

# SmartSleeve: Real-time Sensing of Surface and Deformation Gestures on Flexible, Interactive Textiles, using a Hybrid Gesture Detection Pipeline

Patrick Parzer<sup>1</sup>, Adwait Sharma<sup>1</sup>, Anita Vogl<sup>1</sup>, Jürgen Steimle<sup>2</sup>, Alex Olwal<sup>3</sup>, Michael Haller<sup>1</sup>

<sup>1</sup>Media Interaction Lab, University of Applied Sciences Upper Austria

<sup>2</sup>Saarland University, Saarland Informatics Campus, Saarbrücken, Germany

<sup>3</sup>Google, Inc., Mountain View, California, United States

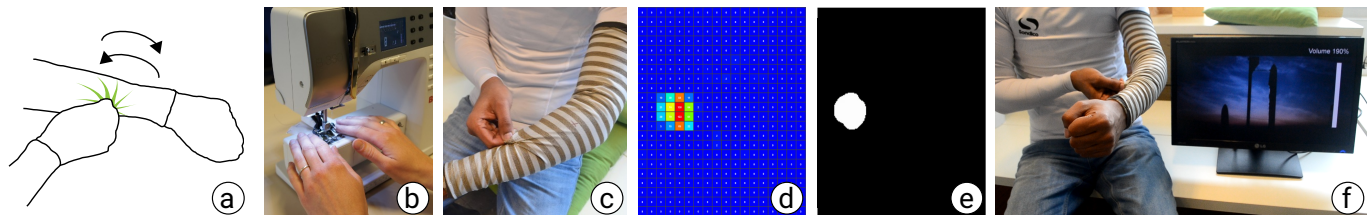


Figure 1: SmartSleeve is a wearable textile that can detect 2D surface and 2.5D deformation gestures, like twist (a). We use an unobtrusive and robust sewn-based connection (b), which withstands high deformation gestures (c). The force distribution values of the gestures (d) are further processed for real-time classification with a hybrid gesture detection algorithm (e) to control a media player (f), for example.

## ABSTRACT

Over the last decades, there have been numerous efforts in wearable computing research to enable interactive textiles. Most work focus, however, on integrating sensors for planar touch gestures, and thus do not fully take advantage of the flexible, deformable and tangible material properties of textile. In this work, we introduce *SmartSleeve*, a deformable textile sensor, which can sense both *surface* and *deformation gestures* in real-time. It expands the gesture vocabulary with a range of expressive interaction techniques, and we explore new opportunities using advanced deformation gestures, such as, *Twirl*, *Twist*, *Fold*, *Push* and *Stretch*. We describe our sensor design, hardware implementation and its novel non-rigid connector architecture. We provide a detailed description of our hybrid gesture detection pipeline that uses learning-based algorithms and heuristics to enable real-time gesture detection and tracking. Its modular architecture allows us to derive new gestures through the combination with continuous properties like *pressure*, *location*, and *direction*. Finally, we report on the promising results from our evaluations which demonstrate real-time classification.

## Author Keywords

Smart Textile, Deformation Gestures, Surface Gestures

## ACM Classification Keywords

H.5.2.: [User Interfaces]: Input devices and strategies.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from [permissions@acm.org](mailto:permissions@acm.org).  
UIST 2017, October 22-25, 2017, Quebec City, QC, Canada  
© 2017 Association for Computing Machinery.  
ACM ISBN 978-1-4503-4981-9/17/10 ...\$15.00  
<https://doi.org/10.1145/3126594.3126652>

## INTRODUCTION

Computing technologies have become mobile and ubiquitous and, as predicted by Mark Weiser [66], weave themselves into the fabric of everyday life. While many objects and surfaces have been augmented with interactive capabilities, including smart phones, tabletops, walls, or entire floors, making clothing interactive is still an ongoing challenge. Over the last few decades, a lot of research in wearable computing has focused on integrating sensors into textiles [6, 40, 7, 50]. Most of the existing work in the design space of interactive clothing focuses on *surface gestures*, planar interactions, such as *touch* and *pressure* [54, 47, 37, 13]. *Basic deformation* has also been shown with *stretch* [4, 61] and *rolling* for 1D input [29, 17].

In this work, we introduce *SmartSleeve*, a deformable textile sensor, which can sense both touch and deformations in real-time. Our hybrid gesture detection framework uses a learning-based algorithm and heuristics to greatly expand the possible interactions for flexible, pressure-sensitive textile sensors as its unified pipeline senses both 2D *surface gestures* and more complex 2.5D *deformation gestures*. Furthermore, its modular architecture allows us to also derive new gestures through the combination with continuous properties like *pressure*, *location*, and *direction*. Thus, our approach allows us to go beyond the touchscreen emulation and basic deformations found in previous work. We particularly emphasize the opportunity to enable both isotonic and isometric/elastic input, as well as, state-changing interaction with integrated passive haptic feedback. This enables us to support a wide range of *deformation gestures*, such as *Bend* [21], *Twist* [65, 31, 62], *Pinch* [65, 29, 31, 62, 17], *Shake* [31], *Stretch* [31], and *Fold* [16, 63, 31]. We further explore the usage of multi-modal input modalities by combining pressure with deformation.

Summarizing, the main contributions of this paper are:

- A hybrid gesture detection pipeline that uses learning-based algorithms and heuristics to enable real-time gesture detection and tracking for flexible textile sensors.

- Two user studies that show the feasibility and accuracy of our gesture detection algorithm.
- A flexible, resistive-pressure textile sensor, with a novel non-rigid connector architecture. We propose a sewn-based connection between the textile sensor and the electronics.
- A set of novel interaction techniques, which arise from the combination of *surface gestures*, *deformation gestures*, and continuous parameters (pressure, location, and direction).

## RELATED WORK

### Limited input gestures on textiles

Several empirical studies investigated how skin or textiles can serve as gestural input surfaces. *More than touch* [65] reported an elicitation study in a non-technological environment that shows how a set of gestures, including touch, grab, pull, press, scratch, shear, squeeze, and twist, are preferably performed on the forearm or the hand. Lee et al. [31] explores deformation-based user gestures by using various materials like plastic, paper and elastic cloth. Bending, folding, rolling, crumpling and stretching were suggested as possible deformations. Troiano et al. [62] investigated how depth and elasticity of a display can be used to simulate deformation and provided a set of gestures including grabbing, pulling, pushing, twisting, pinching or moving.

While researchers presented diverse gesture sets appropriate for textile input spaces, several solutions focused on specific input gestures on textiles. Touch sensitive fabrics were used for a range of gestural input on trousers [23], pockets [47] or sleeves [54]. Stitch-based solutions detect bends and folds [29, 14] by sensing interconnections between seams. Similarly, grabbing a fold at a specific angle is detected by using embroidered pads [17]. GestureSleeve [54] has an interesting approach for extending the input space of a smart watch to the sleeve, but only supports tap and stroke gestures. We choose to focus on a rich set of 2D touch and 2.5D deformation-based gestures on a single sleeve, to combine recent advances in empirical studies with current technological possibilities. We combine directional and pressure sensing that can deliver a wide range of novel interactions, supporting additional degrees of freedom with expressiveness.

### Facilitating deformation-based input with 2.5D

Pressure-sensitive input has been a topic of interest in the HCI community for several years now, with research efforts ranging from explorations of pressure as alternative input metaphor [24, 33, 58] to the development of pressure-sensitive input devices [8, 33, 35]. To date, pressure has been used for a variety of applications such as zooming, scrolling, text entry, or widget control [34, 41, 42, 43]. A comprehensive overview of existing work in the field, can be found in [67]. However, these solutions are limited to a rigid form factor. In contrast, research in the domain of bendable interfaces (e.g., [15, 30, 55]) has demonstrated novel interaction techniques based on flexible sensing or input and output capabilities. Addressing this arising potential, SmartSleeve combines pressure-sensitive input with bending and stretching capabilities into a flexible input sensor that can form the basis for the design of more scalable, flexible, and transformable user interfaces [25].

Optical solutions leverage overhead cameras as in *Photoelastic Touch* [49], or structured light scanners as for *deForm* [12]. While these solutions were able to sense deformations of a

flexible surface or even clay deformations on the surface, they need space for the optical tracking system.

Actuated solutions such as [28] are constructed of pins and servo motors. These approaches require space and power, as well as limits input due to their rigid structures. Ferromagnetic input solutions [27] sense on base of a matrix of sensor coils (copper wire and permanent magnets). However, the form factor is limited, the sensor coils add weight and are more applicable for above-the-surface sensing.

Resistive solutions offer the potential of sensing deformations in thin form factors. *UnMousepad* [46] is constructed of several layers (FSR surface, resistive layer, conductor, clear substrate). *FlexSense* [45] is a thin-film sensing surface based on printed piezoelectric sensors. These solutions are already very thin by providing the ability of sensing deformations, but need a rigid backing.

*SmartSleeve* is designed to be worn directly on body, and thus needs to be fully flexible and soft, while having the capability to recognize a wide range of deformations. This is achieved by its thin textile form factor.

### Enabling always-available micro-interactions

In order to help users to perform micro-interactions, which are short-time interruptions [1], researchers have proposed a variety of ways to enable easy and fast access to mobile devices and overcome the limited interaction space on small form factor devices. Muscle input tracks the muscle tension to sense gestures [48]. Body-projected interfaces provide visual output, which is used for the interaction [18, 22]. Other approaches enlarge the interaction space by using sticky touch sensors [64], artificial skin [26] or enhancing the interaction space of existing devices [2, 38, 13]. While all these approaches are very diverse, they are all location variant.

### Real-time, continuous gesture recognition

A number of pressure based sensing have explored sport and activity tracking. Most closely related to our resistive textile hardware and nature of the signal are [59, 69, 70]. Although these techniques achieve good results, recognizing various types of gestures in a single classifier requires large amounts of training which is a laborious task. It further requires extensive handcrafted features especially for temporal information which is computationally expensive as well.

Typical learning-based gesture recognition approaches detect trained gestures. In this paper, we propose a hybrid algorithm of combining learning-based method with heuristics that are experimentally derived. This combination enables recognition of a wide variety of untrained classes with high accuracy at low computation cost, and shows robustness across different users and sessions.

More related to our approach is the Pose Recognition mechanism in [3] that distinguishes five body poses on the floor. We demonstrate how to extend a similar approach on clothing to derive 13 motion gestures from three trained classes of static ones.

In our work, we are specifically motivated to embrace the challenge of designing a low power algorithm that can seamlessly run in real-time on the limited hardware resources available in wearables.

## SMARTSLEEVE

SmartSleeve, pictured in Figure 1, is a fully wearable and highly deformable textile sensor that covers a large surface, features a high amount of sensors, and offers a high pressure resolution. In this section, we present the design of the sensor and the rapid fabrication. Table 1 provides an overview of the characteristics of the SmartSleeve sensor.

Parameter	Value
Force detected	50-500 g
Sample rate	100 Hz
Sensor resolution	1.66 sensor/inch
Sensor count	360 sensors
Weight in total	124 g
Length of the sleeve	40 cm
Upper arm perimeter	26.5 cm
Elbow perimeter	26.0 cm
Wrist perimeter	16.0 cm

Table 1: Sensor characteristics.

### Sensor Design

The SmartSleeve sensor builds on prior work [37, 32], that introduced pressure-sensitive textile sensors which consist of three layers of fabric. We will outline how this technology can be used as clothing to enhance wearers input possibilities without feeling rigid connection wires or other components added to the fabric.

All layers of SmartSleeve are equally bidirectionally stretchable and deformable. The top and bottom layers are made of Narrow Stripe Zebra fabric distributed by HITEK\*, characterized by alternating strips of conductive and non-conductive fabric, see Figure 2. The strips are 9 mm wide each. The zebra-fabric layers are orthogonally aligned to form a matrix. The middle layer consists of a pressure sensitive fabric (EeonTex™<sup>†</sup> LTT-SLPA 20 k). It has a slightly larger size to prevent the two conductive layers from shorting. Sandwiching all three layers creates a deformable and stretchable pressure-sensing matrix, which can be used to envelop complex 3D geometries. The three loose layers were stitched together along one side of the sensor to prevent the sensor grid from shifting. The sleeve constricts at the forearm part of the sleeve, which would lead electrical shortcuts. To prevent adjacent connections from shorting, an additional stretchable non-conductive fabric has been sewn lengthwise on the conductive layer, which conducts lengthwise (cf. Figure 2 b).

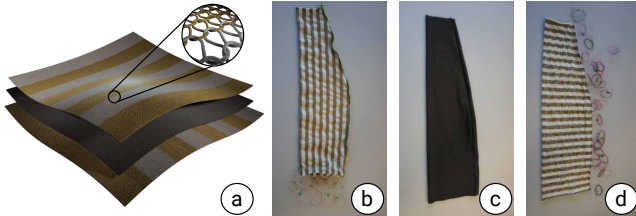


Figure 2: The sandwich architecture of the SmartSleeve sensor (a). The bottom layer (b) and top layer (d) have conductive and non-conductive threads. In-between is the pressure-sensitive layer (c).

\* [www.hitek-ltd.co.uk](http://www.hitek-ltd.co.uk)

<sup>†</sup> [www.eeonix.com](http://www.eeonix.com)

SmartSleeve is designed to cover the complete forearm and half of the upper arm. Even though this sensor technology can be easily scaled up to detect other body regions, prior work has shown that this region is most comfortable for interactions. The sleeve is designed to fit a human with a wrist perimeter size of 16 cm, elbow perimeter of 26 cm and an upper arm perimeter size of 26.5 cm. Early tests have shown that the sleeve has to fit tightly to reduce failure of short cutting adjacent wires, but not too tightly, in order to support deformation gestures. The sensor itself consists of 24 rows (around the arm) and 15 columns (lengthwise), resulting in a total of 360 pressure sensor spots with a sensor density of 1.66 sensors/square inch. SmartSleeve can be worn directly on the skin. To prevent errors from the influence of skin moisture, it was usually worn over a long-sleeved tight-fitting running shirt.

### Unobtrusive and Robust Sewn-Based Connection

In this section, we contribute an unobtrusive and robust method to connect not-rigid, stretchable textiles with the rigid electronics. Prior work has used rigid snap buttons [37, 32] to connect textile with electronics. However, using rigid connections negatively affects the comfort of the sleeve and its robustness. Therefore we explored alternative methods to connect textiles with electronics hardware.

Yarn would be the favourable connection due to its surface and shape behavior. Although many companies produce and sell conductive yarns<sup>‡</sup>, very few of these yarns withstand the soldering temperature, which is required to connect the yarns to the PCB board.

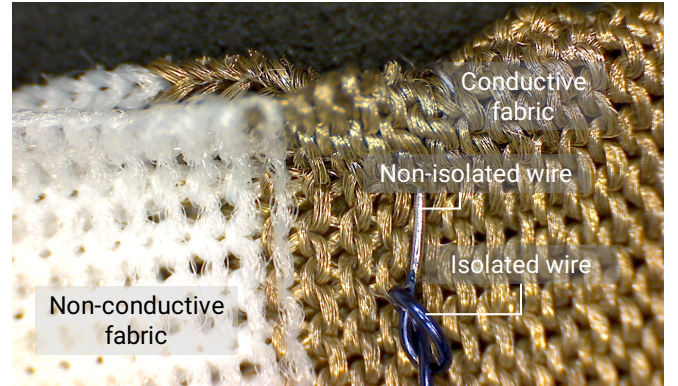


Figure 3: A hand-sewn connection between the textile sensor and the electronics provides a more flexible connection.

A possible alternative is solderable yarn<sup>§</sup>. Although these yarns are highly conductive, they are not insulated, which makes them unsuitable for our design as they would cause shortcuts. Previous research has also shown several ways to insulate conductive yarns by “couching”, iron-on techniques or fabric paint [5]. However, the tight sleeve needed a solution which preserves the look and feel, as well as, the comfort of use as much as possible.

Therefore, we opted for basing the connector on a wire with a small diameter which is conductive and insulated. During

<sup>‡</sup> [www.schoeller-wool.com](http://www.schoeller-wool.com),

[www.bekaert.com](http://www.bekaert.com),

[www.statex.biz](http://www.statex.biz),

[www.araconfiber.com](http://www.araconfiber.com)

<sup>§</sup> *High Flex 3981 7X1 Silver* or *High Flex 3981 Flat Braid* Karl Grimm, [www.karl-grimm.com](http://www.karl-grimm.com)



our experiments with different wires, we found the *Road Runner/Verowire* wire to be the most promising one. These copper wires are normally used for repairing or correcting printed circuit boards. The wire has a small diameter of 0.15 mm, which makes it very deformable. It is coated in solderable enamel or self fluxing polyurethane, which acts as an insulator. The coating can be removed when a high temperature is applied (400–430 °C). Hence, before sewing, we use the solder iron to remove 3 cm of insulation.

A first method consists of handsewing the connections. A close-up of one connection is depicted in Figure 3. It consists of 3 cm of non-insulated wire that is affixed using a stitch to a row or a column strip of the zebra fabric. Although this stitch itself is not stretchable, it requires little area and is therefore straightforward to be sewn by hand.

In addition to the manual fabrication, we also performed initial tests with a sewing machine using different stitching types usable for elastic materials, including *Zig Zag*, *Double Overlock*, and *Super Stretch*, see Figure 4. Straight stitches or stitches which are not adapted for elastic materials would either tear the yarn or reduce the fabrics elasticity. We found that the wire resists enough tension to be sewn with a *Zig Zag* stitch. Therefore, a non-conductive yarn was used as top thread and the wire as bobbin thread [11]. In this way, the bobbin thread can easily float on the back side of the stitch without passing through the fabric substrate when using the machine at maximum speed.

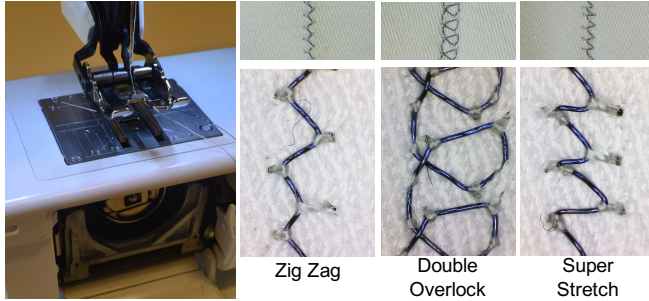


Figure 4: Fabricating the connection using a sewing machine, with *Zig Zag*, *Double Overlock*, or *Super Stretch* stitch.

These different stitching types have different benefits and limitations. As the wire is stiff, the equal stitch distances and the little use of yarn of the *Zig Zag* stitch preserve the comfort of use as much as possible. For the *Super Stretch* stitch, the yarn tension was raised to maintain the elasticity of the stitch. Otherwise, the wire would float in a straight line, which means that the stitch would no longer be elastic. This is due to the differences in the yarn elasticity between regular yarn (top) and wire (bobbin). Because of that, the top yarn can tear more easily. The *Double Overlock* keeps its typical pattern without making any changes regarding yarn tension. Nevertheless, we would not recommend to use this stitch as it needs more yarn, which makes the fabric stiffer and thus reduces the comfort of use. In conclusion, we would suggest to use the typical *Zig Zag* stitch, as the pattern maintains the comfort of use. Due to the simplicity of the pattern it is easily adjustable in its width and it is sewable with the predefined yarn tension and thus less vulnerable for tearing.

## Driver Electronics

SmartSleeve is based on a resistive tactile sensor. This type of sensor is subject to various sources of errors, such as crosstalk, which affect the accuracy of measurements and the gesture recognition. We evaluated different measurement principles and algorithms to determine the best solution to yield high accuracy and reduced crosstalk. First we analyzed how our system behaves with a solution without crosstalk reduction [53]. Further, we evaluated the effects of grounding for crosstalk reduction [10], the zero potential method [56] and virtual grounding [52], the multiplexer op-amp assist approach [51], and the resistive matrix approach [57].

Measurement Principle	Average Error
Without reduction [53]	34.5%
Grounding [10]	32.1%
Virtual Grounding [52]	42.6%
Multiplexer & Op-Amp assisted approach [51]	10.5%
Resistive Matrix Approach [57]	0.93%

Table 2: Overview of the measurement principles.

As depicted in Table 2, the Resistive Matrix Approach yielded the best results and was therefore implemented in our system. The measurement electronics consist of a microcontroller, one single pole double throw switch, four multiplexers and four shift registers. The shift registers are daisy-chained so that they work as one big shift register. The shift register applies ground potential to the measured column while all other columns are connected to high potential. Whenever the shift register is triggered, the low level jumps to the next column. Multiplexers are connected to the row electrodes to forward single lines to the ADC. Each single sensor spot is measured separately, which means starting from the constant resistors which are mounted on the PCB to the first cells in the row and first column to all others. Then all other sensors gets measured row by row.

## INTERACTION TECHNIQUES

The unique SmartSleeve features enable a wide variety of textile interactions. The sensor’s *large input surface* affords input on multiple body locations and with both fine and gross gestures. The sensor resolution enables both conventional *2D Surface Gestures* and *2.5D Deformation Gestures*. The high *pressure resolution* provides continuous force sensing and improves accuracy for detection of deformation-based gestures. The design and implementation of the sleeve is based on previous work [19], which has shown that the forearm is the most comfortable position to interact with.

Based on these properties, we composed a set of candidate gestures and conducted several brainstorming sessions. The candidate gestures were then refined in an iterative design process through several ideations with external participants and two pilot studies, including a guessability study. The result of this iterative process is a set of nine types of gestures, cf. Figure 5, where eight of them have been discussed in previous research. *Surface Gestures* are planar gestures, which are performed on the textile, similar to conventional touch gestures. *Deformation Gestures* (e.g. *Fold*, *Bend*, *Twist*) are based on deforming the textile in more dimensions. Previous work has focused on conceptualizing new gestures [31, 62, 68], and on implementing one or a few gestures in a working system
















Related work	Input space	Surface Gestures					Deformation Gestures							
		Wrist to upper arm					Forearm to upper arm				Wrist			
														
Surface Gestures [68]	Tabletop	o	o	o	o	o								
On-Body Interaction [21]	Skin						×							
More than Touch [65]	Skin	o		o			o	o						
PrintSense [16]	Film	×		×					×					×
Flexy [63]	Film													×
iSkin [64]	Film	×	×	×	×	×								
Pinstripe [29]	Textile							×						
GestureSleeve [54]	Textile	×	×											
Deformable displays [31]	Textile						o	o	o		o		o	o
Elastic displays [62]	Textile						o	o						
Grabbing at an angle [17]	Textile							×						
AugmentedForearm [36]	Textile												×	
<b>SmartSleeve</b>	Textile	×	×	×	×	×	×	×	×	×	×	×	×	×

Table 3: The SmartSleeve gesture set compared with previous work (o = conceptual, × = functional).

[29, 17, 36, 54]. SmartSleeve, however, provides a unified sensing framework which allows us to detect all of them within a single pipeline, as shown in Table 3. Furthermore, most of the gestures can be done at any location on the sleeve, in contrast to previous work, which restricts gestures to smaller, dedicated, instrumented areas.

In addition to classifying a gesture, SmartSleeve detects three properties: The *Location* ( $L$ ) where the gesture is performed on the sleeve, its *Direction* ( $D$ ) and its *Pressure* ( $P$ ) intensity. Note that not all gestures can use all properties: the *Bend* gesture, for example, can detect the pressure intensity, but its location is fixed at the user’s elbow joint. Overall, these properties improve the quality of the gesture recognition, but they can also be used as design parameters. Using a property like *Direction* considerably expands the possible gesture set. To make use of these properties for certain gestures, we implemented a hybrid gesture detection approach, which takes advantage of the properties where appropriate. In total, our gestures set includes 22 gestures (7 *Deformation Gestures* + 2 *Surface Gestures* + 1 derived *Deformation Gesture* + 12 derived *Surface Gestures*) as shown in Figure 5. We will now discuss the *Surface Gestures* and *Deformation Gestures* in more detail.

### Surface Gestures

Previous work has shown that users tend to transfer conventional multi-touch gestures to other modalities - especially for standard commands [65, 31, 62]. Therefore, it was necessary to support a broad set of *Surface Gestures*, as depicted in Figure 5. By making use of location, direction, and pressure properties, we are able to derive even more gestures, as shown in Figure 5. In the case of the derivatives, we distinguish between the following:

**Swipe** 2D motion with the finger or hand on the sleeve affords relative or absolute positioning. Thus, the system can support traditional touch interactions, where surface interactions are mapped to, e.g., navigation, scrolling and panning. The ability to distinguish between finger and hand makes it possible to differentiate between coarse and fine control. In addition, spatial differentiation between input regions can extend the interaction space. For example, in a 3D modelling

application, a movement across the forearm could mean a rotation around the y-axis, while the same movement across the lower arm could be recognized as a rotation around the x-axis. Our eyes-free media player uses finger left/right swipes to skip forward/backward in a track, while a left/right swipes with the hand changes track. When our media player is used with visual feedback, finger motion can be used for cursor control, and swipe for menu option navigation.

**Rub** 1D back-and-forth motion with the finger or hand resembles the metaphor of scratching something out with a pen. Thus, we found it attractive to map it to deletion. It could be used for deletion of an element, like dismissing a message or deleting a calendar entry [68]. In addition, the pressure intensity can be used to delete one or a whole set of items at once. In our media player, rubbing removes the current track from the playlist.

**Spread/Close** These gestures are widely used in “pinch-to-zoom” interactions on multi-touch devices. They are derived from the *Finger* gesture and make use of location and direction (see Figure 5). While these gestures use multiple fingers, the algorithm is the same. This interaction is applicable to scalable interfaces with visual feedback, such as, for map navigation, and image manipulation. These commands (except spreading and closing) can be performed by one finger, multiple fingers, or the full hand—depending on the gesture.

### Deformation Gestures

In addition to *Surface Gestures*, SmartSleeve enables a wide range of *deformable gestures*. The thin and elastic textile sensor material affords freeform manipulation and deformation, while our sensing technique detects gestures, state changes, and continuous manipulation.

**Twist** Pinching the textile and twisting it affords rotational control. The analogy to a physical knob makes it suitable for actions that map to clockwise or counterclockwise motion. In our media player, we use this gesture to increase/decrease the volume. The ability to sense location, also allows multiple virtual knobs along the textile—for example, to control an equalizer or left/right balance. Pressure might be used to control the rate (light touch would change the value more slowly). The physical constraints that prevent the gesture to

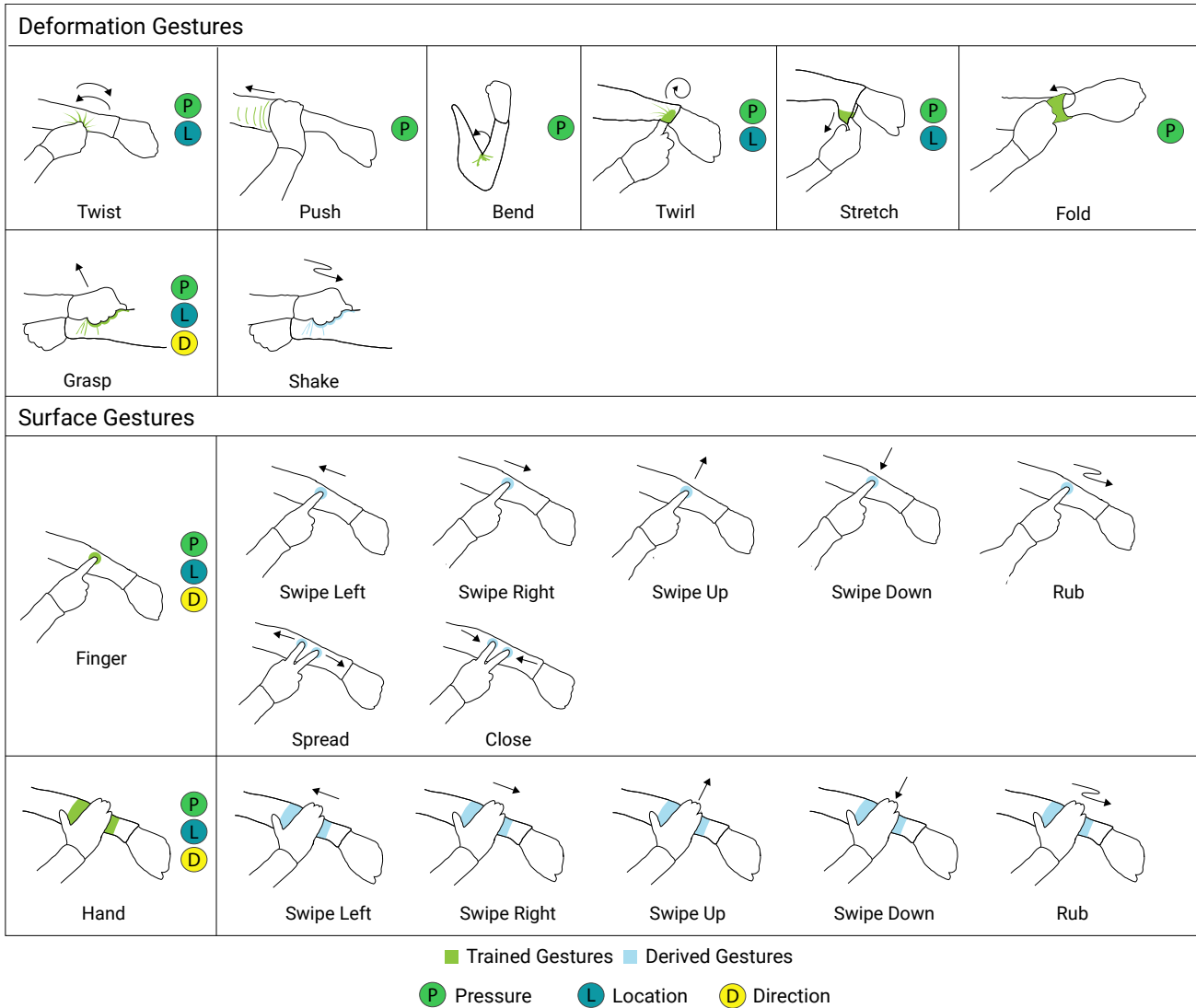


Figure 5: The SmartSleeve gesture set comprises 22 gestures (7 Deformation Gestures + 2 Surface Gestures + 1 derived Deformation Gesture + 12 derived Surface Gestures)

be rotated infinitely match the physical affordances of control knobs that map to a value range. Our sensor currently does not support infinite rotation, which can be found in scroll wheels, or continuous rotary encoders.

**Push** Pushing the sleeve up can be treated as a state change, e.g., to hide information [36]. The compressed sleeve provides implicit visual and tactile feedback about the state. Our media player uses this state to toggle mute, or to hide the UI or media if used with visual feedback.

**Fold** Folding the sleeve is another way to change state. Here, we rely on the difference in *operation* to distinguish it from Push. While the end result may look similar, this operation requires careful effort to perform. With our media player, we map this operation to entering recording mode.

**Twirl** Twirling the textile around the finger requires intentional coordination. It uses the metaphor of the "reminder knot" around a finger. We use it to assign importance to the current item in the interface. The media player lets users rate

a track by assigning a "star" or "like" with the gesture. When used with an audio book, podcast or radio show, it sets a book-mark. One could also imagine saving the currently playing voice mail message, or using it to record a voice memo. Location for the Twirl can be used to later enable retrieval with random access.

**Grasp** Grasping consists of the user grabbing the textile and pulling it together into the fist. We use it as a metaphor for retrieval. This, for example, allows us to complement Twirl with a mechanism for activating a saved item. The location can be mapped to specify which saved item to retrieve.

**Shake** Shake is a derived gesture from *Grasping* (cf. Figure 5). The metaphor is based on grabbing a container with objects and shaking it. We map it to shuffling the tracks in our media player. Other considered mappings would be to clear the list or to close the application [31].

**Stretch** Stretching consists of pulling on the textile at a specific location. It affords elastic input as the textile retracts

when released. We use the metaphor of turntable control, where stretching controls playback speed in our media player. Stretching it towards the user increases the speed, while pulling it away decreases the speed.

**Bend** Bending of the elbow is an example of the implicit sensing that is possible with our technique. As this motion is part of the user’s natural movement, we would need to use a disambiguating mechanism, e.g., pressure or combination with another gesture, to activate it if used as an explicit command. Another opportunity is to use it as implicit input. For our media player, we have explored using the bending that occurs from arm swinging while running as a way to detect the appropriate tempo for the music playlist.

Most of our gestures, can be recognized at different sensor locations and on different parts of the anatomy (e.g., forearm, elbow, upper arm). Some gestures are naturally limited by mechanical, ergonomic or physical constraints. For example, gestures like *Push*, *Fold*, *Twirl* and *Stretch* are performed at the end of the sleeve. Gestures like *Bend* or *Twist* are limited to the user’s physical abilities.

These examples illustrate how 2D surface gestures, 2.5D deformation gestures, and the three continuous properties (location, direction, pressure) have the opportunity to greatly expand the opportunities for linking and mapping information while taking into account nuanced properties, such as, recency and importance [65, 20].

#### HYBRID GESTURE DETECTION ALGORITHM

Our novel algorithm addresses the challenges of learning-based methods that need extensive training and extraction of hand-crafted features for class-label prediction. To alleviate these issues, we combine the results from the trained model with a heuristic-based approach that relies on a set of rules. Heuristic-based models provide a unique capability to react to untrained classes. So, we can train a much smaller set of trained gestures and achieve a wide variety of different interactions.

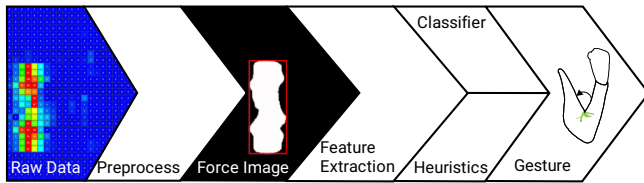


Figure 6: The SmartSleeve gesture detection pipeline is based on six steps.

Figure 6 presents the entire SmartSleeve gesture detection pipeline. Our approach has some similarities with the method used in GravitySpace [3]. Similarly, we reduce a 3D problem to a 2D problem by constructing a force image using the raw sensor matrix data. Next, we construct a force image with a size of  $24 \times 15$  px with an overall framerate of 30 fps. Our algorithm is training a classifier on a per-frame-basis. Similar to the approach *Type-Hover-Swipe* [60], we have developed a simple filter, which averages the current sample with the previous ten samples to handle the *false* deformations on the *SmartSleeve*. Consequently, we can stabilize the natural tremor of hands and obtain temporal information. In addition, we leverage our continuous gesture tracking mechanism for di-

rectional information. Particularly, we incorporate the results from learning-based algorithm and heuristic-based approach to reduce the training of gestures by half as well as decrease computationally expensive feature extraction.

**Preprocessing and Feature Extraction** Initially, we convert the raw sensor data to a grayscale force image. By applying a threshold we remove the noise. Further, we specify the foreground and background on an individual pixel to search for points of interest, which in turn results in the loss of the pressure information. However, this information is later recovered by calculating the average force from the raw sensor data, once the Region of Interest (RoI) is located. In the next step, we apply bilinear upscaling and a Gaussian filtering for smoothing the raw force image. The RoI is selected as a mask inside the bounding box using the blob detection model. We use the contour detection algorithm [9], which makes the gesture classification space invariant. The removal of the pressure information from the force image yielded significant improvements in terms of classification accuracy during an informal pre-study with different users. Additionally, it helps us reducing the number of training trials, as the processed images for feature extraction appear similar even when the applied force changes up to a certain limit for a particular gesture during the training phase.

As all the regions are highly discriminative (see Figure 7), we only compute a simple histogram and a set of properties of the contour’s bounding box, including *height*, *width*, *area*, and *perimeter*, as features for the classification without any further processing.

**Classifying gestures based on image analysis** In order to identify the gestures, we took a subset from Figure 5 based on their variances, namely *Finger*, *Hand*, *Bend*, *Twist*, *Push*, *Grasp*, *Stretch*, *Twirl*, and *Fold*. We experimented using the image features which we extracted above for these nine gestures as input to different classifiers and found that a Support Vector Machine (SVM) yields the most promising results, with parameters  $C = 1.0$  and  $kernel = RBF$ , optimized using a grid search with cross-validation implementation provided by the *scikit-learn* [39] machine learning library. The model assigns probabilities for each type of trained gesture.

**Detecting untrained gestures using heuristics** We extend our frame-by-frame learning-based algorithm for recognizing dynamic (untrained) gestures by adding a set of rules based on the classifier generated probability distribution. This idea is inspired by the *Pose Recognition* technique used in GravitySpace [3]. In our implementation, we combine the highest probability with net force and properties of the blob (*position* and *size*) to produce more gestures (*finger swipe up, down, left, right, and rub; hand swipe up, down, left, right, and rub; spread; close; shake*). As mentioned earlier, we deduce the normalized force from the raw sensor data within a blob and our frame averaging helps us to track the blob within successive image frames.

Specifically, to identify the direction (*up, down, left, or right*) of the gesture, we simply store the consecutive blob’s centroid coordinates in a buffer and compute the slope through each pair of points. Additionally, we implemented a consistency check algorithm to overcome the accidental deviations in slope values since a user might not be able to draw a straight line (e.g., left to right gesture) and the slope might have some unintended



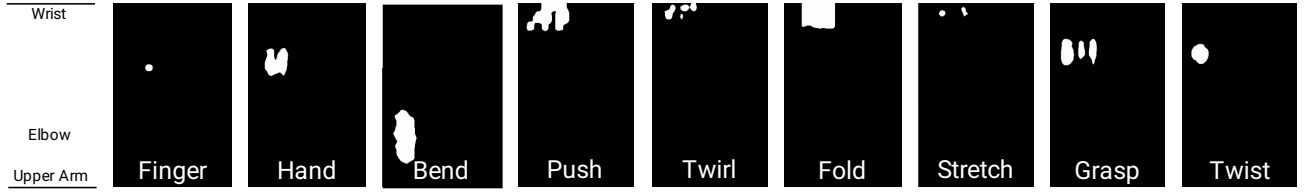


Figure 7: Training input frames after preprocessing.

motions, which can be interpreted as an up- or down-gesture. We fix this problem by splitting the whole gesture into overlapping regions and further ignore small deviations.

Probability distributions of *grasp* and *hand* from the classifier help determine *shake* and *rub* respectively, the blob's centroid coordinates change significantly during *rub* and *shake*, cf. Figure 8, (a) and (b) compared to other gestures, we use the concept of first order derivative from calculus to compute this rate of change, since the obtained value is scalar, we make it absolute and take an average on a finite size to avoid false positives. Gestures including *spread* and *close* exploit the classification probability of *finger* combined with the linear change in area of contour (see Figure 8 (c) and (d), we measure the trend of increment or decrement with simple subtraction between pairs of consecutive area values. Afterwards, we assess the average to determine the action robustly wherein a positive value signifies expansion, and vice versa.

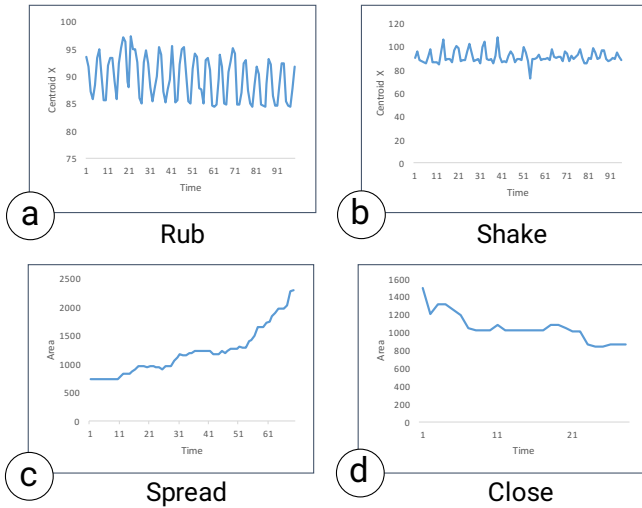


Figure 8: Change in contour's centroid during rub and shake (a, b), and change in area of the bounding box while performing spread and pinch over time (c, d).

Summarizing, our real-time recognition pipeline provides *continuous* gesture detection. In particular, if we attempt a swipe gesture, we have an additional component of changing pressure along the line. This feature allows us an additional input modality and could be used for controlling speed in activities like *fast forward* in a media player.

## EVALUATION

Two empirical studies were conducted to evaluate our hybrid gesture detection algorithm. In the first experiment, we evaluated the learning based algorithm and we were primarily

interested in finding out if the trained gesture set would also be position-invariant. On the other side, in the second experiment, we were focusing on the evaluation of the heuristic approach, where we wanted to find out if the heuristics we implemented will help to enrich the set of gestures while training only a subset of gestures.

## Participants

Six unpaid volunteers (3 female), 23-25 years old ( $\bar{x} = 24.33$ ,  $SD = 0.94$ ), all right-handed were recruited from the local university. All participants used 2D touch interfaces on a daily base, but none of them had experiences with smart clothing interfaces.

## Apparatus

The study was conducted in a quiet room, where the participants were wearing the SmartSleeve tightly fitted as depicted in Figure 9. The instructions were displayed on LG 24" 1920 × 1200 pixel IPS LCD screen.

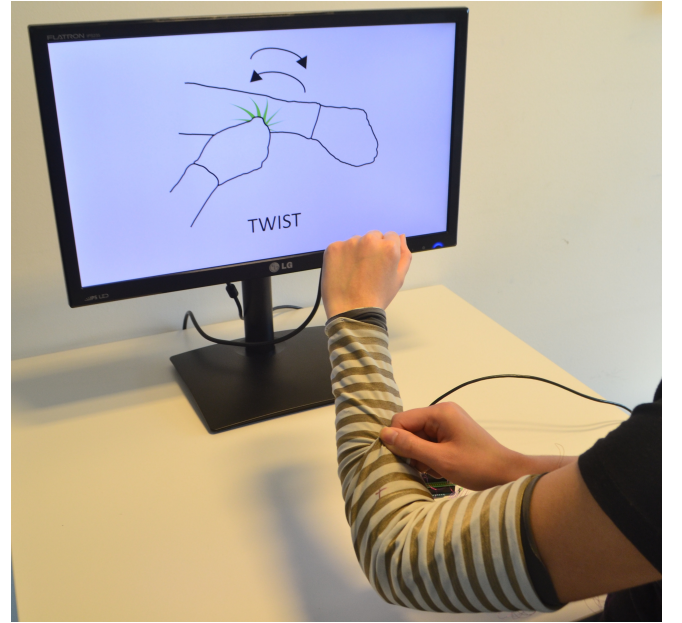


Figure 9: Apparatus used in both experiments consisted of the participant-worn SmartSleeve and a screen where gestures, instructions and feedback were displayed.

## Experiment 1: Position-invariant Gesture Recognition

In the first experiment, we performed a gesture recognition experiment, where we wanted to find out if our approach also provides good accuracy results even if the gestures have to be performed on different locations of the textile.

**Design** At the beginning of the experiment, the nine different gesture types (i.e., *Finger*, *Hand*, *Twist*, *Bend*, *Stretch*, *Fold*, *Push*, *Grasp*, *Twirl*) were demonstrated to participants for clarity. Thereafter, participants were instructed to perform four trials of each gesture type in randomized order to train the gesture recognition engine. The four trials had to be performed on the same location, which was marked accordingly. The training phase took approx. 15 minutes. Next, participants were asked again to perform eight trials of each gesture type on the (a) same location (marked position) as well as on (b) an arbitrary. The order of the gesture type was randomized and presented accordingly on the on-screen prompts (see Figure 9). The on-screen prompt further showed whether the gesture was recognized correctly or wrongly. The testing phase took about 35 minutes per participant. Collected measurements included error rate.

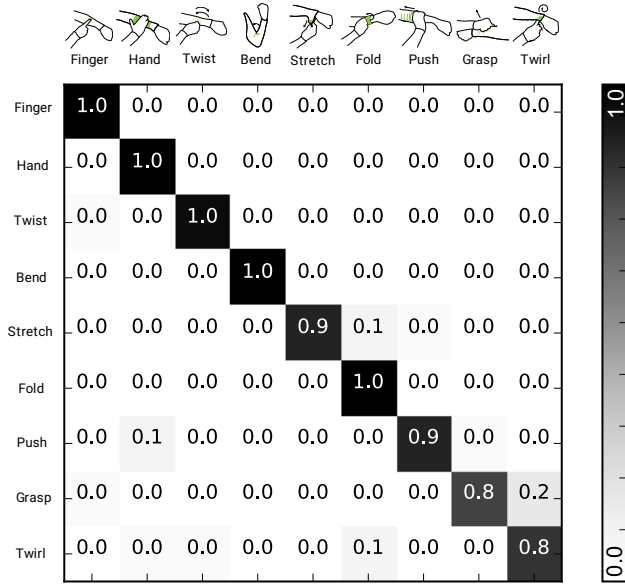


Figure 10: The standard confusion matrix for the SmartSleeve hardware using the same location.

**Results** Our approach reached an average accuracy of 92.0% when the system is trained and tested by the same participant on the same location, cf. Figure 10. Similar results have been achieved when the system is trained and tested by the same participant on different location with an average of 86.9%, cf. Figure 11.

In more detail, the finger gesture achieved an average recognition rate of 96.4% ( $SD = .05$ ), 100% ( $SD = 0.0$ ) for the bend gestures, and 83.6% ( $SD = .167$ ) for the twirl gestures. Using different locations, an average recognition rate of 92.6% ( $SD = .121$ ) for the finger gestures, 98.0% ( $SD = .05$ ) for the bend gestures, and 80.0% ( $SD = .244$ ) for the twirl gestures was achieved.

A repeated measures ANOVA was carried out and revealed a significant effect for the location ( $F_{1,35} = 6.019$ ;  $p < .05$ ) as well as gestures ( $F_{8,35} = 5.865$ ;  $p < .001$ ). Finally, post-hoc analyses on the main effects were conducted based on paired-samples t-tests using the Holms sequential Bonferroni approach. However, no significant differences could be found.

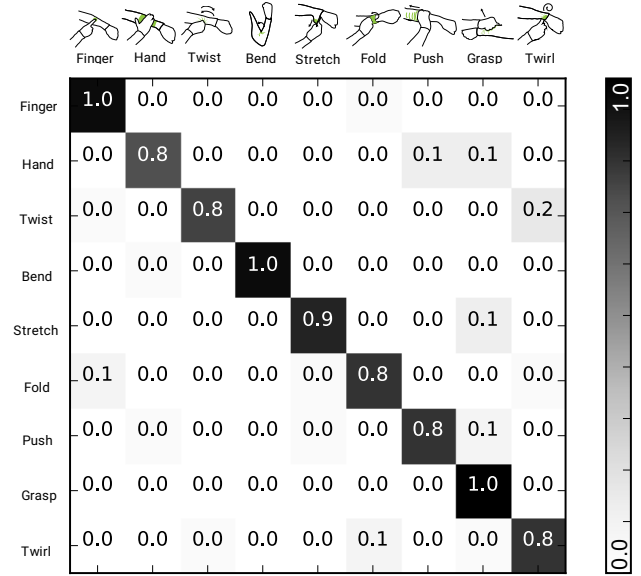


Figure 11: The standard confusion matrix for the SmartSleeve hardware using an arbitrary location on the arm chosen by the participants.

## Experiment 2: Testing the heuristics approach

In the second experiment, we evaluated the performance of the heuristics of the gestures using the parameters *pressure*, *location* and *direction*. We therefore wanted to find out if we are able to recognize more complex gestures, which are based on simple ones.

**Design** We evaluated the recognition implementation with the same participants. As all the used gestures were based on the gestures described before, no training data was required for this evaluation. Again, we showed each new gesture on a screen and the participant performed it accordingly.

For the second experiment, we chose a subset of the gesture classes (i.e., *Finger*, *Hand*, and *Grasp*), as all of them are making use of all the used parameters (*pressure*, *location* and *direction*), cf. Figure 5. Every participant performed each gesture 5 times in total, resulting in an overall of 80 trials per participant. For the gesture class *Finger*, for instance, participants had to perform the gestures *swipe right*, *swipe left*, *swipe up*, *swipe down*, *rub*, *spread*, and *close* with the index finger on an arbitrary location of the SmartSleeve. The same gestures were performed using the *Hand* gesture class. Finally, participants also *grasped* and *shaked* the sleeve. All the gestures were counterbalanced to avoid training effects.

**Results** Overall, 84% ( $SD = 0.11$ ) of all gestures were correctly identified. In more detail, the simple gesture detection (*swipe right*, *swipe left*, *swipe up*, *swipe down*) using the finger achieved an average recognition rate of 83% ( $SD = .06$ ), while we achieved 97% for the *spread* gesture. Only most complex deformation gestures, like the shake achieved a lower average recognition rate of 73%.

## Technical Evaluation

To evaluate the pressure sensing behaviour of the sensor we conducted a technical study, where we applied mechanical stress onto the surface of the SmartSleeve sensor.

**Apparatus** The sensor got stationary mounted onto a flat deformable Styrofoam surface. A hemispherical thrust plate with a 4 mm diameter applied mechanical stress onto the surface of the sensor. The resistance change of the sensor was measured 10 times with a force of 25 g, 50 g, 75 g, 100 g, 250 g, 500 g, 1000 g.

**Results** As shown in Figure 12, the sensor demonstrate promising sensing behaviors from 50 g to 500 g. Underneath 50 g the sensor demonstrates high resistance changes as expected, however according to the loose stacking of the sensor the standard deviation is high. Beyond 500 g the resistance of the sensor shows slight changes, therefore disregard this area in SmartSleeve as well.

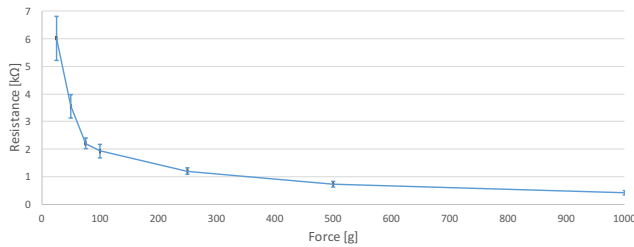


Figure 12: Pressure Sensing under different stress levels.

## DISCUSSION & LIMITATIONS

The SmartSleeve was trained and tested for a period of four months in total. During that time, the sensor was used approximately 120 sessions by several people who also provided early feedback.

### Durability

We observed that the sensor signal did not change significantly even under different pressure conditions. Regarding the connections, we implemented two versions. The first version of our sensor had a rough rigid connection using flat ribbon cables (2.54 mm pitch) connecting the sensor PCB board with the textile. The connections of this prototype broke relatively easily, as the cable was stiff and heavy and the soldering spots were comparably small. Further, having this rigid soldered connection directly on the sleeve leads to breaks, while performing a highly deformable gesture activity. The second version of SmartSleeve with the new sewn connection (as introduced in this paper) was then tested for approximately three months. During that time, the sleeve was used more than 100 times by different participants. Only three smaller issues had to be fixed.

### Pressure Input

As the results of the evaluation show, SmartSleeve accurately recognizes 2D gestures (*surface gestures*) as well as 2.5D deformation-based gestures (*deformation gestures*) that are performed on the textile. Even though our algorithm takes advantage of pressure to detect 2.5D gestures, we have not evaluated the pressure itself for *surface gestures*. The high pressure resolution of the sensor could be used for enhancing surface gestures. Yet, as noted by Rendl et al. [44], pressure is a very subjective property and its perception differs from person to person. Therefore, we did not formally evaluate this aspect in this paper.

## False Positives

The evaluation and exploration of the sleeve show that elbow movements (bending) can result in false positives and lower accuracy, while flexion and extension of shoulder or wrist have no significant influence. Surface gestures are more prone to false activations, given that the more specific deformations occur less frequently than accidental touches.

## Three-layer Approach

While the three-layer approach is a practical solution for a tactile sensor that measures signals via a resistive approach, it was problematic in a few instances when the user grasped only the top layer for performing a gesture. Moreover, a three-layer sandwich is thicker, decreases the comfort of use and needs more implementation effort. We are currently developing a one-layer solution to address these limitations.

## Tailored Clothing

SmartSleeve is tailored for a person having a wrist perimeter of 16 cm, an elbow perimeter of 26 cm and an upper arm perimeter of 26.5 cm. Although all three layers are bidirectionally stretchable, observations with other participants have shown that the sleeve itself has to fit tightly, as it has to follow the rotation of the arm. If only the upper arm and elbow are fitting well, but the sleeve itself is loose on the lower arm, people can rotate the underarm inside the sleeve, which could lead to reduced accuracy. Generally, as SmartSleeve is a personal wearable, it should be tailored for each individual person to enable rich input on clothes.

## CONCLUSIONS AND FUTURE WORK

We introduced *SmartSleeve*, a flexible textile sensor that senses both *surface gestures* and *deformation gestures* in real-time. We provided a detailed description of our hybrid gesture detection pipeline that uses learning-based algorithms and heuristics to enable real-time gesture detection and tracking. Its modular architecture, combined with a large sensor size, a high spatial resolution and a high pressure resolution, allowed us to derive new gestures through the combination with continuous properties like *pressure*, *location*, and *direction*. Finally, we reported on the promising results from our evaluations which demonstrated real-time classification of 9 gestures with 89.5% accuracy.

In future work, we plan improvements to our SmartSleeve hardware, as the current proof-of-concept implementation relies on a wired PC connection for data transmission and power supply. Given the current hardware's dimensions ( $102 \times 53 \times 25\text{mm}$ ), miniaturization of the electronics would allow us to embed it in the textile. Therefore, we are developing a version, which is completely mobile with wireless connectivity. Additionally, we are working on a single-layer textile sensor implementation and developing the algorithm on a hardware level. We also want to explore how SmartSleeve performs in everyday scenarios and how environmental influences, like humidity, affect the sensor. Our initial experiments show that the sensor withstands machine washing at low temperature and slow spin, however, more formal evaluations should be performed to assess its durability.

## REFERENCES

1. Daniel Ashbrook. 2010. *Enabling Mobile Microinteractions*. Dissertation. Georgia Institute of Technology. <http://smartech.gatech.edu/handle/1853/33986>



2. Patrick Baudisch and Gerry Chu. 2009. Back-of-device interaction allows creating very small touch devices. In *CHI*. ACM Press, New York, New York, USA, 1923. DOI: <http://dx.doi.org/10.1145/1518701.1518995>
3. Alan Bränzel, Christian Holz, Daniel Hoffmann, Dominik Schmidt, Marius Knaust, Patrick Lühne, René Meusel, Stephan Richter, and Patrick Baudisch. 2013. GravitySpace: tracking users and their poses in a smart room using a pressure-sensing floor. In *CHI*. ACM Press, New York, New York, USA, 725. DOI: <http://dx.doi.org/10.1145/2470654.2470757>
4. Leah Buechley. 2008. SensorMania - Notes from e-Fashion day. (2008). <http://www.mediatomic.net/36637/en/sensormania>
5. Leah Buechley and Michael Eisenberg. 2009. Fabric PCBs, electronic sequins, and socket buttons: Techniques for e-textile craft. *Personal and Ubiquitous Computing* 13, 2 (feb 2009), 133–150. DOI: <http://dx.doi.org/10.1007/s00779-007-0181-0>
6. Leah. Buechley, Kylie A. Peppler, Michael Eisenberg, and Yasmin B. Kafai. 2013. *Textile messages : dispatches from the world of e-textiles and education* (2 ed.). Peter Lang. 246 pages. <http://dx.doi.org/10.3726/978-1-4539-0941-6>
7. Lina M Castano and Alison B Flatau. 2014. Smart fabric sensors and e-textile technologies: a review. *Smart Materials and Structures* 23, 5 (may 2014), 053001. DOI: <http://dx.doi.org/10.1088/0964-1726/23/5/053001>
8. Jared Cechanowicz, Pourang Irani, and Sriram Subramanian. 2007. Augmenting the mouse with pressure sensitive input. In *CHI*. ACM Press, New York, New York, USA, 1385. DOI: <http://dx.doi.org/10.1145/1240624.1240835>
9. Fu Chang and Chun Jen Chen. 2003. A component-labeling algorithm using contour tracing technique. In *ICDAR*, Vol. 2003. IEEE Comput. Soc, 741–745. DOI: <http://dx.doi.org/10.1109/ICDAR.2003.1227760>
10. Tommaso D'Alessio. 1999. Measurement errors in the scanning of piezoresistive sensors arrays. *Sensors and Actuators A: Physical* 72, 1 (1999), 71–76. DOI: [http://dx.doi.org/10.1016/S0924-4247\(98\)00204-0](http://dx.doi.org/10.1016/S0924-4247(98)00204-0)
11. Lucy E. Dunne, Kaila Bibeau, Lucie Mulligan, Ashton Frith, and Cory Simon. 2012. Multi-layer e-textile circuits. In *UbiComp*. 649. DOI: <http://dx.doi.org/10.1145/2370216.2370348>
12. Sean Follmer, Micah Johnson, Edward Adelson, and Hiroshi Ishii. 2011. deForm: an interactive malleable surface for capturing 2.5D arbitrary objects, tools and touch. In *UIST*. ACM Press, New York, New York, USA, 527. DOI: <http://dx.doi.org/10.1145/2047196.2047265>
13. Markus Funk, Alireza Sahami, Niels Henze, and Albrecht Schmidt. 2014. Using a touch-sensitive wristband for text entry on smart watches. In *CHI EA*. ACM Press, New York, New York, USA, 2305–2310. DOI: <http://dx.doi.org/10.1145/2559206.2581143>
14. Guido Gioberto, James Coughlin, Kaila Bibeau, and Lucy E Dunne. 2013. Detecting Bends and Fabric Folds using Stitched Sensors. In *ISWC*. New York, New York, USA, 53–56. DOI: <http://dx.doi.org/10.1145/2493988.2494355>
15. Antonio Gomes, Andrea Nesbitt, and Roel Vertegaal. 2013. MorePhone: an actuated shape changing flexible smartphone. In *CHI*. ACM Press, New York, New York, USA, 583. DOI: <http://dx.doi.org/10.1145/2470654.2470737>
16. Nan-Wei Gong, Jürgen Steimle, Simon Olberding, Steve Hodges, Nicholas Edward Gillian, Yoshihiro Kawahara, and Joseph A. Paradiso. 2014. PrintSense: a versatile sensing technique to support multimodal flexible surface interaction. In *CHI*. ACM Press, New York, New York, USA, 1407–1410. DOI: <http://dx.doi.org/10.1145/2556288.2557173>
17. Nur Al-huda Hamdan, Jeffrey R Blum, Florian Heller, Ravi Kanth Kosuru, and Jan Borchers. 2016. Grabbing at an angle: menu selection for fabric interfaces. In *ISWC*. ACM Press, New York, New York, USA, 1–7. DOI: <http://dx.doi.org/10.1145/2971763.2971786>
18. Chris Harrison, Hrvoje Benko, and Andrew D. Wilson. 2011. OmniTouch: wearable multitouch interaction everywhere. In *UIST*. ACM Press, New York, New York, USA, 441. DOI: <http://dx.doi.org/10.1145/2047196.2047255>
19. Chris Harrison and Haakon Faste. 2014. Implications of location and touch for on-body projected interfaces. In *DIS*. ACM Press, New York, New York, USA, 543–552. DOI: <http://dx.doi.org/10.1145/2598510.2598587>
20. Chris Harrison, Brian Y. Lim, Aubrey Shick, and Scott E. Hudson. 2009. Where to locate wearable displays?. In *CHI*. ACM Press, New York, New York, USA, 941. DOI: <http://dx.doi.org/10.1145/1518701.1518845>
21. Chris Harrison, Shilpa Ramamurthy, and Scott E. Hudson. 2012. On-body interaction: armed and dangerous. In *TEI*. ACM Press, New York, New York, USA, 69. DOI: <http://dx.doi.org/10.1145/2148131.2148148>
22. Chris Harrison, Desney Tan, and Dan Morris. 2010. Skinput: appropriating the skin as an interactive canvas. In *CHI*. ACM Press, New York, New York, USA, 453. DOI: <http://dx.doi.org/10.1145/1753326.1753394>
23. Florian Heller, Stefan Ivanov, Chat Wacharamanatham, and Jan Borchers. 2014. FabriTouch: exploring flexible touch input on textiles. In *ISWC*. ACM Press, New York, New York, USA, 59–62. DOI: <http://dx.doi.org/10.1145/2634317.2634345>
24. Christopher F Herot and Guy Weinzapfel. 1978. One-point touch input of vector information for computer displays. *ACM SIGGRAPH Computer Graphics* 12, 3 (1978), 210–216. DOI: <http://dx.doi.org/10.1145/965139.807392>
25. David Holman and Roel Vertegaal. 2008. Organic user interfaces: designing computers in any way, shape, or form. *Commun. ACM* 51, 6 (2008), 48. DOI: <http://dx.doi.org/10.1145/1349026.1349037>
26. Christian Holz, Tovi Grossman, George Fitzmaurice, and Anne Agur. 2012. Implanted user interfaces. In *CHI*. ACM Press, New York, New York, USA, 503. DOI: <http://dx.doi.org/10.1145/2207676.2207745>

27. Jonathan Hook, Stuart Taylor, Alex Butler, Nicolas Villar, and Shahram Izadi. 2009. A reconfigurable ferromagnetic input device. In *UIST*. ACM Press, New York, New York, USA, 51–54. DOI: <http://dx.doi.org/10.1145/1622176.1622186>
28. Sungjune Jang, Lawrence H Kim, Kesler Tanner, Hiroshi Ishii, and Sean Follmer. 2016. Haptic Edge Display for Mobile Tactile Interaction. In *CHI*. 3706–3716. DOI: <http://dx.doi.org/10.1145/2858036.2858264>
29. Thorsten Karrer, Moritz Wittenhagen, Leonhard Lichtschlag, Florian Heller, and Jan Borchers. 2011. Pinstripe: eyes-free continuous input anywhere on interactive clothing. In *CHI*. ACM Press, New York, New York, USA, 1313. DOI: <http://dx.doi.org/10.1145/1978942.1979137>
30. Byron Lahey, Audrey Girouard, Winslow Burleson, and Roel Vertegaal. 2011. PaperPhone: understanding the use of bend gestures in mobile devices with flexible electronic paper displays. In *CHI*. ACM Press, New York, New York, USA, 1303. DOI: <http://dx.doi.org/10.1145/1978942.1979136>
31. Sang-Su Lee, Sohyun Kim, Bipil Jin, Eunji Choi, Boa Kim, Xu Jia, Daeop Kim, and Kun-pyo Lee. 2010. How users manipulate deformable displays as input devices. In *CHI*. ACM Press, New York, New York, USA, 1647. DOI: <http://dx.doi.org/10.1145/1753326.1753572>
32. Joanne Leong, Patrick Parzer, Florian Perteneder, Teo Babic, Christian Rendl, Anita Vogl, Hubert Egger, Alex Olwal, and Michael Haller. 2016. proCover: Sensory Augmentation of Prosthetic Limbs Using Smart Textile Covers. In *UIST*. 335–346. DOI: <http://dx.doi.org/10.1145/2984511.2984572>
33. Dinesh Mandalapu and Sriram Subramanian. 2011. Exploring Pressure as an Alternative to Multi-Touch Based Interaction. *IndiaHCI* (2011), 88. DOI: <http://dx.doi.org/10.1145/2407796.2407810>
34. David C. McCallum, Edward Mak, Pourang Irani, and Sriram Subramanian. 2009. PressureText: pressure input for mobile phone text entry. *CHI* (2009), 4519. DOI: <http://dx.doi.org/10.1145/1520340.1520693>
35. Sachi Mizobuchi, Shinya Terasaki, Turo Keski-Jaskari, Jari Nousiainen, Matti Rynanen, and Miika Silfverberg. 2005. Making an impression: force-controlled pen input for handheld devices. *CHI EA* (2005), 1661. DOI: <http://dx.doi.org/10.1145/1056808.1056991>
36. Simon Olberding, Kian Peen Yeo, Suranga Nanayakkara, and Jurgen Steimle. 2013. AugmentedForearm: exploring the design space of a display-enhanced forearm. In *AH*. ACM Press, New York, New York, USA, 9–12. DOI: <http://dx.doi.org/10.1145/2459236.2459239>
37. Patrick Parzer, Kathrin Probst, Teo Babic, Christian Rendl, Anita Vogl, Alex Olwal, and Michael Haller. 2016. FlexTiles: A Flexible, Stretchable, Formable, Pressure-Sensitive, Tactile Input Sensor. In *CHI EA*. ACM Press, New York, New York, USA, 3754–3757. DOI: <http://dx.doi.org/10.1145/2851581.2890253>
38. Jerome Pasquero, Scott J. Stobbe, and Noel Stonehouse. 2011. A haptic wristwatch for eyes-free interactions. In *CHI*. ACM Press, New York, New York, USA, 3257. DOI: <http://dx.doi.org/10.1145/1978942.1979425>
39. F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* 12 (2011), 2825–2830.
40. Ivan Poupyrev, Nan-Wei Gong, Shiho Fukuhara, Mustafa Emre Karagozler, Carsten Schwesig, and Karen E. Robinson. 2016. Project Jacquard: Interactive Digital Textiles at Scale. In *CHI*. ACM Press, New York, New York, USA, 4216–4227. DOI: <http://dx.doi.org/10.1145/2858036.2858176>
41. Philip Quinn and Andy Cockburn. 2009. Zoofing!: faster list selections with pressure-zoom-flick-scrolling. In *OzCHI*. 185. DOI: <http://dx.doi.org/10.1145/1738826.1738856>
42. Gonzalo Ramos and Ravin Balakrishnan. 2005. Zliding: fluid zooming and sliding for high precision parameter manipulation. In *UIST*. 143. DOI: <http://dx.doi.org/10.1145/1095034.1095059>
43. Gonzalo Ramos, Matthew Boulos, and Ravin Balakrishnan. 2004. Pressure widgets. In *CHI*, Vol. 6. 487–494. DOI: <http://dx.doi.org/10.1145/985692.985754>
44. Christian Rendl, Patrick Greindl, Kathrin Probst, Martin Behrens, and Michael Haller. 2014a. Presstures: exploring pressure-sensitive multi-touch gestures on trackpads. In *CHI*. 431–434. DOI: <http://dx.doi.org/10.1145/2556288.2557146>
45. Christian Rendl, David Kim, Sean Fanello, Patrick Parzer, Christoph Rhemann, Jonathan Taylor, Martin Zirkel, Gregor Scheipl, Thomas Rothländer, Michael Haller, and Shahram Izadi. 2014b. FlexSense: a transparent self-sensing deformable surface. In *UIST*. 129–138. DOI: <http://dx.doi.org/10.1145/2642918.2647405>
46. Ilya Rosenberg and Ken Perlin. 2009. The UnMousePad: the future of touch sensing. In *ACM Transactions on Graphics*, Vol. 28. ACM Press, New York, New York, USA, 1. DOI: <http://dx.doi.org/10.1145/1531326.1531371>
47. T. Scott Saponas, Chris Harrison, and Hrvoje Benko. 2011. PocketTouch: through-fabric capacitive touch input. In *UIST*. ACM Press, New York, New York, USA, 303. DOI: <http://dx.doi.org/10.1145/2047196.2047235>
48. T. Scott Saponas, Desney S. Tan, Dan Morris, Ravin Balakrishnan, Jim Turner, and James A. Landay. 2009. Enabling always-available input with muscle-computer interfaces. In *UIST*. ACM Press, New York, New York, USA, 167. DOI: <http://dx.doi.org/10.1145/1622176.1622208>
49. Toshiaki Sato, Haruko Mamiya, Hideki Koike, and Kentaro Fukuchi. 2009. PhotoelasticTouch: transparent rubbery tangible interface using an LCD and photoelasticity. In *UIST*. ACM Press, New York, New York, USA, 43. DOI: <http://dx.doi.org/10.1145/1622176.1622185>
50. Mika Satomi and Hannah Perner-Wilson. 2016. How To Get What You Want. (2016). <http://www.kobakant.at/DIY/?cat=24>
51. R. S. Saxena, R. K. Bhan, and Anita Aggrawal. 2009. A

- new discrete circuit for readout of resistive sensor arrays. *Sensors and Actuators, A: Physical* 149, 1 (2009), 93–99. DOI:<http://dx.doi.org/10.1016/j.sna.2008.10.013>
52. Raghvendra Sahai Saxena, R. K. Bhan, Navneet Kaur Saini, and R. Muralidharan. 2011a. Virtual ground technique for crosstalk suppression in networked resistive sensors. *IEEE Sensors Journal* 11, 2 (feb 2011), 432–433. DOI:<http://dx.doi.org/10.1109/JSEN.2010.2060186>
  53. Raghvendra Sahai Saxena, Navneet Kaur Saini, and R. K. Bhan. 2011b. Analysis of crosstalk in networked arrays of resistive sensors. *IEEE Sensors Journal* 11, 4 (2011), 920–924. DOI:<http://dx.doi.org/10.1109/JSEN.2010.2063699>
  54. Stefan Schneegass and Alexandra Voit. 2016. GestureSleeve: using touch sensitive fabrics for gestural input on the forearm for controlling smartwatches. In *ISWC*. ACM Press, New York, New York, USA, 108–115. DOI:<http://dx.doi.org/10.1145/2971763.2971797>
  55. Carsten Schwesig, Ivan Poupyrev, and Eijiro Mori. 2003. Gummi: user interface for deformable computers. In *CHI EA*. 954. DOI:<http://dx.doi.org/10.1145/765891.766091>
  56. M. Shimojo, M. Ishikawa, and K. Kanaya. 1991. A flexible high resolution tactile imager with video signal output. In *ICRA*. 384–391. DOI:<http://dx.doi.org/10.1109/ROBOT.1991.131607>
  57. Lin Shu, Xiaoming Tao, and David Dagan Feng. 2015. A New Approach for Readout of Resistive Sensor Arrays for Wearable Electronic Applications. *IEEE Sensors Journal* 15, 1 (2015), 442–452. DOI:<http://dx.doi.org/10.1109/JSEN.2014.2333518>
  58. Craig Stewart, Michael Rohs, Sven Kratz, and Georg Essl. 2010. Characteristics of Pressure-Based Input for Mobile Devices. In *CHI*. ACM Press, New York, New York, USA, 801–810. DOI:<http://dx.doi.org/10.1145/1753326.1753444>
  59. Mathias Sundholm, Jingyuan Cheng, Bo Zhou, Akash Sethi, and Paul Lukowicz. 2014. Smart-mat: recognizing and counting gym exercises with low-cost resistive pressure sensing matrix. In *UbiComp*. ACM Press, New York, New York, USA, 373–382. DOI:<http://dx.doi.org/10.1145/2632048.2636088>
  60. Stuart Taylor, Cem Keskin, Otmar Hilliges, Shahram Izadi, and John Helmes. 2014. Type-hover-swipe in 96 bytes: a motion sensing mechanical keyboard. *CHI* (2014), 1695–1704. DOI:<http://dx.doi.org/10.1145/2556288.2557030>
  61. Martijn ten Bhömer and Pauline van Dongen. 2014. Vigour. *Interactions* 21, 5 (Sep 2014), 12–13. DOI:<http://dx.doi.org/10.1145/2641396>
  62. Giovanni Maria Troiano, Esben Warming Pedersen, and Kasper Hornbæk. 2014. User-defined gestures for elastic, deformable displays. *AVI* (2014), 1–8. DOI:<http://dx.doi.org/10.1145/2598153.2598184>
  63. Nirzaree Vadgama and Jürgen Steimle. 2017. Flexy: Shape-Customizable, Single-Layer, Inkjet Printable Patterns for 1D and 2D Flex Sensing. In *TEI*. ACM Press, New York, New York, USA, 153–162. DOI:<http://dx.doi.org/10.1145/3024969.3024989>
  64. Martin Weigel, Tong Lu, Gilles Bailly, Antti Oulasvirta, Carmel Majidi, and Jürgen Steimle. 2015. iSkin: Flexible, Stretchable and Visually Customizable On-Body Touch Sensors for Mobile Computing. In *CHI*. ACM Press, New York, New York, USA, 2991–3000. DOI:<http://dx.doi.org/10.1145/2702123.2702391>
  65. Martin Weigel, Vikram Mehta, and Jürgen Steimle. 2014. More than touch: understanding how people use skin as an input surface for mobile computing. In *CHI*. ACM Press, New York, New York, USA, 179–188. DOI:<http://dx.doi.org/10.1145/2556288.2557239>
  66. Mark Weiser. 1995. Human-computer Interaction. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, Chapter The Computer for the 21st Century, 933–940. <http://dl.acm.org/citation.cfm?id=212925.213017>
  67. G Wilson. 2013. Using Pressure Input and Thermal Feedback to Broaden Haptic Interaction with Mobile Devices. In *PhD Thesis*. 1–273. [http://encore.lib.gla.ac.uk/iii/encore/record/C\\_\\_Rb2982123](http://encore.lib.gla.ac.uk/iii/encore/record/C__Rb2982123)
  68. Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. 2009. User-defined gestures for surface computing. In *CHI*. ACM Press, New York, New York, USA, 1083. DOI:<http://dx.doi.org/10.1145/1518701.1518866>
  69. Bo Zhou, Harald Koerger, Markus Wirth, Constantin Zwick, Christine Martindale, Heber Cruz, Bjoern Eskofier, and Paul Lukowicz. 2016a. Smart soccer shoe: monitoring foot-ball interaction with shoe integrated textile pressure sensor matrix. In *ISWC*. ACM Press, New York, New York, USA, 64–71. DOI:<http://dx.doi.org/10.1145/2971763.2971784>
  70. Bo Zhou, Mathias Sundholm, Jingyuan Cheng, Heber Cruz, and Paul Lukowicz. 2016b. Never kick leg day: A novel wearable approach to monitoring gym leg exercises. In *PerCom*. DOI:<http://dx.doi.org/10.1109/PERCOM.2016.7456520>