Posey: Instrumenting a Poseable Hub and Strut Construction Toy

Michael Philetus Weller
CoDe Lab
Carnegie Mellon University
Pittsburgh, PA, USA
philetus@cmu.edu

Ellen Yi-Luen Do
ACME Lab
Georgia Tech
ellendo@cc.gatech.edu

Mark D Gross
CoDe Lab
Carnegie Mellon University
Pittsburgh, PA, USA
mdgross@cmu.edu

ABSTRACT

We describe Posey, a computationally-enhanced hub-and-strut construction kit for learning and play. Posey employs a ball and socket connection that allows users to move the parts of an assembled model. Hubs and struts are optocoupled through the ball and socket joints using infrared LEDs and photosensors. Wireless transmitters in the hubs send connection and geometry information to a host computer. The host computer assembles a representation of the physical model as the user creates and configures it. Application programs can then use this representation to control computational models in particular domains.

Keywords

Construction Kits; Tangible; Poseable; Toys

Categories and Subject Descriptors

K.3.1 [**Computers and Education**]: Computer Uses in Education; H.1.2 [**Models and Principles**]: User/Machine Systems; H.5.2 [**Information Interfaces and Presentation**]: User Interfaces, Input devices and strategies

INTRODUCTION

Posey is a computationally-enhanced poseable hub-and-strut construction kit. Its hub-and-strut form maps to model anything that can be described in a graph structure, for example an articulated skeleton, a chemical molecule, a kinematic linkage or a building structure. Figure 1 shows our friend Amanda using Posey to create and animate a bear character. We begin with a discussion of traditional construction kit toys, deferring treatment of related work in computationally enhanced kits until later in the paper.

Construction kits display a variety of qualities that have endeared them to several generations of children and educators. Foremost among these qualities is that children enjoy playing with them, and through play they are introduced to the rewards of working out their own plans and dreams in a tangible medium that demands the resolution of issues of form and structure. Working out design problems with construction kits builds both general problem-

TEI 2008, February 18-20, Bonn , Germany.

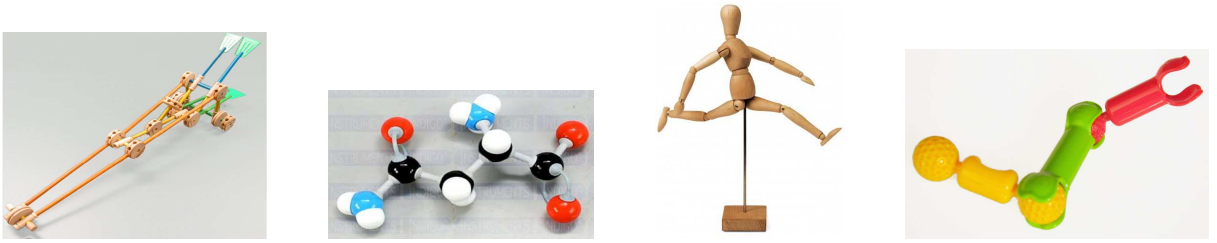
Figure 1: Assembling a bear puppet with Posey and our Puppet Show application.

solving skills and strengthens a child's ability to think and imagine in three dimensions. Construction kits are also used to model the properties of particular domains. For example Lincoln Logs help children understand and explore the design space of log buildings; plastic molecule kits (Figure 2(b)) help students build a robust understanding of the geometry of molecules and the inter-atomic forces that govern their formation. And perhaps most importantly the open-ended nature of construction kit play introduces children to the hacker ethos of building things as a way of having ideas.

A canonical instance of a poseable hub-and-strut kit is Tinkertoy, (Figure 2(a)) which consists of circular wooden hubs with holes drilled in the center and around the circumference and wooden struts of varying lengths. Many hub-and-strut construction kits such as Zome Tool[1] have fixed connection angles between hubs and struts. While a Tinkertoy hub fits tightly onto a strut inserted into one of its perimeter sockets, a hub spins freely on a strut inserted into its center socket, allowing the construction of kinematic models. Tinkertoy demonstrates several of the qualities of the hub-and-strut construction kit form that we believe makes it a particularly interesting tangible interface:

1. The rules for assembling pieces are straightforward and simple to grasp: any strut connects to any hub in the same way. This accessibility promotes experimentation. In contrast, Lego's connection rules are more complex: bricks fit together in several different ways, and connecting two bricks creates subtle constraints on future operations. Lego models are of-

[1] `http://www.zometool.com`



(a) a vehicle built with Tinkertoy

(b) model of Asparagine with Molymod kit

(c) lay figure for exploring human poses

(d) the ball and socket connection of the ZOOB construction toy affords posing

Figure 2: Some examples of uninstrumented hub-and-strut construction kits.

ten realized by following step-by-step instructions.

2. Hub and strut kits lend themselves to building kinetic poseable models. With Tinkertoy in particular, the combination of the ability to rotate certain connections and the scale of the struts invites animating models through physical manipulation.

3. The graph structure of a hub-and-strut model can be mapped to a wide variety of domains.

These properties of the hub-and-strut interface have already been leveraged for several domain-specific tools. Chemistry modeling kits such as Molymod, shown in Figure 2(b), take advantage of the first property to allow students to quickly explore the space of possible organic molecules. Although they are not exactly a construction kit, so-called "lay figures" like the one shown in Figure 2(c) take advantage of the second property to allow the exploration of poses of the human figure.

Since Aish first proposed that physical blocks could be used as an interface for describing 3D models to a computer [1] researchers have constructed a variety of these sorts of kits. By enabling applications running on a personal computer to monitor the topology and configuration of assemblies as they are manipulated these computationally-enhanced construction kits promise to allow software developers to create novel and accessible environments for undirected play. These applications can improve on traditional construction kits in two ways: they can map additional domain-specific information to the current assembly, and they can provide feedback in response to manipulations of the assembly. Systems such as Computational Building Blocks [2], and others discussed below, add computational power to traditional physical construction kits, lending themselves to creative exploratory and 'constructionist' play.

Until now there have been no poseable kits instrumented to capture both the topology and geometry of the model. Prominently, the Topobo project [16, 17] (which is discussed further in Section 5) has demonstrated the potential of poseable construction kit interfaces, but most of the pieces are passive and do not collect information about geometry or topology of the model. We set out to design a kit that would allow connection angles between all hubs and struts to be changed after an assembly has been constructed, and could sense these changes in real time to support adjusting a model's pose as an interaction method. By mapping such a model's three dimensional forms to a domain we can create custom input devices that provide simple intuitive control of complex models.

We were inspired by the ZOOB (an acronym for Zoology, Ontology, Ontogeny, and Botany) construction kit shown in Figure 2(d) to use a ball and socket joint to connect Posey's hubs and struts. A ball and socket joint is easy to connect, is mechanically simpler than a hinge, and provides a wide range of motion. A ball and socket joint can also be designed to hold its position, making assemblies easily poseable.

In the following section we discuss two applications we have built to illustrate the sort of interactions that Posey can support. Next we will turn to the system design and implementation details. We then discuss related work and contrast Posey with existing construction kits. Finally we discuss Posey's contribution to tangible interaction design.

APPLICATIONS

We have built Posey as a general-purpose modeling kit that programmers can use to build applications in various domains. We have concentrated on the physical, electronics, and software design of the Posey kit. However, to test and demonstrate the Posey kit and its API we built two simple applications: A Puppet Show animator for controlling animated characters and a Molecule Explorer for playing with physical chemistry.

Puppet Show

Our puppet show application demonstrates the two stages of interaction afforded by Posey. Puppets are created by assembling a skeleton out of hubs and struts, and then this skeleton can be used as a tangible interface to animate the puppet once it has been skinned from a library of puppet parts.

The onscreen interface (Figure 3(b)) presents two windows, a large 3D view window that shows the skinned animated puppet in real time, and a menu on the right that displays the available puppet parts. The following scenario illustrates the steps involved in building and animating a puppet:

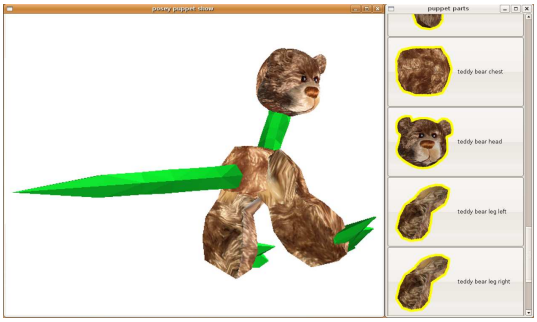
1. Amanda decides to build a bear puppet. When she connects a new hub or strut to the model its representation appears on the screen.

2. Amanda begins to skin her puppet by selecting parts from the puppet part menu. In Figure 1 she has attached a bear head part to her model.

3. Amanda notices the dinosaur tail and decides to make a "bearosaur" instead. She pulls apart the bear's torso and adds a long tail instead, as shown in Figure 3(a).

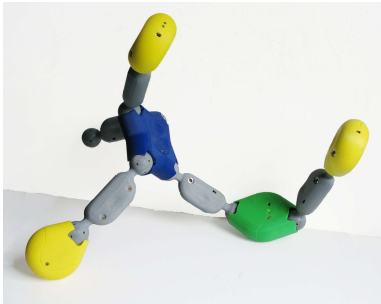
4. Amanda animates her skinned bearosaur model by manipulating the Posey model to rear up and lunge forwards in a fearsome attack.

(a) skeleton model of bearosaur

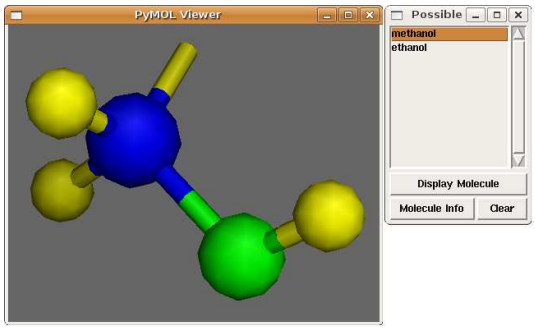


(b) screen shot of bearosaur being animated with model

Figure 3: Mixing puppet parts to build a bearosaur.



(a) incomplete model of methanol molecule



(b) Molecule Explorer interface showing methanol is missing one hydrogen

Figure 4: Constructing a model of methanol with the Posey Molecule Explorer.

Molecule Explorer

The Molecule Explorer is built on top of the PyMOL Molecular Graphics System [4] to use Posey for elementary chemistry modeling. In the Molecule Explorer, Posey hubs represent atoms and struts represent bonds between the atoms. The number of sockets on a hub represents the atom's valence; thus a Posey 1-socket hub represents hydrogen, a 2-socket hub can represent oxygen, a 3-socket hub, nitrogen, and a 4- socket hub can stand for carbon (in the present version, Posey's 4-socket hubs are flat, but it would be simple to manufacture a tetrahedral hub).

The Molecule Explorer onscreen interface comprises two windows (Figure 4(b)), a view window that displays a 3D rendering of the currently selected molecule, and a possible molecule window that lists molecules of which the current Posey model is a subgraph. In the following scenario the students of a high school chemistry class investigate the structure of several organic molecules using Posey and the Molecule Explorer:

1. Felix and Patricia are working with a Posey kit. Patricia suggests that they start with a carbon atom, and hands a four-hub to Felix, who connects it to a hydrogen atom. The "possible molecules" window updates to show only molecules containing a CH group.

2. Patricia selects methane from the list, and the molecule is displayed in the 3D view window. The carbon and hydrogen molecules that are already part of the module are rendered as balls, while the rest of the molecule is rendered as sticks as an aid for students to construct the rest of the molecule.

3. Patricia points out that they need to connect three more hydrogen atoms to build methane. Felix snaps the pieces together and the model of methane on the screen is fully populated with balls, showing that the model is complete.

4. Felix asks Patricia to check out the Wikipedia page. A button on the possible molecules window loads the Wikipedia page for the currently displayed molecule in a web browser. Felix scans the page and is excited to learn that methane is produced by cattle belching.

5. Eager to move on, Patricia grabs the Posey model and removes a hydrogen atom. The "possible molecules" window updates to show several other related molecules they could build and she selects methanol. The view window shows that they need to add an OH group to their model to complete the molecule.

6. Patricia connects an oxygen and hydrogen to the Posey model (Figure 4) and the view window shows that the methanol molecule is complete. Felix quickly clicks to load the Wikipedia page for methanol.

SYSTEM DESIGN

The goal of the project is a kit that invites exploration and provides a wide range of movement, instrumented so that it can sense both which pieces are connected and how they are being manipulated, and communicate this information back to a host computer to provide a tangible 3D modeling interface for applications.

(a) strut (b) one-hub (c) two-hub (d) three-hub (e) four-hub

Figure 5: A Posey kit is composed of struts with a ball at each end and hubs with from one to four sockets.

Component Geometry

We have adopted ball and socket connectors for our construction kit as they are straightforward to connect and disconnect, provide a wide range of motion and are capable of holding their position once posed. A Posey kit is composed of several kinds of components, shown in Figure 5. Hubs come in several forms with from one to four sockets to accept the ball of a strut. A strut is a straight segment with ball connectors at each end that can connect various hubs. The sockets of hubs with more than one socket are arranged so that two hubs can be connected by multiple struts, which makes it possible to construct a wide variety of graphs.

3.1.1 Variation in component geometry

Our simple set of parts includes only one length of strut and four hub geometries. However, we could make struts in several different lengths, and hubs with sockets in different geometries (e.g., a 4-socket tetrahedron, a right-angle 2-socket hub). At present the number of input-output pins on our microprocessor limits us to four sockets, but we could easily replace this processor with one that has more IO pins.

Data Flow

Figure 6 illustrates the data flow from the hubs and struts of a Posey model to an application running on a personal computer. There are three distinct layers of communication:

1. Struts transfer data to hubs over a serial optocouple.
2. Hubs transfer data to the host computer via a Zigbee wireless radio transceiver.
3. The driver software processes raw data received from hubs and places events on a stack for the application to handle.

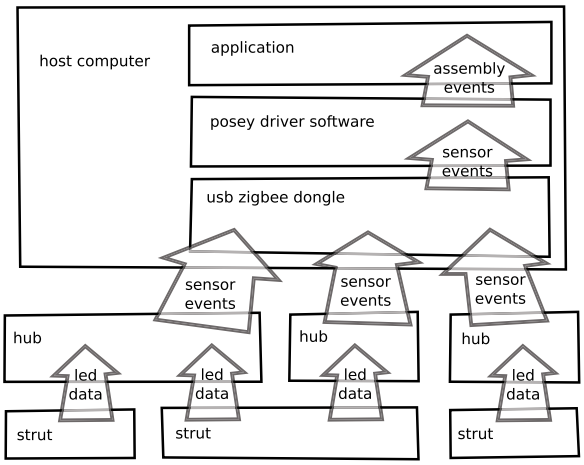


Figure 6: Data flow from Posey model to application on a host computer.

SYSTEM IMPLEMENTATION

Posey is the most recent of our efforts to build a poseable hub and strut construction kit. An earlier and clumsier construction kit we built, FlexM [6], uses hinges and straight sockets that do not permit twisting struts in the hubs. The angle between hub and socket was sensed with potentiometers mounted on the hinges. People who played with FlexM complained that many shapes could not be constructed because the struts are keyed to the sockets and could not be rotated to arbitrary angles. They also complained that the hubs were not inviting to handle, and that models did not hold their pose. And they were dissatisfied with long latency times resulting from FlexM's data transfer model that required the host computer to constantly poll all the hubs.

To address these issues Posey takes an entirely different, and significantly more robust and versatile, approach. With Posey we have adopted a ball-and-socket joint that can connect at a wide range of angles and holds its pose. We have designed the forms of the components to be inviting to handle. We replaced the hinge-potentiometer assembly with an array of optocoupled infrared LEDs and phototransistors embedded in the ball and socket joint. Each part has a microcontroller that drives actuators (the struts' infrared LEDs) and monitors sensors, and when hubs detect a change they initiate communication with the host computer to report it.

In the following, we begin by describing the mechanical design of the individual hubs and struts. We then discuss the electronics embedded in each part. Next we describe our method for sensing the topology and geometry of the Posey model. Finally we describe our event interface for application developers.

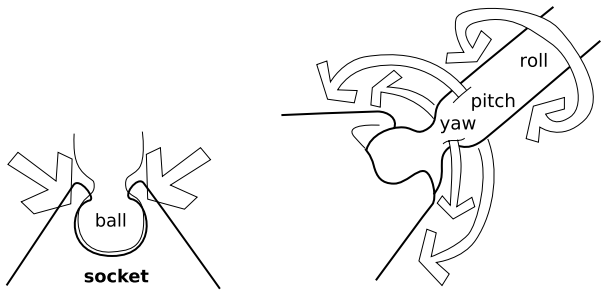
Mechanical Design

The plastic shells for Posey's hubs and struts are printed out of ABS plastic on a Dimension SST^{TM} fused deposition modeler. Each shell is composed of two hollow sections with mounting brackets inside for electronics, as shown in Figure 7.

Ball-and-socket joint

By making the lip of the socket curve in slightly so that the ball must stretch the lip to fit inside, as shown in Figure 8(a), the socket exerts a force on the ball that holds it in place. The tolerances of this overlap must be designed to take into account the material elasticity of the ball and strut. Posey parts are printed in ABS plastic and to grip our 30mm diameter ball we have designed the lip of the socket to extend over the ball by 0.5mm.

Figure 7: Strut shell open to show electronics inside.



(a) cross section showing socket's grip on ball

(b) ball has three degrees of freedom in socket

Figure 8: Posey ball and socket joint.

Range of Motion

Our ball and socket joint has a wide range of motion along three degrees of freedom, as illustrated in Figure 8(b). The ball can roll all the way around in the socket. It can pitch up and down across a 200 degree range, and it can yaw 150 degrees from one side to the other.

Electronics Design

To capture the geometry of a Posey assembly each hub and strut has an embedded processor and electronics. Both hubs and struts have the same control board with an AVR ATmega168 microprocessor, headers exposing the microprocessor's 23 input-output pins and headers to attach power and communication daughter boards. Every part also has a battery daughter board that regulates power from a nominal 3.7 volt lithium-ion battery down to 3.3 volts. Hubs also feature a communication daughter board with an XBEE Zigbee wireless radio transceiver.

Sensing Topology and Geometry

As illustrated in Figure 6 sensor events describing changes in the state of a hub's socket connection are sent over a Zigbee link to the host computer. Our pyposey software library parses this event data to generate a representation of the current topology and geometry of the Posey physical model.

Hub and strut serial optocouple

The balls of each strut are populated with an array of infrared LEDs, and each LED constantly broadcasts a unique three byte address, with the first two bytes identifying the strut and the last byte identifying the individual LED. Each socket of each hub has an array of photosensors that each watch for an infrared serial data stream. The LEDs and photosensors are arranged so that when the ball of a strut is inserted into the socket of a hub, at least one photosensor can see an LED.

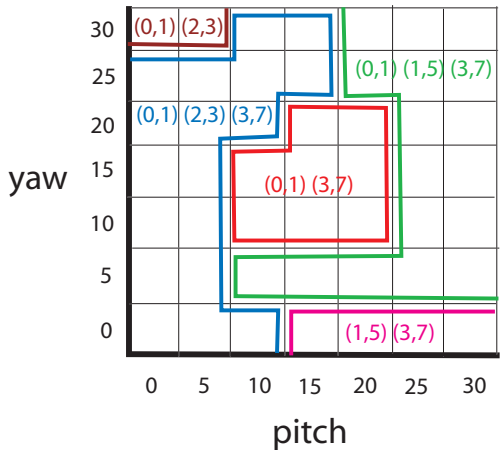


Figure 9: Sectional slice through 3D table relating observed LED-sensor pairs to roll, pitch, and yaw angles. The pairs in each block indicates which hub photosensors can see which strut LEDs.

Hub to host computer Zigbee radio link

When each hub is powered up its Zigbee transceiver initiates a link with the host computer. Hubs track whether each photosensor in each socket is connected to an LED and if it is, also track the connected LED's address. Every time a photosensor receives a new data packet it is compared with the stored state. When there is a change the hub generates a sensor event with the address of the sensor and LED and sends it over the Zigbee link to the host computer.

Inferring angles from Observed LED-Sensor Couples

Depending on the roll, pitch, and yaw of a ball in its socket, as few as one and as many as all four of the socket phototransistors will see an LED. Thus each distinguishable 3-dimensional angle is described by a connection-set with up to four elements, in which each element consists of a pair:

```
(<photo-transistor-id> <LED-id>)
```

For example, the connection-set:

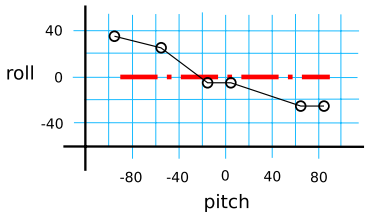
```
[ (hub-1.3.2 strut-4.1.9),
  (hub-1.3.5 strut-4.1.2) ]
```

indicates that two of the phototransistors in hub #1's third socket (#2 and #5) see LEDs from strut #4's first ball (LEDs 9 and 2).

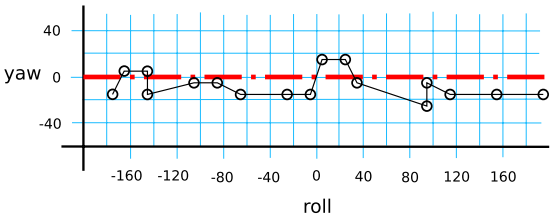
Upon receiving these connection-sets from each hub, the host computer uses a lookup table to translate a connection-set to a three dimensional (roll, pitch, and yaw) angle as shown in Figure 9.

Angle sensing precision

To illustrate the quality of the event data produced by our software drivers we have generated two graphs (Figure 10) showing a typical path resulting from rotating a strut in one axis. The thick dashed lines show the actual rotation executed and the circles show the individual angle position events generated by the driver. The



(a) path generated by pitching from -90 degrees to 90 degrees



(b) path generated by rolling 360 degrees

Figure 10: Graphs of typical paths generated by rotating a strut in one axis. Circles are individual events generated by driver software; the thick dashed line represents actual path of movement.

distance between the thick dashed lines and the line connecting the circles is the result of imprecision in the lookup table. With a few exceptions the path generated by the system is within 20 degrees of the actual path. Although this is a significant margin of error, this is largely due to our current relatively sparse hand-built sensor and LED arrays with a spacing of more than 60 degrees and not an inherent limitation of this angle sensing system.

One feature of the data that is evident from the graph is that events tend to cluster. This is because although the LEDs are uniformly distributed around the ball, the coupling of LEDs to sensors is not uniform. Some areas of the angular 3-space have a denser clustering of couples than others, and when passing through those areas events are generated more frequently.

Event Interface

The pyposey software driver library polls the Zigbee interface for sensor events and tracks the state of all of the parts currently in the Posey model. It then generates higher level assembly events and places them on an event stack. Pyposey exposes two interfaces for application development. The Hub Demon API simply exposes the assembly event stack. The GL Graph API sits on top of the Hub Demon and maintains a graph of subclassable hub and strut nodes to make it easy for developers to create software that uses Posey for input.

The event model structure is that hubs are instantiated when they are attached to a strut, and are garbage collected when they are no longer connected to a strut. Struts only exist as edges between one or more hubs. The geometry of a model is expressed as the roll, pitch and yaw of each strut relative to its attached hub socket.

Hub Demon API

The Hub Demon is written in Python. When it is instantiated it takes as an argument a Queue, a multi-producer, multi-consumer, thread-safe data structure from the Python standard library, and starts up in a separate thread. Sensor events are processed as they are read in from the Zigbee link and are translated to one or more of the five assembly events that can be placed on the event queue to communicate the current state of the Posey model to the application:

create new hub connected to assembly
address address of new hub
size number of sockets in hub

destroy hub removed from assembly
address address of hub removed

connect new strut connected to hub
hub address of hub connected to
number socket number connected to
strut address of strut connected to

disconnect strut disconnected from hub
address address of hub
number socket that is disconnected

configure strut angle changed in socket
address address of connected hub
number index of socket measured from
roll roll angle value
pitch pitch angle value
yaw yaw angle value

Hub Demon Implementation

We read from the Zigbee link through a serial port interface and parse the XML text stream into sensor events. The topology events, create, destroy, connect and disconnect, are all generated directly by the Hub Demon from these sensor events. As new hubs are connected to the assembly the Hub Demon also creates Socket Demons to track the connection angle of each socket of each hub.

Sensor events containing LED-phototransistor coupling data are passed on to the Socket Demon monitoring the socket containing the phototransistor that generated the event. Each Socket Demon then generates dynamic configuration events for its socket by inferring the position of the connected strut from the current coupling of LEDs and phototransistors.

GL Graph API

The pyposey library also provides a higher level graph API that handles create, destroy, connect, disconnect, and configure events, maintains a graph of hub and strut nodes, and renders the graph with OpenGL. Developers can extend this graph model by providing their own subclassed hub and strut node classes.

RELATED WORK

Many have observed the rich tradition of physical construction kits in children's play and learning that dates back to Froebel's "kindergarten gifts" [3, 15]. External representations of ideas is central to Papert's constructionist view of learning: early implementations of the children's programming language Logo employed a physical robot turtle, later abstracted to a "screen turtle", to draw pictures [13]. Resnick and others at the MIT Media Lab have advanced this approach, using physical media to express and explore computational ideas [12, 18, 19]. Like others we have been developing an agenda for computationally-embedded construction kits as well as making the agenda concrete through working prototypes that expose various aspects of these kits [5].

It is difficult to find concrete demonstrations of the educational efficacy of computationally-enhanced construction kits, partly due to the rapidly changing technology landscape of this area of research. Most work so far has been exploratory in nature, revealing

approaches to building tangibly-interactive construction toys that exploit different hardware or software architectures, or different task or domain areas. Our Posey project follows this exploratory approach as well, focusing on the affordances of a hub-and-strut construction modeling scheme for using abstract graph models to express domain concepts.

Poseability and Interactivity

A construction kit has parts that users can assemble in different configurations. We view the design space of these kits along two dimensions: poseability and interactivity. Poseable means that once a particular configuration has been assembled its parts can be moved or posed locally, that is, relative to one another. An interactive kit is equipped with sensing and communication so that assembling and posing operations performed on the model are reported to applications in real time to allow them to respond.

We found kits that enjoy one or the other of these properties, but not both. For example, many traditional (i.e., noncomputational) construction kit toys are poseable but of course not interactive. Computational Building Blocks [2] (mentioned again below) are fully interactive, they detect when and where bricks are added to and removed from a model, but are not poseable. Topobo [16, 17], (also mentioned again below) is poseable, but is not instrumented to detect the topology of the current model.

Static Construction Kits

Various static construction kits enable users to create configurations of computationally-enhanced parts, which a host computer senses. An early effort, Aish's three dimensional Building Block System [1] enabled architects to input models to a CAD system. Frazer et al.'s 3D input devices [8] enabled designers to build models that interface with software that can give design advice. Computational Building Blocks [2] facilitate computer modeling with instrumented snap-together plastic blocks. ActiveCubes [20] capture 3D configuration information of computationally-enhanced physical blocks. In Triangles [10], a construction kit of flat plastic triangles that interface to a computer, each tile corresponds to a different application, such as an email client or a personal calendar; in a later version, to a character or object in a story. Mechanical and electronic magnetic connectors enable constructing a variety of geometric forms that correspond to a suite of applications. Smart Blocks [9] are small cubes that provide a tangible interface for learning about geometric properties such as area and volume. Glume [14] is composed of soft modules each with six stubby arms filled with hair gel that communicate conductively to determine the topology of the overall model. Glume presents an interesting and original tactile quality, but geometry is not directly sensed, only inferred from topology. In contrast, Posey supports modeling not only topology but specific 3D geometries. Although the forms of these kits differ, in each the forms a user can make are static and rigid.

Poseable Construction Kits

Another, somewhat smaller, genre of computationally-enhanced construction kits includes those in which the physical model is poseable, that is, once the user constructs a model the computer can sense adjustment of relative positions of connected parts.

Monkey [7] is a specialized input device for virtual body animation. It is basically a lay figure like the one shown in Figure 2(c) instrumented for use as an input device. 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. However, while Monkey demonstrates the potential of this type of tangible interaction, it cannot be reconfigured and can only be used to control one particular (humanoid) geometry. In contrast, with Posey we can build skeletons for non-humanoid models such as spiders or molecules or bridges.

Topobo [16] is a construction kit of articulating vertebralike pieces for building models with embedded kinetic memory. A Topobo construction is composed of a few active hubs with sensing, communications and actuation, and a variety of passive limb components that attach to the hubs and each other to construct a model of a creature. Each hub records angular movement at its joints when a button is pressed, and then replays the same movement in a loop with its motors after the button is released. Users build a creature, move the model across a terrain, and then watch the model replay its movement from its embedded kinetic memory. The Backpacks project [17] extends Topobo by adding timers and sensors that can modulate the reproduction of recorded movements. These projects demonstrate how attractive a kinematic construction kit interface can be, but neither Topobo or its Backpacks have sensors to detect the topology or geometry of most of the pieces in a kit.

Senspectra [11] is a hub-and-strut construction kit composed of balls with several embedded female headphone jacks that serve as hubs and flexible struts that have male headphone jacks at each end. The struts bend to make it possible to connect hubs and struts in a wide variety of configurations, but are not poseable. Sensors in each strut detect the degree, but not the direction, of bending. Although Senspectra lacks the 3-degree of freedom sensing of the geometry of the connection between each part in a model provided by Posey, in models composed of large numbers of hubs and struts the additional geometric constraints imposed by the topology allow Senspectra to estimate the overall geometry of the model.

Although these projects show the appeal of hub-and-strut kits as a tangible user interface, none has addressed the potential of poseable connections as a mode of specifying 3D form. While Senspectra also supports interaction through kinematic manipulation of a model, its springy struts snap back into place afterwards. Poseable connections support a sculptural medium that can be used to describe a wide range of forms.

USER FEEDBACK

We have demonstrated Posey to a variety of visitors to our lab, including a group of high school students enrolled in a local after school program, and at a meeting of Dorkbot Pittsburgh. Although we have not conducted controlled experiments, we have had the opportunity to observe people interacting with Posey and listen to their feedback on the experience.

We found that people were very drawn to interact with the Posey components. More than 20 high school students waited in line for their chance to play with the kit. We also found that people had no problems mapping their manipulation of the physical model to the behavior onscreen and could animate a puppet as soon as they picked up the model. We observed several people attempt to move or reorient their puppet onscreen by physically moving and rotating the entire puppet, which suggested to us an extension of the interface using accelerometers to capture these gestures.

The overall reaction to Posey has been very positive. Many people identified the playful shapes and satisfying haptic experience of connecting and disconnecting the sockets as being particularly successful aspects of the project. We have also had several visitors return with friends that they thought would also enjoy playing with Posey. The enthusiasm we have seen suggests to us that Posey has identified a fertile space for tangible interfaces.

FUTURE WORK

Applications

Our two demonstration applications show that Posey and its API can be used to model different domain tasks, but we would like to support development of several more applications, to learn what additional Posey features might be desirable. One such application would be a structural modeler for studying structural truss designs used for example, in building bridges. Once the physical model is captured, the application would display the tension and compression forces generated in a structural truss as it is loaded.

Angle Sensing

We will investigate methods for locating the LEDs and sensors to achieve a more uniform distribution of couples throughout the angular space. More accurate values for the strut angles could also be obtained by modeling the constraints imposed on the geometry of the struts by the topology of the assembly and by adding accelerometers to hubs to track their orientation.

CONCLUSION

Posey joins two strands of research into leveraging construction kits for tangible interfaces. The first is the use of construction kits to specify 3D geometry to an application on a host computer [1, 8, 2, 20]. The second is the use of construction kits to invite movement and physical interaction [7, 16, 17]. By combining these two affordances of construction kits Posey provides a straightforward and engaging mode of interaction suited to applications that have traditionally been inaccessible to novices.

Posey's interface also builds on existing computationally-enhanced construction kits. Like other kits [11, 14] Posey can be used to model the topology of graph structures, although Posey is distinguished by a particularly intuitive and haptically satisfying means of connecting its hubs and struts, a ball and socket joint. Posey extends this topology-specifying interface to also allow the connections to be posed in order to specify 3D geometries. In particular this interface supports a model-and-then-animate interaction illustrated by our Puppet Show application that demonstrates the potential to make 3D design more accessible.

ACKNOWLEDGEMENTS

We are especially indebted to Kumiyo Nakakoji (University of Tokyo and Software Research Associates) Kazuhiro Jo (University of Tokyo), and, for production assistance: Andrea Irwin, Jack Bader, Gabe Johnson, Shaun Moon, Jared Friedman, Greg Saul, Tiago Rorke, Yingdan Huang and Don Reeve (Carnegie Mellon). This research was supported in part by the National Science Foundation under Grant ITR-0326054.

REFERENCES

[1] Aish, R. 3d input for caad systems. *Computer-Aided Design*, *11*, 2 (1979), 66–70.

[2] Anderson, D., Frankel, J., Marks, J., Leigh, D., Ryall, K., Sullivan, E., and Yedidia, J. Building virtual structures with physical blocks. In *Proc. UIST*. ACM Press, (1999), 71–72.

[3] Brosterman, N. *Inventing Kindergarten*. H.N. Abrams, New York, NY, USA, 1997.

[4] DeLano, W. L. The pymol molecular graphics system. `http://pymol.sourceforge.net`.

[5] Eisenberg, M., Buecheley, L., and Elumeze, N. Computation and construction kits: Toward the next generation of tangible building media for children. In *Proc. of Cognition and Exploratory Learning in the Digital Age*. (2004), 423–426.

[6] Eng, M., Camarata, K., Do, E. Y.-L., and Gross, M. D. Flexm: Designing a physical construction kit for 3d modeling. *Intl. Journal of Architectural Computing*, *4*, 2 (2002), 27–47.

[7] Esposito, C., Paley, W. B., and Ong, J. Of mice and monkeys: A specialized input device for virtual body animation. In *Proc. of the Symposium on Interactive 3D Graphics*. (1995), 109–114.

[8] Frazer, J., Frazer, J., and Frazer, P. New developments in intelligent modelling. *Computer Graphics*, *81* (1981), 139–154.

[9] Girouard, A., Solovey, E. T., Hirshfield, L. M., Ecott, S., Shaer, O., and Jacob, R. J. K. Smart blocks: A tangible mathematical manipulative. In *Proc. TEI*. (2007), 183–186.

[10] Gorbet, M. and Orth, M. Triangles: Design of a physical/digital construction kit. In *Proc. Designing Interactive Systems*. ACM Press, (1997), 125–128.

[11] LeClerc, V., Parkes, A., and Ishii, H. Senspectra: a computationally augmented physical modeling toolkit for sensing and visualization of structural strain. In *Proc. CHI*. ACM Press, (2007), 801–804.

[12] McNerney, T. S. From turtles to tangible programming bricks: explorations in physical language design. *Personal Ubiquitous Comput.*, *8*, 5 (2004), 326–337.

[13] Papert, S. *Mindstorms: Children, Computers, and Powerful Ideas*. Basic Books, New York, NY, USA, 1980.

[14] Parkes, A., LeClerc, V., and Ishii, H. Glume: exploring materiality in a soft augmented modular modeling system. In *Ext. Abstracts CHI*. ACM Press, (2006), 1211–1216.

[15] Parkinson, E. Re-constructing the construction kit – re-constructing childhood: A synthesis of the influences which have helped to give shape and form to kit-based construction activities in the primary school classroom. *Intl. Journal of Technology and Design Education*, *9*, 2 (1999), 173–194.

[16] Raffle, H., Parkes, A., and Ishii, H. Topobo: A constructive assembly system with kinetic memory. In *Proc. CHI*. ACM Press, (2004), 647–654.

[17] Raffle, H., Parkes, A., Ishii, H., and Lifton, J. Beyond record and play: backpacks: tangible modulators for kinetic behavior. In *Proc. CHI*. ACM Press, New York, NY, USA, (2006), 681–690.

[18] Resnick, M., Bruckman, A., and Martin, F. Pianos not stereos: creating computational construction kits. *interactions*, *3*, 5 (1996), 40–50.

[19] Resnick, M., Martin, F., Berg, R., Borovoy, R., Colella, V., Kramer, K., and Silverman, B. Digital manipulatives: new toys to think with. In *Proc. CHI*. ACM Press, (1998), 281–287.

[20] Watanabe, R., Itoh, Y., Asai, M., Kitamura, Y., Kishino, F., and Kikuchi, H. The soul of activecube - implementing a flexible, multimodal, three-dimensional spatial tangible interface. In *Proc. Intl. Conf. on Advanced Computer Entertainment Technology*. ACM Press, (2004), 173–180.