Published in the Computer Graphics Proceedings (SIGGRAPH'98), July 19-24, 1998, © 1998 ACM

mediaBlocks: Physical Containers, Transports, and Controls for Online Media

Brygg Ullmer, Hiroshi Ishii, and Dylan Glas*
Tangible Media Group
MIT Media Lab

Abstract

We present a tangible user interface based upon *mediaBlocks:* small, electronically tagged wooden blocks that serve as physical icons ("phicons") for the containment, transport, and manipulation of online media. MediaBlocks interface with media input and output devices such as video cameras and projectors, allowing digital media to be rapidly "copied" from a media source and "pasted" into a media display. MediaBlocks are also compatible with traditional GUIs, providing seamless gateways between tangible and graphical interfaces. Finally, mediaBlocks act as physical "controls" in tangible interfaces for tasks such as sequencing collections of media elements.

CR Categories and Subject Descriptors: H.5.2 [User Interfaces] Input devices and strategies; H.5.1 [Multimedia Information Systems] Artificial, augmented, and virtual realities

Additional Keywords: tangible user interface, tangible bits, phicons, physical constraints, ubiquitous computing

1 INTRODUCTION

Computers have traditionally recorded digital information on both "fixed" and "removable" storage media. While removable media have often been limited in capacity, speed, and accessibility, these factors have been offset by the expanded storage and mobility removable media affords.

However, the rise of widespread online connectivity for both computers and other digital media devices (cameras, projectors, etc.) alters this historical role division. Extended capacity and instant mobility are now better afforded by keeping media online. Does removable media risk obsolescence in the online age?

From a user interface standpoint, the process of online media exchange between digital whiteboards, projectors, computers, and other devices is still far from seamless. Reference and manipulation of online media is at present generally limited to GUI-based interaction with file paths, URLs, and hyperlinks – a process quite at odds with most media interfaces.

We believe that coupling the physicality of removable media with the connectivity and unlimited capacity of online content offers a potential solution to this problem. Moreover, we believe this approach suggests new physical-world interface possibilities that go beyond the pervasive graphical user interface.

*{ullmer,ishii,krill}@media.mit.edu
20 Ames St., E15-445, Cambridge, MA 02139
http://tangible.media.mit.edu/

In this paper, we introduce a tangible user interface (TUI) based upon *mediaBlocks*:* small wooden blocks that serve as physical icons ("phicons") [15] for the containment, transport, and manipulation of online media. MediaBlocks do not actually store media internally. Instead, they are embedded with ID tags that allow them to function as "containers" for online content, or alternately expressed, as a kind of physically embodied URL.

MediaBlocks interface with media input and output devices such as video cameras and projectors, allowing digital media to be rapidly "copied" from a media source and "pasted" into a media display. MediaBlocks are also compatible with traditional GUIs, providing seamless gateways between tangible and graphical interfaces. Finally, mediaBlocks are used as physical "controls" in tangible interfaces for tasks such as sequencing collections of media elements.

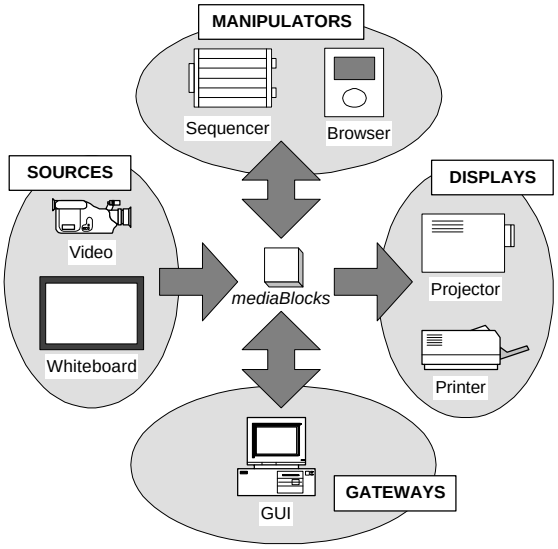

Figure 1: mediaBlocks design space

The mediaBlocks design space is illustrated in Figure 1. MediaBlocks serve as a medium of interchange between media source and display devices; between media devices and GUI-based computers; and between these pre-existing devices and new tangible interfaces for media manipulation. In this fashion, mediaBlocks fill the user interface gap between physical devices, digital media, and online content.

*Note: Our mediaBlocks have no relation to the Magnifi Inc. software product of the same name.

2 FUNCTIONALITY OVERVIEW

The following paragraphs describe the basic functionality of our mediaBlocks interfaces. Detailed consideration of interface use and implementation follows later in the paper.

2.1 Physical Containers

MediaBlocks are phicons (physical icons, [15]) embodied as small wooden blocks. MediaBlocks do not actually store digital media internally. Instead, they physically contain ID tags that are dynamically associated with sequences of online media elements. When used in environments where many media devices are linked to online computation, mediaBlocks act as physical "containers" for online media.

As such, mediaBlocks have a variety of interesting properties. Because contents remains online, mediaBlocks have unlimited "capacity" and rapid transfer speed (copying is instantaneous, while playback is a function of network bandwidth). For the same reason, a lost block is easily replaced. MediaBlocks may also contain live streaming media.

2.2 Physical Transports

One role of mediaBlocks is support for simple physical transport and interchange of media between media devices. While inter-application "copy and paste" is core to the modern GUI, comparably lightweight equivalents have not existed for physical media devices. We have realized a physical analog of "copy and paste" by combining mediaBlocks with physical slots mounted upon associated media devices.

We have implemented mediaBlock support for four media devices: a desktop video projector, network printer, video camera, and digital whiteboard. Inserting a block into the slot of a media source begins recording to an online server. Recording stops when the block is removed. This can be understood as "copying" from the media source into the block. Similarly, contents may be "pasted" into a media display by inserting a block into the associated slot. This will display block contents, with removal halting playback.

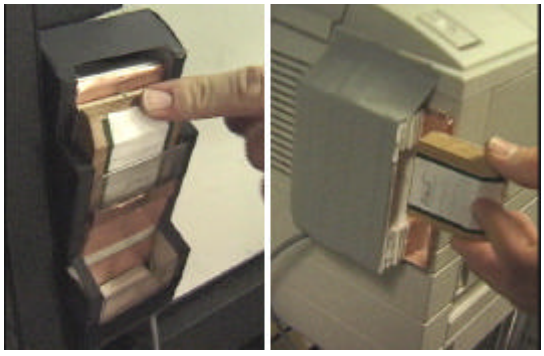

Figure 2: Whiteboard, printer mediaBlock slots

2.3 Physical Gateways

MediaBlock slots have also been implemented for use on general-purpose computers. Slots are mounted on the right margins of computer monitors. When a mediaBlock is inserted into the slot, a GUI window scrolls "out of the block" from the slot's left edge (contiguous with the screen). This window provides GUI access to block contents. (Figure 3)

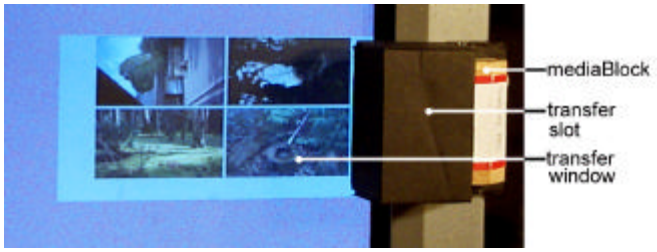

Figure 3: mediaBlock monitor slot

MediaBlock contents may then be transferred to the desktop or to GUI applications with conventional mouse-based "drag and drop" support. Media may also be copied into blocks in this fashion. In this way, mediaBlocks can be used to seamlessly exchange content between computers and media sources, displays, or other computers.

2.4 Physical Browsers

While the transport function of mediaBlocks allows media to be exchanged between various output devices, it does not address interactive control of media playback, especially for mediaBlocks containing multiple media elements. The *media browser* is a simple tangible interface for navigating sequences of media elements stored in mediaBlocks. (Figure 4)

The browser is composed of a detented browse wheel, video monitor, and mediaBlock slot. Useful both in casual viewing and formal presentation contexts, the browser supports the interactive navigation of mediaBlocks sequences for projector-based display, as well as displaying media on its local screen.

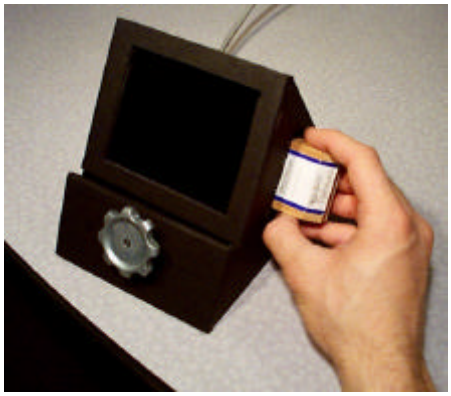

Figure 4: Media browser device

2.5 Physical Sequencers

The media browser provides interactive physical control of mediaBlock display, but does not support modification of mediaBlock contents. The *media sequencer* is a tangible interface using mediaBlocks both as containers and *controls* for physically sequencing media elements. (Figure 5)

Where earlier sections have introduced mediaBlock slots, the sequencer uses physical *racks*, *stacks*, *chutes*, and *pads* as structures that physically and digitally operate upon mediaBlocks. In particular, the rack is a *physical constraint* used to digitally index

and sequence mediaBlock contents as a function of the blocks' physical configuration on the rack.

Figure 5: Media sequencer device

2.6 Integrated Functionality

In addition to illustrating the above mediaBlock functions, the accompanying video figure demonstrates the creation of a multimedia presentation integrating all behaviors we have described.

In less than four minutes of uncut footage, we show recording and transport of digital video and whiteboard content; selecting photographs and authoring textual slides for inclusion; assembling these contents into an integrated presentation; rendering the presentation to a network printer; and presenting this content on the browser-controlled video projector. The resulting content is also Web-accessible at each stage of authoring and delivery.

We believe this interface example is significant for several reasons. First, this example shows the creation, manipulation, and use of complex multimedia content with a simplicity and speed that we believe to be highly competitive with other approaches. Simultaneously, it is key to note that the only keyboard, mouse, or other GUI interaction present in the entire sequence is the composition of a textual slide.

In the remainder of the paper, we will continue to develop the interface process, technical operation, and functional roles that we believe represent a powerful new approach for interaction between people, physical objects, and online digital information.

3 RELATED WORK

The mediaBlocks project was most directly influenced by the metaDESK/Tangible Geospace prototype [15, 8] that introduced the phicon concept. Tangible Geospace was based upon an augmented physical desktop and phicons representing geographical landmarks. Manipulation of phicons controlled the position, rotation, and scaling of spatially coincident graphical landscapes.

Tangible Geospace was developed in part to explore physical instantiation of the GUI metaphor. As the GUI system of windows, controls, and icons itself drew from a metaphor of the physical desktop, the physical analogs of lenses, instruments, and phicons (physical icons) seemed a promising first step towards the realization of tangible interfaces.

In practice, Tangible Geospace served to illustrate a number of major differences between graphical and tangible UIs. The fundamental malleability and extent of control over the GUI's graphical workspace differs substantially from the object persistence and cumulative, potentially inconsistent physical degrees of freedom expressed by TUI elements. In short, the GUI metaphor appeared unable to generalize across the potential design space of tangible user interfaces.

Subsequent research with Triangles and inTouch made a key insight towards resolving this shortcoming [7, 2]. Instead of relying upon pre-existing metaphors of GUI or the physical world (e.g., the metaDESK's optical metaphor [8]), these projects developed new interface metaphors based on the affordances unique to physical/digital artifacts. Both projects served as partial inspiration for mediaBlocks, which continues this design aesthetic.

The whiteboard-based mediaBlock functionality draws upon an earlier whiteboard TUI called the transBOARD [8]. The transBOARD used paper cards called *hypercards* as physical carriers for live and recorded whiteboard sessions. However, the hypercard interaction relied upon barcode wanding, which was found cumbersome in practice.

The ubiquitous computing vision of [16] speaks to moving computation and networking off the desktop and into many devices occupying niche contexts within the physical environment. This insight is core to the mediaBlocks system. Still, the interface prototypes of [16] continued to rely primarily upon GUI-based approaches.

The Bricks work [5] made dynamic association between digital properties and physical handles through the tray device. Also an inspiration for the mediaBlocks work, the Bricks work did not develop physical objects as digital containers or physical manipulation outside of the "Active Desk" context.

Bishop's Marble Answering Machine [3] made compelling demonstration of passive marbles as "containers" for voice messages. Later work by Bishop demonstrated physical objects associated with diverse digital content, and prototyped an early object-GUI gateway.

The Pick-and-Drop work of [11] provides strong support for file exchange between palmtop, desktop, and wall-based GUIs with a pen stylus. However, the technique less directly addresses media exchange between non-GUI devices, or with devices that are not spatially adjacent.

Molenbach's LegoWall prototype (discussed in [6]) used LEGO structures to contain information about ocean-going ships. These objects were combined with display and control objects that, when plugged adjacent to containers, could display shipping schedules, send this data to hardcopy printers, etc.

The AlgoBlock system uses the manipulation of physical blocks to create computer programs [13]. Connections between the blocks create LOGO programs that may be rearranged by the user, esp. in an educational context.

The Digital Manipulatives research of Resnick, Borovoy, Kramer, et al. [12] has developed "societies of objects" including badges, buckets, beads, balls, and stacks. Each is associated with digital semantics responsive to physical manipulations such as shaking, tossing, and stacking objects, or dunking objects in buckets of "digital paint." The manipulatives work makes strong progress towards developing objects with rich digital/physical couplings.

4 SYSTEM OVERVIEW

Figure 6 illustrates our current implementation of the mediaBlocks interface. The center column, media devices, lists the physical devices for which we have integrated mediaBlock support. The left column shows devices supporting operation of mediaBlock slots. The right column presents the computers that manage media recording and playback. The media browser and sequencer SGI machines play additional roles as the controllers for these tangible interfaces. However, these details are beyond the scope of our illustration.

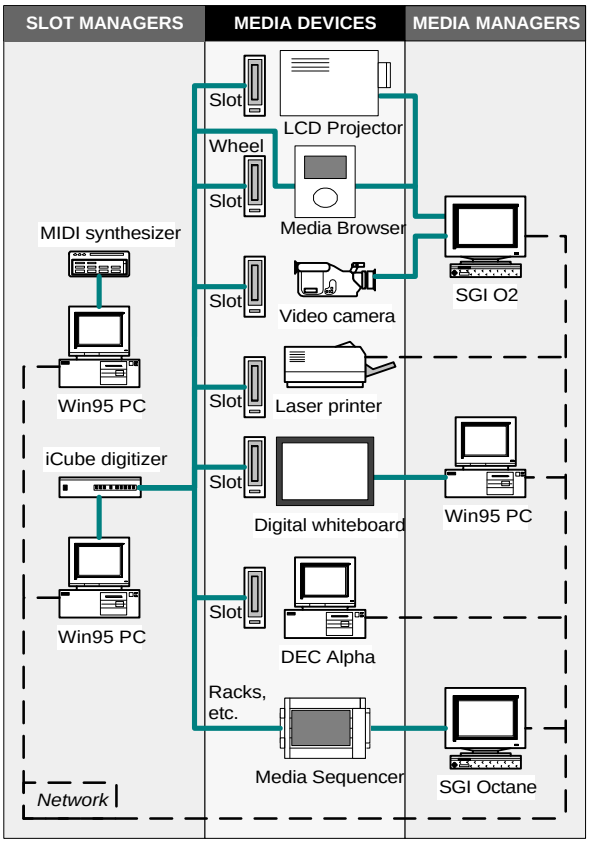

Figure 6: First-generation mediaBlocks system diagram

The mediaBlocks system was implemented with a Tcl- and [incr Tcl]-based tangible interface software/hardware toolkit called 3wish [14]. 3wish includes modules supporting MIDI-based digitizers and synthesizers, Inventor-based 3D graphics, computer vision and magnetic field trackers, etc. 3wish also supports a distributed architecture called *proxy-distributed* or *proxdist computation* [15], which provides strong abstractions for mixed physical/digital systems such as the mediaBlocks interface.

Development of 3wish support for the media sequencer and mediaBlock slots has consumed a large part of the mediaBlocks effort. The mediaBlock idea of objects as physical proxies for online content and computation grew out of early proxdist research, and the influence of 3wish's distributed architecture is visible in the half-dozen computers and tangible interfaces composing Figure 6. However, detailed discussion is beyond the scope of this paper, and is left to [14,15] and future treatments.

5 PHYSICAL CONTAINERS

The paper has emphasized the role of mediaBlocks as "containers" for media, as well as the use of mediaBlock slots for media interchange between various devices. However, a range of removable media devices have realized this basic function. For instance, videotapes and floppy disks are both "physical containers" for electronic media that support media interchange through systems of "physical slots." How do mediaBlocks relate to these well-known technologies?

The comparison will be explored for floppy disks (and other removable media analogs), which share the ability to store digital media of various formats. First, it is clear that mediaBlocks and floppy disks are technically quite different. Floppy disks function by taking information offline, recording media onto the disk's own storage. MediaBlocks instead work by moving information online, referenced by the internal ID of the mediaBlock object.

It is also interesting to note that mediaBlocks transparently support media with widely varying bandwidths. For instance, mediaBlocks are equally practical for digital whiteboard and digital video recordings, even though the characteristic bit rates differ by five orders of magnitude (~100KB vs. ~10GB per hour).

From a user interface standpoint, mediaBlocks and floppy disks also have a number of differences. Floppy disk contents are accessed indirectly through graphical or textual interaction on a host computer. In contrast, mediaBlock contents may be accessed through physical manipulation of the mediaBlock object itself. For example, inserting a target mediaBlock into a digital whiteboard's slot initiates recording "into" the block. Similarly, moving a host mediaBlock on our media sequencer's position rack allows sequences of images to be navigated (see section 6.1).

Building from this distinction, mediaBlocks possess a simplicity and "lightweight" mode of operation rarely found with the floppy disk medium. Media recording, playback, and exchange between our example whiteboard, camera, printer, projector, and computer are all as simple as inserting and withdrawing a mediaBlock from the respective slots.

The mediaBlock support for physical media exchange does not force a "sneaker-net" ethic upon users. Instead, mediaBlocks offer the simplicity and directness of physically referencing *which data* and *which device* when physical proximity is a convenient delimiter. Common "reference in absence" tasks such as dispatching jobs to remote printers for later pick-up or delivery may be supported by shortcut controls (e.g., a "print" button on the whiteboard), or by inserting mediaBlocks into TUI or GUI devices providing remote printer access.

Thus, mediaBlocks act not as a *medium of storage*, but rather a *mechanism of physical reference and exchange*. In this sense, the use of mediaBlocks more closely resembles the interactive process of "copy and paste" propagated out into the physical world than the storage medium of floppy disks. Conceptually consistent with the premise of tangible user interface, this is a major distinction that colors the spectrum of mediaBlocks applications.

Also in point of fact, neither floppy disks nor other removable media are native to our projector, printer, or whiteboard, nor do we know of these features in comparable products. The absence of media drives from these well established and commercially com-

petitive products provides market evidence that mediaBlocks serve new or different conceptual roles.

5.1 Implementation

MediaBlocks are constructed as 5x5x2cm wooden blocks embedded with electronic ID tags. First-generation blocks are tagged with resistor IDs. New efforts use Dallas Semiconductor iButton™ devices, which incorporate both digital serial numbers and nonvolatile memory. It is worth noting that the physical form of mediaBlocks is a product of their intended use, not of technical limitations. Both tag technologies are available as surface-mount devices a few millimeters in diameter, rendering block size technically near arbitrary.

Both resistor- and iButton-based mediaBlocks couple to slots, racks, and other TUI elements with a pair of electrical contacts. For resistor-based blocks, ID is determined with a voltage-divider circuit against a reference resistor, sampled by an Infusion Icube MIDI A/D converter. For the iButton implementation, an interface using Microchip's PIC 16F84 microcontroller was implemented, based upon the iRX 2.0 board [10].

MediaBlocks operate in conjunction with a number of different interface devices. The most common of these is the slot. Slots were prototyped in foamcore, with copper tape contacts. Audio feedback for slot entrance, exit, and status events is currently supported by an external MIDI synthesizer.

MediaBlock phicons are separated from their media "contents" by several levels of indirection. These mappings include:

a) physical block → network address
b) network address → mediaBlock data structure
c) data structure element → individual media contents

The individual steps of this mapping process are illustrated in Figure 7. First, insertion of an ID-tagged block **(1)** is detected electronically by the slot **(2)**. The ID value is registered by a computer hosting the slot's tag reader **(3)**, and transmitted as a block entrance event to the display device's media manager **(4)**. The slot and media managers could be hosted on the same machine. In our implementation, the libraries supporting tag readers and media displays were specific to PC and SGI platforms, respectively, requiring separate computers for **(3)** and **(4).**

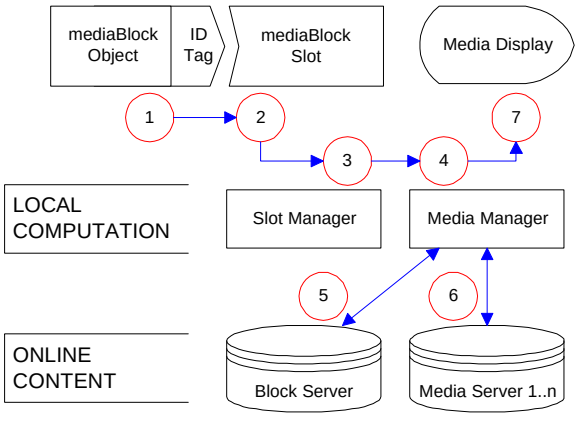

Figure 7: mediaBlock display flow diagram

Once an ID value has been obtained for a mediaBlock, the block server **(5)** provides a data structure describing the block's digital contents, presented in Table 1. With the resistor ID scheme, a central block server is responsible for mapping block IDs to block data structures for all compatible block devices. However, the resistor-based approach does not scale for distributed operation.

The iButton-based mediaBlocks solve this problem by storing the URL of their hosting block server within internal nonvolatile RAM (4096 bits for our chosen model), allowing truly distributed operation. iButtons also support storage of encrypted data, potentially useful for authenticating mediaBlocks and granting read or write permissions by a given media device.

After retrieving the block data structure, the device media manager retrieves the specified media contents from one or more media servers **(6)**. This content is finally sent to the media display under control of display-specific MIME registries **(7)**, similar to Web browser plug-in registries.

mediaList: *List of contained media element addresses*
physidType: *Type of physical ID tag on block*
physidInst: *ID of block tag (usually a number)*
mediaHost: *Media stored on media- or block-server?*
recordBehavior: *New media appends or overwrites old?*
lastObservedLocale: *Last location block observed*
lastObservedTime: *Timestamp of last block sighting*
blockLabel: *Text describing block contents*
blockLabelColor: *Color of paper block label*

Table 1: mediaBlock data structure

Earlier sections have discussed the use of mediaBlocks as a medium of exchange between graphical and tangible interfaces. This works particularly well in conjunction with Microsoft's Internet Explorer 4.0 "Internet shortcuts" feature (and equivalencies provided by other operating systems). "Internet shortcuts" allow distributed online media (e.g., URL-referenced HTTP sources) to be manipulated by the Windows95 desktop and applications indistinguishably from files on local disk drives.

We have explored the synthesis of Internet Shortcuts from media elements dragged out of monitor-slot mediaBlocks with GUI drag and drop. This combination represents a significant step towards truly seamless integration between the online media spaces of graphical and tangible interfaces.

6 PHYSICAL CONTROLS

The previous section discussed the physical containment, transport, and interchange aspects of mediaBlocks. The section also noted that in this capacity, the use of mediaBlocks more closely resembles the software process of "copy and paste" than the storage function of floppy disks.

However, the earlier section did not discuss how users might actively manipulate mediaBlock contents. For example, while both video decks and MS Windows95 CD-ROM drives support "auto-launch" capabilities analogous to mediaBlock slot playback, these systems also provide additional controls for more sophisticated interactions.

Following these examples, we might model interfaces on the physical play/stop buttons and jog/shuttle wheels of VCRs, as we

have done with our media browser. We might also use traditional graphical interfaces to manipulate mediaBlock contents, as we have done with the GUI monitor slot.

At the same time, it is interesting to explore new interface possibilities specific to tangible user interfaces. The slot-based "copy and paste" functionality illustrates the first step of such an approach, binding digital semantics to the insertion of mediaBlocks into physical slots.

We have carried this approach forward in our work with the *media sequencer* interface. Here, we introduce *racks*, *stacks*, *chutes*, and *pads* as physical constraints that enable mediaBlocks to act as physical *controls* for directly manipulating block contents (Fig. 8).

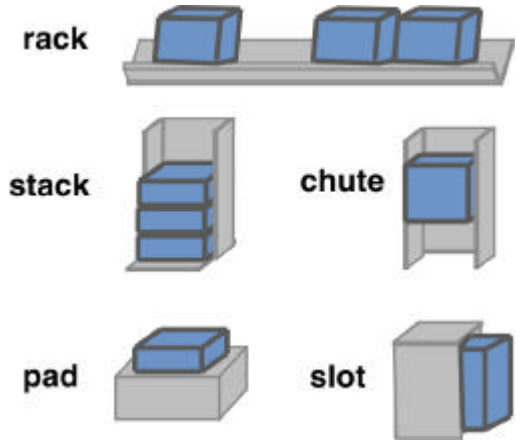

Figure 8: Media sequencer physical constraints, *used in combination with mediaBlocks as physical controls*

The mediaBlock *rack* is the primary physical constraint elements used in the media sequencer. The mediaBlocks rack was inspired by the Scrabble™ board game's combination of letter tiles and the tile rack. In Scrabble, the rack allows letter tiles to be rapidly inserted, sequenced, and grouped into meaningful patterns. In the sequencer context, these physical attributes of position, sequence, and adjacency may be digitally recast as indexing, sequencing, and Boolean AND/OR operations, respectively.

6.1 User Interface

The media sequencer prototype is illustrated in Figures 5, 9, and 10. Sequencer operation is dominated by two physical constraint structures: the *position rack* and *sequence rack*.

When a mediaBlock is placed in the position rack, its contents are shown on the sequencer display as a perspective wall [9]. The focus position of the perspective wall is interactively controlled by the relative position of the mediaBlock on the position rack (Figure 10). Moving the block to the rack's left edge moves the perspective wall to focus on the block's first element. Similarly, the right edge of the position rack corresponds to the block's last element.

The combined position rack and perspective wall serve several purposes. First, they support the interactive viewing of mediaBlock contents with imaging of both detail and context [9]. Secondly, they allow the position rack to select an individual media element for copying between mediaBlocks.

Towards this end, a destination mediaBlock can be placed in the sequencer's *target pad*, physically adjacent the position rack (see Figure 9). Pressing the block on the target pad will append the currently selected media element into the destination block. This is confirmed with audio feedback and a haptic "click," as well as an animation of the selected media transferring "into" the target block. Pressing and holding the block, followed by moving the source mediaBlock to a new location, will copy a range of elements into the target block (analogous to dragging out a selection range with a mouse).

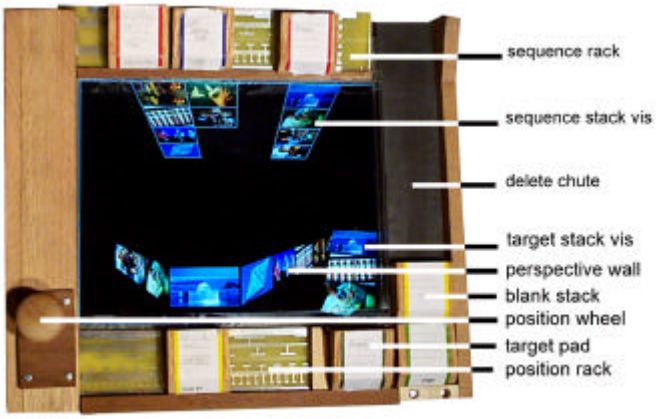

Figure 9: Media sequencer components

We believe this concept of using mediaBlock phicons to physically manipulate their internal contents generalizes to a powerful new interaction technique. Here, we are using mediaBlocks as *physical controls* for directly acting upon their internal state. The nearest equivalent in GUIs might be dragging a graphical folder icon across a scrollbar to index through folder contents. While a somewhat bizarre behavior for the GUI, we believe that use of mediaBlocks as physical controls holds substantial promise in the TUI context.

The *sequence rack* extends the control functionality of mediaBlocks. This rack allows the user to combine the contents of multiple mediaBlocks into a single sequence, which can then be associated with a new mediaBlock carrier on the target pad. When a mediaBlock is placed onto the sequence rack, its contents scroll out of the block into the sequencer display space (Figure 9). Multiple mediaBlocks may be arranged to construct a new sequence.

Both the sequencer screen and target pad are shared between the position and sequence racks. When a mediaBlock is located on the position rack, the target pad is bound to selections from the perspective wall display. When the position rack is clear, the target pad is associated with recording of the sequence rack's aggregate elements.

The use of mediaBlocks as physical controls services only part of the sequence rack's behavior. Especially when a source mediaBlock contains many elements, navigating the perspective wall by incremental steps may be more convenient than using the position rack. The *position wheel* supports such incremental navigation. Haptic detents provide the user with physical feedback, where each detent corresponds with movement of one media element.

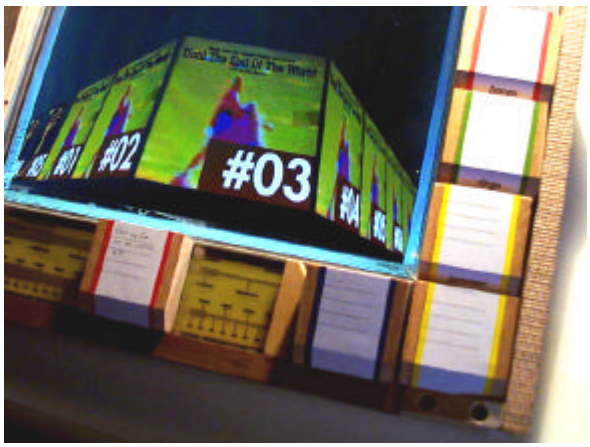

Figure 10: Media sequencer perspective wall (alternate view)
associated with movement of mediaBlock on position rack; a compact disk's music is "contained" within this block

6.2 Implementation

The media sequencer platform was assembled of wood and acrylic, and embedded with a 1280x1024-pixel 32cm-diagonal flat panel display. Position and sequence rack sensing was performed with a custom contact-grid board. The position wheel was tracked with a shaft encoder. Sensor inputs were digitized with the Infusion Systems Icube device, and acquired through the 3wish extensions to the Tcl language [14] using C++ wrappers over the Rogus MIDI interface suite [4]. The position rack perspective wall and other graphical visualizations were written with 3wish's Open Inventor-based routines, and executed on an SGI Octane workstation.

7 DISCUSSION

While mediaBlocks make progress towards broader vocabularies for tangible interface design, the project also illustrates some of the challenges in designing TUIs. As a case in point, we consider some of the design decisions leading to our sequencer prototype's current form. Some readers have questioned the sequencer's integration of a central graphic display, given our research focus upon tangible interface.

While the sequencer indeed integrates a flat-panel display, we argue that the design is the product of a particular interface task and set of design constraints. Our motivating task was the sequencing of presentation media, especially images and video. Our interface inspirations were the Scrabble tile/rack constraint system, and the brick/tray function of [5]. As our task centered upon manipulation of visual media, a graphic display of some sort was essential.

Our original hope was to integrate this display into a rack's footprint, displaying the graphical contents of transparent mediaBlocks in a fashion following the metaDESK's passive lens [15]. However, given that mediaBlocks usually contain multiple media elements, we had difficulty determining an effective method of display within a block's 5x5cm footprint. Thus, we decided upon a visual display external to the mediaBlock/rack system, while maintaining visual contiguity with the associated mediaBlock container.

Here, we explored use of back-projected, front-projected, and integrated displays. While each had advantages, we selected a 32cm-diagonal 1280x1024-pixel integrated flat panel display on the basis of resolution, dot pitch, and compactness of integration. We wished adequate display resolution to support the visually-intensive task. We were also interested in a relatively small, compact device, even at the cost of greater display real estate such as provided by the metaDESK [15].

This was because we wished to make extensive use of physical constraints to support the sequencing task. Simultaneously, we imagined our users making simultaneous use of multiple complementary devices, such as the combination of a general-purpose computer for authoring new content, and the sequencer for assembling and manipulating the presentation. These usage constraints, coupled with the drive for proximity between physical controls and visual displays, drove the system to its current form.

Aspects of the sequencer design remain challenging, including potential inconsistency between the position wheel and rack; linkage between rack-based mediaBlocks and screen-based displays; and the shared screen use by position and sequence racks. Additionally, it is possible that the position rack + mediaBlock selection mechanism is less efficient than (say) directly selecting contents with one's finger. A second-generation sequencer design is under development to add this direct content-selection ability, increase display real estate, and rationalize the behavior of sequencer racks, while maintaining mediaBlock controls for sorting, sequencing, and transporting media into and out of the sequencer.

We believe mediaBlocks' underlying containment and transport functions are fundamentally sound. MediaBlocks' use as physical controls makes a significant extension to this transport function, and has been demonstrated useful in tasks like the video figure's presentation example. Thus, while less well-established than the transport function, our implementation leaves us optimistic about the interface potential of phicon controls.

8 FUTURE WORK

While part of our research motivation lies in seeking new paradigms for interface outside the well-explored GUI context, it is interesting to explore parallels between graphical and physical interaction techniques. One such instance is the historical emergence of consistent interface behavior across multiple GUI applications, notably articulated in the Apple Human Interface Guidelines [1].

Our design of mediaBlock slots and sequencer constraints has been shaped by an interest in TUI analogs for GUI inter-application behaviors. Our use of mediaBlocks for media transport and interchange develops a physical analog of the GUI "copy-and-paste" functionality. Similarly, the sequencer's racks, chute, and other constraints have parallels to GUI "interaction primitives" (e.g., desktop-based clicking, dragging, etc. of icons), without directly embodying GUI widgetry as with the metaDESK [15].

As much of the TUI appeal lies in a diversity of physical embodiments to address specific interface tasks, it is unclear how far analogies to [1] might extend. Nonetheless, prospects for consistent TUI interface vocabularies and widespread TUI/GUI interoperability are highly attractive.

Finally, mediaBlock's online aspect suggests their ability to "contain" live, streaming content: in other words, the ability to operate as media "conduits." We have demonstrated mediaBlocks containing both streaming media sources such as RealAudio™ and RealVideo™ media, as well as pairs of mediaBlock conduits which together broadcast and receive streaming video.

At the same time, the conduit functionality of mediaBlocks represents a significant conceptual expansion with its own user interface questions. For instance, if a user wishes to both record and broadcast a whiteboard session, or both display and store a live video stream, how are these aggregate behaviors best accommodated? These and other open questions remain. As a result, we leave conduits for further discussion in future work.

9 CONCLUSION

We have presented a system of tangible user interface based upon *mediaBlocks*: small, electronically tagged wooden blocks that serve as physical containers, transports, and controls for online media. MediaBlocks do not store media internally, instead serving as phicons (physical icons) which give physical embodiment to online media.

MediaBlocks provide a mechanism for seamlessly exchanging digital contents between diverse media devices, including media sources, displays, general-purpose computers, and specialized tangible interfaces. Towards these ends, we have demonstrated the ability of mediaBlocks to physically "copy" media from a whiteboard and camera source, and "paste" this content into a printer and projector.

We have also shown the use of mediaBlock slots as physical gateways between tangible and graphical interfaces, allowing media to be swiftly exchanged with traditional GUIs.

Additionally, mediaBlocks are used as physical controls for operating upon their own digital "contents." We have demonstrated this ability in the media sequencer by swiftly composing a presentation integrating video, photographs, whiteboard recordings, and text. While the sequencer supports this task with a graphical display, the entire task is accomplished without a keyboard, pointer, or cursor, except for the GUI-based entry of the textual slide.

Finally, we have discussed analogs between TUI media exchange between diverse physical devices, and GUI support for consistent multi-application operation and communication. Similarly, we have demonstrated new roles and visions for seamless interaction with online content beyond the traditional GUI context.

In conclusion, we believe mediaBlocks are a powerful tangible interface for the seamless exchange and manipulation of online content between diverse media devices and people. More generally, we believe mediaBlocks represent a step towards broader tangible interface use as an interaction technique uniting people, computational media, and the physical environment.

10 ACKNOWLEDGEMENTS

Many people have contributed to this work. John Alex and Paul Grayson helped with software and hardware implementation. John Underkoffler and Paul Yarin assisted video production. Andrew Dahley and Andrew Hsu provided design assistance. Matt Gorbet, Scott Brave, and Victor Su provided additional helpful input. Ben Denckla helped with early Rogus efforts. Paul Grayson, Hannes Vilhjalmsson, Joe Marks, Tara Rosenberger, and others provided feedback on paper drafts. This research was sponsored in part by the MIT Media Lab's Things That Think consortium and a Mitsubishi fellowship.

11 REFERENCES

[1] Apple Computer, Inc. *Apple Human Interface Guidelines: The Apple Desktop Interface*. New York: Addison Wesley, 1987.

[2] Brave, S., and Dahley, A. inTouch: A Medium for Haptic Interpersonal Communication. In *CHI'97 Extended Abstracts*, pp. 363-364.

[3] Crampton Smith, G. The Hand That Rocks the Cradle. *I.D.*, May/June 1995, pp. 60-65.

[4] Denckla, B. Rogus C++ MIDI Suite. `http://theremin.media.mit.edu/rogus/`

[5] Fitzmaurice, G., Ishii, H., and Buxton, W. (1995). Bricks: Laying the Foundations for Graspable User Interfaces. *Proc. of CHI'95*, pp. 442-449.

[6] Fitzmaurice, G. G*raspable User Interfaces.* Ph.D. Thesis, University of Toronto, 1996.

[7] Gorbet, M., Orth, M., and Ishii, H. Triangles: Tangible Interface for Manipulation and Exploration of Digital Information Topography. In *Proc. of CHI'98*.

[8] Ishii, H., and Ullmer, B. Tangible Bits: Towards Seamless Interfaces between People, Bits, and Atoms. In *Proc. of CHI'97*, pp. 234-241.

[9] Mackinlay, J., Robertson, G., & Card, S. The Perspective Wall: Detail and context smoothly integrated. In *Proc. of CHI'91*, pp. 173-179.

[10] Poor, R. The iRX 2.0 ...where Atoms meet Bits. `http://ttt.media.mit.edu/pia/Research/iRX2/`

[11] Rekimoto, J. Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proc. of UIST'97,* pp. 31-39.

[12] Resnick, M., Berg, F., et al. Digital Manipulatives: New Toys to Think With. In *Proc. of CHI'98*.

[13] Suzuki, H., Kato, H. AlgoBlock: a Tangible Programming Language, a Tool for Collaborative Learning. In *Proc. of 4th European Logo Conference*, Aug. 1993, Athens Greece, pp. 297-303.

[14] Ullmer, B. 3wish: Distributed [incr Tcl] Extensions for Physical-World Interfaces. In *Proc. of Tcl/Tk'97,* pp. 169-170.

[15] Ullmer, B., and Ishii, H. The metaDESK: Models and Prototypes for Tangible User Interfaces. In *Proc. of UIST'97*, pp. 223-232.

[16] Weiser, M. The Computer for the 21st Century. In *Scientific American*, 265(3), pp. 94-104.