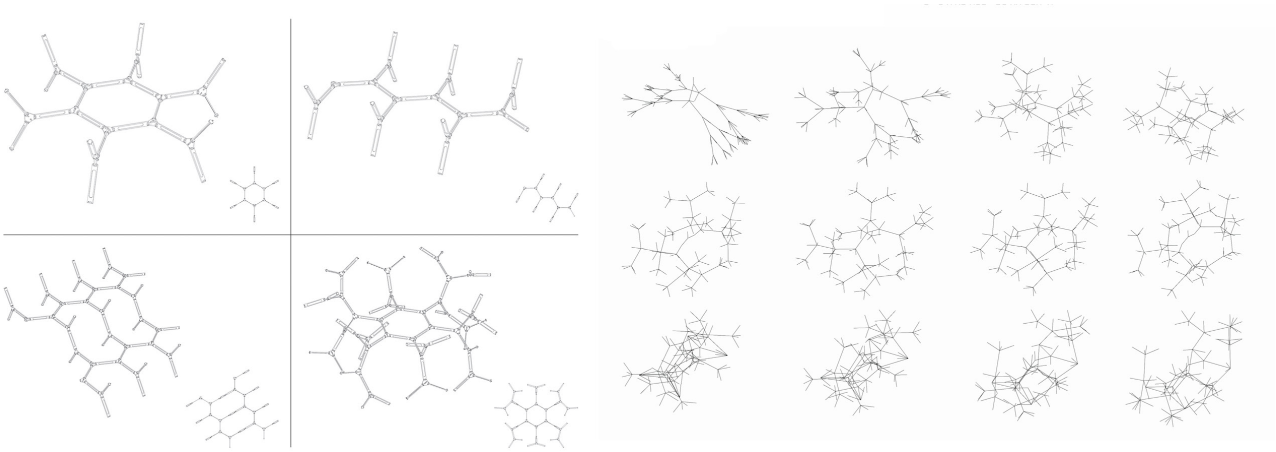
MORPHOLUMINESCENCE

Morpholuminescence was a spatial installation built by a small student team by applying the new skills learned while completing Arcus Animus (the previous project mentioned above). This project was developed as a student lighting design competition entry. The competition brief asked for lighting proposals for retail fitting rooms. The lighting scheme was interpreted from a traditional three-point studio photography lighting set up to highlight the changing subject when modeling in front of a mirror. The posture of the human subject is tracked by proximity sensors to control hinged triangular petals and variably tuned lighting. When the fitting room is unoccupied, the petals drop revealing variable RGB LED lighting to highlight the fitting room area with bright colors. When activated, the petals begin to close to form a faceted but continuous acrylic light surface while the fitting room lighting color and intensity changes—brighter for the task of changing clothes, and then optimized for highlighting the human subject in front of a mirror.

Morpholuminescence was prototyped over two months. Many design revisions were made to work out the mechanical movement of the petals. Basic lever mechanics, proper sizing of servo motors, and calibration of sensors and code were carefully worked out over several design and prototype iterations. Upon completion, the full-scale working prototype was shipped to Florence, Italy and installed in an exhibition at the Beyond Media Festival.

LATTICE INSTALLATIONS

In early 2014, the author conducted a one week long short course at the University of Calgary as the invited Taylor Seminar Lecturer. As a part of this course, the students were tasked to design interactive installations without embedding sensing or computing. As a result, the applied projects involved the design and fabrication of



4

self-assembling spatial lattices. The students worked in small teams and leveraged algorithmic design methods to generate lattice systems. Parallel to the systemic designs, the teams fabricated prototypes and mock-ups in order to work out the component shapes and details. Spatial lattice systems were introduced in order to provide structure and focus to the design problem. Lattices have inherent advantages that aligned neatly with the objective of creating installation-scaled prototypes, namely structural capacity with even distribution of loads, redundancy of parts that provide a systemic robustness, and repeatable patterns of modules.[9]

Repetition was crucial to the students' understanding of lattice systems and the design problem at hand. Small teams developed simple physical components and devised simple rules for an algorithmic approach to understanding part-to-whole relationships as lattice systems were developed. These methods were important for devising more or less complex systems from very simple parts and rules. These methods also enhanced the students' understanding of algorithmic systems, dealing with complexity and ambiguity, and exploring the shared dynamic between a designer and the computer.[10] Iterative physical prototyping was incredibly valuable for experimenting, testing, and simulating lattice components and the resulting systems. In this sense, fabrication (which the students were skilled at) became a useful vehicle for testing parts-to-systems aggregation, details, weight, scale, and effects. Because of extreme time limitations, the students focused on repetition of simple parts, a limited material palette, and extremely simple electronic components, such as coin cell batteries, copper tape, and LEDs.

From the numerous physical prototypes that were developed, it was decided to experiment with self-assembly—the ability of parts to organize into systems without external directions. Skylar Tibbits' work at MIT on self-assembly in design was a useful resource to better understand how such systems could work and provided the inspiration for using magnets to develop interlocking details for assembly and organization.[11, 12] Efforts with prototyping focused on the design problem of creating order from disorder through human interaction with the installation. In other words, actuation was achieved through a widely available source of energy—namely human motion. An observer applying energy to one part of the installation creates a disturbance in the field of components that leads to a chain reaction as magnetized components snap together in particular configurations encouraged by the polar directions of the magnets. As components snap together these connections close primitive circuits (made from copper tape) that turn on LED lights, rendering a real-time visual of the phenomenon. This focus on self-assembling lattices allowed the

Figure 4: Lattice system configuration patterns.

5

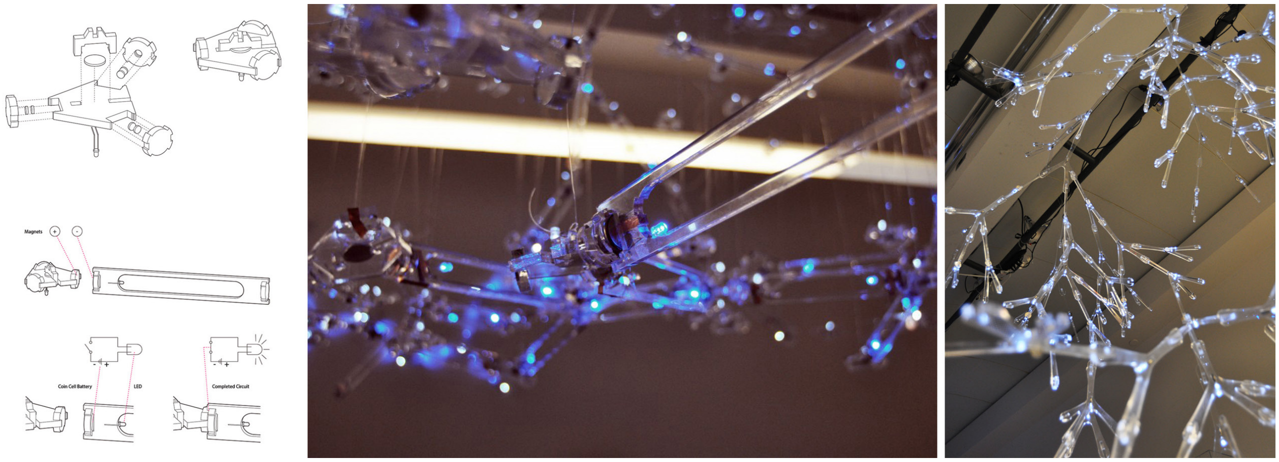


Figure 5: Lattice connection details (left) and installation photos (center and right).

students to experiment with interactive systems without the added challenge of learning more complex skills in engineering, programming, or electrical systems.

The final prototypes, full-sized installations in a gallery space, partly worked but had many problems for further troubleshooting. The first installation was derived from an aggregation of acrylic linear struts and tri-directional joints. Each potential joint was magnetized for quick interlocking assembly while suspended in a triangulated grid. The struts were introduced (one-by-one) to form a three-dimensional lattice system. A second installation was generated algorithmically from a branching variation of a Lindenmayer system. Comprised of acrylic bifurcating joints and struts, the final prototypes physically demonstrated the variability that can be achieved using self-similar component systems. Integrating self-assembling components into the lattice systems presented many challenges, and these challenges seemingly increased as the scale of each system increased. Nevertheless, when reflecting on the week's events with the students, it was clear that much was learned in a very compressed amount of time, and there were well-crafted artifacts to demonstrate the cumulative effort.

DISCUSSION

Robotics, sensing, physical computing, and digital fabrication are all topics that have been prioritized by U.S. funding programs such as the National Science Foundation, the Department of Defense, and the Department of Education. This paper presents the start of a framework--based around the concept of tinkering--for introducing these systems into design education. Play, experimentation, iteration, and the rest of the qualities of tinkering are certainly not new to design education. Indeed, the larger value proposition is that designers are uniquely equipped to facilitate a tinkering framework to provide novel solutions to complex problems and can provide value to multi-disciplinary teams from engineering and science. As opposed to the STEM disciplines that rely on reductive research methods, designers are trained to integrate ideas and solutions at various scales to large problems that can't be well defined or easily measurable. The author shares John Maeda's optimism that innovation lies at the intersection between the STEM disciplines, art, and design:

> "Science, Technology, Engineering and Math – the STEM subjects – alone will not lead to the kind of breathtaking innovation the 21st century demands. Innovation happens when convergent thinkers, who march straight ahead towards their goal, combine forces with divergent thinkers – those who professionally wander, who are comfortable being uncomfortable, and who look for what is real. So what does it mean to add Art to turn STEM to STEAM? The

> problem solving, fearlessness, and critical thinking and making skills that I see every day across campus at the Rhode Island School of Design are the same skills that will keep our country innovating. Design creates the innovative products and solutions that will propel our economy forward, and artists ask the deep questions about humanity that reveal which way forward actually is."[13]

ACKNOWLEDGEMENTS

Arcus Animus was designed by Philip Beesley and PBAi. The workshop was conducted by Beesley and Bradley Rothenberg at Ball State University for a studio co-taught by Mahesh Daas and Joshua Vermillion. "Morpholuminescence" was designed and fabricated by Elizabeth Boone, Adam Buente, and Kyle Perry, also as part of a Ball State University studio co-taught by Mahesh Daas and Joshua Vermillion. The acrylic lattice projects were designed and fabricated by graduate students at the University of Calgary's Faculty of Environmental Design as part of the annual Taylor Seminar led by Joshua Vermillion in collaboration with seminar coordinator Jason S Johnson.

ENDNOTES

1. Stokes, J. "The Future of Cars: P2P Mesh, 4G and the Cloud". web article. WIRED.com. Retrieved from http://www.wired.com/2011/02/the-future-of-cars-p2p-mesh-4g-and-the-cloud/. 2011.

2. Igoe, T., O'Sullivan, D. *Physical Computing*. Boston, MA: Thomson, 2004.

3. Daas, M., Vermillion, J. "An Inconvenient Studio". Proceedings of the 29th Annual Conference of the Association for Computer Aided Design in Architecture (pp. 287-290). Chicago, 22-25 October 2009.

4. Buente, A., Perry, K. "Temporal Synapse". web article. PROJECTiONE. Retrieved from http://projectione.com/temporal-synapse/. 2013.

5. Thune, G., Velikov, K., O'Malley, M., Sauve, L. "The Agency of Responsive Envelopes: Interaction, Politics and Interconnected Systems". International Journal of Architectural Computing, Vol. 10, No. 3, 377-400. 2012.

6. Beesley, P. "Soil and Protoplasm: Designing the Hylozoic Ground Component System". in B. Sheil (Ed.), *Manufacturing the Bespoke: Making and Prototyping Architecture, Architectural Design Reader* (pp. 102-119). London, UK: Wiley. 2012.

7. Wilkinson, K., Petrich, M. *The Art of Tinkering*. San Francisco: Weldon Owen, 2014.

8. Martinez, S. L., Stager, G. S. *Invent to Learn: Making, Tinkering, and Engineering in the Classroom*. Constructing Modern Knowledge Press, 2013.

9. Chilton, J. *Space Grid Structures*. Oxford: Architectural Press, 2000.

10. Terzidis, K. *Algorithmic Architecture*. Oxford: Architectural Press, 2006.

11. Tibbits, S. "Design to Self-assembly". Material Computation: Higher Integration in Morphogenetic Design, Architectural Design 82, 68-73. 2012.

12. Tibbits, S., Falvello, A. "BioMolecular, Chiral and Irregular Self-Assemblies". Proceedings of the 33rd Annual Conference of the Association for Computer Aided Design in Architecture (pp. 267-268). Cambridge, ON. 24-26 October 2013.

13 Maeda, J. "STEM + Art = STEAM," The STEAM Journal: Vol. 1: Iss. 1, Article 34. 2013.