Connective Environments: Towards Interdisciplinary Pedagogical Models for Design of Networked, Interactive, Participatory Systems

Dimitris Papanikolaou

School of Architecture, National Technical University of Athens, Greece

dpapanikolaou@arch.ntua.gr

Abstract. The integration of digital media into the curricula of design and architecture schools raises important pedagogical questions as to what kind of technical skills and what kind of critical thinking should designers of future interactive systems have and how these skills and mindsets should be cultivated. This paper argues for a pedagogical model for the design of intelligent environments that sees designers and architects as engineers, humanists, system thinkers, and researchers. The paper presents these ideas through the case of a new graduate course that introduces design, prototyping, programming and evaluation of physical telepresence media for individual or collective interactions with and through the built environment. The course was offered as a core course to a dual MSc degree program in Architecture and Information Technology and it aimed to cultivate three skills: an engineering ingenuity; a critical understanding of information and computation; and a technical skillset of network-based communications for connecting people, objects, and places. The discussion concludes with teaching experiences and future directions for development.

Keywords: connective technologies, engineering design, human-computer interaction, pedagogy, interactive systems, physical computing.

1 Introduction

The deeper information and computing technology interconnects urban life the more urgent is the need for architects and designers not only to design environments that interact intelligently with people but also to understand their complex emerging behaviors. The increasing integration of digital media into the curricula of design and architecture schools raises important pedagogical questions as to what kind of technical skills and what kind of critical thinking should designers of future intelligent systems have and how these skills and mindsets should be cultivated. While courses on physical computing and interaction design were not uncommon in curricula of architecture schools two decades ago, the recent focus of the discourse on intelligent environments towards sharing, collaboration, and the internet-of-things (IoT) is

gradually shifting the way the built environment is viewed through the lens of digital media. In the paradigm of "interactive environments," the environment is regarded as an entity to interact with, that has its own agency. In the paradigm of "connective environments," the environment is regarded as a medium to connect with others, without an agency of its own. The changing view of the role of the physical environment as an interface, from that of interacting with a computer to that of interacting with a remote human, has both technical and conceptual repercussions. On a technical standpoint, interacting with remote humans requires new skills on long-range network communications, databases and backend system development in addition to traditional skills on digital electronics, physical computing and interaction design that are typical in interacting with computers. On a conceptual standpoint, interacting with humans instead of a computer, calls for a different view on the agency of technology in which the technology becomes a medium, an expressive tool or an instrument.

In this paper, I advocate for a pedagogical model for the design of intelligent environments that sees architects as engineers, humanists, system thinkers, and researchers. I present and discuss a series of assignments with corresponding samples from student work that have been developed over the course of four consecutive years (Fall 2018 - Fall 2021) as part of a new graduate course titled Connective Environments that has been offered as a core course to a dual MSc degree program in Architecture and Information Technology between the College of Art and Architecture and the College of Computing and Informatics at the University of North Carolina at Charlotte and which attracted students from disciplines of design, architecture, computing and informatics. The term “connective” refers to environments that are able to connect people, objects and places, across scales, physically or digitally without, however, enforcing such connections and without themselves providing any agency. The course introduces design, prototyping, programming and evaluation of physical telepresence media for individual or collective interactions in the age of social distancing, with and through the built environment. Pedagogically, the course aims to cultivate three broad skills: an engineering ingenuity; a critical understanding of information and computation; and a technical skillset of network-based communications for connecting people, objects, and places. Through project examples and students’ feedback, the discussion concludes with experiences and lessons learned from the pedagogical approach and future directions for development.

2 Background

2.1 From hybrid to connective environments

Since the early days of computing in the 1950s, our environment becomes increasingly more interconnected, shared and personalized. We engage with more people, we make use of the same things, and we impose our preferences on these things. Hybrid environments merge the physical and digital realms enabling people to

seamlessly interact with information and computing technology through interfaces embedded in the physical world. In Architecture, the design of such interfaces may become indistinguishable from the design of the spaces or objects that embed them, making the ontological distinction between a building and a computer a fundamental design and philosophical problem. While interactive technologies have been used to remotely control spaces with voice or gestural interfaces as early as the 1970s [1], in the last two decades, the advent of the web 2.0 [2] (enabling two-way collaboration), the IoT, and personal computing devices as well as the proliferation of open-source software and hardware, enable people to collectively and remotely control their environment and, designers, to develop and deploy networked interactive systems in ways and scales that were not possible earlier. This leap in scale has important repercussions. The more interconnected our environment becomes, the more it turns from an agency to interact with, to a mediator to connect us with others with relevant needs and desires to ours. Our focus in this paper is on design challenges that emerge when designing connective technologies that involve the digitization, communication and remanifestation of physical traces across places, contexts and time, and on their pedagogical repercussions. What skills, mindsets and pedagogical models do we need for the architects of the interconnected world?

2.2 The environment as an agent to interact with

Physical computing courses have debuted in academic curricula since nearly 30 years ago. Primarily taught as design studios, physical computing courses bring together art, design, and computer science students to collaboratively create interactive projects that enable users to kinesthetically interact with embedded technologies in their surrounding environment. Academic programs like the MIT's Media Lab [3], NYU's Interactive Telecommunications Program (ITP) [4], CMU's Computational Design program [5], Harvard GSD's Responsive Environments and Artifacts Lab (REAL) [6], CU Boulder's ATLAS institute [7], Architectural Association's Design Research Laboratory (DRL) [8], Ecole Cantonale d'Art de Lausanne (ECAL) [9], and, in K12 education, NuVu Studio [10], are some of the many academic programs worldwide that bring design and computing communities together. In parallel to academic programs, renowned research institutions in computing such as Microsoft Research (MSR) [11] and, in the past, Xerox PARC [12] have established art residency programs that bring together artists and designers with in-house computer scientists and engineers to work on conceiving and materializing interactive projects collaboratively in a studio format. Physical computing design studios are reportedly engaging courses in academic programs causing much excitement to both creators and users [13]. Depending on the context that a course on physical computing is offered, the focus of the learning objectives is different. In computer science academic programs, for example, physical computing is taught as an experiential means to teach fundamental concepts of computing in undergraduate programs [14], [15] through a learning by making constructivist approach [16] or as a necessary technical skill to develop and evaluate human-computer interfaces in human-computer interaction (HCI) programs. In Interaction Design academic programs, physical computing is used as a required technical skill to create and experience interactive systems that are

playful, without necessarily evaluating them or addressing an underlying research question [13], [17]. In K12 education, physical computing can be taught as an experiential means to teach students creativity, fundamental concepts of computing, or how real-world systems are designed and function [18]. Independently of the context, a common trait in all flavors of physical computing courses is their focus on the (usually localized) human-computer relationship.

2.3 The environment as a medium to communicate with

While the fields of physical computing and interaction design focus on the relationship between a human and a computing entity, the field of physical telepresence media focuses on the relationship between humans through a computing entity. Physical telepresence media allow two, or more, remote persons to mutually experience each other's presence by mapping and physicalizing their interactions with objects either synchronously or asynchronously. In the relevant literature, examples of telepresence media include co-creative canvas boards allowing users to collectively experience each other's scribbles [19], mutual kinesthetic tactile interactions with force feedback [20], [21], interaction techniques based on shape transmission [22], platforms for physical manifestation of object-based crowd interactions [23], interactive dance visualizations for group interaction [24], furniture for remote interpersonal awareness [25], video games [26], collaborative musical interfaces [27]–[29], collaborative drawing [30], collaborative building [31], collaborative animated storytelling [32], [33], and collaborative kinesthetic games that use motion as a means to encourage collaboration [34]–[36]. The growing research field on telepresence media shares much in common with the discipline of Architecture. Architectural space is not only a passive container for human activities. It is also a medium that conditions our social perception of others by registering traces of human activities on it and by allowing us to experience these traces later. For example, when someone changes the physical state of a space or the physical state or location of objects inside a space, then experiencing these changes later signifies the existence and possibly the activities of the person who made these changes. Therefore, curating, moderating, or enhancing these registering capabilities with the aid of technology is not only a logical extension but also a mandate of the practice and theory of architecture. Such twist requires a unique interaction design approach that expands its focus from the objects that exist inside a space to include the building elements and architectural surfaces that contain and shape the space.

2.4 Space as a computational counterpart in Architecture

Despite the relatively long history of computing in Architecture, design of intelligent environments has done little more than appending computing technology to architectural space and assigning anthropomorphic behaviors to it as an afterthought, often without critical reasoning on how computing and architecture may affect each other. Today, 40 years after the MIT Architecture Machine Group created the Media

Room [1], we still point and talk to buildings, walls, and furniture as if they have mind, soul, and composure [37], [38]. The peculiar relationship of computing to architecture is illustrated in the own words of some of its most important thinkers. In a seminal text for the field of ubiquitous computing, Mark Weiser, acclaimed that the "most profound technologies are those that disappear"[39]. In a discussion with architecture students at the Harvard GSD, Nicholas Negroponte, the founder of the MIT Media Lab, compared the future of information acquisition to the act of "swallowing a pill" [40]. Likewise, Mike Kuniavsky, head of design at Xerox PARC, described the coming age of ubiquitous computing as "magic"[41]. The difficulty in comprehending formalistically ubiquitous computing is also reflected in the increasing role of metaphors in human-computer interaction (HCI) courses as a means to understand unfamiliar concepts in terms of familiar ones [42]. Intelligent buildings, today, are metaphorically seen as "friends," "companions," or "assistants" with a meticulous effort to conceal any visible evidence of technology from their clean surfaces [43]. This attitude is in contrast to the prevailing tendency in architecture design studios to formalistically express function, materials, and tectonic clarity [44]. There are two reasons, I argue, for this. The first is that the medium with which information is manifested (electric voltage) and the speed with which it is processed by a computer are invisible and imperceptibly fast. As a consequence, the actual workings of computation are experientially unnoticeable and difficult to express architecturally. The second reason is that, when architecture students learn Interactive Computing, they are trained in high-level programming languages that further mask the machine code that drives the mechanics of computation. The invisibility of information and computation and the decontextualization of software from hardware, miss the opportunity to use computation as a medium to drive design and architectural discourse, and mask the potential contribution that architecture can make in the field of computing.

2.5 Designing (for) connective environments

Designing for connective environments requires an interdisciplinary approach that integrates the analytic and integrative mindsets of engineering and social sciences with the creative and critical mindsets of humanities. Such approach fits within the broader disciplines of Design and Architecture yet it is distinct from the mindsets of the computational architect, the digital artist and the HCI researcher as these are found today in interaction design courses in Architecture, Digital Arts, Design Studies, or Computer Science programs. I summarize five design characteristics of, or theses for, connective environments that distinct them from the broader field of hybrid environments. 1) When scale of design inquiry expands from that of the object to that of the architectural or urban space, the spatial qualities, constraints and resources are instrumental not only in framing the ways with which the interactants can interact with each other but also in contextualizing the reasons of their interactions. 2) When scope of design inquiry expands from that of the individual to that of the social group, the goals, options, and decisions of each individual, as well as the potential synergies or conflicts between them, are instrumental not only in steering the emerging behavior of the system but also in suggesting possible ways with which the designer can

anticipate and evaluate those behaviors. 3) When scope of design inquiry expands from that of the individual to that of the social group, the design problem of determining a system goal (in the cybernetic sense) becomes that of determining an equilibrium between the different goals of each member of the group that requires a closed-systems-oriented approach. 4) When the environment turns into a medium that conditions communication, the ontology of the boundary between the physical and the digital becomes a major topic of inquiry both for the designer's intention and for the user's perception. Spaces that act as connective media ought to be different than passive spaces that contain connective media. 5) When the environment turns into a medium that conditions communication, its computing and informational capabilities do not necessarily need to be digital, and therefore invisible, but they can also be analog, mechanical, visible and tangible, and therefore of direct architectural importance. For example, a digital signal may have first been sampled, modulated or even transformed mechanically.

2.6 Educating the architect of the interconnected world

Buckminster Fuller argued that a designer is an emerging synthesis of artist, inventor, mechanic, objective economist and evolutionary strategist [45]. The pedagogical approach presented here aims to equip architects of connective environments with similar mindsets, not only in a theoretical context but also in a practical one. There are three main learning objectives that the presented course has. 1) Ability to design enabling technologies to connect humans, objects and places across scales that can be digital or analog. 2) Ability to induce by design synergistic behavior between humans that can be collaborative, playful or strategic. 3) Ability to develop appropriate methods to anticipate emerging synergistic behaviors that can be empirical or analytical. Given that the course is primarily design-oriented, the degree and the manner in which the above skills are developed varies, depending on the skills and background of each participating student as well as on the particular focus that the course may have each year it is offered. In summary, this paper makes five key contributions to the literature on design education for hybrid or interactive environments:

- It identifies a distinct educational subfield of connective environments within the broader field of interactive or hybrid environments.
- It proposes a course design model for studios on interactive technologies that views the environment as a connective medium instead of a conversant agent.
- It proposes a pedagogical model that views designers and architects as engineers, humanists, system thinkers, and researchers.
- It presents assignment descriptions, teaching methods and examples from student projects through a detailed course case study.
- It presents a discussion on the motivation, goals and results from the course design and outcome.

These contributions can be useful for instructors who teach or research in areas related to connected environments to build upon. In the rest of the paper, I present the case study of the course, illustrating how the above concepts are synthesized in practice.

3 Materials and Methods

3.1 Course Description

The course Connective Environments provides foundational skills in designing, prototyping, and programming interactive physical telepresence systems, analog or digital, with an emphasis on network-based communications. The course's premise opens up with the following prompt: "We perceive others implicitly, through the traces of their interactions with physical space near us – what if our urban environments could remotely mediate these traces, allowing spatially unrelated people to experience each other's presence collectively?" Through a project-based approach, students explore how information technology and material or physical constraints may inspire the conception of novel affordances, and how these affordances drive design decisions for closing the loop between sensing and (re)acting. In addition, students learn how to conceptualize, present, and critique designs in a studio format; how to develop, program, and assess interactive systems; how to review state of the art literature in Human-Computer Interaction (HCI), Human-Building Interaction (HBI), and Tangible and Embedded/Embodied Interaction (TEI), and how to write a conference paper. Topics include physical telepresence, shape changing Interfaces, distributed systems, tangible, embodied, and embedded interfaces, and physical/mechanical computing. Due to the breadth of topics and required skills, the course does not cover in depth each technical skill. Entering students are expected to be self-motivated, have basic skills in 3D modeling and fabrication, and be familiar with the Arduino board and programming language (there are plenty tutorials online that students are encouraged to explore before the first day of classes). The course combines lectures, guest talks from experts in the field, lab assignments, readings, and student presentations and discussions. Emphasis is on critical thinking and tradeoffs between technical complexity, end goals, design decisions, functionality, and quality of craft.

The course is organized into four thematic parts that build on each other and combine engineering design, theory of information, theory of computation, physical computing, and network-based communications. The first part introduces engineering design; the second part covers mechanical logic; the third part covers physical interaction; and the last part covers remote interaction. The above thematic parts are integrated into a series of assignments that progress from analog and mechanical to digital and electronic. At the end of the course, students are asked to write a paper that describes their project using the ACM template. Final paper must be of publishable quality at the level of the ACM TEI (Tangible, Embedded and Embodied Interactions)

[46], ACM DIS (Designing Interactive Systems) [47], or ACADIA [48] conferences (short paper or project category).

3.2 Classroom setting

During the period that the course was taught, it had 8-12 participants, depending on the year that it was offered, and the assignments were carried through in teams of two students. The course took place in two classroom settings. In some class meetings, students and instructor met in a seminar setting (round table) and each team presented their progress or the instructor presented a lecture or an open discussion took place. Other times, students and instructor met in a laboratory setting that was equipped with soldering stations, electronics, 3D printing machines and several computer stations, usually when the instructor demonstrated a new lab skill such as soldering. In the lab, students and instructor were free to sit either around a conference table or in one of the workstations. A large white board served as pivotal place to conceive, sketch, and brainstorm ideas collectively.

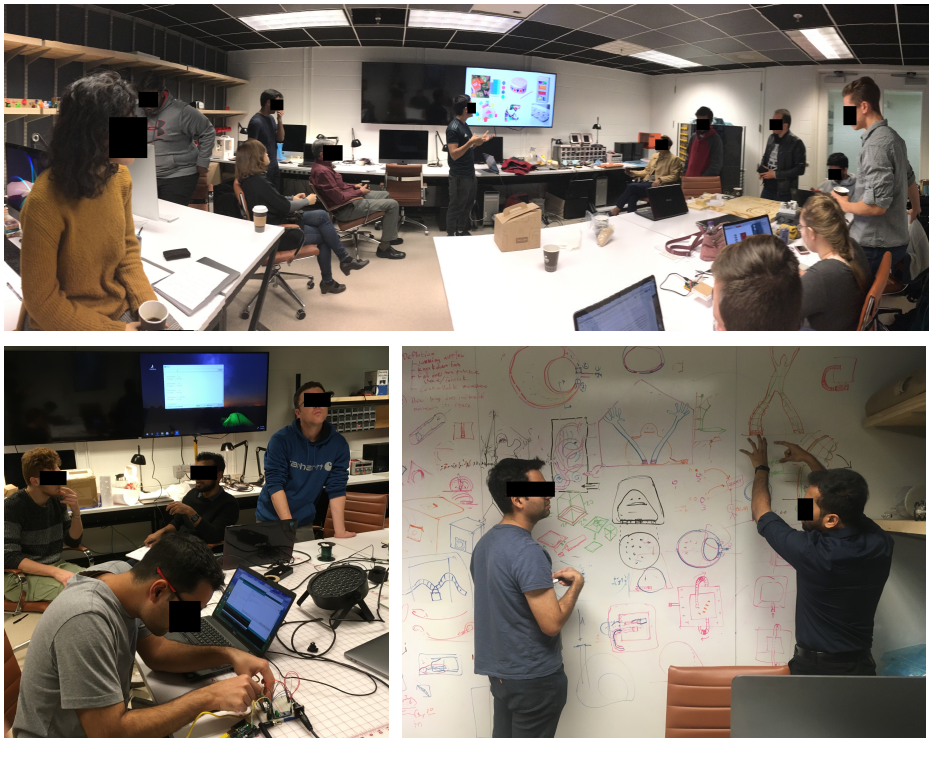

Figure 1: Top: Classroom setting in the lab during reviews. Bottom left: Work session in the lab. Bottom right: Brainstorming session of students in front of the lab's whiteboard.

Occasionally, an invited speaker was brought to speak in the class and usually that happened through teleconferencing in either of the two classroom settings. During in-class reviews (pin-ups), at the end of each assignment or during mid-reviews, students demonstrated their projects and discussion took place on the challenges they confronted. Usually, guest critics, instructor, and other students contributed collectively with feedback. Cultivating a culture of sharing work was essential and the task of having to regularly document work in their websites was instrumental in that. For example, at the beginning of each class, students shared their progress by opening and showing their blog posts or prototypes to the rest of their class in one of the two classrooms settings discussed above.

3.3 Assignments

The course is organized in a series of assignments, the number of which varies based on the year the course is offered. The first assignments are analog and mechanical and include an assignment of reverse-engineering and reinventing a mechanical toy, an assignment of designing a mechanical computing device and an assignment of conceiving an object that can autographically register and communicate traces of human activity [49]. The last assignments teach students physical computing and network communications and include the design of an interactive object and the design of a pair of connected objects for remote interactions. Not all of the assignments described below were assigned each year the course was offered.

3.3.1 Reverse-engineer, design, and make a functional mechanical toy

This assignment asks students to reverse engineer a functional mechanical object, usually a toy car, and then design and make their own version of it. The assignment does not require particular skills other than basic knowledge of a CAD 3D modeling software. The topic of a toy car is chosen because of its balance between simplicity and complexity. Through this assignment students expose themselves to the nuances of engineering design, interdependencies, functionality, team work, and to an understanding that not all parts in a mechanical assembly are equally important. This assignment consists of two parts: reverse-engineer a mechanical toy; and design and make a mechanical toy.

In the first part of the assignment, students learn how to reverse engineer a mechanical toy car addressing the following. 1) Take a photo of the disassembled toy and of each part, list and label parts, and explain what each part does, and how they function to transfer energy. 2) Explain narratively how the mechanism works and transfers energy from the point where a finger winds up the toy to the to the point where the mechanically movable parts propel/move the toy on the table. 3) Create an assembly graph of the toy in which each node is a part and each link is a structural connection. 4) Indicate a valid assembly sequence in the assembly graph and in the graph's adjacency matrix. 5) Explain if there are any steps in the assembly sequence that are more difficult to perform than others and why (for example, installing one part with many concurrent links to other parts may be complicated and thus difficult).

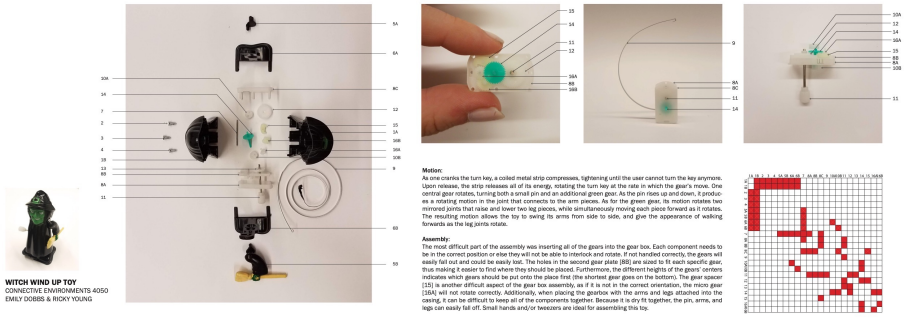

Figure 2: Reverse engineering assignment of a mechanically actuated toy car. Students must disassemble an existing mechanically actuated toy, list all the parts and diagram through graphs their assembly sequence and their function.

In the second part of the assignment, students must collaboratively design, fabricate, and assemble a mechanically actuated toy car, consisting of two functional parts, a chassis and a motor, while meeting specific performance criteria. Students work in teams of two and each teammate is responsible for developing one of the two parts while negotiating with the other teammate about how their parts will interlock, synergize, and transfer energy. Toys must be designed for laser cutting fabrication and easy manual assembly with no adhesives or fasteners. Each team is allowed to use at most one kind of off-the-shelf mechanical components such as pulley-wheels (to use as a flywheel), washer disks, ball-bearings, metal rods, etc.

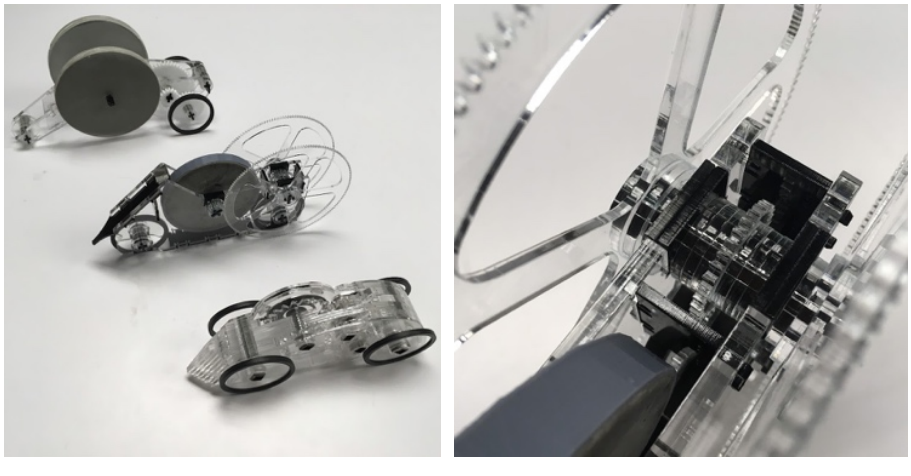

Figure 3: Engineering design assignment. Students get a hands-on exposure to engineering design challenges by collectively designing, fabricating and assembling a mechanically actuated toy car.

The assignment is organized into five meetings. The first meeting covers team-building, concept design, an introduction to gears and mechanisms, flexure joints, and Design for Assembly (DFA) principles. The second meeting introduces parametric modeling with Rhino, Grasshopper, and a gear-generator script, through a guided process. Students refine their design and calculate the energy that their car can store, assuming no friction from materials. At the third meeting, teams are required to create a parametric model of the car in which they can change parameters while calculating performance metrics, and discuss their solution. The fourth meeting is dedicated on first functional prototypes. The fifth meeting is final reviews and demonstration.

Evaluation criteria include the functionality of the toy, its assemblability, and its overall aesthetics. Students are asked to consider their projects as finished products, and to invent solutions compatible with the available materials and fabrication techniques. In addition, teams must address a number of questions such as "how much energy can your car store and how far can it travel?" To succeed, students must exemplify skills on engineering design, mechanical assemblies, ingenuity, and team working. In particular, they develop their designs based on what they can model, fabricate, and assemble. During the final reviews, teams may compete based on how far their toy cars can go (although in most cases, building the toy is sufficiently challenging).

3.3.2 Design and make a mechanical computing object

This assignment asks students to design a mechanical computing object that can perform a simple logical operation, such as adding two binary numbers, on two or more inputs using energy from its surrounding environment other than electricity. The assignment requires students to possess the engineering design skills that are developed with the previous assignment. For most designers, information and computation is an abstract invisible process that converts digital inputs to digital outputs. In contrast, designers learn to think in terms of visible, physical and tactile borders with which they can define space. The purpose of the assignment is to develop a physical, kinesthetic intuition of what information and computation are before delving into digital electronics. Through this assignment, students approach computation and information in the most general sense, on one hand, computation as the "process of storing, transmitting, and transforming information from one form to another", and on the other hand, information as the process of giving form to communicate knowledge.

The assignment is organized into five steps. The first step involves a introduction to concepts of computer logic and logic circuits from the book The Hidden Language of Computer Hardware and Software by Charles Petzold [50]. The second step involves exploration and experimentation with existing mechanically computing artifacts used as case studies including the Turing Trains [51], [52], the Digi-Comp 1 [53] and 2, the Turing Tumble [54], [55], and Dr. Nim [56] through computer simulations or physical demonstrations. Students are prompted to read and do the tutorials for creating relays, logic gates, and flip-flops for one or more of the selected case studies. The third step involves the design of a switch and a relay, that students develop by any physical medium, energy source, and energy flow they want. Considered forms of energy flow included mechanical movement, fluidic movement,

wind flow, vehicle traffic, electrical current, sound, and pressure. In the fourth step, students design the three basic logic gates (AND, OR, NOT) with the relay they developed in step 3. Then, they create a functional prototype for each one of the three gates in the fablab. They can use any material of fabrication method. In the fifth step, students create a mechanically computing machine. For example, they can create a binary adding or subtracting machine, or a machine that compares, divides or multiplies numbers or it performs logical operations, using marbles.

In designing their relays and gates, teams are asked to consider questions such as "How does your relay harvest, store, and release energy in order to change another relay? Does the output of one gate have the same format as the input? How can multiple relays connect in a cascade such that when one relay changes its state, it triggers its connecting relay to change its state as well?" Developing an empirical understanding of these concepts is an important prerequisite for learning concepts and technologies of physical computing that other assignments introduce, more critically.

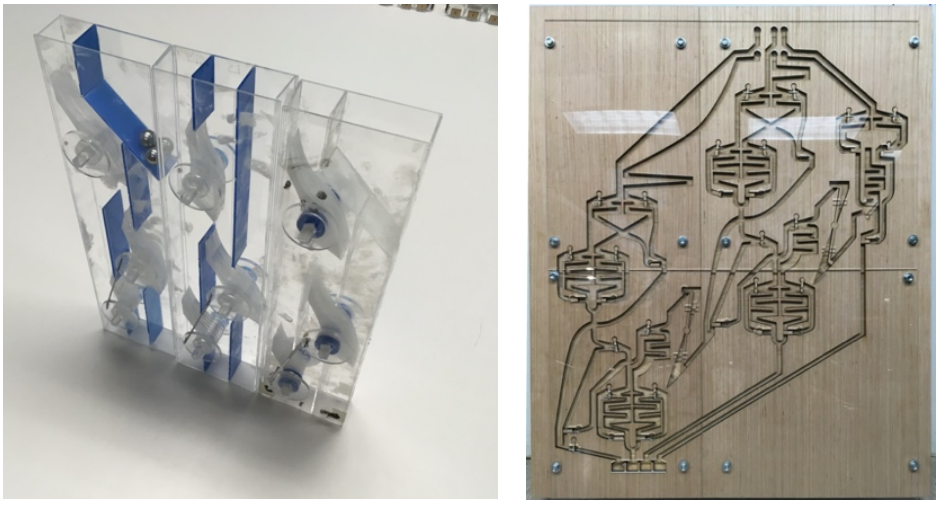

Figure 4: Student projects for fully functional mechanical binary adding machines. The machines can add two 3-bit binary numbers and consist of a series of logic gates. The logic gates consist of flip-flops that change their binary state as marbles pass through.

3.3.3 Design an artifact that autographically registers and manifests traces

This assignment asks students to design a physical artifact that autographically registers and manifests traces of human activity in an analog way. The purpose of this assignment is to introduce students to concepts of mediated interaction and communication from a critical standpoint without delving into technical skills. The assignment may or may not use mechanical computation skills in the way these were developed in the previous assignment. Because this assignment does not require

access to the lab space, it was suitable for remote learning during the COVID-19 lockdown.

The assignment consists of two parts. In the first part, students find and analyze an object that registers in its state the traces of the interactions that people have with it. The analysis must address the following through drawings, diagrams, sketches, or photos: 1) the object and its various states as humans interact with it; 2) the types of interactions that the object affords; 3) how the traces of the interactions are registered in the state of the object and what the interpretation of these traces can reveal about the person that interacted with the object (what did the person do? Who was the person? When did the person interact with the object?); 4) the context through which these interactions occur (what knowledge does someone need to have in order to infer something by observing the object?); 5) a context through which the interpretation of these traces might be meaningful to someone.

In the second part of the assignment, students conceive and design an imaginative object that registers and manifests traces of its interaction with its users, based on the analysis they did in the first part of the assignment. In doing so, students must identify two or more personas who are in a meaningful relationship and describe what interactions they have through this object. Students must contextualize their analysis and design through relevant readings in autographic visualization, data physicalization, and theory of affordances.

3.3.4 Make a software interface for remote co-creation

This assignment asks students to design and develop a web-based system for visual telepresence through which two or more online users can connect and interact remotely. The purpose of this assignment is to introduce the necessary technical skills for allowing multiple individuals to interact in real time through the internet. The assignment has no prerequisites other than a basic understanding of programming with any language. The system must consist of a front-end side (the browser-based interactive visual application that the user will see and interact with), and a back-end side (the server that will handle interactions between users). Communications between users and server must be bidirectional and real-time, similar to the real-time messages that users exchange in a web-based chat application. The technical objective of the assignment is to introduce fundamentals of network-based communications (backend and front-end) without the complexities of physical computing with microcontrollers. Students build everything from scratch using P5.js [57], a JavaScript version of the Processing language. Students use primarily a display screen and speakers as outputs and they use their camera, keyboard, or microphone as inputs. Yet, the scale is not limited to desktop applications as students can imagine their project as a projection mapping on architectural surfaces of large scales. Because this assignment does not require access to the lab space, it was suitable for remote learning during the COVID-19 lockdown as a final project.

Students spend the first two weeks creating a chat application and an interactive canvas sketch application in order to learn the technical concepts and familiarize themselves with the necessary toolchain. Next, they develop their own design by modifying the front end and back ends accordingly. The assignment is organized into five steps. In the first step, students familiarize with the P5.js programming language

by exploring links that are provided and by doing a tutorial on an interactive drawing canvas application. The tutorial teaches how to setup a server and a front end. In the second step, students learn about servers and they develop their own server using Node.js, a programming framework for developing servers using JavaScript with only few lines of code [58]. In the third step, students learn about real-time bidirectional communications using websockets and socket.io [59]. Socket.IO is a JavaScript framework consisting of a client-side script and a server-side script that enables real-time, bidirectional and event-based communication between clients and server. Socket.IO is composed of two parts: a socket.io server that integrates with a Node.JS HTTP Server; and a socket.io client library that loads on the browser side. In the fourth step, students learn about front end and they design their own front-end implementation using P5.js and HTML/CSS. Finally, in the fifth step, students integrate all the above into a project.

3.3.5 Make Hardware Interfaces for Physical Telepresence

This assignment, usually provided as a final project, asks students to design and prototype a network of web-connected interactive objects of architectural quality that can experientially connect remotely located individuals. The connected interactive objects must be able to capture, exchange, and manifest their interactions with humans. This assignment integrates skills, concepts, and critical thinking that students learned and developed in all previous assignments. From a technical standpoint, the physical telepresence assignment is an extension of the visual telepresence assignment. While in the visual telepresence assignment students learn how to setup and program a client-server system that allows two or more web clients to communicate in real-time using web sockets, in the physical telepresence assignment, students extend this system by integrating sensors and actuators to interact with the physical environment, by prototyping and programming a microcontroller (hardware) as a client, and by connecting the microcontroller to the existing backend system they developed in the previous assignment. While there are many communication options and protocols, the assignment focuses on connecting devices through the internet using WiFi.

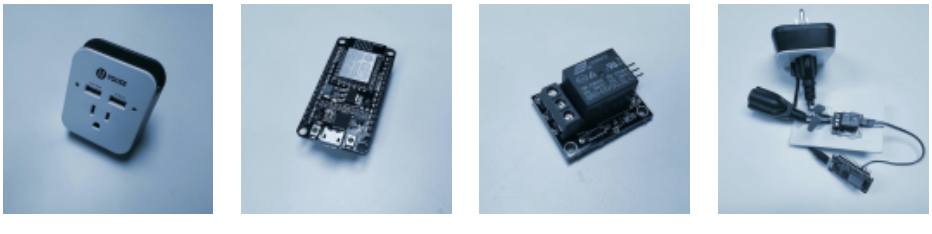

Figure 5: Hardware components for building a network-based application. Power adaptor with USB support; Node MCU ESP8266; 5V Digital Relay; Final assembly.

In the class, we use the ESP32 NodeMCU microcontroller, (or its previous version, the ESP8266 NodeMCU), which is Arduino-compatible. The ESP NodeMCU has been designed to offer a practical and cost-effective solution for makers seeking to add Wi-Fi connectivity to their projects with minimal previous experience in networking. All Arduino-compatible microcontroller boards can be programmed through the Arduino IDE (occasionally requiring a library to be installed). This makes programming easy because whatever you learn in one Arduino board can be applied to all others. Students use two Arduino Libraries: the ESP8266WiFi.h which allows the microcontrollers to connect to Wi-Fi, and the SocketIOClient.h which lets them communicate with the server through a WebSocket protocol.

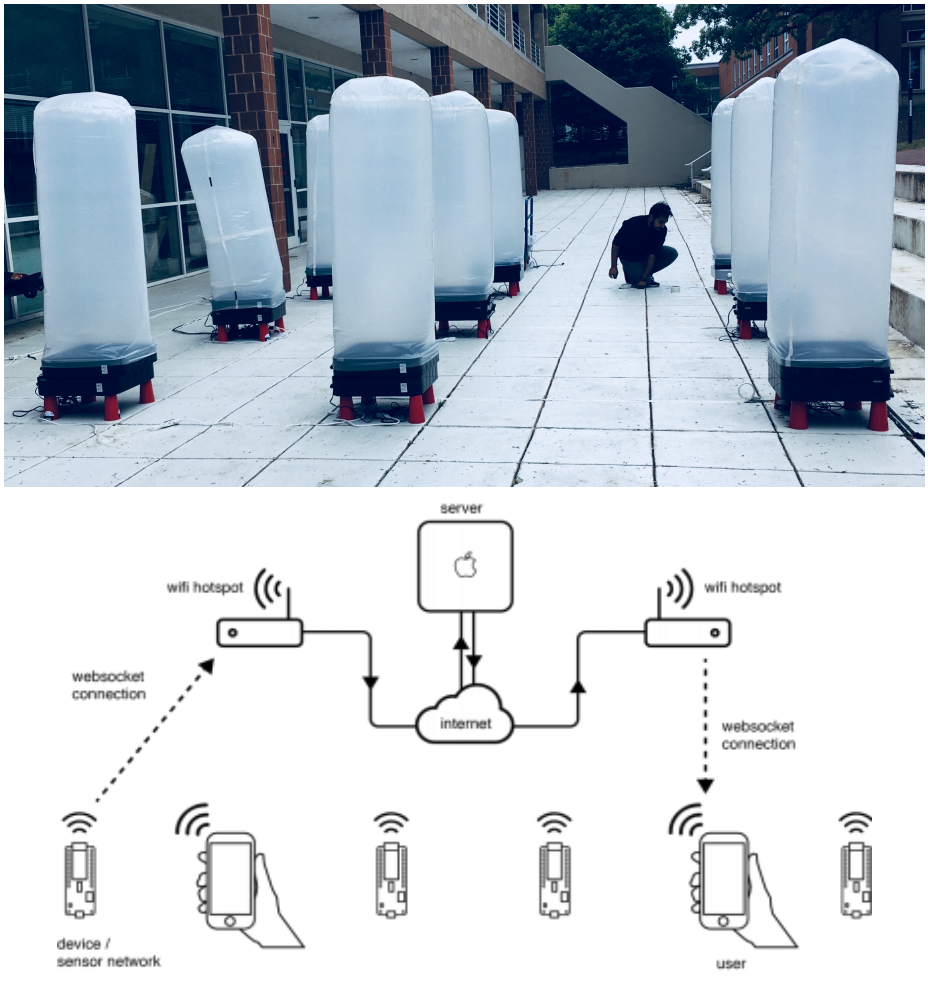

Figure 6: Top: Example of a final project. Students developed a networked of web-connected interactive inflatables that users could actuate using their mobile phones. Bottom: Diagram of a cyber-physical network-based application.

To connect their devices through the internet, students need to use a client-server scheme: clients send requests to the server and the server responds back to the clients' requests. The NodeMCU board can be programmed and serve both as a client and as a server. If students program all devices as clients, they will need to develop a server program that will run locally in a computer, using Node.JS. The devices will first connect to the internet through a local WiFi network (such as the campus network), and then they will access the server by using the IP address of the computer that hosts it. If students program one device as a server and the other devices as clients then their client devices will need to know their server device's IP address in order to access it. In such case, they will program their server device using the Arduino programming language and a dedicated WiFi library. Students must carefully explore each of the above cases in appropriate readings and tutorials, and decide which of the two strategies works better for them. While students are free to choose any of the two networking directions, they are encouraged to program their microcontrollers as clients and use their own computer or a cloud-based service as a server.

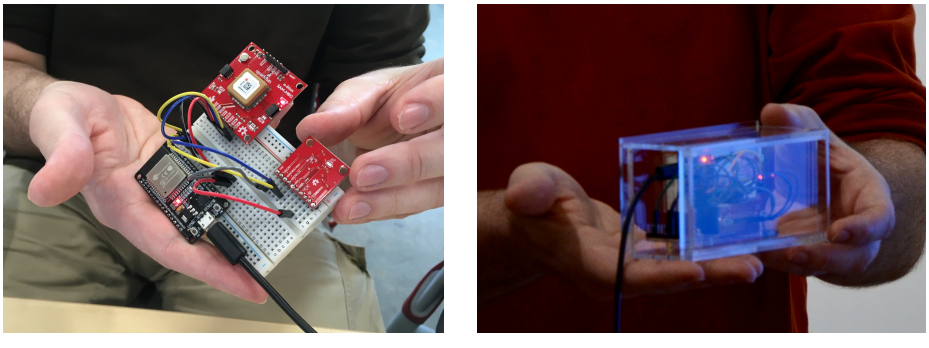

Figure 7: Example of a final project. Students developed a set of two wirelessly connected compasses equipped with GPS modules that point towards the direction of the geographic location of the connected person.

In addition to the microcontroller boards, students need USB data/power cables to connect the board to their computer as well as electronic parts such as jumper wires, breadboards, resistors, push-buttons, LED lights, sensors, or various forms of actuators. Another possibility is to use their microcontrollers to control relays which in turn will control the power supply to other electrical devices such as fans, lights, or home appliances. What parts students will need depends solely on what their project will do and how they want their users to interact with it. Students must do their own research to decide what these parts will be. Although the lectures and lab sessions focused on internet-connected microcontroller-based applications, students were free to expand their projects to other domains. For example, a team developed a project that was based on analog electronics by hacking two old CRT televisions in order to connect two individuals through them.

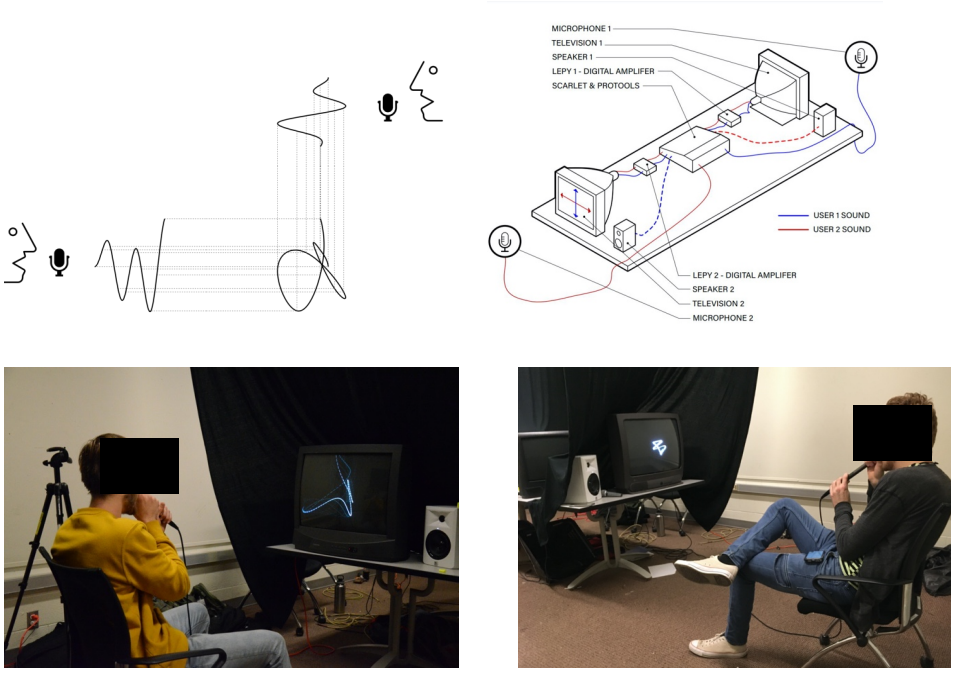

Figure 8: Example of a final project. Students modified two old CRT televisions, routing two microphones to two different axes of each television. When one individual speaks into the microphone, they control the X-axis, while the opposite individual speaks and controls the Y-axis. The vocals of the two participants create a visual representation of the two user's interactions with one another. Left: Construction of a Lissajous curve from two audio signal waveforms. Right: System design: The sound input from user 1 microphone controls the horizontal dimension of the electron beam. The sound input from user 2 microphone controls the vertical dimension of the electron beam. Collectively, the inputs from the two microphones formulate a composite shape. Bottom: Users interacting through the device cocreating composite oscillographic scribbles with their voices during the user study.

3.4 Blog documentation, peer learning, and technical paper

In addition to weekly readings and advancing their projects, students are asked to review, present, and critique papers (usually from relevant ACM SIG conferences), document their research progress weekly in blogs (usually at least one post per week per team), and write a final paper on their final project following the ACM format. Documenting research progress in blogs is essential because it prompts students to consider themselves as knowledge producers for a community that they relate to through their work. Moreover, authoring blogs helped presentation and communication skills, reinforced peer learning, and it provided a constructive transition to the task of authoring a final paper of publishable quality.

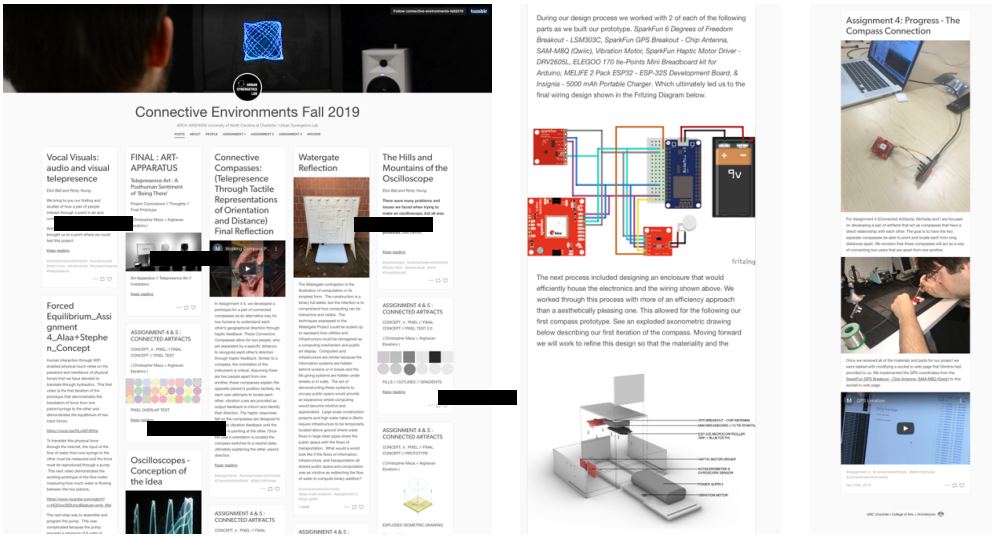

Figure 9: Examples of student weekly blog entries documenting project progress. The class's blog provided a central repository of knowledge exchange and it boosted competition between student teams.

4 Discussion

4.1 General reflections

This paper presented the pedagogical approach of a new course that teaches students how to conceive, critique, design, prototype, and program physical telepresence interfaces and experiences for connective environments, that was initially taught at the University of North Carolina at Charlotte between the College of Art and Architecture and the College of Computing and Informatics during F2018-F2021. The course is under continuous evolution and lessons from each year are being used to inform the curriculum of the course for the following years. In this section, I share some reflections on my experiences with students.

While the course starts with an engineering design bootcamp that focuses on mechanical assemblies, the skills acquired during this introductory phase are essential for team building and for acquiring a hands-on experience on how to design systems of interrelated parts that are useful for the next thematic sections of the course. The engineering design challenge motivates students by providing a design goal with clear objectives and aesthetically stimulating outcomes. This, in turn, strengthens team building skills while makes students to think of system design issues of interdependencies that are common in many projects.

The thematic sections on information theory and mechanical logic (weeks 4-6) were instrumental in shaping the mindset of students of various disciplinary

backgrounds about computation in directions that go beyond those of digital electronics. In a survey at the end of the class, students in one year that the course was taught were asked: *"In the first part of the class, we focus on computation and information theory not digitally but mechanically. In what ways does this approach affect or challenge your understanding or existing preconceptions of computation, specifically in relation to architecture?"* In the words of the students *"Specifically, being a CS student, the concept of computing using physical media has not only been a complete 180 in terms of approach but also has made me broaden my understanding of computation capabilities overall"* and "*Computation is always thought to be this mystic and difficult field to get into. The Mechanical computation, certainly helped in demystifying it and helped in understanding that static structure can do complex computations"* While this thematic section in the course has been transformative, I have also realized that three weeks is too brief, both in terms of theory and in terms of assignment length, for students to take a full advantage of it. In a future version, mechanical logic may be taught as a standalone prerequisite course of Connective Environments.

In some of the years that the course was offered, the focus centered on sensors, actuators and object-based interactions. In other years, the focus of the course centered more heavily on (tele)communication. For example, instead of working with sensors and actuators, students worked with digitally controlled relays to control larger devices whereas mobile personal devices were used as inputs. While the focus of the second half of the course is on digital electronics, students are not required to work with digital electronics for their final project. For example, some students chose to work with analog electronics for their final projects.

The lab culture was an essential part of the course. Most students visited the lab frequently to work on their projects during hours that were outside the normal meeting times of the course. The space of the lab and the available resources reinforced a collaborative spirit as students from different disciplinary backgrounds shared their different skills by teaching each other. This exchange of knowledge was instrumental for the success of the course.

Most students who enrolled in the course, especially those coming from design backgrounds, had no previous experience with academic writing and therefore the notion of what a "contribution" might be in designing a human-computer or human-human interface was unfamiliar to them. As part of the course, student teams had to review selected papers in relevant literature in HCI/TEI/DIS fields and discuss in the class the novelties, design intentions, and contributions of these papers. This process helped them seek to identify similar values in their own work. In addition, the task of having to write about their design project in the same (yet more concise) format as the papers they reviewed, helped them to prioritize these values and communicate them in a coherent way. While the process of writing about their own work was difficult, the structure of the existing templates from selected conferences provided a guideline to students which caused a sense of pride and ownership of the intellectual values of their work that was new to their prior educational experiences.

4.2 Challenges and limitations

Developing and teaching a course with so many technical and conceptual requirements had also its challenges and limitations. First of all, the learning curve for most students was steep and students who enrolled in the class with limited technical backgrounds felt more challenged than those with existing programming skills. At the same time, because the skills that the course covered were so varied, students who were more skilled in software skills felt more challenged in the engineering design assignments whereas students who were more skilled with their hands excelled in those assignments but felt more challenged in the programming aspects. This provided an interesting balance to a group of students who enrolled in the class from diverse backgrounds (Architecture and CS/IT). More specifically, a challenge for a course like this is that it requires a computational programming literacy that can allow students to develop their projects. This means that in the context of a undergraduate curriculum, such course can be offered as a special topics course in year 3 or 4 of an undergraduate program and it can ideally be based on a computational programming foundation that can happen in years 1 or 2 of an undergraduate program. Another challenge is that the course requires access to both an electronics laboratory and a digital fabrication shop in which students can develop their projects.

4.3 Educating on connective environments in a post-COVID world

Most of the presented assignments were designed for a face-to-face setup. During F2020, the course was taught remotely due to COVID19 pandemic restrictions and some of the assignments had to be redesigned. Specifically, in that year the syllabus was redesigned and the mechanical assembly and reverse engineering assignments were replaced with the design of an artifact for communicating traces. Also, the course focused on web-communications and students focused more on learning programming and on designing an interactive web-based application and on speculating its application in the real world. While COVID19 gave admittedly a new twist in the class which students appreciated despite the remote teaching difficulties (in the words of a student: "*The focus on network concepts rather than physical concepts seemed fitting for the [new] context in which the class was situated."*), it also challenged one of the main objectives of the course which was the critical inquiry into the physical environment as a medium for communication (in the words of a student: "*The course was previously focused on physical presence for the final project and the shift to online put an end to that and I feel as though the project lost the focus that it may have had in the past due to the need to accommodate an online review"*). At the same time, the COVID19 experience gave a stronger focus on the telepresence aspects in a "personal" manner as students found a motivation in the pandemic to develop tools for physical telepresence since themselves could not be present (in the words of a student: *"This course was rather challenging at times due to the subject matter, but I feel that this challenge was a helpful motivator. The subject matter was very interesting and it was one of the few courses that I took throughout my time in school that felt like a 21st century class."*). While it is hard to say that any modification in the course curriculum during the pandemic's remote restrictions will continue in the post-pandemic

era, the COVID19 experience gave certainly more ground to the theoretical premise of the course regarding technologies for connective environments.

4.4 Future work

The last year that the course was offered (F2021) gave a stronger focus on inflatable structures as an accessible, affordable, and easy-to-work-with medium to address physical telepresence interactions in large scales. We discovered, however, that the larger scale of the prototypes, even though they were relatively easy to make (we used double sided reinforced tape), took substantial time to develop as students had to train themselves in the techniques of inflatable structures. The "reverse-engineer, design, and make a functional mechanical toy" assignment was also changed to a simpler "reverse-engineer" assignment to allow more time for students in the to work with inflatables. Yet, we found that the omission of the "design, and make a functional mechanical toy" part of the assignment made the "design and make a mechanical computing object" assignment to be more challenging because students did not develop the necessary mechanical assembly skills. This suggests that a course on physical telepresence interaction based on inflatable structures has substantial skills that must be built that could take the form of a new specialized course. This is a direction that I plan to explore further in the future. At the same time, in each year, I found that the mechanical computation assignment even though presented some promising ideas, it never had the necessary time to deep more creatively in them. Thus, another future version of the course could focus entirely on physical remote interactions through mechanical computation means. Therefore, in the future, several different more specialized forms of this course can be developed that will allow deeper exploration from the students and more opportunities for research writing.

4.5 Conclusion

In summary, this paper made five contributions that can be valuable for instructors and researchers in areas related to intelligent, interactive, or networked environments to build on. It identified a distinct educational subfield of connective environments within the broader field of interactive or hybrid environments. It proposed a course design model on interactive technologies that views the environment as a connective medium as opposed to a conversant agent. It proposed a pedagogical model that views architects as engineers, humanists, system thinkers, and researchers. It presented assignment descriptions, teaching methods and examples from student projects through a detailed course case study. And it presented a discussion on the motivation, goals and results from the course design and outcome and an overview on future directions of the work.

Acknowledgements. The work of the following student teams is shown (listed in order of appearance): Seyedehsan Aboutorabi, Gavin Reeb and Saquib Sarwar; Nicklaus Williams and Sri Yeswanth Tadimalla; Manoj Deshpande and Saquib Sarwar; Nicholas Rawlings; Elliot Ball and Richard Young. The Teaching Assistant for the course during F2019 was Atefeh Mahdavi Goloujeh.

CRediT author statement. Dimitris Papanikolaou: Conceptualization, Methodology, Validation, Investigation, Resources, Writing - Original Draft, Writing - Review & Editing, Supervision, Project administration.

References

1. R. A. Bolt, "'Put-that-there': Voice and gesture at the graphics interface," ACM SIGGRAPH Computer Graphics, vol. 14, no. 3, pp. 262–270, 1980, doi: 10.1145/965105.807503.
2. "Web Content Accessibility Guidelines (WCAG) 2.0." Accessed: Nov. 13, 2023. [Online]. Available: https://www.w3.org/TR/WCAG20/
3. "MIT Media Lab Homepage," MIT Media Lab. Accessed: Nov. 12, 2023. [Online]. Available: https://www.media.mit.edu/
4. "NYU Interactive Telecommunications Program." Accessed: Nov. 12, 2023. [Online]. Available: https://tisch.nyu.edu/itp
5. "CMU Computational Design Program Homepage," CMU School of Architecture. Accessed: Nov. 12, 2023. [Online]. Available: https://soa.cmu.edu/computational-design
6. "REAL Harvard GSD," REAL Harvard GSD. Accessed: Nov. 12, 2023. [Online]. Available: https://research.gsd.harvard.edu/real/
7. "CU Boulder ATLAS Institute," ATLAS Institute. Accessed: Nov. 12, 2023. [Online]. Available: https://www.colorado.edu/atlas/
8. "AADRL," AADRL. Accessed: Nov. 12, 2023. [Online]. Available: https://drl.aaschool.ac.uk
9. "Bachelor Media & Interaction Design," ECAL - École cantonale d'art de Lausanne. Accessed: Nov. 12, 2023. [Online]. Available: https://ecal.ch/en/courses-and-research/bachelor/bachelor-media-interaction-design/
10. "NuVu Studio Innovation School." Accessed: Nov. 12, 2023. [Online]. Available: https://cambridge.nuvustudio.com/
11. "Microsoft Research Artist in Residence Program," Microsoft Research. Accessed: Nov. 12, 2023. [Online]. Available: https://www.microsoft.com/en-us/research/group/artist-in-residence/
12. Art and Innovation: The Xerox PARC Artist-in-Residence Program. The MIT Press, 1999. doi: 10.7551/mitpress/1390.001.0001.
13. K. Camarata, M. D. Gross, and E. Y.-L. Do, "A Physical Computing Studio: Exploring Computational Artifacts and Environments," International Journal of Architectural Computing, vol. 1, no. 2, pp. 169–190, Jun. 2003, doi: 10.1260/147807703771799166.
14. O. Shaer, M. S. Horn, and R. J. K. Jacob, "Tangible user interface laboratory: Teaching tangible interaction design in practice," AI EDAM, vol. 23, no. 3, pp. 251–261, Aug. 2009, doi: 10.1017/S0890060409000225.
15. K. Sorathia and R. Servidio, "Learning and Experience: Teaching Tangible Interaction & Edutainment," Procedia - Social and Behavioral Sciences, vol. 64, pp. 265–274, Nov. 2012, doi: 10.1016/j.sbspro.2012.11.031.

16. J. A. Valente and P. Blikstein, “Maker Education: Where Is the Knowledge Construction?,” Constructivist Foundations, vol. 14, no. 3, pp. 252–262, 2019.
17. R. Chen, M. Demko, D. Byrne, and M. Louw, “Probing Documentation Practices: Reflecting on Students’ Conceptions, Values, and Experiences with Documentation in Creative Inquiry,” in Proceedings of the 13th Conference on Creativity and Cognition, in C&C ’21. New York, NY, USA: Association for Computing Machinery, Jun. 2021, pp. 1–14. doi: 10.1145/3450741.3465391.
18. D. Papanikolaou, “Cloudcommuting: Games, Interaction, and Learning,” in Proceedings of the 12th International Conference on Interaction Design and Children, in IDC ’13. New York, NY, USA: ACM, 2013, pp. 459–462. doi: 10.1145/2485760.2485833.
19. N. Davis, C.-Pi. Hsiao, K. Yashraj Singh, L. Li, and B. Magerko, “Empirically Studying Participatory Sense-Making in Abstract Drawing with a Co-Creative Cognitive Agent,” in Proceedings of the 21st International Conference on Intelligent User Interfaces, in IUI ’16. New York, NY, USA: Association for Computing Machinery, Mar. 2016, pp. 196–207. doi: 10.1145/2856767.2856795.
20. S. Brave and A. Dahley, “inTouch: a medium for haptic interpersonal communication,” in CHI ’97 Extended Abstracts on Human Factors in Computing Systems, in CHI EA ’97. New York, NY, USA: Association for Computing Machinery, Mar. 1997, pp. 363–364. doi: 10.1145/1120212.1120435.
21. B. Piper and H. Ishii, “PegBlocks: a learning aid for the elementary classroom,” in CHI ’02 Extended Abstracts on Human Factors in Computing Systems, in CHI EA ’02. New York, NY, USA: Association for Computing Machinery, Apr. 2002, pp. 686–687. doi: 10.1145/506443.506546.
22. D. Leithinger, S. Follmer, A. Olwal, and H. Ishii, “Physical telepresence: shape capture and display for embodied, computer-mediated remote collaboration,” in Proceedings of the 27th annual ACM symposium on User interface software and technology, in UIST ’14. New York, NY, USA: Association for Computing Machinery, Oct. 2014, pp. 461–470. doi: 10.1145/2642918.2647377.
23. M. Deshpande, S. Sarwar, A. Mahdavi, and D. Papanikolaou, “Pneuxels: A Platform for Physically Manifesting Object-Based Crowd Interactions in Large Scales,” in Proceedings of the 2019 ACM International Joint Conference and 2019 International Symposium on Pervasive and Ubiquitous Computing and Wearable Computers, in UbiComp ’19. London, UK: ACM, 2019, pp. 9–12.
24. C. F. Griggio and M. Romero, “Canvas Dance: An Interactive Dance Visualization for Large-Group Interaction,” in Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems, in CHI EA ’15. New York, NY, USA: Association for Computing Machinery, Apr. 2015, pp. 379–382. doi: 10.1145/2702613.2725453.
25. D. Papanikolaou, A. J. B. Brush, and A. Roseway, “BodyPods: Designing Posture Sensing Chairs for Capturing and Sharing Implicit Interactions,” in Proceedings of ACM SIGCHI TEI’15: 9th International Conference on Tangible, Embedded, and Embodied Interaction, Stanford, CA, USA: ACM, Jan. 2015, pp. 375–382. doi: 10.1145/2677199.2680591.
26. “ToT __ Bientôt l’été __ ToT.” Accessed: Jan. 28, 2022. [Online]. Available: http://tale-of-tales.com/bientotlete/
27. B. Bengler and N. Bryan-Kinns, “Designing collaborative musical experiences for broad audiences,” in Proceedings of the 9th ACM Conference on Creativity & Cognition, in C&C ’13. New York, NY, USA: Association for Computing Machinery, Jun. 2013, pp. 234–242. doi: 10.1145/2466627.2466633.
28. S. Jordà, “The reactable: tangible and tabletop music performance,” in CHI ’10 Extended Abstracts on Human Factors in Computing Systems, in CHI EA ’10. New York, NY, USA: Association for Computing Machinery, Apr. 2010, pp. 2989–2994. doi: 10.1145/1753846.1753903.

29. J. Patten, B. Recht, and H. Ishii, “Interaction techniques for musical performance with tabletop tangible interfaces,” in Proceedings of the 2006 ACM SIGCHI international conference on Advances in computer entertainment technology, in ACE ’06. New York, NY, USA: Association for Computing Machinery, Jun. 2006, pp. 27–es. doi: 10.1145/1178823.1178856.
30. A. Guo, I. Canberk, H. Murphy, A. Monroy-Hernández, and R. Vaish, “Blocks: Collaborative and Persistent Augmented Reality Experiences,” Proc. ACM Interact. Mob. Wearable Ubiquitous Technol., vol. 3, no. 3, pp. 83:1–83:24, Sep. 2019, doi: 10.1145/3351241.
31. S. Robinson, M. Jones, E. Vartiainen, and G. Marsden, “PicoTales: collaborative authoring of animated stories using handheld projectors,” in Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work, in CSCW ’12. New York, NY, USA: Association for Computing Machinery, Feb. 2012, pp. 671–680. doi: 10.1145/2145204.2145306.
32. J. C. Tang and S. L. Minneman, “Videodraw: a video interface for collaborative drawing,” ACM Trans. Inf. Syst., vol. 9, no. 2, pp. 170–184, Apr. 1991, doi: 10.1145/123078.128729.
33. “The Collider – Anagram.” Accessed: Jan. 28, 2022. [Online]. Available: https://weareanagram.co.uk/project/the-collider
34. S. Angelia, N. Ohta, and K. Sugiura, “Design and evaluation of educational kinesthetic game to encourage collaboration for kindergarten children,” in Proceedings of the 12th International Conference on Advances in Computer Entertainment Technology, in ACE ’15. New York, NY, USA: Association for Computing Machinery, Nov. 2015, pp. 1–5. doi: 10.1145/2832932.2832967.
35. M. H. Fogtmann, “Designing bodily engaging games: learning from sports,” in Proceedings of the 12th Annual Conference of the New Zealand Chapter of the ACM Special Interest Group on Computer-Human Interaction, in CHINZ ’11. New York, NY, USA: Association for Computing Machinery, Jul. 2011, pp. 89–96. doi: 10.1145/2000756.2000768.
36. M. H. Fogtmann, J. Fritsch, and K. J. Kortbek, “Kinesthetic interaction: revealing the bodily potential in interaction design,” in Proceedings of the 20th Australasian Conference on Computer-Human Interaction: Designing for Habitus and Habitat, in OZCHI ’08. New York, NY, USA: Association for Computing Machinery, Dec. 2008, pp. 89–96. doi: 10.1145/1517744.1517770.
37. H. Larrea-Tamayo, “ARkits : architectural robotics kits,” Thesis, Massachusetts Institute of Technology, 2015. Accessed: Nov. 13, 2023. [Online]. Available: https://dspace.mit.edu/handle/1721.1/98627
38. H. S. Alavi, D. Lalanne, J. Nembrini, E. Churchill, D. Kirk, and W. Moncur, “Future of Human-Building Interaction,” in Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems, in CHI EA ’16. New York, NY, USA: Association for Computing Machinery, May 2016, pp. 3408–3414. doi: 10.1145/2851581.2856502.
39. M. Weiser, “The Computer for the 21st Century,” Sci.Am., vol. 265, no. 3, p. ec, 1991.
40. D. Papanikolaou, Knowledge acquisition will be like swallowing a pill. Harvard Graduate School of Design, 2013.
41. M. Kuniavsky, “The Coming Age of Magic - Orange Cone.” Accessed: Sep. 29, 2018. [Online]. Available: http://www.orangecone.com/archives/2006/10/the_coming_age.html
42. G. Lakoff, “The Contemporary Theory of Metaphor,” 1993, Accessed: Sep. 27, 2018. [Online]. Available: http://www.escholarship.org/uc/item/54g7j6zh
43. J. Kim and M. L. Maher, “Metaphorical Concepts and Framework for Designing Novel Approaches to Interactive Buildings,” in Design Computing and Cognition’20, J. S. Gero, Ed., Cham: Springer International Publishing, 2022, pp. 331–350. doi: 10.1007/978-3-030-90625-2_19.

44. S. Giedion, Space, Time and Architecture: The Growth of a New Tradition, Fifth Revised and Enlarged Edition, 5 edition. Cambridge, Mass. London: Harvard University Press, 2009.
45. B. Fuller, Ideas and Integrities: A Spontaneous Autobiographical Disclosure, Trade Paperback Edition. New York: Collier Books, 1974.
46. "International Conference on Tangible, Embedded and Embodied Interactions (ACM TEI)," ACM TEI Conference. Accessed: Nov. 14, 2023. [Online]. Available: https://tei.acm.org
47. "ACM Designing Interactive Systems Conference." Accessed: Nov. 14, 2023. [Online]. Available: https://dis.acm.org/
48. "ACADIA - Association for Computer Aided Design in Architecture." Accessed: Nov. 14, 2023. [Online]. Available: http://acadia.org/
49. D. Offenhuber, Autographic Design: The Matter of Data in a Self-Inscribing World. The MIT Press, 2023.
50. C. Petzold, Code : the hidden language of computer hardware and software. Redmond, Wash., 2000.
51. A. Chalcraft and M. Greene, "Train Sets," Eureka, vol. 53, pp. 5–12, 1994.
52. B. Hayes, "Trains of Thought," American Scientist; Research Triangle Park, vol. 95, no. 2, p. 108, Apr. 2007.
53. J. T. Godfrey, "Binary digital computer," US3390471A, Jul. 02, 1968 Accessed: Mar. 28, 2019. [Online]. Available: https://patents.google.com/patent/US3390471/en
54. L. Pitt, "Turing Tumble is Turing-Complete." arXiv, Oct. 18, 2021. Accessed: May 16, 2022. [Online]. Available: http://arxiv.org/abs/2110.09343
55. "Turing Tumble - Build Marble-Powered Computers," Turing Tumble - Build Marble-Powered Computers. Accessed: May 16, 2022. [Online]. Available: https://www.upperstory.com/turingtumble
56. The Amazing Dr. Nim board game. 1966.
57. "home | p5.js." Accessed: May 16, 2022. [Online]. Available: https://p5js.org/
58. Node.js, "Node.js," Node.js. Accessed: Mar. 27, 2021. [Online]. Available: https://nodejs.org/en/
59. "Socket.IO." Accessed: Jan. 21, 2022. [Online]. Available: https://socket.io/