ITcon
www.itcon.org - Journal of Information Technology in Construction - ISSN 1874-4753

FORM FOSTERING: A NOVEL DESIGN APPROACH FOR INTERACTING WITH PARAMETRIC MODELS IN THE EMBODIED VIRTUALITY

PUBLISHED: January 2011
EDITOR: Turk Ž.

Flora D. Salim, Research Fellow,
Spatial Information Architecture Laboratory (SIAL), RMIT University, Melbourne, Australia;
flora.salim@rmit.edu.au; http://florasalim.com

Hugo M. Mulder, Senior Engineer,
Arup Advanced Technology + Research, Arup, London, United Kingdom;
hugo.mulder@arup.com;

Jane R. Burry, Research Fellow,
Spatial Information Architecture Laboratory (SIAL), RMIT University, Melbourne, Australia;
jane.burry@rmit.edu.au; http://www.sial.rmit.edu.au/People/jburry.php

SUMMARY: *This paper investigates the assimilation of embodied virtuality with parametric modelling to inform various stages of architectural design. Embodied virtuality is a term introduced by Mark Weiser in 1991 to describe the era of ubiquitous computing. Traditionally, the data used as input to a parametric model needs to co-exist in the digital world, stored in an archive or file system of our computer. However, with the proliferation of sensor and mobile devices, data can now be found everywhere in the physical world and the boundaries between the physical and the digital has become blurred. There is a need to integrate the real world and the digital information space for designing parametric models. The assimilation of digital data from the physical world with the existing CAD tools will create a synergy between form finding and form making. This paper introduces the notion of form fostering, which stands in between physical form making and digital form finding. Form fostering takes various inputs from the physical world to inform a parametric model or to actuate a physical model in the built environment. We present early experiments of Form Fostering, which include the use of a Wii Remote, an Arduino processing board, servo actuators and a camera as haptic input and interaction devices. The potential benefits of designing in embodied virtuality are significant since designers can get real-time input from both the physical context and from interaction with the parametric model.*

KEYWORDS*: parametric design, flexible modelling, physical computing, embodied virtuality, ubiquitous computing, responsive architecture, Wii, Arduino*

REFERENCE: *Flora D. Salim, Hugo M. Mulder, Jane R. Burry (2011) Form fostering: A novel design approach for interacting with parametric models in the embodied virtuality, Journal of Information Technology in Construction (ITcon), Vol. 16, pg. 135-150, http://www.itcon.org/2011/9*

1. THE EMBODIED VIRTUALITY

The adoption of ubiquitous computing and sensory technology into interactive and responsive architectural design changes the ways in which we think about architecture, how we describe architecture and how we design architecture.

Mark Weiser coined the term ubiquitous computing (Weiser, 1991), which refers to computer technologies that disappear and are woven into the fabric of everyday life until they are no longer distinguishable from it. Computation is now embodied in day-to-day objects and our interaction with them is based on our familiarity with these real world objects (Dourish, 2001). Therefore, ubiquitous computing is also referred to as embodied virtuality (Weiser, 1991). Instead of bringing the real-world into the virtual space (the computer), the virtuality now exists in our world, embodied within the real-world objects and representations.

In Milgram's virtuality continuum, mixed reality is the spectrum between physical reality and virtual reality, as illustrated in Fig. 1 (Milgram and Kishino, 1994). In augmented reality, users are interacting with physical objects in the physical world, whereas in augmented virtuality, users are interacting with the virtual world, yet both the virtual and physical objects are displayed seamlessly in the respective worlds (Hughes et al., 2005).

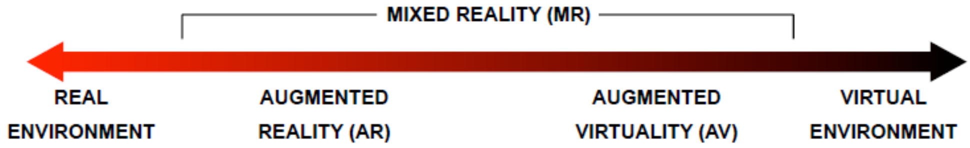

FIG 1: Milgram's virtuality continuum (Milgram and Kishino, 1994)

Embodied virtuality is another form of mixed reality, since there is no longer clear separation between the physical (or the social) and the virtual. The ends of the spectrum have met and become a continuous loop (Fig. 2). Interacting with the embodied virtuality is termed 'embodied interaction'. It is an interaction that happens in the physical and social world which gives forms, substance, and meaning within a specific setting and circumstances (Dourish, 2001). Examples of embodied interaction have been demonstrated from the mid of 1990s, such as in metaDesk or ambientRoom projects by Hiroshi Ishii (Ishii and Ullmer, 1997). This is referred by Ishii as tangible interaction.

FIG 2: Embodied Virtuality Continuum

However, such embodiment is still an unfamiliar term in architectural design. Since the world has become more digital, architects have shifted from designing with pens on papers to modelling with the mouse, viewing computer screens. Three dimensional architectural computer models in the digital space are generally very malleable in parametric software like Digital Project (Dassault Systèmes CATIA), Generative Components (Bentley Microstation) or Grasshopper (McNeel Rhinoceros). Such parametric modelling tools have speed up the process of finding the forms that satisfy design constraints and to the aesthetic and other criteria of the designer. The methods and tools for designing, prototyping and calibration of architecture in the embodied virtuality are still very limited in these types of modelling environment in current practice and research. Given the flexible modelling power of existing parametric software and the inherent parametric way of thinking in design, we posit that there is potential for extending these tools and techniques to allow for a more continuous process between physical and virtual representations while designing architecture.

Although there is no design without parameters, the term parametric design has largely been associated with the digital. Architects who use the parametric approach to design an architectural representation need to set up and work with digital parameters inside the computer. Data associated with the parameters of the model needs to be introduced within the software or accessed from data files; or in the context of embodied virtuality, the data can be taken from the outside world. This research explores real-time embodied interaction with parametric modelling through a series of prototypical experimental models. We present the preliminary outcomes of testing the hypothesis that designers can now manipulate parametric models set up either in the physical or the virtual worlds and directly associate real-time and live data sources from the real world, ambient environments, and haptic devices with parameters of the model. Such a set up allows for simulation of responsive architecture in early stage design.

This paper is organized as follows. Section 2 reviews relevant responsive architecture precedents. Section 3 introduces the notion of form fostering. Section 4 elaborates on form fostering experiments. Section 5 presents our observations, lessons learnt, and future work. Section 6 concludes the paper.

2. RESPONSIVE ARCHITECTURE

The ubiquity of bits and atoms in the mixed reality has taken architectural design to new frontiers (Massachusetts Institute of Technology, 2008). Since the introduction of "kinetic architecture" in 1970 by Zuk and Clark (Zuk and Clark, 1970), there are various terms that have been used to connote the ability of architecture, space, structure, or building to respond to stimuli from users or the environment by means of changes in shape, organization, content or appearance. Some of these responses are more complex than others, can include feedbacks and are adaptive.

The cybernetician Gordon Pask has suggested an architectural system that is involved in a continual conversation with the users and the environment. In 1978, Pask's key collaborators, Cedric Price and John Frazer, conceptualized the Generator project, known as the first concept for an intelligent building (Frazer, 1995). Adaptive architecture has only physically materialized in the last decade. Built examples are the Aegis Hyposurface project (SIAL) by dECOi that is an interactive facetted metal surface driven by pistons that responds with to multiple inputs simultaneously. Kas Oosterhuis (ONL) developed the Muscle interactive pavilion (Oosterhuis and Biloria, 2008) that responds to environmental stimuli and theoretically to other similar pavilions. The Korea Digital Pavillion (Oosterhuis and Lénárd, 2008), in which the forms and shapes of the interactive installation are adaptive to the past and present visitors, is an example of architectural design in the augmented reality. The Digital Water Pavilion, designed by the Carlo Ratti (carlorattiassociati) and the MIT Senseable City Lab, encompasses an interactive and responsive water façade in Zaragoza, Spain (Massachusetts Institute of Technology, 2008). In these examples of responsive architecture, physical changes to the building occur in real-time and are driven by physical events, physical variations in the environment, or physical interaction between the architecture and the users.

To our knowledge, there is hardly any research into modeling and simulating responsive architecture using parametric design tools nor applications in practice. By extending the parametric design approach to accommodate both the physical and the virtual of the embodied virtuality, the premise of responsive architectural prototypes can first be tested in a flexible manner before construction. The methods and tools for designing in the embodied virtuality described in this paper are built upon existing parametric software capabilities.

3. FORM FOSTERING

Form fostering challenges the traditional approach of informing a virtual model purely with virtual 'knowledge'. It seeks to measure data in the physical world and 'capture' relations and interactions that exist in the physical world (physical parameters) in a larger model that is not only virtual. The architectural design models therefore become mixed: partly virtual and partly physical.

Inherently every design is based on parametric and associative thinking. Design decisions are based or dependent on variables and on other decisions. The extent to which certain factors can play a role in a design varies. Parametric modelling uses parameterized relationships between geometrical components in the design to define forms (Monedero, 2000). A parametric model comprises parameters and constraints (Hernandez, 2006). It responds to changes of parameter value or definition and a changing graph of associations without erasure of parts of the model or starting the design model from scratch. Designers alter the value of the parameters to explore design variations that can be generated from the same model (Hernandez, 2006).

Form fostering takes sensory input from the physical worlds and parameterizes the forms with the inputs received. Collaborative adaptation is managed in form fostering since sensory input can vary. Designers make decisions on which sensory inputs need to be used to drive the parametric variations.

Fig. 3 depicts a scenario of associations between parameters in a model following the form fostering approach. A parameter can be associated with another virtual parameter and informed by virtual events. A parameter can also be linked with the physical environment and receives input from sensor or haptic devices. A parameter can also be static or constrained if it is informed for example by a static object in the built environment. This can be useful if an existing building or an already built part of the building is to be included in the model. This allows for design to take place in the mixed reality. In responsive architecture the relations between the building and the environment remain dynamic, even when the construction process has finished: an instance of the parametric model has materialized.

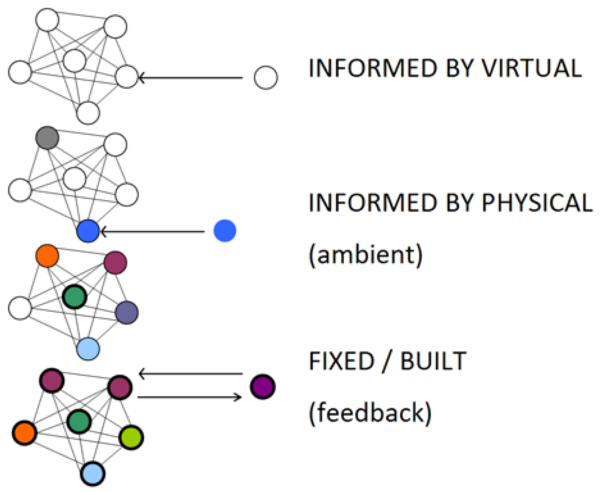

FIG 3: Parametric associations in form fostering

Form fostering requires interfaces for the mixed reality, both to interact with users and with the environment. The dynamic feedback loop is facilitated through informing the virtual models with changes in the physical world as well as informing the physical objects with changes in the virtual world. Real-time updates in the model can be managed in form fostering since designers can now use sensors, which stream data from the physical environment, as input to drive the parametric variations in the model.

4. FORM FOSTERING EXPERIMENTS

The experiments described in this paper use Bentley GenerativeComponents (GC) and McNeel Rhinoceros (Rhino) to build the parametric models. GC is a software that is based on Bentley Microstation and allows users to define objects in Microstation through a different interface. Users can add additional features and functions by coding, compiling and adding Dynamic Link Library (DLL) files as plug-ins to GC. Through GC, parametric relations can be defined between objects in Microstation. Complex models can be produced in this way as the relations can also be based on script or code.

We present in detail three form fostering prototypical experiments designed to test certain capacities “in principle”. The first one is using a Nintendo Wii Remote (Wiimote). And the second one is using an Arduino processing board, a servo actuator and a camera. Both experiments were developed in only four days during SmartGeometry 2009 using low-cost materials. The third experiment uses a set of light sensors to inform a responsive cube façade opening as well as a louver model set up in Rhino. This experiment is a Master student’s summer project that was run in RMIT. These early experiments led to the development of an open source software platform introduced in the Parametrics and Physical Interaction workshop cluster in SmartGeometry 2010. The timeline and collaborators of the experiments presented in this paper are illustrated in Fig. 4.

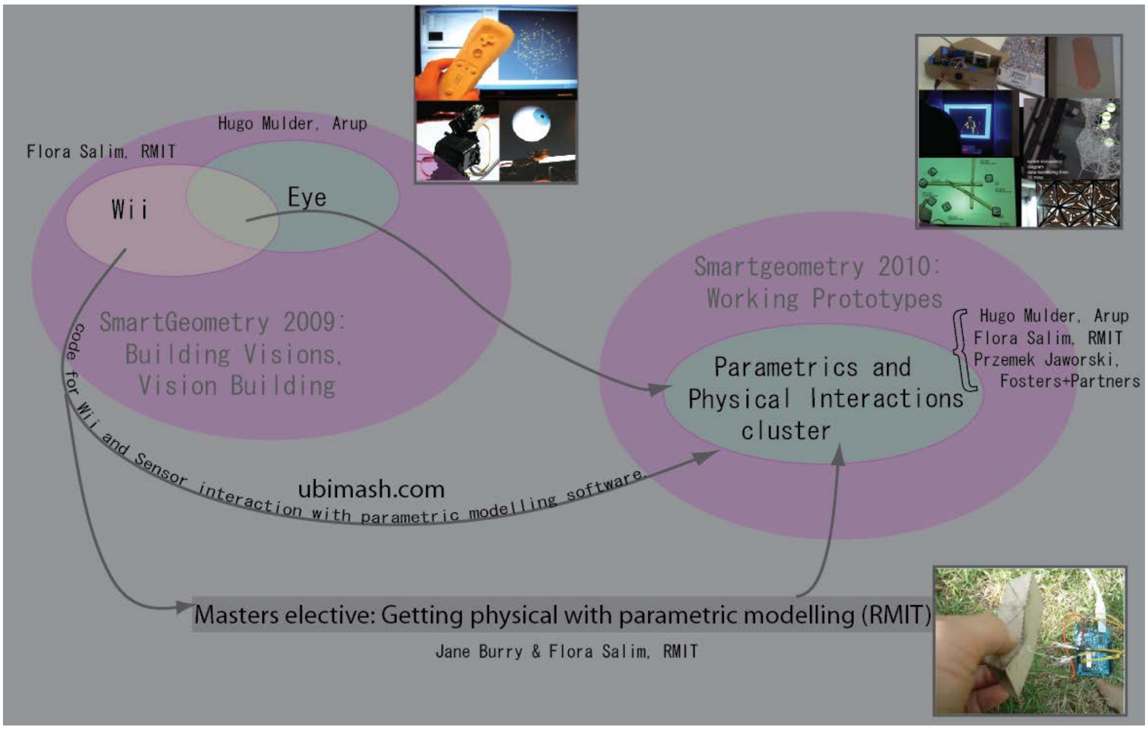

FIG 4: The timeline of form fostering experiments and the people involved

1 The Wii Remote Experiments

A Wii Remote (Wiimote) is the hand held motion detection sensor and controller for Nintendo Wii. The main components of a Wiimote are a 3-axis-accelerometer, an infrared emitter, and a Bluetooth interface. A Wiimote can be used to communicate with any computing device that has Bluetooth interface. Given its internal features, low cost, and the release of the open source for Wiimote development, in C#.NET (Peek, 2007) and Java (WiiRemoteJ, 2008), Wiimote has become a state-of-the-art wireless sensor device. Infrared tracking and augmented reality applications with Wiimote have been developed (Lee, 2008). However, none has actually incorporated the use of Wiimote to interact with parametric modeling for design. In the following experiments we have started to explore the potential of the Wiimote as a more embodied and possibly more intuitive interface in three-dimensional space with which to interact with parametric models. We first developed a Rhino plug-in that utilizes Wiimote to draw curves in 3D space. The intention was not to replace the mouse as an interface for digital designing but to explore the potential of mapping of the user's embodied space onto a 3D model. Interaction with a mouse is restricted to 2D movement on a table top. With the Wiimote, users can either sit or stand while moving their Wiimotes and interacting with the 3D virtual and physical table top models. Their movements and the accelerations of those movements can be mapped parametrically to the relationships in the 3D model on the screen. This also has the potential to bring the interaction with the model into a more inclusive social space for collaborative design meetings. The successful experiment with Rhino 3D motivated the form fostering experiment of the Wiimote interacting with GC.

In the first set of experiments, the Wiimote is staged as an input for the virtual model. The purpose of these experiments is twofold; first, to explore new modes of interaction with 3D parametric models, second, to perform "stress testing" on the model itself. Since multiple parameter value changes are sent within split milliseconds as the Wiimote is moved or pressed, the latter test is useful to inform us how resilient a 3D parametric model is when the values of the parameters are updated with streams of sensor data.

We have developed a custom C# function in GC that allows a Wiimote to interact with the GC environment using Wiimote's sensory information such as acceleration, up/down/left/right and button press/actions. The approach chosen was to input directly to the core of GC, thus various functions such as modifying nodes and changing the camera position in a virtual space were possible. The software architecture is depicted in Fig. 5.

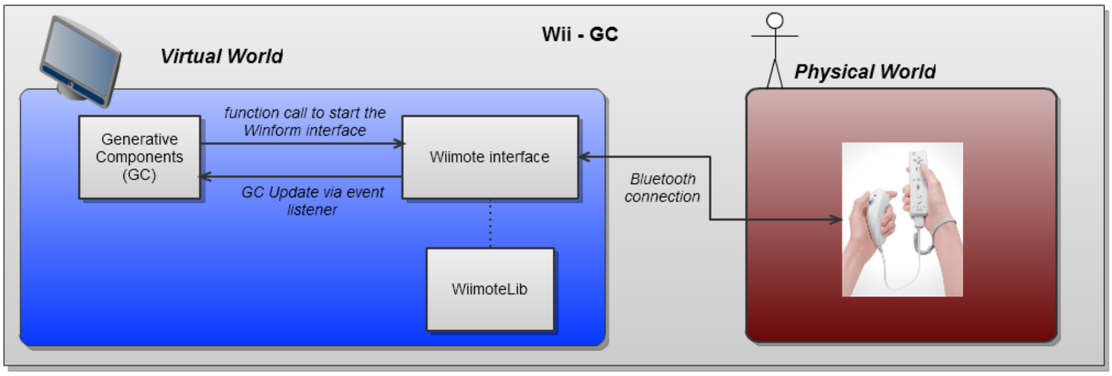

FIG 5: The software architecture of the Wiimote-GC experiment

In the experiment depicted in Fig. 6, the parametric model set up in GC includes a random number of points scattered across random positions in space and a cube that acts as a selector of points that are located within its boundaries. As the Wiimote is moved, the cube moves and resizes accordingly. The points selected by the cube are updated whenever the cube moves.

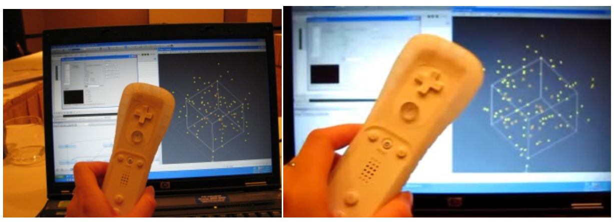

FIG 6: A Wiimote and a cube selector

In the next experiment (Fig. 7), a Wiimote is used to control the dynamic parametric surface. Whenever the Wiimote moves, the surface reforms itself dynamically based on the position of the Wiimote controller point located under the surface. This experiment has an unlimited number of real-world analogies to the creation of responsive surfaces or façades. If other sensors such as light, temperature or wind sensors can be linked to such a surfaces, it would be useful to simulate a surface that can respond, adapt, and reform itself to suit the changing environment (e.g. a roof that extends its eaves on a hot sunny day, or a wall that curves itself when someone sits in a windy non-enclosed space).

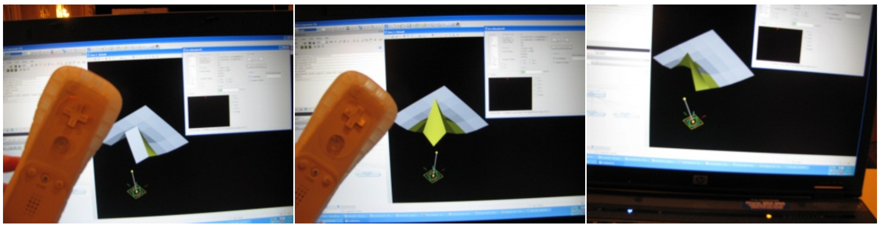

FIG 7: A Wiimote and a parametric surface

In the experiment depicted in Fig. 8, a Wiimote is used to draw a BsplineCurve. Given the unrestricted movement of the Wiimote in the physical space and its ability not only to feed x, y, z positions in space (using infra-red emitter and tracker), but also to stream gravity acceleration data as the Wiimote is moved, it has the potential to be a tool complementary to the mouse for 3D sketching.

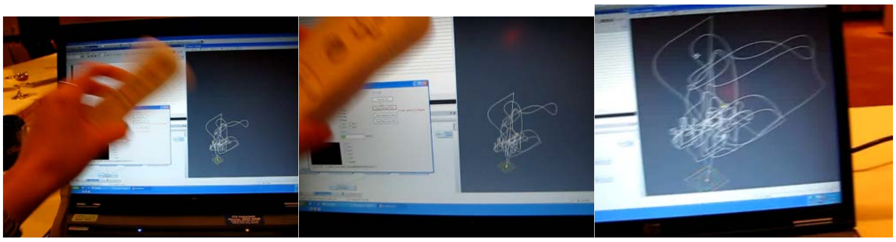

FIG 8: A Wiimote for 3D sketching

The experiment shown in Fig. 9 demonstrates the use of the Wiimote as a camera view controller. The parametric model in GC contains a sphere object. The Wiimote is connected to a GC camera object that is linked with a controller point traveling on a sphere and gives a snapshot of the model from the camera point of view. The camera views the surface from a certain direction and angle from the sphere and can take snapshots of the view.

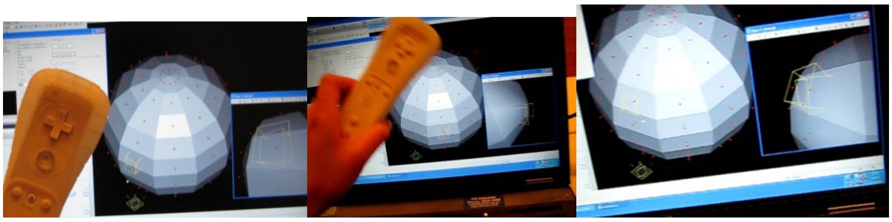

FIG 9: A Wiimote as a camera view controller

The parametric model in the following experiment (Fig. 10) has a direction vector as the parameter interfacing with the Wiimote's x, y and z gravity acceleration movement. The movement of the Wiimote in a certain direction updates the position of the axis (direction vector) of the object and causes the object to move along the same direction as the Wiimote.

FIG 10: Dancing pole

The last experiment (Fig. 11) demonstrates the use of the Wiimote buttons (up/down/left/right/A/B) as a "remote control" to dynamically move components across the grid points in the model or to magnify or shrink the selected object in the model 'on-the-fly'. For example, pressing 'B' button enlarges the GC parameterized component associated with the Wiimote.

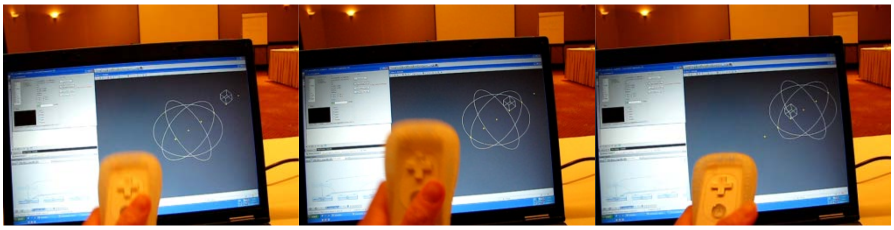

FIG 11: Dynamic object mover, magnifier or shrinkage controller

Through these experiments, we discovered that setting up and calibrating the parametric models in order to get coherent movement with the Wiimote is very challenging initially. We also discovered that the parametric models we set up in GC could not handle more than three "live" parameters. Once we have more than three parameters accepting real-time live updates from Wiimote sensors, the updates became slower and eventually stopped updating. Although this issue might be platform or software specific, sending sensor data streams straight to the parametric models without pre-processing is not efficient. It would be better to pre-process the sensor data on board the devices or in the software that bridges the connection between the devices and CAD program and only transmit data when the device's states have substantially changed.

2 The Eye

The second experiment in SmartGeometry 2009 is using a different protocol. It allows for input into GC and output from GC. The experiment setup was a simple relation between a physical model and a virtual representation of an eye. The virtual model mimics the physical model and vice versa, thereby allowing for calibration and for trans-reality feedback loops.

The physical model of the eye was a little robot that consists of a (web) camera and two servo motors. The orientation of one servo motor allows the camera to rotate along a vertical axis, and the other allows for movement along a horizontal axis. The servos rotate 180 degrees. The view of the camera could therefore cover roughly half a sphere like the eye of a chameleon (Fig. 12).

The two servos are controlled using an Arduino controller board. Arduino is an open-source electronics prototyping platform based on flexible, easy-to-use hardware and software (Arduino, 2009). Arduino controller boards connect to other hardware and have a chip that can be programmed and then run in a stand-alone mode. The board can also stay connected to a USB port of a computer and communicate through a serial protocol. The controller board was programmed to wait for a signal on the serial port. If the signal comes in the right format, the signal is decoded and the two servos are instructed to rotate to a defined angle.

The webcam was also connected to a computer. Open source software (openFrameworks, 2009) was used and adapted to control the webcam and compare consecutive frames. Areas in the image that are different from the previous are marked. Therefore the software can tell that firstly there is motion in the image, and secondly where in the image that motion takes place. The software was adapted to send data to the serial port in a format that

Arduino could interpret. The largest area of motion in the picture indicated the point of interest and the software would make the camera point towards that point of interest. When the camera moves however, consecutive images look completely different. Therefore a delay was added between instructions, and it was found that motion tracking of a single object along an evenly coloured background worked best. With the integration of openCV (OpenCV, 2009) for face recognition and tracking, the background image is ignored and the performance of the motion tracking is improved. The software implementation is depicted in Fig. 13.

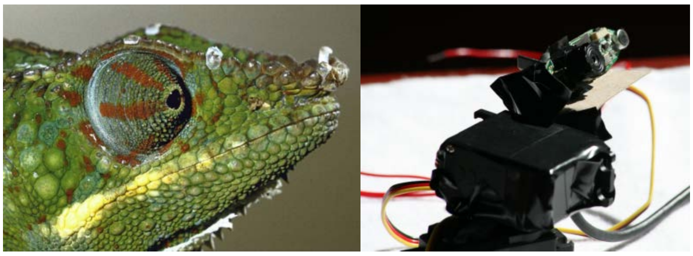

FIG 12: The physical sensor and actuator of the Eye

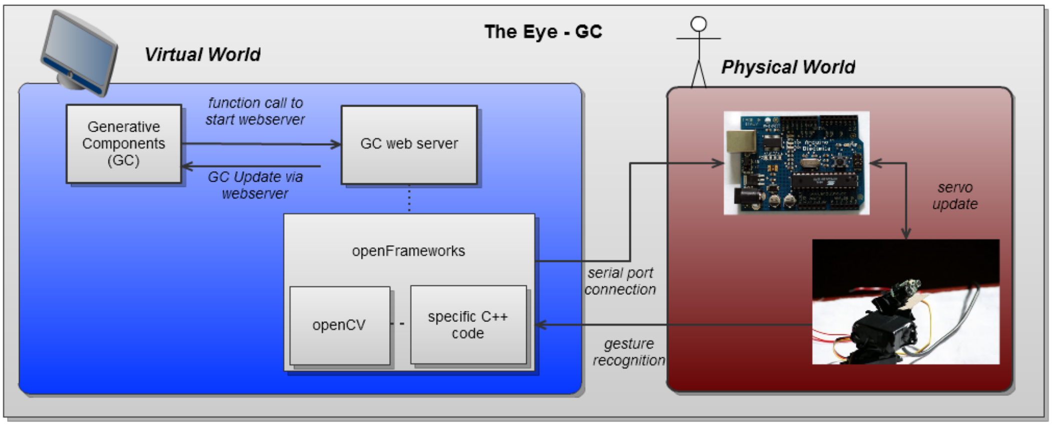

FIG 13: The software architecture of the Eye experiment

The virtual model is a sphere in GC with a dish-like solid to represent the iris and the pupil (Fig. 14). The size of the eyeball was kept fixed, but the rotation of the eye was based on two graph variables for a rotation along a vertical axis and rotation along a horizontal axis. The size of the pupil was controllable through another graph variable. There were two directions of communication: the GC output mode and the GC input mode.

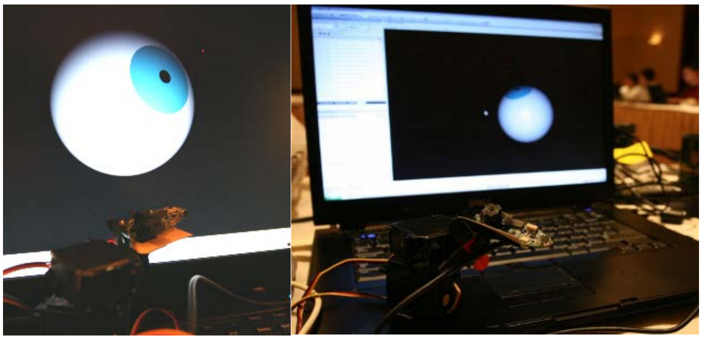

FIG 14: The Virtual Eye interacting with the Physical Eye

In the GC output mode the eyeball was controlled in GC using the sliders alongside the graph variables. A function was written in C# that would create a communication channel with the serial port. On a change of one of the variables in GC, this would send data over that channel. The data in turn would be interpreted by Arduino and rotate the camera. A window on the screen would display the live video stream of the camera.

For the GC input mode use was made of a web server that runs as a plug-in in GC. The web server when started waits for http requests of a specified form. This way the values of graph variables can be changed for example. The camera position was changed by moving objects that the camera would track. Based on the instructions sent to the camera, the software also sends http requests to the local web server. The graph variables would be updated and the graphic display in GC updates accordingly. The size of the pupil was driven by the size of the movement blob in the video stream.

These experiments demonstrated that we can set up parametric models that receive "live" input data from the ambient environment or user interaction. This implies that ephemeral data that comes from the environment, such as light, sound, and wind, does not need to be archived or recorded for simulation purposes. These experiments also demonstrated the potential of new modes of interaction with 3D parametric models. The intended behaviours of the parametric models can be scripted to generate different types of geometry manipulations while interacting with the models.

3 Responsive Façade Design with Ambient Sensor Data

The experiment described in this section was conducted by Bang Zhao, a Master of Architecture student enrolled in an elective course in RMIT. The brief given to him was to design a reading room in Melbourne. The course was conducted during Melbourne's summer time in less than two months. In this course, students need to consider the input, process, and output (IPO) of their models. The source of input data can be varied: the outdoor environments, atmospheres, the city, surrounding buildings, or the indoor environments. The process includes analysis and computation of the data and mapping of the data to the intended outcomes or behaviours. The output includes responsive behaviours, movements, or motorised actuations of the models.

A site was chosen in Melbourne Botanical Garden to design a pavilion for reading or resting (Fig. 15). The design requirements include the need to provide good light environment for reading.

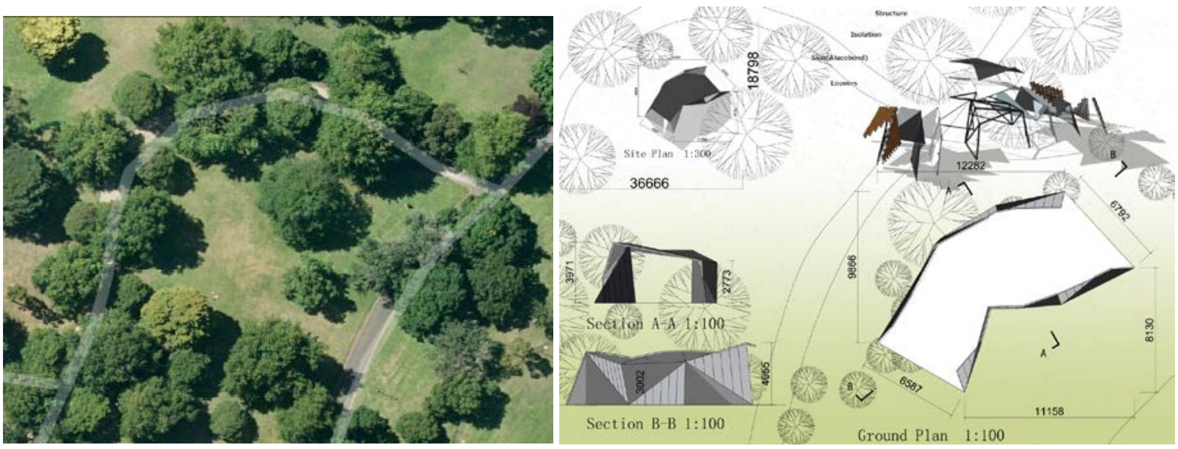

FIG 15: Left – a site chosen for the pavilion in Melbourne Botanical Garden. Right – the pavilion model.

To inform the design, instead of using the traditional approach of using light simulation and a solar path diagram in order to predict the light level at different times of day, experiments were performed with a number of light sensors and a lux reader in the design process. Although there can be a number of parameters in designing a comfortable reading space, such as noise, temperature, wind, and light, given the short time frame of the project, light was chosen as a design constraint of the reading space for proof of concept.

Firstly, we experimented with Rhino 3D models that are responsive to real-time ambient light data input from Arduino Light-Dependent-Resistors (LDR). Changes on the ambient light level trigger fluctuations on LDR readings, which subsequently trigger updates on the parametric models set in Rhino (Fig. 16). The parametric variations in the Rhino model are scripted using RhinoScript.

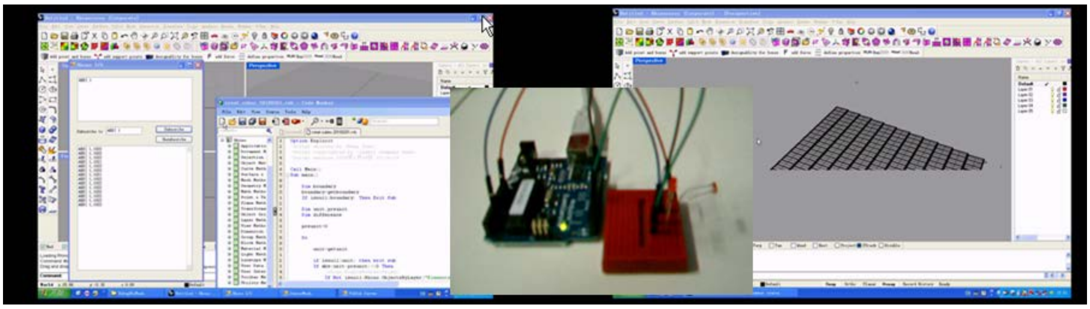

FIG 16: Responsive Rhino 3D models connected to light sensors

Secondly, the student went around a number of popular reading locations around the city of Melbourne and carrying a book, an arduino light sensor and a lux reader to measure the ambient light level in these spaces (Fig. 17). After general observations and analysing these data, we conclude that on average, Melbournians would find 580-600 lux a comfortable light level for reading. This particular part of the study is essential given there can be cultural differences in how people in general respond to a certain light level in doing specific tasks as well as regulatory differences in lighting requirements. This task was also performed to calibrate the arduino light sensor.

RMIT Building 8 Level 7(computer)		400-650 Lux
RMIT aisle		200-400 Lux
Swanston Library		400-800 Lux
RMIT cafeteria		200-400 Lux
Hungry Jack		550-600 Lux
Mel Central cafeteria Level 2		400-500 Lux
Mel Central aisle Level 2		200-900 Lux
Mel Central		1000-1300 Lux
State Library	(Public space)	300-400 Lux
		550-650 Lux
QV		300-900 Lux
McDonald		400-800 Lux
City Library	(Reading)	500-600 Lux
	(Multi-media)	300-550 Lux

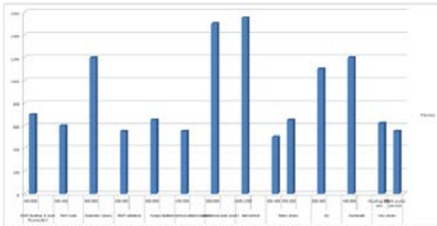

FIG 17: Left – Popular reading locations around Melbourne CBD. Right – the ambient light level measurement.

Thirdly, we calibrated the reading of the Arduino LDR with the lux reader (Fig. 18). Given that the models set up in Rhino can respond to ambient light level read by the LDR, the solution space for the optimal design can be narrowed to those that match the real-time ambient light data.

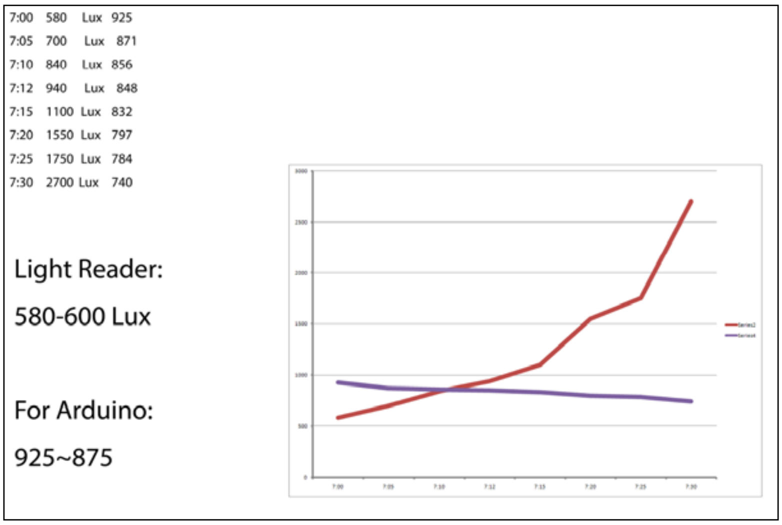

FIG 18: The calibration of the light sensor data and light reader.

Finally, he fabricated a number of pavilion models with different scales, colours (to simulate materials), and louver openings (Fig. 19). Within the fabricated models on site, he used an Arduino board and three LDRs to measure the ambient light level on site at different times of the summer days to determine the best orientation, material properties, and the angle of louver openings.

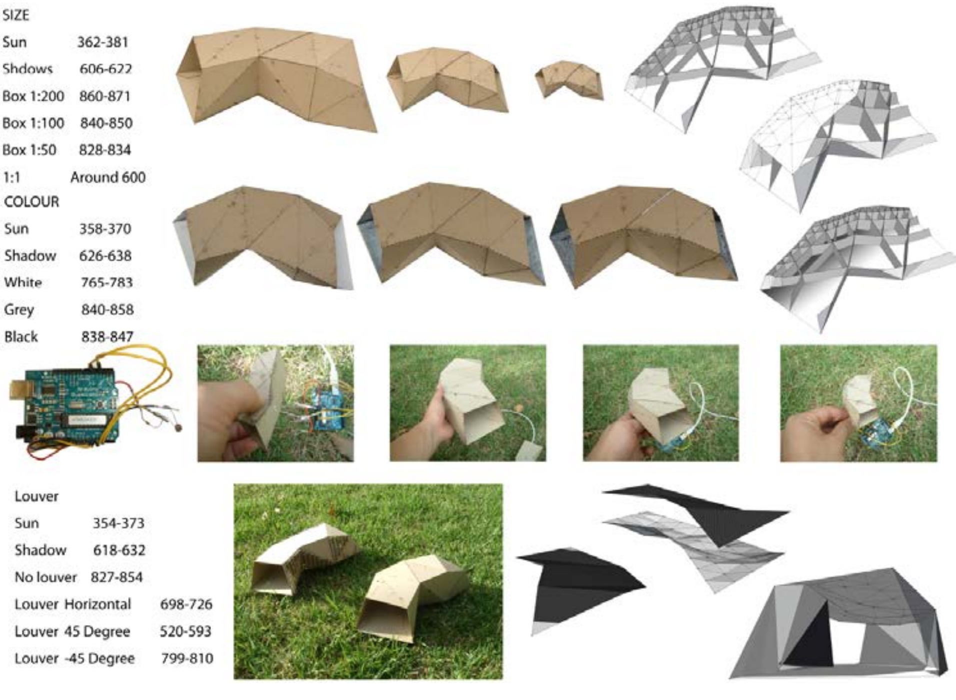

FIG 19: Ambient light analysis using Arduino and light sensors being placed inside the physical models.

This experiment demonstrates the value of working with live / real-time ambient sensor data on site in the modelling process. It also demonstrates the use of embodied virtuality with physical models. The interaction with such physical models becomes embodied since the meaning of the interaction takes place at a certain time, location, and constraints. The manipulation of placement, orientation, size, openings, shadings, and timing of the experiments directly generate data streams that can be further analysed as input to the digital parametric models. Design decisions can be derived from the analysis of the ambient data that are streamed from the digital platform embodied in the physical models.

5. OBSERVATIONS, LESSONS LEARNT, AND FUTURE WORK

The experiments in SmartGeometry 2009 demonstrated practical issues that need resolving such as delays in the system that complicate direct feedback. Also, with increasing complexity of interactions, the response of the parametric software would decrease. Associativity in the parametric model need to be carefully decoupled with associativity with real-time physical inputs, since the 3D visualization engine in the parametric software cannot keep up with the stream of data that updates the 3D models in real-time. Therefore, if the physical input is continuous rather than discrete, updates need to be deferred or aggregated by setting up time intervals for updating physical input.

There were also two different frameworks developed for the Wiimote experiments and the eye experiment. The plug-ins created for those experiments were specifically developed for connecting GenerativeComponents with particular devices. For a more generic approach and development of this field, a unified method for communication between various parts of software and hardware would be beneficial. This would allow flexibility so that "Input", "Output", and "Process" can be openly defined by the designers themselves.

Therefore, the first author invented the UbiMash software platform as a generic and open interoperability tool for designers to connect any physical devices with CAD software. UbiMash stands for Ubiquitous Mashups. It facilitates physical devices such as the Wiimotes, cameras, Arduino, sensors, and reacTIVision tangible interfaces as well as Web 2.0 data (such as from Twitter) to be connected to various parametric design software, such as Rhino, Grasshopper, GenerativeComponents, and Processing. By utilizing UbiMash, the possibilities for form fostering have become open-ended. One could use UbiMash to include ambient environmental sensor data as parameters of the model for performance based design, such as demonstrated in the third experiment in the previous section. Another potential is for designing physical and tangible interaction with CAD software. UbiMash can also be used to generate actuated feedback on physical fabricated models which are linked to the digital model in CAD.

UbiMash is a work-in-progress. It was released as an open-source (UbiMash, 2010) in March 2010 preceding SmartGeometry 2010. We aim to foster an open source community who are progressively working on a generic link between the physical and the virtual and continually testing it to generate useful and practical prototypes.

In SmartGeometry 2010, the first and the second author collaborated with Przemyslaw Jaworski from Fosters+Partners to run the workshop cluster *Parametrics and Physical Interactions*. UbiMash was heavily tested and revised during the four days workshop in cooperation with the workshop participants from academia and industry. The outcomes include seven working prototypes (Fig 20): subjective space scanner, responsive media façade, subject tracking, rapid design and build coordination with tangible markers, responsive bioclimactic ceiling, tangible user interface for urban planning, and mass form finding by Twitter (Salim, Mulder, and Jaworski, 2010).

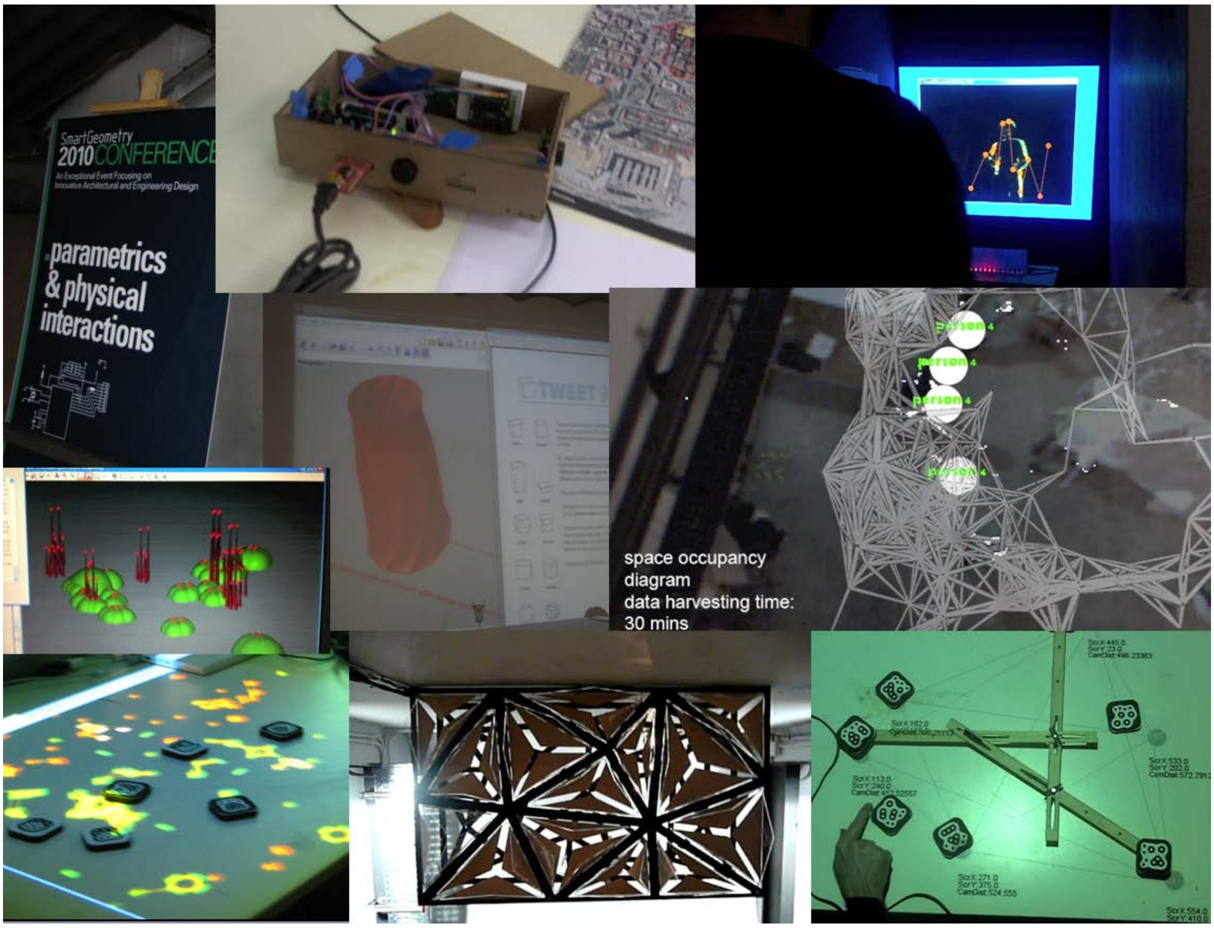

FIG 20: Parametrics and Physical Interactions workshop cluster in SmartGeometry 2010

Based on the experiments reported in previous section and this section, we observe that one of the key principles in form fostering is to *get the right understanding of a model*, not necessarily the right model, in the first place. Our pedagogical experience of passing on the knowledge of form fostering to others require us to teach about *input, behaviors, and output* of a model. As we are not setting up a static model, but kinetic, interactive, or responsive, a model can receive live input from the environment or a physical interface, will behave in certain ways, and subsequently, will generate output or feedback, either in the virtual space or in the physical world.

Since there can be too many data to respond to, data analysis, aggregation, or filtering mechanisms are often required in between a live data source and a model. Further, ambient environmental data often require interpretation, therefore calibration of physical sensing is required before mapping the data to a 3D model.

Fabrication of 3D models designed with form fostering techniques can be a practical means of evaluating the suitability of the model on site, such as demonstrated in the summer elective project. However, fabrication of a responsive model can ultimately lead to a kinetic, interactive, or responsive architecture if configured with a set of sensors and actuators. This is demonstrated in the Bioclimactic responsive ceiling prototyped in SmartGeometry 2010. A light sensor is attached to each frame of the ceiling, with capabilities of sensing light and shade (and therefore people's movement), and a servo motor is attached to three glass+opaque panels in each frame to control the movement of the panels in response to light and shade. People passing under the ceiling could also interact with the sensors in order to generate movements on the panel and experience the variations of light, shade, and reflection from the ceiling across the wall and the floor.

The work will focus on the continuity of the responsive design through virtual and physical stages and includes experiments on capturing social and environmental parameters from the urban environment. An investigation of tools and techniques is also taking place to inform the system with energy analysis software. A number of other

CAD software are also being considered to be linked into UbiMash in order to expand its interoperability and user base.

6. CONCLUSION

The system prototypes for experiments with form fostering validate the hypothesis of the potential of parametric modeling software as a simulation bench for responsive architectural design. When integrated with sensors, actuators, or any physical devices, an early-stage design of responsive architecture can now be simulated in the digital environment. Design variations in the virtual model can be generated by parametric changes from the physical environment and vice versa. Responsive architecture may drive this research, but it is envisaged that the outcomes are more broadly usable in parametric design.

The term *form finding* is now familiar with respect to pre programming a parametric computational model to search out optimal form solutions to a defined system of associations, parameters and constraints. The design progresses through the iterative refinement of these inputs. By contrast, *form fostering* is adopted as the description of a system of searching in which the designer may mediate in real time through either virtual or physical interface in the embodied virtuality. This responsive connection between physical inputs, virtual inputs and bi-directionality of virtual and physical models is the key to the concept. It is proving a novel and valuable approach to marrying the active formal interaction of the designer with ambient inputs from sensing in combined discovery mode. In extended group design conversations, the potential for also linking virtual model intervention to actuated physical model response (as illustrated by the Eye) is very powerful.

We have reported form fostering experiments from various venues, including SmartGeometry 2009, a summer elective in RMIT, and SmartGeometry 2010. The UbiMash software framework was developed throughout the course of this project. UbiMash has enabled connectivity and interoperability between physical devices and 3D CAD software and between social networking and design. Connecting the virtual and the physical world through UbiMash has led so far to prototypes that can be classified as: *(1) design informers*, *(2) responsive prototypes*, and *(3) interfaces for collaborative design*. The Wii and the Eye experiment can be placed under the first classification, the responsive façade design under the second. If the Wii experiment is enhanced with multi-player collaboration via multiple Wiimotes connected to the same model, the experiment can as well be classified under the third category. The SmartGeometry workshop has taught us that there is a great potential for collaborative design by the means of form fostering and UbiMash.

In taking the research forward, we are planning to investigate in more detail the use of mixed reality tools to support real world interdisciplinary design conversations. This demands a high degree of fluency between the virtual and physical representations and design moves.

7. REFERENCES

Arduino. (2009), "Arduino - HomePage". Available from: http://www.arduino.cc/. (accessed 09th November 2009)

Dourish, P. (2001). Where the Action Is: The Foundations of Embodied Interaction. Cambridge, MIT Press.

Frazer, J. (1995). An Evolutionary Architecture, London, Architectural Association Publications.

Hernandez, C.R.B. (2006). Thinking parametric design: introducing parametric Gaudi. Design Studies, 27, 3, 309–324.

Hughes, C. E., Stapleton, C. B., Hughes, D. E. and Smith, E. (2005). Mixed Reality in Education, Entertainment and Training: An Interdisciplinary Approach. IEEE Computer Graphics and Applications, 26,6, 24-30.

Ishii, H., and Ullmer, B. (1997). Tangible Bits: Towards Seamless Interfaces between People, Bits, and. Atoms, Proceedings of CHI'97, 234-241.

Lee, J. C. (2008). Hacking the Nintendo Wii Remote. IEEE Pervasive Computing, 7,3, 39-45.

Massachusetts Institute of Technology. (2008), Digital Water Pavilion Makes A Splash In Spain. ScienceDaily. http://www.sciencedaily.com/releases/2008/06/080617115237.htm.

Monedero, J. (2000). Parametric design: a review and some experiences. Automation in Construction, 9, 4, 369–377.

Milgram, P. and Kishino, F. (1994). Taxonomy of Mixed Reality Visual Displays. IEICE Transactions on Information Systems, E77-D,12, 1321-1329.

Oosterhuis, K. and Biloria, N. (2008). Interactions with proactive architectural spaces: the muscle projects. Communications of the ACM, 51,6, 70-78.

Oosterhuis, K. and Lénárd, I. (2008). ONLogic: speed and vision. Melbourne, Images Publishing.

OpenCV. (2009). "OpenCV Wiki". Available from: http://opencv.willowgarage.com/wiki/. (accessed 9th November 2009)

openFrameworks. (2009). "openFrameworks". Available from: http://www.openframeworks.cc/. (accessed 9th November 2009)

Peek, B. (2007). "Managed Library for Nintendo’s Wiimote". Available from: http://www.brianpeek.com/blog/pages/wiimotelib.aspx. (accessed 18th October 2010)

Salim, F. D., Mulder, H., Jaworski, P. (2010), Demonstration of an Open Platform for Tangible and Social Interactions with Responsive Models, Association for Computer Aided Design in Architecture (ACADIA) 2010 Conference - Life in:formation, New York, US, 21-24 October 2010.

UbiMash. (2010). "UbiMash". Available from: http://ubimash.com (accessed 18th October 2010)

Weiser, M. (1991). The Computer for the 21st Century. Scientific American, Special Issue on Communications, Computers, and Networks, September, 1991

WiiRemoteJ. (2008). "WiiRemoteJ". Available from: http://code.google.com/p/bochovj/wiki/WiiRemoteJ (accessed 18th October 2010)

Zuk, W. and Clark, R. H. (1970). Kinetic Architecture, New York, Van Nostrand Reinhold.