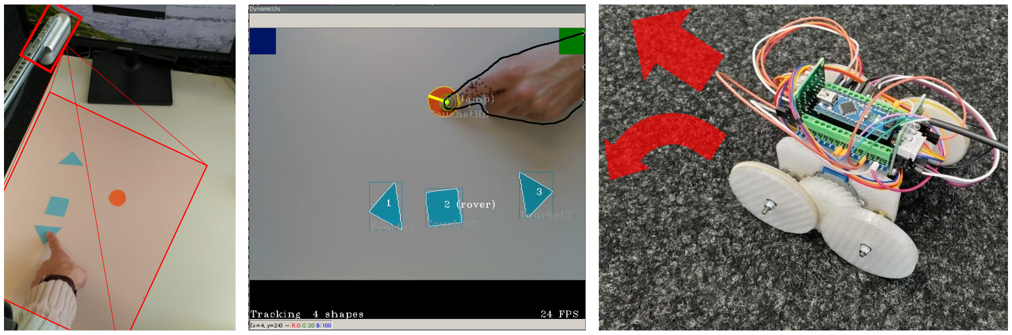
Tailored Controls: Creating Personalized Tangible User Interfaces from Paper

Vincent Becker
Department of Computer Science, ETH Zurich
Switzerland
vincent.becker@inf.ethz.ch

Sandro Kalbermatter
Department of Computer Science, ETH Zurich
Switzerland
sandroka@ethz.ch

Simon Mayer
University of St.Gallen and ETH Zurich
Switzerland
simon.mayer@unisg.ch

Gábor Sörös
Nokia Bell Labs
Budapest, Hungary
gabor.soros@nokia-bell-labs.com



(a) (b) (c)

Figure 1. We present a concept and a prototype of a system for creating personalized user interfaces from paper. Users can cut out preferred shapes from paper and add interaction functionalities to them. The paper shapes and the user's fingers are tracked by an RGBD camera mounted above the interaction surface (Figure 1a). Our processing pipeline recognizes and measures manipulations of the shapes (Figure 1b), such as movements and touches, which can be mapped to certain functions according to the assigned interactions, e.g. for controlling a toy rover as illustrated in Figure 1c.

ABSTRACT

User interfaces rarely adapt to the specific user preferences or the task at hand. We present a method that allows to quickly and inexpensively create personalized interfaces from plain paper. Users can cut out shapes and assign control functions to these paper snippets via a simple configuration interface. After configuration, control takes place entirely through the manipulation of the paper shapes, providing the experience of a tailored tangible user interface. The shapes and assignments can be dynamically changed during use. Our system is based on markerless tracking of the user's fingers and the paper shapes on a surface using an RGBD camera mounted above the interaction space, which is the only hardware sensor required. Our approach and system are backed up by two studies where we determined what shapes and interaction abstractions users prefer, and verified that users can indeed employ our system to build real applications with paper snippet interfaces.

Author Keywords

Personalized user interfaces; Tangible interfaces; User interface framework

CCS Concepts

•**Human-centered computing** → **Human computer interaction (HCI);** *Haptic devices;* User studies;

INTRODUCTION

Tangible user interfaces, such as buttons and knobs, enjoy widespread use for controlling appliances: they naturally arose as controls of purely mechanical devices before the emergence of digital products, and virtual implementations of the interaction patterns they express are ubiquitous in graphical user interfaces (GUIs) for software as well. In fact, with the spread of displays in the past decades, more and more interactions have been transferred from tangible mechanical controls or remote controls to GUIs. These have the advantage of being able to display changing content and are therefore highly flexible and widely applicable. However, GUIs often do not induce any haptic sensation, which humans innately prefer over passive interfaces [42] and moreover GUIs usually do not provide common physical interactions humans are used to, such as easily moving, adding, or removing elements. This has given rise to the creation and proliferation of tangible user interfaces

(TUIs) in the past two decades with the aim of making interactions more natural and binding virtual elements and functions to real objects. TUIs, which can take almost any physical form, are also conducive to the fundamental goal of ubiquitous computing: making computers (and their controllers) disappear physically and mentally into users' environments [37].

However, TUIs usually cannot adapt to a specific user nor to the task the user intends to carry out, simply because they are *hardware* devices and are thus constrained to – and by – their physical form. Consequently, many tangible interfaces that we use in our daily lives (e.g., traditional TV remote controls) are often built to provide interactors for all functions of the controlled device, even if these functions are used only rarely – or never. This can be overcome with interfaces that are able to change their shape during interaction [11] [21]. However, this currently requires complex and expensive electro-mechanical systems.

An intriguing solution for adaptive user interfaces is to let users themselves create and assemble their personalized user interfaces from inexpensive everyday materials (e.g., paper, cardboard, or playdough), and enabling them to link these TUIs to device functions. This would allow users to create exactly the interfaces they require, at their desired level of complexity.

Such an approach enables various applications. First, it benefits the process of user interface prototyping, where paper snippets are commonly used, by making it possible to add actual functionality to the paper shapes and thus being able to rapidly test designed interfaces – this enables us to bridge the "ideation" and "implementation" steps in the design process, which are typically separate [4]. Furthermore, the interfaces can easily be reconfigured, thereby enabling quicker iterations in the design process and an earlier realization of potentially erroneous approaches as the UI does not have to be implemented first, which entails high development cost.

Second, an approach like this could be used in various real applications for example in the domain of building control, where it would permit the creation of *passive* control interfaces for lights, blinds, AC, and other devices that are attached to surfaces and walls without requiring infrastructure such as power and control lines. This would provide an installation that can be easily reconfigured, which becomes necessary as building automation companies are searching for ways to flexibly rearrange the spaces we live and work in, in response to user preferences.

Third, the development towards the increasingly automated operation of industrial equipment entails less frequent direct interactions of workers with machines in industrial shopfloors. In the wake of this process, adaptive, *tactical*, user interfaces would enable us to avoid cluttering workshops with UIs that are operated infrequently (and might not even be suitable for a particular user). Rather, they would enable workers bring their own interfaces with them, quickly assemble and configure them, and use them to interact with machines throughout the production space.

Contributions

This paper presents a system that allows users to create their own, personalized, user interfaces by cutting out arbitrary paper snippets and assembling them on a suitable surface (e.g., a table). We track both the shapes and the user's fingers with a RGBD (colour and depth) camera without the need to employ any markers, enabling a natural control of devices and permitting a wide range of interaction abstractions. In addition, our system allows for dynamically adding and removing paper snippets from the interface.

We determined the types of interactions that such a system should enable through an elicitation study with 20 participants, and deduced 25 abstract interactions, out of which the 13 most common ones were implemented in our prototype – these interaction abstractions cover more than 92% of all observed interactions from the elicitation study. When creating an interface using our system, users cut out paper shapes and assign interaction abstractions to them; being tracked by the RGBD camera, these shapes respond to corresponding user interactions by emitting events that carry the relevant interaction information (e.g., the *distance* a shape was moved; the *angle* it was rotated; the *duration* it was tapped, etc.). These events are subsequently passed to any (networked) consumer where they trigger application functionality – this can be configured using a common dataflow programming language. We evaluated our complete prototype in another user study, which showed that participants were able to use our system to successfully create fully functioning interfaces for several different applications, such as controlling a sound system.

RELATED WORK

Already in 1995, Fitzmaurice et al. [10] presented *Bricks*, an approach that connects physical bricks, laid out on a display surface, to virtual graphical counterparts depicted on the display. The virtual shapes can be manipulated, e.g. moved or rotated by manipulating the corresponding bricks. Shortly after, Ishii et al. [17] presented their vision of *Tangible Bits*, aiming at a coupling of digital information to physical objects and thereby making this information tangible. This coupling should also allow the manipulation of the digital state by the manipulation of the objects. Our approach fits this concept well by coupling components made from simple and inexpensive material, in our case paper shapes created by users themselves, to digital control components.

Dedicated tangible interfaces

Complex tangible interfaces that feature high representation capabilities are enabled by the interface changing its shape [11]. This may be combined with an augmented reality (AR) application [24] to add a tangible representation of the AR content. Shape-changing tangible interfaces can also be used to combine different elements such as sliders or knobs in a single component [21]. Studies on shape-changing interfaces can be found in [7, 22]. These interfaces provide an interesting approach in the area of dynamic and adaptable interfaces, however, they also entail relatively complex hardware and high cost. We intend to follow the opposite direction. Users should be enabled to create their own personalized interfaces from simple, inexpensive material, and should be able to easily change and recreate interfaces themselves.

Printable circuits

One possibility for using paper for interaction elements is to print electronic circuits onto it. A number of projects have recently explored combining paper with electronics. For example, Instant Inkjet Circuits [18] and Printem [5] are simply created by a modified printer. The projects Sketching in circuits [30], PaperPulse [32], CodeCollage [31], and Light-ItUp [14] specifically explore how users combine various interactive paper electronics elements. The recent Pulp non-fiction [41] extends regular paper with capacitive touch and pen sensing. Some paper electronics are even available in commercial products such as Chibitronics[1] or DynamicLand[2]. The fundamental difference to our approach is that the paper, on which the circuits were printed to glued or stuck to, has to stay in a single piece, i.e. the user cannot create independent interaction elements which can be moved or reassembled, and as such the paper circuits act more as printouts of graphical user interfaces than tangible interfaces.

Using everyday objects as interfaces

In addition to the creation of dedicated tangible user interfaces as mentioned above, arbitrary objects in the user's environment can also be used as tangible interfaces. Pohl et al. [29] investigated the space of everyday objects that could be useful as tangible controllers and found that there is both a diverse set of objects available and potential use for them. This concept is employed in several works in order to assign different functions to different objects, e.g. in painting applications [12, 34]. Often, a workspace is instrumented with overhead cameras to track simple objects and their movements and map them to control functions, for example, iCon [6] repurposes physical objects to control other objects by augmenting them with a trackable paper label. Corsten et al. [8] do the same without markers, using an over-head depth camera for pose estimation. Funk et al. [13] present an augmented workspace in which everyday objects can be combined into personalized control widgets. The idea of using objects as tangible interfaces is also used in mobile settings by employing tablets [1, 15, 16] or head-mounted AR displays [2]. In contrast to visual sensing, project Zanzibar [35] contributes a portable rubber mat with embedded NFC readers that recognize various objects placed on top. This mat can be used to create reconfigurable tangible controls similar to our system.

Similar to our work, using simple objects in the user's environment also has the goal of using readily available components as TUIs at no (or hardly any) extra cost. Nevertheless, instead of using arbitrary objects, we intend to let users design their own interfaces – according to their personal preferences and abilities. Furthermore, our interfaces do not interfere with the normal use of objects (e.g., drinking from a mug that is also used as a "dial" TUI).

[1] https://chibitronics.com/
[2] https://dynamicland.org/

Reconfigurable interfaces from simple materials

Besides, using existing physical objects as interfaces, there are other approaches which allow personalized interfaces and reconfiguration as in ours. Kelly et al. [19] recently presented *ARcadia*, which is intended for rapid prototyping. Control elements such as buttons, wheels, or sliders can be cut out from paper or cardboard, arranged and then assigned a certain function in a browser application. Fiducial markers are attached to the shapes in order to recognize and track them using a webcam, e.g. a laptop camera. The laptop can be used to configure the function assignments at the same time, as ARcadia requires an additional input device for this task. Using markers has the advantage of being able to track the shapes very accurately and giving the shapes a unique identity, however, the user has to provide the markers for every shape in the interface construction process, e.g. by printing them. In contrast to ARcadia, our intention was to make the TUI creation process as simple as possible. Hence, we avoid the use of markers and directly track the shapes and the user's fingers. We furthermore enable a broader range of interactions, as we (1) track the fingers separately from the markers, thereby allowing users to simply tap on shapes – or swipe over them – instead of having to cover an entire marker, and (2) study which interaction behavior is employed by participants in an elicitation study resulting in a wider set of possible interactions which goes beyond simple movements and rotations.

The VoodooSketch [3] system allows users to draw user interface elements on light-reflective paper that are recognized by an over-head camera like in our setting. Our system provides similar flexibility and ease of use, the main difference is in creating the widgets via drawing or cutting and the selection of supported materials.

Olberling et al. [28] work relies on printable circuits as mentioned above but furthermore allows cutting interaction elements from the circuit-augmented paper. They presented a special wiring topology for multitouch sensors that make them robust to cutting, so users can define and cut personalized UI elements. An advantage of our approach is that no special material is required. The sensing part of our system, an RGBD camera, can be reused and any kind of paper can be used for cutting out shapes. Moreover, with the camera tracking, our approach allows the position and the movement of the widgets to be taken into account when defining controls.

Object tracking

An important component of these approaches is the recognition and tracking of the elements, which may be printed circuits or real-world object, as mentioned above. There are various technologies for tracking user-defined elements on a surface, all having their advantages and disadvantages. With cameras, for example, ShadowTracking [9] recognizes silhouette shadows from above a well-lit touch screen, VoodoSketch [3] and RetroDepth [20] apply special retro-reflective surfaces. DIRECT [39] combines RGBD and IR input. In our approach, we do not only track the position of the finger, but also its orientation. Instead of vision, also other modalities such as radar sensing [40] have been used. Voelker et al. [36] presented Passive Untouched Capacitive Widgets that can be

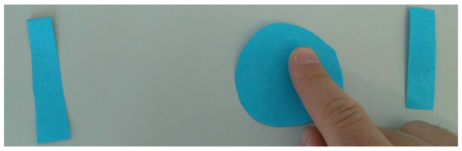


Figure 2. An example for a paper interface: a trackbar.

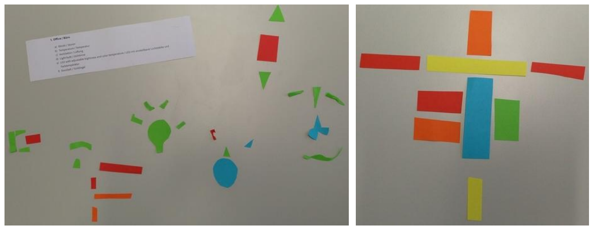


Figure 3. Two examples of interfaces participants designed. Left: An interface for controlling a smart workplace, similar to what one might expect. Right: A complex interface for controlling a movie.

tracked on top of unmodified capacitive touch screens. SLAP-Widgets [38] provide tangible transparent silicon elements that can be reconfigured on the fly as various UI elements. The actual functions are projected onto the blank elements with an overhead projector. The elements are recognized from the bottom via IR-reflective identifiers. In comparison with our approach, the elements cannot be defined by the user in a simple manner.

ELICITATION STUDY

Technically, our approach is based on visually tracking user-cut paper shapes. The manipulations users perform (i.e. movements, rotations, touches, etc.) are then mapped to specific value changes or discrete actions. For example, a user might create a paper circle intending it to be a knob he/she can use to control the volume of a sound system. The user should also be able to assemble basic shapes to groups, which then represent an interface element, e.g. one could use two rectangles and a circle to create a trackbar as shown in Figure 2. In order to infer this mapping from a certain shape or group, we either require a fixed set of shapes and groups with a fixed mapping, which we can automatically recognize, or a set of mappings the user can assign to the shapes. The first case has the advantage of allowing the user to directly employ a shape or group without having to assign an interaction behavior to it, but would limit him/her to the fixed set of known shapes. This observation is closely related to the concept of affordance, introduced by Donald Norman [26], which describes the interaction possibilities a human perceives when encountering an object or another thing in his/her environment, e.g. that one can throw a ball or push a button. We are specifically interested in which affordances different paper shapes or groups have.

Our initial hypothesis was that there would be a common ground on the affordances of certain shapes among users. To validate this hypothesis, we carried out an elicitation study to investigate which shapes users would cut out for specific use cases. We invited 20 participants (seven female, average age: 29.1 years, ranging from 15 to 69 years) and asked them to imagine different scenarios in which they had to control a set of appliances using paper shapes they cut out themselves as controls. We included the four scenarios listed below with the corresponding appliances or functionality which had to be controlled.

1. **Office:** blinds (up/down), temperature, ventilation, a light bulb (on/off), an LED with adjustable brightness and color temperature, a doorbell.
2. **Movie remote control:** play/pause, fast-forward, rewind, volume, back button, ok-button, directions for selection cursor.
3. **Drone:** 3D movement (translation in each direction), rotation (horizontal rotation), taking a picture.
4. **Car:** air temperature, driver's seat temperature, driving modes (eco, comfort, sport), driver's seat adjustment (backrest tilt, seat pitch, seat height).

We provided the participants with sheets of paper in different colors and a pair of scissors, and let them cut out the shapes and groups they believed were best suited to provide the specific control functionality.

In contrast to our assumption, the shapes and assembled groups the participants created varied a lot. While some interfaces were implemented similar to our expectations (and to common everyday interfaces), many of them would not be understandable for another person without further explanation (cf. Figure 3). Often, the interfaces were relatively complex, because a single shape would be assigned a range of different functions. We furthermore did not find a consistent use of differently colored paper.

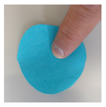
What we did find are common abstractions of interaction elements (or interaction patterns). The interaction elements may not have the same appearance (i.e., shape), but the same behavior. For example, many participants use buttons which could be tapped and which all related to the same underlying behavior despite being different in appearance. Hence, we decided to not implement an automatic recognition of shapes together with an automatic assignment of the interaction behavior, but to implement a system which allows the user to create an arbitrary shape and assign the intended behavior him-/herself from this set of identified common abstractions. The fact that all participants thought that their personal interface made sense with respect to the given task emphasizes the strength of the underlying idea, i.e., that the interfaces can be fully personalized to each individual's preferences.

Touchets

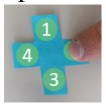
From the elicitation study, we inferred a set of 25 abstract interactions, which we refer to as *touchets*. A *touchet* represents a behavior a certain shape or a group of shapes should exhibit, e.g. a "finger slider" should allow a piece of paper to become a slider manipulated by the movement of the finger on top of it – note that this concept is *independent* of what the shape actually looks like. We found five categories of touchets, the two largest ones being buttons and sliders. In the following, we will shortly describe each identified touchet. In our implementation, touchets propagate their manipulation by returning events, which we also explain here. For some of the touchets we added a picture displaying some elements that actually occurred in the study with visualizations showing how a shape may be moved or touched, however, these are merely examples of shapes or groups that could be assigned the corresponding touchet.

Button Touchets

1. **Button (B):** A simple button which returns a "touched" event when the user's finger touches it.



2. **Hold Button (HB):** A button which can distinguish between a short and a long press. It also returns an event with the duration of being pressed.
3. **Positional Button (PB):** A positional button may contain several clickable zones, which are defined by the user.

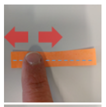


4. **Positional Hold Button (PHB):** The same as a positional button, but with hold buttons instead of buttons.

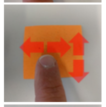
Finger Sliders

Finger sliders react to a user moving his/her finger above them, i.e. they track the position of the finger relative to the slider. They return this position as a relative value whenever the finger is moved. The finger is always kept close to the slider.

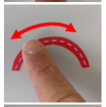
5. **Finger Slider (FS):** A 1D finger slider with a straight axis.



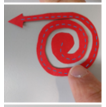
6. **2D Finger Slider (2FS):** A 2D finger slider that reacts to the movement of the finger in two directions.



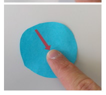
7. **Curved Finger Slider (CFS):** A 1D finger slider, however, with a curved axis.



8. **Special Finger Slider (SFS):** A finger slider with an arbitrary shape.



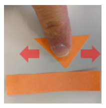
9. **Swipe Button (SB):** A finger slider that returns the direction of the swipe.



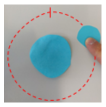
Shape Sliders

Shape sliders are similar to finger sliders with the difference that the value is not set by the finger directly, but by moving an indicator shape relative to a reference shape.

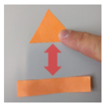
10. **Slider (S):** A 1D slider with an indicator shape and a reference shape. It returns the relative position of the indicator to the main axis of the reference shape, which is defined by the user by identifying the two endpoints of the axis.



11. **Circular Slider (CS):** For circular sliders, the indicator shape can be rotated around the reference shape. The touchet returns the current rotation angle.

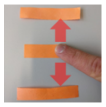


12. **Unbounded Slider (US):** A 1D slider without an upper bound. The returned value is the distance of the indicator shape to the reference.



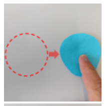
13. **2D Unbounded Slider (2US):** An unbounded slider that returns the distances from the reference shape on two orthogonal axes.

14. **Trackbar (T):** A trackbar consists of three shapes, two representing the lower and upper bounds, and the third being the indicator. It returns the relative position w.r.t. the bounds.

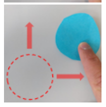


15. **2D Trackbar (2T):** A trackbar where the indicator can be moved on a 2D area bounded horizontally and vertically by four reference shapes. It returns the corresponding relative position for both axes.

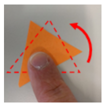
16. **Initial Position Offset Slider (IS):** A single shape that returns the current distance from the position it was first touched along a single axis.



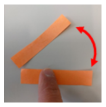
17. **2D Initial Position Offset Slider (2IS):** The same as above only returning the distance from the initial position on two orthogonal axes.



18. **Rotation (R):** A single shape that reports the current rotation angle relative to the position it was in at the beginning of being touched and relative to its initial position on the table.



19. **Angle (A):** Returns the current angle between two shapes.

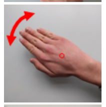


Hand Touchets

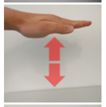
These interactions all concern the movement of the user's hand directly, and independent of the paper shapes. Hence, they do not fit into our concept of interacting with paper shape and were also relatively rare. However, we list them here for completeness.

20. **Hand Horizontal Position (HP):** Returns the 2D position of the hand in the plane parallel to the interaction surface.

21. **Hand Rotation (HR):** Returns the rotation of the hand in the plane parallel to the surface around the center of the hand.



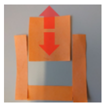
22. **Hand Height (HH):** Returns the height of the hand from the surface.



23. **Hand Tilt (HT):** Returns the two angles of a virtual plane represented by the hand relative to the surface.

Special touchets

24. **Existence (E):** The existence touchet consists of a single shape which returns a boolean value referring to whether it is currently placed on the surface or not.
25. **Containment (C):** Consisting of two shapes, it returns a boolean value referring to whether the smaller shape is currently contained within the larger one or not.



Touchets provide a powerful abstraction from the actual shapes and thereby enable users to personalize their TUIs. In addition, it is possible to assign multiple touchets to the same shape or group, which will then react to more interactions. Hence, our touchets approach allows users to model and create complex interfaces. Besides, this approach has the benefit that we do not have to classify the shapes, but only track each shape and group and its touchet assignment.

Frequency of Touchet Occurrence

The touchets presented above cover the complete set of interaction abstractions observed in the study. When using our approach, one has to assign the touchet functionality to the cut-out paper shapes. Providing the user with a large set of touchets might lead to confusion and make it difficult to choose the appropriate touchet. Hence, we counted the frequency of occurrence of touchet instances in the study in order to select the most popular touchets for our prototype system. In total we found 702 touchet instances. Table 1 shows the frequency of each touchet. Only nine touchets account for over 90% of occurrences. Based on the frequencies, we decided to include all touchets with a count of at least five in our implementation. However, we excluded the "hand height" (HH), as it does not fit into our concept of paper interaction. Another touchet we omit is "existence" (E), as shapes may not have a unique appearance and it hence is difficult to track a shape visually which is taken away from the surface and reused later. The 13 selected touchets cover over 92% of occurrences; we hence provide a nearly complete set of interaction elements without overwhelming the user.

TAILORED CONTROLS

Method Overview

We first describe the typical process of using user-defined paper snippets for controlling appliances.[3] After the user has cut out and arranged his/her shapes, he/she has to assign touchets to the shapes to add functionality, i.e. responsive behavior, to them. This is done by placing the finger in the corner (shown as a blue square in Figure 1b), which opens a menu to select a touchet. For some touchets, the user may have to select several shapes, e.g. for a slider, an indicator and a referential element are required. Similarly, one can assign pre-defined actions to the shapes by opening another menu using the green square. Actions are tags which can be used by the attached application to distinguish which shape was touched (as there might be several buttons for example). From now on the touchet instance emits the corresponding event carrying the relevant information about its state and the action tag whenever it is manipulated by the user. These events then can be used in any connected application, as explained below. Our implementation reacts to all changes immediately, e.g. movements of the reference shapes of a trackbar touchet instance change the corresponding bounds, which directly also changes the trackbar's value.

From the touchets we selected for our prototype, we can derive two main requirements: tracking the shapes and groups which have touchets assigned to them, and tracking the 3D-position of the hand, especially of the fingers. Here, we restrict the finger tracking to a single stretched out finger. A way to fulfil these requirements using only a single hardware sensor is to use an RGBD camera mounted above the interaction surface (cf. Figure 1a). The depth information is particularly important to detect when the finger touches the surface or a shape. An overview of our processing pipeline is depicted in Figure 4. On the depth image, we perform a color-independent hand segmentation based on depth-thresholding. In the hand segment, we find a first estimate of a stretched out finger. We then track the fingertip using an optical flow algorithm, which allows us to obtain the changes in position of key points on the fingertip between each pair of consecutive frames. Knowing the 2D-position of the fingertip, we can obtain the finger's height from the depth image as well and thereby deduce whether the user is currently touching the surface or a shape on the surface. Based on the RGB image, we track all the shapes in the field of view of the camera. As soon as a shape is placed on the surface, it is assigned a unique identifier. Each shape is tracked continuously by a shape tracking algorithm we designed for this purpose, which returns an estimate of the translation and rotation a shape has undergone between two frames. This is a challenging problem, as a significant part of a shape may be occluded by the user's hand. Finally, we combine the information about the finger and shape positions and thereby can detect whether the user is touching a shape, swiping over it, or moving it. If any of the actions performed matches the touchet specification assigned to the touched shape, an event is forwarded to a server, from where this information can be relayed to other applications. For example, a "button" touchet will not react to rotation, but a "rotation" touchet will.

For configuring the propagation of events, we use Node-RED[4], a programming tool which allows to connect and program hardware devices, APIs, and online services in a GUI editor. Using Node-RED, users can completely customize their own application. For frequent use, the Node-RED parts can certainly also be pre-implemented, so that the user can simply connect the touchet events to these pre-configured actions.

Setup

For the RGB-D camera, we use an Intel RealSense D435, which we mount 40 cm above a table as shown in Figure 1a. The camera monitors a rectangular area of about 43 cm by 32 cm. The camera stream is processed by a desktop computer with an Intel i7-4790K 4x4 GHz CPU 16 GB of RAM. All the processing runs on the CPU. The source code of the system can be downloaded from https://github.com/vincentbecker/TailoredControls.

Image Preprocessing

Before detecting the hands and shapes, we first preprocess the color and depth images from the camera. The RGB image is merely aligned with the depth image so that the coordinate systems correspond to each other. The depth image requires further processing due to noise: We first apply a temporal filter to reduce the noise. Additionally, to remove spurious errors in the depth image where the values are clearly too high, we employ a hole filling algorithm, which takes the neighboring pixels of such a hole into account. Both algorithms are available in the RealSense library. Another problem is that the

[3]The use of our prototype is furthermore showcased in a video available at https://youtu.be/p_wS6BhTpQQ.

[4]https://nodered.org/

Touchet	B	HB	R	PB	S	PHB	E	US	T	FS	2T	2FS	SB	CS	HH	IS	A	2IS	HT	2US	C	CFS	SFS	HP	HR
Freq.	233	95	73	66	46	38	33	30	24	14	8	7	7	5	5	4	3	2	2	2	1	1	1	1	1

Table 1. The frequency of occurrence of each touchet in the elicitation study sorted according to the frequency.

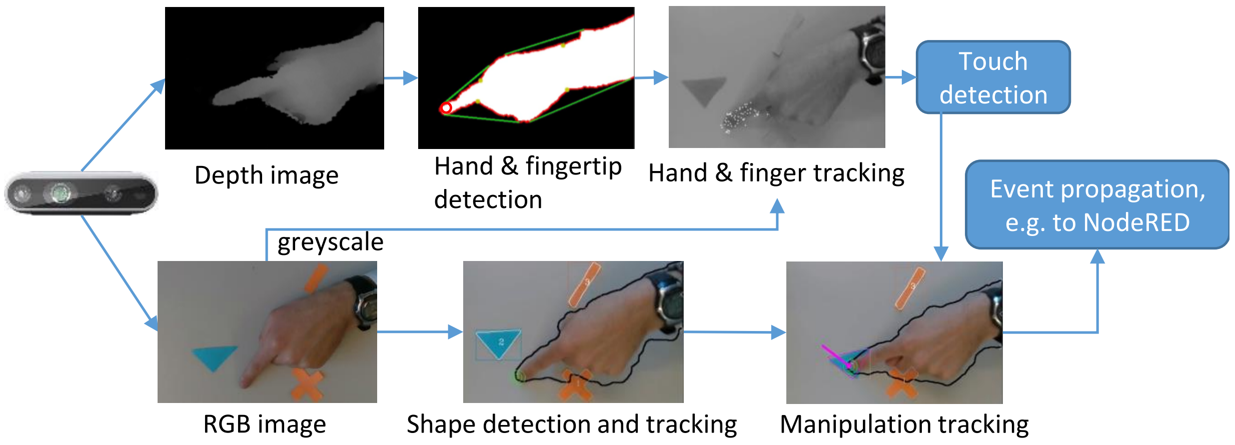

Depth image
Hand & fingertip detection
Hand & finger tracking
Touch detection
Event propagation, e.g. to NodeRED
greyscale
RGB image
Shape detection and tracking
Manipulation tracking

Figure 4. Our processing pipeline mainly consisting of finger and shape tracking based on the images from the depth camera.

camera plane is usually not perfectly parallel to the interaction surface, resulting in some areas of the surface being perceived as further away, which interferes with the hand and finger detection on the surface. In order to prevent this from happening, we take a snapshot of the depth values when starting the system and henceforth subtract these from the currently recorded image. Consequently, all points on the surface have the same depth, which facilitates further processing.

Finger Detection and Tracking

One main requirement is the ability to track the hand and fingers in order to detect when the user is touching and manipulating the shapes. In an optimal case, we would know the pose of the complete hand. For this purpose, we tested several hand pose estimation algorithms such as [27], but none showed a satisfying accuracy although having high computational requirements. Hence, we restricted the way users may manipulate shapes to the most common one, touching them with a single fingertip, and designed an efficient tracking algorithm for this case. Furthermore, we require the contours of the hand to know when shapes are occluded. We assume that there are no objects in the field of view of the camera apart from the paper shapes and the hand. Hence, we can distinguish the hand by applying thresholding to the depth image, as the shapes are flat. We threshold the image at a level of 1 mm to obtain a binary mask of the hand and apply a contour finding algorithm. From all the resulting contours, we take the largest one, as we assume the hand and arm to be the only larger 3D object in the scene. Thereby, we obtain a rough estimate of the hand and arm position. To find the fingers, we search for defects in the convex hull around the hand. The fingertips are the corners of the convex polygon. Unfortunately, even after preprocessing the depth image, there is still a significant amount of noise, which deteriorates the accuracy of the fingertip position. Hence, we do not only search for the fingertip in the 1 mm-thresholded binary mask, but also in higher masks.

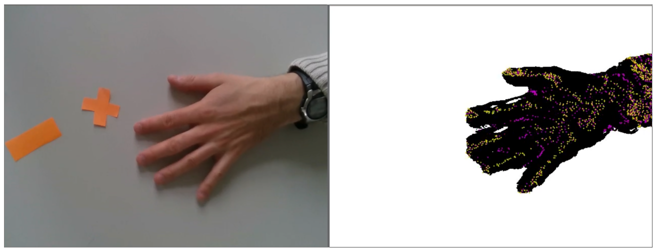


Figure 5. The resulting corner points of the finger finding algorithm applied on different depth levels.

We thereby obtain several corner points along the top of the fingers as illustrated in Figure 5. Afterwards, we apply a closing morphology on the corner points in order to merge nearby points. Fingertips then appear as the ends of point clusters.

We assume that the relevant fingertip for touching is the point furthest away from the arm. We can find the arm as the part of the body which intersects the image frame.

The detection step described above by itself merely provides the current location of the fingertip, but not its movement over time, which is relevant to recognize and track interactions with the shapes. In order to actually track the finger, we apply the Lucas-Kanade optical flow algorithm [25] to the detected finger region in the greyscaled RGB stream. The algorithm is able to find salient keypoints in the region and track their position across multiple frames. We apply it as soon as a finger is detected in the images. As multiple points are tracked, we can not only calculate the translation of the finger across frames, but also its rotation by matching the two corresponding point sets from two consecutive frames. This is done by computing the best-fitting rigid transformation [33]. With the 2D location of the fingertip we can directly extract its height above the surface from the depth map and deduce whether the user is touching a shape or not.

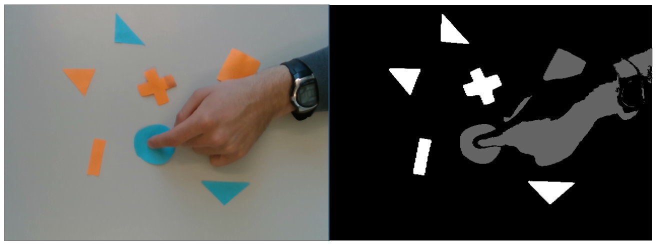


Figure 6. The binary shape masks with the detected hand region.

Evaluation of Fingertip Detection

In order to evaluate the accuracy of the fingertip detection, we recorded several sequences during which a participant moved their fingers below the camera. We then manually annotated the position of the fingertip in order to compare it to our estimation. In total, we annotated 1137 frames from six participants (note that our detection method works independent of skin color). On average, the error was 15 pixels which corresponds to approximately 1 cm, however, this average was influenced by few large errors which occur when the hand is entering the field of view of the camera. The median error was 8.6 pixels, which is accurate enough given shape sizes of a few centimeters.

Shape Tracking

Detecting and tracking the shapes is performed purely on the RGB image, as the depth image does not contain any information about the flat paper shapes. We assume the interaction surface to be unicolored with low saturation, e.g. a standard table. To detect the shapes we convert the RGB image to the HSV color space and threshold the saturation channel (assuming the shapes have a different color than the table). In the resulting binary mask, we find the contours. From these shape candidates, we exclude those in the area covered by the hand contour, as illustrated in Figure 6. Whenever a shape is occluded by the hand or arm, we try to match reappearing shapes by their position to recover the shape identity.

Tracking Manipulations of Shapes

Knowing the position of the fingertip and the shapes, we can detect when a user touches a shape. To fulfil the necessary requirements to recognize all possible touchet interactions, we have to detect for how long a shape is touched and whether it is swiped upon, as well as the translation and rotation of the shape in case of a movement. A swipe can be distinguished from moving the shape by the fact that only the finger, but not the shape itself moves. For movements of shapes, we track the change in translation and rotation from a shape's appearance for every frame since the shape was touched as this might induce a parameter value change which we want to be able to follow continuously. Since the shape is occluded by the finger (and the finger position might also shift during the movement), applying a direct transformation calculation through a best-fit algorithm as for the finger tracking is not possible. Instead, we take a snapshot of the shape in the moment it is first touched and calculate the translation between the currently detected shape and the snapshot. For the rotation, we iteratively test for the best fit employing an efficient ternary search. The search uses the rotation estimate of the finger as a starting point, however, as the finger position and rotation might change independently of the shape during the movement, this can only be used as a first indication. The measure for a fit is the area of intersection between the detected shape and the snapshot. Note that we can also track the rotation of circular shapes, as a circle is partly occluded by the finger during interaction resulting in a perceived shape that has a "hole" in the location of the finger, which allows the tracking of the rotation.

Evaluation of Shape Tracking

To evaluate the shape tracking accuracy, we recorded several sequences when moving a squared shape at different speeds also including rotations. For 1,529 frames, we manually annotated the positions of the four corners of the square and compare them to the estimate calculated by our tracking algorithm. We calculated the average displacement of a corner point. The mean error over all the frames was 6 pixels (0.4 cm), which is accurate enough for our purpose.

USER STUDY

To evaluate the entire process of creating a TUI with our prototype, we invited six new participants to a second study (two females, 18 to 28 years old). As we planned to let the participants build a complete application from scratch, including the connection to and implementation of the Node-RED flow, we only invited computer science students with programming knowledge. We asked them to implement interfaces and the Node-RED flows for the three following applications:

1. Driving a little toy rover as shown in Figure 1c, which can drive forwards, backwards, and rotate to the left and right.

2. Creating a music playback control based on an `MPD`[5] wrapper we provided. The interface should control volume, play and pause, and switching to the next or previous song.

3. Creating an interface for controlling a slide show. In order to do so, we provided a wrapper in Node-RED to access the `xdotool`[6], which allows to emulate keyboard presses. The interface should allow to start and stop a slide show, as well as to move to the next and previous slides.

The order of implementing each of these applications was different for each participant. Three days before the study, we gave the participants basic information on the required components and asked them to get familiar with `MPD` and `xdotool`. During the study, we provided the participants with an overview of the available touchets besides the task descriptions. We set no time limit and let the participants try any interface they wanted. After the study, we asked them several questions concerning our prototype and let them fill in a User Experience Questionnaire (UEQ) [23][7] score sheet in order to obtain a standardized measure of the user experience. The UEQ includes 26 items on a seven-point score ranging from -3 to 3. The output of the UEQ analysis are scores in the following six dimensions: attractiveness, perspicuity, efficiency, dependability, stimulation, and novelty.

[5]https://www.musicpd.org/
[6]https://www.semicomplete.com/projects/xdotool/
[7]http://www.ueq-online.org/

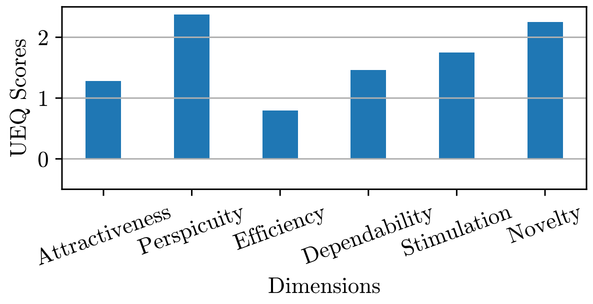


Figure 7. The average results of the UEQ.

All participants successfully completed the tasks. On average, a participant spent 2 hours and 18 minutes on all three tasks. Note that this includes all necessary parts, i.e., the design of the paper interface and the assignment of the touchets, as well as the Node-RED implementation. As participants spent most of the time on the Node-RED implementation, this latter part could also be pre-implemented for certain real-world applications, speeding up the use of our method significantly, e.g. for most of our tasks from a magnitude of half an hour to an hour down to a few minutes. As we let participants try out interfaces as they wished, they did not optimize for time. One participant enjoyed using the system and optimizing his interface so much, that he stayed for five hours. On the other hand, another participant created an entire interface in 11 minutes. Furthermore, the average task time significantly decreased from 70 minutes for the first task to 32 minutes for the second.

The resulting scores for the six dimensions of the UEQ are shown in Figure 7. For all dimensions, the scores are positive, indicating that participants generally enjoyed using our prototype. Most noticeable are the scores for perspicuity, stimulation, and novelty, which imply what it was easy to learn and to understand, that the participants were motivated and excited to use the approach, and that they believed it to be a creative new method for designing interfaces. The worst characteristic is efficiency, because our prototype cannot track fast movements of the finger due to significant motion blur in the depth stream, i.e., sometimes participants had to repeat their interactions. Moreover, several participants mentioned the relatively long time it takes to configure the necessary Node-RED flows, especially for inexperienced users. This most likely also influenced the attractiveness score. However, the Node-RED part is only used to forward the interaction events and could be replaced by a different mechanism, or the flows could be pre-implemented for common devices to be controlled.

As part of a survey that followed the study, participants confirmed that our method was easy to understand and that most of them enjoyed using the prototype, especially to try out different interfaces. They also suggested different scenarios in which the *Tailored Controls* could be used, such as for rapid prototyping, for interfaces in changing environments, or for controlling home devices.

DISCUSSION

Our technical evaluation and the results from the presented user study show that we created a functioning prototype with appropriate performance (e.g., regarding shape detection) which was positively received by participants. This is in spite of the fact that our prototype has several limitations, which are discussed below.

Despite these limitations, we were able to show that our suggested *touchet* taxonomy that we derived from the results of our elicitation study can indeed be used for the implementation of personalized TUIs. With our approach, we significantly progress beyond the state of the art compared to other current systems that enable dynamic TUIs: in addition to not requiring fiducial markers, our system enables more natural interactions due to the explicit finger tracking functionality, and it provides users with a wide range of interaction abstractions to select from when creating TUIs.

We built our prototype with the intention to demonstrate the feasibility of our approach and evaluate it in the user study. However, we believe there are a multitude of different application scenarios where a system such as ours could prove useful as already mentioned in the introduction.

First, our approach can be used for the rapid prototyping of functional TUIs – paper prototyping is a common technique for interface design, however, such interfaces are usually not functional and the effects of using them have to be imagined by prototypers. With our system, it is possible to immediately test and experience functional interfaces, which we believe would spur creativity. This is backed by our user study, where participants were very motivated to try out different interaction elements, in particular because our system does not restrict users to employ predefined generic paper shapes. Consequently, *Tailored Controls* can bridge the gap between the "ideation" and "implementation" stages in the design thinking process [4], enabling the designer to be more creative and faster by providing direct feedback about what a functionally implemented interface reacts like.

Second, our system can be used in the context of dynamically changing workspaces, e.g., to provide ad-hoc interfaces to devices that are used only infrequently – we imagine this to be of use especially in crowded workspaces (e.g., in industrial workshops), where it allows users to carry interfaces with them and deploy them tactically, rather than attaching TUIs to the machines they intend to control. To this end, we want to emphasize that the presented system and our prototype is not limited to paper alone, but can be used with other (sufficiently flat) materials.

Third, our approach enables a very high degree of personalization in terms of the TUIs that can be created by users. It thus enables individualized user interfaces, for instance in the context of accessibility constraints: with our approach, it is possible to create interfaces that are optimally suited for individual disabled persons, given their specific type of disability and context constraints.

Limitations and Future Work

Our approach has several limitations, both from a theoretical and a practical perspective, which we intend to approach in the future. Our current implementation of the finger tracking algorithm only allows the use of a single finger for manipulations and there is no multi-touch support. Furthermore, it is difficult to track very fast movements of the fingertip, as fast movements create sequences of blurry images in the depth stream. Finally, the interaction space is currently limited to a relatively small flat area. We envision that these problems can be solved in the future by better hand pose estimation algorithms on the one hand and by improved hardware on the other. Our approach could directly be adapted to such improvements. A further issue is that our system requires an RGBD camera to be available to monitor the scene, i.e. the approach is most applicable for fixed workspaces. With future improvements, we however expect the area which may be covered to grow.

On a philosophical level, the use of flat paper shapes may no be considered tangible in a strict sense. However, we view tangibility as the physical, material presence in the elements used, i.e. that they can be touched, moved, or reconfigured without interacting with a digital proxy device that displays the user interface. We believe already this property of the paper shapes is beneficial for the interaction process, as it enables the interactions to take place in the real world, a space the user is naturally accustomed to. In terms of taking advantage of the full properties of paper, one could experiment with other forms paper interfaces may take, such as folding or crushing the paper, which would further enhance the tangible experience. Nevertheless, none of our participants in the elicitation produced any of these interfaces, neither did anyone mention that this could be a possibility, and we did not restrict them only to cut out paper shapes. Furthermore, one could include different types of paper or materials, such as fabric, which is an interesting aspect of future work. We partly already included this in the study by providing differently colored paper, however, this property was rarely incorporated in the interface designs by the participants. Further, we do believe our approach could naturally be used with other (flat) materials as well, such as fabric or even play dough, which would enable an even easier reconfiguration of interfaces. This could also solve practical environmental issues paper shapes might have, for example they are easily blown away by wind or a draft.

CONCLUSION

We presented the *Tailored Controls* approach that enables the simple creation of personalized TUIs that are made from plain paper but can be connected to virtually any application. In a first user study with 20 participants, we found 25 interaction abstractions implicitly employed by the participants. We built a functional prototype and implemented 13 of the most common of these abstractions. Our system tracks a user's fingertip as well as the paper shapes and is thereby able to detect interaction events, including tapping, rotating, swiping, sliding, and many more. Our prototype allows the simple creation of user interfaces for diverse applications, as we proved in a second user study, and demonstrates the feasibility of our initial concept. By using a set of abstractions that covers a wide range of potential interaction elements, we are able to give users the freedom to create the paper shapes they want and afterwards add corresponding functionality themselves, instead of having to choose from a predefined set of generic elements. The only hardware required is an RGBD camera that is mounted above the interaction surface.

Tailored Controls demonstrates the feasibility of inexpensive reconfigurable TUIs, and one main contribution is to enable further research in the direction of customizable personalized TUIs. Immediate applications of our approach include the rapid prototyping of functional TUIs, the creation of tactical interfaces for sporadic interactions, and the generation of individualized TUIs that are optimal for concrete individual contexts, for instance for the benefit of disabled persons. In addition, triggered by the expected proliferation of RGBD cameras, we expect our approach to enable novel interaction methods in domestic and public environments, as well as in mobile scenarios.

ACKNOWLEDGEMENTS

We would like to thank all study participants for their time and effort and the anonymous reviewers for their constructive feedback.

REFERENCES

[1] Daniel Avrahami, Jacob O. Wobbrock, and Shahram Izadi. 2011. Portico: Tangible Interaction on and Around a Tablet. In *Proceedings of the 24th Annual ACM Symposium on User Interface Software and Technology (UIST '11)*. ACM, New York, NY, USA, 347–356. DOI: http://dx.doi.org/10.1145/2047196.2047241

[2] Vincent Becker, Felix Rauchenstein, and Gábor Sörös. 2019. Investigating Universal Appliance Control through Wearable Augmented Reality. In *Proceedings of the 10th Augmented Human International Conference (AH '19)*. 9. DOI:http://dx.doi.org/10.1145/3311823.3311853

[3] Florian Block, Michael Haller, Hans Gellersen, Carl Gutwin, and Mark Billinghurst. 2008. VoodooSketch: Extending Interactive Surfaces with Adaptable Interface Palettes. In *Proceedings of the 2nd International Conference on Tangible and Embedded Interaction (TEI '08)*. ACM, New York, NY, USA, 55–58. DOI: http://dx.doi.org/10.1145/1347390.1347404

[4] Tim Brown. 2008. Design Thinking. *Harvard business review* 86 (07 2008), 84–92, 141.

[5] Varun Perumal C and Daniel Wigdor. 2015. Printem: Instant Printed Circuit Boards with Standard Office Printers & Inks. In *Proceedings of the 28th Annual ACM Symposium on User Interface Software & Technology (UIST '15)*. ACM, New York, NY, USA, 243–251. DOI: http://dx.doi.org/10.1145/2807442.2807511

[6] K. Cheng, R. Liang, B. Chen, R. Laing, and S. Kuo. 2010. iCon: Utilizing Everyday Objects As Additional, Auxiliary and Instant Tabletop Controllers. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '10)*. 1155–1164. DOI:http://dx.doi.org/10.1145/1753326.1753499

[7] Marcelo Coelho and Jamie Zigelbaum. 2011. Shape-changing Interfaces. *Personal Ubiquitous Comput.* 15, 2 (Feb. 2011), 161–173. DOI: http://dx.doi.org/10.1007/s00779-010-0311-y

[8] Christian Corsten, Ignacio Avellino, Max Möllers, and Jan Borchers. 2013. Instant User Interfaces: Repurposing Everyday Objects As Input Devices. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces (ITS '13)*. ACM, New York, NY, USA, 71–80. DOI: http://dx.doi.org/10.1145/2512349.2512799

[9] Florian Echtler, Manuel Huber, and Gudrun Klinker. 2008. Shadow Tracking on Multi-touch Tables. In *Proceedings of the Working Conference on Advanced Visual Interfaces (AVI '08)*. ACM, New York, NY, USA, 388–391. DOI: http://dx.doi.org/10.1145/1385569.1385640

[10] George W. Fitzmaurice, Hiroshi Ishii, and William A. S. Buxton. 1995. Bricks: Laying the Foundations for Graspable User Interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '95)*. ACM Press/Addison-Wesley Publishing Co., New York, NY, USA, 442–449. DOI: http://dx.doi.org/10.1145/223904.223964

[11] Sean Follmer, Daniel Leithinger, Alex Olwal, Akimitsu Hogge, and Hiroshi Ishii. 2013. inFORM: Dynamic Physical Affordances and Constraints Through Shape and Object Actuation. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 417–426. DOI: http://dx.doi.org/10.1145/2501988.2502032

[12] Kaori Fujinami, Mami Kosaka, and Bipin Indurkhya. 2018. Painting an Apple with an Apple: A Tangible Tabletop Interface for Painting with Physical Objects. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 162 (Dec. 2018), 22 pages. DOI: http://dx.doi.org/10.1145/3287040

[13] Markus Funk, Oliver Korn, and Albrecht Schmidt. 2014. An Augmented Workplace for Enabling User-defined Tangibles. In *CHI '14 Extended Abstracts on Human Factors in Computing Systems (CHI EA '14)*. ACM, New York, NY, USA, 1285–1290. DOI: http://dx.doi.org/10.1145/2559206.2581142

[14] Anneli Hershman, Juliana Nazare, Jie Qi, Martin Saveski, Deb Roy, and Mitchel Resnick. 2018. Light It Up: Using Paper Circuitry to Enhance Low-fidelity Paper Prototypes for Children. In *Proceedings of the 17th ACM Conference on Interaction Design and Children (IDC '18)*. ACM, New York, NY, USA, 365–372. DOI: http://dx.doi.org/10.1145/3202185.3202758

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

[16] Valentin Heun, Shunichi Kasahara, and Pattie Maes. 2013b. Smarter Objects: Using AR Technology to Program Physical Objects and Their Interactions. In *CHI '13 Extended Abstracts on Human Factors in Computing Systems (CHI EA '13)*. ACM, New York, NY, USA, 961–966. DOI: http://dx.doi.org/10.1145/2468356.2468528

[17] Hiroshi Ishii and Brygg Ullmer. 1997. Tangible Bits: Towards Seamless Interfaces Between People, Bits and Atoms. In *Proceedings of the ACM SIGCHI Conference on Human Factors in Computing Systems (CHI '97)*. ACM, New York, NY, USA, 234–241. DOI: http://dx.doi.org/10.1145/258549.258715

[18] Yoshihiro Kawahara, Steve Hodges, Benjamin S. Cook, Cheng Zhang, and Gregory D. Abowd. 2013. Instant Inkjet Circuits: Lab-based Inkjet Printing to Support Rapid Prototyping of UbiComp Devices. In *Proceedings of the 2013 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '13)*. ACM, New York, NY, USA, 363–372. DOI: http://dx.doi.org/10.1145/2493432.2493486

[19] Annie Kelly, R. Benjamin Shapiro, Jonathan de Halleux, and Thomas Ball. 2018. ARcadia: A Rapid Prototyping Platform for Real-time Tangible Interfaces. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 409, 8 pages. DOI: http://dx.doi.org/10.1145/3173574.3173983

[20] David Kim, Shahram Izadi, Jakub Dostal, Christoph Rhemann, Cem Keskin, Christopher Zach, Jamie Shotton, Timothy Large, Steven Bathiche, Matthias Niessner, D. Alex Butler, Sean Fanello, and Vivek Pradeep. 2014. RetroDepth: 3D Silhouette Sensing for High-precision Input on and Above Physical Surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1377–1386. DOI: http://dx.doi.org/10.1145/2556288.2557336

[21] Hyunyoung Kim. 2018. Fostering Design Process of Shape-Changing Interfaces. In *The 31st Annual ACM Symposium on User Interface Software and Technology Adjunct Proceedings (UIST '18 Adjunct)*. ACM, New York, NY, USA, 224–227. DOI: http://dx.doi.org/10.1145/3266037.3266131

[22] Hyunyoung Kim, Celine Coutrix, and Anne Roudaut. 2018. Morphees+: Studying Everyday Reconfigurable Objects for the Design and Taxonomy of Reconfigurable UIs. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 619, 14 pages. DOI: http://dx.doi.org/10.1145/3173574.3174193

[23] Bettina Laugwitz, Theo Held, and Martin Schrepp. 2008. Construction and Evaluation of a User Experience Questionnaire. In *Proceedings of the 4th Symposium of the Workgroup Human-Computer Interaction and Usability Engineering of the Austrian Computer Society on HCI and Usability for Education and Work (USAB '08)*. Springer-Verlag, Berlin, Heidelberg, 63–76. DOI: http://dx.doi.org/10.1007/978-3-540-89350-9_6

[24] Daniel Leithinger, Sean Follmer, Alex Olwal, Samuel Luescher, Akimitsu Hogge, Jinha Lee, and Hiroshi Ishii. 2013. Sublimate: State-changing Virtual and Physical Rendering to Augment Interaction with Shape Displays. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13)*. ACM, New York, NY, USA, 1441–1450. DOI: http://dx.doi.org/10.1145/2470654.2466191

[25] Bruce D. Lucas and Takeo Kanade. 1981. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of the 7th International Joint Conference on Artificial Intelligence - Volume 2 (IJCAI'81)*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 674–679.

[26] Donald A. Norman. 2002. *The Design of Everyday Things*. Basic Books, Inc., New York, NY, USA.

[27] Markus Oberweger and Vincent Lepetit. 2017. DeepPrior++: Improving Fast and Accurate 3D Hand Pose Estimation. In *International Conference on Computer Vision Workshops*.

[28] Simon Olberding, Nan-Wei Gong, John Tiab, Joseph A. Paradiso, and Jürgen Steimle. 2013. A Cuttable Multi-touch Sensor. In *Proceedings of the 26th Annual ACM Symposium on User Interface Software and Technology (UIST '13)*. ACM, New York, NY, USA, 245–254. DOI: http://dx.doi.org/10.1145/2501988.2502048

[29] Henning Pohl and Michael Rohs. 2014. Around-device Devices: My Coffee Mug is a Volume Dial. In *Proceedings of the 16th International Conference on Human-computer Interaction with Mobile Devices & Services (MobileHCI '14)*. ACM, New York, NY, USA, 81–90. DOI: http://dx.doi.org/10.1145/2628363.2628401

[30] Jie Qi and Leah Buechley. 2014. Sketching in Circuits: Designing and Building Electronics on Paper. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '14)*. ACM, New York, NY, USA, 1713–1722. DOI: http://dx.doi.org/10.1145/2556288.2557391

[31] Jie Qi, Asli Demir, and Joseph A. Paradiso. 2017. Code Collage: Tangible Programming On Paper With Circuit Stickers. In *Proceedings of the 2017 CHI Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '17)*. ACM, New York, NY, USA, 1970–1977. DOI: http://dx.doi.org/10.1145/3027063.3053084

[32] Raf Ramakers, Kashyap Todi, and Kris Luyten. 2015. PaperPulse: An Integrated Approach for Embedding Electronics in Paper Designs. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems (CHI '15)*. ACM, New York, NY, USA, 2457–2466. DOI: http://dx.doi.org/10.1145/2702123.2702487

[33] Olga Sorkine-Hornung and Michael Rabinovich. 2017. *Least-Squares Rigid Motion Using SVD*. Technical Report. ETH Zurich. https://igl.ethz.ch/projects/ARAP/svd_rot.pdf

[34] Martin Spindler, Victor Cheung, and Raimund Dachselt. 2013. Dynamic Tangible User Interface Palettes. In *Human-Computer Interaction – INTERACT 2013*, Paula Kotzé, Gary Marsden, Gitte Lindgaard, Janet Wesson, and Marco Winckler (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 159–176.

[35] Nicolas Villar, Daniel Cletheroe, Greg Saul, Christian Holz, Tim Regan, Oscar Salandin, Misha Sra, Hui-Shyong Yeo, William Field, and Haiyan Zhang. 2018. Project Zanzibar: A Portable and Flexible Tangible Interaction Platform. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 515, 13 pages. DOI: http://dx.doi.org/10.1145/3173574.3174089

[36] Simon Voelker, Kosuke Nakajima, Christian Thoresen, Yuichi Itoh, Kjell Ivar Overgaard, and Jan Borchers. 2013. PUCs: Detecting Transparent, Passive Untouched Capacitive Widgets on Unmodified Multi-touch Displays. In *Proceedings of the 2013 ACM International Conference on Interactive Tabletops and Surfaces (ITS '13)*. ACM, New York, NY, USA, 101–104. DOI: http://dx.doi.org/10.1145/2512349.2512791

[37] Mark Weiser. 1999. The Computer for the 21st Century. *SIGMOBILE Mob. Comput. Commun. Rev.* 3, 3 (July 1999), 3–11. DOI: http://dx.doi.org/10.1145/329124.329126

[38] Malte Weiss, Julie Wagner, Yvonne Jansen, Roger Jennings, Ramsin Khoshabeh, James D. Hollan, and Jan Borchers. 2009. SLAP Widgets: Bridging the Gap Between Virtual and Physical Controls on Tabletops. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '09)*. ACM, New York, NY, USA, 481–490. DOI: http://dx.doi.org/10.1145/1518701.1518779

[39] Robert Xiao, Scott Hudson, and Chris Harrison. 2016. DIRECT: Making Touch Tracking on Ordinary Surfaces Practical with Hybrid Depth-Infrared Sensing. In *Proceedings of the 2016 ACM International Conference on Interactive Surfaces and Spaces (ISS '16)*. ACM, New York, NY, USA, 85–94. DOI: http://dx.doi.org/10.1145/2992154.2992173

[40] Hui-Shyong Yeo, Ryosuke Minami, Kirill Rodriguez, George Shaker, and Aaron Quigley. 2018. Exploring Tangible Interactions with Radar Sensing. *Proc. ACM Interact. Mob. Wearable Ubiquitous Technol.* 2, 4, Article 200 (Dec. 2018), 25 pages. DOI: **http://dx.doi.org/10.1145/3287078**

[41] Yang Zhang and Chris Harrison. 2018. Pulp Nonfiction: Low-Cost Touch Tracking for Paper. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems (CHI '18)*. ACM, New York, NY, USA, Article 117, 11 pages. DOI: **http://dx.doi.org/10.1145/3173574.3173691**

[42] Oren Zuckerman and Ayelet Gal-Oz. 2013. To TUI or not to TUI: Evaluating performance and preference in tangible vs. graphical user interfaces. *International Journal of Human-Computer Studies* 71, 7 (2013), 803–820. DOI: **http://dx.doi.org/10.1016/j.ijhcs.2013.04.003**