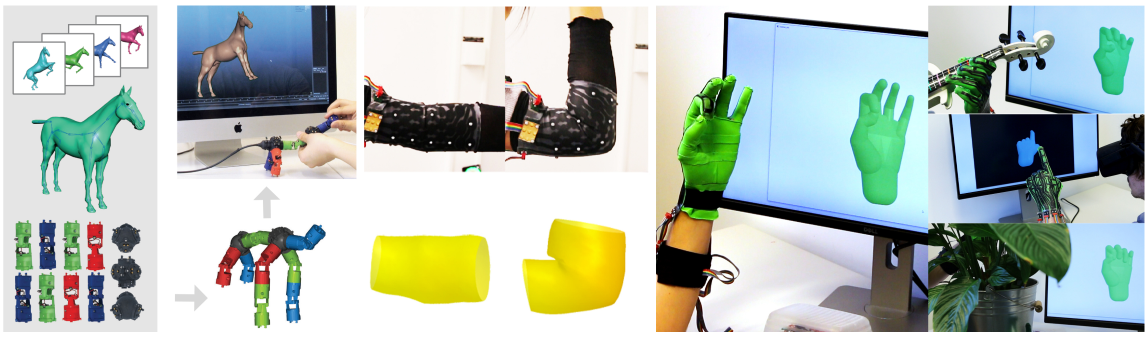
Diss. ETH No. 26332

# Self-sensing Devices for Motion and Deformation Capture

A thesis submitted to attain the degree of
**Doctor of Sciences** of **ETH Zurich**
(Dr. sc. ETH Zurich)



presented by
**Oliver Lucas Glauser**
MSc ETH ME, ETH Zurich, Switzerland
born on 15.02.1988
citizen of Switzerland

accepted on the recommendation of
**Prof. Dr. Olga Sorkine-Hornung**, examiner
**Prof. Dr. Otmar Hilliges**, co-examiner
**Dr. Robert Y. Wang**, co-examiner

2019

# Abstract

Capturing and modeling dynamic 3D shapes is a core problem in computer graphics and essential in many application areas. In this thesis, we present input devices and complementary algorithms to digitalize movement and deformation. We design our devices to be *self-sensing*, meaning that they rely on internal sensors only and therefore, neither require cameras nor any other external setup. This makes them very mobile and unaffected by issues typically problematic for vision-based systems such as light changes, fast motions, objects outside the field of view and above all, occlusions.

In the first part, we address the problem of articulated 3D character animation. We present a modular and tangible input device with embedded Hall effect sensors — in contrast to existing hardware solutions, our design is not prone to gimbal locking. We demonstrate in a user experiment that this leads to speedup by a factor of 2. Furthermore, we introduce an algorithm that deduces small and easily controllable input devices from professional rigs and a mapping from the devices' reduced degrees of freedom to the full ones of the unmodified rigs. We discuss a variety of animation results created with characters available online.

In the second part, we introduce a method to capture dense surface deformations without requiring line of sight. To that end, we propose a soft and stretchable sensor that measures its local area stretch densely. Moreover, we contribute a fabrication pipeline for such sensors, using only tools readily available in modern fablabs. The sensor concept and fabrication are verified in a series of controlled experiments. Finally, a wearable sensor prototype paired with a data-driven prior is employed to capture moving body parts like a wrist or an elbow and objects like an inflating and deflating balloon.

In the third part, we propose a glove for accurate hand pose estimation. It builds on the stretch sensor array concept introduced in the second part. The resulting glove features 44 sensors and is fully soft, stretchable and thin. We use a data-driven model that exploits the spatial layout of the sensor itself. The glove's abilities are demonstrated in a series of ablative experiments, exploring different models and calibration methods. In comparison with commercial options, ours achieves a 35% lower error.

# Zusammenfassung

Die Erfassung und das Modellieren dynamischer 3D-Formen ist ein zentrales Problem in der Computergrafik und in vielen Anwendungsbereichen unerlässlich. In dieser Arbeit stellen wir Eingabegeräte zur Digitalisierung von Bewegung und Verformung vor. Unsere Geräte sind *self-sensing*, was heisst, dass sie ausschließlich auf internen Sensoren basieren und daher weder Kameras noch andere externe Instrumentierung erfordern. Das macht sie sehr mobil und nicht anfällig für Umstände, die typischerweise problematisch sind für kamerabasierte Systeme, wie Lichtveränderungen, schnelle Bewegungen, Objekte ausserhalb des Sichtfeldes und, vor allem, verdeckte und sich selber verdeckende Objekte.

Im ersten Teil befassen wir uns mit dem Erstellen von artikulierten 3D-Charakteranimationen. Wir präsentieren ein modulares und greifbares Eingabegerät, das mit Hall-Effekt-Sensoren ausgestattet ist. Im Gegensatz zu bisherigen Hardwarelösungen ist unser Design nicht anfällig für Gimbal-Sperre. In einem Benutzerexperiment zeigen wir, dass dies im Vergleich doppelt so schnell zum Ergebnis führt. Darüber hinaus führen wir einen Algorithmus ein, der handliche und leicht kontrollierbare Eingabegeräte von professionellen Rigs ableitet und eine Zuordnung der reduzierten Freiheitsgrade der Geräte zu den vollständigen Freiheitsgraden der nicht modifizierten Rigs vornimmt. Wir zeigen und diskutieren eine Auswahl an Animationsergebnissen, die mit online verfügbaren Charakteren erstellt wurden.

Im zweiten Teil führen wir eine Methode zum Erfassen dichter Oberflächenverformung ohne Sichtlinie ein. Zu diesem Zweck schlagen wir einen weichen, dehnbaren Sensor vor, der die eigene Flächendehnung dicht messen kann. Darüber hinaus stellen wir eine Herstellungsmethode für solche Sensoren vor, bei der nur Werkzeuge verwendet werden, die in modernen Fablabs verfügbar sind. Das Konzept und die Herstellung des Flächendehnungssensors werden in einer Reihe kontrollierter Experimente verifiziert. Schliesslich wird ein tragbarer Sensorprototyp, in Kombination mit einem datengetriebenen geometrischen Modell, verwendet, um sich bewegende Körperteile wie ein Handgelenk oder einen Ellbogen und Objekte wie einen Ballon zu erfassen.

Im dritten Teil entwickeln wir einen Handschuh zur kontinuierlichen und genauen Erfassung von Handposen. Er basiert auf dem im zweiten Teil vorgestellten

Flächendehnungssensor-Konzept. Der Handschuh verfügt über 44 Sensoren und ist vollständig weich, dehnbar und dünn. Wir verwenden ein datengetriebenes Modell, das die räumliche Anordnung der Sensorzellen ausnutzt. Die Eignung des Handschuhs wird in einer Reihe von ablativen Experimenten demonstriert, in denen verschiedene Modelle und Kalibrierungsmethoden untersucht werden. Im Vergleich zu kommerziellen Optionen erzielen wir einen um 35% niedrigeren Fehler.

# Acknowledgments

I'm most grateful to Olga Sorkine-Hornung for convincing me to pursue a PhD in the first place and to have her as an advisor and for the privilege to do it at the amazing Interactive Geometry Lab at ETH Zurich. It was great to always have all the support and resources needed, to feel Olga's enthusiasm and her trust in me and to benefit from her broad scientific expertise and experience.

Just as important were my co-supervisors Otmar Hilliges and Daniele Panozzo. They were crucial by suggesting exciting and challenging research directions, in constructive discussions and in contributing innovative solution proposals to the questions at hand.

I would also like to thank Robert Y. Wang for accepting to be in my PhD committee and devoting his valuable time to attend my defense.

I was very lucky to have the chance to collaborate with Wan-Chun Alex Ma and Shihao Wu. With their devotion to our projects and their positive spirit, they were most essential in making this work possible and they also became good friends.

It is only because of my Master's thesis that I got in touch with Computer Graphics in the first place and thanks to Cédric Pradalier and Alec Jacobson this first encounter was exciting and a lot of fun, and therefore made me consider continuing in this direction.

Besides the directly involved collaborators, many other people were vital in making this thesis possible with their support, insightful discussions, their expertise, giving a helping hand in fabrication and in many other ways, to name just a few: Christine de St. Aubin, Olga Diamanti, Raoul Hopf, Severin Klingler, Roi Poranne, Samuel Rosset, Michael Rabinovich, Riccardo Roveri, Christian Schüller, Herbert Shea, Evgeni Sorkine, Rafael Wampfler, Yifan Wang, Katja Wolff, Jeannine Wymann and Ji Xiabon.

I'm also very thankful to all members of IGL during my time for the collaborative and very friendly work atmosphere. I'm especially grateful to Marianna Berger for the administrative support, for patiently dealing with the many packages with hardware parts that arrived for me.

It would have been much harder without the very enjoyable coffee and lunch breaks with all the inspiring and openminded friends and colleagues (IGL, CGL, CVG, DRZ, ...) from all over the world. They provided a diversion from the hurdles in research (that at this very moment, sometimes seemed unsolvable) and always included very engaging and interesting discussions on all kind of topics.

Finally, I would like to thank my friends and family for providing assistance, motivation, and many adventures together during this time. And most importantly, I could never have done any of this without the unconditional love and support from my partner in everything: Jeannine Wymann.

# Contents

# CHAPTER 1

# Introduction

The digital acquisition of shape deformation is one of the key research topics in computer graphics, geometry processing and beyond. Modeling and analysis of moving and deforming shapes are critical in a large variety of application areas, such as medicine and the life sciences, product design and engineering, and last but not least in film and games industries.

Especially for applications related to film and games, animated virtual objects or characters are, instead of being captured, often created by manual manipulation of digital deformation handles and motion curves using modeling/animation software, via a mouse and keyboard interface. But traditional animation UIs require substantial training and user expertise, in particular, due to cognitively demanding mapping between the 2D mouse interface and the deformation degrees of freedom.

For capturing actual physical 3D shapes laser and structured light scanners provide increasingly accurate results and become ever more affordable. However, capturing continuously moving and deforming shapes is much more challenging due to the added time dimension and real-time requirements. For full human body capture, optical, marker-based systems have been extensively studied and seen commercial success. Also, marker-less approaches using multiple cameras have been proposed. Such professional-grade motion capture systems can achieve high accuracy but are expensive, lack portability, need for calibration, and are restricted to small capture volumes, confining such technology to labs and dedicated production companies. More affordable solutions based on single RGB or depth cameras have seen a recent, rapid evolution but are often data-driven and highly optimized for the task of recognizing human poses and do not generalize well to other anatomies or shapes. And they still suffer from issues such as the need for an externally mounted camera, and situations typically problematic for vision-based systems, such as light-changes, objects outside the field of view and above all, (self-)occlusions.

In this thesis, we explore self-sensing input devices and complementary algorithms that do not rely on a vision-based setup overcoming the outlined disadvantages. Ideally, they should allow us to model and capture moving and deforming shapes and achieve the required precision for professional use but also provide generality and ease-of-use to make them accessible to a wider audience. The research challenges are the mechanical and electronic design to enable these goals, and new algorithms to bridge the gap between the sensor output and the digital object representation.

Two modes for deformation capture are studied: (i) via skeletal movement and pose capture using a tangible, modular input device, and (ii) via surface deformation capture by a dense map of area stretch sensors. The former is a complete redesign of our previous modular input device [Jacobson et al. 2014b]. Specifically, we introduce a hardware design, overcoming major limitations in prior work via a novel physical angle parametrization. And we derive an algorithm to compute compact device configurations. The latter is a new type of wearable sensors specifically targeted at the dense capture of physical deforming 3D shapes. And, we show how the developed sensor technology can be adapted and used to interactively capture accurate skeletal hand poses when combined with a data-driven model. In the following, we give short summaries for each of these three parts and what the concrete contributions are.

**Modular Input Device for Rig Animation.** We introduce a novel approach to digital character animation, combining the benefits of tangible input devices and sophisticated rig animation algorithms. The symbiotic software and hardware approach facilitates the animation process for novice and expert users alike. Our algorithm derives a small device configuration from complex character rigs, often containing hundreds of degrees of freedom, and a set of sparse sample poses. Importantly, only the most influential degrees of freedom are controlled directly, yet detailed motion is preserved based on a pose interpolation technique. We design a modular collection of joints and splitters, which can be assembled to represent a wide variety of skeletons. Each joint piece combines a universal joint and two twisting elements, allowing it to accurately sense its configuration.

**Contributions.** We overcome limitations inherent to all previous tangible devices by allowing users to directly control complex rigs using only a small set (5-10) of physical controls. This avoids oversimplification of the pose space and excessively bulky device configurations. Our mechanical design provides a smooth inverse kinematics-like user experience and is not prone to gimbal locking. Comparative user experiments show significant improvements over the closest state-of-the-art in terms of accuracy and time in a keyframe posing task.

**Stretch Sensor Arrays for Deformation Capture.** We introduce a hardware and software pipeline to fabricate wearable sensors and use them to capture deformations without a line of sight. Our first contribution is a low-cost fabrication pipeline to embed multiple aligned conductive layers with complex geometries into silicone compounds. Overlapping conductive areas from separate layers form local capacitors that measure dense area changes. Contrary to existing fabrication methods, the proposed technique only requires hardware that is readily available in modern fablabs. While area measurements alone are not enough to reconstruct the full 3D deformation of a surface, they become sufficient when paired with a data-driven prior. The novel semi-automatic tracking algorithm, based on an elastic surface geometry deformation, allows capturing ground-truth data with an optical mocap system, even under heavy occlusions or partially unobservable markers. The resulting dataset is used to train a regressor based on deep neural networks, directly mapping the area readings to global positions of surface vertices.

**Contributions.** We contribute a novel concept, stretchable sensor arrays, for measuring stretch densely, including a low-cost fabrication pipeline to fabricate such sensors. We demonstrate how such a sensor combined with a data-driven prior can be applied to interactively estimate dense surface deformation. To be able to capture high-quality ground-truth data with an optical motion capture system, even under heavy occlusions or partially unobservable markers, we introduce a semi-automatic tracking algorithm, based on an elastic surface geometry deformation.

**Hand Pose Estimation with a Stretch Glove.** We propose stretch-sensing soft glove to interactively capture hand poses with high accuracy and without requiring an external optical setup. We explain how the device can be fabricated and calibrated at a low cost. We introduce a data-driven model that only requires a short per-user calibration that is performed on-the-fly using just the glove. The model is trained only once, using an inexpensive off-the-shelf hand pose reconstruction system to gather the training data. The glove's capabilities are demonstrated in a series of ablative experiments, exploring different models and calibration methods.

**Contributions.** We demonstrate how the stretchable sensor arrays, in combination with a custom textile part, can be made into thin, fully soft data gloves. Furthermore, we suggest and validate a data-driven model exploiting the spatial layout of the sensor itself. Our data glove has as many as 44 sensors embedded, more sensor than any other data glove before. Comparing against commercial data gloves, it achieves a 35% improvement in reconstruction accuracy.

## 1.1 Thesis outline

The dissertation is divided into six chapters. The remaining chapters are organized as follows.

**Chapter 2** presents related works, subdivided into three parts: In the first part, works related to the introduced tangible and modular input device are presented. Then, literature with respect to stretchable sensor arrays is reviewed. Finally, we give an overview of research on capturing hand poses with a focus on wearable data gloves.

**Chapter 3** presents the novel approach to digital character animation, combining the benefits of tangible input devices and sophisticated rig animation algorithms. We introduce a modular hardware design not prone to gimbal locking and an algorithm that derives a small device configuration from complex character rigs.

**Chapter 4** presents the hardware and software pipeline to fabricate flexible wearable sensors and how we use them to capture deformations without line of sight. We introduce a capacitive sensor concept that allows capturing dense area stretch.

**Chapter 5** presents our stretch-sensing soft glove to interactively capture hand poses with high accuracy and without requiring an external optical setup. We show how the sensor arrays introduced in Chapter 4 can be turned into a low-cost data glove.

**Chapter 6** summarizes our contributions and discusses interesting avenues for future work.

**Appendix A** provides additional details related to Chapter 3 on the Fault-resistant distributed protocol, Evaluating Pose Reachability and the User study.

**Appendix B** provides additional details related to Chapters 4 and 5 on the Silicone Mixtures and the Measurement setup.

**Appendix C** provides additional details related to Chapter 5 on the Network Architecture and the Interconnections.

## 1.2 Publications

The work on this thesis resulted in the following peer-reviewed publications:

**[Glauser et al. 2016a]** O. Glauser, W.-C. Ma, D. Panozzo, A. Jacobson, O. Hilliges, and O. Sorkine-Hornung. Rig Animation with a Tangible and Modular Input Device. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, Jul 2016a

**[Glauser et al. 2019a]** O. Glauser, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. Deformation capture via soft and stretchable sensor arrays. *ACM Transactions on Graphics*, Mar 2019a

**[Glauser et al. 2019b]** O. Glauser, S. Wu, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. Interactive hand pose estimation using a stretch-sensing soft glove. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, Jul 2019b

We also presented the outcomes at scientific conferences as juried demo, video and exhibition:

**[Glauser et al. 2016b]** O. Glauser, B. Vartok, W.-C. Ma, D. Panozzo, A. Jacobson, O. Hilliges, and O. Sorkine-Hornung. Rig animation with a tangible and modular input device. In *UIST '16 Adjunct Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, Oct 2016b

**[Glauser et al. 2018]** O. Glauser, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. Deformation capture via self-sensing capacitive arrays (video). In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018

**[Glauser et al. 2019c]** O. Glauser, S. Wu, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. A stretch-sensing soft glove for interactive hand pose estimation. In *ACM SIGGRAPH 2019 Emerging Technologies*, Jul 2019c

During the course of this thesis, the following peer-reviewed papers were also published:

**[Jacobson et al. 2014b]** A. Jacobson, D. Panozzo, O. Glauser, C. Pradalier, O. Hilliges, and O. Sorkine-Hornung. Tangible and modular input device for character articulation. *ACM Trans. Graph.*, Jul 2014b

**[Wolff et al. 2018]** K. Wolff, R. Poranne, O. Glauser, and O. Sorkine-Hornung. Packable springs. *Computer Graphics Forum*, May 2018

CHAPTER 2

# Related Work

This thesis relates to several areas of literature ranging from character rig animation to dense motion capture modalities to interactive hand pose capture. For clarity, the related work section consists of three parts.

In the first part, work in the fields of inverse kinematics, pose and rig space, input devices, motion capture and retargeting and skeleton simplification with direct implications on our suggested tangible and modular input device are presented. In the second part, the most important literature concerning dense deformation captures and the stretchable sensor arrays are reviewed. This spans from camera-based motion capture, self-sensing input devices, IMU base methods, strain and bend sensors, to sensor fabrication. In the third part, we give an overview of research on capturing hand poses: From vision-based approaches to wearables using orientation, bend or strain sensors to data glove calibration methods.

## 2.1 Modular Input Device for Rig Animation

Real-time character articulation is a core subject of computer graphics, receiving much attention in the literature. We refer to a survey of skinning techniques [Jacobson et al. 2014a], and turn our attention to methods of reducing deformation control, and specifically to those assuming a skeletal rig. We discuss comparisons to existing joint hardware designs in Section 3.2.1.

**Inverse kinematics.** The well-studied problem of reducing the control of a complicated *rig* to sparse user input is typically approached by trying to distill high-level controls from a given set of low-level controls. For example, traditional inverse kinematics (IK) abstracts an arbitrarily complicated kinematic tree of rotational or translational joints into just the position of leaf nodes in the tree: end-effectors in robotics, or more saliently to character animation, the head, hands, and feet.

Classic IK ignores the effect of the kinematic tree's implied deformation of the character's surface geometry. So-called MeshIK approaches remedy this by replacing the usual "joint work" with a geometric energy capturing the quality or physical plausibility of the deforming surface geometry [Sumner et al. 2005]. MeshIK (and all mesh-based deformation methods [Botsch and Sorkine 2008]) reduce control from the individual mesh vertex level to the placement of a few user-specified handles. Under the framework of linear blend skinning, MeshIK can achieve real-time performance [Jacobson et al. 2012]. Given sparse user constraints, skeletal skinning transformations are optimized according to their impact on the shape's surface.

Although the user effort is reduced by the above techniques, so is the space of possible deformations: linear blend skinning cannot feasibly represent high-frequency effects such as muscle bulging.

**Pose and rig space.** Pose space interpolation [Buck et al. 2000; Lewis et al. 2000] has been successfully applied to character animation. These example based techniques leverage both the desired shapes, sculpted by trained modeling artists, and the abstract *pose space* which is formed by the rig controls. In particular, deformations may be interpolated and extrapolated according to associated lower-dimensional abstract poses. We take advantage of such pose space interpolation techniques, specifically [Buck et al. 2000], to animate the rig with the reduced degrees of freedom from our input device, since it is impossible to directly control a complex rig with hundreds of bones through our modular input device.

Driving an arbitrary character rig has also been considered in the context of physical simulation. For example, a physical simulation's deformation of a character's geometry can be projected onto a generic rig [Hahn et al. 2012]. Rather than mapping sparse user controls, this maps a potentially very dense physical simulation to a dense rig. Though distinct in motivation, our rig reduction bares similarity to [Hahn et al. 2012], but we require only rig evaluation without resorting to finite differencing.

**Input devices.** Despite their ubiquity, 2D mouse and discrete keyboard interfaces struggle to control multi-modal 3D entities such as rotations or translations [Jacob et al. 1994]. Previous works have overcome this via sketching [Öztireli et al. 2013; Hahn et al. 2015; Guay et al. 2013, 2015], but manipulation is still indirect through a projection onto 2D device coordinates. Following [Ishii and Ullmer 1997], recent physical input devices demonstrate technologies for direct manipulation. The software of [Jacobson et al. 2014b] does not consider gross mismatches between the virtual character's skeleton and their modular input device. Crucially, it assumes a simple skeleton and a matching input device that has as many physical joints as virtual bones in the skeleton. In reality, rigs are very complicated, causing physical configurations to grow too bulky quickly. Furthermore, determining an appropriate configuration of components is far from trivial. The "dinosaur input device" of [Knep et al. 1995] also had to address the problem of having too many rig parameters for a given input device. Their solution was to map a short sequence of one-dimensional measurements from the input device (say, along a T-rex's tail) to a chain of rotations along a virtual bone chain. In contrast, our proposed mapping is widely general. Our design integrates rotational degrees of freedom (DoFs) per joint but still allows to separate translation and rotation, which is preferable and in accordance with prior findings ([Zhai and Milgram 1998; Masliah and Milgram 2000]).

The recent flexible bending input devices of [Chien et al. 2015] and [Nakagaki et al. 2015] consist of a chain of single DoF elements. They do not directly offer the degrees of freedom needed for character animation, since they do not support axial rotational DoFs and lack the modularity required to control a large variety of characters.

Other physical input systems exist, such as computer vision systems for 6-DoF tracking [Held et al. 2012; Shiratori et al. 2013] and specialized dolls for humans and hands [Esposito et al. 1995; Feng et al. 2008; Yoshizaki et al. 2011; Celsys, Inc. 2013; Achibet et al. 2015]. See [Jacobson et al. 2014b] for a complete history of tangible animation input devices.

**Motion retargeting.** The human body itself becomes an input device via performance capture. The mapping of an actor's performance to a character can be non-trivial in the presence of proportional disparities (a short actor controlling a giant [Gleicher 1998]) or gross differences in the source and target (a human actor controlling a sheep [Rhodin et al. 2014; Fender et al. 2015]). However, the input device is fixed and known: hands are always connected to arms connected to torsos, etc.

The more general problem of animation retargeting considers mapping the deformation of one object onto a similar object [Sumner and Popović 2004; Baran et al. 2009]. Most relevant to our work is the recent effort to map the deformation of a simplified skeleton (e.g. from a control optimization) to a character controlled by a complex rig [Holden et al. 2015]. Similarly, [Jin et al. 2015] consider mapping skeleton deformations to a sub-skeleton or vice-versa. In contrast, our work optimizes (Sec. 3.2.2) a quality metric for an input device to rig mapping as means to *propose* a particular input device configuration. That is, the very set of control parameters is determined by our optimization.

**Skeleton simplification.** To propose a modular device configuration, we rely on the ability to reduce a combinatorial tree of rig deformers to a simple geometric skeleton closely matching a constellation of device components. Pure skeleton simplification has been considered previously in the context of level-of-detail animations [Ahn and Wohn 2004; Savoye and Meyer 2008] and curved skeleton extraction [Au et al. 2008; Tagliasacchi et al. 2012]. These methods either assume a skeleton or rely on analysis of the shape's surface geometry. We consider rigs composed of arbitrary deformers, some of which have no associated geometric "bone". We avoid relying too heavily on specific character surface geometry, as it could change during animation prototyping or may simply be unavailable. Instead, in the spirit of pose space deformation [Lewis et al. 2000], our optimization assumes a small set of sample *rig* poses.

Example-based methods are a popular means to solving a problem that is in some sense the inverse of ours: Given a full animation of a character's geometry, determine a skeletal rig [Schaefer and Yuksel 2007; Le and Deng 2014]. [Huang 2015] recently demonstrated automatic generation of plausible example poses. Our method would immediately benefit from such example poses.

## 2.2 Stretch Sensor Arrays for Deformation Capture

The stretchable sensor arrays relate to several areas of literature ranging from digital animation, fabrication to motion capture and self-sensing input devices. We briefly review the most important work in these areas.

**Camera-based motion capture.** The acquisition of articulated human motion using cameras is widely used in graphics and other application domains. Commercial solutions require wearing marker suits or gloves and depend on multiple calibrated cameras mounted in the environment. To overcome these constraints, research has proposed marker-less approaches using multiple cameras (cf. [Moeslund et al. 2006]); sometimes these rely on offline [Bregler and Malik 1998; Ballan et al. 2012; Starck and Hilton 2003] and more recently online processing [Rhodin et al. 2015; de Aguiar et al. 2008; Stoll et al. 2011; Elhayek et al. 2017], but always require fixed camera installations. Neumann et al al. [Neumann et al. 2013] capture muscle deformations of a human shoulder and arm with a multi-camera system and derive a data-driven statistical model.

Recent pose estimation methods exploit deep convolutional networks for body-part detection in single, fully unconstrained images [Chen and Yuille 2014; Newell et al. 2016; Tompson et al. 2014b; Toshev and Szegedy 2014; Wei et al. 2016]. However, these methods only capture 2D skeletal information. Predicting 3D poses directly from 2D RGB images has been demonstrated using offline methods [Bogo et al. 2016; Tekin et al. 2016; Zhou et al. 2016] and in online settings [Mehta et al. 2017]. Monocular *depth* cameras provide additional information and have been shown to aid robust skeletal tracking [Ganapathi et al. 2012; Ma and Wu 2014; Taylor et al. 2012; Shotton et al. 2013; Taylor et al. 2016] and enable dense surface reconstruction even under deformation [Zollhöfer et al. 2014; Newcombe et al. 2015; Dou et al. 2016]. Multiple, specialized structured light scanners can be used to capture high-fidelity dense surface reconstructions of humans [Pons-Moll et al. 2015]. For a discussion of camera-based hand tracking methods, see Sec. 2.3.

All vision-based approaches struggle with visual clutter, (self-)occlusions and difficult lighting conditions, such as bright sunshine in the case of depth cameras, high contrast or lack of illumination in the case of color cameras. Furthermore, all camera based systems require line-of-sight and often precise calibration, and are therefore not well suited in many scenarios, such as outdoors. Our sensor is a first step in removing these limitations, allowing mobile and self-contained sensing, without line of sight.

**Self-sensing input devices.** A key feature of our methods is the capability of measuring the sensor's own deformation without requiring any external cameras. Such *self-sensing* input devices, usually not designed for motion capture, have been first demonstrated in the Gummi system [Schwesig et al. 2004], which simulated a handheld, flexible display via two resistive pressure sensors. Other early work used the ShapeTape sensor [Danisch et al. 1999] for input into a 3D modeling application [Balakrishnan et al. 1999]. Metallic strain gauges embedded into flexible 3D printed 1D strips measure the bending and flexing of custom input devices [Chien et al. 2015]. Rendl et al. [Rendl et al.

2014] use eight transparent printed electrodes on a transparent and flexible 2D display overlay to reconstruct 2.5D bending and flexing of the sheet in real time, but do not allow for stretch. [Bächer et al. 2016] propose an optimization based algorithm to design self-sensing input devices by embedding piezoresistive polymer traces into flexible 3D printed objects. [Sarwar et al. 2017] use polyacrylamide electrodes embedded in silicone to produce a flexible, transparent $4 \times 4$ sensing grid, and [Xu et al. 2016] propose a PDMS based capacitive array; both are limited to detecting touch gestures. Hall effect sensors embedded into hot-pluggable and modular joints can measure joint angles of tangible input devices used for character animation [Jacobson et al. 2014b]. While demonstrating the rich interactive possibilities afforded by flexible input devices, none of the above approaches are directly suitable for the acquisition of dense non-rigid surface deformation.

**Inertial measurement units (IMUs).** Attaching sensors directly onto the body overcomes the need for line-of-sight and enables use without infrastructure. IMUs are the most prominent type of sensors used for pose estimation. Commercial systems rely on 17 or more IMUs, which fully constrain the pose space, to attain accurate *skeletal* reconstructions via inverse kinematics [Roetenberg et al. 2007]. Good performance can be achieved with fewer sensors by exploiting data-driven methods [Tautges et al. 2011; Liu et al. 2011; Schwarz et al. 2009] or taking temporal consistency into account, albeit at high computational cost and therefore requiring offline processing [von Marcard et al. 2017], and Huang et al. [Huang et al. 2018] use a bi-directional RNN to learn this mapping from synthetic data and reconstruct full-body poses in real time. While IMUs provide mobility and accuracy, they cannot sense dense surface deformations. For a discussion of data gloves making use of IMUs, see Sec. 2.3.

**Strain gauges, stretch and bend sensors.** Strain sensors fabricated from stretchable silicone and attached directly to the skin have been proposed to measure rotation angles of individual joints [Lee et al. 2016]. Shyr et al. [Shyr et al. 2014] propose a textile strain sensor, made from elastic conductive yarn, to acquire bending angles of elbow and knee movements. [Mattmann et al. 2008] and [Lorussi et al. 2004] use strain gauges embedded into garments to classify discrete body postures. Specifically designed for the capture of wrist motion, [Huang et al. 2017] use five dielectric elastomer sensors and achieve an accuracy of $5^\circ$ for all motion components, highlighting the difficulty of reconstructing joint orientation of complex, multi-axial joints such as the wrist, shoulder or ankle. Bending information can be used to recover articulated skeletal motion. Typically, resistive bend sensors are used. However, these suffer from hysteresis [Bächer et al. 2016]; imprecise placement and sensor slippage can impact accuracy [Kessler et al. 1995b]. A soft bend sensor that is insensitive to stretching and mountable directly on the user's skin is proposed in [Shen et al. 2016], increasing angular accuracy, but it is inherently limited to measuring uni-axial bending. In Sec. 2.3 we cover related works of data gloves applying these types of sensors.

We propose a wearable, soft and stretchable silicone-based capacitive sensor design, focused on measuring dense area changes, which allows us, in combination with a data-driven reconstruction technique, to accurately capture dense, articulated and non-rigid deformations, see Chapter 4.

**Fabrication.** Producing capacitive elastomer stretching sensors is challenging, and the mechanical, electrical and thermal properties all depend on the type of material used and the pattern of conductive traces or electrodes. Another challenge is that the silicone is hydrophobic, hence the adhesion of non-silicones is extremely difficult. For an extensive review of various ways to manufacture conductive layers for such sensors or actuators, we refer to [Rosset and Shea 2013]. Composites of carbon black (conductive powder) and silicone are widely used, see e.g. [Araromi et al. 2015; Rosset et al. 2016; Huang et al. 2017; O'Brien et al. 2014]. A large range of fabrication methods for manufacturing conductive trace patterns have been proposed. Most methods rely on the potentially costly fabrication of intermediate tools like screen printing masks [Jeong and Lim 2016; Wessely et al. 2016], molds [Huang et al. 2017; Sarwar et al. 2017] or stencils [Rosset et al. 2016]. To circumvent the adhesion issue, specialized plasma chambers are often required to selectively pre-treat the base layer [Jin et al. 2017]. An alternative procedure, introduced by [Lu et al. 2014], involves patterning conductive PDMS sheets, manually removing excess parts with tweezers, sealing the resulting circuit with PDMS and bonding multiple such circuit layers to form capacitive touch sensors (as demonstrated by [Weigel et al. 2015]). Similar to [Araromi et al. 2015], our process leverages a standard laser cutter to etch away the negative sensor pattern, opening up the possibility to digitally design electrode patterns and produce them with low error tolerance. However, in contrast to prior work, our fabrication method (Sec. 4.3.3) does not require a plasma chamber or manual alignment and gluing of the different layers. Hence it allows for the production of larger sensors with a high alignment quality (see Fig. 4.8). To the best of our knowledge, we are the first to propose a fabrication method that requires almost no specialized hardware and enables creating large high-resolution multi-layer sensor *arrays*.

**Capacitive (touch) sensing** Ever since the introduction of the Theremin [Glinsky 2000], an experimental musical instrument, researchers have explored the use of capacitive sensing in the context of HCI. Most notably, capacitive coupling effects are the basis of early [Beck and Stumpe 1973; Lee et al. 1985] and virtually all modern touchscreen devices [Rekimoto 2002]. Capacitive coupling effects exist naturally between many objects (including humans) and their surroundings, and by measuring the changes in relative values it is possible to recover relative position, proximity and other properties. The seminal works by Smith [Smith 1995] and Zimmermann et al. [Zimmerman et al. 1995] introduced and categorized the various electric field sensing aspects to the interaction research community and demonstrated applications that went well beyond binary touch detection. Since then capacitive coupling effects have been used to sense touch, detect and discriminate user grip and grasp, detect and track objects on interactive surfaces, track 3D positions and proximity and coarsely classify 3D poses and gestures. We refer to the survey by Grosse-Puppendahl et al. [Grosse-Puppendahl et al. 2017] for an exhaustive treatment. Notably, flexible and bendable sensors [Gotsch et al. 2016; Han et al. 2014; Poupyrev et al. 2016] and those directly worn on the user's skin [Weigel et al. 2015; Kao et al. 2016; Nittala et al. 2018] have been proposed. However, virtually all of the above work measures one or a combination of different capacitive coupling effects, that is, the change in capacitance due to a conductive object (such as a finger) approaching an electrode. Our work is fundamentally different in that we do not sense capacitive coupling effects but instead measure changes in the electrodes' properties themselves: under deformation, the area of the electrode's plates changes, which in turn changes the capacitance of the plate and hence the charge time of the capacitor. In Chapter 4, we show how this effect can be leveraged to recover, using appropriate geometric priors, detailed 3D surface deformations, albeit at the cost of requiring a custom read-out scheme.

## 2.3 Hand Pose Estimation with a Stretch Glove

The majority of hand pose reconstruction methods are based on either an external vision setup or a set of sensors embedded into a data glove. Most gloves employ sensors from three categories: IMUs (inertial measurement units), bend (flex) sensors, and strain (stretch) sensors. For a complete overview we refer to the surveys [Dipietro et al. 2008; Rashid and Hasan 2018]. Other work has used wrist worn IR cameras [Kim et al. 2012] or magnetic sensing [Chen et al. 2016] for hand pose estimation, and capacitive wrist bands [Truong et al. 2018] or electromyography (EMG) [Saponas et al. 2009] for gesture recoginition. In the following, we summarize the works most closely related to our data glove.

**Camera based hand tracking.** A variety of vision based approaches for the problem of hand pose estimation have been proposed in the computer vision and graphics literature (cf. [Erol et al. 2007]). Marker based MoCap approaches (e.g., Vicon [Vic 2019]) require multiple, calibrated cameras and, compared to the full-body case, marker occlusions are a more severe problem. In consequence, learning based approaches to marker labelling under occlusion have been proposed [Han et al. 2018]. However, the need for multiple cameras restricts the applicability of such approaches. Wang and Popović [Wang and Popović 2009] propose a marker-like glove, requiring only one RGB camera. With the widespread availability of consumer grade depth cameras, single sensor solutions have received intense attention [Sharp et al. 2015; Sun et al. 2015; Tagliasacchi et al. 2015; Tang et al. 2013, 2014, 2015; Taylor et al. 2016; Wan et al. 2016, 2017; Zhang et al. 2016]. Depth based approaches can be categorized into model fitting based methods (e.g., [Oikonomidis et al. 2011a; Tkach et al. 2016]) and per-frame classification [Sun et al. 2015; Tang et al. 2014; Wan et al. 2017; Tang et al. 2015]. Moreover, many hybrid approaches that initialize a coarse hand pose estimate via discriminative approaches and then refine this via minimization of some error functional have been proposed [Sridhar et al. 2013; Tkach et al. 2016; Taylor et al. 2016; Tkach et al. 2017; Taylor et al. 2017]. Others deploy convolutional neural networks (CNNs) to regress 3D hand poses from depth images [Oberweger et al. 2015a; Oberweger and Lepetit 2017; Sinha et al. 2016; Ge et al. 2017; Tang et al. 2014; Wan et al. 2017] or even from just a single RGB image [Simon et al. 2017; Spurr et al. 2018; Mueller et al. 2018; Cai et al. 2018; Zimmermann and Brox 2017].

In contrast to vision-based approaches, our work in Chapter 5 relies only on intrinsic sensor readings and, once trained, requires no additional external infrastructure, opening the door to usage scenarios where traditional motion capture approaches are not applicable.

**IMU sensor gloves.** IMUs consist of a 3-axis accelerometer, a 3-axis gyroscope and a 3-axis magnetometer. Gloves based on 15 [Fang et al. 2017], 16 [Connolly et al. 2018], or 18 [Lin et al. 2018] IMUs have been suggested to recover hand pose. One major drawback of IMUs in the context of hand pose estimation is their rigidity and bulkiness compared to the size of human fingers.

**Bend sensor gloves.** Bend (flex) sensors have been very successfully applied in commercial products like the CyberGlove [Cyb 2019], the VPL Glove [VPL 2019], the 5DT glove [5DT 2019] or the recent ManusVR glove [Man 2019], with the latter also employing two IMUs. [VPL 2019] and [5DT 2019] are equipped with optical flex sensors. Some glove designs leverage off-the-shelf flex sensors [Gentner and Classen 2008; Zheng et al. 2016; K Simone et al. 2007], whereas others focus on designing novel, soft bend sensors [Kramer et al. 2011; Shen et al. 2016; Ciotti et al. 2016]. Typically such gloves feature between 5 and 22 (CyberGlove) sensors, whereas the human hand has at least 25 DoFs. A larger amount of sensing elements is difficult to place and typically increases the complexity of the glove design and consequently the manufacturing cost (cf. CyberGlove [Cyb 2019]) and may hinder dexterous and natural hand movements. In contrast, our design consists of a single sheet of silicone composite, and the amount of sensing elements is only limited by the surface area and space for routing of connecting leads. We compare with two state-of-the-art gloves [Man 2019; Cyb 2019] in Sec. 5.4.

**Strain sensor gloves.** Elastic strain sensors have the potential to allow for very slim and comfortable gloves. Starting with [Lorussi et al. 2005], many different strain sensor gloves, glove parts, or novel sensors tailored for hand capture have been proposed. Most of the presented strain sensor gloves are resistive [O'Connor et al. 2017; Michaud et al. 2016; Hammond et al. 2014; Lorussi et al. 2005; Park et al. 2017; Ryu et al. 2018; Chossat et al. 2015], either using a piezoresistive material, an elastic conductive yarn or conductive liquid channels. Liquid sensors are superior in terms of hysteresis, but their fabrication is often highly involved. Gloves based on capacitive stretch sensors, similar to ours, [Atalay et al. 2017] or video demos by commercial stretch sensor manufacturers [Str 2019; Ban 2018], combine the advantages of a slim form factor, no hysteresis, and softness. At most 15 strain sensors are used for a full glove by [Park et al. 2017], including abduction sensors. This is still significantly less than the amount of DoFs of a full hand; therefore many of the suggested designs are only demonstrated in the context of gesture recognition [Ryu et al. 2018; O'Connor et al. 2017; Hammond et al. 2014; Lorussi et al. 2005] and are not suitable for continuous full hand pose estimation. Some works show pose capture of a part of the hand [Michaud et al. 2016; Park et al. 2017]. Only [Park et al. 2017] and [Chossat et al. 2015] (11 sensors) demonstrate the capture of a full hand, but without evaluating the resulting accuracy. Our sensor design incorporates almost three times as many strain sensors (Fig. 5.3), as the closest comparison and to the best of our knowledge, we are the first to demonstrate the feasibility of accurate, continuous reconstruction of full hand poses from strain sensors alone.

**Calibration.** To provide reasonable accuracy, appropriate calibration is crucial for data gloves [Kessler et al. 1995a], due to specific sensor characteristics and the large variations in shape of different hands. Calibration is often equivalent to finding model parameters, such as gain, offset, or adjusting cross-coupling effects of a custom hand deformation model. Min-max pose calibration [Menon et al. 2003], ground truth calibration [Chou et al. 2000], and inverse kinematics (IK) calibration [Griffin et al. 2000; Hu et al. 2004] are among the most common approaches. Wang and Neff [Wang and Neff 2013] elegantly combine all three calibration methods to build a Gaussian process regression model, allowing to reconstruct joint-angles with high accuracy. Menon et al. [Menon et al. 2003] and Chou et al. [Chou et al. 2000] fit a hand-sensor model to individual users. The work of [Menon et al. 2003] assumes specific joint angles for poses to be performed by the user, while [Chou et al. 2000] track a set of markers to overcome the fixed angle assumption. Kahlesz et al. [Kahlesz et al. 2004] and Steffen et al. [Steffen et al. 2011] introduce models mapping from several sensors to one pose parameter to reduce cross-coupling effects. Griffin et al. [Griffin et al. 2000] ask the user to pose the hand while the thumb and one fingertip touch, and fit model parameters by minimizing fingertip distances. Hu et al. [Hu et al. 2004] extend this method by applying a vision system, tracking fingertip positions, and Zhou et al. [Zhou et al. 2010] extract user-specific calibration parameters from a single image via a ANN. Fischer et al. [Fischer et al. 1998] use a neural network to learn a mapping from sensor readings to fingertip positions.

We propose a simple yet effective per-user calibration procedure: First, a non-personalized model is trained to map from sensor readings to pose parameters. For new hands, minimal and maximal capacitance values per sensor are captured and used to normalize sensor readings. Note that this is different from the classic min-max calibration, where specific joint-angles or poses are assumed to correspond to the min and max values (e.g., [Menon et al. 2003]). We discuss the calibration details in Sec. 5.4.

CHAPTER 3

# Modular Input Device for Rig Animation

## 3.1 Introduction

At the heart of interactive character animation lies a contradiction. Animators must draw from a large space of poses to breath life and depth into the character, but animators also wish to do so with a very small number of control parameters. Standard rigging tools offer sophisticated ways to span a large space of shape deformations, but their parameters are endless, and indirectly mapping them to the keyboard and mouse makes existing interfaces cumbersome and difficult to learn. Recent tangible input devices promise direct and natural manipulation, but at the cost of either grossly simplifying the pose space or of accepting complex and bulky physical setups. In contrast, we present a novel software/hardware approach co-designed to help animators traverse a large space of poses via *fluid* manipulation of a tangible controller.

Specifically, we contribute: (*a*) a novel hardware design, overcoming major limitations in prior work via a novel physical angle parametrization; (*b*) an algorithm to compute a device configuration and instructions to assemble it, using only a small set of modules; and (*c*) a method to bridge the disparity between the input device's *few* degrees of freedom to the character's *many* control parameters. Our contribution allows users to move beyond static keyframing towards fluid animation of a variety of complex characters with arbitrary topologies.

The approach is general and can directly control professional-grade rigs

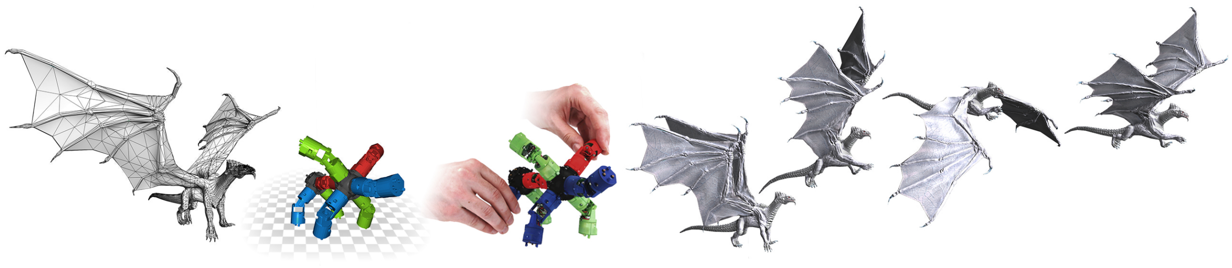


**Figure 3.1:** *Left to right: Taking a rigged 3D character with many degrees of freedom as input, we propose a method to automatically compute assembly instructions for a modular tangible controller, consisting only of a small set of joints. A novel hardware joint parametrization provides a user-experience akin to inverse kinematics. After assembly the device is bound to the rig and enables animators to traverse a large space of poses via fluid manipulations. Here we control 110 bones in the dragon character with only 8 physical joints and 2 splitters. Detailed pose nuances are preserved by a real time pose interpolation strategy.*

without relying on a specific shape deformation algorithm. It only relies on a rig with a skeletal structure and a set of sample poses as input. From this input rig, the most important control parameters are extracted. Using a given hardware kit, the algorithm then computes a device configuration and a mapping of physical rotations onto the extracted rig control parameters. The algorithm is guided by an objective functional that measures reachability in a set of sparse sample poses. Manipulating the physical proxy induces a new pose in the rig. To preserve pose details, the rest of the rig controls are synthesized from the sample poses via pose space interpolation in real-time (see Fig. 3.1).

We demonstrate that complex characters, often containing hundreds of bones, can be controlled with a compact tangible device consisting of much fewer pieces. Furthermore, our method is integrated directly into Autodesk's Maya® 3D animation software, emphasizing its practical applicability. We illustrate the method's utility by downloading and animating a variety of rigs without further modification. Results from a user study compare favorably to the hardware design of [Jacobson et al. 2014b] both in terms of accuracy and posing time, providing an average speed-up of $2\times$. Finally, we qualitatively show that our method enables more fluid control, of more complex characters, with less complex physical devices.

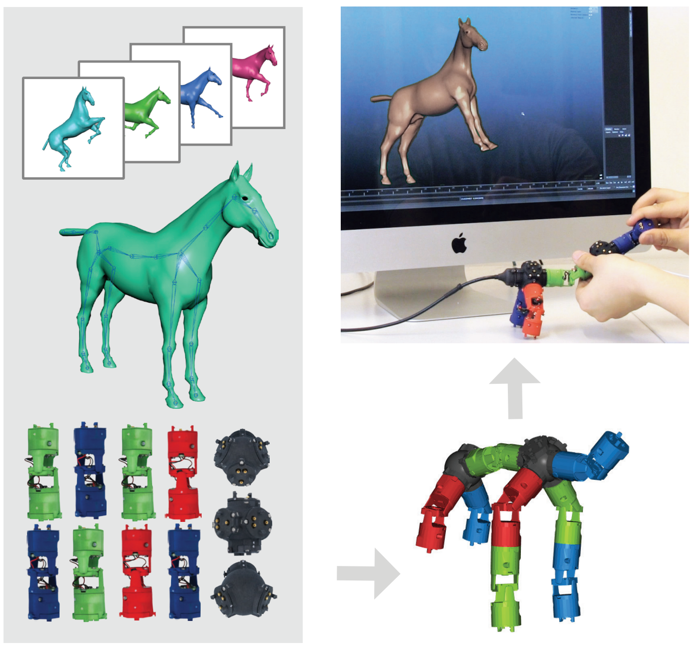


**Figure 3.2:** *Illustration of our pipeline from input character to fluid tangible animation. The horse has 29 bones, controlled by 8 joints.*

## 3.2 Method

Our approach provides a natural interface for posing and animating complex 3D characters using only a small set of physical controls, allowing novice users and experts to create animation sequences. The proposed solution consists of hardware and software contributions, designed in unison and complementing each other.

To gain an intuition, consider the following example: the user provides a rigged 3D character with a sparse set of sample poses (readily available online). Furthermore, the user indicates the kit (number of joints and splitters) to use. We then analyze the rig and the poses, identifying the DoFs with the most influence on pose reachability, weighted by the amount of controlled surface. Optimizing for direct control of these most important nodes, and using only the available parts, a device configuration and assembly instructions are computed. We solve a challenging, discrete assignment problem: finding a configuration of modular hardware pieces that maximizes coverage of a large pose space. In addition to the physical setup we also compute a mapping between sensed rotations and the rig's parameters, which ultimately control the character's pose.

After assembly the physical device is bound to the virtual rig and user inputs are mapped onto the rig. Manipulating the device induces a similar deformation onto the 3D character. In most cases the physical configuration has significantly fewer DoFs than the rig (see Fig. 3.2). To maintain expressiveness and add details back into the resulting motion, we use a pose space interpolation scheme to synthesize detailed pose nuances (cf. Fig. 3.1).

### 3.2.1 Hardware

Our design, inspired by [Jacobson et al. 2014b], follows a modular approach, decomposing the control structure into joints that measure 3D rotations, and splitters, which allow for branching. This design enables dynamic rearrangement of the parts into arbitrary topologies. However, one of the main limitations of [Jacobson et al. 2014b] is due to the mechanical joint design. Relying on twist-bend-twist joints (Fig. 3.3, left) the user has to decompose rotations into Euler angles, making the device prone to gimbal lock: moving the endpoint of a single joint between two coordinates on a unit sphere typically cannot be done by tracing the shortest path (i.e., the geodesic) but rather requires a sequence of individual rotations, with the endpoint zig-zagging over the surface (see Fig. 3.4).

**Mechanical design.** We propose a new joint design to overcome this limitation and provide a much improved user-experience by allowing for smooth tracing of geodesics – both for individual joints and chains of joints. The design is based on the Universal (or Cardan) joint augmented with two infinite twisting rings. The joint itself consists of a pair of hinges, oriented orthogonally to each other and connected via a ring shaped cross-shaft (Fig. 3.3, right). This design enables the joint to bend in any direction and maintains accuracy with an angular error below 0.5°. Fig. 3.4 illustrates the effect of our design

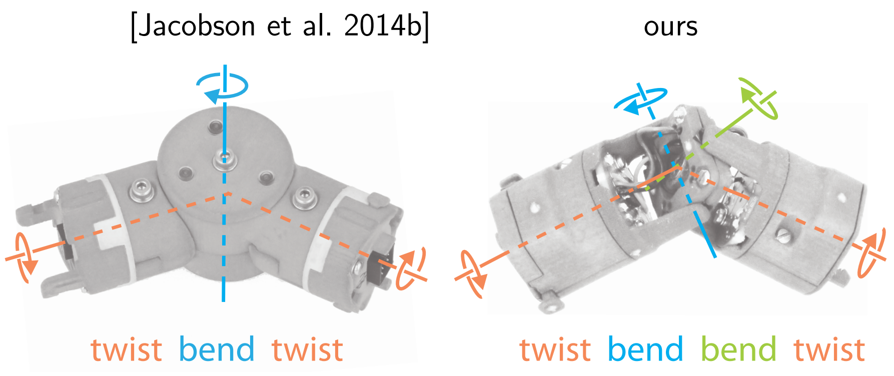


**Figure 3.3:** *Comparison of joint designs. Left: twist-bend-twist design of [Jacobson et al. 2014b]. Right: our proposed joint design.*

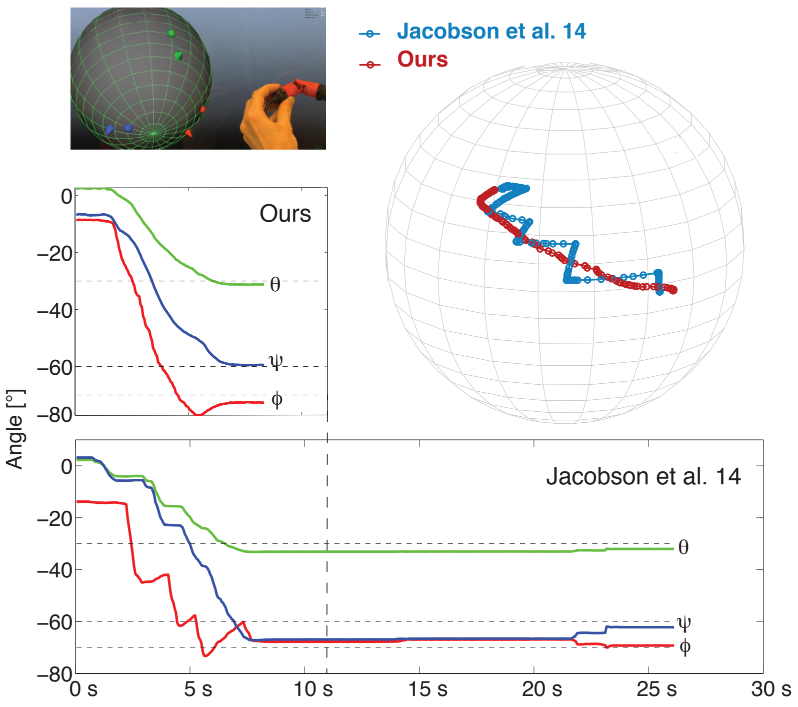


**Figure 3.4:** *Our physical angle parametrization allows for direct tracing of geodesics. The joints of [Jacobson et al. 2014b] force angular decomposition of rotations, resulting in a zig-zag pattern. Ours is also significantly faster in reaching the target position.*

when tracing geodesics on a sphere. Our design allows for much smoother and faster trajectories and a shorter path compared to [Jacobson et al. 2014b]. To enable rotation about the shaft itself, we incorporate twisting rings at either end of the joint. While a single twist would suffice, we experimentally found that two make user interaction smoother.

The joints in [Jacobson et al. 2014b] also featured twist rings but were limited in their rotation range due to the need for wired connections. We overcome this limitation using a pair of slip rings (inset), conducting power and transmitting electrical signals while allowing for infinite rotations. We use the same splitter geometries as [Jacobson et al. 2014b].

**Sensing and communication.** To recover the joint's rotations we utilize magnets and Hall effect sensors. The two bending angles are measured by embedding a passive axial magnet into the cross-shaft. This generates a local

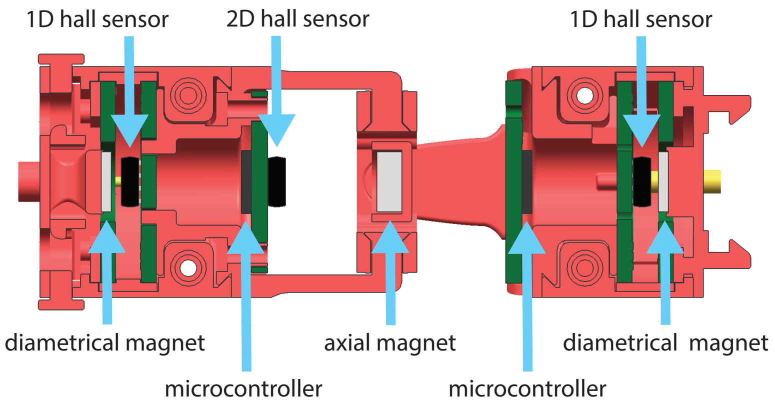


**Figure 3.5:** *Joint cross-section showing passive diametrical and axial magnets, Hall sensors and micro controllers.*

magnetic field and its orientation is sensed by a 2D Hall sensor, located at the base of the outgoing shaft. Analogously, we sense twist by leveraging diametrical magnets and 1D Hall sensors (see Fig. 3.5).

Each joint contains three sensors and two micro-processors, which reconstruct angles from sensor readings, serialize them and transmit downstream to the host computer. While the slip rings have desirable mechanical properties, their electrical connectivity is less reliable than wire-based solutions. To deal with connection dropouts we designed a fault tolerant protocol, robust to random disconnects and missing packages, allowing us to reconstruct the topology of the device consistently. For details we refer to Appendix A.1.

### 3.2.2 Rig retargeting

Tangible input devices (e.g., [Esposito et al. 1995; Knep et al. 1995; Yoshizaki et al. 2011; Celsys, Inc. 2013; Jacobson et al. 2014b]) promise natural manipulation of 3D characters. However, because they map degrees of freedom directly from device to virtual character, these are limited to controlling simple skeletons. In reality animation rigs are very complicated and have tens to hundreds of degrees of freedom (see Fig. 3.9), making direct control via tangible input devices impractical. We propose a set of algorithms to overcome this inherent limitation.

In professional rigs, a small number of degrees of freedom control large surface areas and hence predominantly define the character's pose. The rest of the rig then adds more subtle details (see Fig. 3.7). This observation is crucial to our approach. We propose to control these most important nodes directly using our hardware, while driving the remaining DoFs implicitly. To accomplish this we have developed methods to compute an optimized hardware configuration, a mapping that induces realistic poses of the character, and an interpolation scheme to add lost expressiveness at animation time.

At the heart of our algorithm is an objective functional that measures how well a given input character rig $\mathcal{C}$, with a set of sparse sample poses $\mathbf{P}^{\mathcal{C}}$, can be approximated by a device configuration $\mathcal{D}$. We define the pose error as the $L^2$ distance between the position of rig nodes, resulting in the following optimization problem:

$$\underset{\mathcal{D},\mathbf{P}^{\mathcal{D}}}{\arg\min} \sum_i \mathbf{W} \left( F(\mathcal{C}, \mathcal{M}(\mathbf{p}_i^{\mathcal{D}})) - F(\mathcal{C}, \mathbf{p}_i^{\mathcal{C}}) \right)^2, \tag{3.1}$$

where $F(\mathcal{C}, \mathbf{p})$ is the forward kinematic function that returns the position of the nodes of $\mathcal{C}$ in the pose $\mathbf{p}$ and $\mathbf{W}$ is a diagonal matrix weighting each node. A node's weight is proportional to the surface area which it directly controls. Each node also moves its connected subtree: by subtracting the area of each subtree, we ensure that each part of the surface is considered only once. Minimizing the energy in Eq. (3.1) computes $\mathcal{D}$, which is an injective assignment of hardware joints $\mathbf{j}$ to bones $\mathbf{b}$, and splitters $\mathbf{s}$ to branching nodes $\mathbf{n}$ of the rig (see Fig. 3.6). This also induces a retargeting function $\mathcal{M}$ which converts the device poses $\mathbf{p}^{\mathcal{D}}$ to rig poses $\mathbf{p}^{\mathcal{C}}$ by relating local joint rotations directly to bones. Generally speaking, there are more bones than joints and all bones without joint assignment move rigidly following the forward kinematic chain during fitting. Later these will be controlled by the pose interpolation scheme, discussed below. The device configuration $\mathcal{D}$ uniquely determines how to assemble the input device, and we visualize it as a set of assembly instructions, presented to the user as a 3D rendering (Fig. 3.6, right).

Evaluating Eq. (3.1) is fast since it only compares the positions of the rig nodes. However, finding the assignment $\mathcal{D}$ that spans the largest pose space is very challenging since it contains a discrete element: deciding which pieces of hardware to use and how to connect them.

To make the assignment problem computationally tractable, we propose an iterative algorithm with two alternating phases: First, we assign all available joints $\mathbf{j}$, ignoring all branching nodes in the rig. Second, we assign splitters $\mathbf{s}$ to branching nodes $\mathbf{n}$. Since the last step may remove already assigned joints, we repeat this procedure until there are no more joints or splitters left.

Fig. 3.7 illustrates the effect and importance of our algorithm. We plot pose error (measured in Eq. (3.1)) as function of the hardware kit size. Initially the error is reduced by 90% once reaching 5 joints and by 98% with 10

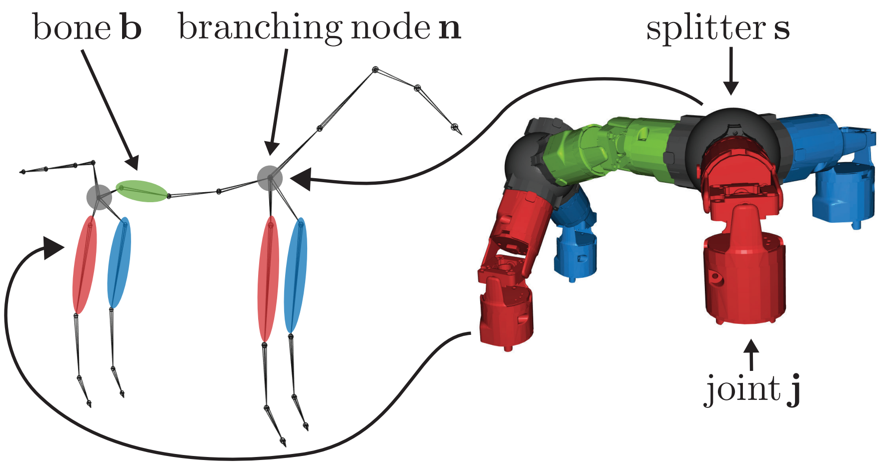


**Figure 3.6:** *Our algorithm converts a rig of bones connecting nodes (left) to a device consisting of joints and splitters (right).*

joints. However, the remaining 17 DoFs only contribute marginally to the reachability of poses. Hence, a full device would be unnecessarily bulky and require a large set of parts.

**Joint assignment.** The first phase of the algorithm assigns all available joints in the kit, maximizing $\mathcal{D}$'s span of the pose space. This assignment is computed in a greedy way: we try to assign a joint $\mathbf{j}$ to each unassigned bone $\mathbf{b}$ and for each assignment we minimize Eq. (3.1), which reduces to a smooth nonlinear optimization (details in Appendix A.2). We then pick the assignment with the lowest energy and we add it to $\mathcal{D}$. This procedure is repeated until all joints are assigned. The resulting sparse assignment $\mathcal{D}$ inherits the connectivity of the original rig $\mathcal{C}$: intuitively, the algorithm computes a sparse assignment, and then collapses all unassigned edges.

**Splitters assignment.** The second phase of the algorithm assigns splitters to branching nodes. The branching of the input rig can be arbitrary both in terms of valence and geometry. This makes a straightforward application of the above assignment strategy infeasible, since we only want to use a limited number of splitters with fixed valences and geometries.

Instead of greedily assigning splitters, we propose a global strategy based on integer linear programming (ILP) to optimally assign available splitters to branching nodes. The program is augmented with linear constraints that encode hierarchical dependencies and physical feasibility. Intuitively, we seek to globally minimize the mismatch in valence and geometry of branching

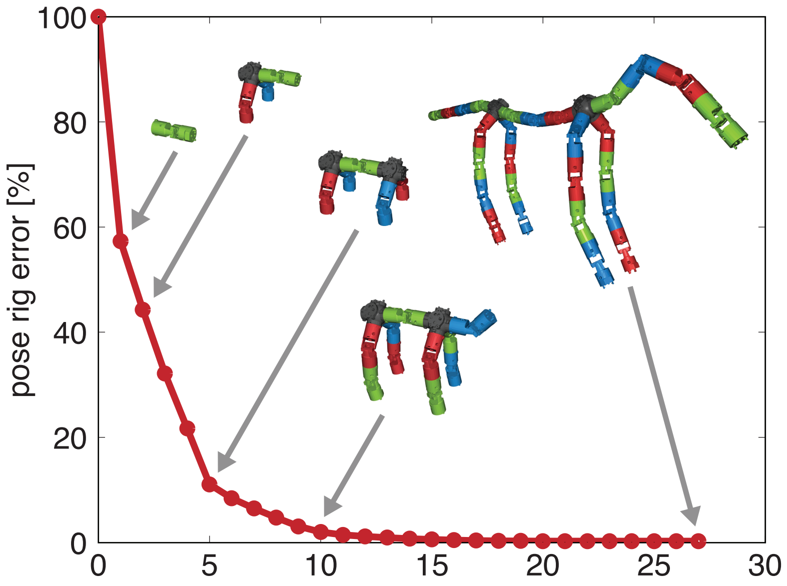


**Figure 3.7:** *Influence of joint number in the kit on pose accuracy.*

nodes and splitters, weighted by the impact on the total pose error of the downstream assignment. For example, sometimes it may be acceptable to prune an entire subtree of the rig if its influence on the final poses is low. More formally we write:

$$\arg\max_{\mathbf{x}} \ \mathbf{c}^T\mathbf{x}, \tag{3.2}$$

where $\mathbf{c}$ is the gain of assigning a specific splitter $\mathbf{s}$ to a specific branching node $\mathbf{n}$. The corresponding gain is defined as the reduction in energy (Eq. (3.1)) when replacing all fully rigid (not assigned with splitter $\mathbf{s}$) outgoing branches of $\mathbf{n}$ with non-rigid bones up to the next branching node. And $\mathbf{x}$ is the vector of binary variables that encode the assignment of splitters to branching nodes. To compute $\mathbf{c}$ and $\mathbf{x}$ we perform the following procedure.

First, we enumerate all possible combinations of branching nodes $\mathbf{n}$ in the rig and all splitters $\mathbf{s}$. Branching nodes $\mathbf{n}$ have one inlet $\mathbf{g}^n$ and 2 or more outlets $\mathbf{g}_i^{\mathbf{n}}$. Each of the outlets has an orientation, represented as rotation from $\mathbf{g}^{\mathbf{n}}$ to $\mathbf{g}_i^{\mathbf{n}}$. The splitter also has one inlet $\mathbf{g}^{\mathbf{s}}$ and outlets $\mathbf{g}_j^{\mathbf{s}}$, where generally the valence and orientations differ with respect to $\mathbf{n}$. See Fig. 3.8 for an illustration.

Second, for each $(\mathbf{n}, \mathbf{s})$-pair (as in Fig. 3.8, left and middle) we map the inlet and enumerate all permutations of outlet assignments. For each such mapping we find the best possible orientation for the splitter $\mathbf{s}$ via [Kabsch

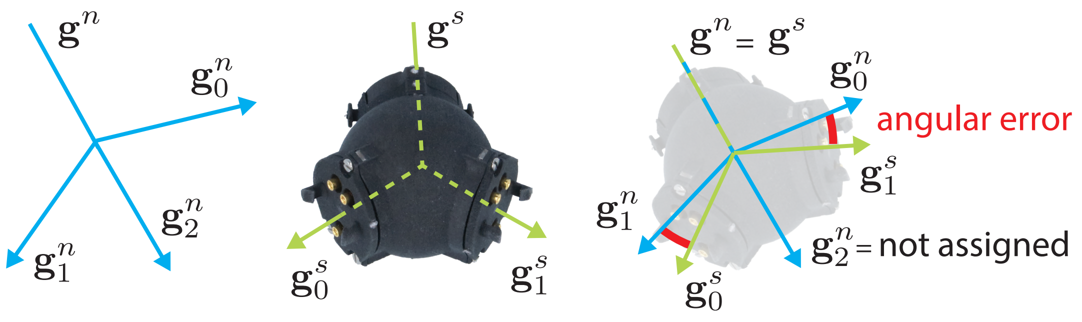


**Figure 3.8:** *Branching node $\mathbf{n}_0$ with outlets $\mathbf{g}_0^n$, $\mathbf{g}_1^n$ and $\mathbf{g}_2^n$ (left). Candidate splitter $\mathbf{s}_0$ with outlets $\mathbf{g}_0^s$ and $\mathbf{g}_1^s$ (middle). Assignment $\mathbf{x}_0$, aligning the pair $\mathbf{n}_0$ and $\mathbf{s}_0$ (right).*

1976] The resulting assignment is then used to compute the gain via Eq. (3.1), comparing to a fully rigid configuration. Note that we set the remaining rotational offset as rest-pose rotation on the outgoing physical joint. Furthermore, we filter out assignments where matched gates have a remaining angular error exceeding 60°(Fig. 3.8, right).

**Physical validity.** We augment the ILP in Eq. (3.2) with a set of linear equality constraints, ensuring that for each splitter $\mathbf{s}_j$ and assignment vector $\mathbf{x}$ only one entry is set to 1. Let $\mathcal{A}(\mathbf{s}_j)$ be the set of assignments which contain splitter $\mathbf{s}_j$, then:

$$\sum_{i \in \mathcal{A}(\mathbf{s}_j)} \mathbf{x}_i \leq 1, \quad \forall j, \tag{3.3}$$

enforces that no splitter is assigned more than once. Analogously, let $\mathcal{A}(\mathbf{n}_j)$ be the set of assignments which contain branching node $\mathbf{n}_j$, then:

$$\sum_{i \in \mathcal{A}(\mathbf{n}_j)} \mathbf{x}_i \leq 1, \quad \forall j, \tag{3.4}$$

precludes double assignments to nodes.

As previously noted, often it is the case that the valence of $\mathbf{n}$ and $\mathbf{s}$ differs, resulting in configurations of $\mathcal{D}$ where entire subtrees of the rig $\mathcal{C}$ are not present. The following set of hierarchical constraints ensures that no splitters are wasted on assignments to such "virtual" branches and that they do not contribute to the gain:

$$\sum_{i \in \mathcal{A}(\mathbf{n}_c)} \mathbf{x}_i \leq \sum_{k \in \mathcal{A}(\mathbf{n}_p, m)} \mathbf{x}_k, \tag{3.5}$$

for all nodes $\mathbf{n}_p$,$\mathbf{n}_c$ s.t. $\mathbf{n}_p$ is the parent of $\mathbf{n}_c$ and $\mathbf{n}_c$ is connected to the chain of the $m$-th outlet of $\mathbf{n}_p$. The notation $\mathcal{A}(\mathbf{n}_p, m)$ denotes the set of assignments that contains $\mathbf{n}_p$ and have its $m$-th outlet used, i.e., the branch attached to the $m$-th outlet is not "virtual".

**Implementation details.** The rig reduction and hardware interface are implemented in C++ using Eigen, and we use Gurobi [Gurobi Optimization, Inc. 2015] to solve the ILP program Eq. (3.2) with constraints (3.3,3.4,3.5). The solver may return several solutions, for example having multiple identical splitter outlet assignments for one node. We favor the solution with the lowest average rotational error of the splitters, meaning that the geometry of the solution better approximates the rig geometry.

Typically hardware kits only contain few splitters. Hence it is likely that there are remaining, unassigned branchings. Obviously, joints can be seen as splitters with a single outlet, allowing us to cover at least one branch for such cases. This is realized by assigning "phantom" splitters of valence 1 to the kit while solving the ILP. These are simply removed before generating the assembly instructions, leaving only the attached joint (cf. Fig. 3.9, bottom row).

Note that having multiple copies of each splitter in a kit is common. For efficiency, we compute the candidate assignments only once for each kind of splitter. Constraints Eq. (3.3) are also grouped to act on classes of splitters instead of individual $\mathbf{s}$.

**Realtime pose interpolation.** The algorithm discussed so far produces a device $\mathcal{D}$ controlling a subset of the DoFs of the rig, leaving the remaining DoFs unchanged. To synthesize those missing rig control values which can add detailed pose nuances back into the final animation, we employ [Buck et al. 2000] for pose-space interpolation. Principal component analysis on a set of sample poses is used to choose the largest principal vectors of the dominant dimensions. These map the original high dimensional poses, defined as stacked quaternions representing rotations of the active rig controls, into a lower dimensional space. Delaunay triangulation is used to create a piecewise linear manifold for interpolation.

Given a new pose, we first project it onto this space, locate the projected point in the triangulation, then find the barycentric coordinates inside the simplex that contains it. The coordinates then act as blending weights. We then use Slerp to interpolate the quaternions and feed them to the previously rigid joints. The technique is efficient enough for realtime computation of the barycentric blending weights and generates continous interpolation results. More samples in the pose interpolation will generally increase the quality and

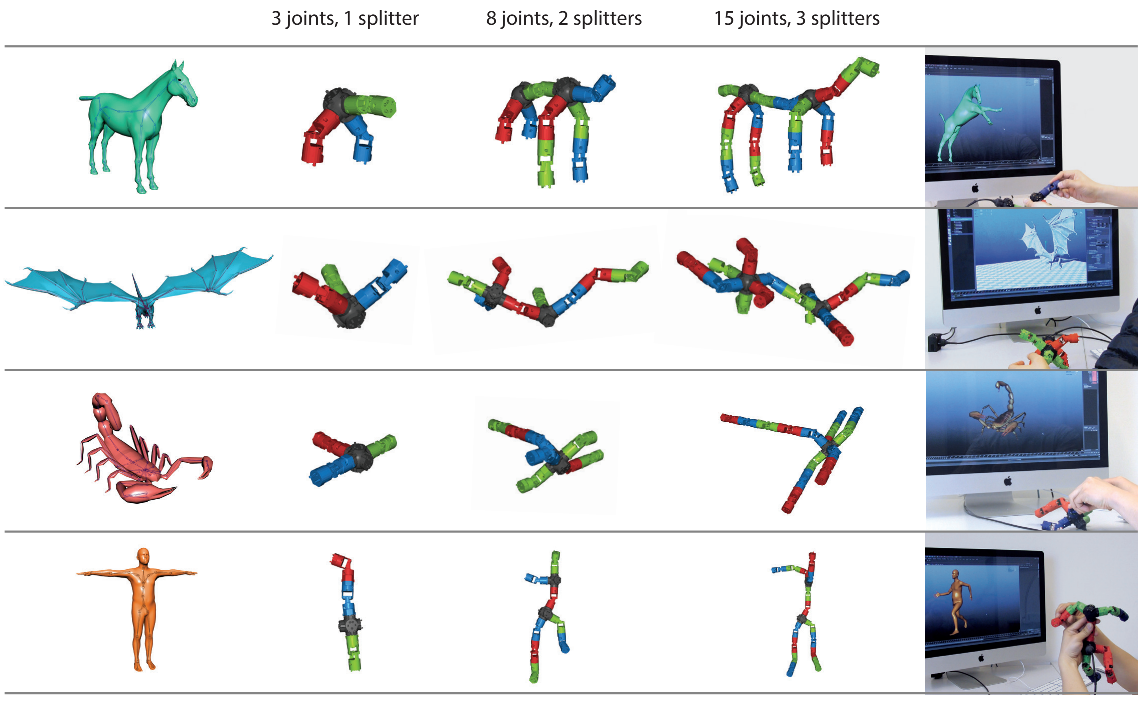


**Figure 3.9:** *Depending on the available kit, device build instruction plans with different complexity are generated by our algorithm. Note that the models have much higher degrees of freedom than the generated control structures. The inputs were (nr. bones/nr. sample poses): Horse: (29/25 galloping, going up) – Dragon: (110/12 flying, some walking); Scorpion (62/20 walking, attacking); Dancer (22/6). Note that the device for the Dancer is asymmetric due to the asymmetry in the input poses: the left arm of the character moves almost rigidly with the torso and it is thus not necessary to have any joint controlling the left arm.*

there is no upper limit on input poses. Since our experiments with higher-dimensional pose spaces did not lead to improved results likely due to the curse of dimensionality, a 2D pose space is used (as proposed in [Buck et al. 2000]).

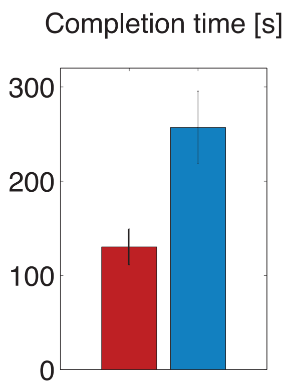
## 3.3 Evaluation

We briefly report on a series of experiments that we conducted to evaluate our algorithms for the computation of the physical device configurations and to compare our proposed hardware design to the closest state-of-the-art [Jacobson et al. 2014b].

For our experiments we fabricate a hardware kit containing 18 joints and 7 splitters. We implemented a Python program for Autodesk's Maya® [Maya 2014], driving characters which we acquired online.

**Rig reduction and device configuration.** Fig. 3.9 qualitatively demonstrates the results from our method for different 3D characters ranging in complexity from 22 bones (*Dancer*) to 110 bones (*Dragon*), using a small number of sample poses as input (*Dancer*: 6; *Horse*: 25). Optimized device configurations are computed based on three hardware kits of increasing size. Our method produces intuitive and sensible configurations, placing splitters and joints such that they articulate the most important features of the character. For each of the examples the amount of direct control a user has increases with number of joints. However, there clearly is a trade-off between directness of the mapping and physical complexity. Fig. 3.7, plotting pose error as function of joints, illustrates this further: initially additional joints drastically decrease the error but the curve levels off, and beyond 10 joints gains are marginal. Experimentally we found that most characters can be controlled well with 5-10 joints. This supports our initial goal of combining the benefits of tangible control and rig animation algorithms. Please note that the algorithm can produce asymmetric configurations even for symmetric characters, e.g., for the *Dancer*. This can be due to asymmetric input poses (rightmost column), or due to the limited number of joints in the kit (leftmost column), which forces the algorithm to introduce a "splitter" replacement (in red). This enables the articulation of the character's arm despite the lack of splitters. In such cases pose interpolation controls the un-mapped limbs causing whole character to move smoothly, albeit at the cost of direct control (cf. Sec. 3.4).

**Hardware comparison.** Intertwined with our algorithmic contributions is the hardware design, which we discussed in Sec. 3.2.1. To isolate the effect of the new joint design we repeat the posing experiment reported in [Jacobson et al. 2014b]. We concentrate on directly comparing *their* device with *ours*, assuming that results will carry over to the established baseline (a mouse-based Maya-like UI). We faithfully recreated the reported experimental conditions with 10 participants, recruited from our university (2 female, 8 male; ages ranging from 25 to 35). Device presentation was counterbalanced by half of the participants starting with *our* device, half with *theirs*. Per posing experiment, participants were asked to replicate poses shown on-screen as close as possible and stop once they



reached a satsifactory accuracy. The results indicate that the new hardware considerably outperforms the existing design by [Jacobson et al. 2014b]. *Ours* requires significantly less time (see inset; *ours* mean = 130.24 s, standard deviation = 19.07; *theirs* mean = 256.94 s, SD = 38.62) and achieves higher relative accuracy, represented by the remaining pose error in percent of the original pose error at the beginning of each experiment (*ours*: mean = 21.66% SD = 3.62; *theirs* mean = 36.23% SD = 5.48). These differences also translate into the more lenient metric of "work" reported in [Jacobson et al. 2014b], essentially the area under the plot lines in Fig. 3.10 (*ours*: mean = 89.65% SD = 18.8; *theirs* mean = 474.1% SD = 123.7). A Student's $t$-test reveals that all differences are statistically significant (all $p$-values $\leq 0.05$).

Please note that the speed-up factor of 2 (130.24 s to 256.94 s) reported here is a very conservative estimate. It compares the time elapsed until both devices reach *their own minimum pose error*. In contrast, when measuring the more meaningful average ratio between the time elapsed until the more accurate device reaches the *minimum pose error of the less accurate* device, *ours* achieves a speed-up of $3\times$ (see also Fig. 3.10).

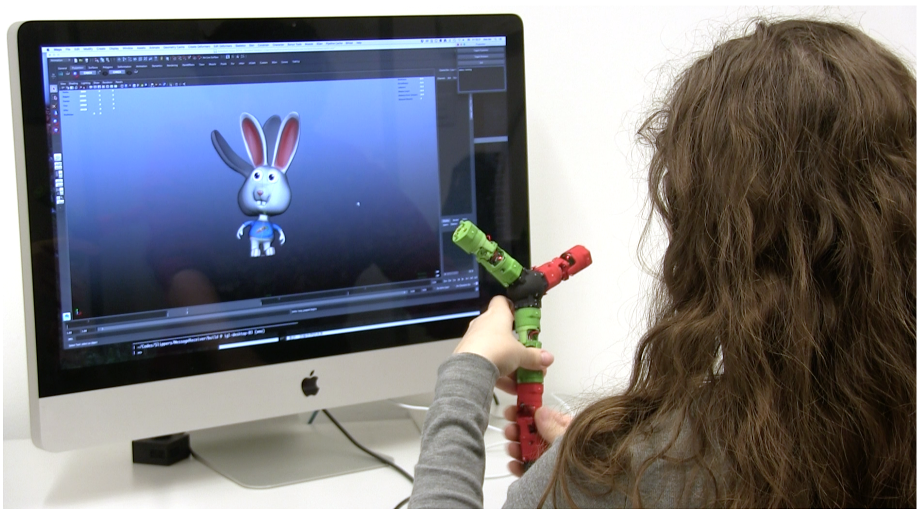
This difference was also clearly noticeable by observing the subjects during the study: with our hardware the posing can be done rapidly and with minimal experimentation, while many participants needed a long time to understand how to reach the target pose with the old design, due to the requirement to decompose rotation. Fig. 3.4 illustrates this further showing the effect of directly tracing geodesics with our design versus sequential manipulation of individual Euler angles in the old device.

## 3.4 Additional results

To assess the resulting animation qualitatively, we report on a number of examples produced with our approach, consisting of hardware and algorithms that compute the device configuration and pose interpolation at runtime. Please also see the accompanying video[1].

**Interactive rig control.** Fig. 3.11 shows a sequence of a user driving the *Human* (24 bones) with 7 joints and 2 splitters. The joints are bound to the bones in the torso and the limbs. Since we only control 7 bones directly, the remaining DoFs are controlled via pose interpolation, which reintroduces pose nuances.

[1]`https://igl.ethz.ch/projects/rig-animation-input-device/riganimation2016_video.mp4`



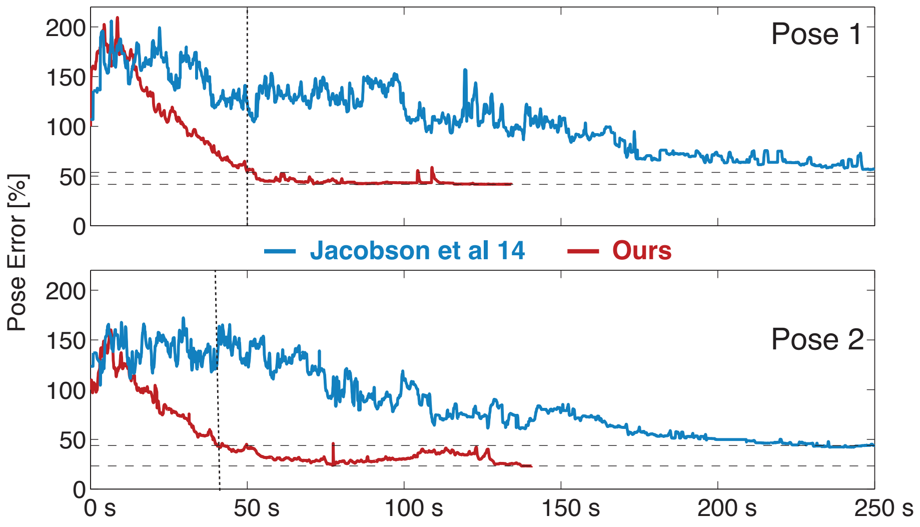


**Figure 3.10:** *Top: Experimental condition with on-screen stimuli. Bottom: Two of the poses in the user study, averaged across all participants, show* ours *(red) and (blue) decreasing pose distance from 100% to minimal values at completion (dashed lines).* Ours *is significantly faster and achieves better accuracy.*

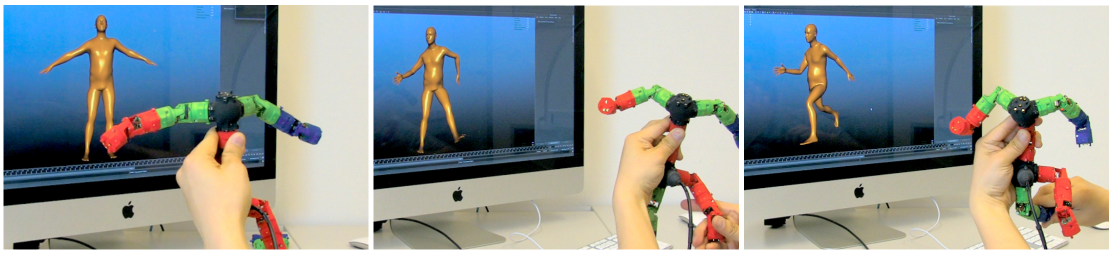


**Figure 3.11:** *Interactive animation of the Human character.*

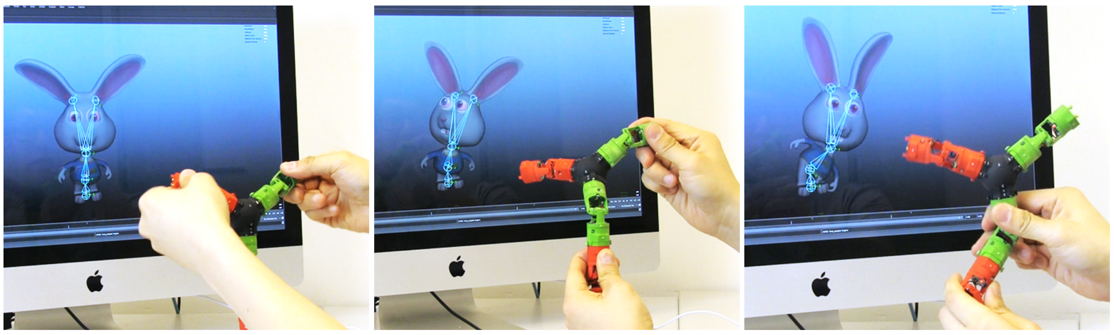


**Figure 3.12:** *Physical inverse kinematics. The user positions end-effectors with the device, and the 3D character smoothly follows.*

**Physical inverse kinematics.** An important design goal was to create a user experience that resembles inverse kinematics, allowing the user to work directly with positional constraints, rather than having to deal with individual rotations. In Fig. 3.12, a user positions only the end-effectors of the chain of joints, and the remaining configuration follows smoothly, resulting in a fluid animation of the on-screen character.

**Pose interpolation.** The impact of the pose interpolation scheme is demonstrated in Fig. 3.13. Here we drive the *Scorpion* (64 bones) without (top) and with (bottom) pose interpolation. In both cases we use an identical device that only controls 6 of the bones directly. The approach is particularly useful for such complex characters, which are challenging to pose, especially for novice users. In the extreme, the technique can be used to drive animation sequences with very few pieces, an example is shown in Fig. 3.14.

**Keyframing.** Prior tangible devices heavily focus on keyframe posing. This is also possible with our approach. Fig. 3.15 shows a selection of frames from an animated sequence of the Dragon rig (110 bones), driven by only 8 joints and 2 splitters. The posing of this 30 seconds long animation consisting of 32 key frames took an inexperienced user 35 minutes, of which 15 were spent on non-posing tasks, such as global transformations and camera positioning.

**Abstract parameter control.** Our device is also useful for controlling non-rotational rig parameters. In Fig. 3.16, we control a blendshape rig with four expressions. We manually map the angles of a single joint onto the four expressions to create a simple facial animation.

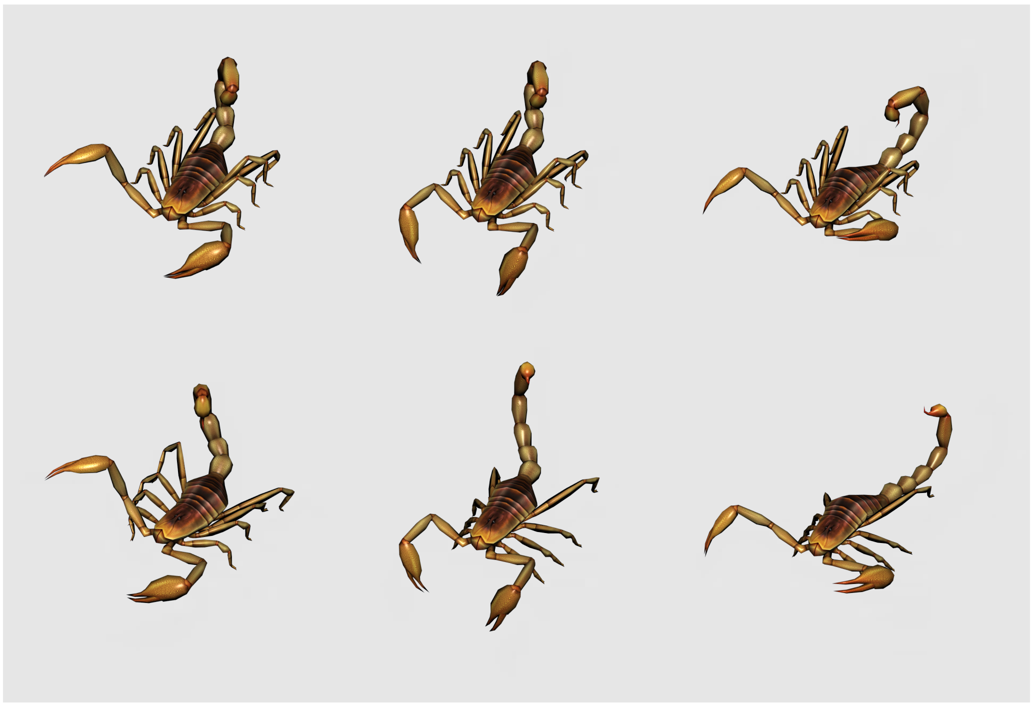


**Figure 3.13:** *Effect of pose interpolation. Top: without interpolation. Bottom: with interpolation. Note the additional pose details in the tail and the legs.*

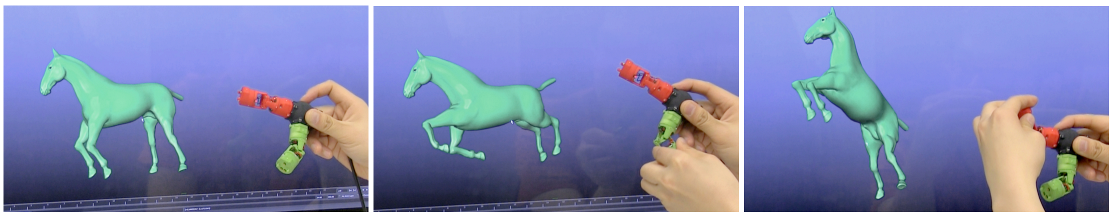


**Figure 3.14:** *An extreme case for pose interpolation. We control the Horse rig with only two joints. The red joint controls the torso and the green joint controls one of the rear legs directly; the rest of the pose is synthesized by pose interpolation.*

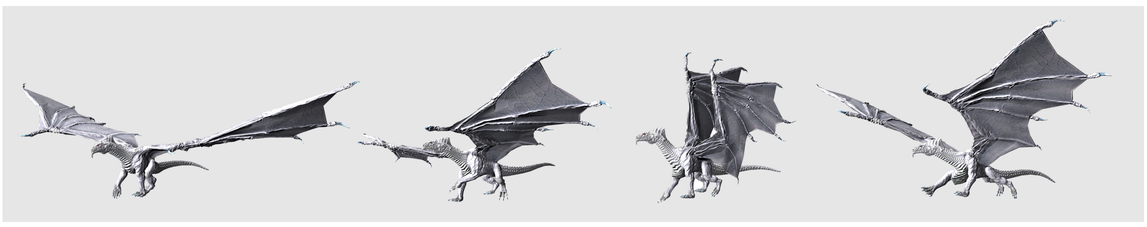


**Figure 3.15:** *We show some of the keyframes generated by an inexperienced user from a key framing session. This Dragon rig contains 110 bones, and a 30 seconds long animation can be generated in 35 minutes, of which 15 were spent on non-posing tasks.*

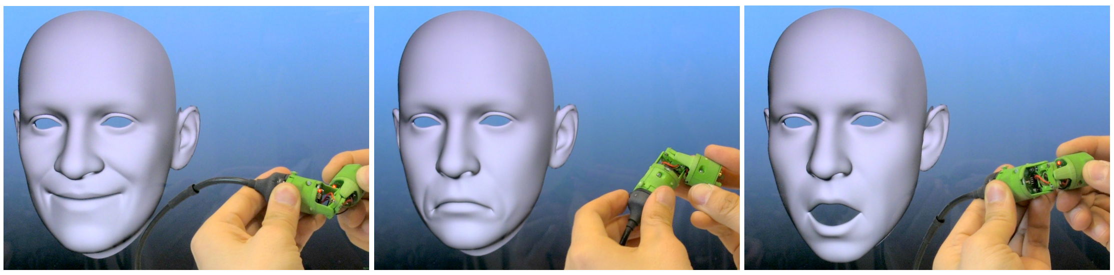


**Figure 3.16:** *Controlling a face blendshape rig with a single joint. We manually map the angles of the joint such that they semantically match the facial expressions (i.e., a $\vee$ shape for smiling, a $\wedge$ for frowning, and twist opens the mouth).*

## 3.5 Conclusion

We proposed a symbiotic pair of novel software and hardware to help animators traverse a large space of poses via fluid tangible manipulation. The key insight is to drive only the most important degrees of freedom of a rig directly, limiting physical device size and simultaneous manipulations. Furthermore, we have demonstrated that our hardware provides a much improved user experience, resembling inverse kinematics. Empirically we have shown that this yields speeds-ups of a factor of 3 compared to the most related work in terms of character posing time.

Together, our software and hardware are an important step beyond static keyframing and into the territory of motion sculpting. We believe our approach already provides a valuable tool chain to quickly sketch animations using a variety of professional rigs. However, it is not without limitations and opportunities for future work.

Currently, we employ a standard pose interpolation scheme. While this already produces good results, it is sometimes an issue that desired poses are not represented in the example set, or that the pose set is biased. For example, our Horse character comes with an extended set of galloping poses, which causes the horse's legs to curl even for non-running poses. Exploring more sophisticated interpolation schemes, prioritizing user-specified DoFs over those from example poses, would be interesting future work.

In terms of hardware, avoiding gimbal lock is a large leap in usability, but further improvements could be achieved. Especially with complex device configurations, it would be desirable to be able to control joint friction, perhaps dynamically. However, this is a non-trivial design challenge both in terms of mechanics and potential negative impact on the desired IK properties. Similarly, designing additional types of joints, such as prismatic or telescopic joints, would allow us to support rigs with translational degrees of freedom (such as cartoon characters) and is a challenging but interesting area for future work.

Our device only senses local rotations, while the virtual character's global position and orientation have to be controlled externally. Augmenting the device with an additional sensing mechanism, perhaps vision-based, or leveraging inertial sensors, would further decrease the barriers to fully fluid character animations.

We have concentrated exclusively on character animation. However, we believe that the wide range of motion, modular nature and high accuracy provided by our hardware make it an ideal candidate to be explored in other application domains, such as general purpose input devices (e.g., game-pads, joysticks) or as controllers for virtual or augmented reality applications.

CHAPTER 4

# Stretch Sensor Arrays for Deformation Capture

## 4.1 Introduction

Motion capture is an essential tool in many graphics applications, such as character animation for movies and games, sports, biomechanics, VR, and AR. Most commonly, motion capture systems are camera based, either relying on body-worn markers or more recently markerless. Vision based approaches can be highly accurate and in the case of multiview or depth imaging, they can provide dense surface reconstructions. However, such systems rely on extensive infrastructure and are therefore mostly confined to lab and studio use. Other sensing modalities, such as body-worn inertial and magnetic sensors, or resistive and capacitive distance sensors have been explored to provide more mobility, yet these are typically limited to capturing skeletal deformation only.

We introduce a new, practical and affordable approach to deformation sensing and motion capture. Our approach bridges the gap between vision-based and inertial approaches by providing accurate sensing of dense surface deformations while being wearable, and hence practical for scenarios in which stationary cameras are unsuited, for example to capture muscle bulging below clothing.

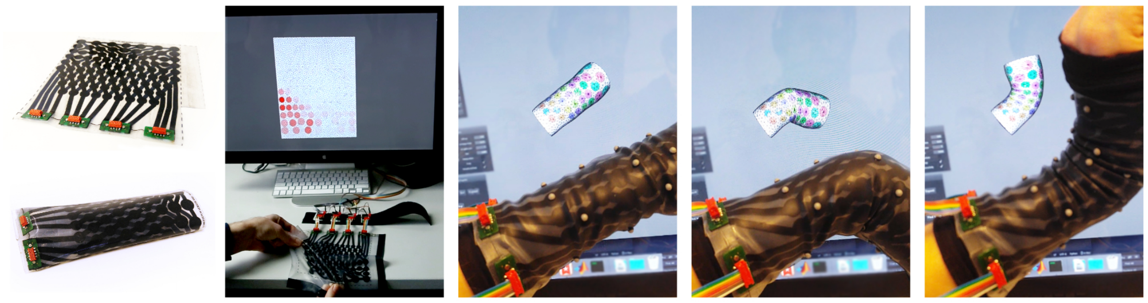


**Figure 4.1:** *Left to right: We propose a method for the fabrication of soft and stretchable silicone based capacitive sensor arrays. The sensor provides dense stretch measurements that, together with a data-driven prior, allow for the capture of surface deformations in real-time and without the need for line-of-sight.*

**Capacitive sensor array.** We propose to leverage a capacitive sensor array, fabricated entirely from soft and stretchable silicone, that is capable of reconstructing its *own* deformations. The sensor array provides dense measurements of area change, which can be leveraged to reconstruct the underlying 3D surface deformation without requiring line-of-sight (see Fig. 4.2). We furthermore contribute a data driven surface reconstruction technique, allowing for the capture of non-rigid deformations even in challenging conditions, such as under heavy occlusion, at night, outdoors, or for the acquisition of uncommon deformable objects. Conductive polymers have been leveraged to fabricate resistive bend sensors [Rendl et al. 2012; Bächer et al. 2016], and are the basis of soft capacitive distance sensors, which are now readily available commercially [Str 2018; Par 2018]. Such stretchable capacitive sensors are enticing, since they are thin, durable, and may be embedded in clothing or directly worn on the body. However, so far fabrication has been involved and required specialized equipment, driving up cost. Moreover, such sensors have not been demonstrated to be accurate enough for motion capture and are typically limited to measurement of uniaxial deformation. Please note that capacitive sensing is often considered synonymous with touch sensing [Grosse-Puppendahl et al. 2017; Lee et al. 1985; Rekimoto 2002], in which capacitive coupling effects are leveraged to detect finger contact with a static sensor. In our context, however, the term is used in a different sense, referring to the fact that capacitance changes when an electrode undergoes deformations.

**Custom fabrication method.** We introduce a fabrication method for soft and stretchable capacitive deformation sensors, consisting of multiple bonded layers of conductive and non-conductive silicone. Crucially, the method only requires casting silicone and etching conductive traces by a standard

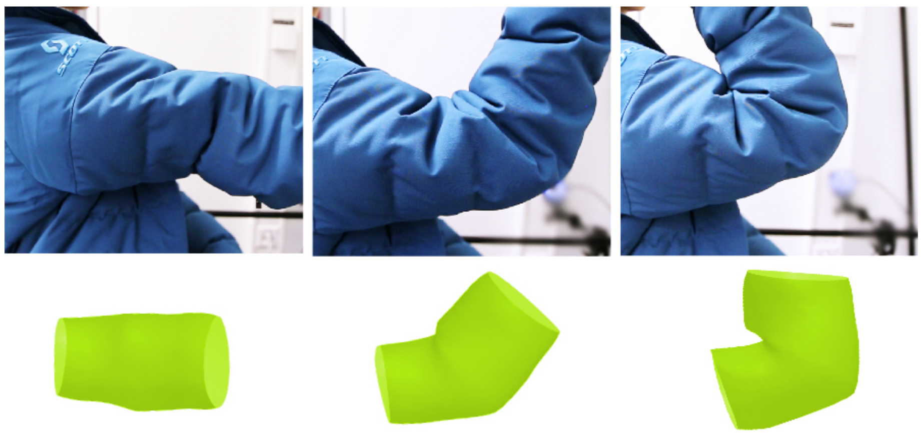


**Figure 4.2:** *An elbow "hidden" below by a jacket. Top: Video frames for comparison. Bottom: With our approach the dense surface deformation is estimated without requiring line of sight.*

laser cutter, and can thus be performed using hardware commonly available in a modern fabrication lab. The precision and accuracy of our sensors is comparable to commercial solutions, and the involved material costs are low. Our approach supports embedding many sensor cells of custom shape in a single thin film. Each cell measures changes of its own area, caused by deformation of the surface it is attached to. The resulting sensor array can be read out at interactive rates.

**Geometric prior.** While providing a rich signal, the area measurements alone are not sufficient to uniquely reconstruct the full 3D sensor shape due to isotropy and lack of direct bending measurement. They are however sufficient when paired with an appropriate geometric prior, if expected deformations involve some amount of non-area preserving stretch. In addition to the hardware, we propose an effective pipeline to acquire the deformation of the sensor worn by a user, for example wrapped around the wrist or an elbow. We propose a data driven technique based on a neural network regressor to reconstruct the sensor geometry from area measurements. At runtime, the regressor estimates the location of a sparse set of vertices, and the dense deformed surface is computed by a nonlinear elastic deformation method, obtaining a high-resolution reconstruction in real-time (see Fig. 4.1).

To acquire the necessary training data, we overcome an additional challenge: optical tracking systems struggle with the heavy occlusions and large deformations typical for natural motions of wrists, elbows and other multi-axial joints. Furthermore, when capturing other non-rigidly deforming objects, skeletal priors cannot be leveraged to recover missing markers. We thus introduce a semiautomatic ground truth acquisition technique, enabling capture of the necessary training data in minutes and reducing tedious manual cleanup to a minimum. The approach leverages an elastic simulation of the sensor to disambiguate the marker tracks, deal with unlabelled markers and correctly attribute marker positions to the digital mesh model of the sensor.

**Evaluation.** We demonstrate our sensors in action by acquiring dense deformations of a wrist and lower part of the hand (see Fig. 4.1), an elbow, an inflating balloon, and muscle bulging. We also capture deformations of flat sensors, both in and out of plane, which shows the precision and localization properties of our capacitive sensor arrays. Finally, we evaluate the prediction accuracy of the learning based prior quantitatively.

## 4.2 Overview

We present a stretchable silicone elastomer based sensor and its corresponding fabrication procedure. The sensor senses its *own deformation* and estimates the local surface area changes during deformation when wrapped around an object or a body part of interest (e.g., a wrist). The sensor array is fabricated layer onto layer entirely from 2-component silicone elastomer with conductive elements made from the same silicone but mixed with carbon black particles. The conductive layers can be designed to contain custom electrode patterns via etching with a standard laser cutter. This approach avoids the production of masks or molds and makes interlayer alignment very straightforward and precise.

As a further contribution we introduce a silicone-based capacitive area sensor *array*, whereas prior work only demonstrated individual stretch sensing elements, and arrays only to detect dense touch or pressure (e.g., [Lipomi et al. 2011; Sarwar et al. 2017; Nittala et al. 2018; Engel et al. 2006; Ponce Wong et al. 2012; Wissman et al. 2013; Block and Bergbreiter 2013; Woo et al. 2014]). Our key insight is that such arrays could also be used to attain dense localized area changes, given an appropriate read-out scheme. Our arrays are made by placing electrode strips in two conductive layers, separated by a dielectric, together forming a non-uniform grid of capacitors. Furthermore, we propose a scanning based read-out scheme that does not require individually connected capacitors, which would require a large number of layers or a large portion of the sensor area dedicated to connection leads. Instead, we propose a time-multiplexing procedure to indirectly read out capacitance values, which allows for a drastically simplified routing of electric connections.

By integrating all the capacitance readings, we can acquire area changes with a sufficient granularity and accuracy to reconstruct the geometry of an object, given suitable geometric priors. These dense area measurements are therefore combined with a deep learning based regressor to attain 3D position estimates of key points on the surface and an elastic deformation optimization to obtain dense deformation reconstructions.

In the following sections we provide a brief primer on capacitive sensing (Sec. 4.3.1), detail our sensor design (Sec. 4.3.2) and detail the fabrication (Sec. 4.3.3). We then complete our method by introducing our data capture and cleanup, learning and surface reconstruction approaches (Sec. 4.4).

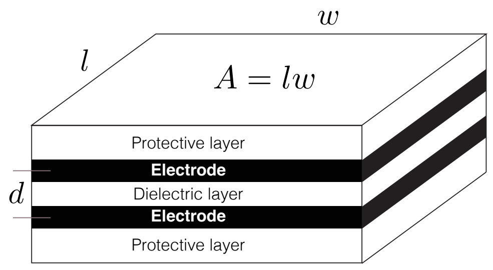
## 4.3 Sensor Design

### 4.3.1 Preliminaries

The capacitance $C$ (in Farads) of a plate capacitor is given by

$$C = \epsilon_r \epsilon_0 \frac{A}{d} = \epsilon_r \epsilon_0 \frac{lw}{d}, \tag{4.1}$$

where $A$ is the area of overlap of the two electrodes (in square meters), $\epsilon_r$ is the dielectric constant, $\epsilon_0$ is the electric constant and $d$ is the separation between the plates (in meters). Assuming a rectangular plate capacitor, $l$ is its length and $w$ the width. While originally derived for static plate capacitors, this relationship also holds for capacitors made from silicone elastomers [Atalay et al. 2017; Huang et al. 2017; O'Brien et al. 2014]. To minimize capacitive coupling effects with other objects, capacitors are typically shielded via insulating layers (see inset).



Using Eq. (4.1), and assuming the same Poisson ratio of width and thickness of the sensor ($d/d^0 = w/w^0$), a linear relationship between the ratio of the stretched capacitor's length $l$ to the rest pose length $l^0$, and the ratio of the capacitance of the stretched capacitor $C$ to the rest pose capacitance $C^0$ can be established:

$$\frac{C}{C^0} = \frac{\epsilon_r \epsilon_0 \frac{lw}{d}}{\epsilon_r \epsilon_0 \frac{l^0 w^0}{d^0}} = \frac{l}{l^0} \frac{w}{w^0} \frac{d^0}{d} = \frac{l}{l^0}. \tag{4.2}$$

Prior work applies this principle to the design of capacitive, uni-axial stretch sensors [Atalay et al. 2017] by continuously measuring a capacitance, which is then transformed to length measurement using Eq. (4.2). Note that here, an assumption is made that stretch only happens along $l$, which typically requires fabricating isolated, individual capacitors (Fig. 4.3a). Our aim is to create a dense array of sensing elements, for which stretch may occur in multiple directions and hence each sensing element captures changes in area.

### Area changes.

Starting from Eq. (4.1), and assuming volume conservation ($V = V^0 \Leftrightarrow A\,d = A^0 d^0 \Leftrightarrow d^0/d = A/A^0$) and constant stretch throughout the entire sensor cell, the ratio of capacitance before and after deformation can be expressed as

$$\frac{C}{C^0} = \frac{\epsilon_r \epsilon_0 \frac{A}{d}}{\epsilon_r \epsilon_0 \frac{A^0}{d^0}} = \frac{A}{A^0}\frac{d^0}{d} = \left(\frac{A}{A^0}\right)^2. \tag{4.3}$$

Thus, if we know the current capacitance $C$ of a sensor cell and have recorded its rest pose area $A^0$ and capacitance $C^0$, we can compute the change in area between the rest state and the current configuration as

$$\frac{A}{A^0} = \sqrt{\frac{C}{C^0}}. \tag{4.4}$$

**Touch vs. pressure vs. stretch.** We note that there are fundamental differences between capacitive sensing of touch, pressure, and stretch. The majority of the HCI literature on capacitive sensing measure capacitive coupling effects (e.g., changes in capacitance due to an approaching finger). Applied pressure can be measured capacitively since the thickness $d$ is reduced, which leads to a higher capacitance $C$ (see Eq. (4.1)). Finally, in our work, both the overlap area $A$ and the thickness $d$ change due to the deformation of the sensor, requiring a custom read-out scheme (cf. Fig. 4.5). We now explain how a naive implementation, designed for touch or pressure sensing, must be modified in order to capacitively sense deformation.

## 4.3.2 Sensor layout

Dense surface deformation capture requires a sensor that can measure local changes in the surface geometry with high density. This need has to be

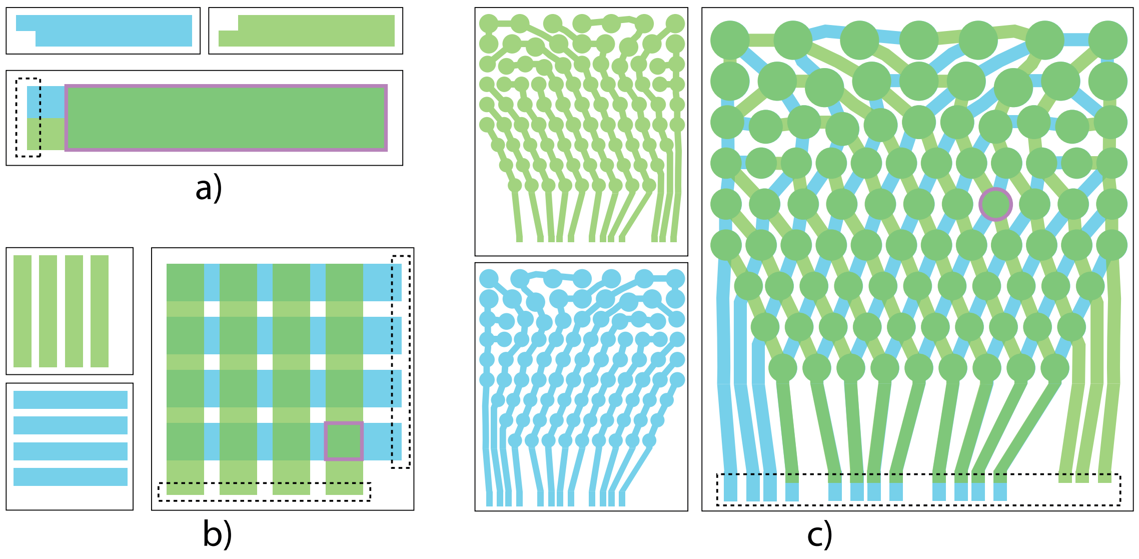


**Figure 4.3:** *Various electrode strip patterns, with the bottom layer in blue and the top layer in green. When overlaid, the overlapping regions form* sensor cells*; we highlight one cell in each example in pink. The dashed lines outline the places where the read-out circuit is connected. Example (a) is a classic elastomer strain sensor with 2 leads and 1 sensor cell; (b) is our array concept with 8 leads and 16 sensor cells; (c) depicts our actual prototype sensor, a warped grid that brings all connection leads to the bottom side of the sensor, with 24 leads and 92 sensor cells.*

balanced with the complexity of the electrical design, so that the fabrication remains feasible. Our proposed concept of the sensor array (Fig. 4.3b), which we call simply *sensor* from now on, strikes this balance with its two-electrode-layers design. The sensor is made of two conductive layers with $n$ and $k$ independent electrode strips on each layer, respectively. We call the individual electrodes *strips*, but they may have any shape. Overlapping sections of two electrode strips from separate layers form a local capacitor, which we call a *sensor cell* $S$. We lay out the strips in a non-uniform grid arrangement, as shown in Fig. 4.3c. Each pair of strips from top and bottom layers crosses at most once, amounting to $s$ sensor cells ($s \leq kn$). This design allows routing all strips to the same side of the sensor, where the silicone-based traces are connected to a PCB for the measurement of capacitances (Fig. 4.4). However, since sensor cells are daisy-chained, we cannot directly read each one independently. We now derive a read-out scheme that provides the desired localized area measurements.

**Sensor read-out.** As mentioned, our sensor is designed to consist of only two capacitive layers, which renders individual addressing of capacitors

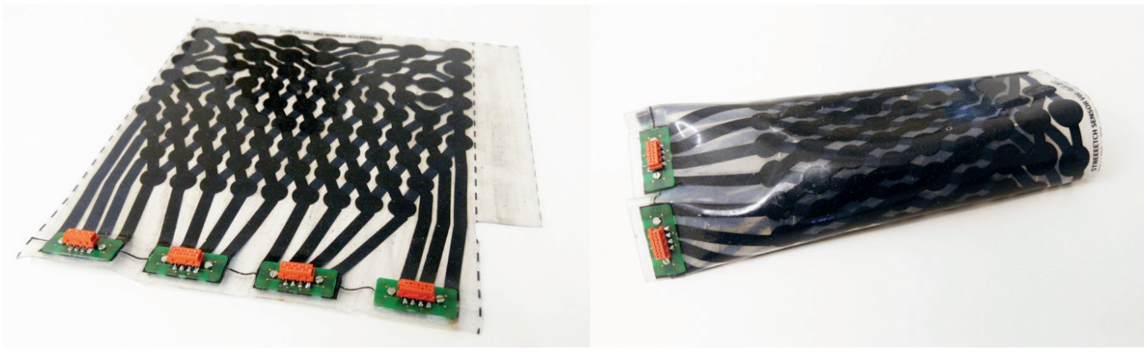


**Figure 4.4:** *Left: Our prototype sensor with connector boards. Both conductive layers contain 12 electrode strips each, and the overlaps amount to 92 sensor cells. Right: Using silicone glue, the topology of a flat sensor can be changed to form e.g. a cylinder. See Fig. 4.14 for our second and larger fabricated sensor.*

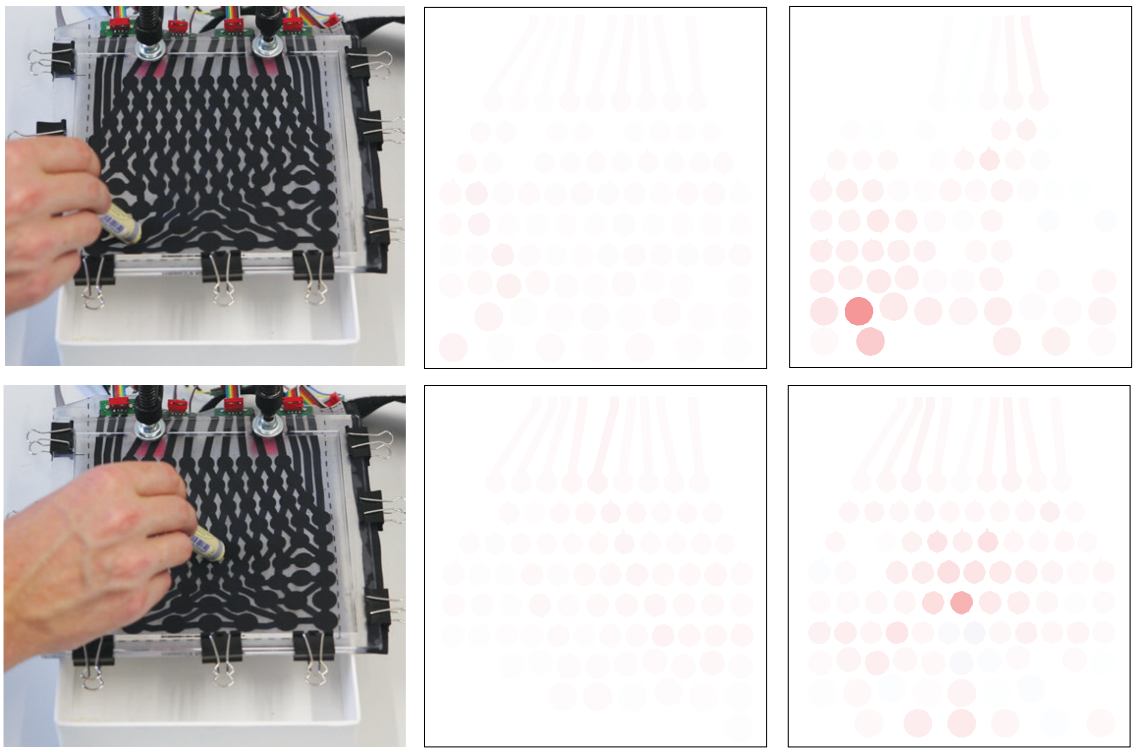


**Figure 4.5:** *A naive scanning scheme (mutual-capacitance approach, using charging time to measure capacitance) results in underestimation of the magnitude of stretch, leads to not well-localized measurements, and even gives incorrect readings. Left: Sensor is deformed by poking with a pen. Middle: Change of magnitude per sensor cell, measured by the naive scanning scheme. Right: Change of magnitude per sensor cell, measured by our proposed scheme (see the respective video clip in supplemental material).*

difficult without sacrificing sensor surface for complex routing of electrical traces. We experimentally verified that simple scanning schemes common in mutual capacitive touchscreens cannot be applied in the case of geometrically deforming and overlapping capacitor plates and traces, see Fig. 4.5. We propose a time-multiplexing scheme, in which a voltage is applied to a subset of strips from both layers in turn, and the remaining strips are connected and serve as the second plate of the local capacitor. A simple example of a sensor composed of a 3×2 grid of electrode strips, with a total of $s = kn = 6$ sensor cells, is shown in Fig. 4.6. For each such measurement, the cells where the combined electrode strips overlap are measured in parallel. The capacitances of these cells add up, leading to a linear relationship between the individual sensor cell capacitances and the measured, combined capacitance. This can be expressed in matrix form:

$$\mathbf{M}\,\mathbf{C}_c = \mathbf{C}_m\,.$$

Here, $\mathbf{M}$ is an $s \times s$ binary matrix with rows encoding different measurement combinations, so that $\mathbf{M}$ transforms the vector of sensor cell capacitances $\mathbf{C}_c$ into the measured capacitances $\mathbf{C}_m$. Using our example in Fig. 4.6 to illustrate the composition of this linear system of equations, the vector $\mathbf{C}_c$ is

$$\mathbf{C}_c = \left[C_{1A},\, C_{2A},\, C_{1B},\, C_{2B},\, C_{1\Gamma},\, C_{2\Gamma}\,\right]^\top, \tag{4.5}$$

where $C_{1A}$ denotes the sought localized capacitance of sensor cell $1A$, and so on. Each row of $\mathbf{M}$ corresponds to a measurement, where the row elements corresponding to jointly read sensor cells are set to 1 and the remaining elements to 0. In our example (Fig. 4.6), the highlighted row of $\mathbf{M}$ corresponds to a measurement where electrodes 1 and $\Gamma$ are connected to serve as the source electrode, and $2, A, B$ as the ground electrode. This leads to cells $1A, 1B$ and $2\Gamma$ to form parallel capacitors, and the read-out values are summed.

To reconstruct $\mathbf{C}_c$ from measurements $\mathbf{C}_m$, the matrix $\mathbf{M}$ needs to be invertible, which is the case if it has $s$ linearly independent rows. The matrix $\mathbf{M}^I$ is formed by iteratively connecting one strip from the top and bottom layer as source electrode, with all remaining strips connected as the ground electrode, resulting in the required $s$ linearly independent rows. We experimentally found that taking additional measurements with all remaining combinations of strips, collected in matrix $\mathbf{M}^{II}$, and solving the resulting over-constrained linear system in the least square sense leads to extra robustness:

$$\mathbf{C}_c = \mathbf{M}^{+}\mathbf{C}_m. \tag{4.6}$$

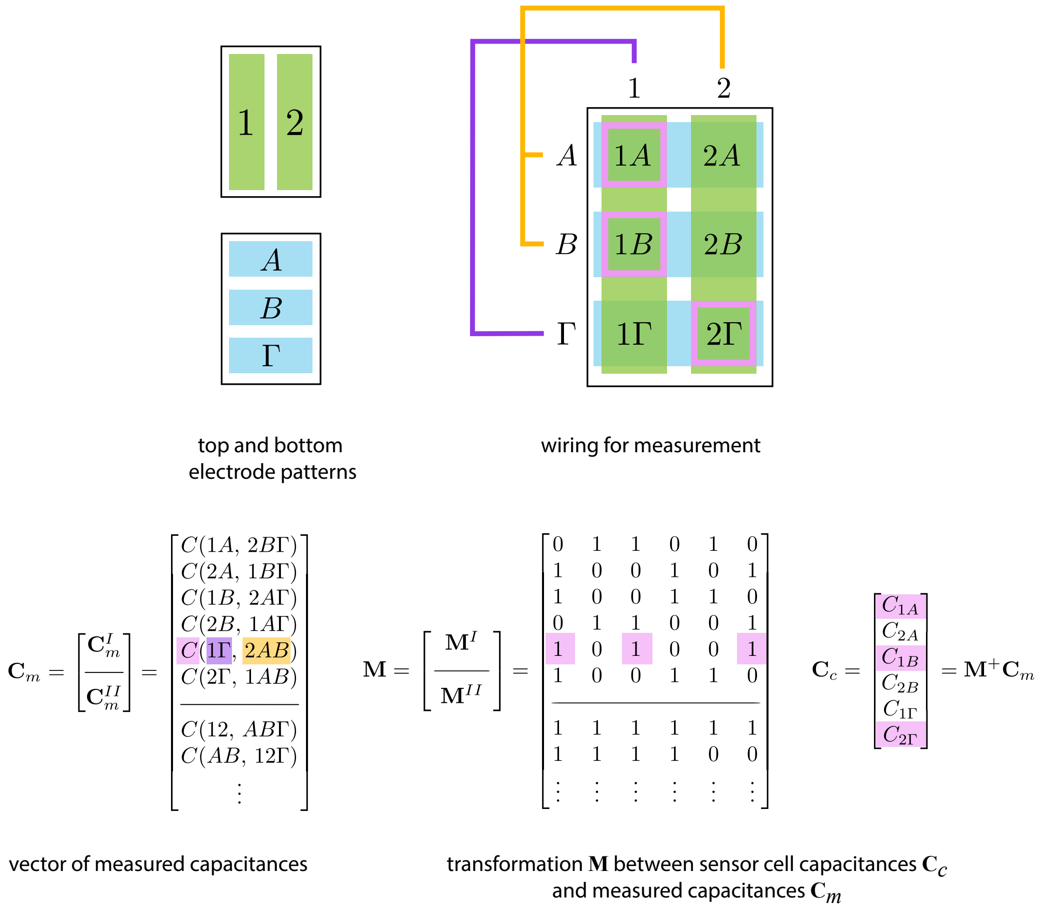

1
2
A
B
Γ
top and bottom electrode patterns
1A
2A
1B
2B
1Γ
2Γ
wiring for measurement
C(1A, 2BΓ)
C(2A, 1BΓ)
C(1B, 2AΓ)
C(2B, 1AΓ)
C(1Γ, 2AB)
C(2Γ, 1AB)
C(12, ABΓ)
C(AB, 12Γ)
vector of measured capacitances
transformation M between sensor cell capacitances C_c and measured capacitances C_m
C_c = M^+ C_m

**Figure 4.6:** *Measuring capacitance of sensor cells via selective combinations of strips. The measured combination in this example is comprised of strips* 1 *and* $\Gamma$ *as the source electrode, and strips* 2*,* $A$ *and* $B$ *as the ground electrode. The resulting overlaps are highlighted in pink. The measurement contributes the equation* $C(1\Gamma, 2AB) = C_{1A} + C_{1B} + C_{2\Gamma}$ *to the linear system that recovers the individual sensor cell capacitances.*

Here,

$$\mathbf{M} = \left[\frac{\mathbf{M}^I}{\mathbf{M}^{II}}\right], \quad \mathbf{C}_m = \left[\frac{\mathbf{C}_m^I}{\mathbf{C}_m^{II}}\right], \tag{4.7}$$

where $\mathbf{C}_m^I, \mathbf{C}_m^{II}$ represent the capacitance readings of the mandatory part $\mathbf{M}^I$ and the additional measurements $\mathbf{M}^{II}$, respectively.

**Non-uniform stretch.** Since our sensor cells have non negligible size (Fig. 4.4), the uniform stretch assumption may not hold in practice. We therefore model a sensor cell $S_j$ more accurately by splitting it into several elements (triangles) $e_i \in S_j$, each with an individual (uniform) area stretch. Applying

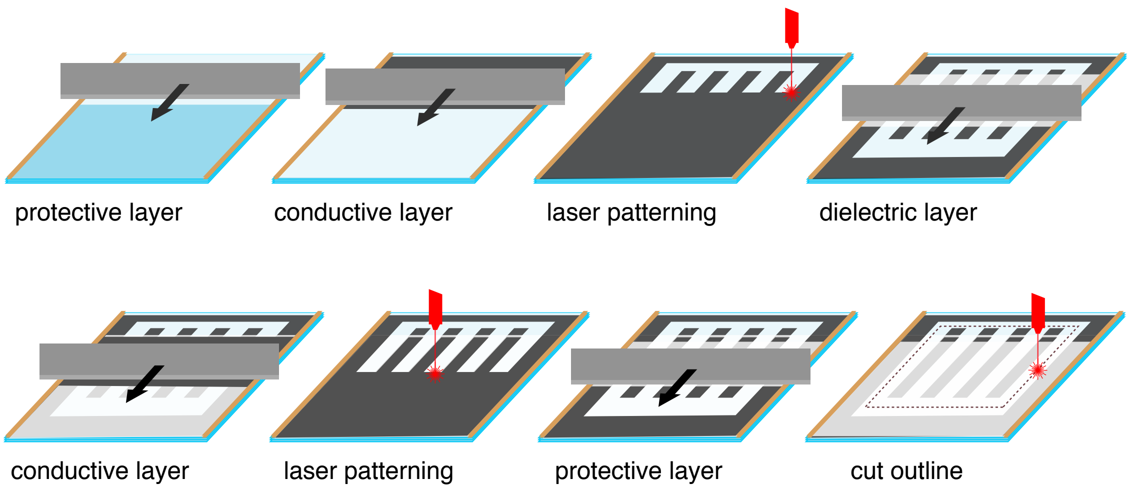


**Figure 4.7:** *The proposed fabrication pipeline consists of eight main steps. Top (from left to right): Casting a protective layer; casting a conductive layer; etching the negative electrode strip pattern with a laser cutter; dielectric layer. Bottom (from left to right): conductive layer; etching again; protective layer; cutting the desired outline.*

Eq. (4.3) to each element, the capacitance $C_j$ of the sensor cell becomes

$$\frac{C_j}{C_j^0} = \frac{1}{C_j^0} \sum_{e_i \in S_j} \left( C_j^0 \frac{A_i^0}{A_j^0} \right) \left( \frac{A_i}{A_i^0} \right)^2 = \frac{1}{A_j^0} \sum_{e_i \in S_j} \frac{{A_i}^2}{A_i^0}, \tag{4.8}$$

where $C_i^0 = C_j^0 A_i^0 / A_j^0$ is the rest pose capacitance of element $e_i$. This holds because in rest state, the thickness $d$ is constant, and hence the rest state capacitance is proportional to the area $A_i^0$.

### 4.3.3 Fabrication

We propose a fabrication pipeline, illustrated in Fig. 4.7, for silicone-based sensors with arbitrarily shaped electrodes.

**Structure.** The sensor consists of two conductive layers with a dielectric layer between them, and it is encased by shielding layers (see inset on the previous page). During fabrication the sensor rests on a flat glass plate to which the silicone elastomer sticks well but the final sensor can be easily detached. We provide the description of the chemical composition of the silicone mixtures in Appendix B.1. The layers are cast one by one by spreading the silicone using a blade; the correct thickness is ensured by Kapton tape

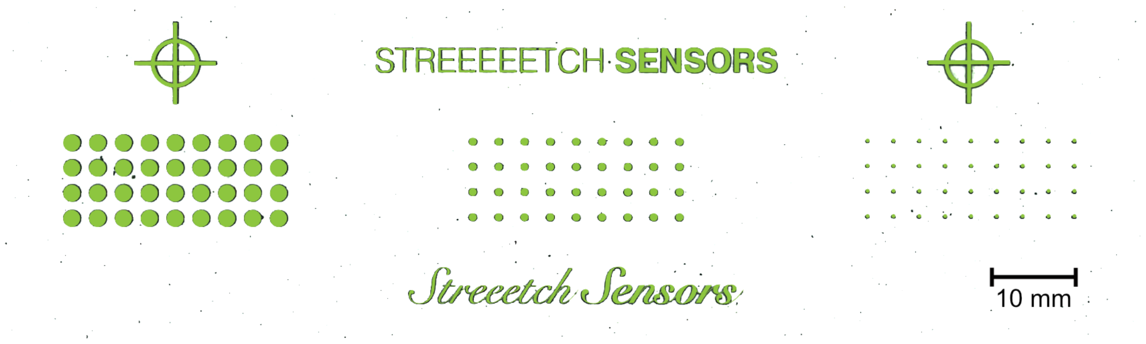


**Figure 4.8:** *To demonstrate the alignment quality of our fabrication method, we produced a test pattern with two identical conductive (black) layers. The fabricated pattern was scanned with a flatbed scanner. The scan is overlaid with the digital design (green). Wherever the alignment is perfect, only the green layer is visible.*

(65 $\mu$m thickness) at the borders of the glass plate. After the casting of each layer the sensor is cured for 20 minutes in an oven at 100 °C.

The second, conductive layer (silicone mixed with carbon black) is directly cast onto the shielding layer, and after curing, the desired pattern is etched with a laser cutter. The etching is done with a 100 Watt Trotec Speedy 360 laser cutter. Two rounds of etching are carried out with the following settings: 20 Power, 60 Speed and 500 Pulses/inch. This vaporizes the carbon black to create non-conductive areas between traces, while the underlying silicone-only layer stays intact. The resulting dust can be carefully removed with isopropyl alcohol without damaging the electrodes. The sensor is completed by adding another dielectric, the second capacitive layer (which is also etched and cleaned) and finally another shielding layer. The overall process takes around 3.5 hours (1 h for mixing and casting, 1.5 h for curing and 1 h for laser etching) for producing a sensor of 200$\times$200 mm.

In previous works [Lu et al. 2014; Araromi et al. 2015], the alignment of the different layers of a multilayer sensor had to be done manually. Aligning the layers with high accuracy and without wrinkles can prove a difficult task, especially for larger sensors like ours. With our approach, a high alignment quality is achieved by design, since we directly cast layers onto one another (see the accompanying video[1] from 01:05) and place the base glass plate in the laser cutter aligned with physical stoppers before etching. Fig. 4.8 shows an alignment experiment.

[1]`https://igl.ethz.ch/projects/deformation-capture-sensor-arrays/Deformation-Capture-Sensor-Arrays-2019.mp4`

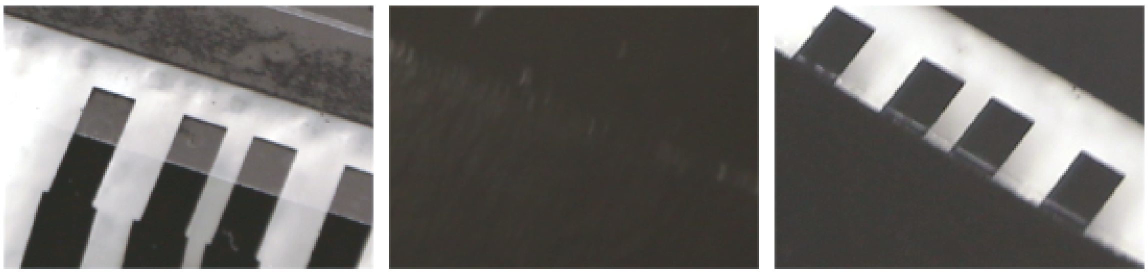
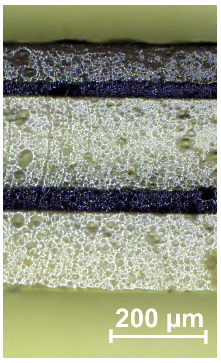


**Figure 4.9:** *Left: sensor after casting the dielectric layer, the connector pads are covered by transparent sticky tape. Middle: after casting the second conductive layer. Right: after removal of the sticky tape (before curing in the oven); the connector pads stay exposed.*



The thickness of the final sensor is about 500 µm, the conductive layers are 45 µm thick each (for the basic protective layer we use 4 layers of offset tape, and for the dielectric layer 2 layers of offset tape). The inset on the right shows a cross section of the sensor layers under a microscope. The sheet resistance of a conductive layer is in the order of 1 kOhm (four-point probe). The stiffness (Young's Modulus) of the pure layered RTV is 729.6±13.4 kPA, with two embedded conductive layers 979.6±16.6 kPA (calculated from three samples each with the setup and method as described in [Hopf et al. 2016]).

**Connectors.** The electrode strips must be connected to our electronic boards for measurement (see Appendix B.2 for details). During fabrication we cover the connectors with sticky tape before casting the remaining layers. The tape is removed before curing the corresponding layer, re-exposing the connectors, see Fig. 4.9.

**Finalization.** The sensor is cut to the desired outline shape with the laser cutter. The resulting sensor is then pulled off the glass plate, and silicone adhesive can be optionally used to close the sensor to form, for example, a cylinder (Fig. 4.4) to wrap a wrist or an elbow.

## 4.4 Surface deformation reconstruction

Our sensor is equipped with simple rest state geometry, represented by a triangle mesh $\mathcal{S} = (\mathcal{V}, \mathcal{F})$, where $\mathcal{V}$ is the set of 3D vertex positions and $\mathcal{F}$ is the connectivity (the set of faces). The connectivity $\mathcal{F}$ comes from meshing

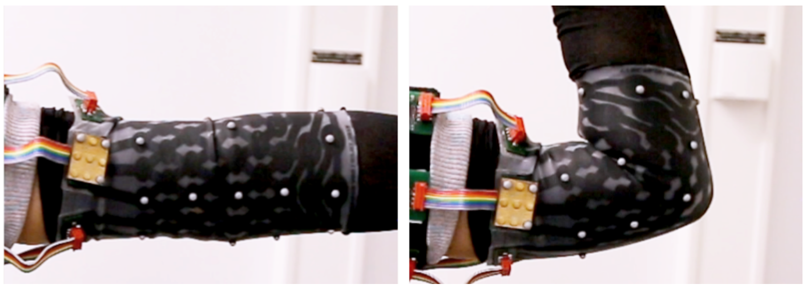


**Figure 4.10:** *Our sensor on an elbow. Left: rest pose; right: close to fully bent.*

the electrode layout (Sec. 4.3.2): we represent each sensor cell $S_j$ with a fan of triangles and mesh the overall layout using Delaunay triangulation using [Shewchuk 1996]. We set the rest state geometry $\mathcal{V}^0$ to the canonical shape corresponding to the chosen topology: e.g., for the sensor in Fig. 4.4 (right), we use a circular cylinder of dimensions corresponding to the intrinsic size of the produced sensor. As the sensor is pulled onto a deforming object and capacitance changes are measured, the goal is to reconstruct the deformed geometry $\mathcal{V}(t)$ for each frame $t$, given the measured capacitances $C_j(t)$ of all sensor cells $S_j$.

Through the relation of capacitance to area (Eq. (4.8)), our sensor provides rich, localized area change measurements at interactive frame rates, but areas alone are not sufficient to define the shape of a general deforming surface in 3D, since area is an intrinsic quantity. We therefore pair these measurements with a data-driven geometric prior, acquired by simultaneously capturing the deformation of the object of interest using our sensor and an optical tracking system, and then training a regressor that maps the capacitance measurements to marker vertex positions.

To this end, we define a sparse set of vertex indices $\mathcal{M}$ and attach reflective markers onto the corresponding physical locations. To simplify the marker attachment process, the set $\mathcal{M}$ is a subset of the mesh vertices corresponding to centers of circular sensor cells. The set is chosen to obtain a regular coverage of the cylindrical sensor, allowing a maximal distance of 5 centimeters in-between the individual markers. For all experiments we used a single, fixed marker pattern per sensor layout. Placing the sensor onto the object of interest (Fig. 4.10), we simultaneously record sensor readings and 3D marker positions tracked by an 8-camera OptiTrack setup [Opt 2018]. Untreated silicone is highly specular, but we found that a matte finish can be attained by densely etching the outer layer on the laser cutter (with 60 Power, 100 Speed, and 500 Pulses/inch). The captured and processed data for each frame $t$ consists of:

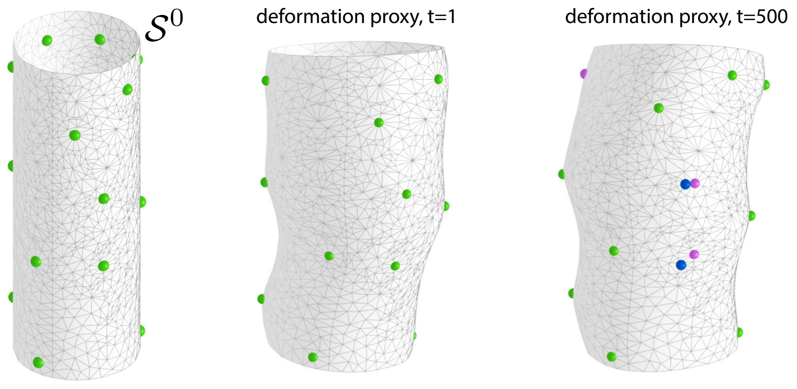


**Figure 4.11:** *Left: The rest state sensor mesh $\mathcal{S}^0 = (\mathcal{V}^0, \mathcal{F})$ with marker vertices $\mathcal{M}$ in green. Middle: $\mathcal{S}^0$ deformed by the marker positions of the first frame in a wrist capture session. Right: the labeled markers in green, two unlabeled marker observations in blue and the two candidate matches in pink; the mesh geometry is estimated by elastically deforming $\mathcal{S}^0$ using the green markers as positional constraints.*

- Coordinate frame transformation $\mathbf{T}(t) \in \mathcal{C}^{3\times4}$ (a $3 \times 3$ rotation and a translation, recovered from 3 designated markers);
- Marker positions $\mathbf{p}_i(t) \in \mathcal{C}^3$ w.r.t. the local frame, for each marker vertex $i \in \mathcal{M}$;
- A vector $\mathbf{C}_c(t)$ of capacitance values of all sensor cells, obtained as described in Sec. 4.3.2.

This data is used to train a regressor $g_\theta(\mathbf{C}_c(t))$ that maps sensor cell capacitance values to marker vertex position estimates $\hat{\mathbf{p}}$. Given $g$, we can employ the sensor at run-time and use the marker positions predicted by $g$ as positional constraints that guide the deformation of the sensor mesh $\mathcal{S}$.

### 4.4.1 Capturing and processing training data

A fundamental challenge with marker based approaches are incorrectly labeled or lost markers, an issue exacerbated in settings like ours, where heavy occlusions and strong non-rigid deformations are combined with the lack of a simple skeletal prior. Fig. 4.12 provides an illustrative example of tracking 12 wrist-mounted physical markers. The OptiTrack system outputs 165 individual marker observations due to frequent tracking failures (sequence length is 1.5 minutes). This problem quickly becomes unwieldy; in capturing real data we encountered more than 500 marker labels in a dataset of 17000 frames (3 minutes) of 21 physical markers.

Manual cleanup, label merging, and correct attribution would require hours of manual labor and make the acquisition of our deformation prior impractical. We therefore employ a novel semiautomatic marker cleanup and labeling pipeline.

The mocap system outputs a set of marker labels $\mathcal{I} = \{1, 2, \ldots, N\}$, and for each frame $t$, a binary indicator that tells whether the marker was visible in that frame. For each frame $t$ where marker $j$ is visible, the system also outputs its 3D position $\mathbf{p}_j(t) \in \mathcal{C}^3$. We seek an assignment of marker vertices $i \in \mathcal{M}$ to tracked marker labels $j \in \mathcal{I}$, providing a 3D position in each frame $t$. Our main insight is to employ a state-of-the-art elastic deformation technique to create a proxy deformation of $\mathcal{S}^0$, using reliably labeled marker vertices as positional constraints. This allows us to match each unlabeled marker to its closest marker vertex on the proxy.

**Initialization.** Usually the number of tracked labels $N$ is much larger than the actual number of physical markers, because some markers are temporarily lost, and are then given a new label when they re-enter. We initialize the assignment of marker vertex indices by picking 3 tracked markers in the first frame and manually matching them with their corresponding mesh vertices in our rest pose mesh $\mathcal{S}^0$. We then rigidly transform $\mathcal{S}^0$ to align it with the tracked data (i.e., put it in the same coordinate system) by solving the Procrustes problem. We then assign a 3D position to all remaining marker vertices of the mesh by searching for the closest tracked marker position in this frame. This way we obtain $|\mathcal{M}|$ pairings between marker labels and mesh vertex indices, as typically in the first frame (rest pose) all markers are visible.

**Labeling.** We sort the unassigned tracked markers in chronological order according to the first frame they are visible at. For each unassigned marker $j^*$ and for each frame $t$ where $j^*$ is visible, we elastically deform $S^0$ to match the captured geometry in $t$ by imposing the marker vertices in $\mathcal{M}$ that already have matched marker positions in frame $t$ as positional constraints. The output is a set of deformed "proxy" meshes, one for each such frame, which we use to find a match for $j^*$. For robustness, we pick the mesh vertex whose average $L^2$ distance over all frames is the smallest. We accept the match only if this distance is below a threshold $\tau$ (25 mm in our experiments), otherwise $j^*$ is marked as an outlier.

Every successful labeling provides an extra positional constraint for the deformations, improving the quality of the proxy (and thus the success

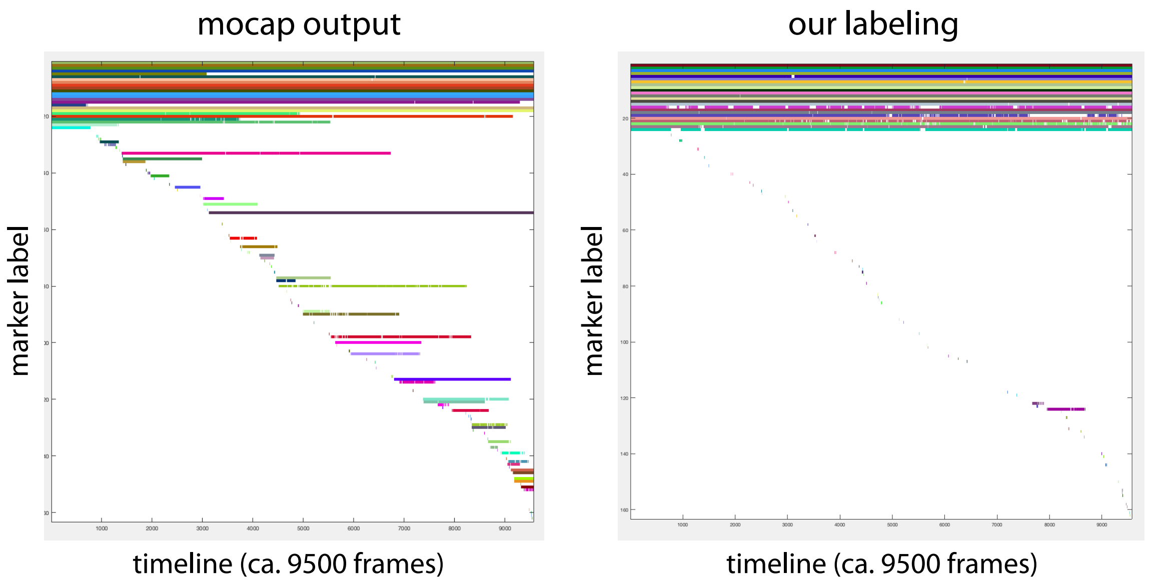


**Figure 4.12:** *Marker labeling. For each individual label, we plot horizontal bars spanning the frames where it is visible. Left: Captured markers directly from the mocap system. There are 165 individual labels due to periods of occlusion and subsequent failure to pick up the track, despite the actual number of markers being only 12. Right: sanitized and relabeled markers using our semiautomatic approach. A minority of outliers remain in a few frames; they are discarded from the dataset.*

rate) for subsequent labeling passes. In our implementation, we use the deformation optimization method by Wang et al. [Wang et al. 2015], a state-of-the-art nonlinear elastic deformation technique that expects solely sparse positional constraints as input.

As a post-processing step, we visually inspect the produced assignments via 3D renderings and plots of $x, y, z$ coordinates over time, to detect incorrect merges. If any are present, we can separate them and rerun the labeling algorithm again. One iteration of this procedure was sufficient for most of our capture sessions.

Our MATLAB implementation takes below 15 minutes per session, allowing us to have a 3 minutes long captured session cleaned in around 10 minutes. Note that we are not guaranteed to find observed 3D positions for each marker vertex of our mesh in each and every frame $t$, due to occlusions, outliers and possible failures of our assignment heuristic. We thus discard frames with unassigned markers, which are around 20 % in our acquisition sessions. We encountered one case where too many markers were missing in some frames due to heavy occlusions in the folded elbow, which hampered the regressor training due to insufficient data. We resorted to synthetic

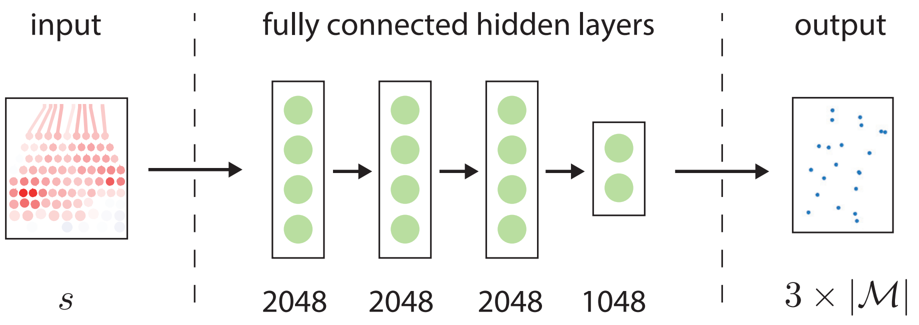


**Figure 4.13:** *To train a sensor with s sensor cells and $|\mathcal{M}|$ markers, our network takes s capacitance readings as input and outputs $|\mathcal{M}|$ vertex position estimates, through three fully connected layers with 2048 units each and one fully connected layer with 1024 units. E.g. for our sensor in Fig. 4.4 there are 92 inputs and 63 ($3 \times 21$) outputs.*

3D data for those frames, taking missing marker vertex positions from the deformation proxy.

### 4.4.2 Regressor training

We wish to recover dense surface deformations in real time. To this end, we learn a function $g_\theta(\mathbf{C}_c)$, parametrized by a deep neural network, that maps from sensor cell capacitances $\mathbf{C}_c \in \mathcal{C}^s$ to marker positions $\hat{\mathbf{p}} \in \mathcal{C}^{3\times|\mathcal{M}|}$ (in a local frame). We have experimentally verified that nonlinear function approximators such as the fully connected multi-layered neural network used here, perform better than linear models due to the nonlinearities in the mapping from area change to capacitance (Table 4.1).

Our network architecture, depicted in Fig. 4.13, takes $s$ sensor cell capacitance readings as inputs of a linear layer, followed by three fully connected layers with 2048 units each and one fully connected layer with 1024 units. A final linear output layer predicts the marker vertex positions $\hat{\mathbf{p}}$. The input and all hidden layers are followed by a ReLu activation function and a BatchNorm layer. Given a training set $\mathcal{D} = \{(\mathbf{C}_c^i, \mathbf{p}^i)\}$ of $K$ vectorized ground truth input-output pairs, we perform training via a weight-regularized $L^2$ loss:

$$\mathcal{L}_{\text{reg}} = \sum_{i=1}^{K} \left\| g(\mathbf{C}_c^i) - \mathbf{p}^i \right\|_2^2 + \lambda \left\| \theta \right\|_2^2 , \tag{4.9}$$

where $\theta$ are the model parameters and $\lambda$ is a regularization factor.

We implement the network using pyTorch [Paszke et al. 2017] and train it with the ADAM optimizer with a learning rate of $10^{-4}$, mini-batch size of 256, regularization $\lambda = 10^{-5}$ and default values for all other parameters [Kingma and Ba 2014]. All inputs are normalized to be zero-mean unit variance.

### 4.4.3 Capturing dense surface deformation at runtime

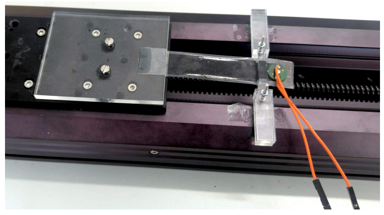
Once the neural network is trained and the regressor $g_\theta$ is available, we can deploy our sensor standalone, uncoupled from the optical tracking and estimate the dense surface deformation of an object without line-of-sight. This is illustrated in Fig. 4.2, where the sensor is worn underneath clothing, rendering vision based approaches infeasible. The regressor provides 3D positions $\hat{\mathbf{p}} = g_\theta(\mathbf{C}_c)$ of the marker vertices $\mathcal{M}$ given current sensor measurements $\mathbf{C}_c$. We note that the network is able to compensate for inaccuracies in area estimates from capacitive readings (see Fig. 4.19), which in particular occur under extreme stretch (see Sec. 4.5.2). To reconstruct the current surface deformation, we deform the rest state mesh $\mathcal{S}^0$ using the method proposed by Wang et al. [Wang et al. 2015], where the marker vertices $\hat{\mathbf{p}}$ again serve as positional constraints.

## 4.5 Experiments and results

To demonstrate the utility of our proposed approach, we evaluate its components in an ablative manner. First, we quantitatively assess the sensor concept and the corresponding fabrication method (Sec. 4.5.1) and then demonstrate the applications in reconstruction of surface deformations, both qualitatively and quantitatively (Sec. 4.5.2). Our experiments are performed with two sensor layouts, shown in Figures 4.4 and 4.14. The layouts are manually designed, non-uniform grids, with all strips routed to the same side of the sensor, where they are connected to a connector PCB. The first layout is used both in its flat form and as a cylinder.

### 4.5.1 Sensor characterization

**Distance sensor comparison.** We verify the accuracy of our sensors by fabricating (15×50 mm) a uni-axial sensor with the same dimensions as a commercially available Parker Hannifin industrial sensor [Par 2018]. We stretch both sensors (with a motorized linear stage, see inset) to various lengths and directly compare the



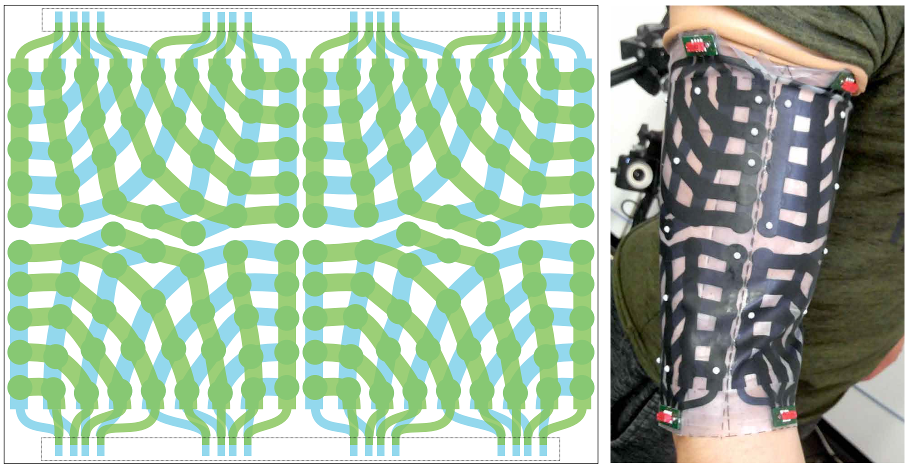


**Figure 4.14:** *We fabricated a second larger sensor (300x250 mm) with 144 sensor cells and connectors on two sides. Left: The sensor layout consists of four identical sub-sensors that can be read out in parallel. Right: The produced sensor, glued to form a cylinder and worn on a biceps.*

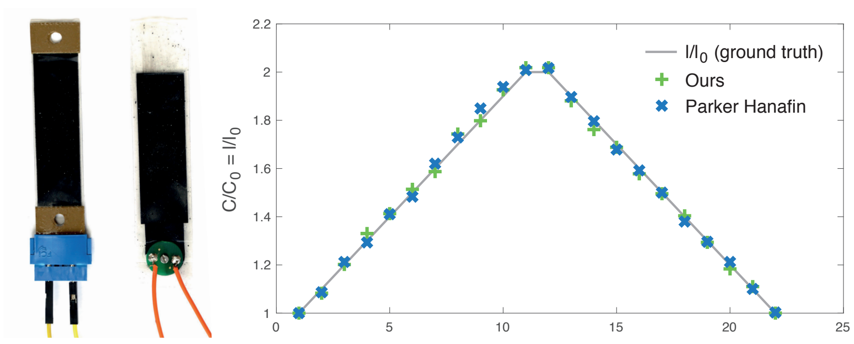


**Figure 4.15:** *Left: An industrial sensor by Parker Hanafin and a sensor of the same dimension fabricated by us. Right: Comparison of their accuracy.*

readings. The average relative error of the two sensors is comparable (Fig. 4.15), with a slight but non-significant edge for the Parker Hanafin sensor (0.0085) over ours (0.0096). Overall, we conclude that the accuracy of our measurements is high and comparable to commercial solutions. We note that there was no observable hysteresis in our experiments.

**Longterm sensor behavior.** In a second set of experiments, we evaluate whether and how the sensor response changes under longterm cyclic stretch and large stretch. For the longterm experiment, the uni-axial sensor is pre-stretched a few times and then continuously stretched and relaxed for 5 h 30 min by a factor of 2x. The sensor response stays constant (see Fig. 4.16). The

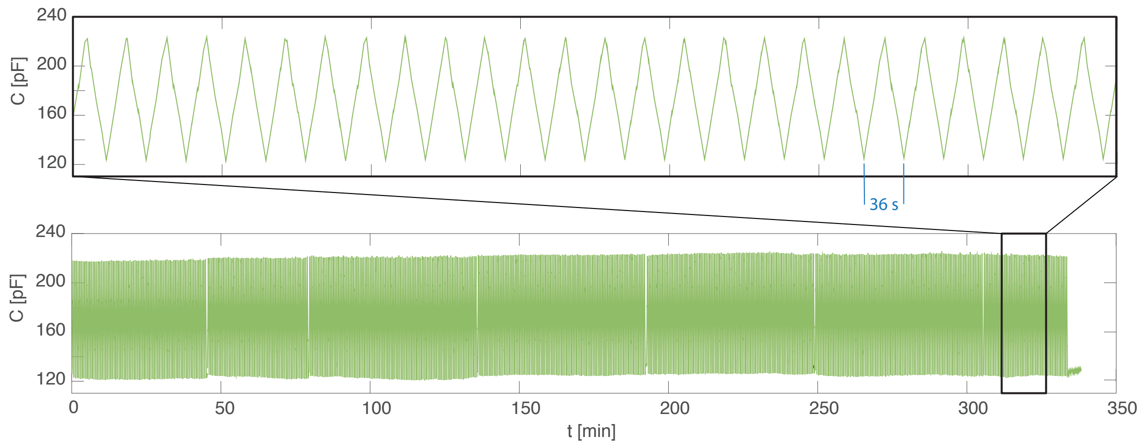


**Figure 4.16:** *The uni-axial sensor response stays constant during a cyclic stretch (2x) test of 5 hours and 30 minutes (about 550 cycles).*

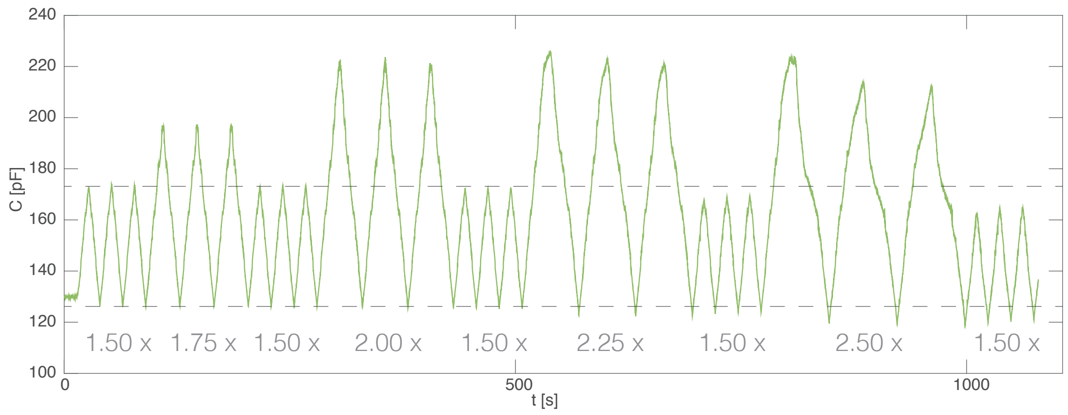


**Figure 4.17:** *After a stretch factor of 2.25x, the sensor response when stretched by a factor of 1.5x has changed compared to the first three rounds.*

maximally allowed stretch before (internal) material damage occurs is found by stretching the sensor a few times to a baseline factor of 1.5x, increasing the maximum stretch factor in each round (see Fig. 4.17). These experiments show that our fabricated sensor can be stretched without noticeable internal damage by 100% (2x) for at least 5 h 30 min. In our experiments, this stretch factor was never surpassed when capturing body parts.

**2D stretch localization.** To assess the localization capabilities of our sensor layout, we perform a simple experiment, in which we fix a flat sensor to a frame and poke it in different locations. Eq. (4.4) states that the sensor cells' capacitance changes directly relate to area changes. The proposed readout scheme (cf. Sec. 4.3.2) allows us to measure and localize stretch. Fig. 4.18

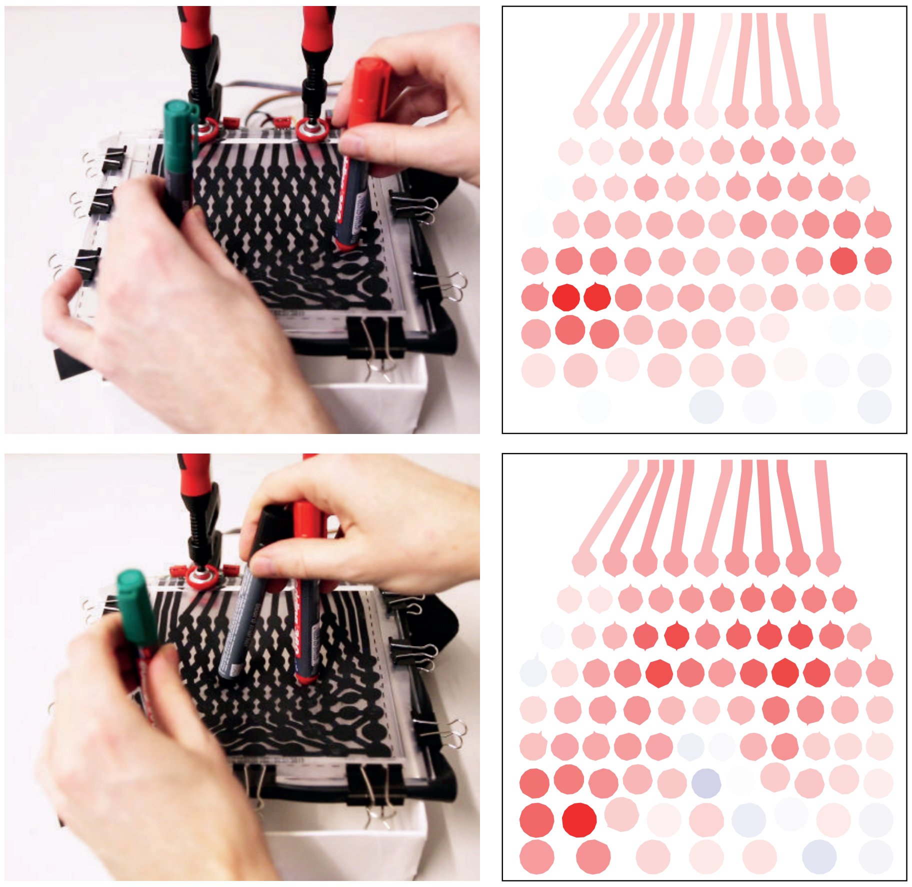


**Figure 4.18:** *Left: the sensor is fixed to a frame and poked with pens. Right: Area change magnitude measured per sensor cell.*

visualizes two example frames extracted from the video in the supplemental material. This capability could be explored in other application scenarios, including detection of touch and pressure.

**2D stretch quantification.** To better understand the accuracy of recovered stretch measurements, we attach clips on strings to a flat sensor, so that we can apply spatially varying tension forces by selectively pulling on the strings. Additionally, we place reflective markers on the sensor, so that we can estimate the actual stretch per sensor cell. Fig. 4.19 visualizes the results. We report an average relative error of 7.7% when comparing the measured capacitance ratio $C_c/C_c^0$ with the theoretical capacitance ratio calculated by Eq. (4.8) per sensor cell from the tracked areas. This error is likely due to our approximate sensor model, which neglects the influence of the (changing) resistance of the electrodes. Close inspection of Fig. 4.19 reveals that this effect is negligible for our purposes.

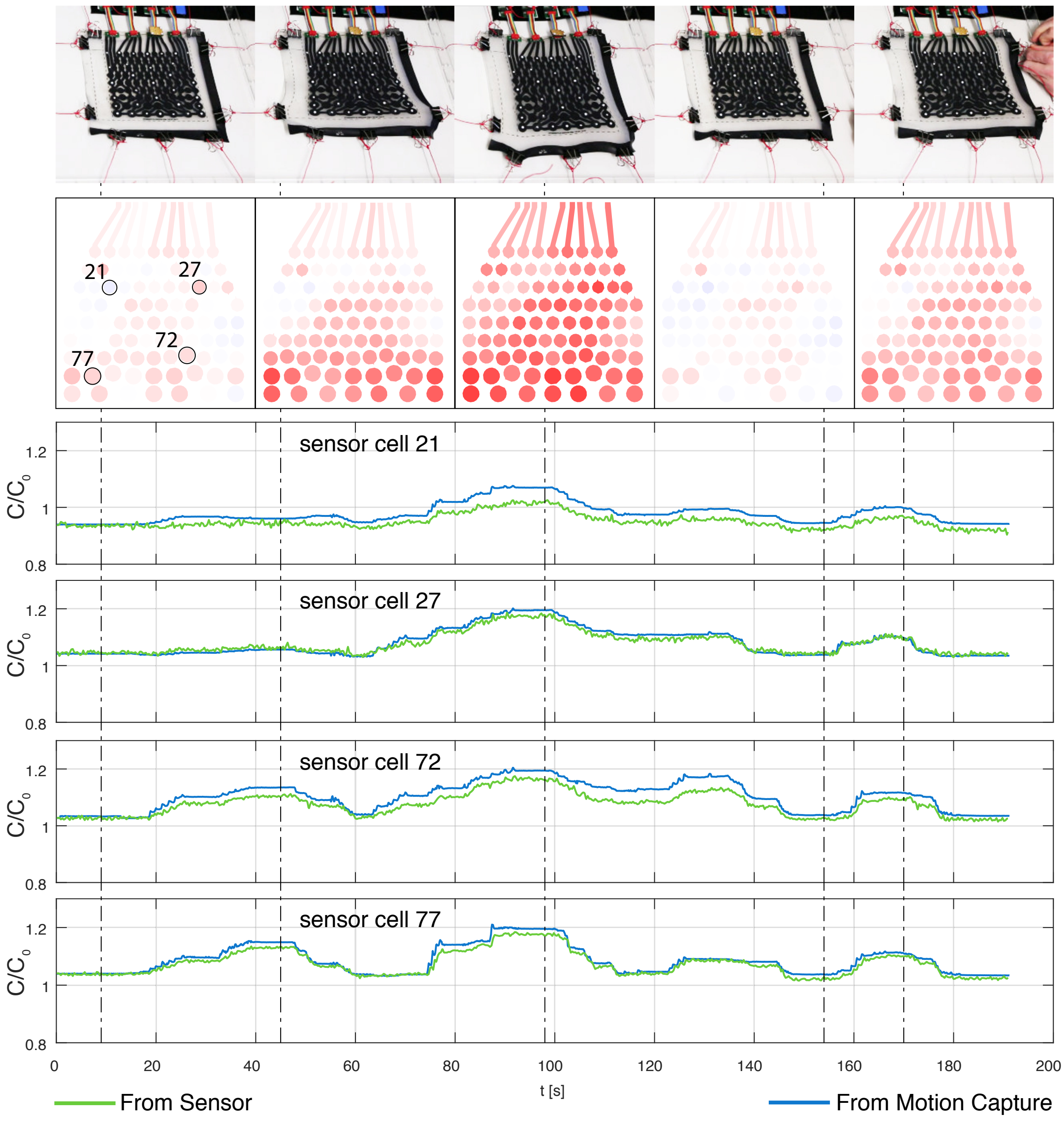


**Figure 4.19:** *The sensor is dynamically stretched by selectively pulling on the strings its attached to. Top: A set of sample frames. Middle: Stretch intensity per cell at the sample frames. Bottom: The relative capacitance of four selected sensor cells over time, comparing ground truth (estimated through mocap markers) in blue and the capacitance change recorded by our sensor in green. The dashed vertical lines show the locations of the sample frames on the timeline.*

| Marker error | | mean | std | max |
|---|---|---|---|---|
| Balloon | LR | 3.59 | 1.90 | 12.84 |
| | SVM | 3.22 | 2.73 | 25.05 |
| | *ours* | **2.75** | **1.86** | **12.85** |
| Biceps | LR | 7.64 | 5.06 | 53.00 |
| | SVM | 6.86 | 5.18 | 52.24 |
| | *ours* | **3.85** | **2.39** | **25.81** |
| Elbow | LR | 7.65 | 3.31 | 39.95 |
| | SVM | 6.73 | 5.26 | 59.79 |
| | *ours* | **3.46** | **2.48** | **30.82** |
| Wrist | LR | 12.8 | 4.99 | 71.89 |
| | SVM | 4.36 | 2.71 | 44.12 |
| | *ours* | **3.51** | **2.14** | **27.22** |
| Forearm | LR | 10.64 | 3.94 | 52.03 |
| | SVM | 4.38 | 2.30 | 32.11 |
| | *ours* | **4.02** | **2.66** | **38.52** |

**Table 4.1:** *Comparison of prediction accuracy of the chosen DNN regressor (*ours*) with a linear regression model (LR) and a non-linear support vector machine with an RBF kernel (SVM). All errors are in millimeters, lower is better.*

### 4.5.2 Surface deformation capture

**Predictor comparison.** To validate our design choice of parameterizing the regression problem of Eq. (4.9) with a neural network, we perform a comparison with several alternative models as baseline. Table 4.1 summarizes the results of a three-way comparison with linear regression and non-linear SVM using an RBF kernel. The neural network achieves the lowest mean and max errors and produces the lowest standard deviation across all datasets used in our experiments.

**Non-skeletal 3D deformation.** To demonstrate the deformation capture abilities of our sensor, we use it to measure the shape of a balloon that is aperiodically inflated (up to a maximum diameter of about 120 mm) and deflated. Despite the apparent simplicity of the setup, the deformation is freeform, and it is not possible to rely on standard geometric priors, such as a skeleton. We captured a 5-minute session with the mocap system (2451 frames), and used the cleaned data to train a regressor (Sec. 4.4.2). To validate the system, we recorded an additional 1:40 min sequence (946 frames). The errors between our regressor and the mocap output are small, 2.75 mm on average, with a maximum of 12.85 mm (Fig. 4.20, rightmost column). Note that the maximal resolution of our mocap system, which is used as ground truth for these measurements, is 0.2 mm. Fig. 4.20 shows four frames extracted from the video in the supplemental material.

As a non-skeletal body part example, we captured a biceps muscle of ca. 36 cm in circumference being flexed, together with a small part of the elbow, using a larger sensor (see Fig. 4.14). We captured a 6-minute training session with the mocap system (2305 frames) and an additional 2 min test sequence (1224 frames). We report an average marker error of 3.85 mm, with a maximum of 25.81 mm (Fig. 4.21, rightmost column). Fig. 4.21 shows four frames (extracted from the video in the supplemental material).

**Uni-axial deformation.** We wrap our sensor around an elbow to capture its movement. This is a challenging scenario due to the strong occlusions when the elbow is fully bent and due to the local non-rigid surface deformation. We use 12 minutes of training data (5369 frames) and a 2-minute test sequence (1329 frames). Our sensor accurately matches the test sequence (Fig. 4.22) and enables deformation sensing even when worn below clothing (Fig. 4.2). In this example, the mean error is 3.46 mm and max error is 30.82 mm. In Fig. 4.22 we show four frames extracted from the full video sequence.

**Multi-axial deformation.** Our sensor successfully reconstructs very challenging scenarios, such as a wrist movement containing both a multi-axial skeletal deformation and volume changes when the fingers are splayed. For the wrist example, we trained on a 15-minute session (8799 frames), and tested on a 2:45 minutes session (1774 frames). Even in this case, the errors are low, with a mean of 3.51 mm and max error of 27.22 mm (see Fig. 4.23).

**Twisting motions.** The sensor also manages to capture the twisting motion of a forearm. For this example the model is trained on a 8-minute session (1846 frames), and evaluated on a 2 minutes session (1320 frames). For such a scenario the errors are slightly higher with a mean error of 4.02 mm and max error of 38.53 mm, (see Fig. 4.24). The peak in error corresponds to predictions of the markers on the hand when the wrist is fully bent, see Fig. 4.24 on the right.

**Interpolation behavior.** To demonstrate the robustness of our predictor in test situations with strains deviating from the training data, we artificially

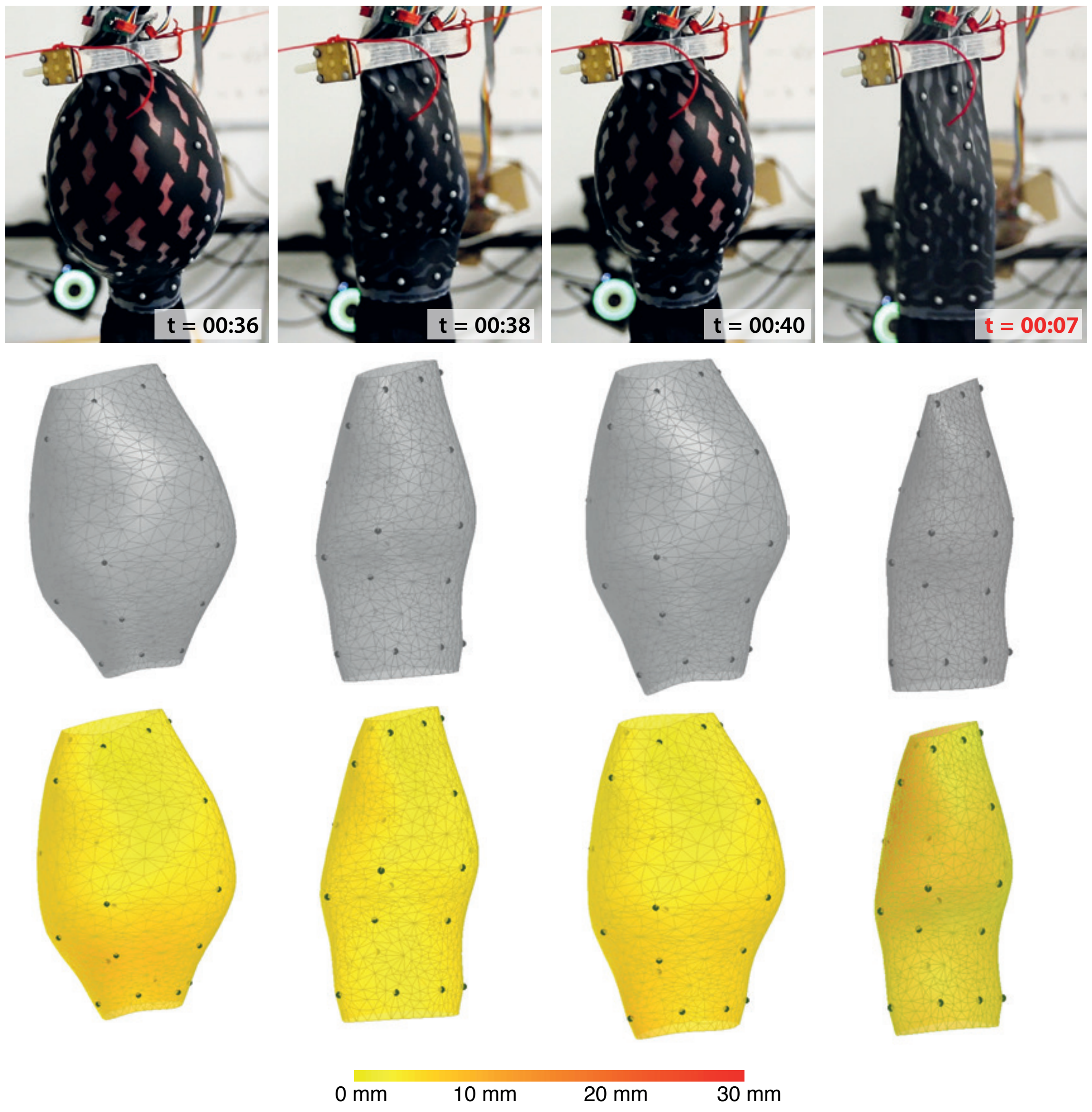


**Figure 4.20:** *Four frames of a 1:40 min long balloon capture session. Top: Video frames for comparison. Middle: Mocap ground truth. Bottom: Reconstruction based on the sensor measurements and the trained prior. The rightmost frame corresponds to the frame with the largest individual marker error.*

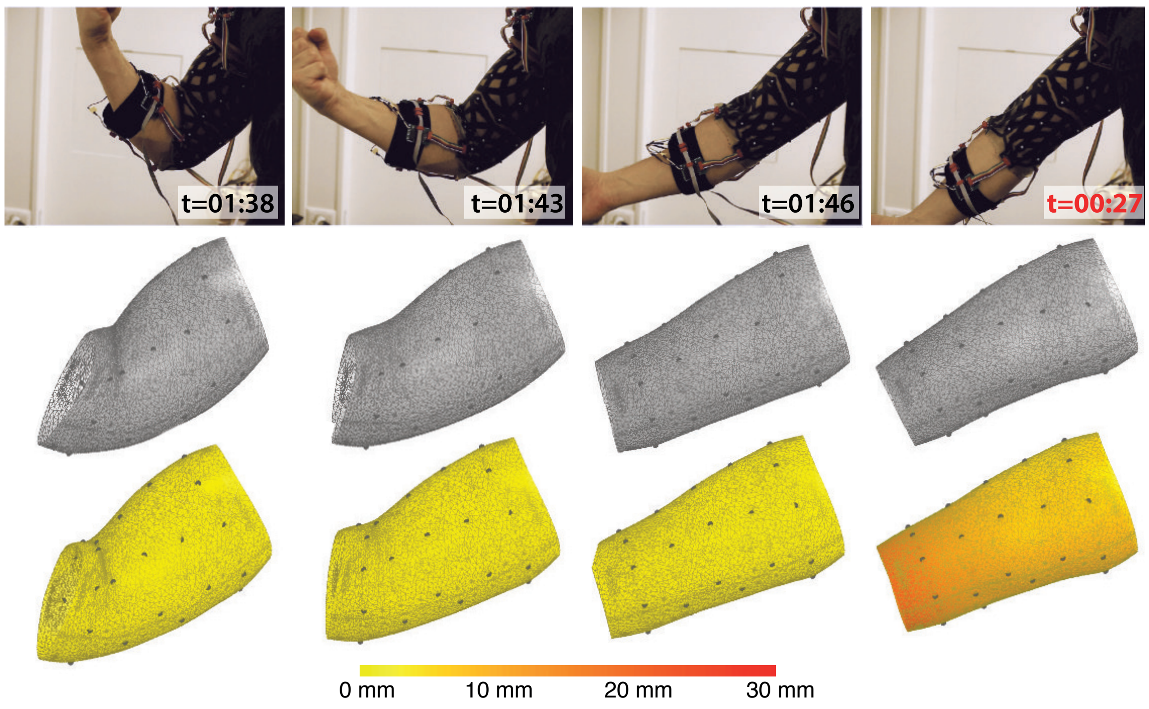


**Figure 4.21:** *Four frames of a 2-minute long biceps capture session. Top: Video frames for comparison. Middle: Mocap ground truth. Bottom: Reconstruction based on the sensor measurements and the trained prior. The rightmost frame corresponds to the frame with the largest individual marker error.*

reduce the training data of the wrist example, while keeping the test set fixed. We only keep training frames where the angle $\alpha$ between the arm and the palm is $\alpha < \gamma$ or $\alpha > \beta$ ($\alpha$ is the angle between a line connecting two markers on the arm and another line connecting two markers on the back of the hand). Table 4.2 shows the remaining number of training frames and the resulting mean and maximum error for a selection of angular limits. The first block (where frames with large angles are removed) shows that the network does not extrapolate well. Note that this is to be expected, since most machine learning approaches do not generalize well to situations where the training and test data statistics differ significantly. However, as shown in the middle and the lower block, the method manages to interpolate well, even though there are now training samples at shallow angles. This holds true as long as the training set is large enough. The last row of Table 4.2 shows the results of exceeding this lower limit in terms of training data size.

**Real-time reconstruction.** To demonstrate the real-time capabilities of our approach, we have implemented a live system in which a user may wear the

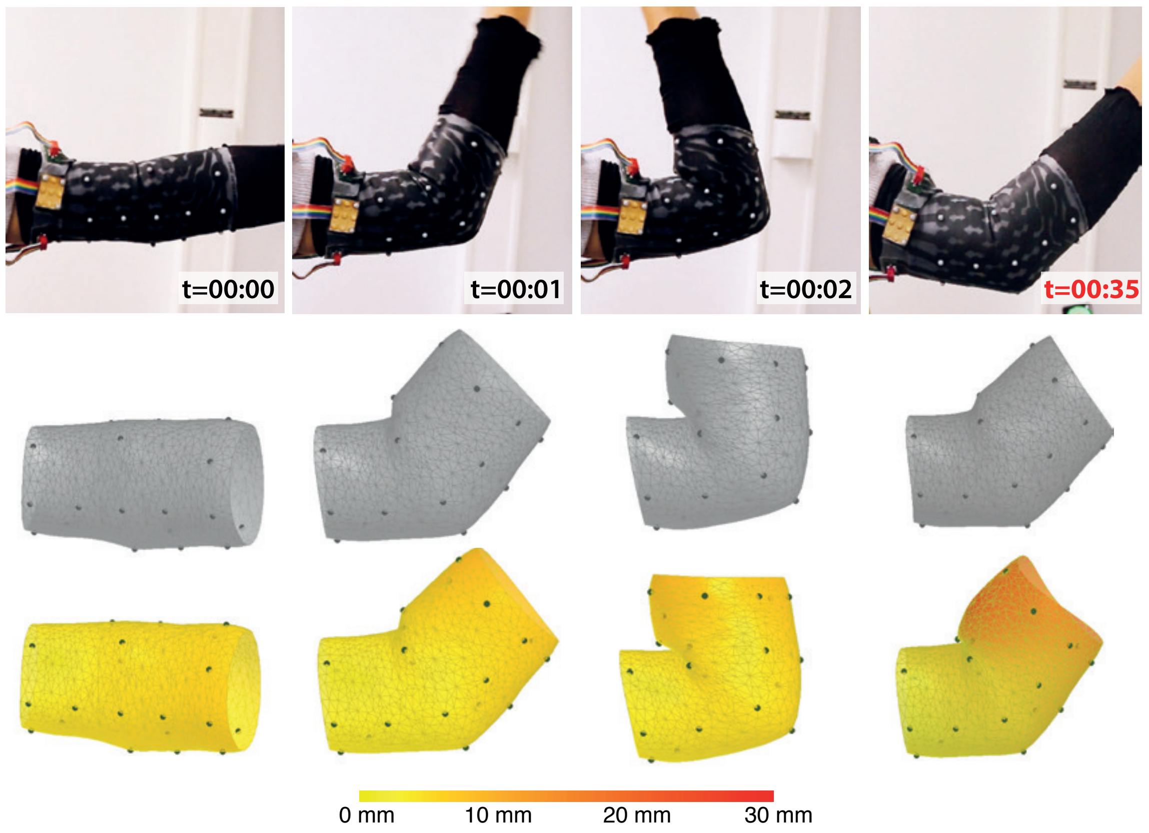


**Figure 4.22:** *Four frames of an elbow capture session. Top: Video frames for comparison. Middle: Mocap ground truth. Bottom: Reconstruction based on the sensor measurements and the trained prior. The rightmost frame corresponds to the largest individual marker error.*

sensor, and we deform a cylindrical (in rest pose) mesh at interactive rates (approximately 8 Hz). See Figures 4.1, 4.25, and the accompanying video for the results. Note that in this setting, the users wear the sensor long after the training data was acquired; when taking the sensor off and putting it on again, one only needs to make sure that the alignment of the sensor and the body part is approximately the same. For the wrist example we quantitatively evaluated this effect of taking the sensor off and putting it on again with an imperfect alignment. For a 2-minute test sequence, the in-session mean error is 4.06 mm (max: 38.28 mm) while the out-of-session mean error is 6.80 mm (max: 47.22 mm).

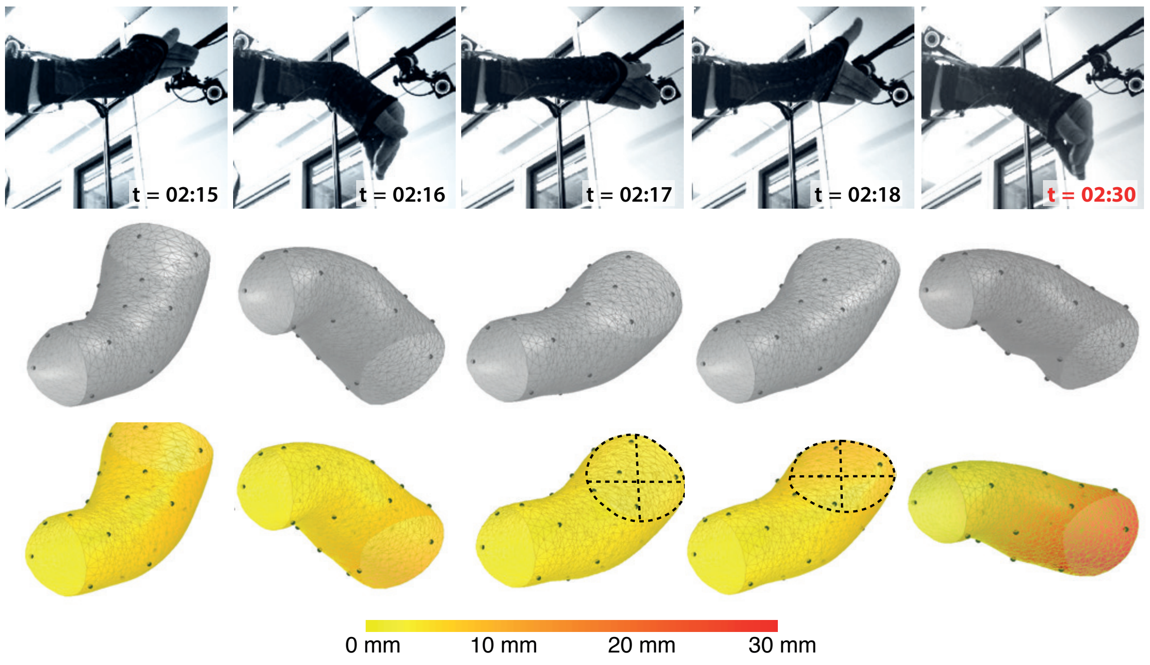


**Figure 4.23:** *Four frames from a wrist capture session. Top row: Video frames for comparison. Middle: Mocap ground truth. Bottom: Reconstruction based on the sensor measurements and our trained prior. In the third and fourth frames, note how our sensor correctly senses its shape when the fingers are splayed. The frame corresponding to the largest individual marker error is shown on the right.*

| $\gamma$ | $\beta$ | #frames | mean | max |
|---|---|---|---|---|
| $\alpha_{\text{min}}$ | $\alpha_{\text{max}}$ | 8799 | 3.51 | 27.22 |
| 60 | $\alpha_{\text{max}}$ | 8668 | 3.14 | 28.40 |
| 40 | $\alpha_{\text{max}}$ | 7335 | 3.40 | 49.20 |
| 30 | $\alpha_{\text{max}}$ | 5777 | 4.07 | 50.55 |
| 20 | $\alpha_{\text{max}}$ | 3229 | 6.59 | 76.96 |
| 20 | 30 | 6251 | 3.35 | 26.45 |
| 20 | 40 | 4693 | 3.89 | 31.31 |
| $\alpha_{\text{min}}$ | 20 | 5570 | 3.41 | 35.76 |
| $\alpha_{\text{min}}$ | 30 | 3022 | 4.67 | 47.76 |
| $\alpha_{\text{min}}$ | 40 | 1464 | 7.38 | 52.50 |

**Table 4.2:** *Predictor accuracy of the wrist test example with artificially reduced training data. It shows the ability of handling strains in the test data not previously seen during training. The training is reduced to frames with $\alpha < \gamma$ or $\alpha > \beta$, where $\alpha$ is the angle between the arm and the palm and $\gamma$, $\beta$ are angular limits.*

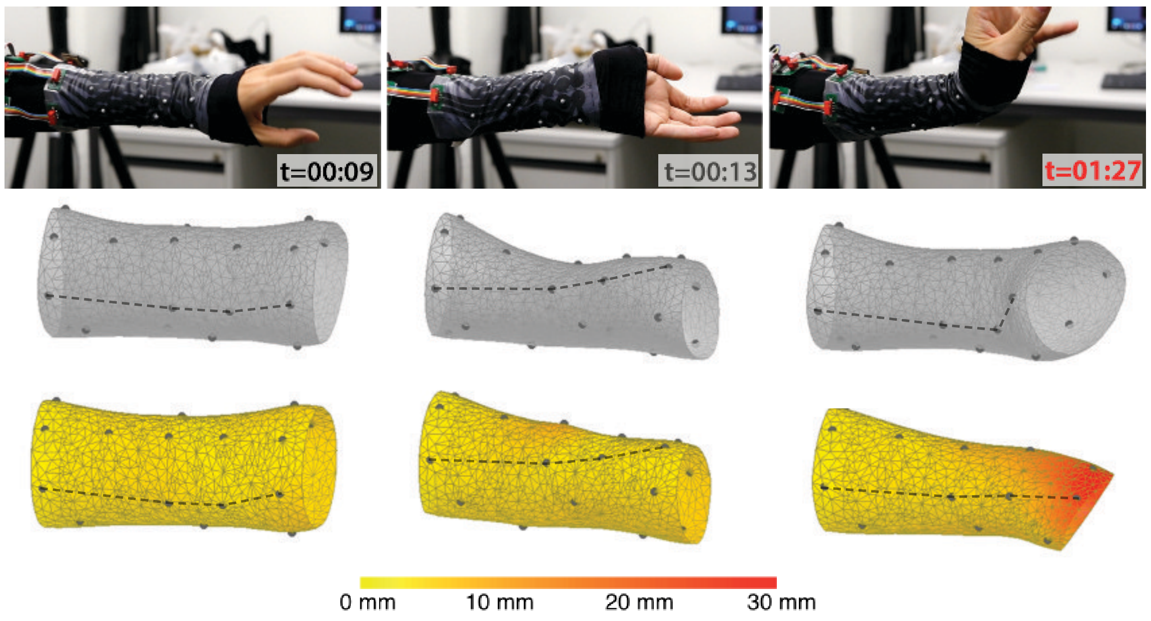


**Figure 4.24:** *Three frames from the forearm capture session. Top row: Video frames for comparison. Middle: Mocap ground truth. Bottom: Reconstruction based on the sensor measurements and our trained prior. Be aware that our sensor is only able to capture local stretch occurring below the sensor. The frame with the highest individual error is shown on the right: The sensor fails to correctly predict the bending of the wrist.*

## 4.6 Conclusion

We proposed a soft and stretchable capacitive sensor array that allows measuring localized area changes. When paired with a learned geometric prior, it can reconstruct complex deformations without line-of-sight.

Our fabrication method and sensor layout open the door to multiple exciting future work venues. The most obvious is combining our area sensor with bend sensors to measure both extrinsic and intrinsic surface geometry, to e.g., also capture isometries. Furthermore it would be compelling to find a way to capture distance changes in such a dense array setting. These extensions would allow to estimate the deformation of general surfaces (like clothing) even if there's no non-area preserving stretching or twisting occurring. Another practical addition would be an assisting mechanism for correct placement of the sensor on the measured object: at present, we simply take a photograph before the training session and peruse it when putting the sensor on again for live session capture.

The acquisition of a large dataset of training sequences with multiple users is necessary to generalize our approach to multiple users, skipping the per-

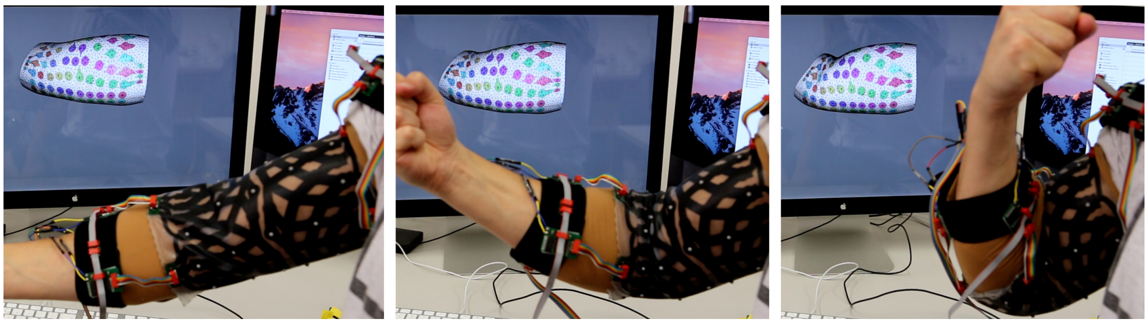


**Figure 4.25:** *Three frames from a live capturing session of the biceps.*

user training session. As with other sensing modalities (e.g., EMG, EEG), additional research into solving the cross-session problem may be required in this setting. Furthermore, the computational design of sensor layouts that are optimized for a specific set of deformations is also an interesting challenge that would directly benefit from the flexibility and simplicity of our fabrication pipeline. Finally, more complex sensor (3D) geometries such as data gloves appointed with our sensor array would enable a number of compelling use cases, such as reconstructing fine-grained hand shape in real-time, sidestepping the various issues (occlusions, lighting) associated with other sensing modalities.

We note that we employ a sparse set of markers as our *ground truth*, and effectively reconstruct this set from our sensor readings. Ideally we would like to have densely captured 3D geometry for training, and match it to denser sensor readings. As discussed in Sec. 2.2, spatially and temporally dense 3D capture is highly challenging and currently invariably involves some degree of model fitting. A realistic simulator that generates large quantities of high-quality synthetic data could be an alternative. It would be interesting to develop a denser version of our sensor design for more direct, dense geometry measurements. This comes with its own challenges, such as properly housing the electronic boards and a time multiplexing strategy to keep the read-out frame rates interactive; we leave this as future work.

CHAPTER 5

# Hand Pose Estimation with a Stretch Glove

## 5.1 Introduction

Hands are our primary means to manipulate physical objects and communicate with each other. Many applications such as gaming, robotics, biomechanical analysis, rehabilitation and emerging human-computer interaction paradigms such as augmented and virtual reality (AR/VR) critically depend on accurate means to recover the full hand pose even under dexterous articulation. These challenging applications require that a hand tracking solution fulfills the following requirements: 1) it must be *real-time*, 2) it should work in a variety of environments and settings, and 3) it should be minimally invasive in terms of user instrumentation.

In many applications, hand pose is recovered via commercial motion cap-

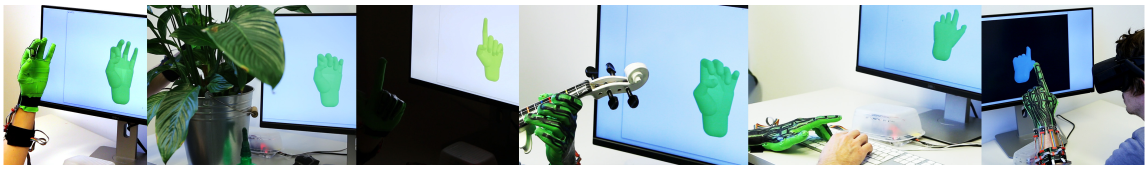


**Figure 5.1:** *Our stretch-sensing soft glove captures hand poses in real time and with high accuracy. It functions in diverse and challenging settings, like heavily occluded environments or changing light conditions, and lends itself to various applications. All images shown here are frames from recorded live sessions.*

ture systems (MoCap) such as Vicon [Vic 2019], but these require expensive infrastructure and markers placed on the user. Marker-less approaches to the task of hand pose estimation include multiple cameras [Ballan et al. 2012; Tompson et al. 2014a; Oikonomidis et al. 2011b], or more recently, a single depth camera [Oberweger and Lepetit 2017; Oberweger et al. 2015b; Tang et al. 2014; Wan et al. 2016] or even monocular camera [Spurr et al. 2018; Iqbal et al. 2018; Cai et al. 2018; Mueller et al. 2018; Zimmermann and Brox 2017]. Despite this significant progress, vision-based methods require externally mounted cameras with the whole hand visible in the image. This limitation presents a practical barrier for many applications, in particular those where heavy occlusions can be expected, such as while interacting with an object, wearing gloves or other items of clothing or while working in cluttered environments. Thus camera-based techniques are limited to applications with a controlled environment and impose physical constraints on immersive user experiences.

Mounting sensors directly onto the user's hand removes the need for direct line-of-sight and can improve robustness and reliability. Not surprisingly, a variety of glove-like devices have been proposed in research (e.g. [Chossat et al. 2015]) and are available commercially (e.g., [Cyb 2019; Man 2019]). Such approaches typically leverage inertial measurement units (IMUs), bend sensors, strain sensors or combinations thereof to capture local bone transformations. While potentially accurate, placing a sufficient amount of sensing elements on a glove in order to capture all the degrees-of-freedom (DoFs) of the hand is challenging due to space constraints. Hence, most existing solutions use fewer sensors than there are DoFs in the human hand. This inherently restricts the reconstruction fidelity.

We propose a self-sensing (i.e., without the need for external sensing) hand-pose estimation approach in a thin, unobtrusive form-factor. Our approach leverages two key observations: 1) as introduced in Chapter 4, it has recently become feasible to produce soft, stretchable sensor arrays entirely from silicone, and 2) modern data-driven techniques can be leveraged to map the resulting sensor readings (which are no longer trivially related to bone transformations) to hand poses. The combination of these two observations leads to our contribution: a soft, self-sensing glove, consisting of an over-complete sensor array (i.e., more sensing elements than DoFs) that can accurately reconstruct hand poses without an optical setup and requiring only minimal calibration. Furthermore, our glove is thin and easy to put on and take off without sacrificing a tight adaptive fit that is crucial for high repeatability.

The proposed glove senses local stretch magnitude exerted on the embedded silicone sensors by measuring their capacity changes. The stretch-driven sensors are small, soft and low-cost. Our fabrication process as proposed in Chapter 4 has so far only been shown to capture simple cylindrical shapes at a frame rate of 8 Hz. The main hardware contribution of Chapter 5 is a much more elaborate sensor design in the form of a wearable glove, which requires several improvements to the fabrication process, including integration of the sensor array with a textile cut pattern, as well as a redesign of the readout scheme to enable querying the glove at 60 Hz.

Since the stretch sensors are not in a one-to-one relation with the degrees of freedom of the hand, the reconstruction of the pose is a highly involved task. While in Sec. 4.4.2 we use an out-of-the-box deep neural network that maps capacitance to 3D vertex positions for this purpose, we discover that a data representation based on prior knowledge of geometric neighborhood and spatial correspondence, both in the input and output domain, allows a neural network to more efficiently discover the inter-dependencies between the joints in the human hand and in consequence outperforms several baseline architectures.

Attaining a sufficiently large and diverse training data corpus for hand pose estimation is a notoriously difficult problem due to the absence of ground-truth acquisition approaches. While this is particularly severe in the case of (2D) image-based approaches (where no instrumentation whatsoever may be used), we observe that our glove design is so unobtrusive, that it is invisible to a depth-camera. This allows us to leverage a state-of-the-art model-fitting based hand tracking approach [Tkach et al. 2017] to capture a large training dataset consisting of one million samples from 10 subjects of time-synchronized sensor readings and the corresponding joint-angle configurations, including a set of shape parameters per person, which is released to the public domain[1] to foster future research.

To validate the utility and performance of our data capture and regression setup, we carry out extensive experiments using different calibration regimes, varying from employing a personalized model for a specific hand, to applying our model to different users with significant variation in hand shapes and sizes. The quality of our reconstruction deteriorates gracefully, offering different calibration options depending on the accuracy required by the application. Finally, we compare with two commercial gloves, demonstrating that our solution shows substantial improvement in reconstruction accuracy (35%), which we believe may have a major impact in real-world applications, especially when paired with the low cost and simple fabrication of our device.

[1]https://igl.ethz.ch/projects/stretch-glove/

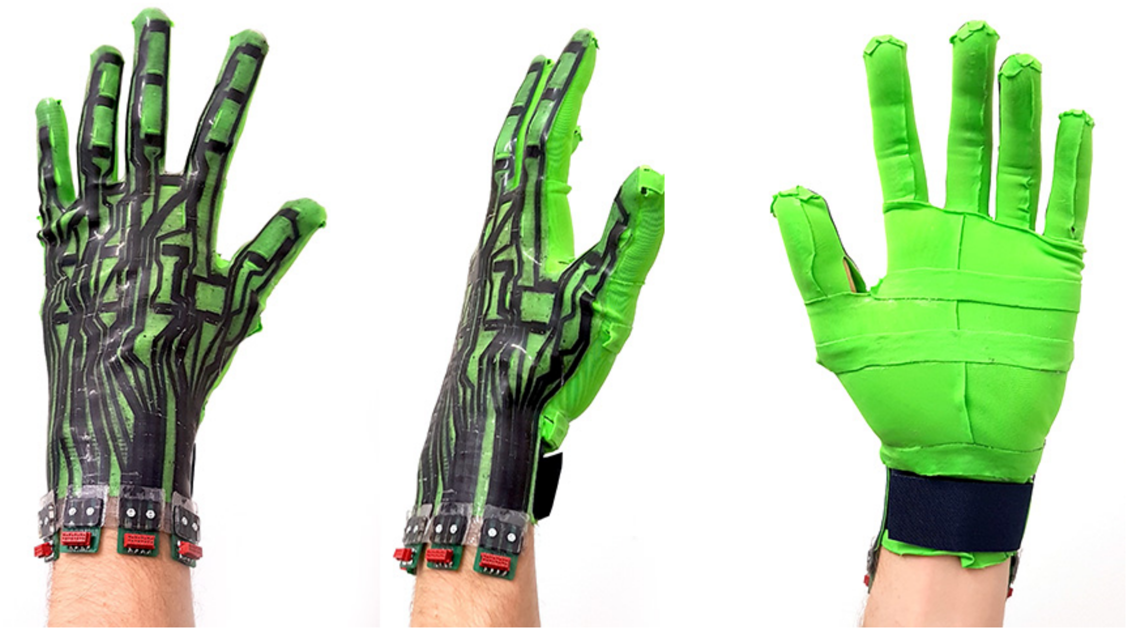


**Figure 5.2:** *Our glove consists of a full soft composite of a stretchable capacitive silicone sensor array and a thin custom textile glove (green).*

## 5.2 Composite Capacitive Glove

The goal is to develop a thin and lightweight glove that is comfortable to wear, yet delivers high pose reconstruction accuracy without requiring elaborate calibration or a complex setup.

Fig. 5.2 illustrates our final design, consisting in a dense stretch sensor array. It is easy to put on, unobtrusive to wear, and manufacturable at a low-cost (material cost around 15 USD, not including 60 USD for prototype electronics). At the heart of our data glove lies a silicone based stretch sensor array, specifically designed for the purpose of reconstructing dexterous hand articulations. Our design features 44 individual stretch sensors on a hand-shaped silicone sensor array, attached to an elastic textile to form a thin form factor glove. The total weight is just 50 g and its thickness is only 1.2 mm, making it comfortable to wear even for extended use. Our glove adapts well to a range of hand sizes and shapes: one single size fits the hand of all members of our research group.

The sensor is a composite material, consisting of a textile layer paired with conductive and non-conductive silicone layers, fabricated following the procedure described in Sec. 4.3.3, but adapted to the more complex geometry and motion of the hand.

### 5.2.1 Sensor design

**Sensor.** We use the the stretch array sensors as introduced in Sec. 4.3.2. The space-efficient design allows us to place as many as seven sensors on thin objects like fingers. And for our 44 sensors on the glove, only 27 leads in 2 layers are needed, compared to 45 with a non-matrix approach, a reduction of 42.5%. Capacitive stretch sensors are appealing since they are based on the principle of a shape-changing capacitor, which, unlike many resistive sensors, does not suffer from hysteresis.

**Readout scheme.** The matrix layout means that sensor cells cannot be read directly. In Sec. 4.3.2 we explain how our custom time-multiplexed readout scheme works. However, the procedure and choice of resistors used in the original configuration, would lead to an insufficient readout rate of only 5 Hz in our setting. The sensor readout scheme neglects the lead resistance. Therefore, high charging resistors (56 kOhm and 470 kOhm) are required to achieve physically accurate stretch reading. For the glove application, the readings only need to be repeatable but do not necessarily directly correspond to physically meaningful stretch values. This allows us to use lower charging resistors (47 kOhm and 220 kOhm), improving the readout rates. Further, instead of solving for $\mathbf{C}_c$ every full cycle of combined measurements (180 updates), we solve every 16 updates. We experimentally found that this setup provides good sensor readings, and that more frequent solving has a negative impact on the frame rate due to the limits of the micro-controller-host communication bottleneck. Our readout scheme has a capture rate of about 60 Hz. To filter out noise in the readings, we mean-filter the last five frames of $\mathbf{C}_m$ before solving for $\mathbf{C}_c$.

The readings $\mathbf{C_c}$ are fed to a deep neural network that outputs hand poses (see Sec. 5.3). They can then be queried by an application, e.g., to render the hand in VR or perform collision detection with virtual objects for interaction. In our live experiments, the hand poses are filtered by a so-called 1€-Filter [Casiez et al. 2012].

**Sensor layout.** The sensor layout (Fig. 5.3) is manually designed by adding sensors in stages: (i) longer sensors directly correspond to the main joints of the fingers (21-24, 32-36, 40-42) and the thumb (0, 20); (ii) abduction sensors in-between the fingers (16, 25-27); (iii) perpendicular sensors on the fingers (8-9, 29-31, 37-39, 43) and the thumb (1, 28); (iv) a regular grid of both horizontal (2, 4, 7, 10, 17-19) and vertical (3, 5, 6, 11-15) sensors on the back of the hand. The subtle differentiation into horizontal and vertical sensors is the result of

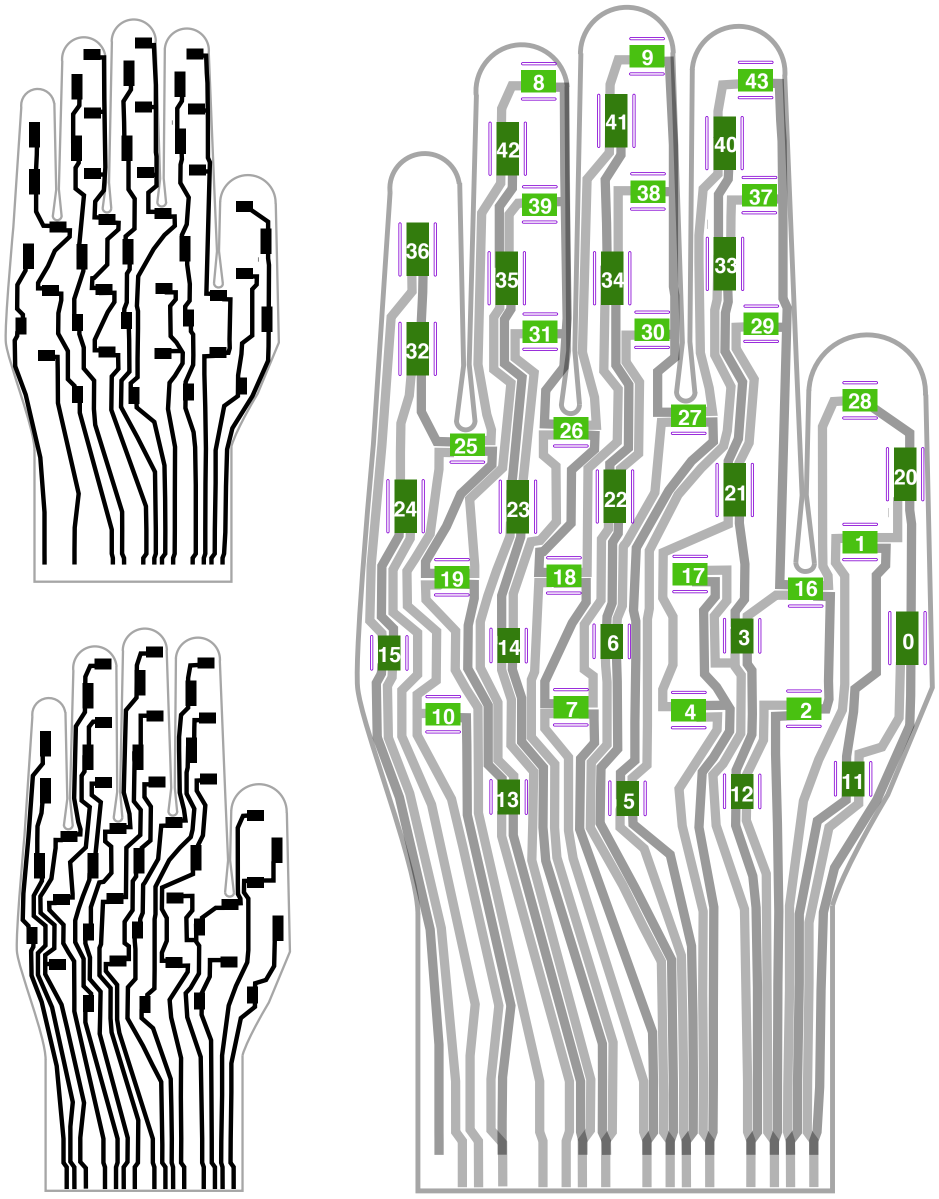


**Figure 5.3:** *Left: Patterns of the two conductive layers. Right: Wherever the two conductive layers overlap, local capacitors form (marked in green) and serve as local stretch sensors.*

the ventilation cuts, explained in the next paragraph. Fig. 5.10 shows how each of these sensor categories helps to improve the reconstruction accuracy. Finally, the sensors are connected by leads in two layers, such that each pair of connected traces (from different layers) overlap at most once. We consider reduction of the lead lengths and avoidance of stretch absorption by the nearby cuts when determining the final sensor placement. For these reasons, e.g., the sensors over the knuckles (32-36, 40-42) are not centered, leaving some blank space.

Note that for good sensitivity with respect to finger abduction it is important that the sensors are pre-stretched when the glove is put onto the user's hand. It is thus crucial to fabricate the sensor array in the rest pose shown in Fig. 5.3, right. In particular, the fingers must be parallel without any gap in-between.

**Cuts.** Thin cuts (Fig. 5.3 right, in purple) with rounded ends are added via laser cutting on two sides of the rectangular sensors to enhance the wearing comfort by increasing ventilation. They also have a minor, yet positive, effect on the readings, since they lower stretch force resistance, thus making the sensors more sensitive to stretching parallel to the cuts. For example, sensors 21, 33 or 40, located over the joints of the index finger, are much less sensitive to volume changes of the finger, while sensors like 43, 37, 29 are mainly sensitive to volume or diameter changes of the finger (e.g., due to muscle bulging). In Fig. 5.3 (right) the sensors more sensitive to vertical stretch are colored in dark green, and the ones more sensitive to horizontal stretch in light green.

### 5.2.2 Fabrication

Our glove is made of a composite consisting of a silicone sensor sheet and an elastic textile, only requiring tools available in a modern fablab. It is fabricated in a two-stage approach, as outlined in Fig. 5.4: first, we fabricate the soft silicone sensor array (steps 1-8), covering the back side of the glove, and then we attach textile parts to the silicone sheet and close them up to form a soft and wearable glove (steps 9-12). While our sensor array is based on the ideas introduced in Chapter 4, the original fabrication process cannot be directly applied. For readability, first, we detail the full sensor fabrication process in the glove setting and discuss the differences afterwards in Sec. 5.2.3.

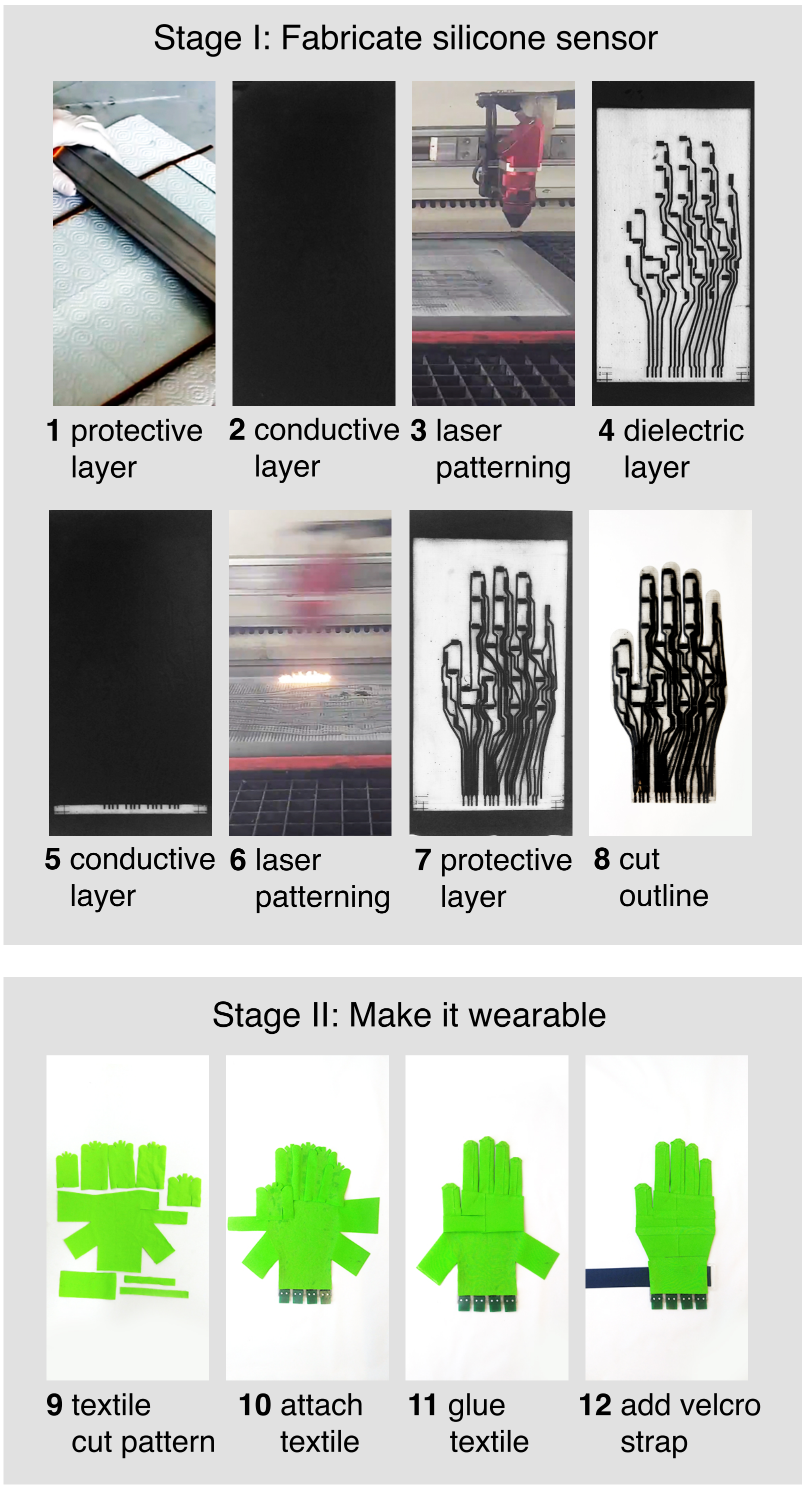


**Figure 5.4:** *The fabrication of a glove consists of two main stages: Fabricating the silicone sensor (1-8) and textile design for the glove (9-12). Note that the conductive layers (2,5) are a mixture of silicone and carbon black and therefore, mostly black.*

**Stage I: Silicone.** The hand-shaped silicone sensor array (see Sec. 5.2.1) consists of two conductive, patterned layers with a dielectric layer in-between and encapsulated by two shielding layers. It is produced layer by layer using the following steps.

First, we cast an insulating base layer onto a glass plate, controlling the thickness by attaching tapes at the borders of the glass plate. Next, a conductive layer, made from Silbione RTV 4420 silicone [Sil 2018] mixed with carbon black (conductive powder, [Ens 2018]), is cast directly onto the first layer. The laser cutter then removes, by repeatedly etching (5 times) the negative of the pattern shown in Fig. 5.3 (lower left), leaving the full base layer with the conductive traces on top. Then, a pure silicone dielectric layer is cast, followed by another conductive layer, which is also etched (Fig. 5.3, upper left). Finally, another insulating shielding layer is added.

Note that the conductive layers are produced with a thickness of 220 $\mu$m to allow for the needed leads of just 2 mm width (see Fig. 5.3). To keep the connection pads at the base of the sensor exposed, a thin tape is used to cover the pads before casting (for the last three layers) and removed before the curing in the oven. Additional information on the sensor-to-read-out-circuit interconnection are provided in Appendix C.2.

The laser cutter parameters in the etching step are Power=30, Speed=40, PPI=500 (Trotec Speedy 300 laser cutter). Using a higher power during the etching process would make the silicone sensor cross-linked with the base glass and hard to peel off in the end. After every full etching cycle, the sensor is cleaned from dust residue by carefully wiping it with a towel and isopropyl alcohol.

After every casting step the sensor is cured in the oven for 20 minutes at 90 °C. Before curing in the oven, sensors have to be left sitting for 15 minutes, to let the solvent evaporate. Otherwise, bubbles can form during curing due to the evaporation of the solvent from within, with the uppermost part of the layer already cured.

Finally, the sensor is cut into a hand shape with the laser cutter. An accurate alignment of the etching and cutting steps in the laser cutter is crucial to avoid cuts in the sensors, as this could lead to short circuits between the conductive layers. The overall thickness of our sensor is 0.85 mm.

**Stage II: Textile.** The silicone sensor array is not wearable. There is no easy way to attach it firmly to the hand, and gluing two sheets of silicone together is a difficult (while not impossible) task. Attempting to put on or

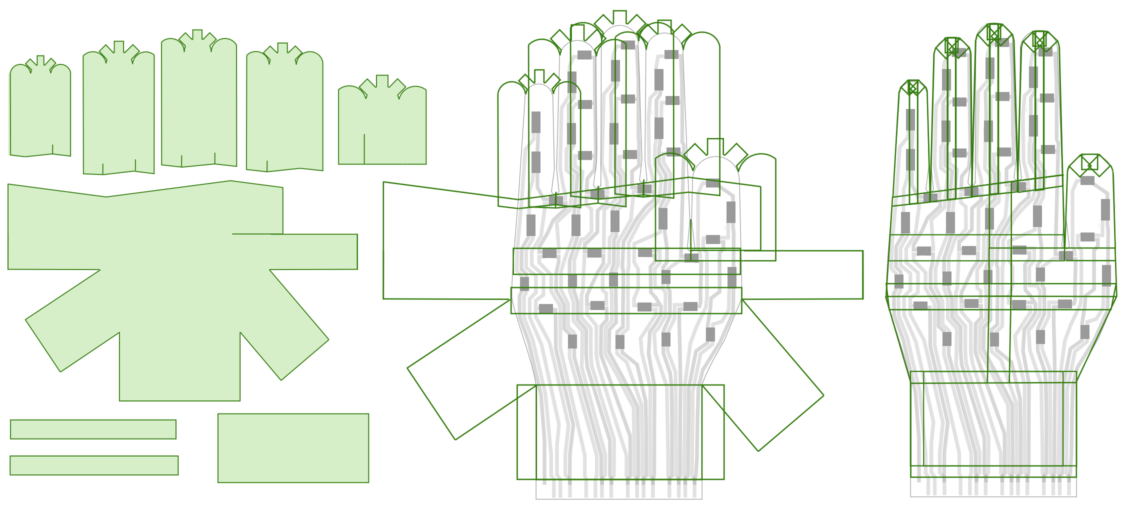


**Figure 5.5:** *Left: The textile cut pattern is made from a large palm part, one for each finger and 3 extra flaps for connections. Middle: Alignment of the cut flat pattern parts with the flat, hand-shaped silicone sensor. Right: Finished glove after closing up the flaps with textile glue.*

take off such a glove is very cumbersome due to large friction and tightness. We attempted to attach the sensor to a standard glove, but found that it is challenging to get a proper alignment with the major centers of articulation, and it is also difficult to do it robustly and with the needed repeatability.

Therefore, we propose a simpler and more effective solution, exploiting a laser cutter to cut a custom textile pattern (see Fig. 5.5). The textile parts can be attached to the silicone sensor while laying on a flat surface. First, a PET mask that covers the sensors and the cuts is placed on the sensor, then everything is covered with Sil-Poxy silicone adhesive ([Sil 2019]), and finally, the mask is carefully removed, and the textile parts are placed and firmly attached.

In a second step, the different textile parts are closed up, using HT 2 textile glue ([HT2 2019]), and the seams are bonded with an electric iron. A highly elastic jersey textile (80% polyamid with 20% elastane) with a thickness of 0.35 mm is used. Finally, we attach a wrist strap with a velcro fastener to reinforce the tightness and ensure a repeatable alignment of sensor cells to joints.

### 5.2.3 Comparison to the original sensor arrays

The glove sensor is a composite material made of a silicone layer (see Appendix C.2) and an additional textile layer. The latter is crucial in making a functional glove, since pure silicone cannot be draped over complex geometries without the risk of immediate damage, especially due to the large friction with the human skin. In terms of the silicone sensor layer, there are two major differences: (1) the readout scheme is different (Sec. 5.2.1), allowing for a frame rate of 60 Hz (vs. 8 Hz), and (2) we seek to reconstruct a much more complex geometry that is articulated in more complex ways than the simple cylindrical shapes and single axis joints. The thin structures of the hand require high sensor density, but offer little surface area to place the sensor cells. To overcome this problem, we use much thinner leads (2 mm vs. 6 mm) and smaller local sensor cells (5 x 7.5 mm and 5 x 11.5 vs. 15 mm diameter). To keep the total resistance of the longest leads in a useful range, the conductive layers have to be five times thicker (220 $\mu$m instead of 45 $\mu$m). As a consequence, more etching cycles are required during fabrication, and the solvent must pre-evaporate to prevent bubbles from forming during curing in the oven.

## 5.3 Data-driven hand pose estimation

Our composite capacitive glove contains more sensors (44) than the number of DoFs in the human hand model we consider (25), however, the stretch sensors' readings are not in a direct relationship with the joint rotation centers of the hand. Furthermore, sensor readings vary from person to person due to the different hand geometry. Designing a custom mapping from sensor readings to hand joint configurations manually is a highly complex task (e.g. [Kahlesz et al. 2004]), which requires experiments and manual work to adapt the model to the specific sensor layout. We propose instead a data-driven approach that can learn this highly non-linear mapping, across different sessions and users. While acquiring training data for hand pose estimation is generally difficult, gloves are a special case since they can be unobtrusive enough to be essentially invisible to a depth camera. Therefore, it is possible to capture training data efficiently using an off-the-shelf hand tracking system [Tkach et al. 2017].

Any standard neural network architecture could be used in our setting, including fully connected networks (FCN) and long short-term memory networks (LSTM). However, we observe that these standard approaches struggle to exploit the geometric layout of our data. By constructing an ad-hoc data layout and a network that implicitly encodes the sensor geometry and topology, we considerably improve the accuracy over standard baselines.

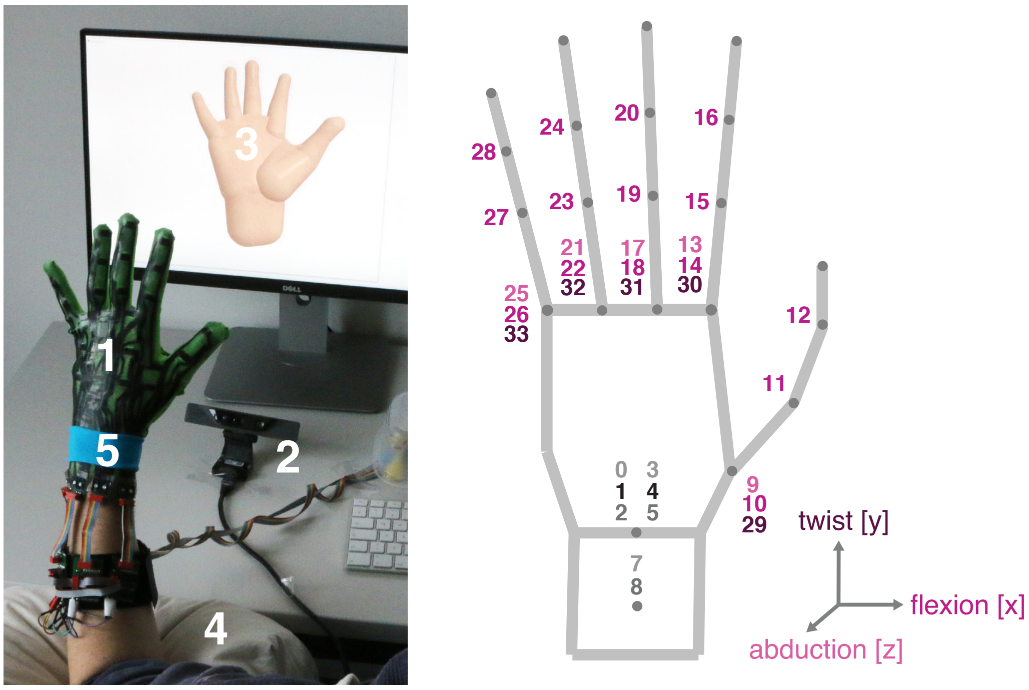


**Figure 5.6:** *Left: Our training data capturing setup. 1) Hand with glove; 2) RealSense SR300 depth camera; 3) computer running [Tkach et al. 2017]; 4) pillow for comfort; 5) blue segmentation wristband as required by [Tkach et al. 2017]. Right: The 34 hand pose parameters proposed by [Tkach et al. 2017]. Our glove only captures the 25 degrees of freedom in color. The DoFs in gray are the global translation and rotation, and the rotation of the wrist; global pose parameters cannot be captured using only stretch sensors.*

### 5.3.1 Data acquisition

Our setup for capturing training data is shown in Fig. 5.6 (left). For capturing the reference hand poses, we use an inexpensive Intel RealSense SR300 depth camera [Rea 2019]. Depth frames are fed to the (unmodified) algorithm of [Tkach et al. 2017], which requires a blue strip on the wrist for depth segmentation. We use their calibration method to compute the hand shape parameters per user. To capture meaningful training data, a good synchronization between the different data sources is crucial. To this end, we incorporate our code for communication with the glove sensor into the publicly available source code of [Tkach et al. 2017]. This allows for unified collection, evaluation and logging of both sensor and pose data.

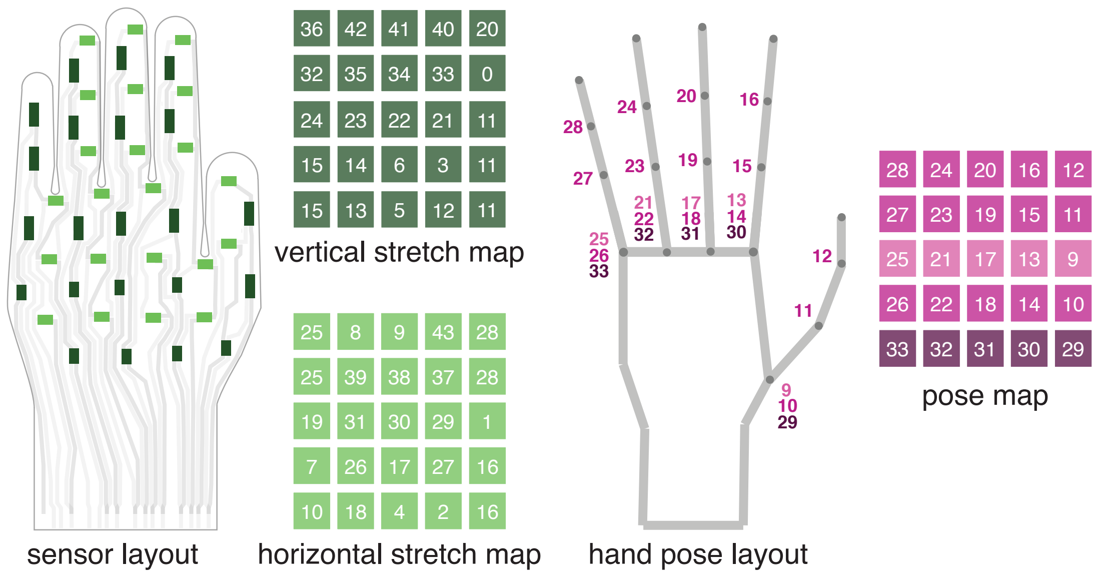


**Figure 5.7:** *The geometric correspondences between the input (sensor layout on the left) and output features (hand model on the right) should be considered. Both the input and the output can be naturally ordered in corresponding grid structures.*

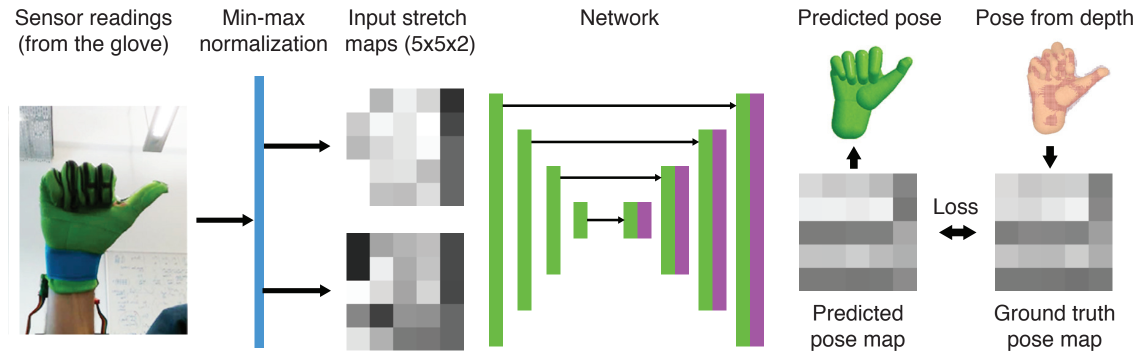


**Figure 5.8:** *From left to right: The sensor readout is normalized by the hand-specific min-max, arranged in two 5x5 stretch maps, and fed through a U-Net network that predicts a 5x5 pose map. The loss is the $L_2$-norm difference between the predicted pose map and the ground truth pose map derived from the hand poses captured by [Tkach et al. 2017].*

### 5.3.2 Data representation and network

For $N$ frames in the training data, the input $\mathbf{X} = \{x_i\}_{i \in N} \subset \mathcal{C}^{44}$ to our regression model is the readout from the 44 stretch sensors, while the target output $\mathbf{Y} = \{y_i\}_{i \in N} \subset \mathcal{C}^{25}$ are the 25 hand pose parameters as defined in [Tkach et al. 2017], covering the full pose DoFs of the hand (see Fig. 5.6).

**Data representation.** Our key observation is that the spatial correspondences between input and output features should be considered. A meaningful ordering and organization of features make the learning task easier. For example, a group of nearby sensors on the thumb (sensor cells 0, 1, 2, 11, 16, 20, 28 in Fig. 5.3, right) taken together should have a higher impact on the prediction of the thumb movement (parameters 9, 10, 11, 12, 29 in Fig. 5.7). Meanwhile, some high-level hand gestures, like the clenching of a fist, cause more uniform sensor actuation, which should be encoded globally and hence makes a priori definition of these inter-dependencies difficult. Training an FCN to learn such global-local information is theoretically possible, but practically it would require excessive amounts of model capacity, training data, and hyper-parameter tuning. We opt instead to directly build this geometric and topological prior into our network architecture to regularize the learning process and improve the reconstruction performance.

We use a fully convolutional neural network (CNN) and 2D grid representations as input and regression target. More specifically, we use $5 \times 5$ matrices to organize our input and output data. Fig. 5.7 shows how we map sensor cell readings and pose parameters to 2D grids, each capturing the spatial relationships. We use one matrix to organize the output, but a stack of two for the input, since each sensing location has two types of sensors measuring horizontal and vertical stretch (see Sec. 5.2). For example, both sensors 29 and 33 are located around the knuckle of the index finger, but each sensor captures different stretch directions.

**2D network.** We use the U-Net network architecture [Ronneberger et al. 2015] to transfer the organized sensor readout to hand pose parameters. The downsampling and upsampling structure of the network can encode the global information, while the symmetric skip connections between the encoder and the decoder can preserve the local correspondences. Fig. 5.8 illustrates the structure of U-Net and how the network transforms the 2D sensor data to hand poses. We use $L_2$ loss for our regression task, $\mathcal{L}_{\text{reg}} = \sum_{i=1}^{25} \left\|\hat{y}_i - y_i\right\|_2$, where $\hat{y}$ is the prediction and $y$ is the target pose parameter.

Experiments show that our model compares favorably to alternative network architectures. We provide a comparison against baselines in Sec. 5.4.4 and describe the experimental setup in detail in Appendix C.1.

### 5.3.3 Data processing

We improve our data quality by removing outliers and by using a min-max normalization method for calibration. That is, the input to our network is the processed and mapped sensor data, see Fig. 5.8.

**Outlier removal.** We remove frames that are likely to be outliers by detecting finger collisions, since they indicate unfeasible poses. We filter out frames where the collision energy defined in [Tkach et al. 2017] is above 80, indicating that the estimated pose is likely unnatural and wrong. This filter only removes about 2% of the data.

**On-the-fly calibration.** Ideally, the per-sensor reading magnitude should be normalized to become insensitive to the hand size. We observe that, once the glove is put on, the minimum and maximum magnitude of each sensor's readings is fixed, which can be used to normalize the sensor data. Therefore, we find that a per-sensor min-max calibration is a reasonable trade-off between cost and accuracy. The key is to find the min and max magnitude after the glove is put on. In practice, we propose a short calibration phase, where the user should freely explore different extreme poses, yielding the min and max values per sensor, which we then use to normalize the sensor data to the $[-1, 1]$ range. To make this process even more robust, we use a median filter (over 20 frames) while extracting the min and max values. This simple calibration method works surprisingly well in practice, due to the complexity and tightness of our soft glove, which provides a proper alignment.

## 5.4 Experiments and results

We show how our glove and the symbiotic data-driven hand pose reconstruction method can capture accurate hand poses (Sec. 5.4.3) on a large dataset (Sec. 5.4.1). We compare our glove's performance on a pose sequence with two commercial state-of-the-art gloves (Sec. 5.4.3). Finally, we evaluate the proposed network architecture, contrasting it with alternative baselines (Sec. 5.4.4).

In the setting where a new glove user only needs to perform a minimal on-the-fly calibration (min-max normalization using a generic, pre-trained model), we achieve an overall mean error of only 7.6 degrees. In a comparison sequence, with a mean pose error of 6.8 our glove outperforms the ManusVR glove (mean error: 11.9) and the CyberGlove (mean error: 10.5). The proposed

2D network architecture can achieve a mean error of about 1 degree lower than the baseline fully-connected network.

### 5.4.1 Dataset

Our experiments are performed on a large data set captured from 10 people (except where noted), including a wide range of hand sizes and shapes. The hand length varies from 17 to 20.5 cm, the width from 9 to 11 cm, and the aspect ratio of length to width from 1.6 to 2.1. For each person we capture five sessions using our data acquisition setup; each session lasts about 5 minutes. During three of the five sessions, the participant keeps the glove on continuously, while in-between the other two sessions the glove is taken off. We refer to these two regimes as *intra-session* and *inter-session*, respectively. To encourage the participants to explore the space of hand poses fully, we show a printed gallery of example poses during the recording sessions. During data acquisition and method development, one of our gloves was in use for over 25 hours (cumulative) — consistently capturing sensor data in high quality.

### 5.4.2 Evaluation on hand pose capture

We envision a standard scenario for our hand capture method, in which the proposed neural network is trained only once, preferably on a large data set containing samples from different hands. This way, a new user only needs to execute the on-the-fly calibration method for less than a minute before using the glove for interaction.

In our experiments, we refer to models that are trained using the data from all participants except leaving one participant out as test data as *generic models*. We also evaluate *personalized models*, which are trained on the data from one person only. This allows for even more accurate pose reconstruction and provides further insight into the capabilities of our glove. Table 5.1 summarizes results of all our models. For all experiments we used a medium sized glove (20$\times$12.5 cm); despite the single size, it can handle a large variety of hands. The most significant error is produced by the smallest hand (last row of Table 5.1) – for a more accurate tracking, a smaller size glove would be required.

**Personalized and Generic models.** For the personalized model, we perform experiments on two types of data: using training and testing on intra-sessions

| Hands | | Personalized | | Generic | | Fine-tuned |
|---|---|---|---|---|---|---|
| Size | L×W×H | (1) Intra | (2) Inter | (3) w/o Calib | (4) w Calib. | (5) Tuned |
| 940 | 19×11×4.5 | 4.8 | 5.5 | 6.4 | 6.6 | 5.7 |
| 792 | 19×9.5×4 | 5.3 | 6.8 | 7.9 | 7.0 | 6.1 |
| 792 | 18×11×4 | 6.2 | 7.8 | 8.6 | 8.5 | 6.6 |
| 836 | 19×11×4 | 5.2 | 5.2 | 7.3 | 6.9 | 5.6 |
| 850 | 18×10.5×4.5 | 6.8 | 6.3 | 8.1 | 7.3 | 6.8 |
| 1025 | 20.5×10×5 | 5.1 | 5.9 | 8.5 | 7.5 | 6.4 |
| 840 | 20×10.5×4 | 5.2 | 5.9 | 8.8 | 8.0 | 7.2 |
| 680 | 17×10×4 | 5.6 | 5.7 | 8.6 | 8.2 | 6.7 |
| 800 | 20×10×4 | 6.1 | 6.5 | 9.0 | 7.3 | 6.0 |
| 612 | 17×9×4 | 7.5 | 6.6 | 10.1 | 9.1 | 8.1 |
| Average error | | 5.8 | 6.2 | 8.3 | 7.6 | 6.5 |
| Time investment | | 2 h | 2.5 h | 0 | 1 min | 20 min |
| External hardware | | Yes | Yes | No | No | Yes |

**Table 5.1:** *Mean pose angle errors (in degrees) over sessions captured from people with different hand sizes and aspect ratios. The* Size *column lists bounding box volume in cm*$^3$*, and the second column gives the bounding box dimensions in cm. We report different scenarios: Personalized models, trained on sessions (1) with the exact same alignment like the test session, or (2) when the glove is taken off in-between; generic models trained on sessions from the other 9 participants (leave-one-out): (3) uses the per-feature min-max sensor data obtained from the training data, and (4) uses the personalized, on-the-fly min-max calibration. (5) Generic model fine-tuned with a short (5 minutes) session of personal training data.* External hardware *refers to the depth camera and GPU necessary for training data capture and processing.*

only and using both types of sessions for training, tested on an inter-session. For the former, we use two sessions to predict the other one. For the latter, we use three intra-sessions and one inter-session to predict the other inter-session. The intra-session samples usually have better performance than inter-session ones. This is due to better alignment of the glove during a continuous session. The *Intra* and *Inter* columns in Table 5.1 show the mean angular reconstruction errors for ten different hands. On average, the mean error for the intra-sessions is 5.8 degrees versus 6.2 for the inter-sessions. The small error difference suggests that our soft glove provides consistent alignment across different sessions even when the glove is taken off in-between. See Fig. 5.14 for example frames from a real-time capture session.

A generic model is crucial for real-world applications aimed at a wide and diverse audience, since training a personalized model is time consuming (2 hours or more) and requires additional equipment (depth camera and GPU). We evaluate the approach in two variants: *with* and *without* using the calibration method described in section 5.3.3. In the case without calibration, a per sensor min-max values over all users are obtained from the training data and applied for normalization, both in training and in testing. Columns (3) and (4) in Table 5.1 show the effectiveness of our calibration method: the average pose reconstruction error is 7.6 degrees with calibration versus 8.3 without. An angular reconstruction error of 7.6 is satisfactory for many applications (see Fig. 5.13 and the supplementary video[2] for a visualization of different reconstruction errors). To further improve the reconstruction quality with minimal personalized data, we apply fine-tuning on then unseen data. That is, we load the network parameters from a pre-trained generic model and then use a small learning rate of $1 \times 10^{-6}$ and batch size of 64 to further optimize all the network parameters, which helps in avoiding catastrophic forgetting. The results are reported in column (5) of Table 5.1; they are comparable in performance to a personalized model, but require a much lower investment of time.

**Application scenarios.** Our method supports five standard application scenarios, summarized in Table 5.1:

1. An intra-session personalized model gives the best performance, but it requires to always keep the glove on.
2. If a depth camera is available to the user, personal training data (20 minutes) can be captured with [Tkach et al. 2017] and used to train a personalized model for about 2 hours.
3. If there is no time or ability (e.g., in a rehabilitation context) for training and calibration, our generic model can be used, combined with the per sensor min-max values extracted from the training set.
4. By first exploring some hand-poses to gather personal min-max values on-the-fly and then using these values to normalize the sensor data, the accuracy of the generic model can be significantly improved, within less than a minute of calibration time.
5. A trade-off alternative to scenarios (2) and (4) is to capture only 5 minutes of personal training data and fine-tune the generic model for about 15 minutes.

[2] `https://igl.ethz.ch/projects/stretch-glove/Stretch-Sensing-Glove-2019_video.mp4`

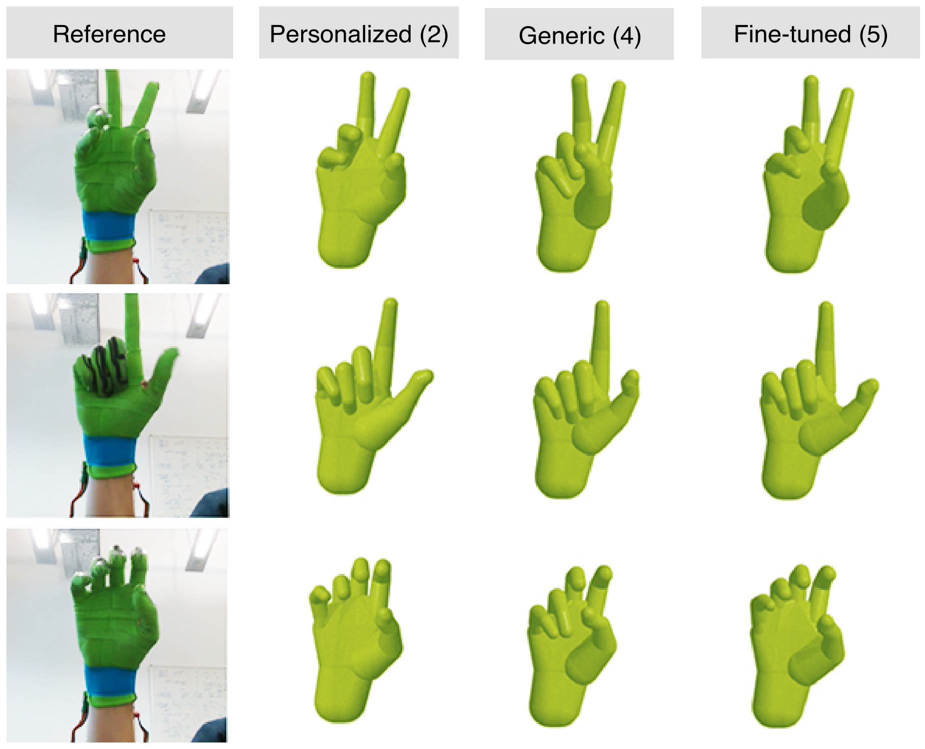


**Figure 5.9:** *Visual comparison of different models on three example frames. From left to right: ground truth pose, reconstruction of personalized (2), generic (4), and fine-tuned (5) models. While all models manage to capture these poses well, the personalized model (2) performs best. We carefully chose these frames to highlight the differences. Most poses have visually similar results for all models as shown in the accompanying video.*

Options (3) and (4) require only the glove and a pre-trained model, while the others need a depth camera and a GPU to train or fine-tune the model. We believe that (4) is the most practical scenario, but applications requiring higher accuracy might benefit from a custom model (2) or (5). In practice, all of our models can capture hand poses reasonably well, and a visual comparison of models (2), (4) and (5) is shown in Fig. 5.9.

**Number of sensors and training data.** To illustrate the benefits of a dense sensor array, we run an ablation study on the number of sensor cells used, to simulate glove designs with fewer sensors (see Fig. 5.10). The results show that using more sensors leads to higher reconstruction accuracy, a 28% decrease in the mean error when going from 14 to 44 sensors.

Our training, validation, and test datasets for the personalized models contain 85K, 10K, and 15K samples, respectively. The numbers of samples for the non-personalized model are 800K, 90K, and 120K. To study the necessity of

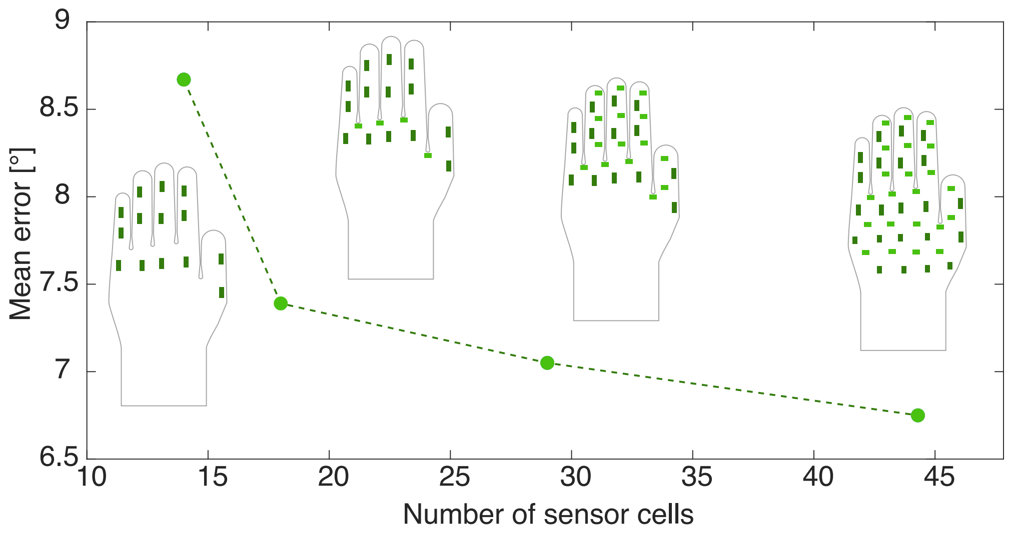


**Figure 5.10:** *As the number of sensors increases, the mean reconstruction error of a captured session decreases: from 8.67 with just 14 sensors covering the main joints to 6.75 for our full glove with 44 sensors.*

| Percentage | 100% | 75% | 50% | 25% | 10% |
|---|---|---|---|---|---|
| Personalized (2) | 5.50 | 6.32 | 6.47 | 6.57 | 7.14 |
| Generic (4) | 6.57 | 6.98 | 7.51 | 7.96 | 9.85 |

**Table 5.2:** *This table shows that the mean session error increases if less training data is available (or used). The number of samples in the training data is 85K for the personalized and 800K for the generic model.*

using such a large training data set, we gradually and randomly remove parts of our training data; the resulting reconstruction errors of the personalized model (2) and the generic model (4) are shown in Table 5.2. The drop in reconstruction accuracy demonstrates the benefit of having a large dataset, and hence the importance of our unobtrusive glove that allows for a convenient data acquisition setup.

**Generalization to a different glove.** In all the experiments presented so far, we use a single glove prototype (*Glove I*), for both training data capture and testing. To evaluate the reproducibility of our fabrication procedure, we fabricate a second glove (*Glove II*) and assess how well a model trained on data from Glove I predicts poses (Fig. 5.11) using readings from Glove II. Table 5.3 summarizes the results, they are very encouraging, especially given that our current fabrication process includes some manual steps (see Sec. 5.2).

| Model<br>Trained on | Generic w Calib.<br>Glove I | Fine-tuned<br>Glove I & II | Personalized<br>Glove II |
|---|---|---|---|
| Error | 8.80 | 5.73 | 5.30 |

**Table 5.3:** *Generalization to a different glove: This table summarizes errors when evaluating model variants on a training session captured with Glove II. From left to right: Generic model trained on Glove I data only; Generic (Glove I) model fine-tuned with 5 minutes of data from Glove II; Personalized model trained on Glove II only.*

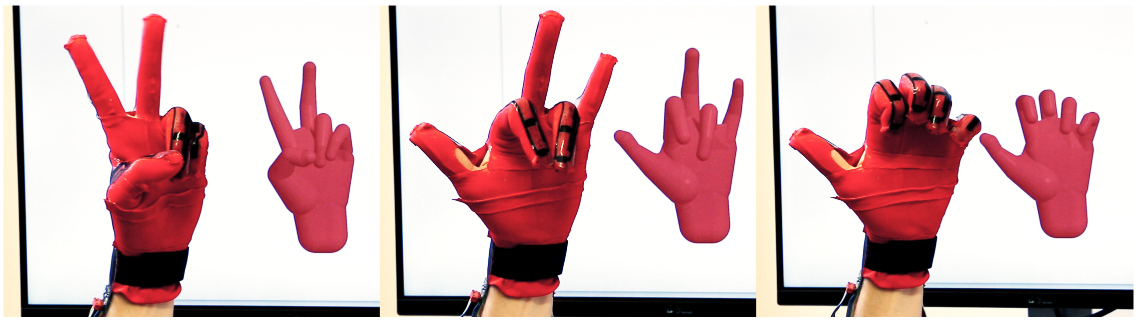


**Figure 5.11:** *Glove II can predict hand poses with reasonable accuracy using a model trained only on the data captured with Glove I.*

We believe that an automated, industrialized version of our glove fabrication process could further improve the reproducibility of our composite glove.

**Object interaction.** In Fig. 5.1 and the supplemental video, we demonstrate our glove interacting with different objects. In general, touching or pressing onto capacitive sensor arrays influences the readings due to body capacitance or deformation of the local capacitors. But usual grabbing and holding of objects makes contact mostly occur on the inside of the hand or at the finger tips where no sensors are placed. In Appendix B.2 we illustrate the effect of touching a capacitive sensor array in an experiment.

### 5.4.3 Comparison to state-of-the-art data gloves

We compare our glove to two state-of-the-art commercial glove products: a data glove by ManusVR [Man 2019] and the CyberGlove II [Cyb 2019] by CyberGlove Systems LLC. To the best of our knowledge, the ManusVR glove has ten bend sensors and 2 IMUs, while the CyberGlove II is equipped with 22 flex sensors. Before the evaluation, we calibrate the two state-of-the-art gloves with their proprietary software. Both routines ask the user to perform

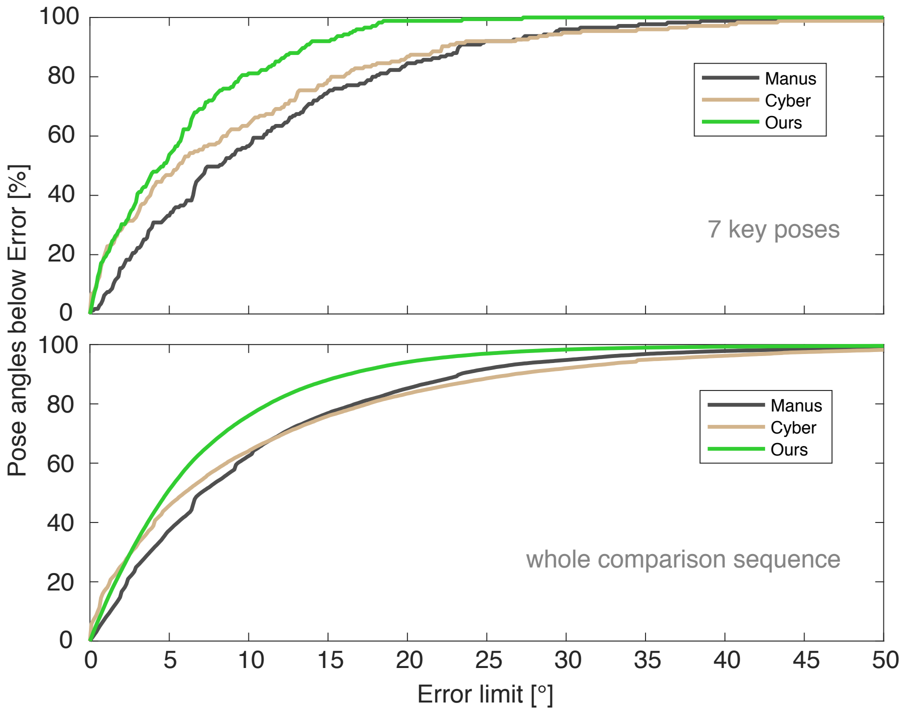


**Figure 5.12:** *Top: Cumulative error plot of the key poses, comparing different gloves. Over the key poses, our glove predicts 92% of the angles below an error of 15 degrees (CyberGlove: 79%, ManusVR: 75%). Bottom: Cumulative error plot over the whole comparison session.*

a given set of hand poses and only take a matter of minutes, comparable in time investment to our min-max sensor normalization, which we use for the comparison (generic model (4)). The gloves are queried through the provided SDKs to record pose data.

For each of the three gloves (ManusVR, CyberGlove and ours) the same sequence of 60 hand poses and a duration of about 3 minutes is recorded. Alongside, the hand pose angles are also captured by the depth tracking system [Tkach et al. 2017], which we use as *ground truth*. This choice might introduce a bias in the comparison due to the use of the same tracking system for training data acquisition. The angles received from the ManusVR, and the CyberGlove are mapped (following the description in the SDK) to the 25 degrees of freedom of [Tkach et al. 2017]. Some pose angles come with an offset, therefore all angles from the two state-of-the-art gloves are shifted, so that in the first frame of the recorded sequence they exactly match the ground truth. Over the whole sequence, the ManusVR glove has a mean error of 11.93 degrees, the CyberGlove 10.47 degrees and ours 6.76 degrees — this is 35 % lower than the next best result. As the sequences are not exactly

| Network size | FCN | LSTM | ResNet | U-Net | CGAN |
|---|---|---|---|---|---|
| 3M | 6.63 | 5.81 | 6.06 | 5.63 | 5.59 |
| 13M | 6.95 | 6.02 | 6.12 | 5.50 | 5.51 |
| 50M | 7.10 | 6.38 | 6.20 | 5.55 | 5.47 |

**Table 5.4:** *Comparison of different networks for the personalized model in terms of mean angle-error in degrees. (2). From left to right: five different network architectures. From top to bottom: varying amounts of network parameters. We adjust the sizes or numbers of layers for each network to meet the target number of parameters.*

| Network size | FCN | LSTM | ResNet | U-Net | CGAN |
|---|---|---|---|---|---|
| 3M | 7.64 | 7.68 | 7.28 | 6.81 | 7.09 |
| 13M | 7.58 | 7.65 | 7.35 | 6.57 | 6.50 |
| 50M | 7.98 | 7.76 | 7.18 | 6.65 | 6.61 |

**Table 5.5:** *Comparison of different networks for the generic model in terms of mean angle-error in degrees. (4), trained on the leave-one-out dataset. From left to right: five different network architectures. From top to bottom: varying amounts of network parameters, similar to Table 5.4.*

the same, in Fig. 5.13 we additionally show seven poses of the comparison sequence with the corresponding mean error over all degrees of freedom. Fig. 5.12 (top) shows a cumulative error plot comparing the percentage of angular degrees of freedom below a specified error threshold (on the x-axis) for the seven poses shown in Fig. 5.13. We observe that 92% of the angles have an error below 15 degrees for our glove, while for the CyberGlove it is 79% and the ManusVR glove 75%. The lower part of Fig. 5.12 shows a cumulative error plot for the entire comparison sequence.

### 5.4.4 Comparison of networks

We report results from experiments with two 1D baselines (FCN, LSTM) and three types of 2D network architectures: ResNet [He et al. 2016], U-Net [Ronneberger et al. 2015], and conditional generative adversarial network (CGAN) [Isola et al. 2017] . In Tables 5.4 and 5.5 we compare the five types networks on our personalized model (2) and generic model (4). In general, the 2D-based networks are faster to converge and lead to lower reconstruction error. The performance of FCN is not satisfactory, especially when the training set is not diverse, as in the case of the personalized model. LSTM yields smooth results with higher reconstruction accuracy than FCN, but it

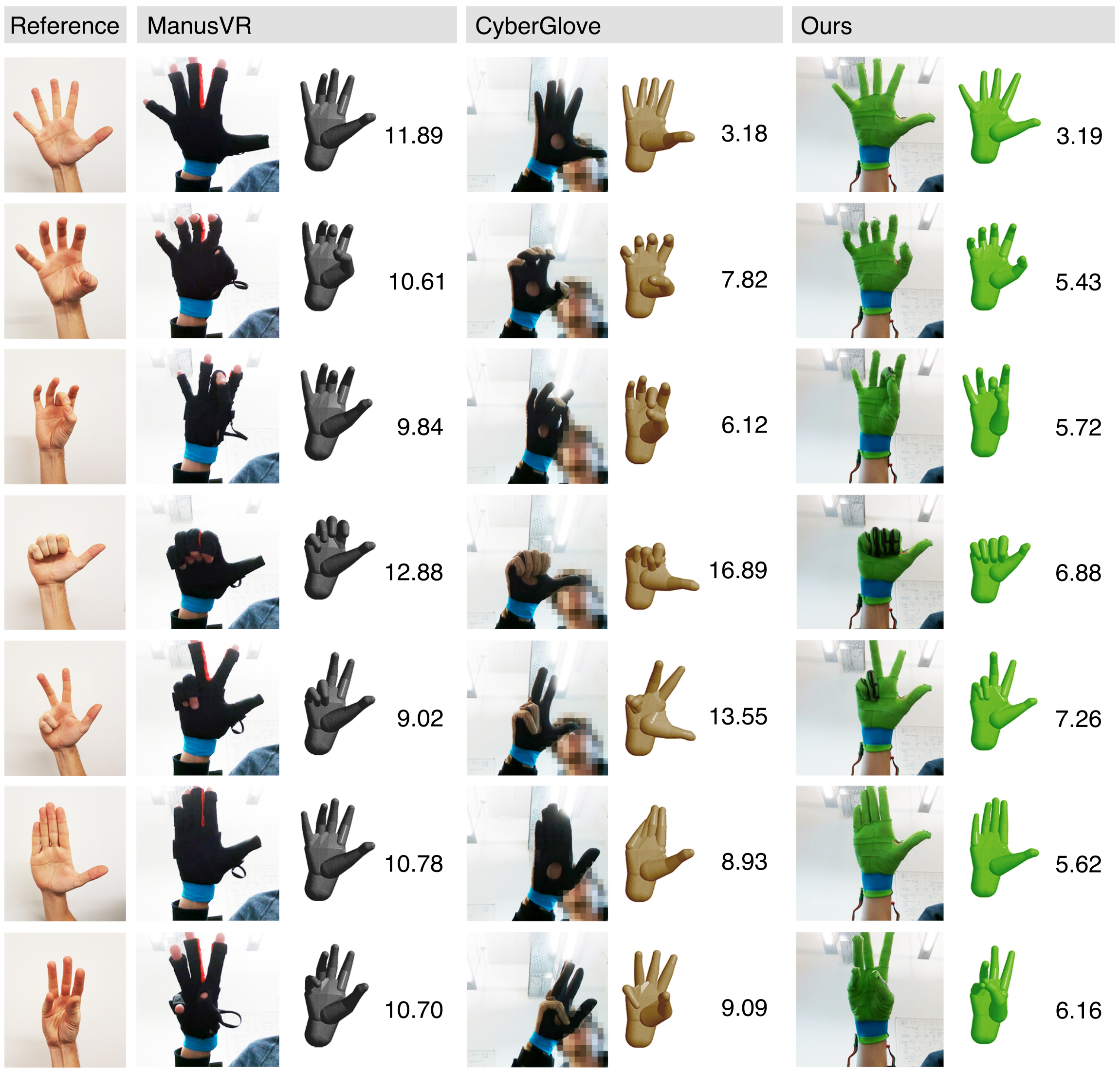


**Figure 5.13:** *Key pose comparison of different gloves: ManusVR, CyberGlove and ours; we show a video frame of the pose, the pose as captured by the glove, and the mean angular error for this specific pose. Note how our glove has the lowest error for every pose but one (first row). To find similar poses, the lowest mean pose parameter difference is used.*

tends to over-smooth some high frequency poses, like the touching of two fingers. Among the three 2D-based networks, ResNet already outperforms the FCN baseline considerably, but leaves room for improvement. Both U-Net and CGAN achieve high reconstruction accuracy. In our experiments, the predicted poses of U-Net are visually more stable than those predicted by CGAN. Thus the 13M U-Net is used for all other experiments. It yields the lowest error for both personalized and generic models (Tables 5.4 and 5.5). Experiments with networks with fewer than 3M parameters lead to an increased error. For comparison, we also trained an SVM on the data of Table 5.5, which results in a higher but still acceptable error of 7.8 degrees.

The models compared here cover a broad spectrum of modern machine learning techniques. An exploration of more advanced network architectures against our baselines, like a combination of LSTM and CNN, would be an interesting direction. Hence, we release all our training data[1].

## 5.5 Conclusion

For now, we focused on the core task of a data glove — capturing accurate hand poses. Furthermore, an optimal data glove should be comfortable to wear, real-time, low cost and easy to use. We achieve these goals via several technical contributions, including a glove-adapted stretch sensor layout and fabrication, a wearable composite of silicone and textile layers, an improved sensor readout scheme to achieve interactive frame rates, a structure-aware data representation and a minimal on-the-fly calibration method. Extensive experiments exploring different scenarios demonstrate the power of our data-driven model and the capabilities of the proposed stretch sensing glove.

To further improve the functionality and applicability of our glove, some essential features and many intriguing extensions are to be explored in the future.

**Applicability.** In the presented state, our glove does not come with a global translation and rotation tracking and is still cable-bound. Position and orientation tracking are essential for a "real-world" data glove. Removing the need for a cable (e.g., adding a battery and wireless data transmission) is a well-studied task. To obtain global translation and rotation information, the straightforward solution would be to use an off-the-shelf tracker (e.g., [Viv 2019]). Such a solution needs an extensive setup and still suffers from occlusion. Alternatively, an experimental setup of a sparse set of additional stretch sensors on the arm might allow tracking the hand position. For the

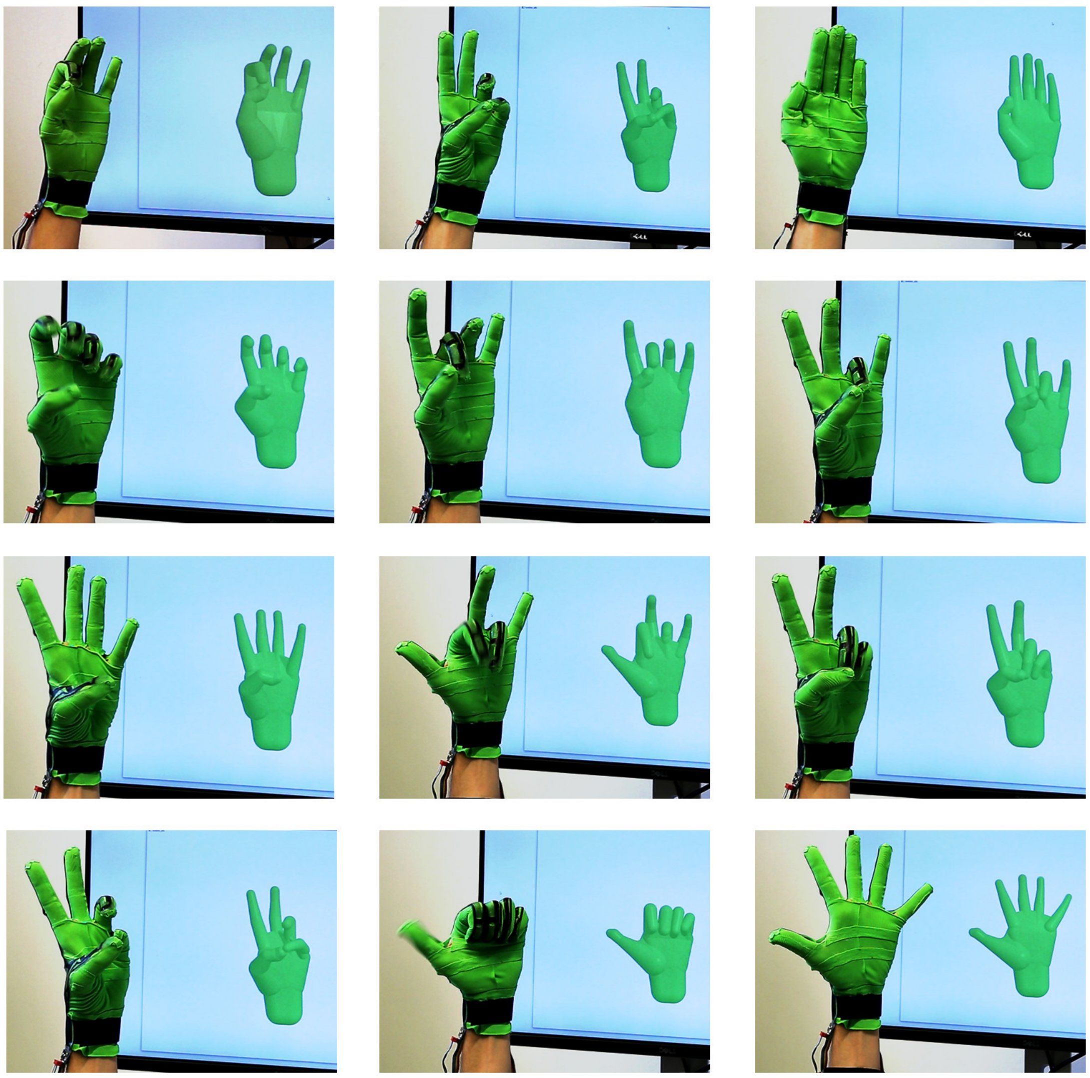


**Figure 5.14:** *A gallery of real-time session frames showing diverse poses predicted by a personalized model.*

wrist and the elbow, we have already demonstrated in Chapter 4 how stretch sensors could provide high-quality surface tracking. The efficient fabrication of such gloves at a larger scale requires further research and development. For the fabrication of the (flat) silicone sensors, a conveyor belt system combining the necessary production steps (casting, curing, laser cutting, and cleaning) is conceivable, while the textile glove part would probably need more fundamental adaptations to be better suited for further automation.

**Noise and latency.** Remaining prediction inaccuracies may be due to the following sources: noise and latency of the sensor readings, material hysteresis, training dataset size and overfitting. In the future, we will research which part contributes to the overall systematic error the most. Empirically, we believe adding more training data and reducing sensor noise are the most promising directions to minimize jitter. The overall latency as seen in the accompanying video ranges from 125 to 200 ms. About 45-90 ms of the latency is due to the oldest of the 180 readings. The inference time of the network model is about 5 ms. The remaining lag comes from un-optimized data communication, filtering, and rendering. For an example of sensor readings and pose predictions over time see Fig. 5.15.

**Customization.** So far we only fabricate medium (M) sized gloves, which are already able to handle a large variety of hands, as demonstrated in Table 5.1. However, we also observe that too small or too large hands can lead to a lower reconstruction accuracy. Therefore, likely two more sizes (an S and an L) are required. Per-person bespoke gloves and how they could further improve the accuracy is another promising direction for future research, especially since our fabrication pipeline trivially allows for adjustments of the size, shape and layout of the sensors. It is conceivable that even better sensor layouts could be found by an optimization based on a simulation, captured data, or a combination thereof.

**Extensions.** Employing a more involved motion tracking system like [Romero et al. 2017] to acquire training data would be more costly, but could also lead to improved accuracy. In many application scenarios (e.g., when used in combination with AR or VR headsets) cameras are already present — even though often with occlusion and out of field-of-view situations. Therefore, it would be interesting to explore how sensor readings from our glove can be fused with camera based pose predictions. A dense stretch sensor glove might be able to predict not only hand pose but also hand shape parameters. Our soft and thin glove is an ideal candidate to be worn below haptic devices [Hinchet et al. 2018] or soft hand exoskeletons [Polygerinos et al. 2015] that do not come with built-in hand pose capture sensors.

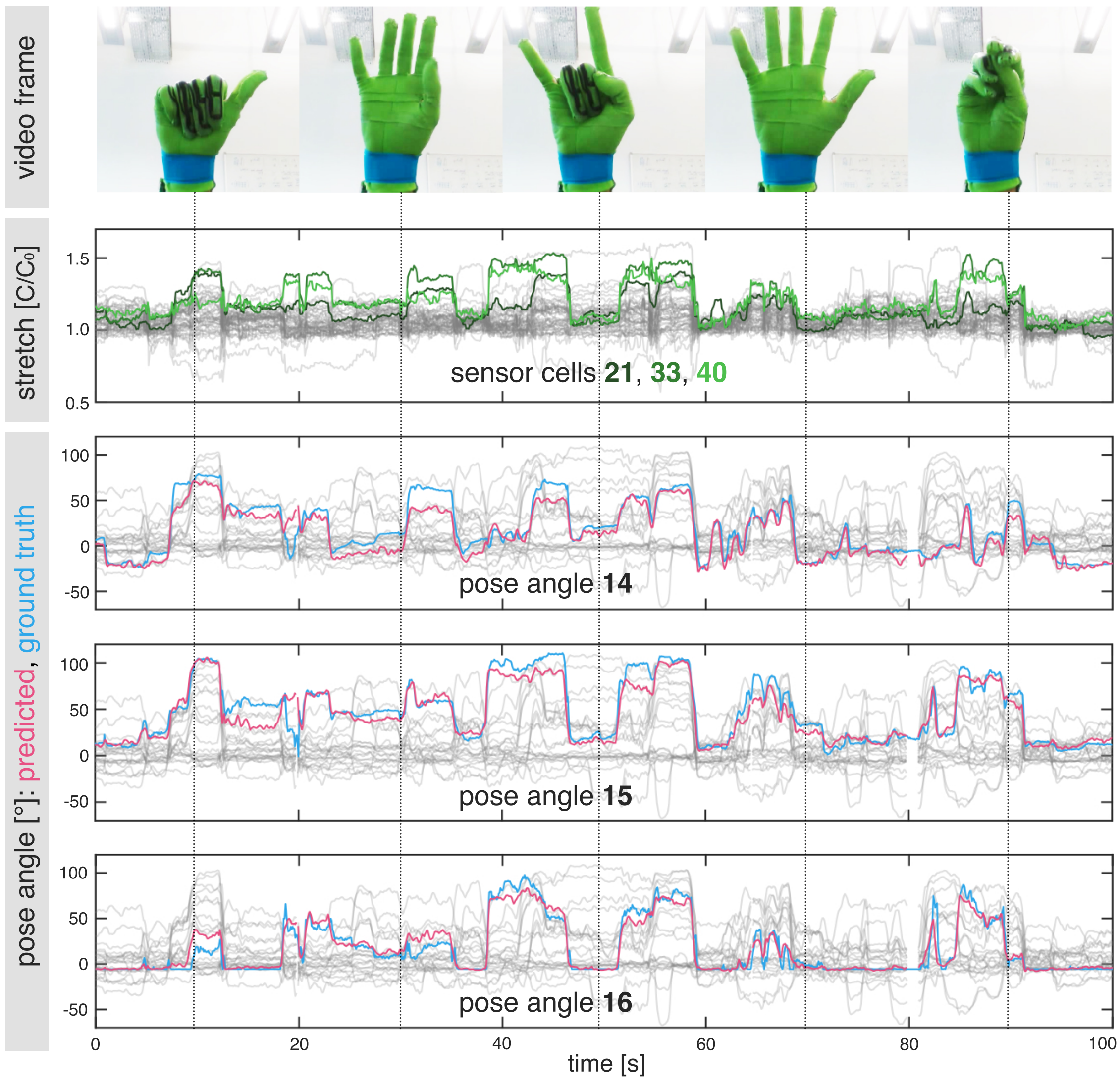


**Figure 5.15:** *Visualization of sensor readings and pose reconstruction over time (x-axis). From top to bottom: Five video frames (the dotted lines indicate the correspondences); plot of stretch sensor readings (y-axis), showing all 44 in light gray, with the readings of the three cells (21, 33, 40 in Fig. 5.3) over the knuckles on the index finger highlighted in green; finally, three plots of predicted and reference pose angles (y-axis), showing all 25 predicted angles in light gray, with the three flexion angles (14, 15, 16 in Fig. 5.7) of the index finger highlighted in red, and the reference angles captured by [Tkach et al. 2017] in blue.*

# CHAPTER 6

# Conclusion

In this chapter, we will conclude the thesis by summarizing its main contributions and by discussing potential future extensions and applications.

## 6.1 Contributions

In this thesis, we propose novel hardware devices and complementary algorithms for the capture of moving and deforming shapes. A key property of the introduced devices is that they do not rely on cameras or any other external instrumentation. The *self-sensing* devices demonstrate how internal sensors that record changes in angles (Hall effect sensors) and area (capacitive stretch sensors) can be applied for digital motion and deformation acquisition. The co-designed software and hardware solutions attain the required precision for professional use but also provide generality and ease-of-use to make the technologies accessible to wider audiences.

In Chapter 3, we introduce a symbiotic pair of software and hardware to help animators traverse a large space of poses via fluid tangible manipulation. The key insight is to drive only the most important degrees of freedom of a rig directly, limiting physical device size and simultaneous manipulations. Furthermore, we demonstrate that our hardware provides a much-improved user experience, resembling inverse kinematics. Empirically we show that this yields speeds-up of a factor of 2 compared to the most related work in terms of character posing time. Together, our software and hardware are an important step beyond static keyframing and into the territory of motion sculpting. We believe our approach provides a valuable toolchain to quickly sketch animations using a variety of professional rigs.

In Chapter 4, we introduce a soft and stretchable capacitive sensor array concept that allows measuring dense, localized area changes. Furthermore, we introduce a pipeline for the fabrication of silicone sheets with two embedded layers of conductive strip patterns. Contrary to existing methods, the proposed technique implicitly provides accurate inter-layer alignment and only requires hardware that is readily available in modern fablabs. We demonstrate how our sensor when paired with a data-driven geometric prior, can be employed to accurately reconstruct complex deformations without line-of-sight.

In Chapter 5, we introduce a data glove for capturing accurate hand poses, building on the dense area sensor arrays. An optimal data glove should be comfortable to wear, real-time, low cost and easy to use. We achieve these goals via several technical contributions, including a glove-adapted stretch sensor layout and fabrication, a wearable composite of silicone and textile layers, an improved sensor readout scheme to achieve interactive frame rates, a structure-aware data representation, and a minimal on-the-fly calibration method. Extensive experiments exploring different scenarios verify the power of our data-driven model and the capabilities of the proposed stretch sensing glove.

## 6.2 Future Work

First, we will summarize the limitations and direct extensions to explore. In-depth discussions are already provided in the respective chapters and therefore, we will only highlight the most intriguing future avenues. In the second part, we provide a broader outlook of specific applications and developments related to the thesis.

### 6.2.1 Limitations and Extensions

Currently, we employ a standard pose interpolation scheme for the rig animation tool. Finding a meaningful mapping from a subset of degrees of freedom to the complete set has a much broader range of applications than just our specific 3D character animation device. Therefore, exploring more sophisticated mapping and interpolation schemes could be very interesting. In terms of hardware design, adding dynamic joint friction control or even actuators could prove to be very useful in many scenarios. To to do so while keeping the current compact form factor will be a very challenging task.

At the moment, the stretch sensor arrays have to be paired with a data-driven prior to capture surface deformation. They would be even more powerful if they would be able to directly capture the deformation of general surfaces (like clothing). This could be achieved by somehow modifying the sensor arrays to measure dense distances instead of areas and to enrich them with bend sensors. Alternatively, it would be promising to explore how a (much) larger dataset of training sequences from multiple users could help to derive a more general data-driven prior. Such a prior might allow capturing multiple users or even unseen users' bodyparts, skipping the current per-user training session. So far, we manually devised our sensor layouts. Computationally designed sensor layouts optimized for a specific object or a type of deformations are another interesting extension that would directly benefit from the flexibility and simplicity of our fabrication pipeline.

In the presented state, our stretch glove accurately tracks the hand's pose but not its global translation and rotation. An intriguing solution might be found by exploring how readings from a sparse set of additional stretch sensors on the arm's joints could be used to estimate the hand's position and orientation. Especially, given that for the wrist and the elbow, we have already demonstrated how stretch sensors can provide high-quality surface tracking. In the context of this work, we only fabricated one sized (medium) gloves that are already able to handle a large variety of hands. But we observe that too small or too large hands can lead to a lower reconstruction accuracy. Likely, at least two more sizes (small and large) are required. In this sense, it would also be interesting to investigate per-person custom gloves and how they could help to further improve the accuracy, especially since our digital fabrication pipeline trivially allows for adjustments of the size, shape, and layout of the sensors. Furthermore, it would be exciting to try to find even better sensor layouts by an optimization based on a simulation, captured data, or a combination thereof. Existing data gloves, including ours, cannot estimate both the shape and the pose of a hand. Due to the type and a large number of sensors, our glove, and a model trained on a big enough dataset, should be able to predict hand shape parameters, too. The correct hand shape is crucial for plausible and accurate finger-object or finger-finger interactions in augmented and virtual reality applications.

### 6.2.2 Outlook

For the presented self-sensing devices and complementary algorithms, there are many more applications than just the ones we directly proposed and explored. For the tangible input device, we focused on character animation for now. But we believe that our hardware can be useful e.g., as a general input device for gaming or to control a modular robot. And for the stretch sensor arrays, we only demonstrated them in the context of capturing dense surface deformation of body parts but there are many other promising directions to investigate, for instance, as a general input device e.g., for gaming, as a soft robot's skin, or for surveillance tasks in medicine or of civil structures. For the data gloves, there's already concrete interest in using them for ergonomics testing of cars or surgical tools, for hand motion capture for films or to use it in virtual reality applications. The following parts are a sampler of specific applications and related areas where we see promising future opportunities for our work.

**3D Animation Training.** After publishing our work on the rig animation input device and releasing the according video online, we got numerous inquiries from interested novice animators. But even more noteworthy was the interest from schools for 3D animation to use our tool to train future professionals. We still hope our device or a similar one could one day be made available for a broad audience like for kids as a toy, for novice or amateur animators and animation students.

**Tangible 3D Modeling.** In the early stages of an animation project, for example, when working on the storyboard, it could be very helpful to not only be able to efficiently draft 3D character animations but also have a tool to design, or should we say build, the characters themselves. Our modular tool would be able to do that, too, meaning that as the parts get plugged together, a virtual character with the same topology directly evolves. [Leen et al. 2017] proposed a similar system for static structures. This way, within just a few minutes a character could not only be animated but beforehand also modeled.

**Robot Sensor Skin.** In robotics and computer vision, many powerful vision-based algorithms exist, enabling autonomous robots to map their environment, localize themselves and interact with their surroundings and, in a more limited way, with humans. In addition to cameras, robots are mostly sensorized with IMUs and often with some sparse force sensors. In comparison with humans, it becomes apparent that an essential sensor is missing: The human skin. It lets us feel touch, stretch, pressure, and temperature. The skin sensor is crucial in any setting with (partial) occlusion of the eye sensor's field of view. Examples are grabbing an object and being able to estimate its shape or sensing body contact under occlusion by a blanket or a table.

Furthermore, the skin sensor is very important for providing warning signals. Humans immediately feel when they just lightly touch something with their leg, arm or feet, even when their eyes (cameras) are looking somewhere else. These early warnings prevent bumping into objects and eventually, getting out of balance and falling. In the context of a robot, such a dense skin sensor can potentially save lives if the object hit is a human: The skin sensor's signal will make the robot stop immediately. In a way, robots without a dense sensor skin are like humans wrapped into a thick plastic foil preventing the skin from sensing.

Not surprisingly, there's already a large body of work (e.g., [Mittendorfer and Cheng 2011] or very recently [Sundaram et al. 2019; Duan et al. 2019]), suggesting such sensor skins for robotics and related applications. The introduced dense matrix concept and most importantly the simple and cheap fabrication technique could be another important step towards the broader use of low-cost dense sensor skins. Due to the capacitive sensor principle, our dense stretch arrays can be easily adapted to measure pressure instead of stretch or even both quantities at once.

Very related is the work by [Kim et al. 2018]. They mount inflated balloons around rigid robotic arms to achieve a very soft skin to protect the robot and collaborating humans. Our dense stretch sensors would be an optimal fit for such balloons' hulls being able to track its shape and occurring contacts.

**Soft Robotics.** The rapidly progressing field of soft robotics [Bao et al. 2018] is another exciting discipline. Soft robots are often just employing an open-loop control strategy [Wang et al. 2018]. It would be exciting to use our silicone sensors as a skin for soft robots allowing for real-time estimation of its shape and consequently, to close the control loop. There are closely related opportunities for deployment in soft exoskeletons and prosthetics [Asbeck et al. 2014].

**Human Capture.** Dense surface tracking without a line of sight has many more applications in human capture than just digitalizing motions for films and games. In rehabilitation, it could help to first assess the patient's physical abilities and then survey progress during rehabilitation training. In the same way, athletes could keep track of their fitness state and progress, and analyze and improve their technique of sports-specific movements. In this context, it would be beneficial to extend the sensors to additionally measure quantities such as sweat using exposed electrodes, the heartbeat or breathing volume and frequency by measuring stretch or the temperature through changing capacitance or resistance, while, in an optimal scenario, keeping the sensor fully soft.

**Advances in Materials and Fabrication.** For applications involving humans, breathability and minimizing the mechanical resistance are crucial in making the sensor arrays comfortable to wear. We experimented with ventilation cuts for the data gloves which led to a significantly improved experience, compared to the bare sensor arrays. But there's unexploited potential for adding more and larger openings to cover as little surface of the human skin as possible. This would also make it easier and faster to put the sensor on, which is a major criterion of making wearables being used.

In material science, there's a large and growing body of research on improving the electrical conductivity and robustness of stretchable materials, and their production [Matsuhisa et al. 2019]. In our case, we chose the mix of silicone and carbon black for our conductors due to the availability and ease of handling. Our sensor would directly profit from employing advances in materials, for example, to make them thinner and increase the sensor cell density.

Broader use of our sensors and gloves would require an adapted fabrication method. The current laser-cutter-based technique is great for prototyping and customization but for a larger-scale production faster processes are required, promising candidates are variants of masking, casting or stamping.

**Self-Sensing Cloth.** Imagine a fully, or at least mostly, soft and stretchable cloth that can sense its shape with a reasonable resolution. Such a cloth could be employed as a skin for robots, to digitalize the motion of humans or be embedded in passive objects like beds or chairs.

The development of such a cloth is a very challenging problem. Taking our stretch sensors matrix as a starting point, adding cuts along the sensor cells could turn the area sensors into directional distance sensors. To get the missing bending measurements Hall effect sensors (like in the character animation device) could be employed, this would mean to embed pairs of small magnets and receiving conductive strips into the silicone sheet. To ensure that the different deformation modes happen where the according sensors are placed, the cloth would probably have to be locally rigidified. But it's conceivable instead of an engineered deformation model, using a data-driven could help to avoid that because this way the sensor readings do not need to be actually physically meaningful but only repeatable, as we already demonstrated for hand pose estimation with the glove. In general, it would be promising to further explore how neural networks can be used to make sense of (to a human mind) physically non-meaningful measurements.

A strength of our work lies in having developed and demonstrated a full pipeline including the sensor concept, its fabrication, showing how to capture dense deformation and how to successfully apply it in the well-researched field of capturing hands. In the next step, we can improve and iterate on single pipeline parts, and either push towards a self-sensing cloth vision or make the sensor perfectly adapted to one of the specific applications described.

**AI in Cyber-Physical Systems.** For data-driven approaches, the acquisition of large enough bodies of suitable training data becomes a major challenge. For our sensor and specifically the glove, we found methods to efficiently capture sufficient data to successfully demonstrate an example of artificial intelligence in a cyber-physical system. We believe that more training data could further increase the reconstruction quality and generalization ability of our examples.

Possible additional sources to acquire more training data, besides the already employed systems, are simulations, pose estimation from 2D images (e.g., from a webcam stream) and even more accurate multi-camera setups as employed by [Romero et al. 2017]. Since the resulting training dataset would stem from this set of sources having different accuracy levels, the interesting question becomes how to optimally combine them to arrive at the best model. [Mayer et al. 2018] investigate this common problem for the case of learning disparity and optical flow estimation.

**Virtual and Augmented Reality.** It would be fascinating to bring our modular input device into VR or AR by directly overlaying the device skeleton with the character mesh, making the animation process feel even more tangible. The current hardware design only allows for two discrete bone lengths between joints. This results in length differences between the tangible animation device and the virtual character's skeleton. The challenge would be to find an elegant solution to deal with this disparity.

Recent commercial VR (e.g., Oculus Quest) and AR (e.g., Microsoft HoloLens) headsets already include cameras to map the environment, to estimate the spatial and rotational orientation and to track the body, including the hands. But in many situations, these cameras do not have an open line of sight and will, therefore, struggle to reconstruct the hand. Examples are hands being occluded by themselves, by the body, due to grabbing an object, behind an object or simply by being out of the field of view, e.g. when they are behind the back. In these situations, a wearable sensor such as glove can take over to deliver a seamless and robust tracking experience which is crucial for hand-tracking to be able to replace the current physical controllers. And, whenever the cameras have a full view of the hand both sensor sources (cameras and glove) can be fused to provide a more accurate result, or the camera data can be used to calibrate the glove or even fine-tune its data-driven model. Another advantage of wearables like gloves is the option to add haptic feedback. By partially rigidifying sensor cells, having a thick enough dielectric layer and applying high voltages, it's conceivable to deliver tactile feedback with only slightly adapted stretch sensor arrays. [Rosset and Shea 2013] explain the principle of such elastomer actuators and provide examples.

APPENDIX A

# Modular Input Device for Rig Animation

## A.1 Fault-resistant distributed protocol

The presence of slip-rings makes the electric connections between the pieces unreliable: when many pieces are moved at the same time, it is common for pieces to temporarily lose power or get disturbed on the data channel that corrupts some packages. We evaluated the protocol proposed by [Jacobson et al. 2014b], but discovered that it is not applicable to our scenario, since it requires the global bus to be stable for long periods to do a synchronized read on all sensors. We propose a fault-resistant protocol that can gracefully recover from random disconnects or missing packages, while allowing to easily reconstruct the topology of the device.

We assign a global identifier to all components (the hash of the timestamp of the programming time) and we equip each component with two messaging queues, one for incoming and one for outgoing messages. Each component acts as a repeater for messages that come from pieces below it, sending them to its parent. All messages contain the ID of the creator of the message, the ID of the parent, the message itself and a consistency hash to detect communication errors. The ID of the parent is initialized as empty by the message creator; each time a message is received with an empty parent ID, the receiving component fills the field with its own ID. The message distribution is done using a polling mechanism: every 5 ms, each component probes its children for messages, and queues messages, which are in turn passed to the parent when queried.

Our protocol supports three types of messages, generated by the pieces at different frequencies: data messages (50 Hz), which contain angle readings, type messages (2 Hz), which contain the color and type of the component, and child messages (2 Hz), which describe the rotation associated with each slot of a splitter. This protocol robust thanks to its distributed nature and built-in error checking: the topology can be reconstructed by storing the parent-child relations in a hash table, and disconnected pieces can be detected by keeping a timestamp for every component. If no messages are received from a component for more than 2 s, then the subtree starting at that component has been detached from the device.

## A.2 Evaluating Pose Reachability

Given a candidate assignment $\mathcal{D}$, we want to find the device poses $\mathbf{P}^{\mathcal{D}}$ which minimize the energy in Eq. (3.1). Each pose $\mathbf{p}_i^{\mathcal{D}}$ of the device should be optimized to lead to the corresponding sample pose $\mathbf{p}_i^{\mathcal{C}}$, and it is uniquely defined by a set of rotations $R_{ij}^{\mathcal{D}}$, each associated with a joint $j$. The assignment $\mathcal{D}$ associates the rotations $R_{ij}^{\mathcal{D}}$ to rotations in the rig; as discussed in Sec. 3.2.2, all the unassigned rig nodes inherit the rotation of their parent.

The overall pose error $E$ is defined as the sum of the individual pose errors $E_i$ (Eq. (3.1)). Since we have a fixed candidate assignment $\mathcal{D}$, this error can be optimized separately for each pose:

$$E = \arg\min_{\mathbf{P}^{\mathcal{D}}} \sum_i E_i(\mathbf{p}_i^{\mathcal{D}}) = \sum_i \arg\min_{\mathbf{p}_i^{\mathcal{D}}} E_i(\mathbf{p}_i^{\mathcal{D}}). \tag{A.1}$$

For the sake of clarity, from now on we omit the index $i$, since all the equations are referred to a single pose (except $w_b$, which are constants shared by all poses):

$$E_i = \arg\min_{R^{\mathcal{D}}} \sum_b^m w_b |\mathbf{n}_b - \tilde{\mathbf{n}}_b(R^{\mathcal{D}})|_2^2. \tag{A.2}$$

The weights $w_b$ account for the difference in surface area (see Sec. 3.2.2). Eq. (A.2) highlights the variables we wish to optimize for, that is, the rotations of the device that induce a deformation of the rig with nodes $\tilde{\mathbf{n}}_b$ that better approximate the position of the sample nodes $\mathbf{n}_b$. We denote by $\tilde{\mathbf{n}} = (\tilde{\mathbf{n}}_0, \tilde{\mathbf{n}}_1, ..., \tilde{\mathbf{n}}_m)$ the induced node positions through the device $\mathcal{D}$ in the sample pose; $b$ is a stack of the rest pose bones of rig $\mathcal{C}$ and $H$ a $3m \times 3m$

matrix encoding the rig hierarchy, and

$$\tilde{R}^{\mathcal{C}} = M_{\mathcal{D}} R^{\mathcal{D}}, \tag{A.3}$$

where $M_{\mathcal{D}}$ is a $3m \times 3l$ matrix representing the map from the pose of the device to the set of bone rotations $\tilde{R}^{\mathcal{C}}$ ($3m \times 3$ matrix) in rig pose $\tilde{\mathbf{p}}^{\mathcal{C}}$, $R^{\mathcal{D}}$ is a $3l \times 3$ matrix vertically stacking joint rotation matrices $R_j^{\mathcal{D}}$, and $l$ is the number of joints in device $\mathcal{D}$. The value of $\tilde{\mathbf{n}}$ relates to $R^{\mathcal{D}}$ in the following way:

$$\tilde{\mathbf{n}} = H \operatorname{diag}(\tilde{R}^{\mathcal{C}}) b = H \operatorname{diag}(M_{\mathcal{D}} R^{\mathcal{D}}) b, \tag{A.4}$$

where $\operatorname{diag}(\tilde{R}^{\mathcal{C}})$ is a $3m \times 3m$ matrix with the vertically stacked rotation matrices in $\tilde{R}^{\mathcal{C}}$ on its diagonal.

Note that the pose error is a sum of squares (Eq. (A.2)), which we optimize using the Gauss-Newton minimization method. For the optimization update steps, we use a differential representation, where each rotation associated with a joint $j$ is encoded as a vector $\omega_j = (\omega_{j,1}, \omega_{j,2}, \omega_{j,3})$:

$$R_j^{\mathcal{D}^{k+1}} = R_j^{\mathcal{D}^k} e^{J(\omega_j)}, \tag{A.5}$$

where $R_j^{\mathcal{D}^k}$ is a fixed rotation computed at the previous step, and $e^{J(\omega_j)}$ is the change computed in the current iteration. The optimization variables are the entries of $J(\omega_j)$, which is defined as:

$$J(\omega_j) = \begin{bmatrix} 0 & -\omega_{j,3} & \omega_{j,2} \\ \omega_{j,3} & 0 & -\omega_{j,1} \\ -\omega_{j,2} & \omega_{j,1} & 0 \end{bmatrix}. \tag{A.6}$$

This exponential map parametrization of the rotation space elegantly and efficiently ensures, without additional constraints, that $R_j^{\mathcal{D}^{k+1}}$ is a rotation. We stack all the variables in a vector $\omega = (\omega_1, \omega_2, ..., \omega_l)$. In every iteration, we set $\omega$ to:

$$\omega = -(D^T D)^{-1} D \mathbf{r} \tag{A.7}$$

where $\mathbf{r}$ is a $3m$-vector

$$\mathbf{r} = \mathbf{W}(\mathbf{n} - \tilde{\mathbf{n}}(R^{\mathcal{D}^k})) \tag{A.8}$$

and $D$ is a $3m \times 3l$ matrix whose element at row $a$ and column $c$ is:

$$D_{ac} = \frac{\partial \mathbf{r}_a}{\partial \omega_c}. \tag{A.9}$$

The optimization is initialized with the rotations of the sample pose. Note that this initial guess does not reproduce the sample pose, since not all bones have a joint assigned in $\mathcal{D}$. We stop the iterations when:

$$|E_i^{k+1} - E_i^k| \leq |E_i^{k+1}| \cdot 10^{-6} + 10^{-6}. \tag{A.10}$$

## A.3 User study details

We ran an experiment to compare *our* design to [Jacobson et al. 2014b], which in turn already establishes a baseline of equivalence to mouse and keyboard. We asked 10 users (2 female, 8 male) to participate in our experiment (cf. Sec. 3.3). Most of the users reported some prior experience with 3D tools and even with Maya, but none were professional animators. The device presentation was balanced, and exactly half of the participants started with ours device. We instructed the participants to adjust an on-screen character's pose according to a semi-transparent target pose. The participants self-reported as soon as they considered their posing to be "good enough". The procedure consisted of one practice block to familiarize oneself with the device, followed by three blocks of timed posing. After a relaxation period, we repeated the procedure with the second device. Finally, we conducted exit interviews with each participant.

**Experiment metrics.** Based on the three timed blocks, we computed three metrics. First, *task completion time* is the elapsed time until the minimal pose error is reached in each posing task. Second, we recorded the *minimal pose error* [%] reached per pose. Finally, we computed the *work* metric, as proposed in [Jacobson et al. 2014b]. This is the integral of the pose error [%] over time (cf. Fig. 3.10). In all three metrics, pose error percentages are always in relation to the initial pose error.

**Additional results.** For an in-depth discussion of results please refer to Sec. 3.3. Fig. A.1 shows that our device outperforms [Jacobson et al. 2014b] in all of the three metrics.

In addition to the above metrics, we also computed absolute angular errors and compared these across device designs. Note that the presented numbers should not be confused with the accuracy of our device (0.5 degrees).

The absolute average angular error per joint, as reached by our *best* participant, was 5.06 degrees (elapsed time: 216.3 s). In comparison, the same user achieved an average error of 14.51 degrees (elapsed time: 245.5 s) with the old design.

The minimal absolute average angular error a user reached was 6.32 degrees with *both* devices (rounded to 2 digits). However, the user reached this minimal error 41.7 s faster with our device (126.6 s) than with [Jacobson et al. 2014b] (168.3 s).

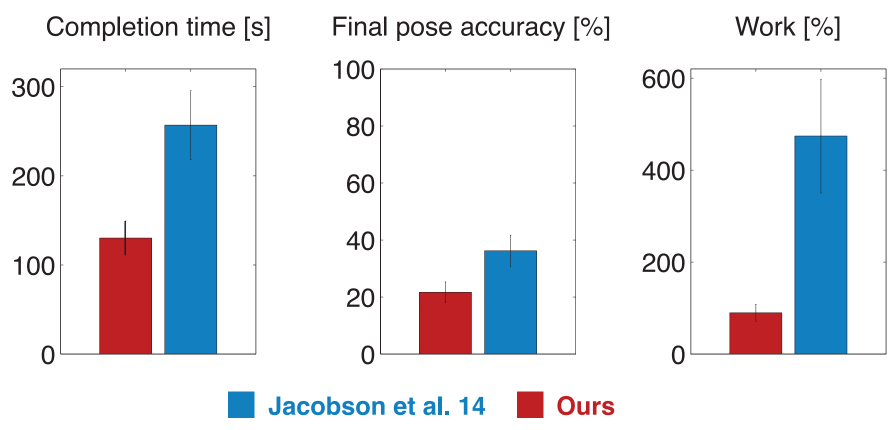


**Figure A.1:** *Additional experimental results: Comparing our design (red) to the design of [Jacobson et al. 2014b] (blue). Our device performs better in every of the three metrics (time, accuracy, work).*

The final average angular error (across all participants) for our device was 11.07 degrees (standard deviation: 4.51 degrees) versus 19.68 degrees (SD: 10.40 degrees) for [Jacobson et al. 2014b].

# APPENDIX B

# Stretch Sensor Arrays for Deformation Capture

## B.1 Silicone Mixtures

We used the following mixtures for the three types of silicone layers:

**Protective layer:** Silbione RTV 4420 [Sil 2018] component A (weight ratio=1.0) and Toluol (1.0) are mixed, then Silbione RTV 4420 (1.0) component B is added.

**Conductive layer:** Silbione RTV 4420 component A (1.0) and Toluol (2.0) are mixed, then Silbione RTV 4420 (1.0) component B is added. In a separate container, Imerys Enasco 250 P [Ens 2018] conductive carbon black (0.2) is mixed with isopropyl alcohol (2.0) by slowly adding the isopropyl alcohol while stirring. Then both compositions are combined and mixed for about 3 minutes. The 2-component silicone Silbione RTV 4420 was chosen due to its tear behavior as evaluated in [Bernardi et al. 2017] and the Imerys Enasco 250 P carbon black as suggested in [Brunne et al. 2011].

**Dielectric layer:** Same as the protective layer.

## B.2 Measurement setup

In our setup capacitance is indirectly measured by timing the charging of a capacitor until a predefined voltage level, since the charging time is linearly

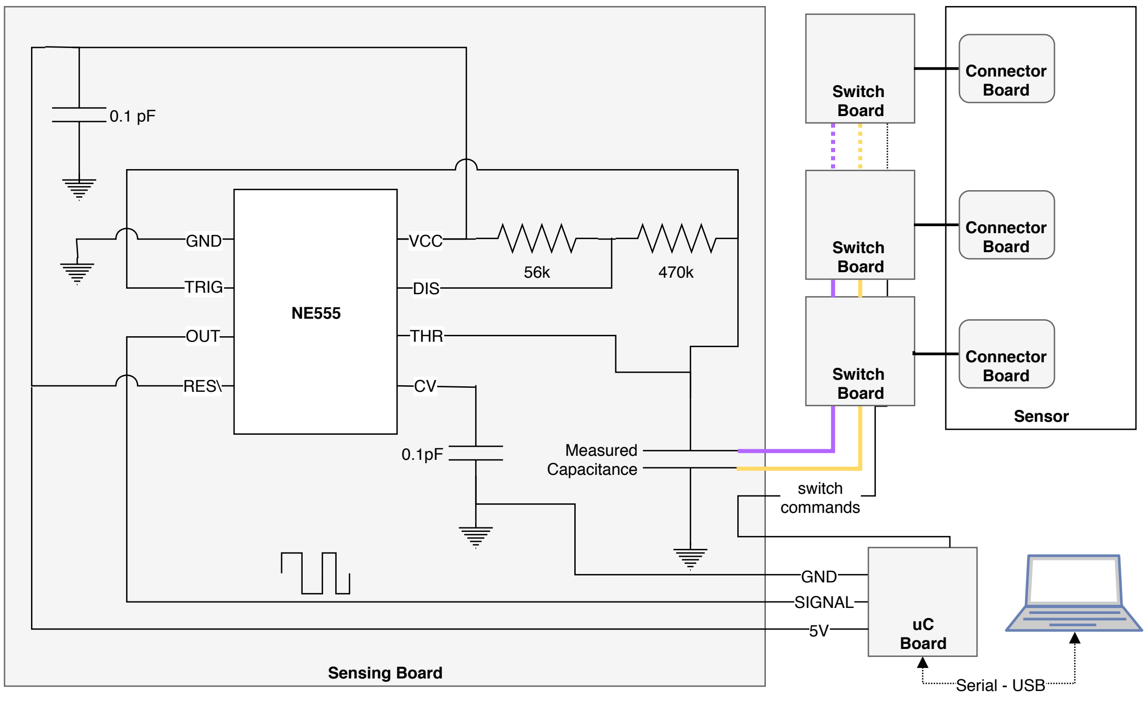


**Figure B.1:** *Our modular setup consists of two parts. Left: The capacitance sensing circuit is implemented with a NE555 timer IC, resulting in a square SIGNAL of the charging time that is read by the uC and sent to the computer. Right: The uC board and the switch boards go through all combinations, dynamically connecting the current set of source electrode strips (purple) and ground electrode strips (yellow); see Sec. 4.3.2 and Fig. 4.6 for details.*

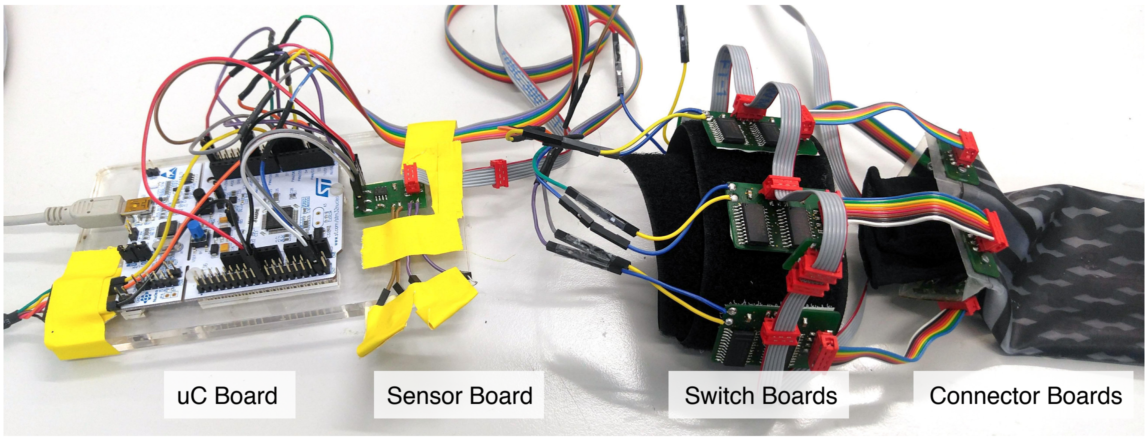


**Figure B.2:** *Our custom modular measurement setup with the four types of boards. Up to 8* switch boards *(and according* connector boards*) can be daisy chained.*

proportional to the capacitance. However, our setting is more challenging, since we have to dynamically reconnect the electrodes following the measurement protocol described in Sec. 4.3.2. For this purpose, we design a modular measuring system (Fig. B.1 right and Fig. B.2), composed of three kinds of custom boards: the *connector board*, which is directly placed in contact with the sensor, the *switch board*, which is connected to the connector board by a set of flexible wires and the *sensing board* that contains the electronics needed to measure the charging times and send them to the connected computer. The connector boards are placed on the sensor on the exposed sensor pads that are shown in Fig. 4.9, supported by a PET foil and screwed into an acrylic counter-holder. The PET foil acts as intermediary from stretchable (silicone sensor), through flexible (PET), to fully rigid (*connector board*). The switch boards enable switching through the sensor combinations and they can be daisy-chained to allow for a wide variety of sensor layouts. The switching is controlled from the *uC board*: A STM32 microcontroller on a NUCLEO-F446RE board [STM 2018]. The microcontroller continuously transmits the charging time measurements to the computer via a USB-serial connection.

The capacitance measuring circuit (Fig. B.1 left) is implemented using a NE555 timer IC. It outputs a square wave SIGNAL with a frequency $f$ which is converted to capacitance by $C = 1/(f \cdot (R_1 + 2R_2) \cdot \ln(2))$, where $R_1$ and $R_2$ are the charging resistors. The larger these charging resistors are, the slower the capacitors are charged and dis-charged and the longer it takes for a complete measuring round (going through all sets of combined electrodes as shown in Fig. 4.6) and get the local capacitance changes updated. Note that our model neglects the influence of the resistance of the electrodes themselves. The full resistance for the longest electrode strip is about 50 kOhm. We experimentally found that setting $R1$ = 470 kOhm and $R2$ = 47 kOhm is a good compromise that produces sufficient accuracy while still supporting an interactive frame rate of 8 Hz. The parasitic capacitance of the circuit has to be subtracted from all the capacitance measurements. This can be simply done by continuously measuring the capacitance between two unconnected connector board pads. A nylon sock is worn below the sensor when capturing human body part deformation. As demonstrated in Fig. B.3, it shields the in silicone embedded capacitor array from body capacitance and lowers the friction between the sensor and the skin, to e.g. pull a cylindrical sensor over a wrist with much less effort.

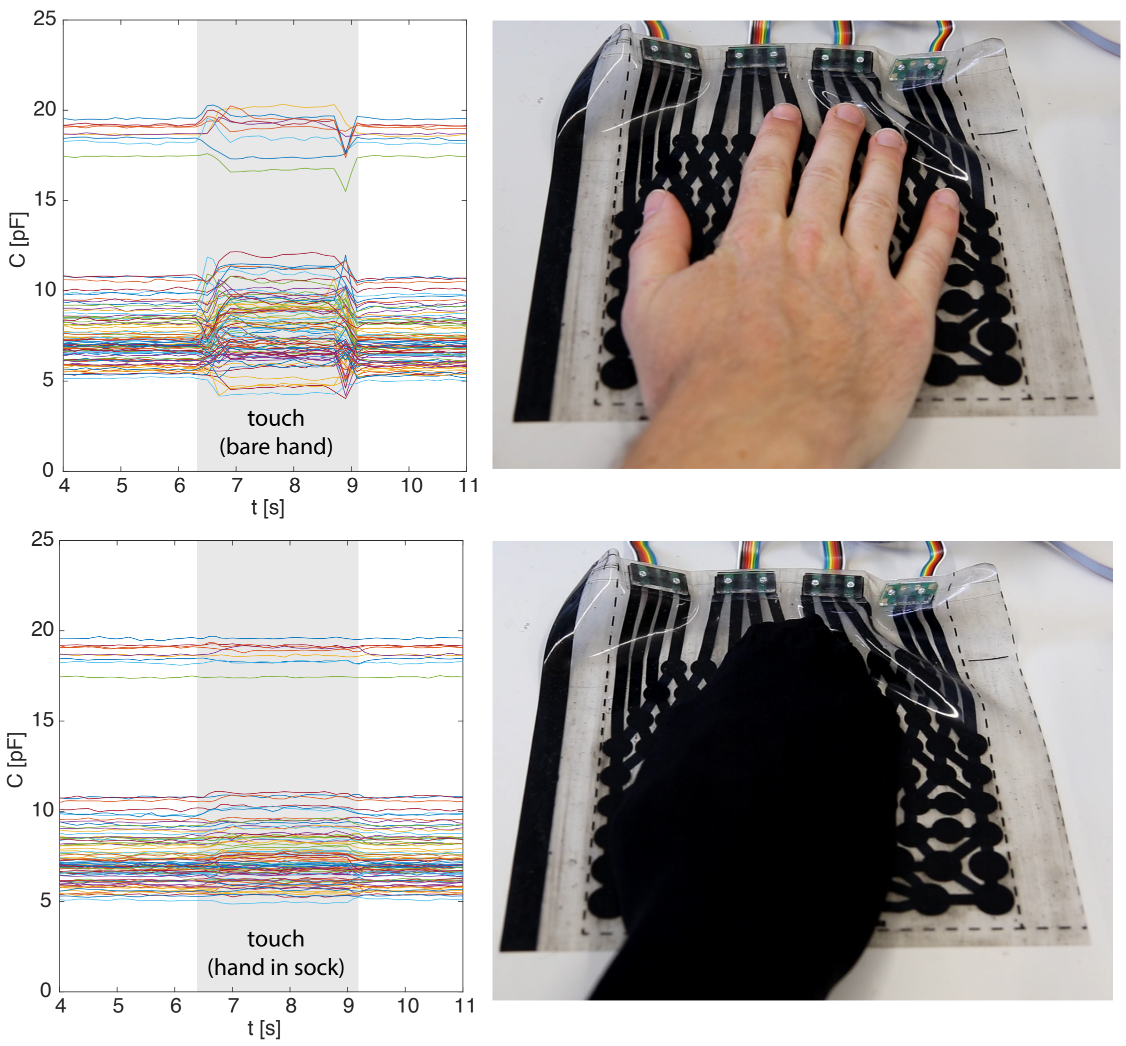


**Figure B.3:** *This experiment demonstrates the effect of the nylon sock, worn below the sensor. Top: If the sensor is touched without the sock, the influence of the body capacitance creates clear spikes in the capacitance measured per sensor cell. Bottom: If the nylon sock is worn the same effect is minimal.*

APPENDIX C

# Hand Pose Estimation with a Stretch Glove

## C.1 Network details

Our model is implemented in Pytorch and trained on an NVIDIA 1080Ti GPU. In the following, we describe the different network architectures for the size of 13M parameters. Networks of other sizes (3M and 50M) have the same structure but different numbers (or sizes) of layers.

For the FCN, we use the same network architecture as shown in Fig. 4.13, i.e., five fully connected layers: F44-F2048-F2048-F2048-F2048-F1024-F26. For the LSTM, we use two hidden layers, and each layer has 512 features in the hidden state. We use a window size of 5 and observe that the bigger the window size, the smoother the reconstruction, but also the lower the reconstruction accuracy. We use a standard SGD optimizer with a learning rate of 0.01 and batch size of 1024 for both the FCN and LSTM.

For ResNet, we use a 2-stride convolution and a 2-stride up-convolution for both the encoder and decoder networks and 12 residual blocks in-between. The architecture of U-Net is shown in Table C.1. The generator $G$ of CGAN has the same structure as U-Net, i.e., C64-C128- C256-C512-C512-C256-C128-C64, while the discriminator $D$ has five convolution layers: C64-C128-C256-C512-C1.

We use the ADAM optimizer ($lr = 0.0002, \beta_1 = 0.5, \beta_2 = 0.999$) for training the CNN networks. Xavier [Glorot and Bengio 2010] is used for weights

| Input → Output Shape | Layer Information |
|---|---|
| (5, 5, 2) → (5, 5, 64) | CONV-(N64, K5×5, S1, P2), ReLU |
| (5, 5, 64) → (3, 3, 128) | CONV-(N128, K3×3, S2, P1), BN, ReLU |
| (3, 3, 128) → (2, 2, 256) | CONV-(N256, K3×3, S2, P1), BN, ReLU |
| (2, 2, 256) → (1, 1, 512) | CONV-(N512, K4×4, S2, P1), BN, ReLU |
| (1, 1, 512) → (2, 2, 256) | CONV-(N512, K4×4, S2, P1), BN, ReLU |
| (2, 2, 256) → (3, 3, 128) | CONV-(N256, K3×3, S2, P1), BN, ReLU |
| (3, 3, 128) → (5, 5, 64) | CONV-(N128, K3×3, S2, P1), BN, ReLU |
| (5, 5, 64) → (5, 5, 1) | CONV-(N64, K5×5, S1, P2), Tanh |

**Table C.1:** *Network architecture of U-Net. N: the number of output channels, K: kernel size, S: stride size, P: padding size, BN: batch normalization.*

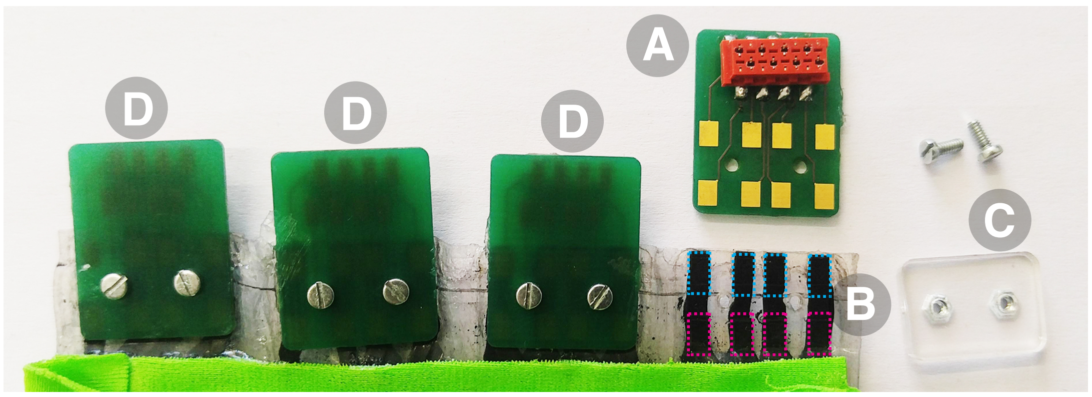


**Figure C.1:** *(A) Connector board with eight pads and connector socket. (B) Exposed lead ends of the bottom (blue) and top (red) layer. (C) Acrylic counter-holder. (D) Three installed connector boards.*

initialization. We use a batch size of 1024 and 256 for training the generic and personalized models, respectively. We choose the model that has the minimal error in the validation set for testing. In general, the training of personalized and generic models takes around 2 and 5 hours, respectively, except that the training time of LSTM is about three-fold. The inference time of a trained model is approximately 0.003 seconds.

## C.2 Interconnections

To connect the individual leads of the fully soft silicone sensor to the read-out circuit (see Appendix B.2 for details) rigid printed circuit boards (PCB) are placed on the exposed sensor leads at the wrist end of the glove, supported by a PET foil and screwed into an acrylic counter-holder, see Fig. C.1. The PET foil acts as intermediary from stretchable (silicone sensor), through flexible (PET), to fully rigid connector PCBs.

# Bibliography

Bando. https://www.youtube.com/watch?v=2XskNHtarj0, 2018.

Imerys ENSACO 250G. http://www.imerys-graphite-and-carbon.com/wordpress/wp-app/uploads/2014/04/Polymer_compounds1.pdf, 2018. Accessed: 2018-01-18.

OptiTrack. http://optitrack.com/products/prime-13/, 2018. Accessed: 2018-01-19.

Parker Hannifin EAP Sensor. http://ph.parker.com/us/en/electroactive-polymer-technology-monitors-movement-and-stretch-eap-sensor-evaluation-kits, 2018. Accessed: 2018-01-18.

STM32 Nucleo-F446RE. http://www.st.com/en/evaluation-tools/nucleo-f446re.html, 2018. Accessed: 2018-01-18.

Silbione RTV 4420. https://silicones.elkem.com/EN/our_offer/Product/90060082/90060081/SILBIONE-RTV-4420-B-U1, 2018. Accessed: 2018-01-18.

StretchSense. https://www.stretchsense.com/, 2018. Accessed: 2018-01-18.

5DT Glove. http://www.5dt.com/data-gloves/, 2019.

CyberGlove. http://www.cyberglovesystems.com/, 2019.

HT2 Textile Glue. https://consumer.guetermann.com/en/product-finder/textile-glue-ht2, 2019.

ManusVR glove. https://manus-vr.odoo.com/hardware, 2019.

Intel Real Sense SR3000. https://software.intel.com/en-us/realsense/sr300, 2019.

Sil-PoxySilicone Adhesive. https://www.smooth-on.com/product-line/sil-poxy/, 2019.

Stretchsense. https://www.youtube.com/watch?v=XJCqdpzxMME, 2019.

VPL Data Glove. https://www.britannica.com/technology/VPL-DataGlove, 2019.

Vicon. http://www.vicon.com, 2019.

HTC Vive Tracker. https://www.vive.com/eu/vive-tracker/, 2019.

M. Achibet, G. Casiez, A. Lécuyer, and M. Marchal. Thing: Introducing a tablet-based interaction technique for controlling 3D hand models. In *Proc. CHI*, 2015.

J. Ahn and K. Wohn. Motion level-of-detail: A simplification method on crowd scene. In *Proc. CASA*, 2004.

O. A. Araromi, S. Rosset, and H. R. Shea. High-resolution, large-area fabrication of compliant electrodes via laser ablation for robust, stretchable dielectric elastomer actuators and sensors. *ACS Applied Materials & Interfaces*, 7(32):18046–18053, 2015. doi: 10.1021/acsami.5b04975. URL http://dx.doi.org/10.1021/acsami.5b04975. PMID: 26197865.

A. T. Asbeck, S. De Rossi, I. Galiana, Y. Ding, and C. J. Walsh. Stronger, smarter, softer: Next-generation wearable robots. *IEEE Robotics & Automation Magazine*, 21(4):22–33, 2014. URL http://dx.doi.org/10.1109/MRA.2014.2360283.

A. Atalay, V. Sanchez, O. Atalay, D. M. Vogt, F. Haufe, R. J. Wood, and C. J. Walsh. Batch fabrication of customizable silicone-textile composite capacitive strain sensors for human motion tracking. *Advanced Materials Technologies*, 2 (9):1700136–n/a, 2017. ISSN 2365-709X. doi: 10.1002/admt.201700136. URL http://dx.doi.org/10.1002/admt.201700136. 1700136.

O. K.-C. Au, C.-L. Tai, H.-K. Chu, D. Cohen-Or, and T.-Y. Lee. Skeleton extraction by mesh contraction. *ACM Trans. Graph.*, 27(3), 2008.

M. Bächer, B. Hepp, F. Pece, P. G. Kry, B. Bickel, B. Thomaszewski, and O. Hilliges. Defsense: Computational design of customized deformable input devices. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 3806–3816, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3362-7. doi: 10.1145/2858036.2858354. URL http://doi.acm.org/10.1145/2858036.2858354.

R. Balakrishnan, G. Fitzmaurice, G. Kurtenbach, and K. Singh. Exploring interactive curve and surface manipulation using a bend and twist sensitive input strip. In *Proceedings of the 1999 Symposium on Interactive 3D Graphics*, I3D '99, pages 111–118, New York, NY, USA, 1999. ACM. ISBN 1-58113-082-1. doi: 10.1145/300523.300536. URL `http://doi.acm.org/10.1145/300523.300536`.

L. Ballan, A. Taneja, J. Gall, L. Van Gool, and M. Pollefeys. Motion capture of hands in action using discriminative salient points. In *Proc. ECCV*, pages 640–653. Springer, 2012.

G. Bao, H. Fang, L. Chen, Y. Wan, F. Xu, Q. Yang, and L. Zhang. Soft robotics: Academic insights and perspectives through bibliometric analysis. *Soft robotics*, 5(3):229–241, 06 2018. doi: 10.1089/soro.2017.0135. URL `https://www.ncbi.nlm.nih.gov/pubmed/29782219`.

I. Baran, D. Vlasic, E. Grinspun, and J. Popović. Semantic deformation transfer. *ACM Trans. Graph.*, 28(3), 2009.

F. Beck and B. Stumpe. Two devices for operator interaction in the central control of the new cern accelerator. Technical report, CERN, 1973.

L. Bernardi, R. Hopf, D. Sibilio, A. Ferrari, A. Ehret, and E. Mazza. On the cyclic deformation behavior, fracture properties and cytotoxicity of silicone-based elastomers for biomedical applications. *Polymer Testing*, 60:117 – 123, 2017. ISSN 0142-9418. doi: https://doi.org/10.1016/j.polymertesting.2017.03.018. URL `http://www.sciencedirect.com/science/article/pii/S0142941817301538`.

P. D. Block and S. Bergbreiter. Large area all-elastomer capacitive tactile arrays. In *SENSORS, 2013 IEEE*, pages 1–4, Nov 2013. doi: 10.1109/ICSENS.2013.6688345.

F. Bogo, A. Kanazawa, C. Lassner, P. Gehler, J. Romero, and M. J. Black. Keep it smpl: Automatic estimation of 3d human pose and shape from a single image. In *European Conference on Computer Vision*, pages 561–578. Springer, 2016.

M. Botsch and O. Sorkine. On linear variational surface deformation methods. *IEEE Transactions on Visualization and Computer Graphics*, 14(1):213–230, 2008.

C. Bregler and J. Malik. Tracking people with twists and exponential maps. In *Computer Vision and Pattern Recognition, 1998. Proceedings. 1998 IEEE Computer Society Conference on*, pages 8–15. IEEE, 1998.

J. Brunne, S. Kazan, and U. Wallrabe. In-plane deap stack actuators for optical mems applications. volume 7976, pages 7976 – 7976 – 10, 2011. doi: 10.1117/12.880232. URL `https://doi.org/10.1117/12.880232`.

I. Buck, A. Finkelstein, C. Jacobs, A. Klein, D. H. Salesin, J. Seims, R. Szeliski, and K. Toyama. Performance-driven hand-drawn animation. In *Proc. NPAR*, 2000.

Y. Cai, L. Ge, J. Cai, and J. Yuan. Weakly-supervised 3d hand pose estimation from monocular rgb images. In *Proc. ECCV*, pages 1–17. Springer, Cham, 2018.

G. Casiez, N. Roussel, and D. Vogel. Filter: A simple speed-based low-pass filter for noisy input in interactive systems. In *Proc. of the Conf. on Human Factors in Computing Systems*, pages 2527–2530, 2012.

Celsys, Inc. QUMARION, 2013. http://www.clip-studio.com.

K.-Y. Chen, S. N. Patel, and S. Keller. Finexus: Tracking precise motions of multiple fingertips using magnetic sensing. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 1504–1514, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3362-7. doi: 10.1145/2858036.2858125. URL `http://doi.acm.org/10.1145/2858036.2858125`.

X. Chen and A. L. Yuille. Articulated pose estimation by a graphical model with image dependent pairwise relations. In *NIPS*, pages 1736–1744, 2014.

C.-y. Chien, R.-H. Liang, L.-F. Lin, L. Chan, and B.-Y. Chen. Flexibend: Enabling interactivity of multi-part, deformable fabrications using single shape-sensing strip. In *Proc. UIST*, 2015.

J.-B. Chossat, Y. Tao, V. Duchaine, and Y.-L. Park. Wearable soft artificial skin for hand motion detection with embedded microfluidic strain sensing. In *Proc. ICRA*, pages 2568–2573. IEEE, 2015.

T.-S. Chou, A. Gadd, and D. Knott. Hand-eye: A vision-based approach to data glove calibration. In *Proc. Human Interface Technologies*, 2000.

S. Ciotti, E. Battaglia, N. Carbonaro, A. Bicchi, A. Tognetti, and M. Bianchi. A synergy-based optimally designed sensing glove for functional grasp recognition. *Sensors*, 16:811, 06 2016.

J. Connolly, J. Condell, B. O'Flynn, J. T. Sanchez, and P. Gardiner. Imu sensor-based electronic goniometric glove for clinical finger movement analysis. *IEEE Sensors Journal*, 18(3):1273–1281, Feb 2018. ISSN 1530-437X.

L. A. Danisch, K. Englehart, and A. Trivett. Spatially continuous six-degrees-of-freedom position and orientation sensor. volume 3541, pages 48–56, 1999. doi: 10.1117/12.339112. URL `http://dx.doi.org/10.1117/12.339112`.

E. de Aguiar, C. Stoll, C. Theobalt, N. Ahmed, H.-P. Seidel, and S. Thrun. Performance capture from sparse multi-view video. In *ACM SIGGRAPH 2008 Papers*, SIGGRAPH '08, pages 98:1–98:10, New York, NY, USA, 2008. ACM. ISBN 978-1-4503-0112-1. doi: 10.1145/1399504.1360697. URL `http://doi.acm.org/10.1145/1399504.1360697`.

L. Dipietro, A. M. Sabatini, and P. Dario. A survey of glove-based systems and their applications. *Trans. Sys. Man Cyber Part C*, 38(4):461–482, 2008.

M. Dou, S. Khamis, Y. Degtyarev, P. Davidson, S. R. Fanello, A. Kowdle, S. O. Escolano, C. Rhemann, D. Kim, J. Taylor, P. Kohli, V. Tankovich, and S. Izadi. Fusion4d: Real-time performance capture of challenging scenes. *ACM Trans. Graph.*, 35(4):114:1–114:13, July 2016. ISSN 0730-0301. doi: 10.1145/2897824.2925969. URL `http://doi.acm.org/10.1145/2897824.2925969`.

X. Duan, S. Taurand, and M. Soleimani. Artificial skin through super-sensing method and electrical impedance data from conductive fabric with aid of deep learning. *Scientific Reports*, 9(1):8831, 2019. doi: 10.1038/s41598-019-45484-6. URL `https://doi.org/10.1038/s41598-019-45484-6`.

A. Elhayek, E. de Aguiar, A. Jain, J. Thompson, L. Pishchulin, M. Andriluka, C. Bregler, B. Schiele, and C. Theobalt. Marconl-convnet-based marker-less motion capture in outdoor and indoor scenes. *IEEE transactions on pattern analysis and machine intelligence*, 39(3):501–514, 2017.

J. M. Engel, N. Chen, K. S. Ryu, S. D. Pandya, C. Tucker, Y. Yang, and C. Liu. Multi-layer embedment of conductive and non-conductive pdms for all-elastomer mems. 2006.

A. Erol, G. Bebis, M. Nicolescu, R. D. Boyle, and X. Twombly. Vision-based hand pose estimation: A review. *Computer Vision and Image Understanding*, 108(1): 52–73, 2007.

C. Esposito, W. B. Paley, and J. Ong. Of mice and monkeys: a specialized input device for virtual body animation. In *Proc. I3D*, 1995. ISBN 0-89791-736-7. doi: 10.1145/199404.199424. URL `http://doi.acm.org/10.1145/199404.199424`.

B. Fang, F. Sun, H. Liu, and D. Guo. Development of a wearable device for motion capturing based on magnetic and inertial measurement units. *Scientific Programming*, 2017:1–11, 01 2017.

A. Fender, J. Müller, and D. Lindlbauer. Creature teacher: A performance-based animation system for creating cyclic movements. In *Proc. SUI*, 2015.

T.-C. Feng, P. Gunawardane, J. Davis, and B. Jiang. Motion capture data retrieval using an artist's doll. In *Proc. ICPR*, pages 1–4, 2008. ISBN 978-1-4244-2175-6.

M. Fischer, P. van der Smagt, and G. Hirzinger. Learning techniques in a dataglove based telemanipulation system for the dlr hand. In *Proc. ICRA*, volume 2, pages 1603–1608. IEEE, 1998.

V. Ganapathi, C. Plagemann, D. Koller, and S. Thrun. Real-time human pose tracking from range data. In *European conference on computer vision*, pages 738–751. Springer, 2012.

L. Ge, H. Liang, J. Yuan, and D. Thalmann. 3d convolutional neural networks for efficient and robust hand pose estimation from single depth images. In *Proc. CVPR*, 2017.

R. Gentner and J. Classen. Development and evaluation of a low-cost sensor glove for assessment of human finger movements in neurophysiological settings. *Journal of neuroscience methods*, 178:138–47, 12 2008.

O. Glauser, W.-C. Ma, D. Panozzo, A. Jacobson, O. Hilliges, and O. Sorkine-Hornung. Rig Animation with a Tangible and Modular Input Device. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, Jul 2016a.

O. Glauser, B. Vartok, W.-C. Ma, D. Panozzo, A. Jacobson, O. Hilliges, and O. Sorkine-Hornung. Rig animation with a tangible and modular input device. In *UIST '16 Adjunct Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, Oct 2016b.

O. Glauser, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. Deformation capture via self-sensing capacitive arrays (video). In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Oct 2018.

O. Glauser, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. Deformation capture via soft and stretchable sensor arrays. *ACM Transactions on Graphics*, Mar 2019a.

O. Glauser, S. Wu, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. Interactive hand pose estimation using a stretch-sensing soft glove. *ACM Transactions on Graphics (Proceedings of ACM SIGGRAPH)*, Jul 2019b.

O. Glauser, S. Wu, D. Panozzo, O. Hilliges, and O. Sorkine-Hornung. A stretch-sensing soft glove for interactive hand pose estimation. In *ACM SIGGRAPH 2019 Emerging Technologies*, Jul 2019c.

M. Gleicher. Retargetting motion to new characters. In *Proc. SIGGRAPH*, 1998.

A. Glinsky. *Theremin: ether music and espionage*. University of Illinois Press, 2000.

X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Proc. of the Int. Conf. on Artificial Intelligence and Statistics*, pages 249–256, 2010.

D. Gotsch, X. Zhang, J. Burstyn, and R. Vertegaal. Holoflex: A flexible holographic smartphone with bend input. In *Proceedings of the 2016 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, CHI EA '16, pages 3675–3678, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4082-3. doi: 10.1145/2851581.2890258. URL `http://doi.acm.org/10.1145/2851581.2890258`.

W. Griffin, R. Findley, M. Turner, and M. Cutkosky. Calibration and mapping of a human hand for dexterous telemanipulation. In *ASME IMECE Symposium on Haptic Interfaces for Virtual Environments and Teleoperator Systems*, 2000.

T. Grosse-Puppendahl, C. Holz, G. Cohn, R. Wimmer, O. Bechtold, S. Hodges, M. S. Reynolds, and J. R. Smith. Finding common ground: A survey of capacitive sensing in human-computer interaction. In *Proc. CHI*, pages 3293–3315. ACM, 2017. ISBN 978-1-4503-4655-9. doi: 10.1145/3025453.3025808.

M. Guay, M.-P. Cani, and R. Ronfard. The line of action: An intuitive interface for expressive character posing. *ACM Trans. Graph.*, 32(6):205:1–205:8, Nov. 2013. ISSN 0730-0301. doi: 10.1145/2508363.2508397. URL `http://doi.acm.org/10.1145/2508363.2508397`.

M. Guay, R. Ronfard, M. Gleicher, and M.-P. Cani. Space-time sketching of character animation. *ACM Trans. Graph.*, 34(4):118:1–118:10, July 2015. ISSN 0730-0301. doi: 10.1145/2766893. URL `http://doi.acm.org/10.1145/2766893`.

Gurobi Optimization, Inc. Gurobi optimizer reference manual, 2015. http://www.gurobi.com.

F. Hahn, S. Martin, B. Thomaszewski, R. Sumner, S. Coros, and M. Gross. Rig-space physics. *ACM Trans. Graph.*, 31(4), 2012.

F. Hahn, F. Mutzel, S. Coros, B. Thomaszewski, M. Nitti, M. Gross, and R. W. Sumner. Sketch abstractions for character posing. In *Proc. SCA*, 2015.

F. L. Hammond, Y. Mengüç, and R. J. Wood. Toward a modular soft sensor-embedded glove for human hand motion and tactile pressure measurement. In *Proc. IROS*, pages 4000–4007. IEEE, 2014.

J. Han, J. Gu, and G. Lee. Trampoline: A double-sided elastic touch device for creating reliefs. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 383–388, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3069-5. doi: 10.1145/2642918.2647381. URL `http://doi.acm.org/10.1145/2642918.2647381`.

S. Han, B. Liu, R. Wang, Y. Ye, C. D. Twigg, and K. Kin. Online optical marker-based hand tracking with deep labels. *ACM Trans. Graph.*, 37(4):166:1–166:10, July 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proc. CVPR*, June 2016.

R. Held, A. Gupta, B. Curless, and M. Agrawala. 3D puppetry: A kinect-based interface for 3D animation. In *Proc. UIST*, pages 423–434, 2012. ISBN 978-1-4503-1580-7. doi: 10.1145/2380116.2380170. URL `http://doi.acm.org/10.1145/2380116.2380170`.

R. Hinchet, V. Vechev, H. Shea, and O. Hilliges. DextrES: Wearable haptic feedback for grasping in VR via a thin form-factor electrostatic brake. In *Proc. UIST*, 2018.

D. Holden, J. Saito, and T. Komura. Learning an inverse rig mapping for character animation. In *Proc. SCA*, 2015.

R. Hopf, L. Bernardi, J. Menze, M. Zündel, E. Mazza, and A. Ehret. Experimental and theoretical analyses of the age-dependent large-strain behavior of sylgard 184 (10:1) silicone elastomer. *Journal of the Mechanical Behavior of Biomedical Materials*, 60:425 – 437, 2016. ISSN 1751-6161. doi: https://doi.org/10.1016/j.jmbbm.2016.02.022. URL `http://www.sciencedirect.com/science/article/pii/S1751616116000758`.

H. Hu, X. Gao, J. Li, J. Wang, and H. Liu. Calibrating human hand for teleoperating the hit/dlr hand. In *Proc. ICRA*, volume 5, pages 4571–4576 Vol.5, April 2004.

B. Huang, M. Li, T. Mei, D. McCoul, S. Qin, Z. Zhao, and J. Zhao. Wearable stretch sensors for motion measurement of the wrist joint based on dielectric elastomers. *Sensors*, 17(12), 2017. ISSN 1424-8220. doi: 10.3390/s17122708. URL `http://www.mdpi.com/1424-8220/17/12/2708`.

W. Huang. Deformation learning solver, 2015. https://github.com/WebberHuang/DeformationLearningSolver.

Y. Huang, M. Kaufmann, E. Aksan, M. J. Black, O. Hilliges, and G. Pons-Moll. Deep inertial poser: learning to reconstruct human pose from sparse inertial measurements in real time. *ACM Trans. Graph.*, 37(6), 2018.

U. Iqbal, P. Molchanov, T. Breuel, J. Gall, and J. Kautz. Hand pose estimation via latent 2.5 d heatmap regression. *arXiv preprint arXiv:1804.09534*, 2018.

H. Ishii and B. Ullmer. Tangible bits: Towards seamless interfaces between people, bits and atoms. In *Proc. CHI*, 1997.

P. Isola, J.-Y. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *Proc. CVPR*, July 2017.

R. J. K. Jacob, L. E. Sibert, D. C. McFarlane, and M. P. Mullen, Jr. Integrality and separability of input devices. *ACM Trans. Comput.-Hum. Interact.*, 1(1): 3–26, Mar. 1994. ISSN 1073-0516. doi: 10.1145/174630.174631. URL `http://doi.acm.org/10.1145/174630.174631`.

A. Jacobson, I. Baran, L. Kavan, J. Popović, and O. Sorkine. Fast automatic skinning transformations. *ACM Trans. Graph.*, 31(4), 2012.

A. Jacobson, Z. Deng, L. Kavan, and J. Lewis. Skinning: Real-time shape deformation. In *ACM SIGGRAPH 2014 Courses*, 2014a.

A. Jacobson, D. Panozzo, O. Glauser, C. Pradalier, O. Hilliges, and O. Sorkine-Hornung. Tangible and modular input device for character articulation. *ACM Trans. Graph.*, Jul 2014b.

H. Jeong and S. Lim. A stretchable radio-frequency strain sensor using screen printing technology. *Sensors*, 16(11), 2016. ISSN 1424-8220. doi: 10.3390/s16111839. URL `http://www.mdpi.com/1424-8220/16/11/1839`.

H. Jin, S. Jung, J. Kim, S. Heo, J. Lim, W. Park, H. Y. Chu, F. Bien, and K. Park. Stretchable Dual-Capacitor Multi-Sensor for Touch-Curvature-Pressure-Strain Sensing. *Sci Rep*, 7(1):10854, Sep 2017.

M. Jin, D. Gopstein, Y. Gingold, and A. Nealen. AniMesh: Interleaved animation, modeling, and editing. *ACM Trans. Graph.*, 34(6), 2015.

L. K Simone, N. Sundarrajan, X. Luo, Y. Jia, and D. Kamper. A low cost instrumented glove for extended monitoring and functional hand assessment. *Journal of neuroscience methods*, 160:335–48, 04 2007.

W. Kabsch. A solution for the best rotation to relate two sets of vectors. *Acta Crystallographica Section A: Crystal Physics, Diffraction, Theoretical and General Crystallography*, 32(5):922–923, 1976.

F. Kahlesz, G. Zachmann, and R. Klein. 'visual-fidelity'dataglove calibration. In *Computer Graphics International, 2004.*, pages 403–410. IEEE, 2004.

H.-L. C. Kao, C. Holz, A. Roseway, A. Calvo, and C. Schmandt. Duoskin: rapidly prototyping on-skin user interfaces using skin-friendly materials. In *Proceedings of the 2016 ACM International Symposium on Wearable Computers*, pages 16–23. ACM, 2016.

G. D. Kessler, L. F. Hodges, and N. Walker. Evaluation of the cyberglove as a whole-hand input device. *ACM Trans. Comput.-Hum. Interact.*, 2(4):263–283, 1995a.

G. D. Kessler, L. F. Hodges, and N. Walker. Evaluation of the cyberglove as a whole-hand input device. *ACM Transactions on Computer-Human Interaction (TOCHI)*, 2 (4):263–283, 1995b.

D. Kim, O. Hilliges, S. Izadi, A. D. Butler, J. Chen, I. Oikonomidis, and P. Olivier. Digits: Freehand 3d interactions anywhere using a wrist-worn gloveless sensor. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 167–176, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1580-7. URL `http://doi.acm.org/10.1145/2380116.2380139`.

T. Kim, S. J. Yoon, and Y. Park. Soft inflatable sensing modules for safe and interactive robots. *IEEE Robotics and Automation Letters*, 3(4):3216–3223, Oct 2018. ISSN 2377-3766. doi: 10.1109/LRA.2018.2850971.

D. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

B. Knep, C. Hayes, R. Sayre, and T. Williams. Dinosaur input device. In *Proc. CHI*, pages 304–309, 1995.

R. K. Kramer, C. Majidi, R. Sahai, and R. J. Wood. Soft curvature sensors for joint angle proprioception. In *Proc. IROS*, pages 1919–1926. IEEE, 2011.

B. H. Le and Z. Deng. Robust and accurate skeletal rigging from mesh sequences. *ACM Trans. Graph.*, 33(4):84, 2014.

H. Lee, J. Cho, and J. Kim. Printable skin adhesive stretch sensor for measuring multi-axis human joint angles. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4975–4980, May 2016. doi: 10.1109/ICRA.2016.7487705.

S. Lee, W. Buxton, and K. C. Smith. A multi-touch three dimensional touch-sensitive tablet. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '85, pages 21–25, New York, NY, USA, 1985. ACM. ISBN 0-89791-149-0. doi: 10.1145/317456.317461. URL `http://doi.acm.org/10.1145/317456.317461`.

D. Leen, R. Ramakers, and K. Luyten. Strutmodeling: A low-fidelity construction kit to iteratively model, test, and adapt 3d objects. In *Proceedings of the 30th Annual ACM Symposium on User Interface Software and Technology*, UIST '17, pages 471–479, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-4981-9. doi: 10.1145/3126594.3126643. URL `http://doi.acm.org/10.1145/3126594.3126643`.

J. P. Lewis, M. Cordner, and N. Fong. Pose space deformation: A unified approach to shape interpolation and skeleton-driven deformation. In *Proc. SIGGRAPH*, 2000.

B.-S. Lin, I.-J. Lee, S.-Y. Yang, Y.-C. Lo, J. Lee, and J.-L. Chen. Design of an inertial-sensor-based data glove for hand function evaluation. *Sensors*, 18:1545, 05 2018.

D. Lipomi, M. Vosgueritchian, B. Tee, S. L Hellstrom, J. Lee, C. Fox, and Z. Bao. Skin-like pressure and strain sensors based on transparent elastic films of carbon nanotubes. 6:788–92, 10 2011.

H. Liu, X. Wei, J. Chai, I. Ha, and T. Rhee. Realtime human motion control with a small number of inertial sensors. In *Symposium on Interactive 3D Graphics and Games*, pages 133–140. ACM, 2011.

F. Lorussi, W. Rocchia, E. P. Scilingo, A. Tognetti, and D. D. Rossi. Wearable, redundant fabric-based sensor arrays for reconstruction of body segment posture. *IEEE Sensors Journal*, 4(6):807–818, Dec 2004. ISSN 1530-437X. doi: 10.1109/JSEN.2004.837498.

F. Lorussi, E. P. Scilingo, M. Tesconi, A. Tognetti, and D. De Rossi. Strain sensing fabric for hand posture and gesture monitoring. *IEEE transactions on information technology in biomedicine*, 9(3):372–381, 2005.

T. Lu, L. Finkenauer, J. Wissman, and C. Majidi. Rapid prototyping for soft-matter electronics. *Advanced Functional Materials*, 24(22):3351–3356, 2014. doi: 10.1002/adfm.201303732. URL `https://onlinelibrary.wiley.com/doi/abs/10.1002/adfm.201303732`.

Z. Ma and E. Wu. Real-time and robust hand tracking with a single depth camera. *The Visual Computer*, 30(10):1133–1144, 2014.

M. R. Masliah and P. Milgram. Measuring the allocation of control in a 6 degree-of-freedom docking experiment. In *Proc. CHI*, 2000. ISBN 1-58113-216-6. doi: 10.1145/332040.332403. URL `http://doi.acm.org/10.1145/332040.332403`.

N. Matsuhisa, X. Chen, Z. Bao, and T. Someya. Materials and structural designs of stretchable conductors. *Chem. Soc. Rev.*, 48:2946–2966, 2019. doi: 10.1039/C8CS00814K. URL `http://dx.doi.org/10.1039/C8CS00814K`.

C. Mattmann, F. Clemens, and G. Tröster. Sensor for measuring strain in textile. *Sensors*, 8(6):3719–3732, 2008. ISSN 1424-8220. doi: 10.3390/s8063719. URL `http://www.mdpi.com/1424-8220/8/6/3719`.

Maya. Autodesk, `http://www.autodesk.com/maya`, 2014.

N. Mayer, E. Ilg, P. Fischer, C. Hazirbas, D. Cremers, A. Dosovitskiy, and T. Brox. What makes good synthetic training data for learning disparity and optical flow estimation? *International Journal of Computer Vision*, 126(9):942–960, Sep

2018. ISSN 1573-1405. doi: 10.1007/s11263-018-1082-6. URL https://doi.org/10.1007/s11263-018-1082-6.

D. Mehta, S. Sridhar, O. Sotnychenko, H. Rhodin, M. Shafiei, H.-P. Seidel, W. Xu, D. Casas, and C. Theobalt. Vnect: Real-time 3d human pose estimation with a single rgb camera. volume 36, July 2017. doi: 10.1145/3072959.3073596. URL http://gvv.mpi-inf.mpg.de/projects/VNect/.

A. Menon, B. Barnes, R. Mills, C. Bruyns, X. Twombly, J. Smith, K. Montgomery, and R. D. Boyle. Using registration, calibration, and robotics to build a more accurate virtual reality simulation for astronaut training and telemedicine. In *WSCG*, 2003.

H. O. Michaud, L. Dejace, S. De Mulatier, and S. P. Lacour. Design and functional evaluation of an epidermal strain sensing system for hand tracking. In *Proc. IROS*, pages 3186–3191, 2016.

P. Mittendorfer and G. Cheng. Humanoid multi-modal tactile sensing modules. In *IEEE TRO - Special Issue on Robotic Sense of Touch*, pages 401–410, Jun 2011. URL ;.

T. B. Moeslund, A. Hilton, and V. Krüger. A survey of advances in vision-based human motion capture and analysis. *Computer vision and image understanding*, 104(2):90–126, 2006.

F. Mueller, F. Bernard, O. Sotnychenko, D. Mehta, S. Sridhar, D. Casas, and C. Theobalt. Ganerated hands for real-time 3d hand tracking from monocular rgb. In *Proc. CVPR*, pages 49–59, 2018.

K. Nakagaki, S. Follmer, and H. Ishii. Lineform: Actuated curve interfaces for display, interaction, and constraint. In *Proc. UIST*, 2015.

T. Neumann, K. Varanasi, N. Hasler, M. Wacker, M. Magnor, and C. Theobalt. Capture and statistical modeling of arm-muscle deformations. *Computer Graphics Forum*, 32(2pt3):285–294, 2013. doi: 10.1111/cgf.12048. URL https://onlinelibrary.wiley.com/doi/abs/10.1111/cgf.12048.

R. A. Newcombe, D. Fox, and S. M. Seitz. Dynamicfusion: Reconstruction and tracking of non-rigid scenes in real-time. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 343–352, 2015.

A. Newell, K. Yang, and J. Deng. Stacked hourglass networks for human pose estimation. In *ECCV*, pages 483–499, 2016.

A. S. Nittala, A. Withana, N. Pourjafarian, and J. Steimle. Multi-touch skin: A thin and flexible multi-touch sensor for on-skin input. In *Proceedings of the 2018 CHI Conference on Human Factors in Computing Systems*, CHI '18, pages 33:1–33:12, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5620-6. doi: 10.1145/3173574.3173607. URL `http://doi.acm.org/10.1145/3173574.3173607`.

M. Oberweger and V. Lepetit. Deepprior++: Improving fast and accurate 3d hand pose estimation. In *International Conference on Computer Vision Workshops*, 2017.

M. Oberweger, P. Wohlhart, and V. Lepetit. Hands deep in deep learning for hand pose estimation. *arXiv preprint arXiv:1502.06807*, 2015a.

M. Oberweger, P. Wohlhart, and V. Lepetit. Training a feedback loop for hand pose estimation. In *Proc. ICCV*, pages 3316–3324, 2015b.

B. O'Brien, T. Gisby, and I. A. Anderson. Stretch sensors for human body motion. In *Proc. SPIE*, volume 9056, page 905618, 2014.

T. F. O'Connor, M. E. Fach, R. Miller, S. E. Root, P. P. Mercier, and D. J. Lipomi. The language of glove: Wireless gesture decoder with low-power and stretchable hybrid electronics. *PloS one*, 12(7):e0179766, 2017.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Efficient model-based 3D tracking of hand articulations using Kinect. In *Proc. BMVC*, 2011a.

I. Oikonomidis, N. Kyriazis, and A. A. Argyros. Full DOF tracking of a hand interacting with an object by modeling occlusions and physical constraints. In *Proc. ICCV*, 2011b.

A. C. Öztireli, I. Baran, T. Popa, B. Dalstein, R. W. Sumner, and M. Gross. Differential blending for expressive sketch-based posing. In *Proc. SCA*, 2013.

W. Park, K. Ro, S. Kim, and J. Bae. A soft sensor-based three-dimensional (3-d) finger motion measurement system. *Sensors (Basel, Switzerland)*, 17(2):420, 02 2017.

A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in pytorch. 2017.

P. Polygerinos, Z. Wang, K. C. Galloway, R. J. Wood, and C. J. Walsh. Soft robotic glove for combined assistance and at-home rehabilitation. *Robotics and Autonomous Systems*, 73:135 – 143, 2015. ISSN 0921-8890.

R. D. Ponce Wong, J. Posner, and V. Santos. Flexible microfluidic normal force sensor skin for tactile feedback. 179:62–69, 06 2012.

G. Pons-Moll, J. Romero, N. Mahmood, and M. J. Black. Dyna: A model of dynamic human shape in motion. *ACM Trans. Graph.*, 34(4):120:1–120:14, July 2015. ISSN 0730-0301. doi: 10.1145/2766993. URL `http://doi.acm.org/10.1145/2766993`.

I. Poupyrev, N.-W. Gong, S. Fukuhara, M. E. Karagozler, C. Schwesig, and K. E. Robinson. Project jacquard: Interactive digital textiles at scale. In *Proceedings of the 2016 CHI Conference on Human Factors in Computing Systems*, CHI '16, pages 4216–4227, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-3362-7. doi: 10.1145/2858036.2858176. URL `http://doi.acm.org/10.1145/2858036.2858176`.

A. Rashid and O. Hasan. Wearable technologies for hand joints monitoring for rehabilitation: A survey. *Microelectronics Journal*, 02 2018.

J. Rekimoto. Smartskin: An infrastructure for freehand manipulation on interactive surfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '02, pages 113–120, New York, NY, USA, 2002. ACM. ISBN 1-58113-453-3. doi: 10.1145/503376.503397. URL `http://doi.acm.org/10.1145/503376.503397`.

C. Rendl, P. Greindl, M. Haller, M. Zirkl, B. Stadlober, and P. Hartmann. Pyzoflex: Printed piezoelectric pressure sensing foil. In *Proceedings of the 25th Annual ACM Symposium on User Interface Software and Technology*, UIST '12, pages 509–518, New York, NY, USA, 2012. ACM. ISBN 978-1-4503-1580-7. doi: 10.1145/2380116.2380180. URL `http://doi.acm.org/10.1145/2380116.2380180`.

C. Rendl, D. Kim, S. Fanello, P. Parzer, C. Rhemann, J. Taylor, M. Zirkl, G. Scheipl, T. Rothländer, M. Haller, and S. Izadi. Flexsense: A transparent self-sensing deformable surface. In *Proceedings of the 27th Annual ACM Symposium on User Interface Software and Technology*, UIST '14, pages 129–138, New York, NY, USA, 2014. ACM. ISBN 978-1-4503-3069-5. doi: 10.1145/2642918.2647405. URL `http://doi.acm.org/10.1145/2642918.2647405`.

H. Rhodin, J. Tompkin, K. I. Kim, V. Kiran, H.-P. Seidel, and C. Theobalt. Interactive motion mapping for real-time character control. *Comput. Graph. Forum*, 33(2), 2014.

H. Rhodin, N. Robertini, C. Richardt, H.-P. Seidel, and C. Theobalt. A versatile scene model with differentiable visibility applied to generative pose estimation. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 765–773, 2015.

D. Roetenberg, H. Luinge, and P. Slycke. Moven: Full 6dof human motion tracking using miniature inertial sensors. *Xsen Technologies, December*, 2(3):8, 2007.

J. Romero, D. Tzionas, and M. J. Black. Embodied hands: Modeling and capturing hands and bodies together. *ACM Trans. Graph.*, 36(6), 2017.

O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In *Inte. Conf. on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

S. Rosset and H. R. Shea. Flexible and stretchable electrodes for dielectric elastomer actuators. *Applied Physics A*, 110(2):281–307, Feb 2013. ISSN 1432-0630. doi: 10.1007/s00339-012-7402-8. URL `https://doi.org/10.1007/s00339-012-7402-8`.

S. Rosset, O. A. Araromi, S. Schlatter, and H. R. Shea. Fabrication Process of Silicone-based Dielectric Elastomer Actuators. *J Vis Exp*, (108):e53423, Feb 2016.

H. Ryu, S. Park, J.-J. Park, and J. Bae. A knitted glove sensing system with compression strain for finger movements. *Smart Materials and Structures*, 27(5):055016, 2018.

T. S. Saponas, D. S. Tan, D. Morris, R. Balakrishnan, J. Turner, and J. A. Landay. Enabling always-available input with muscle-computer interfaces. In *Proceedings of the 22Nd Annual ACM Symposium on User Interface Software and Technology*, UIST '09, pages 167–176, New York, NY, USA, 2009. ACM. ISBN 978-1-60558-745-5. doi: 10.1145/1622176.1622208. URL `http://doi.acm.org/10.1145/1622176.1622208`.

M. S. Sarwar, Y. Dobashi, C. Preston, J. K. M. Wyss, S. Mirabbasi, and J. D. W. Madden. Bend, stretch, and touch: Locating a finger on an actively deformed transparent sensor array. *Science Advances*, 3(3), 2017. doi: 10.1126/sciadv.1602200. URL `http://advances.sciencemag.org/content/3/3/e1602200`.

Y. Savoye and A. Meyer. Multi-layer level of detail for character animation. In *Workshop in Virtual Reality Interactions and Physical Simulation "VRIPHYS" (2008)*, 2008.

S. Schaefer and C. Yuksel. Example-based skeleton extraction. In *Proc. SGP*, 2007.

L. Schwarz, D. Mateus, and N. Navab. Discriminative human full-body pose estimation from wearable inertial sensor data. *Modelling the Physiological Human*, pages 159–172, 2009.

C. Schwesig, I. Poupyrev, and E. Mori. Gummi: A bendable computer. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '04, pages 263–270, New York, NY, USA, 2004. ACM. ISBN 1-58113-702-8. doi: 10.1145/985692.985726. URL `http://doi.acm.org/10.1145/985692.985726`.

T. Sharp, C. Keskin, D. Robertson, J. Taylor, J. Shotton, D. Kim, C. Rhemann, I. Leichter, A. Vinnikov, Y. Wei, et al. Accurate, robust, and flexible real-time hand tracking. In *Proc. of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, pages 3633–3642, 2015.

Z. Shen, J. Yi, X. Li, M. H. P. Lo, M. Z. Chen, Y. Hu, and Z. Wang. A soft stretchable bending sensor and data glove applications. *IEEE Int. Conf. on Real-time Computing and Robotics (RCAR)*, 3(1):22, 2016.

J. R. Shewchuk. Triangle: Engineering a 2D quality mesh generator and Delaunay triangulator. In *Applied Computational Geometry: Towards Geometric Engineering*, volume 1148 of *Lecture Notes in Computer Science*, pages 203–222. 1996.

T. Shiratori, M. Mahler, W. Trezevant, and J. Hodgins. Expressing animated performances through puppeteering. In *Proc. 3DUI*, 2013.

J. Shotton, T. Sharp, A. Kipman, A. Fitzgibbon, M. Finocchio, A. Blake, M. Cook, and R. Moore. Real-time human pose recognition in parts from single depth images. *Communications of the ACM*, 56(1):116–124, 2013.

T.-W. Shyr, J.-W. Shie, C.-H. Jiang, and J.-J. Li. A textile-based wearable sensing device designed for monitoring the flexion angle of elbow and knee movements. *Sensors*, 14(3):4050–4059, 2014.

T. Simon, H. Joo, I. Matthews, and Y. Sheikh. Hand keypoint detection in single images using multiview bootstrapping. In *Proc. CVPR*, volume 2, 2017.

A. Sinha, C. Choi, and K. Ramani. Deephand: Robust hand pose estimation by completing a matrix imputed with deep features. In *Proc. CVPR*, 2016.

J. R. Smith. *Toward electric field tomography*. PhD thesis, Massachusetts Institute of Technology, 1995.

A. Spurr, J. Song, S. Park, and O. Hilliges. Cross-modal deep variational hand pose estimation. In *Proc. CVPR*, 2018.

S. Sridhar, A. Oulasvirta, and C. Theobalt. Interactive markerless articulated hand motion tracking using rgb and depth data. In *Proc. ICCV*, Dec. 2013.

J. Starck and A. Hilton. Model-based multiple view reconstruction of people. In *null*, page 915. IEEE, 2003.

J. Steffen, J. Maycock, H. Ritter, H. Liu, and D. Schilberg. Robust dataglove mapping for recording human hand postures. In *Intelligent Robotics and Applications*, 2011.

C. Stoll, N. Hasler, J. Gall, H.-P. Seidel, and C. Theobalt. Fast articulated motion tracking using a sums of gaussians body model. In *Computer Vision (ICCV), 2011 IEEE International Conference on*, pages 951–958. IEEE, 2011.

R. Sumner and J. Popović. Deformation transfer for triangle meshes. *ACM Trans. Graph.*, 23(3):399–405, 2004.

R. W. Sumner, M. Zwicker, C. Gotsman, and J. Popović. Mesh-based inverse kinematics. *ACM Trans. Graph.*, 24(3):488–495, 2005.

X. Sun, Y. Wei, S. Liang, X. Tang, and J. Sun. Cascaded hand pose regression. In *Proc. CVPR*, pages 824–832, 2015.

S. Sundaram, P. Kellnhofer, Y. Li, J.-Y. Zhu, A. Torralba, and W. Matusik. Learning the signatures of the human grasp using a scalable tactile glove. *Nature*, 569 (7758):698–702, 2019. doi: 10.1038/s41586-019-1234-z. URL `https://doi.org/10.1038/s41586-019-1234-z`.

A. Tagliasacchi, I. Alhashim, M. Olson, and H. Zhang. Mean curvature skeletons. *Comput. Graph. Forum*, 31(5), 2012.

A. Tagliasacchi, M. Schroeder, A. Tkach, S. Bouaziz, M. Botsch, and M. Pauly. Robust articulated-icp for real-time hand tracking. In *Proc. SGP*, 2015.

D. Tang, T.-H. Yu, and T.-K. Kim. Real-time articulated hand pose estimation using semi-supervised transductive regression forests. In *Proc. ICCV*, pages 3224–3231, 2013.

D. Tang, H. Jin Chang, A. Tejani, and T.-K. Kim. Latent regression forest: Structured estimation of 3d articulated hand posture. In *Proc. CVPR*, June 2014.

D. Tang, J. Taylor, P. Kohli, C. Keskin, T.-K. Kim, and J. Shotton. Opening the black box: Hierarchical sampling optimization for estimating human hand pose. In *Proc. ICCV*, pages 3325–3333, 2015.

J. Tautges, A. Zinke, B. Krüger, J. Baumann, A. Weber, T. Helten, M. Müller, H.-P. Seidel, and B. Eberhardt. Motion reconstruction using sparse accelerometer data. *ACM Transactions on Graphics (TOG)*, 30(3):18, 2011.

J. Taylor, J. Shotton, T. Sharp, and A. Fitzgibbon. The vitruvian manifold: Inferring dense correspondences for one-shot human pose estimation. In *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*, pages 103–110. IEEE, 2012.

J. Taylor, L. Bordeaux, T. Cashman, B. Corish, C. Keskin, T. Sharp, E. Soto, D. Sweeney, J. Valentin, B. Luff, A. Topalian, E. Wood, S. Khamis, P. Kohli, S. Izadi, R. Banks, A. Fitzgibbon, and J. Shotton. Efficient and precise interactive hand tracking through joint, continuous optimization of pose and correspondences. *ACM Trans. Graph.*, 35(4):143:1–143:12, 2016.

J. Taylor, V. Tankovich, D. Tang, C. Keskin, D. Kim, P. Davidson, A. Kowdle, and S. Izadi. Articulated distance fields for ultra-fast tracking of hands interacting. *ACM Trans. Graph.*, 36:244:1–244:12, 2017.

B. Tekin, P. Márquez-Neila, M. Salzmann, and P. Fua. Fusing 2d uncertainty and 3d cues for monocular body pose estimation. *arXiv preprint arXiv:1611.05708*, 2016.

A. Tkach, M. Pauly, and A. Tagliasacchi. Sphere-meshes for real-time hand modeling and tracking. *ACM Trans. Graph.*, 35(6):222:1–222:11, 2016.

A. Tkach, A. Tagliasacchi, E. Remelli, M. Pauly, and A. Fitzgibbon. Online generative model personalization for hand tracking. *ACM Trans. Graph.*, 36(6), 2017.

J. Tompson, M. Stein, Y. Lecun, and K. Perlin. Real-time continuous pose recovery of human hands using convolutional networks. *ACM Trans. Graph.*, 33(5):169, 2014a.

J. J. Tompson, A. Jain, Y. LeCun, and C. Bregler. Joint training of a convolutional network and a graphical model for human pose estimation. In *NIPS*, pages 1799–1807, 2014b.

A. Toshev and C. Szegedy. Deeppose: Human pose estimation via deep neural networks. In *CVPR*, pages 1653–1660, 2014.

H. Truong, S. Zhang, U. Muncuk, P. Nguyen, N. Bui, A. Nguyen, Q. Lv, K. Chowdhury, T. Dinh, and T. Vu. Capband: Battery-free successive capacitance sensing wristband for hand gesture recognition. In *Proceedings of the 16th ACM Conference on Embedded Networked Sensor Systems*, SenSys '18, pages 54–67, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5952-8. URL `http://doi.acm.org/10.1145/3274783.3274854`.

T. von Marcard, B. Rosenhahn, M. J. Black, and G. Pons-Moll. Sparse inertial poser: Automatic 3d human pose estimation from sparse imus. volume 36, pages 349–360, 2017.

C. Wan, A. Yao, and L. Van Gool. Hand pose estimation from local surface normals. In *European Conference on Computer Vision*, pages 554–569. Springer, 2016.

C. Wan, T. Probst, L. Van Gool, and A. Yao. Crossing nets: Combining gans and vaes with a shared latent space for hand pose estimation. In *Proc. CVPR*, July 2017.

H. Wang, M. Totaro, and L. Beccai. Toward perceptive soft robots: Progress and challenges. *Advanced science (Weinheim, Baden-Wurttemberg, Germany)*, 5 (9):1800541–1800541, 07 2018. doi: 10.1002/advs.201800541. URL https://www.ncbi.nlm.nih.gov/pubmed/30250796.

R. Y. Wang and J. Popović. Real-time hand-tracking with a color glove. *ACM Trans. Graph.*, 28(3):63:1–63:8, 2009.

Y. Wang and M. Neff. Data-driven glove calibration for hand motion capture. In *Proc. SCA*, pages 15–24, 2013.

Y. Wang, A. Jacobson, J. Barbič, and L. Kavan. Linear subspace design for real-time shape deformation. *ACM Trans. Graph.*, 34(4):57:1–57:11, July 2015. ISSN 0730-0301. doi: 10.1145/2766952. URL http://doi.acm.org/10.1145/2766952.

S.-E. Wei, V. Ramakrishna, T. Kanade, and Y. Sheikh. Convolutional pose machines. In *CVPR*, pages 4724–4732, 2016.

M. Weigel, T. Lu, G. Bailly, A. Oulasvirta, C. Majidi, and J. Steimle. iskin: Flexible, stretchable and visually customizable on-body touch sensors for mobile computing. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2991–3000, New York, NY, USA, 2015. ACM. ISBN 978-1-4503-3145-6. doi: 10.1145/2702123.2702391. URL http://doi.acm.org/10.1145/2702123.2702391.

M. Wessely, T. Tsandilas, and W. E. Mackay. Stretchis: Fabricating highly stretchable user interfaces. In *Proceedings of the 29th Annual Symposium on User Interface Software and Technology*, UIST '16, pages 697–704, New York, NY, USA, 2016. ACM. ISBN 978-1-4503-4189-9. doi: 10.1145/2984511.2984521. URL http://doi.acm.org/10.1145/2984511.2984521.

J. Wissman, T. Lu, and C. Majidi. Soft-matter electronics with stencil lithography. *2013 IEEE SENSORS*, pages 1–4, 2013.

K. Wolff, R. Poranne, O. Glauser, and O. Sorkine-Hornung. Packable springs. *Computer Graphics Forum*, May 2018.

S.-J. Woo, J.-H. Kong, D.-G. Kim, and J.-M. Kim. A thin all-elastomeric capacitive pressure sensor array based on micro-contact printed elastic conductors. *J. Mater. Chem. C*, 2:4415–4422, 2014. doi: 10.1039/C4TC00392F. URL http://dx.doi.org/10.1039/C4TC00392F.

D. Xu, A. Tairych, and I. A. Anderson. Stretch not flex: programmable rubber keyboard. *Smart Materials and Structures*, 25(1):015012, 2016. URL `http://stacks.iop.org/0964-1726/25/i=1/a=015012`.

W. Yoshizaki, Y. Sugiura, A. C. Chiou, S. Hashimoto, M. Inami, T. Igarashi, Y. Akazawa, K. Kawachi, S. Kagami, and M. Mochimaru. An actuated physical puppet as an input device for controlling a digital manikin. In *Proc. CHI*, pages 637–646, 2011.

S. Zhai and P. Milgram. Quantifying coordination in multiple dof movement and its application to evaluating 6 DOF input devices. In *Proc. CHI*, pages 320–327, 1998. ISBN 0-201-30987-4. doi: 10.1145/274644.274689. URL `http://dx.doi.org/10.1145/274644.274689`.

J. Zhang, J. Jiao, M. Chen, L. Qu, X. Xu, and Q. Yang. 3D hand pose tracking and estimation using stereo matching. *arXiv preprint arXiv:1610.07214*, 2016.

Y. Zheng, Y. Peng, G. Wang, X. Liu, X. Dong, and J. Wang. Development and evaluation of a sensor glove for hand function assessment and preliminary attempts at assessing hand coordination. *Measurement*, 93, 06 2016.

J. Zhou, F. Malric, and S. Shirmohammadi. A new hand-measurement method to simplify calibration in CyberGlove-based virtual rehabilitation. *IEEE Transactions on Instrumentation and Measurement*, 59(10):2496–2504, Oct 2010. ISSN 0018-9456.

X. Zhou, M. Zhu, S. Leonardos, K. G. Derpanis, and K. Daniilidis. Sparseness meets deepness: 3d human pose estimation from monocular video. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4966–4975, 2016.

T. G. Zimmerman, J. R. Smith, J. A. Paradiso, D. Allport, and N. Gershenfeld. Applying electric field sensing to human-computer interfaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '95, pages 280–287, New York, NY, USA, 1995. ACM Press/Addison-Wesley Publishing Co. ISBN 0-201-84705-1. doi: 10.1145/223904.223940. URL `http://dx.doi.org/10.1145/223904.223940`.

C. Zimmermann and T. Brox. Learning to estimate 3d hand pose from single rgb images. In *Proc. ICCV*, pages 4913–4921, 2017.

M. Zollhöfer, M. Nießner, S. Izadi, C. Rehmann, C. Zach, M. Fisher, C. Wu, A. Fitzgibbon, C. Loop, C. Theobalt, et al. Real-time non-rigid reconstruction using an rgb-d camera. *ACM Transactions on Graphics (TOG)*, 33(4):156, 2014.