Self-Contained Kinematic Calibration of a Novel Whole-Body Artificial Skin for Human-Robot Collaboration

Kandai Watanabe[1], Matthew Strong[1], Mary West[1],
Caleb Escobedo[1], Ander Aramburu[1], Krishna Chaitanya Kodur[2], Alessandro Roncone[1]

Abstract— In this paper, we present an accelerometer-based kinematic calibration algorithm to accurately estimate the pose of multiple sensor units distributed along a robot body. Our approach is self-contained, can be used on any robot provided with a Denavit-Hartenberg kinematic model, and on any skin equipped with Inertial Measurement Units (IMUs). To validate the proposed method, we first conduct extensive experimentation in simulation and demonstrate a sub-cm positional error from ground truth data—an improvement of six times with respect to prior work; subsequently, we then perform a real-world evaluation on a seven degrees-of-freedom collaborative platform. For this purpose, we additionally introduce a novel design for a stand-alone artificial skin equipped with an IMU for use with the proposed algorithm and a proximity sensor for sensing distance to nearby objects. In conclusion, in this work, we demonstrate seamless integration between a novel hardware design, an accurate calibration method, and preliminary work on applications: the high positional accuracy effectively enables to locate distributed proximity data and allows for a distributed avoidance controller to safely avoid obstacles and people without the need of additional sensing.

I. INTRODUCTION

In recent years, robots have started to leave the structured environments characteristic of factories and laboratories, and they have progressively transitioned to operating with and around people. However, modern robotic manipulators are limited in their ability to operate in close proximity with humans because they currently lack the necessary sensing capabilities for whole-body perception of humans and obstacles in their immediate surroundings. Traditionally, collaborative robots designed to work with people (such as the Franka Panda robot in Fig. 1) resort to either external, sparse, and high-resolution sensing (e.g. RGB-D cameras [1] which have low bandwidth and are prone to occlusions) or on-board, low-resolution, contact-based perception (through e.g. wrist-mounted force/torque sensors, torque sensors on the robot's joints [2], or a combination of the two); the rest of the body is rarely taken into account. Under this perspective, distributed *artificial skins* are a promising solution to effectively enable whole-body awareness of contact and information-rich perception of the nearby space, with the goal of robustly perceiving humans and safely operating in close proximity with them [3]–[9]. However, existing artificial skin technologies are limited by the following: i) they often demand a considerable amount of time to design,

[1]Human Interaction and RObotics (HIRO) Group, Computer Science Department, University of Colorado, Boulder, CO 80309, USA `name.surname@colorado.edu`.
[2]Heracleia Human-Centered Computing Lab, University of Texas at Arlington, Arlington, TX, 76019 USA.
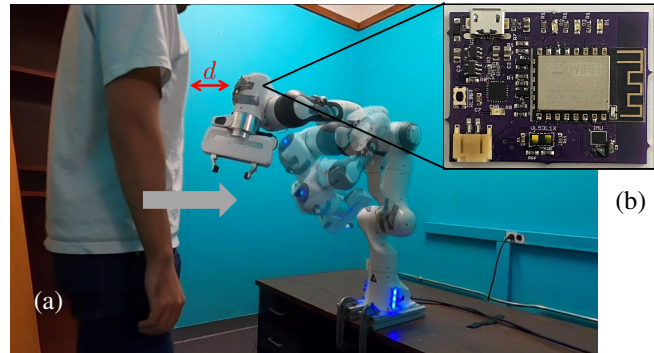
Fig. 1: We present an algorithm that autonomously calibrates the poses of multiple units of a novel artificial skin prototype mounted on a robot; such poses can then be used to precisely locate sensor data in the robot's frame of reference. a) example of trajectory redirecting for human-aware robot safety: as a person approaches the robot, proximity sensors are activated and the robot successfully avoids the human; b) close-up of the self-contained skin prototype used for validation of the algorithm.

set up, and deploy the hardware; ii) they are usually tailored to a specific platform and cannot easily be ported to different robots; iii) they generally focus on touch/pressure sensing, and they consequently do not enable perception in the nearby space of a robot; iv) they require manual, precise, and time-consuming installation and calibration. Collectively, these limitations demand a significant overhead, with the net result of limiting research and progress in the field.

In this work, we aim at mitigating these drawbacks by presenting a comprehensive framework for perception, calibration, and robot operation in close proximity with humans. First, we present an accurate, automated, and self-contained algorithm for *kinematic calibration* [10], which contributes to addressing issue iv). For the purposes of this paper, we define kinematic calibration as the problem of locating the six-dimensional pose (i.e. position and orientation) of multiple skin sensors distributed along a robot's body. Our method does not rely on the use of external metrology systems (such as the laser pointers detailed in [11], [12]), and it exclusively resorts to accelerometer data as read from Inertial Measurement Unit (IMU) sensors conveniently located on the robot skin. Secondly, we introduce a novel design for a wireless artificial skin that is self-contained, self-powered and capable of proximity and inertial sensing (cf. Fig. 1).

As detailed in Section III, our open-source design is robot-independent, it can be readily utilized on a variety of different applications, and it focuses on proximity sensing as the most fundamental feature for a robot to operate around people—thus contributing to issues i), ii), and iii).

The theoretical contribution of the algorithm detailed in Section IV consists of two main components, which collectively result in improved convergence and greater accuracy if compared to prior work. First, we introduce a novel formulation of the optimization problem which is now divided in two distinct steps—orientation estimation and subsequent position estimation. Second, acceleration-based calibration of the skin is simplified without loss of accuracy to further accelerate convergence. The proposed method is initially evaluated in simulation and subsequently validated on a real robot platform (see Section V); collectively, our results demonstrate that our contribution significantly outperforms existing state-of-the-art approaches in terms of convergence accuracy, thus bringing us one step closer to a future of plug-and-play and multi-functional sensing. Finally, the effectiveness of our holistic approach is demonstrated with a minimal obstacle avoidance controller (similarly to [1], [4], [8], [13]–[17]) in which calibrated proximity readings are used to safely operate around people (Section VI). Our prototype interaction effectively demonstrates how on-board, distributed, high-bandwidth proximity sensing constitutes a promising direction for future work in close-proximity Human-Robot Collaboration.

II. BACKGROUND AND RELATED WORK

In the following, we briefly detail prior research relevant to the purposes of this work, and we clarify how our work is positioned with respect to prior efforts.

A. Kinematic calibration: Calibration of a robot's kinematic chain is a core problem for industrial and collaborative robotics, as precise modeling of a robot's kinematics and dynamics has direct impact on performance and operation of a robot manipulator. For this reason, kinematic calibration has been extensively investigated in past decades, and a number of techniques have been devised for both *open-loop* and *closed-loop* approaches (please refer to [18] for an overview). However, these works are different in nature from the proposed method in that they are improving an already available kinematic model—which is a significant prior that facilitates convergence of the optimization and accuracy of the results. In our work, we are concerned with kinematically calibrating artificial skin sensors that are randomly placed on a robot's surface, i.e. without any prior information to rely upon. Of particular relevance to this paper is the work performed in the area of calibration of robot kinematics based on IMU data [19], which overlaps with the work in human body estimation for motion capture, e.g. [20], [21]. However, in addition to dealing with a different problem (calibration of an existing kinematic model vs calibration of a free-moving skin unit), existing approaches typically rely on sensor fusion techniques that integrate information coming from a variety of different sensors (e.g. magnetometers, GPS,

ultra-wideband radio, cameras, and more). In our work, we exclusively employ accelerometer data as we are constrained by the compact hardware prototype we have introduced.

B. Artificial skins for robotics: In recent years, there has been steady and significant progress on developing technologies for distributed artificial sensing in robotics. While the majority of prior work focuses on pressure and touch sensing for robotic grippers to enhance manipulation and grasping (e.g. [16], [17], [22]), a parallel line of research has aimed attention at whole-body, dense coverage of a robot's surface [5]–[7], [23]. Our paper capitalizes on this body of work, and improves it in two directions: i) by focusing on prior-to-contact perception of the nearby space of a robot rather than on-contact, touch-based perception (see also [23], [24]), which is an under-explored area of interest; ii) by developing a plug-and-play, self-contained hardware element that can be replicated at scale and utilized to cover a variety of different robots with minimal overhead. Our work is based on the belief that artificial skins should not be limited to few sensing capabilities but rather house a variety of features. To our knowledge, this is the first attempt at providing such degree of modularity and flexibility.

C. Kinematic calibration of artificial skins: To realize safe, robust and reliable obstacle detection, it is of paramount importance to precisely locate the relative pose of each sensor with respect to the kinematic chain of the robot. As mentioned in Section II.A, this process is not to be confused with calibration of a robot's kinematic model, as the sensors to be calibrated can be placed anywhere on the robot body. Traditionally, kinematic calibration of artificial skin is a procedure that is manually performed by the robot operator, and only recent work has started to automate it. [25], [26] presented a method where the iCub humanoid robot [7] performed "self-touch" actions on its robotic skin, allowing for an observation of the 3D position of the end-effector and computation of its Denavit-Hartenberg [DH] parameters [27]. While this approach is autonomous and does not rely on external measurements, the iCub's bi-manual self-touch capabilities would not be possible on commercially-available collaborative manipulators due a lower number of Degrees of Freedom (DoFs) and consequently reduced manipulability. Of more generalizability is the work detailed in [28], where tactile sensors and kinematic information of the robot arm were estimated using acceleration data collected from a set of IMUs mounted within the skin. However, while this work serves as reference for the method detailed in this paper, its average real-world positional error renders it ineffective in practice—as detailed in Section VI.

III. NOVEL DESIGN OF ARTIFICIAL SKIN

In the following, we introduce a modular design for a whole-body artificial skin with dynamic sensing capabilities. We define a *skin unit* (SU) as the most minimal, atomic skin element needed to perform autonomous kinematic calibration and nearby-space perception; for the purposes of the algorithm detailed in Section IV, the SU shown in Fig. 1b is composed of an inertial measurement unit (IMU) and a

proximity sensor; contact sensing can be achieved through a combination of accelerometer and gyroscope data. The proposed skin is characterized by off-the-shelf sensors that enable low overall cost (approximately $36) and low power consumption (with an operational range of $110 - 160$mA), which effectively allows for a self-contained package to ease prototyping and deployment. The SUs used in this work (see Fig. 5 for reference) operate with external $100mAh$ batteries and can operate continuously for up to 10 hours.
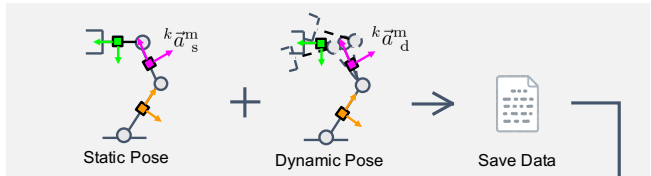
Sensor Selection and Electronics Considerations: The 33×36mm SU prototype detailed in Fig. 1b is a stand-alone design equipped with a WiFi-enabled ESP8266 micro-controller, a USB-to-UART bridge for programming, visual-based hardware debugging, and several system status LEDs. The SU is a 2-layer stack-up rigid PCB fabricated on 0.15mm Kingboard copper clad laminate $175TgFR4$ substrate, and it is powered by means of an integrated Lithium-Polymer battery (for an operation time of 7 to 10 hours on a single charge). We selected an LSM6DS3 iNEMO inertial measurement unit, which is characterized by a full-scale acceleration range of $\pm 2/\pm 4/\pm 8/\pm 16g$, an angular rate range of $\pm 125/\pm 245/\pm 500/\pm 1000/\pm 2000$ degrees per second, and robustness against mechanical shock. Proximity sensing is provided by a VL53L1X time-of-flight (ToF) sensor, which was chosen for its accuracy, ranging distance (up to 4m), field of view ($27°$), ranging frequency of 50 Hz, I2C communication protocol, and small package. Both sensors were also chosen for their high bandwidth: with six SUs mounted on the robot (Fig. 1) and sending data wirelessly, we were able to receive data at a rate of 100Hz from the IMUs and 50Hz from the distance sensor (although the sensor is capable of a nominal bandwidth of 100Hz). All files related to the design can be found at the associated GitHub repository[1].
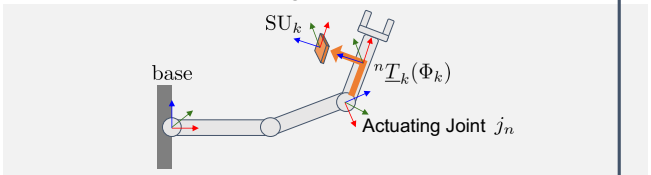
IV. ACCELEROMETER-BASED KINEMATIC CALIBRATION

The goal of kinematically calibrating SUs is to identify the 6D poses (i.e, 3D positions and 3D orientations) of each SU mounted on the robot. With a known pose, the robot can make better sense of tactile information; in Fig. 1 and Section VI, we use this information to demonstrate how an obstacle-avoidance controller can utilize proximity sensor data to detect objects at a distance (up to four meters) and alter the robot's trajectory to prevent collisions.

In this work, we formalize the problem of estimating SU poses along a robot arm with the Denavit-Hartenberg convention detailed in [10]. Importantly, our novel kinematic calibration algorithm is exclusively based on accelerometer readings, as our preliminary testing has indicated that angular velocity readings from the gyroscope introduce significant noise and drift to the data and do not improve performance. We accomplish this by collecting three-axis linear acceleration data (a_x, a_y, a_z) from each SU in multiple robot configurations. Fig. 2 shows a high-level overview of

Fig. 2: Overview of the kinematic calibration algorithm: 1) collect IMU data ${}^k\vec{a}^m_s$ and ${}^k\vec{a}^m_d$ at static and dynamic poses for all SU_k, $\forall k = \{1, \cdots, K\}$; 2) define a kinematic model from joint j_n to SU_k expressed as ${}^nT_k$ that is parameterized by Φ_k; 3) optimize the parameters Φ_k by minimizing the static and dynamic error functions E_s and E_d for each SU_k. q_n and $\dot{q}_n$ represent joint angle and velocity of j_n.

the proposed kinematic calibration algorithm; its three core elements are detailed below.

A. Data Collection and Generation

The data collection step is tasked with obtaining sensor readings used in the batch optimization algorithm detailed in Section IV-C. Data collection is split in two parts: static-pose data collection and dynamic-pose data collection. For the purposes of this work, the order in which these steps are performed is not relevant. The static-pose step collects accelerometer readings in multiple joint configurations with a stationary robot, which serves as a baseline to compensate for static forces acting on the sensor when the robot is motionless—i.e., gravity and baseline measurements. The dynamic-pose step is tasked with collecting data with each individual joint oscillating in a sinusoidal pattern to span its full operational range; such information is used to perform the optimization detailed in Section IV-C after the baseline measurements are compensated for. More specifically, we independently actuate each joint p with a low-level velocity control defined by $\dot{q}_p = A\sin(2\pi ft)$, where f is the frequency, t is the time, $\dot{q}_p$ is the joint velocity, and A is the amplitude of the pattern. This formulation allows to flexibly and conveniently reach either of joint angle, velocity or acceleration limit; in particular, exerting large acceleration allows for a larger signal-to-noise ratio, thus helping the optimizer identify acceleration values in the presence of

---

[1] Finalized circuit schematics can be seen at https://github.com/HIRO-group/RoboSkin_Circuit_Schematics.
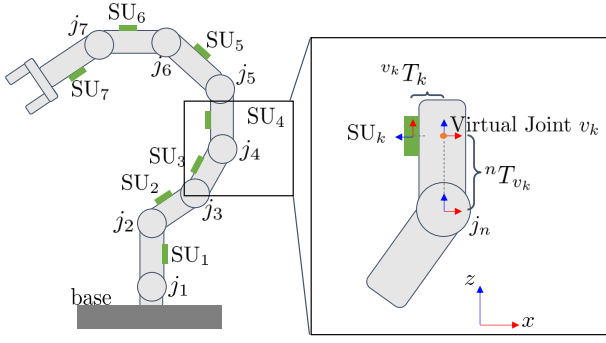
Fig. 3: Schematic depiction of SUs mounted on a robotic arm (left) with each kinematic element of the robot labeled. The zoomed-in image on the right illustrates how the SU poses are estimated through DH parameters of the joint connecting to the previous link: the transformation from the n-th joint to the k-th SU can be computed via an intermediate "virtual joint" v_k located perpendicular to both the joint's and the SU's reference frames. The axes depict the three coordinate systems: joint n, SU k, and corresponding virtual joint v_k.

noise. The data collection process is performed once and saved for later use.

B. Kinematic Model Considerations

Before running the optimization step detailed in Section IV-C, it is imperative to define a suitable kinematic model of the robot. In this paper, we employ the modified Denavit-Hartenberg representation detailed in [18], which is based on the original work in [27]. The DH notation is particularly convenient because it allows for expression of relative poses of kinematic elements along a robot's chain with only four parameters rather than six, thus reducing the dimensionality of the optimization problem. This dimensionality reduction is achieved by leveraging the standard mechanics of prismatic and revolute joints used in the vast majority of robot arms; however, the same cannot be said of skin units, which can be placed arbitrarily along a robot arm and cannot take advantage of this simplification. To guarantee compatibility with the DH convention, a "virtual joint" is located between joint n and SU k, as depicted in Fig. 3. This conveniently allows for the problem of estimating the 6 degree-of-freedom (DoF) transformation matrix ^nT_k between the nth joint's reference frame and the kth SU's frame to be decomposed into the estimation of two, DH-compatible, 4-DoF matrices:

$$^nT_k = \begin{bmatrix} ^nR_k & ^n\vec{r}_k \\ 0 & 1 \end{bmatrix} = ^nT_{v_k} \left[\Phi\left(v_k\right) \right] \cdot ^{v_k}T_k \left[\Phi\left(SU_k\right) \right], \quad (1)$$

where ^nR_k and $^n\vec{r}_k$ are the rotational and translational components of ^nT_k, and $\Phi_k = \left[\Phi\left(v_k\right), \Phi\left(SU_k\right)\right]$ are the DH-compatible parameters to be estimated—composed of $\Phi\left(v_k\right) = (d_{v_k}, \theta_{v_k}, 0, 0)$ for virtual joint v_k, and $\Phi\left(SU_k\right) = (d_{SU_k}, \theta_{SU_k}, a_{SU_k}, \alpha_{SU_k})$ for the corresponding SU_k, where d, θ, a and α represent displacement along z-axis, rotation around z-axis, displacement along x-axis and rotation along x-axis of the local frame respectively (See Fig. 3). Note that

the above equation still represents a 6-DoF problem, thus flexibly allowing for any SU pose to be represented relative to its corresponding joint n.

C. Parameter Optimization

The last step of the optimization algorithm pertains with parameter optimization, which is tasked with calibrating the kinematic model detailed in Section IV-B is with the data collected in Section IV-A. Initially, the DH parameters for all SUs are randomly initialized within a given range (cf. Section V). At every iteration step, the ground truth accelerations are estimated from the kinematic model as detailed in Section IV-C.1; then, a global optimization algorithm is used to estimate the DH parameters by minimizing the error between actual and predicted accelerations. This sequence iterates until it reaches a stopping condition; that is, when the iterative update in DH parameters estimations becomes sufficiently small. Details are provided below.

1) Acceleration Estimation: The acceleration $^k\vec{a}_k$ exerted on SU_k in its frame of reference (FoR) k can be estimated via the Newtonian Equation of motion for a rotating coordinate system. More specifically, the total acceleration can be seen as a composition of three components: gravitational acceleration $\vec{g}_k$, centripetal acceleration $\vec{a}_{\text{cen},k}$ and tangential acceleration $\vec{a}_{\text{tan},k}$. These three components can be formalized as follows:

$$^k\vec{a}_k(\Phi_k, q_n, \dot{q}_n, \ddot{q}_n) = {}^kR_n(\Phi_k, q) \cdot \left(\vec{g}_k + \vec{a}_{\text{cen},k} + \vec{a}_{\text{tan},k}\right) \tag{2}$$

$$\vec{a}_{\text{cen},k}(\Phi_k, q_n, \dot{q}_n) = \vec{\dot{q}}_n \times \left(\vec{\dot{q}}_n \times {}^n\vec{r}_k(\Phi_k, q)\right) \tag{3}$$

$$\vec{a}_{\text{tan},k}(\Phi_k, q_n, \ddot{q}_n) = \vec{\ddot{q}}_n \times {}^n\vec{r}_k(\Phi_k, q) \tag{4}$$

With reference to Eq. (3), the centripetal acceleration $\vec{a}_{\text{cen},k}$ is composed of the joint angular velocity $\vec{\dot{q}}_n = [0, 0, \dot{q}_n]$ (which can be measured during data collection), and the position vector $^n\vec{r}_k$ which represents the translational component of ^nT_k in Eq. (1). Since they are all defined in the nth rotating joint's reference frame, it has to be rotated from reference frame j_n to SU_k through ^kR_n. Note that the full acceleration vector calculation is parameterized by Φ_k, which are the to-be-optimized DH parameters.

The tangential component of joint motion on acceleration reading of the SU requires the measure of joint angular acceleration, which is not readily available in robot manipulators and therefore requires a numerical estimation. Conventional methods (e.g. [28]) employ the second derivative of the position vector $^n\vec{r}_k$ to estimate a SU's acceleration. However, this differentiation does not include gravity and incorrectly incorporates an additional centripetal force from the SU moving on a rotating joint, as detailed in Fig. 4. This additional centripetal component is not only redundant, but also negatively affects optimization performance; therefore, in this work we remove non-tangential forces by employing Eq. (5), followed by Eq. (6):
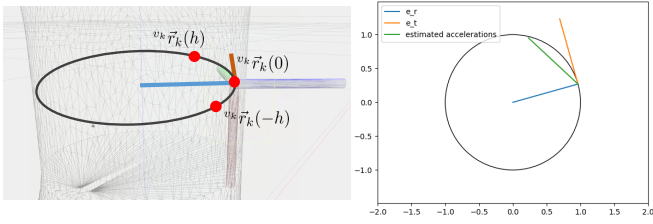
Fig. 4: Illustration of the numerically computed tangential accelerations for joint n in its respective FoR. The plot on the right shows the accelerations computed by taking the second derivative of the position vectors (the red dots on the left). The accelerations include not only a tangential acceleration but also an unnecessary centripetal acceleration.

$$\vec{a}_{\text{tan},k} = \vec{e}_{\text{tan}} \cdot \frac{\vec{r}(h) + \vec{r}(-h) - 2 \cdot \vec{r}(0)}{h^2} \quad (5)$$

$$^k\vec{a}_{\text{tan},k} = {}^kR_n \cdot \vec{a}_{\text{tan},k} \quad (6)$$

Above, $\vec{r}(h)$ is a parametrization of position vector $^n\vec{r}_k$ as function of discrete time $h = 0.001$, and $\vec{e}_{\text{tan}}$ is an unit vector in the tangential direction.

2) Error Functions: For each SU_k attached to joint n, we optimize its DH parameters Φ_k by separately minimizing two distinct error functions: the static acceleration error E_s and the dynamic acceleration error E_d. For each recorded static pose p, the E_s only holds rotational information because it depends exclusively on gravitational acceleration; therefore, it can be used to estimate the rotational DH parameters included in $^nR_k(\Phi_k)$. It is worth noting that there is a misalignment in coordinate frames: the measured acceleration $^k\vec{a}_{k,p}^m$ is defined in its own reference frame k, whereas the gravity vector $^b\vec{g}$ is defined in the base frame. Therefore, the acceleration vector is converted into base frame using a rotation matrix ^bR_n as computed from forward kinematics. The static acceleration error is then defined as the L2 norm between the measured acceleration and the gravity vector:

$$E_s(k) = \frac{1}{P} \sum_{p=1}^{P} \left| {}^bR_n \cdot {}^nR_k(\Phi_k) \cdot {}^k\vec{a}_{k,p}^m - {}^b\vec{g} \right|^2, \quad (7)$$

where P represents the total number of poses collected. Conversely, the dynamic acceleration error $E_{d,k}$ is composed of both rotational and translational information. Upon optimization of $^nR_k(\Phi_k)$ from Eq. (7), the translational component of the problem can be then estimated via the L2 norm between the measured acceleration $^k\vec{a}_{k,d,p}^m$ and the ground truth estimated from kinematics $^k\vec{a}_{k,d,p}(\Phi_k)$ and expressed in the SU's reference frame:

$$E_d(k) = \frac{1}{P} \sum_{p=1}^{P} \sum_{\substack{d>0, \\ d=n-2}}^{n} \left| {}^k\vec{a}_{k,d,p}^m - {}^k\vec{a}_{k,d,p}(\Phi_k) \right|^2, \quad (8)$$

where d represents the sole joint being actuated during the data collection process to exert an acceleration on each SU. Three previous joints before the nth joint are used to compute this error function. $^k\vec{a}_{k,d,p}$ can be computed from Eq. (2) for
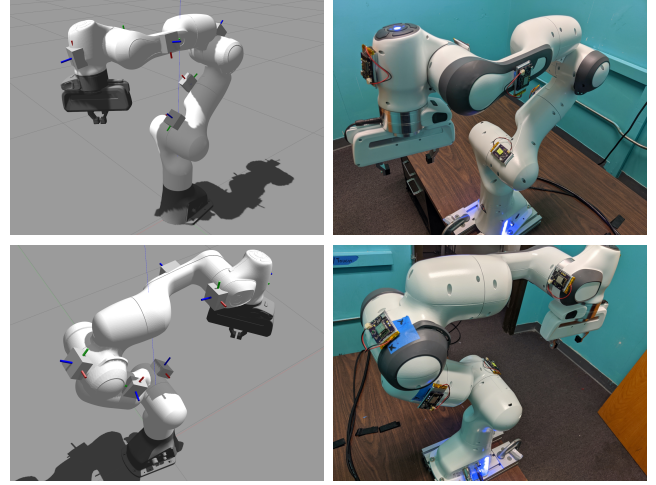

Fig. 5: Comparison between estimated SU poses (left) and ground truth poses (right) on a Franka Emika Panda robot arm. The grey boxes represent the estimated SU with x (red), y (green), z (blue) axes respectively.

joint d and pose p. The optimization procedure is repeated for all SUs.

V. EXPERIMENT DESIGN

To validate the proposed approach with reproducible and precise ground truth information, a quantitative evaluation is conducted in simulation via the Robot Operating System (ROS, [29]) and the Gazebo simulator [30]. A subsequent deployment on a real-world Franka Panda robot (see Figs. 1 and 5) is then used to provide qualitative analysis and a control example. We demonstrate that our algorithm outperforms prior work [28] and allows for precise autonomous calibration on 7-DoF robotic arms in both simulated and real-world environments. In the following, we detail parameters of our experimental evaluation so as to improve reproducibility and repeatability of this work. Data and algorithms are available at the associated GitHub repository[2].

Due to limitations of the original algorithm presented in [28], we modified it by including gravitational acceleration to follow Eq. (2). For clarity, in the following we report a comparison of the proposed method against this enhanced version—which we hereinafter refer to as modified Mittendorfer's method (or *mMM*), as the original work was orders of magnitude worse than ours. Similarly to [28], the base of the robot serves as the starting point for kinematic exploration; for simplicity, we aligned the base of the Panda arm to coincide with the world reference frame. In both the simulation and the real world, we mount and simultaneously calibrate six SUs placed perpendicularly to the surface of each link—one per link except for the more proximal link. It is worth noting that an SU placed on the first link cannot be calibrated because there exists an infinite number of solutions with the same acceleration values, due to the fact that all the dependent values $\vec{\dot{q}}$, $\vec{\ddot{q}}$, and $\vec{r}$ are

---

[2] https://github.com/HIRO-group/ros_robotic_skin

TABLE I: Comparison of the Average L2 Norm positional error [cm] and absolute distance in quaternion space (with units multiplied by 10^{-1} for reasons of space) of our method (OM) against the modified Mittendorfer method (mMM). 4 different sets of poses are tested and optimization is run 10 times per set; average and standard deviations of these 40 trials are below.

		SU_1	SU_2	SU_3	SU_4	SU_5	SU_6	Average
Positional	OM	0.28 ± 0.15	0.39 ± 0.15	0.78 ± 0.39	1.15 ± 0.88	0.80 ± 0.36	0.25 ± 0.095	0.66 ± 0.60
Error [cm]	mMM	5.3 ± 7.1	2.6 ± 0.23	5.4 ± 4.9	9.5 ± 9.0	1.4 ± 0.91	2.3 ± 2.4	4.4 ± 5.9
Quaternion	OM	0.10 ± 0.058	0.054 ± 0.041	0.023 ± 0.012	0.019 ± 0.013	0.021 ± 0.023	0.045 ± 0.053	0.044 ± 0.059
Distance	mMM	5.5 ± 5.9	0.16 ± 0.11	3.6 ± 6.1	0.057 ± 0.022	0.044 ± 0.015	2.5 ± 4.3	2.0 ± 4.4

TABLE II: True and Estimated DH Parameters of one of the four SU configuration sets we tested.

	j_1 to SU_1		j_2 to SU_2		j_3 to SU_3		j_4 to SU_4		j_5 to SU_5		j_6 to SU_6	
	true	est	true	est	true	est	true	est	true	est	true	est
θ_v	1.571	1.571	0.000	-0.014	-1.571	-1.571	3.141	3.141	-1.571	-1.571	1.571	1.565
d_v	0.060	0.060	-0.080	-0.076	0.080	0.085	-0.100	-0.007	0.030	0.030	0.000	0.001
θ_{SU}	-1.571	-1.545	0.000	-0.001	1.571	1.567	3.141	3.141	1.571	1.541	-1.571	-1.570
d_{SU}	0.060	0.062	0.050	0.053	0.060	0.060	0.100	0.142	0.050	0.048	0.050	0.049
a_{SU}	0.000	0.001	0.000	0.000	0.000	0.001	0.000	-0.001	0.000	-0.001	0.000	-0.001
α_{SU}	1.571	1.583	1.571	1.568	1.571	1.570	1.571	1.565	1.571	1.570	1.571	1.571

equal along the first link. We used the native Gazebo IMU plugin to simulate the IMU physics and behavior. The DH parameters are initialized randomly within reasonably large bounds: $\theta_v \in [-\pi; \pi]$, $d_v \in [-1; 1]$, $\theta_{SU} \in [-\pi; \pi]$, $d_{SU} \in [-1; 1]$, $a_{SU} \in [0; 1]$, $\alpha_{SU} \in [-\pi; \pi]$ (units are in meters for displacements and in radians for angles).

Data collection is performed only once both in simulation and in the real world. $P = 16$ poses were collected, taking less than 20 minutes. We used a joint velocity controller to oscillate at 2rad/s. Both IMUs and controller messages were published and controlled at 100Hz. Two seconds of rest were set between each sequence to prevent outliers and to make sure the robot could reach its desired position in time.

The parameter optimization step leveraged a randomized global optimization algorithm (MLSL_LDS) [31] together with a derivative-free local optimizer (LN_NEWUOA) [32] of the NLopt library [33]. Although we do initialize the DH parameters within a set of boundaries, we use NEWUOA unconstrained optimization to allow for better exploration of the DH parameter search space—which, in our case, is a 36-dimensional space, due to each one of the six SUs having six parameters to optimize. Due to the nature of the DH convention, there are multiple solutions for each SU's DH parameters, and using unconstrained methods allows us to more systematically explore those solutions and avoid local minima. It is worth noting that performance of the proposed algorithm is increased when optimizing all IMUs concurrently, as data coming from multiple IMUs helps with redundancy and noise reduction; however, the proposed method works even when calibrating each IMU independently from the rest. The optimization for the static acceleration error is terminated once the error is below a threshold of 0.01 or if the parameter difference per iteration is less than 0.001. Similarly, the optimization for the dynamic acceleration error is stopped once the change in the SU DH parameters reaches a threshold of less than 0.001. Both data collection and optimization were performed on a Lenovo Thinkpad X1 Extreme, 2nd Generation, with 12 Intel i7-9750H CPUs and a GeForce GTX 1650.

VI. RESULTS AND DISCUSSION

A. Simulation experiments

In simulation, we evaluate our algorithm based on the average L2 Euclidean distance and quaternion distance over all of the skin units. To demonstrate the robustness of our approach, we randomized our experiment by the following: i) we tested 4 different sets of SU configurations, i.e. we placed the SUs in 4 different randomized locations along the robot's body; ii) we optimized SU poses 10 times per set. Additional tests do not seem to alter average results nor to increase variance. The mean and standard deviation of the resulting calibration on these 40 trials are depicted in Table I, and the corresponding estimated DH parameter values are shown in Table II. As mentioned in Section V, we chose not to compare against the original method in [28] because of poor performance (about 90cm positional error) and low convergence characteristics of the vanilla algorithm. According to Table I, while the modified version of Mittendorfer's method (mMM) had an average error of 4.4cm and a high variance of ± 5.9cm, our method (OM) has an error of 0.66cm for the given SU placements, a six-fold improvement; similar considerations can be said of the orientation error. In addition to the accuracy, we also validated the effectiveness of separating the original optimization problem into two parts. Thanks to this solution, we measured a 4x speed up of the separated problem (433 seconds) with respect to the original problem (1797 seconds).

B. Real-world experiments

To further test the effectiveness of the algorithm, we conducted an experiment in the real world—and more specifically on a Franka Panda robot. The following is worth noting: i) the work in [28] did not perform real-world calibration experiments, and to the best of our knowledge our paper is the first time such an approach has been validated on a real robot platform; ii) we expect somewhat reduced performance due to the introduction of additional layers of complexity—from possible time delays in the robot network that misalign collected data, to sensor noise in the IMUs, to hardware
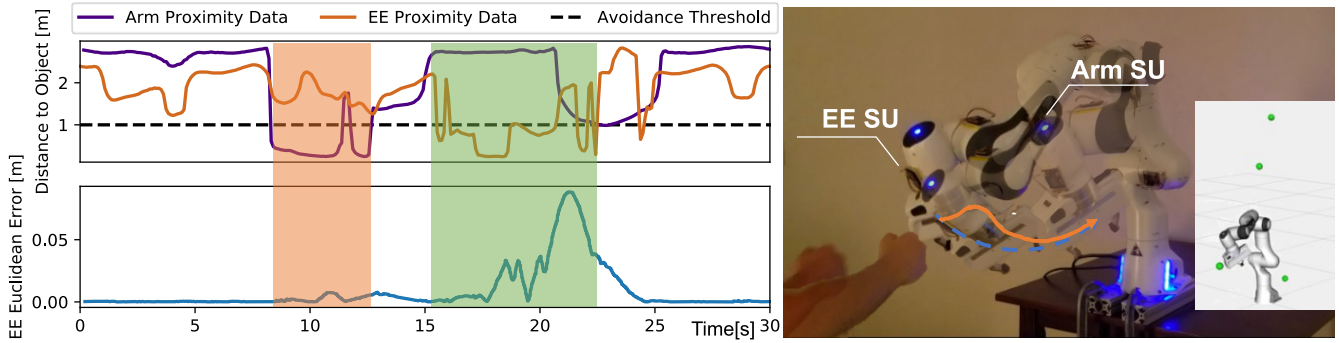
Fig. 6: Trajectory redirection for obstacle avoidance. Right: a human approaches a robot with calibrated skin units; their accurate pose estimation allows the robot to precisely perceive the person from a distance. Bottom-Right: real-time visualization of the robot and obstacles (shown as green spheres) as sensed by the proximity sensors placed on the calibrated SUs; only below-threshold signals (arm and end-effector SU) affect the robot's motion. Left: combined graph of proximity data over time vs. the robot's end effector error along a circular trajectory. The two shaded areas indicate intervals when the human approaches a skin unit (below the avoidance threshold), causing only a temporary perturbation in the error. The perturbation stays low in the orange colored area (arm proximity activation) compared to the green area (end effector proximity activation) because the robot leverages the redundancy afforded by its kinematic chain to avoid colliding with the person while concurrently satisfying its main task objective.

limitations of the platform. Similarly to the simulation, IMU data is retrieved at 100Hz and recorded for 3 seconds at each pose; however, we lengthen the resting time to 10 seconds in order to allow for better separation of data collected at each pose. Additionally, we modified the oscillation magnitude to 2.0 and frequencies to 0.75, 0.25, 0.35, 0.55, 0.75, 0.9 and 0.9Hz for each joint respectively, in order to exert high acceleration but restrain the displacement. Data collection on the real robot was conducted on a computer with a real-time kernel and took less than 30 minutes, to allow ample time for the real robot to move from one position to another. We collected data for $P = 12$ different poses. A qualitative comparison of the actual SU poses against the estimated SU poses are is shown Fig. 5. The grey boxes in simulation represent the estimated SUs and the corresponding pictures show the actual SU placements. Although only qualitative, it is clear how the poses are, for the vast majority, very similar from one another—apart from one SU that seems to be more displaced apart. On the real robot, the resulting average position error for all SUs was 5.1cm and the average orientation error was 0.12, whereas mMM achieves a 9.2cm error for position and a 0.43 error for orientation. As mentioned in Section V, the original contribution from [28] was significantly worse in the real world, with errors of up to 90cm—i.e., an order of magnitude more than our work.

C. Control example

After calibration, the estimated poses are then used for an obstacle avoidance control example, depicted in Fig. 6 and shown in the accompanying video. To demonstrate the overall system, we implemented an obstacle avoidance controller to track waypoints in Cartesian space. We formalized it as a quadratic optimization problem to compute the optimal joint velocities while avoiding obstacles. To leverage the skin units for obstacle avoidance, we first find points on

the robot's body closest to the obstacles and set constraints to restrict the approaching velocity. Further details can be found in [9]. In the example in Figure 6, the robot is tasked with following a continuous circular trajectory in operational space, and constantly monitors the calibrated proximity readings at the nominal frequency of 50Hz. The bottom right image in Fig. 6 shows a visualization of the robot and the obstacles detected in real time by the proximity sensors; the robot will take corrective actions if an obstacle is sensed within one meter of a skin unit (as depicted by the dashed line in the graph). It is worth mentioning that in this demonstration we have chosen an arbitrary distance of one meter to guarantee operational safety in presence of humans; however, such threshold can be modified as function of task progress and perceived comfort of the human in presence of the robot. The effectiveness of the control example is concretely demonstrated in the left graph. The two shaded areas indicate when the human is approaching a skin unit below the avoidance threshold: the perturbation stays low in the orange colored area (arm proximity activation), whereas it is more prominent in the green area (end effector proximity activation); in the first case, the robot effectively leverages its redundancy to avoid colliding with the person—which is not possible when the obstacle is approaching the end-effector of the robot's kinematic chain. Our control example demonstrates that the estimation accuracy is sufficient for the motivations highlighted in Section I—that is, nearby space perception for close-proximity Human-Robot Collaboration. The controller is able to quickly adjust its trajectory after a skin unit senses an obstacle, leading to operationally safe behavior. In addition to these results, we have shown further information on both the algorithm and the control example in the accompanying video (also available at this link `https://youtu.be/LzVkLmw5WA0`).

VII. Conclusions and Future Work

In this paper, we presented an accurate method for kinematically calibrating multiple skin units mounted on a collaborative robotic arm. Our approach is robust to noise and scales gracefully with the number of SUs, while not requiring expensive tracking systems with computational overhead. The proposed work can effectively estimate SU poses with sub-cm precision in simulation, and less than 5cm in the real world, for a total time from installation to effective utilization of less than 40 minutes (30m for data collection $+ \sim$ 7m for optimization of multiple SUs). We have open-sourced our code and data for the research community, and we are hopeful that this demonstration will be the first of many to leverage calibrated skin unit poses for a future of plug-and-play distributed artificial tactile sensing for robotics.

In future work, we will focus our attention in four directions: i) extensively investigate the trade-off between calibration accuracy and safe robot operation in human-populated environments (by comparing against human skin sensitivity and capability); ii) understand optimal placement of SUs along the robot's body to achieve best coverage and robust perception of the robot's surroundings; iii) further iterate on the artificial skin technology, adding heterogeneous sensing, flexible electronics, and dense whole-body coverage; iv) leverage i-iii) for novel robot behavior, such as environment-informed null-space control and human-aware motion planning. Ultimately, these applications will bring us closer to the realization of true, inherently safe human-robot interaction.

References

[1] F. Flacco, T. Kröger, A. De Luca, and O. Khatib, "A depth space approach to human-robot collision avoidance," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 338–345.

[2] E. Magrini, F. Flacco, and A. De Luca, "Control of generalized contact motion and force in physical human-robot interaction," in *IEEE international conference on robotics and automation*, 2015, pp. 2298–2304.

[3] R. S. Dahiya and M. Valle, *Robotic tactile sensing: technologies and system*. Springer Science & Business Media, 2012.

[4] A. Roncone, M. Hoffmann, U. Pattacini, and G. Metta, "Learning peripersonal space representation through artificial skin for avoidance and reaching with whole body surface," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2015, pp. 3366–3373.

[5] F. Mastrogiovanni, L. Natale, G. Cannata, and G. Metta, "Special issue on advances in tactile sensing and tactile-based human–robot interaction," *Robotics and Autonomous Systems*, pp. 227–229, 2015.

[6] G. Cheng, E. Dean-Leon, F. Bergner, J. R. G. Olvera, Q. Leboutet, and P. Mittendorfer, "A comprehensive realization of robot skin: Sensors, sensing, control, and applications," *Proceedings of the IEEE*, vol. 107, no. 10, pp. 2034–2051, 2019.

[7] G. Metta, L. Natale, F. Nori, G. Sandini, D. Vernon, L. Fadiga, C. Von Hofsten, K. Rosander, M. Lopes, J. Santos-Victor, *et al.*, "The iCub humanoid robot: An open-systems platform for research in cognitive development," *Neural networks*, vol. 23, no. 8-9, pp. 1125–1134, 2010.

[8] D. H. P. Nguyen, M. Hoffmann, A. Roncone, U. Pattacini, and G. Metta, "Compact real-time avoidance on a humanoid robot for human-robot interaction," in *ACM/IEEE International Conference on Human-Robot Interaction*, 2018, pp. 416–424.

[9] C. Escobedo, M. Strong, M. West, A. Aramburu, and A. Roncone, "Contact anticipation for physical human–robot interaction with robotic manipulators using onboard proximity sensors," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2021.

[10] B. Siciliano and O. Khatib, *Springer handbook of robotics*. Springer, 2016.

[11] C. S. Gatla, R. Lumia, J. Wood, and G. Starr, "An automated method to calibrate industrial robots using a virtual closed kinematic chain," *IEEE Transactions on Robotics*, vol. 23, no. 6, pp. 1105–1116, 2007.

[12] J.-S. Hu, J.-J. Wang, and Y.-J. Chang, "Kinematic calibration of manipulator using single laser pointer," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2012, pp. 426–430.

[13] F. Flacco, T. Kroeger, A. De Luca, and O. Khatib, "A depth space approach for evaluating distance to objects," *Journal of Intelligent & Robotic Systems*, vol. 80, no. 1, pp. 7–22, 2015.

[14] A. Roncone, M. Hoffmann, U. Pattacini, L. Fadiga, and G. Metta, "Peripersonal space and margin of safety around the body: learning visuo-tactile associations in a humanoid robot with artificial skin," *PloS one*, vol. 11, no. 10, 2016.

[15] A. Jain, M. D. Killpack, A. Edsinger, and C. C. Kemp, "Reaching in clutter with whole-arm tactile sensing," *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 458–482, 2013.

[16] W. Yuan, S. Dong, and E. Adelson, "Gelsight: High-resolution robot tactile sensors for estimating geometry and force," *Sensors*, vol. 17, no. 12, p. 2762, 2017.

[17] A. Yamaguchi and C. G. Atkeson, "Implementing tactile behaviors using fingervision," in *2017 IEEE-RAS 17th International Conference on Humanoid Robotics (Humanoids)*. IEEE, 2017, pp. 241–248.

[18] J. Hollerbach, W. Khalil, and M. Gautier, "Model identification," in *Springer handbook of robotics*. Springer, 2016, pp. 113–138.

[19] G. Du and P. Zhang, "Imu-based online kinematic calibration of robot manipulator," *The Scientific World Journal*, vol. 2013, 2013.

[20] D. Roetenberg, H. Luinge, and P. Slycke, "Xsens mvn: Full 6dof human motion tracking using miniature inertial sensors," *Xsens Motion Technologies BV, Tech. Rep*, vol. 1, 2009.

[21] R. Zhu and Z. Zhou, "A real-time articulated human motion tracking using tri-axis inertial/magnetic sensors package," *IEEE Transactions on Neural systems and rehabilitation engineering*, vol. 12, no. 2, pp. 295–302, 2004.

[22] R. Patel and N. Correll, "Integrated force and distance sensing using elastomer-embedded commodity proximity sensors." in *Robotics: Science and systems*, 2016.

[23] D. Hughes, J. Lammie, and N. Correll, "A robotic skin for collision avoidance and affective touch recognition," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1386–1393, 2018.

[24] D. Göger, H. Alagi, and H. Wörn, "Tactile proximity sensors for robotic applications," in *2013 IEEE International Conference on Industrial Technology (ICIT)*. IEEE, 2013, pp. 978–983.

[25] A. Roncone, M. Hoffmann, U. Pattacini, and G. Metta, "Automatic kinematic chain calibration using artificial skin: self-touch in the icub humanoid robot," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2014, pp. 2305–2312.

[26] K. Stepanova, T. Pajdla, and M. Hoffmann, "Robot self-calibration using multiple kinematic chains—a simulation study on the icub humanoid robot," *IEEE Robotics and Automation Letters*, vol. 4, no. 2, pp. 1900–1907, 2019.

[27] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower pair mechanisms based on matrices," *Journal of applied mechanics*, vol. 77, no. 2, pp. 215–221, 1955.

[28] P. Mittendorfer and G. Cheng, "Open-loop self-calibration of articulated robots with artificial skins," in *IEEE International Conference on Robotics and Automation*, 2012, pp. 4539–4545.

[29] M. Quigley, K. Conley, B. Gerkey, J. Faust, T. Foote, J. Leibs, R. Wheeler, and A. Y. Ng, "Ros: an open-source robot operating system," in *ICRA workshop on open source software*, vol. 3, no. 3.2. Kobe, Japan, 2009, p. 5.

[30] N. Koenig and A. Howard, "Design and use paradigms for gazebo, an open-source multi-robot simulator," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 2149–2154.

[31] A. H. G. R. Kan and G. T. Timmer, "Stochastic global optimization methods," *Mathematical Programming*, vol. 39, pp. 27–78, 1987.

[32] M. J. D. Powell, "The NEWUOA software for unconstrained optimization without derivatives," in *Proc. 40th Workshop on Large Scale Nonlinear Optimization*, 2004.

[33] S. G. Johnson, "Nlopt." [Online]. Available: https://nlopt.readthedocs.io/en/latest/