

## **FLEXM: DESIGNING A PHYSICAL CONSTRUCTION KIT FOR 3D MODELING**

MARKUS ENG, KEN CAMARATA, ELLEN DO, MARK D. GROSS

*Design Machine Group, Dept. of Architecture  
University of Washington  
Seattle, WA 98195 USA  
<http://dmg.caup.washington.edu>*

**Abstract.** This paper presents our investigation in designing a hub and strut kit that interfaces to a 3D graphics application. FlexM is a prototype in progress of a flexible physical interface for manipulating and building 3D geometry. Using the FlexM hub and strut components, designers can build and explore 3D geometry with the ease of a toy and the power of a computer. The hubs transmit the model's topology and geometry to the computer, where the model is rendered on the screen in real time.

### **1. INTRODUCTION**

Designers often struggle with a standard CAD software interface for three reasons: 1) they are trained in drafting and physical model building, 2) manipulating the screen user interface with a mouse lacks the direct tactile feedback of the object's form, which is counter-intuitive to the designer's training; 3) using CAD software requires the explicit commitment of action (e.g. defining a cube through specific coordinates) with complex commands and menu operations.

The absence of haptic stimulus can diminish the pleasure of creating. Using a 3D graphics application, like CAD, demands an additional level of mental abstraction. Designers understand 3D geometry and spatial hierarchy in a design because they can perceive it in a physical model. But, to construct the same model on the computer forces the designer to operate by clicking and dragging the mouse in x-y-z coordinates, rotating and extruding figures, shifting from model space to paper space, etc. These operations are unrelated to the designer's training in drafting and model building. In order to become adept in using the software, the designer must understand the

application's logic, and follow specific strategies of construction, which may bear little relation to how the modeled artifacts are actually made.

The difficulties designers experience with CAD applications could be improved by giving back to the designer the haptics of physical construction. What if CAD possessed the attributes of toys, which are accessible, intuitive and fun. Construction kits, like K'nex™, LEGO™ Technic, Magnetic Geomag™ or Ramagon™, are easy to use, have flexible, moving pieces, and allow endless combinations for creativity (Anderson et al. 2000). Children and adults enjoy playing with construction toys because they facilitate the exploration of geometry, mechanics and kinematics. However, construction toys lack the ability to transfer the physical design into a digital format in order that the design can be further explored in greater depth with feedback or simulation that can further inform the design construction.

What if CAD could be a tool for exploration, not just a graphics presentation tool? What if we could instill the creative potential of the construction toy into CAD through a physical, toy-like interface. This interface would be a physical model, built from a construction kit. A digital model would be reconstructed from the physical model through its computer interface. Having a digital model makes it possible to test design scenarios using simulations to predict daylighting, structural stability, acoustics, solar heat gain, and other design performance criteria. A digital format also affords the ease of generating a multitude of design alternatives.

FlexM intends to bridge the gap between the tactile interaction of play and the 3D graphics application through a physical, flexible model. The FlexM hub and strut construction kit is the modeling interface between the designer and the computer.

## **2. RELATED WORK**

FlexM is about playing, building and visualizing. Play is a form of thinking with one's hands. Building is creating form, and visualizing is interpreting information. To elaborate on these goals, the related work falls into three categories: traditional construction toys, computational construction kits and parametric based graphics.

Construction kits have been around in consumer society since the 1910's. Gilbert invented the Erector set in 1913 (Erector web site). In the same year Pajeau conceived of Tinker Toys (Tinker Toys web site). Current kits, like LEGO™ and K'nex™ toys, as well as the Hoberman Sphere, offer dynamic, expanding joints. The Hoberman Sphere, a compact ball of interconnected, folded hinges, expands into a sphere, over four times its original size. Its transformation from a stellated polyhedron to a geodesic sphere encompasses both a technical and a toy-like appeal (Hoberman 1996). FlexM follows in that spirit.

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

Computational construction kits are a new development, now possible because of the miniaturization of electronic devices. Gorbet and Orth's (1997) Triangles is a construction kit of flat, plastic triangles, which interface to a computer. Each triangle tile corresponds to a different application, like an email program, or a personal calendar. The user activates the program through the tile face. The pieces have integrated, mechanical and electronic, magnetic connectors, which allows the user to build a variety of geometric forms that correspond to his suite of applications.

Anderson et al.'s Computational Building Blocks (2000) facilitates computer modeling with LEGO™ like blocks. This work expands on the pioneering explorations by Aish (1979), and subsequent research by Frazer (1981), and Dewey and Patera (1987). Aish's Building Block System is a block set for interactively representing the structure and physical properties of the world. Frazer's 3D input devices, "Machine Readable Models" and "Intelligent Modeling Systems," allows designers to build models that interface with software that can give design advice. Dewey and Patera developed processors to manipulate the geometry of 3D models. All these projects, however, lack a real-time interface for detecting moving pieces. Computational Building Blocks are static pieces. Although the Triangles have hinges, they assemble to make a static, rigid form.

Several projects track the movements of physical objects to generate animation. Monkey™ is a specialized input device for virtual body animation (Esposito et al. 1995). It resembles a mechanical mannequin with articulated limbs. Instead of constructing a simulation of human animation and locomotion using a screen interface, the animator poses and moves the Monkey™ to define the character's animation. Topobo is another project involving character animation (Raffle et al. 2003). It is a construction kit of articulating vertebra-like pieces for building posable forms with embedded kinetic memory. The embedded memory records the angular movement at the joints. Users build a creature, move the model across a terrain, and then watch the model replay its movement from its embedded kinetic memory.

Similar to Topobo's mechanical widgets, Phidgets is a construction kit of physical computing widgets: sensors, motors, radio frequency ID readers, and a software interface for user interaction (Greenberg and Fitchett 2001). For example, users can use a motion sensor at a doorway to activate a light in the adjacent room to signal someone entering. Phidgets do not require any knowledge of processors, communication protocols or programming. Their ease of use, modularity and ability to facilitate event-driven interaction make them a handy resource for building tangible user interfaces.

CUBIK is a tangible modeling interface to aid architects and designers in 3D modeling. It takes the form of a mechanical cube (Lertsithichai and Seegmiller 2002). The designer manipulates dials on the cube's face to

expand or contract the face's dimension. CUBIK's corresponding graphic user interface (GUI) displays in real-time how the cube is expanding or contracting. The communication between the GUI and CUBIK is bi-directional. The designer can manipulate the physical cube through the GUI,

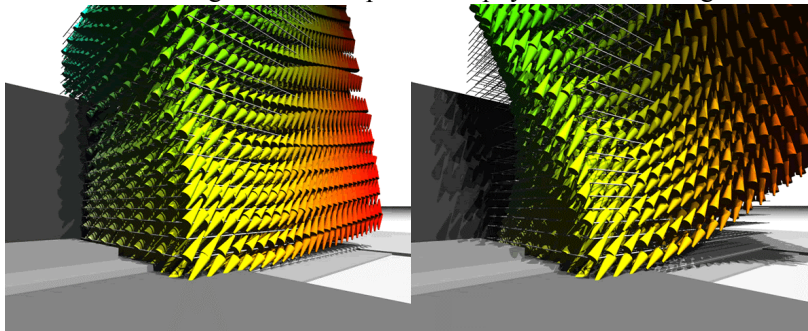


Figure 1: *Parametric based graphics rendered in PovRay.*

or change the cube's shape in the GUI via the mechanical cube. Both CUBIK and Monkey™ have engaging physical interfaces, which encourage the spontaneous act of play.

One motivation for FlexM arose from our experience with the Persistence of Vision raytracing engine (POV-Ray). POV-Ray is a script-based, shareware 3D graphics application (POV-Ray website). The user can create complicated forms through an algorithm, macro or user defined function. Artists and designers can script visually complex, parametric designs (Figure 1). Yet, for most people, writing script is neither an easy nor an intuitive task. We want to make it easy for designers to create and manipulate 3D models using a physical interface. Therefore, we proposed a tangible interface that could shear, twist or distort the computer model, in place of learning the POV-Ray script.



Figure 2: *Tinker Toys, K'nex, Zometools*

### 3. DESIGN CONCEPT

FlexM is a computationally enhanced construction kit of hubs and struts for the designer to explore 3D systems through graphics with the spirit of a toy and the power of a computer. It relates to the class of construction toys that includes Tinker Toys, K'nex and Zometools, which consists of hubs and struts. With these basic pieces one can build a multitude of shapes (Figure 2). Because these hubs are rigid, they do not easily support forms that could be squished, twisted or deformed. The FlexM hub is designed to be flexible, allowing a greater range of movement than its toy counterparts do.

We envision the designer building a model with FlexM hubs and struts, manipulating the model, and seeing the digital version of the model change with his manipulations in real-time. The model can refer to a building's spatial composition, a simplified version of the building with key points mapping to actual points on the building, or its structural frame. What if one had the option to build the digital model strictly with the FlexM components without touching a mouse or a keyboard? Or, the designer could use an existing digital model, and map key points to the FlexM model. Because the

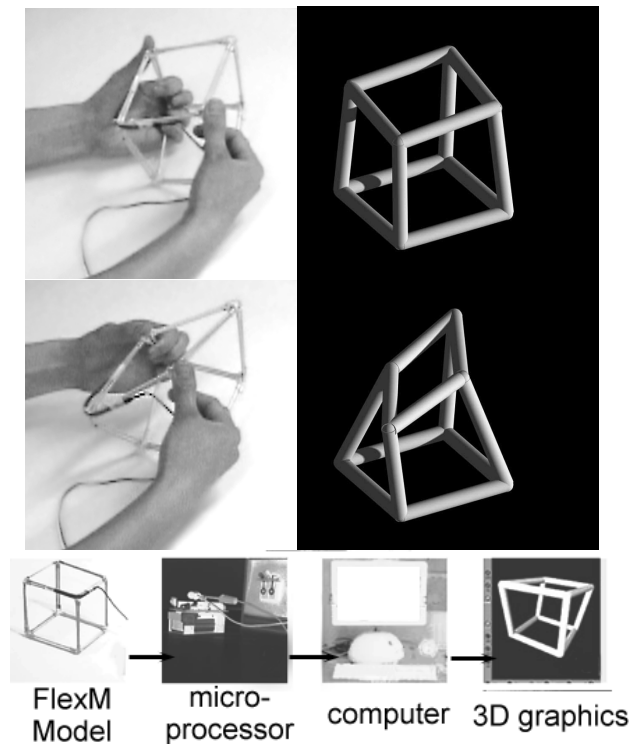


Figure 3: *Flexible model driving the graphics display.*

hubs have a flexible connection, the designer can shear, twist, and rotate parts of the model. The model connects to a computer via a Handyboard microprocessor, which transfers the model's topology and geometry to a receiving CAD or 3D graphics application. The graphics program uses these data to reconstruct and update the model. The application not only displays the model's changing form, but can also record the data for animation or separate modeling case scenarios. Figure 3 illustrates the FlexM concept of a flexible model driving the graphics display. This was our first prototype built of sticks and surgical tubing with a bend sensor mounted at the corner. The computer graphics figures are generated in VRML.

Figure 4 shows the latest prototype of the FlexM flexible hub interfacing to the FormWriter 3D graphics program (Gross 2001) via a Handyboard microprocessor. A mirror image of the FlexM hub displays on the screen. As the designer squishes the angles at the hub, the hub on the screen also squishes. The designer then builds a cube with seven more hubs. The FormWriter program updates the screen to display the cube model.

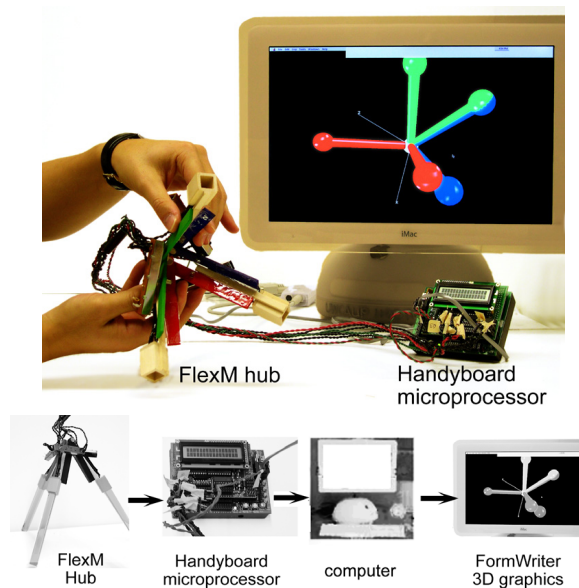


Figure 4: *FlexM with Handyboard and corresponding graphics display.*

#### 4. INVESTIGATION

FlexM is a computationally enhanced construction kit interfacing a 3D modeling program. The computational enhancements are the following: photosensors are used for determining topology; the Handyboard

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

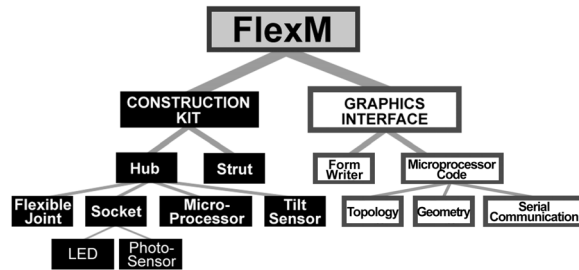


Figure 5: *FlexM* project diagram.

microprocessor reads, records and transmits the sensor data to the host computer; and the software that translates the data into a 3D model in the FormWriter graphics program. FlexM is composed of specially constructed hubs and struts that send the model's topology and geometry information to a graphics application on a host computer (Figure 5). Much of the design focuses on the flexible hub. It consists of flexible joints, sockets to connect the hubs with each other, tilt sensors for orientation, and LEDs and photosensors for communication. The microprocessor operates them and send their data to a graphics interface.

In order to reconstruct the model digitally, the graphics application must capture the model's topology and geometry. The topology defines what components are connected to each other. The geometry explains the angles of their connections and the orientation of these angles. For example in Figure 6, both models have the same topology (cube 1 is connected to 2, etc.), but different geometries. In the model on the right the first three cubes are rotated 90 degrees counter-clockwise along the strut connecting cube 3 and 4. The FlexM hub components represented in Figure 6 would identify and send the model's topology and geometry. The next sections detail how each of the parts achieves this.

### 4.1 TOPOLOGY

FlexM modules have two physical, components: hubs and struts. A model is

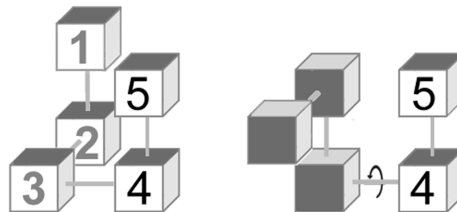


Figure 6: *Both models share the same topology, but their geometries differ.*

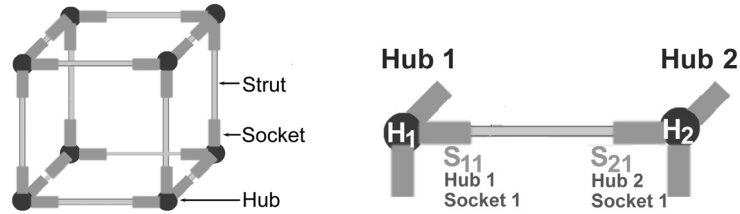


Figure 7: (left) *Definition of the parts, defining the model's topology.* (right) *Detail of Hub and socket topology. Hub 1 connects to Hub 2, through Hub 1's Socket 1 and Hub 2's Socket 1.*

a collection of  $n$  hubs ( $H_1, H_2, H_3, \dots, H_n$ ), which are vertices on the model. For example, a model of a cube would have eight hubs, one for each vertex (Figure 7, left image). Each hub has a set of  $m$  sockets ( $S_{11}, S_{12}, S_{13}, \dots, S_m$ ), equivalent to the connections on each hub. In the cube example, each of the eight hubs have three sockets or connection points. Two hubs are connected through a single strut, which make up the edges of the model. The cube has 12 edges (half the number of sockets, since it takes two connected sockets to define an edge). A strut connects to a socket on each pair of hubs (Figure 7, right image).

A method to identify the model's topology can be explained through the Flashlight Game. In Figure 8, Person A points a flashlight. Person B sees the light and knows that it is person A. In turn, person B shines the flashlight. Likewise, person A sees the light and knows that it is person B. Person C also sees the light and knows that it is person B. Person C shines the flashlight, and person B sees the light and know it is person C. This associative coupling of information defines the two-person connections.

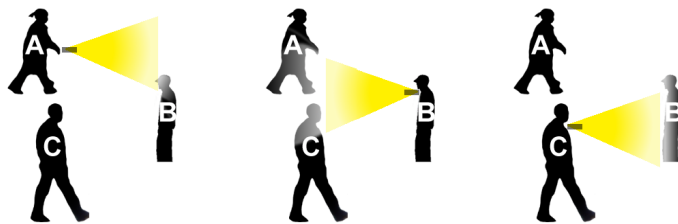


Figure 8: *Flashlight game*

In the present FlexM prototype, each hub has three sockets, which are both the physical and electronic connectors to the hubs via the struts. Using light as the communication medium, the LED and photosensor define the connection, corresponding to the flashlight and person in the Flashlight Game. Each socket has a high intensity LED and a photosensor. When



## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

switched on, the LED casts light from the socket through the acrylic strut to a photosensor in the socket of the attached hub (Figure 9). Each hub in turn flashes on, streaming light to the attached hubs. Each of the receiving hubs checks if their photosensors can see the “sending hub.” This process produces a list of connections between the hubs (hub 1 connects to hub 2, socket 1; hub 2 connects to hub 3 socket 4; etc.). Because this current prototype only allows one connector between two hubs, one does not have to include the socket of the “sending hub.” The connection “hub A socket 1 connects to hub B socket 2” can be deduced from the combination of “hub A connects to hub B socket 2” and “hub B connects to hub A socket 1.” This list is the connectivity information an interfaced graphics program needs to reconstruct the digital model.

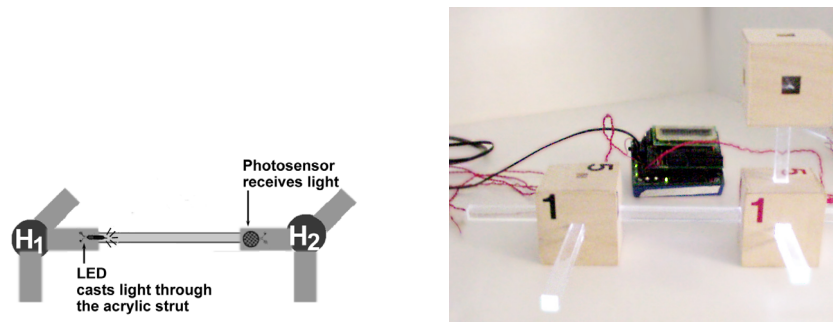


Figure 9: (left) *FlexM* socket connection. (right) *Early model of lit hub with one socket for each face.*

### 4.1.1 Socket Design

The socket design went through ongoing modifications during the development of three hub prototypes: rigid cube, flexible knee-braced hinge, and the flexible hinge (Figure 10). To simplify the geometry, the socket remained a square, which prevents the strut from rotating in the socket.

The first socket design incorporated a rollerball switch to identify when a strut was connected to the socket (Figure 11). This was needed to

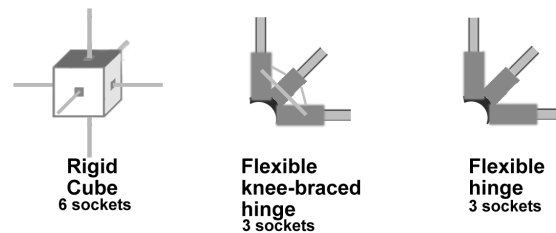


Figure 10: *Evolution of hub design with socket.*

differentiate light readings from the lit LED and ambient lighting. (Section 4.1.2 Strut Design discusses ambient light further.) The inserted strut engaged the switch, which allowed the activation of the LED and photosensor. When the switch was off (no strut), no LED would flash on and no reading would be made from the photosensor. Despite these advantages, the rollerball switch was not used in later designs to simplify the circuitry and the construction, and because the ambient lighting issues were resolved by employing a stronger light source.

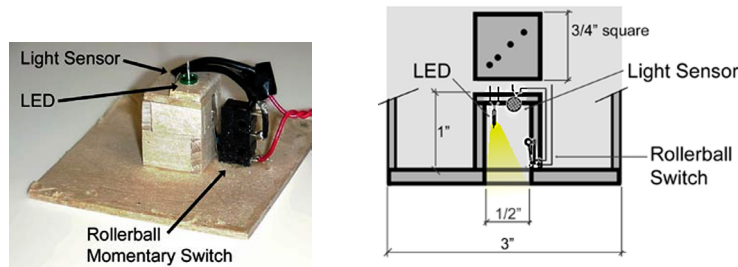


Figure 11: (left) Inside face of hub cube with the socket components. (right) Corresponding section detail of socket.

The second socket design simplified the hub wiring by reducing the number of LEDs from six LEDs (one per socket) to three LEDs (three per hub). Three LEDs are placed in the center of the hub, and the six sockets have an aperture for the light to shine through (Figure 12, left image). This approach reduces the number of LEDs from six (one for each socket) to three, resulting in fewer soldering points. In spite of the time and labor savings, the alternate design gave a reduced performance in light intensity (Figure 12, right image). The drop in light intensity increased the range of photosensor readings, which complicated the programming. Light from the core LEDs was not strong enough for a photosensor reading to be distinguished from an ambient light reading. The decrease in light was due to LED orientation and wiring obstruction. It was difficult to orient three LEDs to shine equal amounts of light to six socket apertures with wiring obstructing the line of sight. Due to the complications from the reduction of LEDs, we decided to keep the original design of having six LEDs, one for each socket.

Simplicity and performance were the driving factors for the socket design. Each hub socket has a high intensity LED and a photosensor. The rollerball switch complicated the wiring and the programming with no significant benefit. The attempt to consolidate the LEDs in the hub's core presented problems with light intensity due to line of sight issues and wiring obstruction.

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

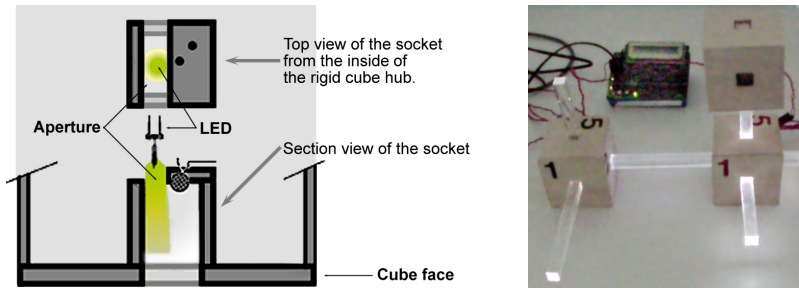


Figure 12: (left) Section cut of cube face showing the LED shining through an aperture in the socket. (right) This photo shows the difference in light intensity between the cube with LEDs in the hub's core (left cube) and cubes with LEDs in each socket (right cubes).

### 4.1.2 Strut Design

The strut serves two functions: physical connection between the hubs and the medium to channel the light to the connecting hub. The initial strut design was a square, hollow, wooden strut. The strut was square in section to prevent rotation between the hubs (Figure 13). A round dowel would have to have a guiding notch to prevent rotation, and would require boring a hole through a 4-6 inch dowel.

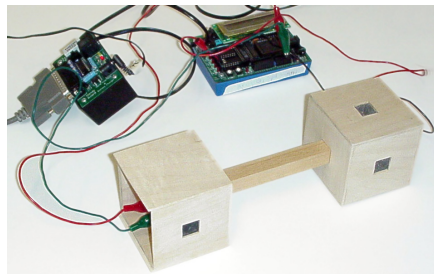


Figure 13: Hubs connected by a hollow wooden strut.

The second iteration of the strut was the square, acrylic strut. Acrylic is an excellent medium for transmitting light. The difference in material did not influence the strut's function as a mechanical connector. Based on light studies we performed, the wood strut has a critical difference in transmitting light, compared to the acrylic one (Table 1). Wood has two advantages. First, ambient light does not interfere with the light sensor reading. Second, the light sensor readings from struts of varying length are easily discernible. The strut's length affects the attenuation of light to the sensor. The differing light sensor values are discrete, which makes it possible to identify one length of a strut from another.

		<b>PHOTOSENSOR READING</b>	
		0=highest level of light	255=no light
		No Ambient Light	Ambient Light
<b>WOOD STRUT</b>	4-inch length	57	55
	6-inch length	99	96
<b>ACRYLIC STRUT</b>	2-inch length	17	12
	6-inch length	15	8

Table 1. This table shows how strut material, strut length and ambient lighting affect the photosensor reading of a standard LED. Note that a high intensity LED yields a zero photosensor reading, the highest measurable amount.

The transparency in acrylic creates both disadvantages and advantages in design. It would be difficult to differentiate between different lengths of acrylic struts. Because ambient light cancels out the potential for light attenuation, one would not be able to distinguish different lengths of the acrylic struts. Furthermore, even if the acrylic struts were sheathed with an opaque film to block ambient lighting, its low index of refraction (similar to sea water) would not attenuate the light to discrete values. Nevertheless, the transparent acrylic transmits a strong signal to the photosensor. Another advantage of acrylic is the fabrication. It is less labor intensive than the wooden struts. Each wood strut required cutting, gluing and sanding down four wooden strips to form the hollow, square strut. The acrylic strut only needed to be cut to size from the square rod.

Ambient light introduces a noise issue for the acrylic strut. In lighting condition studies, it is observed that, the acrylic strut collects more light from the room's fluorescent lighting than the sending LED. By replacing the LED light with a high intensity LED, the interference of ambient light is resolved. The high intensity LED exceeds the maximum threshold of the light sensor, which indicates the highest light reading.

In conclusion, acrylic struts with high intensity LED lights make up an effective system to identify hub connectivity.

#### 4.2 GEOMETRY

LEDs and photosensors identify the connections, but they do not measure *angles* of the connections, the geometry. We explored different sensors to measure the angle between the struts at the hub: bend sensor, sliding potentiometer and rotational potentiometer. However, these sensors only measure the angle, not the vector. One also needs the orientation of the angle. Figure 6 shows that both hub models have the same topology *and the same angles between the connections, 90 degrees*. Rotating cubes 1, 2, and 3

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

in the right model changes their orientation, not the angle. To resolve orientation we utilized an array of tilt sensors to identify orientation.

### 4.2.1 Bend Sensor

The bend sensor was the first device explored in the first FlexM hub prototype because of its ability to read an angle bend and its unobtrusiveness in structure (Figures 3). Two issues precluded using it in subsequent FlexM hub prototypes. The bend sensors gave inconsistent discrete readings and the sensor readings changed over time due to mechanical fatigue. Furthermore, the readings appeared inconsistent between multiple bend sensors. This made calibrating the bending angle to the sensor complicated.

### 4.2.2 Sliding Potentiometer

Due to the inconsistencies in sensor readings with the bend sensor, we



Figure 14: (left) *Flexible hinge concept.* (right) *Sliding potentiometer measuring the angle bend.*

investigated the sliding potentiometer to measure the strut angle at the hub. Its implementation into the hub design marks a shift in the hub design. The previous designs involved cubes with fixed sockets—this minimized variables in resolving the connectivity issue. The hub design evolved from the rigid cube to the flexible hub with hinges. In the latest hub prototype, we

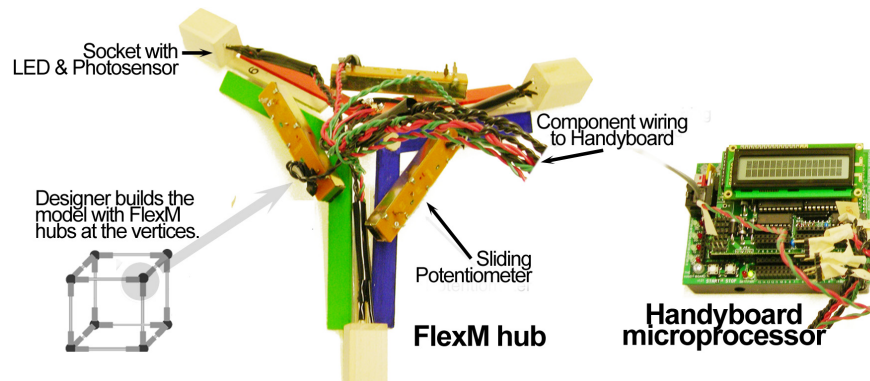


Figure 15: *Current hub design with sliding potentiometers in the hinges. The wiring connects the LEDs, photosensors and potentiometers to the HandyBoard.*

incorporated sliding potentiometers into the hinges (Figures 14).

Figure 15 shows the integration of the socket and its LED and photosensor with the sliding potentiometer at the hinge. The lower left diagram of the cube shows how the flexible hub fits in the model. Each hub has wires connecting it to the Handyboard, which reads the sensors and activate the LEDs. The Handyboard transmits the model's configuration via a serial port to a computer. On the computer, a receiving 3D graphics application displays the model.

This hub has three sliding potentiometers, one for each hinge. The three hinges facilitate the movement in three axes. Based mechanically on the hinge concept in Figure 14, each angle on the hub can move independently from the other, or be constrained by the other ones. If the potentiometers were not so bulky, the hinge could even bend from convex angles to concave angles. Although the sliding potentiometer added to the sturdiness of the hinge by acting as a knee brace, it also added unwanted bulk.

#### 4.2.3 Rotational Potentiometers

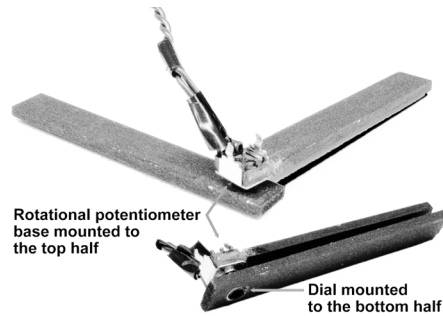


Figure 16: *Rotational potentiometer mounted to a hinge.*

Rotational potentiometers turned out to be the solution to the bulkiness of the sliding potentiometer. By allowing the potentiometer to be the mechanical hinge, we simplified the hinge design. Figure 16 shows the basic hinge module with half of the hinge attached to the base of the sliding potentiometer, and the other half attached to its dial. We are in the process of building a flexible hub with the rotational potentiometers.

#### 4.2.4 Mercury Tilt Sensors

Because the sensors to measure the hub connection only measure the angle, not the vector, orientation information is absent. Figure 6 illustrates how models with identical angles can have different configurations based on the direction of the angle. Figure 17 illustrates the difference between angle

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

versus vector measurement. On the left flexible hub, the bend of the hinge corresponds to the hinge's angle. On the right flexible hub, the hinge is a composite of two vectors. For the FlexM hub, we simplify the vector measure by breaking it down to an angle and orientation measurement.

We have incorporated mercury tilt switches in determining the

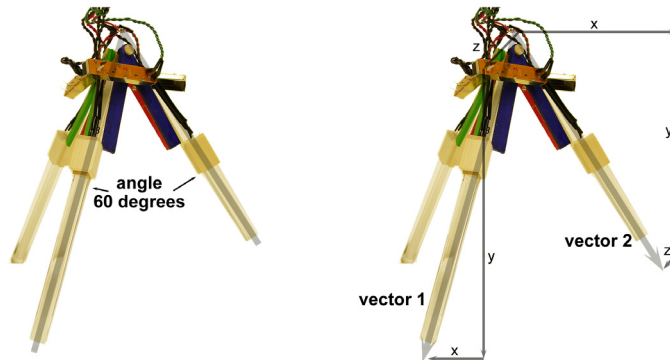


Figure 17: *Angle measurement versus vector measurement.*

orientation of the hub. The tilt switch is a small tube with a drop of mercury in it. In the standard vertical position, the mercury settles to the bottom, closing the circuit (the ON position) (Figure 18, left image). Likewise, the circuit is open (mercury does not make contact with the internal wiring) when the switch is upside down or sideways (the OFF position).

The wiring diagram for the mercury tilt switches is based on orientation studies for Navigational Blocks (Camarata et al. 2000). Navigational Blocks

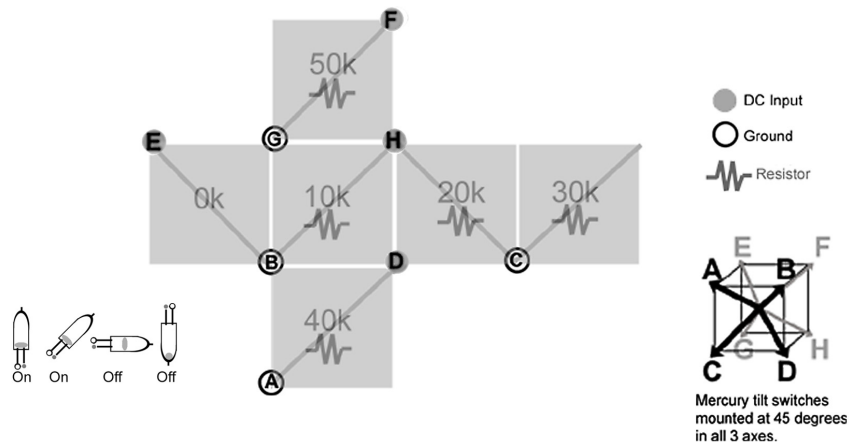


Figure 18: *Wiring diagram for mercury tilt switches.*

is a tangible user interface to query an informational kiosk. Each face of the block has a different category to look up information. The user queries the kiosk by rotating the cube so that the desired category faces up. For example, to look up people, the user places the block with the WHO face up. In response the kiosk displays the choices of people to look up.

The wiring diagram in Figure 18 shows the connections for the mercury tilt switches, laid out as a folded cube. The lettered circles represent a tilt switch. When you fold up the cube, the wiring configuration is set up to identify which side is facing up. No matter which side is facing up, only two tilt switches will be on. This is achieved by tilting the switches 45 degrees in all three axes.

The cube diagram in the lower right shows the orientation of the eight switches. In this configuration, only one pair of switches will be on in any of the six face orientations on the cube. Each pair of switches corresponds to a separate cube face (orientation). Because it takes two switches to be on, in order to close the circuit, only one pair of switches will return a resistance value. The computer knows which side is up based on the resistance value of the circuit, because each pair of switches has a different resistance value in the circuit. There are three circuits: switches E, B, H and C; switches A and D; and switches G and F. Because this wiring diagram is configured around a cube, it only identifies six discrete orientations. To measure continuous orientation values would require other techniques, for example using three gyroscopes to measure pitch, yaw and roll.

#### 4.3 PROGRAMMING FOR THE PHYSICAL MODEL INTERFACE

There are two programming components for FlexM: the physical model interface on the Handyboard and the graphics application on the host computer. The physical model interface collects the topology and geometry information and transmits them to the graphics application to render the digital model in real-time.

The goal of the programming for the physical model interface is to identify the model's topology and geometry. Photosensors and LEDs in the hub's sockets are used to identify how the hubs are connected to each other. The potentiometers at the hinges of the flexible hub measure the angles, and the tilt sensors measure the orientation of the hub. To illustrate the programming, we will walk through a simple example. First, we will explain how the flexible hubs in the physical model connect electronically to the Handyboard. Then, we will explain how the Interactive-C program in the Handyboard works.

##### *4.3.1 Programming to identify the model's topology*



## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

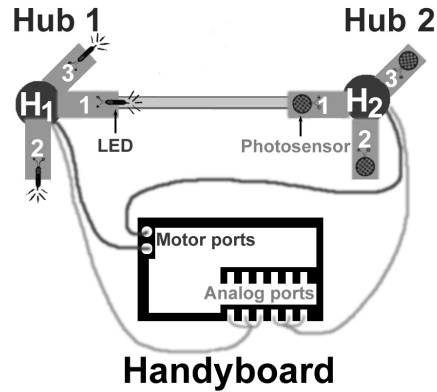


Figure 19: The three LEDs connect to a single motor port on the Handyboard. Each photosensor connects to its own analog port.

In Figure 19, hub 1 socket 1 connects to hub 2 socket 1. For each hub, the three LEDs are wired together and connect to a single motor port on the Handyboard, which turns the lights on and off. With this setup, the number of hubs is limited by the number of motor ports. The three photosensors, one for each socket, are wired separately and connect to separate analog ports, which read the resistance value from the sensors. This arrangement makes it possible to track the socket to the analog port. Likewise, the motor port maps to the hub.

In order to identify the topology the program's goal is to create a table of values in the format of: **hub A connects to hub B at socket X**. This is achieved by:

- 1) Initiate the Sending Hub.
- 2) Poll all other hubs as Receiving Hubs and record the connections in the table.
- 3) Repeat steps 1-2 for all the other hubs.
- 4) Send the table of hub connections to the host computer.

The Interactive-C program starts by turning on the LEDs in the first hub (hub 1) by setting the corresponding motor port value to 15. The program keeps track of this hub as the "sending hub" because it is the hub sending the light to the attached hubs.

While the LED's in the first hub remain on, the program polls the photosensors on the next hub (hub 2) via the Handyboard's analog port for a resistance value. The program tracks hub 2 as the "receiving hub". The resistance values range from zero (greatest amount of light) to 255 (no light). The photosensors return a zero resistance value, when the high intensity

LEDs shine at them. The program cycles through the photosensors in the hub's sockets for zero values. When it encounters a photosensor value of zero, the program stops polling the other sockets on that hub, since two hubs can only connect at a single connection. The program then records the connection information into a table. After checking all of the sockets on hub 2, it checks the photosensor values on hub 3, and all remaining hubs in the same fashion.

The program turns off the LEDs in hub 1 by setting the motor port value to zero. It goes to the next hub (hub 2) making it the "sending hub." All of the other hubs become "receiving hubs." The program cycles through the receiving hubs for zero photosensor values, and records the connection information. The program loops through all the other hubs as "sending hubs" compile the topology information.

In the last step, the Interactive-C program sends the table of hub connections to the host computer via a serial port.

#### 4.3.2 Programming for finding the model's geometry

The first part of identifying the geometry is measuring the angles at the hub. Although each socket is bound by two hinges, the socket angle measurement at the hubs is only one angle. The Interactive-C program has a table that correlates the sockets to the hinge: socket 1 relates to hinge A; socket 2 relates to hinge B; socket 3 relates to hinge C; etc.

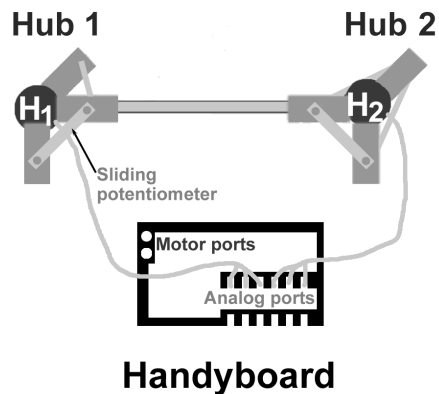


Figure 20: Each sliding potentiometer connects to its own analog port on the Handyboard.

These angles are derived from sliding potentiometers or rotational potentiometers. Figure 20 shows the flexible hubs with the sliding potentiometers. Each potentiometer connects to the Handyboard through a separate analog port. The potentiometers return a resistance value between zero (smallest hinge angle) to around 180 (hinge angle of 180 degrees). The

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

Interactive-C program converts the resistance value to their corresponding angle in the range of 1 degree to 180 degrees. Because each hinge maps to a specific analog port, the Interactive-C program knows the angle of each hinge.

After the program sends the host computer the topology information (section 4.3.1), it sends the angle information in the format: **socket x has angle A**. The program steps are:

- 1) Initiate the process by sending a zero value to signal the beginning of the list.
- 2) Send the angle information for all the hinges.
- 3) Repeat step one, to signal a new list.
- 4) Compare the old angle with the new angle for each of the hinges. If the angle has changed noticeably, send the new angle values.
- 5) Repeat steps 3 and 4.

Step 4 reduces the amount of information transmitted to the host computer. No angle information is transmitted, if it has not changed from before.

### 4.3.2 Graphics Application Programming

The graphics application interfacing with the FlexM model is FormWriter (Gross 2001), a LISP application for 3D geometry (Figure 21). It is designed for the non-programmer to generate 3D geometry with a simplified LISP scripting language. Users create geometry through rudimentary turtle-geometry based commands.

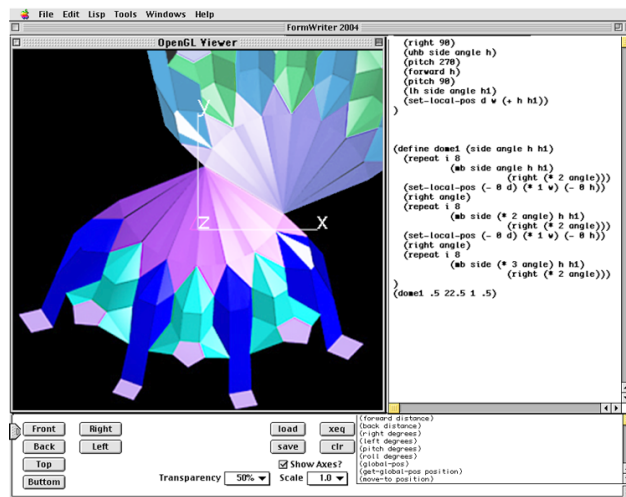


Figure 21: *FormWriter* graphics program.

## 5. DESIGN

We have built several prototype hubs in designing the computationally enhanced hub and strut construction kit. Each design iteration focused on a separate aspect of the kit. This section outlines how we will integrate the parts to build the complete kit.

The flexible hub will have three articulating arms, one for each socket connection. Each socket will have a high intensity LED and a photosensor for identifying the model's topology. The sockets will be attached to two hinges as shown in Figure 15. The hinges will have rotational potentiometers to serve as the mechanical hinge and measure the angle of the hinge.

The goal of future prototypes is to reduce the size and production time, and minimize the size for ease of use. Past prototypes have been hand crafted with basswood, but the future hub will be fabricated out of plastic through rapid prototyping to reduce assembly time. Due to the wiring and the sliding potentiometers, the prototype appears cumbersome—more rigid than flexible. Future designs will incorporate the less bulky rotational potentiometer. The tangling of wires tethered to the Handboard microprocessor and the limitation of analog and motor ports will be replaced by the smaller BasicStamp microprocessor. Communication between hubs will be through radio frequency in place of the wiring to the Handyboard.

With FlexM, building a digital model with physical components is easy and intuitive. Its struts can come in a range of lengths and materials to represent wood, steel, glass, etc. Furthermore, the flexible hinges allow study and exploration in animating geometrical transformations. The FlexM model could also interface to a structural engineering or molecular modeling program. As the engineer investigates the form in hand by manipulating it, he could receive the benefit of analysis from the simulation on the screen. Designers can explore rigidity and structural integrity for curvilinear or non-orthogonal structures, such as those described in the book "Twist&Build" (Vollers 2001). One could map a FlexM model to a molecular structure to explore protein folding.

In conclusion, the FlexM construction kit follows in the toy-like spirit of other hub and strut construction kits. It is not just a toy, but a tool through its interface with a computer, which renders a digital version of the FlexM model in real-time. It can also interface with engineering application that add the benefit of feedback and analysis.

## 6. ACKNOWLEDGEMENTS

We would like to thank the members of the Design Machine Group (DMG) for their support and dialogue of ideas throughout this project. We are especially grateful to DMG member Mike Weller for his enthusiasm and

## FLEXM: A FLEXIBLE DESIGN CONSTRUCTION TOY

contributions. This research was supported in part by the National Science Foundation under Grant CCLI-0127579 and ITR-0326054. The views and findings contained in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.

### 7. REFERENCES

- Aish, R., 3D Input for CAAD Systems. *Computer-Aided Design*, 11(2):66-70, Mar. 1979.
- Anderson, D, Frankel, J., Marks, J., Agarwala, A., Beardsley, P., Hodgins, J., Leigh, D., Ryall, K., Sullivan, E., Ydida, J, Tangible Interaction + Graphical Interpretation: A New Approach to 3D Modeling. In Proc. of SIGGRAPH 2000: 393-402. 2000.
- Camarata, K., Gross, M., Do, E. Navigational Blocks: navigating information space with tangible media. In Proc. of ACM Conference on Intelligent User Interfaces (2003).
- Dewey, D., and Patera, A. Geometry-defining processors for partial differential equations. In B. Alder, editor, *Special Purpose Computers*, pages 67-96. Academic Press, 1988.
- Erector Sets. <http://www.ideafinder.com/history/inventions/erectorset.htm>
- Frazer, J., Frazer, J., and Frazer, P., New developments in intelligent modelling. In Proc. of Computer Graphics 81, pages 139-154. Online Publications, 1981.
- Esposito, C., Paley, W. B., and Ong, J. Of mice and monkeys: A specialized input device for virtual body animation. In Proc. of Symposium on Interactive 3D Graphics, pages 109-114, 213, Monterey, California, Apr. 1995.
- Gorbet, M., and Orth M. Triangles: Design of a Physical/Digital Construction Kit. In Proc. of the Symposium on Designing Interactive Systems 1997: 125-128. 1997.
- Greenberg, S. and Fitchett, C. Phidgets: Easy development of physical interfaces through physical widgets. In Proc. of the ACM UIST 2001 Symposium on User Interface Software and Technology, November 11-14, Orlando, Florida. ACM Press. [www.cpsc.ucalgary.ca/grouplab/papers/http://Phidgets.com](http://www.cpsc.ucalgary.ca/grouplab/papers/http://Phidgets.com)
- Gross, M. FormWriter: A Little Programming Language for Generating Three-Dimensional Form Algorithmically. In Proc. of CAAD Futures 2001, Eindhoven, 8-11 July 2001, pp. 577-588.
- Hoberman, C. Faltstrukturen für temporäre Gebäude (Temporary Unfolding Structures). *Detail*, Dec. 1996, vol. 36 no. 8, pages 1184-1185.
- Lertsithichai, S. and Seegmiller, M. CUBIK: A bi-directional tangible modeling interface. In Proc. of the Conf. on Human Factors in Computing Systems, CHI 2002, pages 756-757. 2002.
- Persistence of Vision Raytracer. <http://povray.org>.
- Raffle, H., Parkes, A. and Ishii, H. Topobo: A constructive assembly system with kinetic memory. In Proc. of the ACM CHI 2004, April 24-29, Vienna, Austria, ACM Press. <http://tangible.media.mit.edu/>. 2003
- Tinker Toys. <http://www.yesterdayland.com/popopedia/shows/toys/ty1079.php>
- Vollers, K., *Twist&Build: creating non-orthogonal architecture*. 010 Publishers, Rotterdam, The Netherlands. 2001.