Foxels: Build Your Own Smart Furniture

Florian Perteneder*
Kathrin Probst
Joanne Leong†
Media Interaction Lab, University of Applied Sciences Upper Austria

Sebastian Gassler
Christian Rendl
Patrick Parzer
Media Interaction Lab, University of Applied Sciences Upper Austria

Katharina Fluch
Sophie Gahleitner
Media Interaction Lab, University of Applied Sciences Upper Austria

Sean Follmer
Mechanical Engineering, Stanford University

Hideki Koike
School of Computing, Tokyo Institute of Technology

Michael Haller
Media Interaction Lab, University of Applied Sciences Upper Austria

Figure 1: Foxels are modular, smart furniture building blocks that enable users to create their own interactive furniture.

ABSTRACT

Introducing interactive components into furniture has proven difficult due to the different lifespans of furniture and digital devices. We present *Foxels*, a modular, smart furniture concept that allows users to create their own interactive furniture on demand by simply snapping together individual building blocks. The modular design makes the system flexible to accommodate a variety of interactive furniture setups, making it particularly well-suited for re-configurable spaces. Considering the trade-off between ease-of-use and high versatility, we explore a number of interaction methods that can be applied to modular interactive furniture, thereby extending the well-known tangible programming paradigm. After explaining our implementation, we demonstrate the validity of the proposed concepts by presenting how Foxels can be used in an ideation workshop along with many additional real-world examples.

*Also with School of Computing, Tokyo Institute of Technology.
†Also with MIT Media Lab, MIT.

TEI '20, February 9–12, 2020, Sydney, NSW, Australia

ACM ISBN 978-1-4503-6107-1/20/02…$15.00
https://doi.org/10.1145/3374920.3374935

CCS CONCEPTS

• **Human-centered computing → Interactive systems and tools**; *Collaborative interaction*; *Ubiquitous and mobile computing systems and tools.*

KEYWORDS

IoT, Interactive Environments, Constructive Assembly

ACM Reference Format:
Florian Perteneder, Kathrin Probst, Joanne Leong, Sebastian Gassler, Christian Rendl, Patrick Parzer, Katharina Fluch, Sophie Gahleitner, Sean Follmer, Hideki Koike, and Michael Haller. 2020. Foxels: Build Your Own Smart Furniture. In *Fourteenth International Conference on Tangible, Embedded, and Embodied Interaction (TEI '20), February 9–12, 2020, Sydney, NSW, Australia.* ACM, New York, NY, USA, 12 pages. https://doi.org/10.1145/3374920.3374935

1 INTRODUCTION

As the nature of work has been and keeps on undergoing major transitions, new forms of self-directed, creative, and collaborative work take place alongside established forms of directed, repetitive, and individual work [24, 39]. Tools and technology aim to support these shifts, along with interaction concepts that attempt to blend the power of computing with natural work practices and collaboration styles [21, 28, 29].

Although the kinds of work people do and the types of tools they use have changed dramatically in recent years, many workplaces have not. Standard office desks and meeting rooms with a large display placed in front of a conference table are still the norm rather than the exception. These environments hardly support the wide array of activities people perform when working collaboratively on creative tasks in

co-located spaces today. This lack of flexible environments that can be adapted for various tasks has been identified by various furniture manufactures.

This lead to a trend towards more flexible and modular furniture pieces targeting open-concept work spaces. One popular example is furniture built from modular (LEGO®-like) boxes [9, 14, 25, 58] that can be stacked together in just a few assembly steps to create the furniture pieces needed on demand. However, we have observed that these trends have not been reflected in recent research in interactive environments, besides some works covering specific use cases [62, 65]. Research in interactive environments still resemble concepts where static furniture pieces are augmented with input and output capabilities (e.g., touch sensing, integrated displays) [22, 30, 47, 56] developed two decades ago. These all-in-one solutions, however, can not fulfill today's requirements of modularity and flexibility. Another problem of these integrated solutions is the different life-time of furniture pieces and digital components, which are quickly outdated and often difficult to replace.

To address these shortcomings we present *Foxels* (= Furniture Voxels), a modular, smart furniture building block concept that allows users to create their own interactive furniture. To our knowledge, this is the first system facilitating user-customizability of furniture pieces in both the physical form and digital functionality. The basic modular building blocks, which are aware of their arrangement, can be easily snapped together to form different smart furniture solutions without the need for additional configuration. Arranging the building blocks augmented with electronics does not only create a physical shape but also defines the function following the tangible programming paradigm, which is used in many cube-based constructive assemblies [15, 19, 31, 52, 63]. While this way of defining functionality is simple to understand and easy to use it is also quite limited as there are no means to establish signal flows without direct physical attachment. Also it is hardly possible to change and configure the behavior of individual blocks and handle multiple types of data. Hence, we investigate ways to extend the tangible programming paradigm and aim to better understand the trade-off between ease-of-use and high versatility when supporting the interaction modalities required by modern work practices. In summary, our contributions are as follows:

- The design and implementation of a *hard- and software solution* to transform an existing modular furniture set into a furniture-size constructive assembly that allows to create a vast number of interactive furniture pieces by connecting cubes with input sensors and output elements.
- The design of *interaction methods* to extend the limits of tangible programming while keeping it easy-to-use to provide more flexibility and support a larger variety of applications than existing cubic constructive assemblies.
- The *support of multiple types of data*, such as key/control commands, images, and simple value-based signals. This enables more complex scenarios that can support co-located collaborative creative work.
- A demonstration of how the dynamic nature of the Foxels supports the different phases of an *ideation workshop* along with various *application scenarios*. This shows the concept's versatility regarding physical configuration and interactive functionality.

2 RELATED WORK

With Foxels we aim to create interactive, modular furniture for collaborative creative work. Hence, this work is at the intersection of the research streams on *Interactive Furniture*, *Cubic Constructive Assemblies*, and *Interaction and Programming* of networks that include input and output capabilities.

Interactive Furniture

Interactive furniture has historically been considered a key ingredient for the design of future work environments [22, 30, 47, 56]. Many works considered the augmentation of existing infrastructure in rooms with sensing and actuation capabilities: augmented walls serve as vertical interactive surfaces that extend the concepts of traditional white- and blackboards [10, 17, 21, 33]; augmented desks and tabletop displays bring digital content onto the physical desk [4, 18, 53, 59, 64]; augmented chairs are used as unobtrusive sensing devices [1, 42, 55] or as interactive input devices [6, 11, 46]; augmented floors serve as large-scale sensing platforms [44, 49, 51] or for foot-based input [2, 45, 61].

Beside these rather static examples only few works have introduced more fluid approaches of integrating interactive capabilities into furniture. *Z-Tiles* [49] are modular nodes that can be used to create a pressure-sensitive floorspace of varying size and shape. *InGrid* [65] is an interactive table that consists of individual tiles offering unique affordances for interaction with tangible and intangible objects. *TRANSFORM* [62] and *Proxemic Transitions* [16] present interactive, shape-changing desks that change their functional, aesthetic, and ergonomic qualities, e.g., by holding and moving physical objects, creating dividers on demand, or morphing from a horizontal into a vertical surface.

In contrast to these works, which are either static, only modular within a specific frame [49, 65] or flexible within the confines of a specific furniture form factor [16, 62], our goal was to create a system that is physically and digitally customizable. We envisioned a system being modular and flexible on a furniture scale, rather than consisting out of ready-made roomware components [56]. This means that a table does not have to stay a table but can be changed into a chair or bench etc. with the possibility to craft its interactive capabilities to complement its form factor.

Cubic Constructive Assemblies

As physical computerized objects, Foxels are essentially *Tangible User Interfaces (TUIs)* that allow users to leverage their physical manipulation abilities to interact with digital information [12, 26, 27, 60]. Within this domain, the cubic shape is particularly popular as it provides many advantageous affordances [36, 54], specifically for arrangement into *Constructive Assemblies*, a category of TUIs that involves the interconnection of modular physical, interactive units to formulate larger constructions [8, 32, 37, 38]. While not all of these *cubic constructive assemblies* support full 3D arrangements, we focus our overview on the ones that do.

Most cubic constructive assemblies feature dedicated input and output cubes [5, 19, 31, 63] that use mechanic [63] or magnetic [5, 19, 23, 31, 52] connectors for creating 3D arrangements. A noteworthy exception is *i-Cube* [15] that uses no such connectors and provides a set of similar cubes that only use different decals to provide visual distinction. Generally, there are many different ways of how signals between the individual cubes are exchanged, ranging from optical [31], inductive [15], mechanical [5], to electrical [19, 52] implementations. While some works use these signals directly for triggering actions [5, 19, 31], others track the 3D arrangements that are built from different cubes [15, 52, 63]. While most implementations require active components, *RFIDBricks* [23] presents a clever way of identifying arrangements with passive cubes by separating RFID antennas and chips. Still, to create a standalone system that provides input as well as output without an external computational device, we decided to use active components.

In contrast to all these works, Foxels do not only allow for creating interactive functionality, but also serve as modular furniture. Therefore, the most distinct feature is size. While most existing works are using cubes of 5–15 cm [5, 15, 19, 23, 31, 52, 63] side length, Foxels are based on a commercially available mobile furniture cube set [14] where a single cube has a side length of 36 cm. Creating a constructive assembly on a furniture-size level comes with a number of challenges. This involves the design of robust and reliable connection mechanisms and protocols, a schema to power large high-power components (e.g. large displays), and the development of unique components to solve unique issues pertaining to the furniture domain (e.g. ensuring smooth and stable surfaces for writing, seating, or storage).

Interaction and Programming

With the increasing popularity of *IoT (Internet of Things)* devices, various approaches of how to program networks that contain input sensors and output devices have emerged. Kubitza and Schmidt [35] provide an excellent overview of the different methods, of which *Tangible Programming* is most relevant to cubic constructive assemblies. Most of the cubic constructive assemblies implement some form of tangible programming, as chaining and arranging elements seems to be a natural and easy way of interacting with such sets. Nevertheless, there are two main limitations of what can be achieved with tangible programming as demonstrated by the fact that is mainly applied for playful interactions in childrens' toys or educational environments [3, 15, 19, 31, 32, 40]. First, the programming being based on physical stacking or chaining of individual elements poses limitations on what arrangements are possible, and poses a restriction in terms of need for physical connectedness. Second, the data that elements within a constructive assembly are exchanging and propagating is often limited to on/off states or simple signals [5, 19, 31], which poses limitations on the richness of interactions that are possible. To overcome physical limitations, some tangible concepts have proposed extending tangible programming via network communication [50, 52] or gestural interaction [41]. Other works considered ways to enable the exchange and display of dynamic graphical information across individual tiles [34, 41, 48].

Similar to many previous works, we use tangible programming due to its simple interaction without the need for explicit configuration. However, the collaborative work scenarios we aim to support demand more versatile interactions and the support of different data types. Being aware of these limitations, we design Foxels to support complex data (e.g. key/control commands, images) in addition to simple signals and address the tension between simplicity of use and maximal versatility by presenting a number of novel interaction approaches to extend the tangible programming paradigm without losing the ease-of-use of its core functionality.

3 DESIGN CONSIDERATIONS

While modular furniture concepts provide flexibility to support various working scenarios, they lack digital support. In contrast, many smart furniture environments use static designs that are inextricably linked with electronic components and thus offer little flexibility. Addressing these shortcomings, we designed a constructive assembly based on individual building blocks and embedded digital capabilities. It can be used to create various furniture configurations with suitable interactive functionality while providing the option to exchange outdated blocks. The concept should be usable without much effort and the need for specific infrastructure (e.g. server, application for configuration) while providing high flexibility to support many scenarios. This dilemma of handling the trade-off between constraint and expressiveness in constructive assemblies is described by Leong et al. [37].

To ensure that the constructive assembly meets the *structural requirements* to build stable, usable furniture, we based Foxels on *Bene Pixels* [14], a commercially available modular furniture concept. While most building blocks of this system

are cubic, it also includes flat shapes, which can be used as seats, shelves, and working surfaces to provide more versatility when creating furniture arrangements (see Figure 2). One important design consideration is that every element of the constructive assembly also provides furniture affordances (e.g. work surface, storage, or seating area). We believe that it is important that this furniture aspect is not lost when interactive capabilities are added. This is a major difference between Foxels and related cubic constructive assemblies.

Another goal is to enable *high versatility of functionality* while maintaining *easy and simple interaction*. Due to its great simplicity and utility without the need of external devices, we use tangible programming as a basic method for configuring functionality. This means that the physical arrangement (e.g. stacking, placing side-by-side) allows for ad-hoc creation of interactive functionality. The concept works particularly well when individual modules have one specific, pre-defined functionality per building block (e.g. pressure sensing, display, sound). Moreover, it enables the seamless integration of new functionalities by introducing new building blocks. However, this *one-function-per-block* design also comes with its drawbacks, especially when considering more complex scenarios. For instance, a sound block can be used to play a warning tone, a piece of music, or an entire playlist of songs. In many cases it is impractical and uneconomic to create individual blocks for all these functionalities. Therefore, we restrict the one-function-per-block design to the technical functionality (i.e. *Sound Foxel* plays all kinds of sounds) and investigate ways to expand the tangible programming paradigm and find complementary interaction paradigms to increase the expressiveness of the constructive assembly. These attempts include enabling communication between distant blocks and external devices using tangible utilities, switching behavior via physical interactions, changing function parameters via AR, and using external devices to control multiple Foxels with respect to their arrangement.

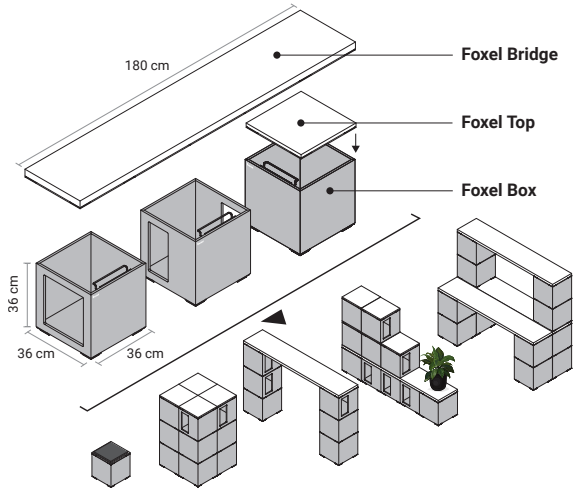

Figure 2: Modular building blocks and examples for their combination into different furniture configurations, shown with embellishments such as seat pillows and plants.

4 CONCEPT

We developed and implemented the Foxel concept considering the balance between ease-of-use and maximal versatility. Doing this, our main focus was on extending the tangible programming paradigm with complementary interaction concepts. Before discussing them in depth, we provide an overview of the Foxel types we built and the ways in which they handle the different types of data they can exchange.

Foxel Types

To demonstrate the concept, we implemented a total of 24 functional Foxels, of which 15 are unique. The selection was informed by discussions with potential users. Figure 3 provides an overview of all elements and categorizes them into *Input* (waiting for human input), *Utility* (connector), and *Output* (providing visual, auditory, or an alternative form of output) Foxels. Examples for Input Foxels are the *Button Foxel* and the *FSR (Force-Sensing Resistor) Foxel*, which is triggered by a person sitting on it. Examples for Output Foxels are the *LED Foxel*, the *PowerSwitch Foxel*, which contains a powerbar that can be used to control external devices, the *Projector Foxel*, which projects images and other content on walls, and the *Mail Foxel*, which allows data to be sent to a specified email address. We anticipate that there are many more Foxels that could be created, however, we believe that the options we chose provide an impression of the possibilities this interactive modular furniture concept provides.

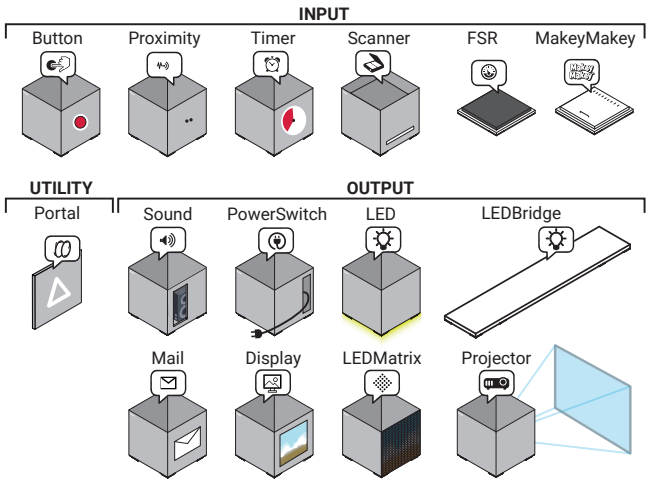

Figure 3: There are three categories of Foxels: Input Foxels wait for human input, Output Foxels produce visual or auditory signals, and Utility Foxels serve as connectors.

Communication Protocols

Foxels communicate on two levels. All cubes exchange metadata about their current arrangement. The more important stream of communication, however, governs the interaction between Input and Output Foxels. It is typically only exchanged between neighbours. As mentioned before, cubic constructive assemblies are usually quite restricted regarding the data they exchange and the functions they provide. Due to their intended use, Foxels were designed to handle complex content such as key/control commands and images in addition to simple signals.

We developed a model that abstracts the communication between the single units. Native functionality is encoded into a protocol by the *Input Foxel*, which is then decoded to be interpreted by the receiving *Output Foxel* (see Figure 4). The functions used to encode and decode information can be changed to alter the way of how Foxels interact with each other. For instance, a *Sound Foxel* can start to play a warning tone or a preconfigured playlist upon a button press. A Foxel may support multiple protocols, depending on its functionality. Each protocol must be paired with complementary functions. If two Foxels both can converse in multiple protocols, they will negotiate the one to use following a defined ranking. This enables the use of tangible programming even with complex and diverse data and ensures that there is a valid solution in almost every case. E.g., a pressure sensitive *FSR Foxel* makes a *Display Foxel* switch to the next stored image when being triggered.

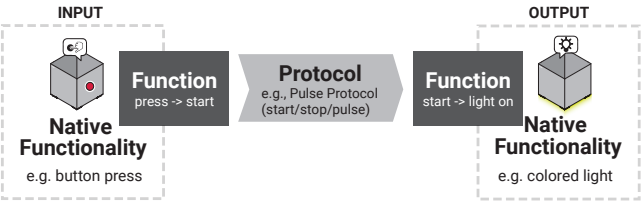

Figure 4: Mapping the native functionality to standardized protocols via functions ensures high compatibility and simplifies changing behaviors.

User Interaction

The main idea behind the Foxel interaction concept is to enable users with little to no programming skills to design and implement their own smart furniture environments with little effort. In the most basic way this should be possible without using an external configuration device (e.g. smartphone, tablet). When deploying the easy-to-understand tangible programming paradigm, we soon experienced the limitations of this concept. Inspired by the levels of interaction introduced in *roBlocks* [52], we started to investigate ways to enrich the concept with additional interaction concepts.

Tangible Programming. We designed Foxels so that functional arrangements can be created by vertical stacking and horizontal chaining following the principles of *tangible programming* [35, 40]. While all Foxels exchange data about the detected arrangement, in this section, we focus on the functional data that is passed between them. When being connected, a Foxel detects its neighbors and communicates with them by sending messages (events and/or data). Vice versa, a Foxel can receive messages from all its neighbors. While functional data is only sent to the immediate neighbor, Foxels forward the data they receive to their other neighbours. This way, signals can travel between a number of Foxels (cf. Figure 5, A).

Portal Connections. As the requirement of a physical connection can be restricting when building furniture setups, we also considered the communication across multiple Foxels even if the blocks are not physically connected to each other (cf. Figure 5, B). Therefore, we designed so-called *Foxel Portals*, that come in linked pairs with a symbol on the top for identification. In order to communicate across two disconnected Foxels, one simply has to mount two of these portals at a side of each of the two boxes. Portal connections can also be used to connect external devices, (e.g. interactive whiteboards) to Foxel arrangements. To do this, we use a special *Foxel Portal* that provides a link to the external device and enables dataflow.

Physical Interaction to Change Behavior. While tangible programming provides a large amount of possibilities, there are situations when it would be helpful to change the behavior of a single Foxel. For instance, as discussed before, the required output of a *Sound Foxel* can be quite different depending on the scenario. So, we implemented a way to toggle between the different behaviors of a Foxel. With a simple *knock-gesture* on the wooden box, the default behavior is altered (cf. Figure 5, C). In contrast to rotation- or tilting gestures, the knock-gesture also works within a Foxel arrangement. An embedded buzzer provides acoustic feedback whenever a knock is detected, whereas the number of beeps indicates the current behavior.

AR-Based Parameter Editing. Using a dedicated gesture only allows users to step through a preconfigured series of behaviors. To gain more control over individual parameters, other means of interaction are necessary. One interesting idea was to use AR to extend the interface of the Foxels virtually. Inspired by the work of Heun et al. [20], we implemented a prototype to investigate the idea of changing behavior parameters via a smartphone AR application (cf. Figure 5, D). The AR interface enables users to adapt and control the internal parameters that define a Foxel's behavior. The configurations are shown in a real-time preview.

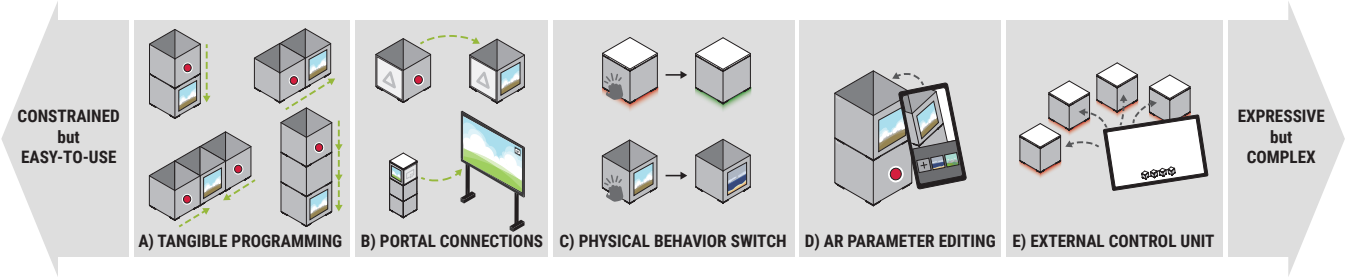

Figure 5: Overview of the different interaction methods supported by Foxels on an axis that represents the trade-off between constrained and expressive interaction options.

Controlling Multiple Foxels Remotely. For rapid switching between different activities (e.g. from group finding towards brainstorming in a workshop) it can be useful to configure multiple Foxels at once. The idea is that a context switch, made for instance via a tablet, smartphone, or a speech interface, instantaneously changes the behavior of all Foxels. We designed a tablet application that can detect arranged pieces of furniture consisting of more than one Foxel and enables users to assign specific higher level functions to them to create desired sequences of actions (cf. Figure 5, E).

5 IMPLEMENTATION

In this section we provide details on the physical components, structure, and software that form the Foxel Platform.

Foxel Base

The standard Foxel is a wooden box (side length of 36 cm; approx. 4 kg), enhanced with electronics to provide additional interactive functionality. Despite their different form factor, Tops, and Bridges (see Figure 2) have similar functionality. To demonstrate the concept and start from a solid base we build upon an existing modular furniture concept [14], which includes building blocks of various shapes that are compatible with each other (see Figure 2). Each Foxel contains a microcontroller and – dependent on the concrete type – various sensors or output elements. We support two widely-used IoT devices with off-the-shelf WiFi capabilities, i.e., NodeMCU [43] and Raspberry Pi (RPi) [13]. The choice depends on the requirements of the sensors and output elements as all basic functionality (discovery, topology tracking, communication, etc.) is implemented for both platforms.

Power

While our early prototypes used an integrated powerline to reduce maintenance effort, we turned away from this approach. This was due to safty-reasons as there are high amounts of current necessary to power components such as displays and due to the delays caused by the start-up each time a microcontroller / RPi was connected/re-connected. Thus, all building blocks (except for the LED-matrix display and the projector) are now powered using powerbanks of different capacities (2,500 - 13,000 mAh), depending on the power consumption of the block. When no additional external consumers (e.g. LED strips, displays) is connected, the Foxel will run up to 24 hours when housing an RPi and for multiple days when powering a NodeMCU on one single charge. The powerbanks are attached in a way so they can easily be swapped when running out of power.

Magnetic Connectors

For the topology tracking we considered and tested RFID-based and inductive approaches before settling on direct electric connections as they do not require additional expensive components. We first used pogo-pins, but as they are not well suited for lateral connections we switched to magnetic pins, which are used by many cubic constructive assemblies [19, 23, 31, 52]. As pointed out by Goh et al. [15], *physical coupling* can be frustrating for users, when the blocks seem perfectly aligned but still no detection occurs due to minor offsets. To compensate for the margin of error that comes with the large size of the Foxels, we use magnet balls mounted on springs that provide some flexibility when being connected (see Figure 6). The connectors were designed from off-the-shelf components and integrated into furniture building blocks using standard manual tools.

Topology Tracking

To ensure that any two building blocks can be snapped together in different arrangements, we propose a rotation-invariant connector layout (see Figure 6). As due to their design Foxels are always considered to stand upright, we use different connector layouts on the sides and top/bottom. Each sideways facing face contains four connectors (one sender, one receiver, two ground connectors), while there are two connectors on the top (one receiver, one ground connector), and eight connectors on the bottom (four senders, four ground connectors) to ensure tracking of all rotation

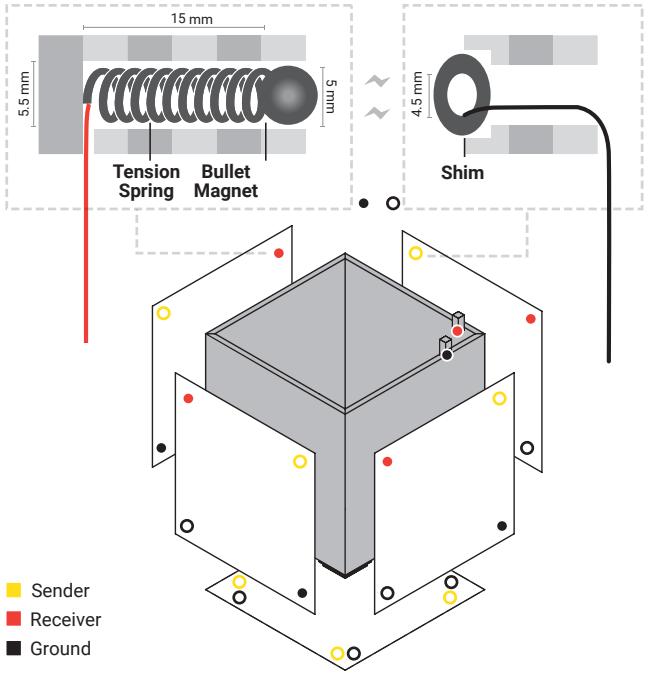

Figure 6: Arrangement of the magnetic connectors across the sides of a Foxel. Full circles represent spring connectors and rings shim connectors. While the top pins are used as sender (yellow) and receiver (red), the bottom pins on each side are used as ground (black circles/rings).

variations. With this layout, building blocks connected sideways can discover each other bidirectionally, while stacked blocks only share a unidirectional connection. This reduces the amount of necessary connectors, but requires building blocks to learn about their bottom neighbour indirectly. The building blocks continuously transmit a 32-bit ID using a custom PWM (*Pulse-Width-Modulation*) protocol at a low frequency (500 Hz). Once snapped together, they exchange their unique IDs via the sender/receiver connectors. When a block detects or loses its neighbor, it updates its stored topology and informs all other Foxels about the new configuration.

Foxel Network

To minimize the necessary infrastructure as pointed out in the design considerations, we did not want a system that requires an external server. Therefore, we implemented a peer-to-peer solution, which only requires the building blocks to connect to the same WiFi. When connected they can find themselves using *Multicast DNS* (mDNS) [57]. As soon as another block is found, it will be contacted to confirm its existence and get information about its neighbors. This and all subsequent communication, including the signal/data exchange is implemented using standard *HTTP requests*.

Foxel Software Platform

As the Foxel system supports different controllers the basic functionality (topology tracking, mDNS support, HTTP communication) is implemented on multiple platforms. This includes C/C++ for the NodeMCU, Python for the RPi and .NET/C# for the whiteboard sketching application and the tablet remote control. All implementations consist of a library part that encapsulates this core functionality and an application part that encodes the specific functionality of the Foxel or the device (e.g. tablet/whiteboard application). New functionality is implemented on the target device and then connected to the Foxel system via the according library calls. The AR prototype only uses a subset of the platform features and was developed in Android using ARCore.

6 APPLICATIONS

In this section, we show how Foxels can support an ideation workshop as an exemplary creative process by transitioning between a number of settings. Further individual interactive furniture arrangements highlight the system's versatility.

Vignette of an Ideation-Workshop

A typical start for an ideation workshops is the presentation of the problem statement. The **Presenter Podium** (see Figure 7a) uses the *Scanner Foxel* to scan prepared notes and visualizes them using the *Projector Foxel* to brief the participants on the workshop topic. For the subsequent brainstorming session the podium becomes part of a table within the **Brainstorming Studio**, where the participants sit around small tables while engaged in ideation. Figure 7b illustrates the setup where random images are rendered on the *Display Foxel* for inspiration. Pushing the *Button Foxel* below causes a switch to the next image. With the knock-gesture, users can change the behavior of the Foxel to e.g. render letters for the ABC method or portrait photos for a persona description. The *Display Foxel* remains the centerpiece in the **Idea Presentation Bar** (see Figure 7c) composed of a standing table with an integrated *Display Foxel* and a *Scanner Foxel*. This setup provides a casual atmosphere for the presentation of the created ideas. The paper sketches can be quickly digitized with the scanner and are automatically sent to the display for presentation. The display also acts as a storage keeping its data. This is useful in the next step, where a *MakeyMakey Foxel* [7] as input controller is placed on top of the *Display Foxel* to form a **Voting Station** (see Figure 7d). It lets participants browse and vote on the captured ideas stored in the *Display Foxel*. Finally, the **Physical Postbox** is used to send the best ideas to the participants via email using the *Mail Foxel*. This can either be done by combining it with the *Scanner Foxel* as shown in Figure 7e to create a analog/digital hybrid postbox or by stacking it on top of the *Display Foxel*.

Figure 7: Vignette of an Ideation-Workshop: (a) Presentation Podium, (b) Brainstorming Studio, (c) Idea Presentation Bar, (d) Voting Station, (e) Physical Postbox.

Additional Scenarios

This section highlights the utility of the interaction levels and the versatility of the Foxels for collaborative and other types of work.

Tangible Programming. Following the principle of tangible programming (cf. Section 4), building blocks simply have to be physically connected in order to communicate with each other without additional configuration. This principle can be seen in the **Reading Corner** in Figure 8a1, which uses a *Button Foxel* to trigger the *LED Bridge*. By replacing the button with a *Proximity Foxel* it can be easily modified to light up automatically whenever a person is detected underneath. Building a desk in front, the reading corner can be turned into an interactive **Personal Workspace**, where a *Timer Foxel* triggers the *LED Bridge* to provide lighting for the working person (see Figure 8a2) for a specified amount of time.

Portal Connections. Foxel Portals enable the communication across building blocks, even when they are not physically connected to each other. Figure 8b1, for example, illustrates an adaption of the previously discussed **Personal Workspace** enhanced with a *Portal Connection*, so that users can sit down on an interactive stool consisting of an *LED Foxel* and an *FSR Foxel*, which triggers the *LED Foxel Bridge* integrated in the desk. Similarly, Figure 8b2 illustrates the usage of a *Foxel Portal* that is linked to an **Interactive Whiteboard**. This example shows how external devices (e.g., smart whiteboards) can be linked using portals. Alternatively, the portal can also be attached to a *Scanner Foxel* to enable ad-hoc digitization and sharing of handwritten sketches.

Physical Interaction to Change Behavior. The behavior of a single building block can be re-configured by using the knock-gesture. Figure 8c1, demonstrates a simple **Light Indicator**. By default, the *LED Foxel* lights up red whenever a person sits on the *FSR Foxel* stacked on top, notifying all the co-workers that the person is immersed in work. On a knock, the LED changes its behavior, and switches the color to green, notifying the co-workers that the person would be available for discussion. Another example is the **Brainstorming Studio** (see Figure 7b) discussed in the previous section where knocking allows for switching between different brainstorming types. Finally, Figure 8c2 shows a stacked **Arcade Machine** as a more lighthearted example where a *MakeyMakey Foxel* is used as an input controller for an *LED Matrix Foxel* and knocking switches between games.

AR-Based Parameter Editing. Using an AR interface extends the configuration possibilities of a Foxel significantly. Due to the wild grain of the wooden boxes we were able to use them as natural features for the detection to identify the Foxel (via a reference photo from each side) as shown in Figure 8d1. When connected, the prototype shows an interface to control the Foxel at hand. For configuring the **Ambient Light Parameters** of an *LED Foxel* it shows a color picker (see Figure 8d2), while an **Image Upload** interface is provided for the *Display Foxel*. During configuration changes are displayed in a real-time preview (see Figure 8d2).

Controlling Multiple Foxels Remotely. Figures 8e1,2 illustrate how the same setup of assembled building blocks can be used within different applications when having more control over the behavior of the individual Foxels. A set of stools can be used to shuffle people in a **Group Finding** scenario (cf. Figure 8e2), for randomly picking a person to take the lead, which we call **Introduction Token**, or for assigning people into **Working Groups** in a warm-up exercise.

7 DISCUSSION

One of the main goals of our design was to investigate the trade-off between ease-of-use and versatility. Considering the limitations of tangible programming we believe that exploring extensions to this method was a fruitful approach to gain insight on the usefulness of constructive assemblies in a collaborative work context. Early feedback on situations where the interactive/functional requirements and the physical arrangement contradict each other lead to the development of *Foxel Portals* to gain more flexibility. Their physical nature nicely complements the tangible programming approach without breaking the interaction paradigm while providing the important ability to route signals between distant Foxels.

Figure 8: Application Scenarios showcasing Tangible Programming (a), Portal Connections (b), Physical Interaction via Knocking (c), AR-Based Parameter Editing (d), Controlling Multiple Foxels via a Tablet Application (e).

Foxels, such as the *Display* and *Projector Foxels* that can virtually show any content, made it clear that we need more control over their behaviors to gain more versatility and flexibility. This was partly achieved by introducing the different communication protocols as they provide a functional switch based on the received input. In addition, the knocking gesture as an explicit behavior switch opened up a number of new interaction possibilities. Still, this method did only allow switching between predefined behaviors and not for altering internal parameters of Foxels (e.g. target addresses of the *Mail Foxel*, playlist of the *Sound Foxel*). Using AR turned out as a great way how to address this issue. While it required us to depart from the 'no-external-device-for-configuration' paradigm, it truly offers a powerful interaction interface, which, however, stays optional in our system. Complex scenarios that follow a predefined sequence of events are generally hard to implement solely using tangible programming. We addressed this challenge by introducing an application that can control multiple Foxels remotely. While we believe that the idea of scripting the behavior of furniture arranged from modular building blocks is quite powerful, our implementation only scratches on the surface of what is possible. One interesting idea to explore is to enable users to script their own sequences of actions that react to the dynamic interaction with the modular furniture in the room.

8 LIMITATIONS

The primary limitation of the Foxel system is the effort needed to setup a desired furniture arrangement, which can scale up quickly as the number of building blocks increases. This means that pragmatically, the current Foxels system is best suited for groups of 3-6 persons. Scalability is also an issue on a technical level. While we are satisfied with the peer-to-peer architecture, one disadvantage of forgoing a central unit is the difficulty of debugging or deploying updates to multiple Foxels. This could be addressed by the remote control application in the future. While the current powering solution has its flaws, it could be improved by enabling charging via the connectors during inactivity.

9 CONCLUSION & FUTURE WORK

In this paper, we introduced *Foxels*, a modular, smart furniture concept that allows users to create their own interactive furniture environments by arranging cubic building blocks. Considering the trade-off between ease-of-use and expressiveness we investigated a number of interaction methods extending the tangible programming paradigm to enhance the versatility of our smart furniture set. We believe that the large number of applications shows the utility of our work and will hopefully inspire researchers and practitioners to further explore the concept of modular, interactive furniture.

For future work, we would like to iterate on the Foxel hardware and speed up the creation process to implement a larger number of Foxels. Moreover, we see a lot of potential in exploring ways of scripting the furniture behavior so that it can react dynamically to changes of the arrangement. In addition, it would be interesting to see how users use Foxels when performing creative collaborative work, so that we can get insights regarding the capabilities and benefits of a modular smart furniture platform.

ACKNOWLEDGEMENTS

This work was supported by the Austrian Research Promotion Agency GmbH (FFG) project #861079 - Innovation Playground. We thank Didi Lenz and Christian Lettner (Bene AG) for providing us with Bene Pixels.

REFERENCES

[1] Bert Arnrich, Cornelia Setz, Roberto La Marca, Gerhard Tröster, and Ulrike Ehlert. 2010. What Does Your Chair Know About Your Stress Level? *IEEE Transactions on Technology in Biomedicine* 14, 2 (2010), 207–214. https://doi.org/10.1109/TITB.2009.2035498

[2] Thomas Augsten, Konstantin Kaefer, René Meusel, Caroline Fetzer, Dorian Kanitz, Thomas Stoff, Torsten Becker, Christian Holz, and Patrick Baudisch. 2010. Multitoe: High-Precision Interaction with Back-Projected Floors Based on High-Resolution Multi-Touch Input. In *Proceedings of the 23nd Annual ACM Symposium on User Interface Software and Technology (UIST '10)*. ACM Press, New York, NY, USA, 209–218. https://doi.org/10.1145/1866029.1866064

[3] Ayah Bdeir. 2009. Electronics As Material: LittleBits. In *Proceedings of the 3rd International Conference on Tangible and Embedded Interaction (TEI '09)*. ACM Press, New York, NY, USA, 397–400. https://doi.org/10.1145/1517664.1517743

[4] Xiaojun Bi, Tovi Grossman, Justin Matejka, and George Fitzmaurice. 2011. Magic Desk: Bringing Multi-Touch Surfaces into Desktop Work. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11)*. ACM Press, New York, NY, USA, 2511–2520. https://doi.org/10.1145/1978942.1979309

[5] Jeeyong Chung, Kyungeun Min, and Woohun Lee. 2013. CUBEMENT: Democratizing Mechanical Movement Design. In *Proceedings of the 8th International Conference on Tangible, Embedded and Embodied Interaction (TEI '14)*. ACM Press, New York, NY, USA, 81–84. https://doi.org/10.1145/2540930.2540933

[6] Michael Cohen. 1999. The Internet Chair. *International Journal of Human-Computer Interaction* 15, 2 (1999), 297–311. https://doi.org/10.1207/S15327590IJHC1502_7

[7] Beginner's Mind Collective and David Shaw. 2012. Makey Makey: Improvising Tangible and Nature-based User Interfaces. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction (TEI '12)*. ACM Press, New York, NY, USA, 367–370. https://doi.org/10.1145/2148131.2148219

[8] Alexandru Dancu, Catherine Hedler, Stig Anton Nielsen, Hanna Frank, Zhu Kening, Axel Pelling, Adviye Ayça Ünlüer, Christian Carlsson, Max Witt, and Morten Fjeld. 2015. Emergent Interfaces: Constructive Assembly of Identical Units. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM Press, New York, NY, USA, 451–460. https://doi.org/10.1145/2702613.2732509

[9] Kazam Design. 2014. BrickBox Modular Storage. www.brickbox.es/en

[10] Scott Elrod, David Lee, Rich Gold, David Goldberg, Frank Halasz, Elin Pedersen, Ken Pier, John Tang, and Brent Welch. 1992. Liveboard: A Large Interactive Display Supporting Group Meetings, Presentations, and Remote Collaboration. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '92)*. ACM Press, New York, NY, USA, 599–607. https://doi.org/10.1145/142750.143052

[11] Alex Endert, Patrick Fiaux, Haeyong Chung, Michael Stewart, Christopher Andrews, and Chris North. 2011. ChairMouse: Leveraging Natural Chair Rotation for Cursor Navigation on Large, High-Resolution Displays. In *CHI '11 Extended Abstracts on Human Factors in Computing Systems (CHI EA '11)*. ACM Press, New York, NY, USA, 571–580. https://doi.org/10.1145/1979742.1979628

[12] George W. Fitzmaurice, Hiroshi Ishii, and William A. S. Buxton. 1995. Bricks: Laying the Foundation for Grapable User Interfaces. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '95)*. ACM Press, New York, NY, USA, 442–449. https://doi.org/10.1145/223904.223964

[13] Raspberry Pi Foundation. 2012. Raspberry Pi. www.raspberrypi.org

[14] BENE GmbH. 2017. BENE Pixel Furniture Concept. www.bene.com/en/office-furniture-concepts/office-furniture/pixel-en

[15] Wooi Boon Goh, L. L. Chamara Kasun, Fitriani, Jacquelyn Tan, and Wei Shou. 2012. The i-Cube: Design Considerations for Block-based Digital Manipulatives and Their Applications. In *Proceedings of the Designing Interactive Systems Conference (DIS '12)*. ACM Press, New York, NY, USA, 398–407. https://doi.org/10.1145/2317956.2318016

[16] Jens Emil Grønbæk, Henrik Korsgaard, Marianne Graves Petersen, Morten Henriksen Birk, and Peter Gall Krogh. 2017. Proxemic Transitions: Designing Shape-Changing Furniture for Informal Meetings. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. Press, ACM, New York, NY, USA, 7029–7041. https://doi.org/10.1145/3025453.3025487

[17] François Guimbretière, Maureen Stone, and Terry Winograd. 2001. Fluid Interaction with High-Resolution Wall-Size Displays. In *Proceedings of the 14th International Symposium on User Interface Software and Technology (UIST '01)*. ACM Press, New York, NY, USA, 21–30. https://doi.org/10.1145/502348.502353

[18] Doris Hausen, Sebastian Boring, and Saul Greenberg. 2013. The Unadorned Desk: Exploiting the Physical Space around a Display as an Input Canvas. In *Proceedings of the 14th IFIP TC13 Conference on Human-Computer Interaction (INTERACT '13)*. Springer-Verlag, Berlin, Germany, 140–158. https://doi.org/10.1007/978-3-642-40483-2_10

[19] Felix Heibeck. 2013. Cuboino. Extending Physical Games. An Example. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems on (CHI EA '13)*. ACM Press, New York, NY, USA, 2935–2938. https://doi.org/10.1145/2468356.2479578

[20] Valentin Heun, James Hobin, and Pattie Maes. 2013. Reality Editor: Programming Smarter Objects. In *Proceedings of the 2013 ACM Conference on Pervasive and Ubiquitous Computing Adjunct Publication (UbiComp '13 Adjunct)*. ACM Press, New York, NY, USA, 307–310. https://doi.org/10.1145/2494091.2494185

[21] Otmar Hilliges, Lucia Terrenghi, Sebastian Boring, David Kim, Hendrik Richter, and Andreas Butz. 2007. Designing for Collaborative Creative Problem Solving. In *Proceedings of the 6th ACM SIGCHI Conference on Creativity & Cognition (C&C '07)*. ACM Press, New York, NY, USA, 137–146. https://doi.org/10.1145/1254960.1254980

[22] Steve Hodges and Gifford Louie. 1994. Towards the Interactive Office. *Conference Companion on Human Factors in Computing Systems* (1994), 305–306. https://doi.org/10.1145/259963.260512

[23] Meng-Ju Hsieh, Rong-Hao Liang, Da-Yuan Huang, Jheng-You Ke, and Bing-Yu Chen. 2018. RFIBricks: Interactive Building Blocks Based on RFID. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM Press, New York, NY, USA, 1–10. https://doi.org/10.1145/3173574.3173763

[24] Google Inc. 2010. The Decisive Decade – How the Acceleration of Ideas will Transform the Workplace by 2020. https://lp.google-mkto.com/rs/google/images/Google-Decisive-Decade-Report-Future-Foundation.pdf

[25] Steelcase Inc. 2019. FlexBox. Website. https://www.steelcase.com/eu-en/products/cupboards-cabinets/flexbox

[26] Hiroshi Ishii. 2008. Tangible Bits: Beyond Pixels. In *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction (TEI '08)*. ACM Press, New York, NY, USA, xv–xxv. https://doi.org/10.1145/1347390.1347392

[27] Hiroshi Ishii and Brygg Ullmer. 1997. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In *Proceedings of the SIGCHI conference on Human factors in computing systems (CHI '97)*. ACM Press, New York, NY, USA, 234–241. https://doi.org/10.1145/258549.258715

[28] Robert J.K. Jacob, Audrey Girouard, Leanne M. Hirshfield, Michael S. Horn, Orit Shaer, Erin Treacy Solovey, and Jamie Zigelbaum. 2008. Reality-Based Interaction: A Framework for Post-WIMP Interfaces. In

Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '08). ACM Press, New York, NY, USA, 201–210. https://doi.org/10.1145/1357054.1357089

[29] Hans Christian Jetter, Harald Reiterer, and Florian Geyer. 2014. Blended Interaction: Understanding Natural Human-Computer Interaction in Post-WIMP Interactive Spaces. *Personal and Ubiquitous Computing* 18, 5 (oct 2014), 1139–1158. https://doi.org/10.1007/s00779-013-0725-4

[30] Brad Johanson, Armando Fox, and Terry Winograd. 2002. The Interactive Workspaces Project: Experiences with Ubiquitous Computing Rooms. *IEEE Pervasive Computing* 1, 2 (apr 2002), 67–74. https://doi.org/10.1109/MPRV.2002.1012339

[31] Ichiga Kawaguchi. 2016. Hikari Tsumiki 2.0. https://gugen.jp/entry2016/2016-121

[32] Majeed Kazemitabaar, Jason McPeak, Alexander Jiao, Liang He, Thomas Outing, and Jon E Froehlich. 2017. MakerWear: A Tangible Approach to Interactive Wearable Creation for Children. In *Proceedings of the 2017 CHI Conference on Human Factors in Computing Systems (CHI '17)*. ACM Press, New York, NY, USA, 133–145. https://doi.org/10.1145/3025453.3025887

[33] Scott R. Klemmer, Mark W. Newman, Ryan Farrell, Mark Bilezikjian, and James A. Landay. 2001. The Designers' Outpost: A Tangible Interface for Collaborative Web Site. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST '01)*. ACM Press, 1–10. https://doi.org/10.1145/502348.502350

[34] Naohiko Kohtake, Ryo Ohsawa, Takuro Yonezawa, Yuki Matsukura, Masayuki Iwai, Kazunori Takashio, and Hideyuki Tokuda. 2005. u-Texture: Self-organizable Universal Panels for Creating Smart Surroundings. In *Proceedings of the 7th International Conference on Ubiquitous Computing (UbiComp'05)*. Springer-Verlag, Berlin, Heidelberg, 19–36. https://doi.org/10.1007/11551201_2

[35] Thomas Kubitza and Albrecht Schmidt. 2017. meSchup: A Platform for Programming Interconnected Smart Things. *Computer* 50, 11 (nov 2017), 38–49. https://doi.org/10.1109/MC.2017.4041350

[36] Kevin Lefeuvre, Soeren Totzauer, Michael Storz, Albrecht Kurze, Andreas Bischof, and Arne Berger. 2018. Bricks, Blocks, Boxes, Cubes, and Dice: On the Role of Cubic Shapes for the Design of Tangible Interactive Devices. In *Proceedings of the 2018 Designing Interactive Systems Conference (DIS '18)*. ACM Press, New York, NY, USA, 485–496. https://doi.org/10.1145/3196709.3196768

[37] Joanne Leong, Florian Perteneder, Hans-Christian Jetter, and Michael Haller. 2017. What a life! Building a Framework for Constructive Assemblies. In *Proceedings of the 11th International Conference on Tangible, Embedded, and Embodied Interaction (TEI '17)*. ACM Press, New York, NY, USA, 57–66. https://doi.org/10.1145/3024969.3024985

[38] Rong-Hao Liang, Liwei Chan, Hung-Yu Tseng, Han-Chih Kuo, Da-Yuan Huang, De-Nian Yang, and Bing-Yu Chen. 2014. GaussBricks: Magnetic Building Blocks for Constructive Tangible Interactions on Portable Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM Press, New York, NY, USA, 3153–3162. https://doi.org/10.1145/2556288.2557105

[39] Stanley McChrystal, David Silverman, Tantum Collins, and Chris Fussell. 2015. *Team of Teams: New Rules of Engagement for a Complex World*. Penguin.

[40] Timothy S. McNerney. 2004. From Turtles to Tangible Programming Bricks: Explorations in Physical Language Design. *Personal and Ubiquitous Computing* 8, 5 (sep 2004), 326–337. https://doi.org/10.1007/s00779-004-0295-6

[41] David Merrill, Jeevan Kalanithi, and Pattie Maes. 2007. Siftables: Towards Sensor Network User Interfaces. In *Proceedings of the 1st International Conference on Tangible and Embedded Interaction (TEI '07)*. ACM Press, New York, NY, USA, 75–78. https://doi.org/10.1145/1226969.1226984

[42] Bilge Mutlu, Andreas Krause, Jodi Forlizzi, Carlos Guestrin, and Jessica Hodgins. 2007. Robust, Low-Cost, Non-Intrusive Sensing and Recognition of Seated Postures. *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology* (2007), 149–158. https://doi.org/10.1145/1294211.1294237

[43] NodeMcu. 2018. NodeMcu. www.nodemcu.com

[44] Robert J. Orr and Gregory D. Abowd. 2000. The Smart Floor: A Mechanism for Natural User Identification and Tracking. *CHI '00 Extended Abstracts on Human Factors in Computing Systems* (2000), 275–276. https://doi.org/10.1145/633292.633453

[45] Joseph Paradiso and Matthew Reynolds. 1997. The Magic Carpet: Physical Sensing for Immersive Environments. In *CHI '97 Extended Abstracts on Human Factors in Computing Systems (CHI EA '97)*. ACM Press, New York, NY, USA, 277–278. https://doi.org/10.1145/1120212.1120391

[46] Kathrin Probst, David Lindlbauer, Michael Haller, Bernhard Schwartz, and Andreas Schrempf. 2014. A Chair as Ubiquitous Input Device: Exploring Semaphoric Chair Gestures for Focused and Peripheral Interaction. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM Press, New York, NY, USA, 4097–4106. https://doi.org/10.1145/2556288.2557051

[47] Jun Rekimoto and Masanori Saitoh. 1999. Augmented Surfaces: A Spatially Continuous Work Space for Hybrid Computing Environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM Press, New York, NY, USA, 378–385. https://doi.org/10.1145/302979.303113

[48] Jun Rekimoto, Brygg Ullmer, and Haruo Oba. 2001. DataTiles: A Modular Platform for Mixed Physical and Graphical Interactions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '01)*. ACM Press, New York, NY, USA, 269–276. https://doi.org/10.1145/365024.365115

[49] Bruce Richardson, Krispin Leydon, Mikael Fernström, and Joseph A. Paradiso. 2004. Z-Tiles: Building Blocks for Modular, Pressure-sensing Floorspaces. In *CHI '04 Extended Abstracts on Human Factors in Computing Systems (CHI EA '04)*. ACM Press, New York, NY, USA, 1529–1532. https://doi.org/10.1145/985921.986107

[50] Modular Robotics. 2019. Cubelets. www.modrobotics.com

[51] Domnic Savio and Thomas Ludwig. 2007. Smart Carpet: A Footstep Tracking Interface. *21st International Conference on Advanced Information Networking and Applications Workshops* (2007), 754–760. https://doi.org/10.1109/AINAW.2007.338

[52] Eric Schweikardt and Mark D. Gross. 2006. roBlocks: A Robotic Construction Kit for Mathematics and Science Education. In *Proceedings of the 8th International Conference on Multimodal Interfaces (ICMI '06)*. ACM Press, New York, NY, USA, 72–75. https://doi.org/10.1145/1180995.1181010

[53] Thomas Seifried, Michael Haller, Stacey D. Scott, Florian Perteneder, Christian Rendl, Daisuke Sakamoto, and Masahiko Inami. 2009. CRISTAL: A Collaborative Home Media and Device Controller Based on a Multi-touch Display. In *Proceedings of the ACM International Conference on Interactive Tabletops and Surfaces (ITS '09)*. ACM Press, New York, NY, USA, 33–40. https://doi.org/10.1145/1731903.1731911

[54] Jennifer G. Sheridan. 2003. Exploring Cube Affordance: Towards a Classification of Non-Verbal Dynamics of Physical Interfaces for Wearable Computing. In *IEE Eurowearable '03*, Vol. 2003. IET, 113–118. https://doi.org/10.1049/ic:20030156

[55] Ali A. N. Shirehjini, Abdulsalam Yassine, and Shervin Shirmohammadi. 2014. Design and Implementation of a System for Body Posture Recognition. *Multimedia Tools and Applications* 70, 3 (2014), 1637–1650. https://doi.org/10.1007/s11042-012-1137-6

[56] Norbert A. Streitz, Jörg Geißler, Torsten Holmer, Shin'ichi Konomi, Christian Müller-Tomfelde, Wolfgang Reischl, Petra Rexroth, Peter

Seitz, and Ralf Steinmetz. 1999. i-LAND: An interactive Landscape for Creativity and Innovation. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '99)*. ACM Press, New York, NY, USA, 120–127. https://doi.org/10.1145/302979.303010

[57] Stuart Cheshire. 2001. Multicast DNS. www.multicastdns.org

[58] IKEA Systems. 2000. IKEA Kallax. www.ikea.com/us/en/catalog/categories/series/27534

[59] Peter Tandler, Thorsten Prante, Christian Müller-Tomfelde, Norbert Streitz, and Ralf Steinmetz. 2001. Connectables: Dynamic Coupling of Displays for the Flexible Creation of Shared Workspaces. In *Proceedings of the 14th Annual ACM Symposium on User Interface Software and Technology (UIST '01)*. ACM Press, New York, NY, USA, 11–20. https://doi.org/10.1145/502348.502351

[60] Brygg Ullmer, Hiroshi Ishii, and Dylan Glas. 1998. mediaBlocks: Physical Containers, Transports, and Control for Online Media. In *Proceedings of the 25th Annual Conference on Computer Graphics and Interactive Techniques (SIGGRAPH '98)*. ACM Press, New York, NY, USA, 379–386. https://doi.org/10.1145/280814.280940

[61] Jo Vermeulen, Kris Luyten, Karin Coninx, Nicolai Marquardt, and Jon Bird. 2015. Proxemic Flow: Dynamic Peripheral Floor Visualizations for Revealing and Mediating Large Surface Interactions. *Lecture Notes in Computer Science* 9299 (2015), 264–281. https://doi.org/10.1007/978-3-319-22723-8_22

[62] Luke Vink, Viirj Kan, Ken Nakagaki, Daniel Leithinger, Sean Follmer, Philipp Schoessler, Amit Zoran, and Hiroshi Ishii. 2015. TRANSFORM As Adaptive and Dynamic Furniture. In *Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15)*. ACM Press, New York, NY, USA, 183. https://doi.org/10.1145/2702613.2732494

[63] Ryoichi Watanabe, Yuichi Itoh, Masatsugu Asai, Yoshifumi Kitamura, Fumio Kishino, and Hideo Kikuchi. 2004. The Soul of ActiveCube - Implementing a Flexible, Multimodal, Three-Dimensional Spatial Tangible Interface. *Computers in Entertainment* 2, 4 (oct 2004). https://doi.org/10.1145/1037851.1037874

[64] Pierre Wellner. 1994. Interacting with Paper on the DigitalDesk. *Commun. ACM* 36, 7 (1994), 87–96. https://doi.org/10.1145/159544.159630

[65] Mounia Ziat, Josh Fridstrom, Kurt Kilpela, Jonathan Fancher, and James J. Clark. 2014. InGrid: Interactive Grid Table. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14)*. ACM Press, New York, NY, USA, 559–562. https://doi.org/10.1145/2559206.2574821