Building Virtual Structures With Physical Blocks

David Anderson², James L. Frankel¹,², Joe Marks²,
Darren Leigh³, Kathy Ryall³, Eddie Sullivan², Jonathan Yedidia²

¹Frankel and Associates, Inc., Lexington, MA 02421
²MERL — A Mitsubishi Electric Research Laboratory, Cambridge, MA 02139
³University of Virginia, Dept. of Computer Science, Charlottesville, VA 22903

ABSTRACT
We describe a tangible interface for building virtual structures using physical building blocks. We demonstrate two applications of our system. In one version, the blocks are used to construct geometric models of objects and structures for a popular game, Quake II™. In another version, buildings created with our blocks are rendered in different styles, using intelligent decoration of the building model.

KEYWORDS: Tangible user interfaces, transmedia.

OVERVIEW
Few people know how to use graphics modeling packages, but everyone can build things out of blocks. Starting from this premise, and with the long-term goal of developing an accessible modeling tool for building virtual worlds, we developed a novel object-modeling system comprising building blocks that self-describe the geometric structures into which they are assembled. Each building block contains a microcontroller, and is able to communicate with the blocks to which it is physically connected. The blocks in an assembled structure use a distributed algorithm to first discover how they are connected to their immediate neighbors. This information is then relayed from block to block — each of our block structures is essentially a self-configuring, store-and-forward computer network — until it reaches the host computer. From the block connectivity data that it collects, and knowledge of the shape of each block, the host computer recovers the geometric structure of the assembled blocks. The structure is then rendered in various styles, ranging from a literal rendition in which blocks look like Lego™ bricks, to decorative interpretations in which structural elements are identified automatically and decorated appropriately. Once described, the virtual structures are available for viewing by the user. Sensors and transducers in the blocks also allow the physical structure to serve as an I/O device for interacting with the virtual structure. Fig. 1a-e shows a physical block structure and sample renderings of the virtual model recovered from it.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

UIST '99. Asheville, NC
© 1999 ACM 1-58113-075-9/99/11... $5.00

RELATED WORK
The idea of a self-describing construction kit is not new: several groups have developed such systems over the past 20 years, beginning with the pioneering work of Aish and Frazer [1, 3], and continuing with more recent projects [2, 4].

Our self-describing building blocks resemble several previous systems, but differ in the following key ways:

- Lego™-like block design: Basing the physical design of our blocks on the ever-popular Lego™ brick has two distinct advantages. One is the structural rigidity and geometric constraint inherent in the design, which facilitates the recovery of 3D geometry from connectivity data. The more significant advantage is configurability: our blocks can be arranged in geometric structures of greater variety.

- Minimalist physical/electrical connection: To make our blocks more configurable requires the use of simple, symmetric connectors. Our block has eight plugs on top, and eight jacks on the bottom. The plugs and jacks have only two conductors each, one for power distribution and one for bidirectional signals.

- Asynchronous, distributed communication: Simple connectors lead to a more complicated communication architecture. In particular, our blocks do not enjoy the advantage of a common bus, which greatly simplified some of the previous systems. All communication in our block structures is based on asynchronous message passing between physically connected blocks.

TECHNICAL DETAILS
A block consists of a 100mm (L) × 50mm (W) × 25mm (H) plastic box that is drilled to accommodate slightly modified DC power connectors. The dimensions of the box and the locations of the plugs and jacks are such that our building blocks can be configured like Lego™ bricks. The connectors are mounted on a PC board that also accommodates a microcontroller (a PIC16C77), various passive components, and optional transducers and sensors (Fig. 1f).

Each connector has just two conductors. However, instead of using one for power and one for ground (normal usage for such a connector), we use the inner pin as a signal line for bidirectional communication, and the outer sleeve for power distribution. Each block is wired internally so that connector sleeves carrying power and ground, respectively, are ar-
Figure 1: A physical block structure comprising 50 blocks (a), a close-up of the structure (b), renderings of the virtual model computed from it, two literal (c,d), and one more decorative (e). The literal renderings use preassigned shapes and colors to render the blocks. The virtual model is parsed and augmented automatically for the decorative rendering. A bottom view of the printed circuit board inside each block (f) shows eight connector jacks, four LEDs, and the microcontroller. The polarity of the ‘X’ connector sleeves is different from that of the ‘O’ sleeves.

ranged in an alternating pattern (Fig. 1f). Thus each block will have at least one connection to power and one to ground in any typical Lego™-like structure. A full-wave bridge rectifier copes with this ambiguity.

A fully assembled block structure computes its own geometry in three phases. In Phase 1, each block determines the pins with which it connects to some other block. In Phase 2, connected blocks exchange block and pin ID data over their connected pins. During these two phases, all blocks compute and communicate in parallel, using power-on for initial (but approximate) synchronization. In Phase 3, the connectivity data collected by each block in Phase 2 are communicated to a host computer via the drain, a special block that can be attached to any part of a block structure, and that supplies power to the structure and a serial connection to the host computer. Phase 3’s information-draining operation is performed by a distributed, depth-first traversal of the block structure. This operation is serial, but the relay of data packets from block to block back to the drain is pipelined, thus achieving some parallelism in Phase 3, too.

At the end of Phase 3, the host computer has complete connectivity information for the block structure. The host also has shape and appearance data for each block, indexed by block ID, and recorded when a block is programmed. A simple recursive procedure now gives the 3D geometry of the block structure, which is used to produce a scene description suitable for a variety of 3D graphics applications (Fig. 1c,d).

Our blocks might appear too coarse in scale and shape to build richly decorated virtual structures. However, even a limited ability to parse a structure allows the host computer to perform detailing tasks that would be tedious for the user. Our system generates a description of a block structure as a set of logical axioms, one to assert the existence and location of each block. These axioms serve as input to a Prolog program that identifies structural elements of a block structure, interpreted as a building of some kind. Recognized structural elements are then colored and textured distinctively, or decorated with additional geometry, to automatically enhance the visual appearance of the rendered model (Fig. 1e).

REFERENCES