From Multi-modal Tactile Signals to a Compliant Control*

Emmanuel Dean-Leon, Florian Bergner, Karinne Ramirez-Amaro and Gordon Cheng

***Abstract*— In this paper, we present an *end-to-end* approach transforming multi-modal tactile signals into a compliant control to generate different dynamic robot behaviors. This is obtained by fusing multi-modal sensor signals from our artificial skin and joint sensors with different control approaches. One advantage of these compliant behaviors is to produce safer robots, especially for physical Human-Robot Interaction. A key component of our framework is a robot parametric modeling based on the artificial skin multi-modal sensors (proximity, force and acceleration). These generated models are used to control a robot improving and even changing its dynamic behavior. We validate our framework in a real wheeled humanoid robot, where our presented framework enables a stiff robotic system to be compliant and react to multi-modal tactile events (pre-contacts and contacts).**

I. Introduction

Physical Human–Robot Interaction (pHRI) [1] allows a robot to share spaces with human co-workers. This expands the potential applications of robotic systems into different fields since their flexibility can be extended by combining the high adaptability of the human partners with the accuracy and endurance of a robot. This is of particular importance for the new way of teaching robot sequences using programming by demonstration methods which explicitly involves physical interactions with the robot [2].

One of the most important requirements to develop adequate robots for pHRI is the ability to detect tactile interactions with the surrounding environment. This means the robot should be able to detect and identify pre-contact, as well as contact interactions. This includes recognizing the location, magnitude, and nature (dynamic characteristics) of the contacts. The sensing ability enables robots to discriminate between unforeseen collisions and intended tactile interactions (differentiate between *good* and *bad* contacts). The first direct benefit of this capability is the increment of the overall robot's safety, especially when performing cooperative activities with human counterparts, e.g. co-manipulation and compliant tasks (frequent tasks in the industrial and service domains). Being able to detect and classify contact reactions is a critical element to accomplish common tasks for humans, e.g. navigate in confined spaces [3], open doors [4] or manipulate objects [5].

The contact information can be generated using Force/Torque sensors (F/T) [6], commonly located at the end-effector, e.g. wrist or ankles, or using Inertial Measurement Units (IMU) [7] based on models of the contact stiffness. Another interesting methodology is to use proprioceptive sensors to estimate the external contacts based on the internal generalized forces, this method is known as *momentum-based residual* [8], [9]. The main advantage of this technique is that it doesn't require F/T sensors to estimate the contact forces, and for the case of single contact points, the position of the contact and the applied force can be estimated. However, for multiple contact points, the location of the contacts is needed to estimate the exerted forces [10]. Moreover, a precise knowledge of the robot's dynamic model is required which imposes hard restrictions for this technique.

On the other hand, humans can handle naturally multiple contacts with the environment, e.g. exploiting environment constraints to reduce efforts, handling heavy objects using the whole-body, using multiple external contacts for support, etc. The principal element used in this sort of activities is our skin covering the entire body. This sensitive and multi-modal skin is a key component to handle contact rich scenarios in the everyday life. In this sense, direct measurement of the tactile interactions using artificial skin has gained popularity in the last years. For an extensive review on artificial skin, its principal motivation, and the most used technologies see [11]. The preferred sensor used in the artificial skin is a *tactile* sensor, e.g. [12], [13] and [14]. However other

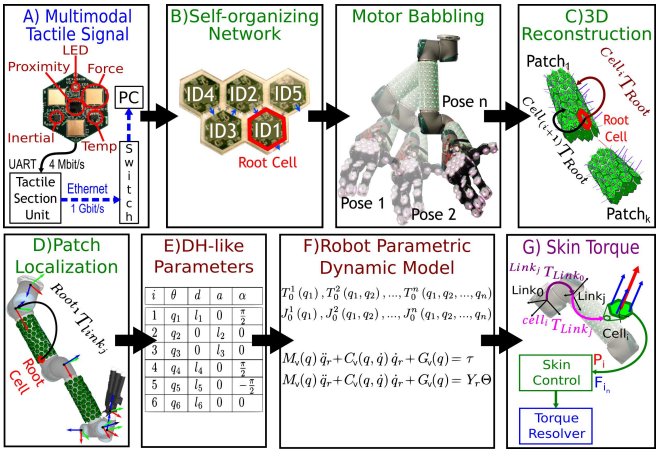

Fig. 1. Block diagram of our *end-to-end* approach. A) Multi-modal tactile signals and processing, B) Network self-exploration and configuration, C) Automatic reconstruction of the 3D surface and skin patches definition, D) Skin patch calibration with respect to the robot links, E) Robot parametrization using the multi-modal signals of the skin, F) Parametric modeling (Kinematic and Dynamic) of the robot for control design, and G) Control and command interface synthesis.

*This work has received funding from the European Community's Seventh Framework Program (FP7/2007-2013) under grant agreement no. 609206.

Institute for Cognitive Systems, Technical University of Munich Arcisstrasse 21, 80333 München, Germany {dean, florian.bergner, karinne.ramirez, gordon}@tum.de

technologies have been widely used, e.g. [15], [16], [17]. In the literature, we can find several examples of using artificial skin to successfully control a robot. The robot ARMAR-III [18] presents skin patches based on piezo-resistive sensor arrays with embedded multiplexers. The patches have different shapes and are designed for the multiple parts of the robot. Another example is the work of Cannata *et al.* [12] where the authors demonstrate an artificial robot skin based on a set of distributed capacitive tactile sensors that are integrated on the iCub robot. These sensors cover the arms, legs and chest of the robot and provide tactile feedback to detect potential contacts with the environment. Also, Cirillo *et al.* [17] introduce a flexible skin capable of measuring three components of the applied force and the location of the contact point. This skin is used to control a KUKA LWR, and can recognize simultaneously intentional and unintentional contacts.

The existing full-body skins are only modular at the level of individual patches of skin, reducing their flexibility [5]. Furthermore, large-scale artificial robot skin implicates complex problems: a) sensor element organization, b) sensor calibration, c) sensor integration with the robotic system, among others. Fast and flexible reconfigurable sensor systems and its associated control methods are a key component to increase the usability of robots. Especially in industrial scenarios, such systems are expected to decrease the set-up time to reconfigure robotic systems and increase the safety for HRI [19]. However, manual calibration of a growing number of distributed sensors on a large and arbitrarily surface is slow, prone to errors and intractable. Moreover, the general integration of sensor components and controls is not a trivial task and is usually customized for a specific robot. The integration of additional sensors impacts directly on the complexity of the control design since we need to include different mechanisms to fuse all these sensors. Furthermore, the multi-modality is considered as a main and an unquestionable feature of human-robot interaction [20]. This reveals the need to develop multi-modal control approaches that can be adaptable to different robots. Therefore, the development and integration of technologies such as control fusion, artificial skin, and multi-modal control design are needed to increase the capabilities of standard robots, to improve the safe physical interaction with robots, and to simplify the robot programming.

In general, all the approaches found in the literature consider either manual calibration, or detailed *a priori* information, e.g., CAD models of the surface hosting the artificial skin, the local transformations between the sensor elements and the robot joints, or kinematic and dynamic models of the robot to design controllers and integrate the skin sensors signals into the control loop. In this work we present a *End-to-end* approach to transform multi-modal tactile signals into a compliant control method. This approach is able to use the multi-modal information from the artificial skin to generate different dynamic behaviors. The main component of our approach is our self-organizing, self-calibrating multi-modal artificial skin [21], which is used to automatically generate kinematic and dynamic models to render controllers. Our method is not limited to known surfaces and does not require detailed information of the robot. Fig. 1 depicts all the steps of our approach. Our complete approach has been implemented using ROS middleware which provides a convenient infrastructure to develop hardware interfaces in an easy manner. In addition, several robot manufacturers already provide interfaces for their robots in ROS[1]. This simplifies the integration of our sensor/control approach with different robots[2]. The following sections are devoted to describe the different stages of our approach and its experimental validation.

II. Automatic Configuration of the Robotic Skin

In order use the multi-modal signals of our artificial skin to synthesize control schemes, we need to generate adequate Kinematic and Dynamic models for both the robot and the robotic skin. As the engineering of these models can be a complex and tedious task, we try to automate the model generation and reduce the human intervention to a minimum. The automatic model generation fully takes advantage of the multi-modal tactile signals and structural information of our robot skin [21]. The modeling process is divided into 3 groups (in total 6 stages, see Fig. 1): *i)* Skin *intrinsics* calibration, *ii)* Skin *extrinsics* calibration (Sec.II-A), and *iii)* Robot parametrization and modeling (Sec. III).

A. Artificial Multi-Modal Robot Skin

Our artificial skin [21] is a modularized, multi-modal robotic skin. The skin consists of hexagonally shaped *skin cells* (see Fig. 1 A). Each skin cell has the same set of sensors which transduce tactile information of different modalities, such as vibrations (acceleration sensor), pressure (capacitive force sensors), pre-touch (proximity sensor) and temperature (temperature sensor). Micro-controllers on each skin cell sample and filter the sensor data and pack the acquired information. Our artificial skin uses a *event mode* [22], [23] where the skin cells only send novel information asynchronously, on occurrence to the PC. The event mode allows huge reductions (up to 20% of non-event mode) in terms of communication bandwidth and processing power. Neighboring skin cells are connected to each other via Flex-PCBs and exchange information. All the connected skin cells realize a meshed and highly redundant skin network. Skin cells which are directly connected to each other via Flex-PCBs shape an entity called *Skin Patch*. Skin patches are connected to interface boxes which ease the communication between the skin network and the PC.

B. Self-Organizing Skin Cell Network

The skin network describes the communication network between skin cells, skin patches, and interface boxes. The

[1]http://rosindustrial.org/

[2]The only requirement is that the robot provides an external control interface, e.g., position, velocity or torque command interfaces. Besides the UR-5 robot, currently, we are working on the deployment of our method on the following robots: UR-10, PAL REEM-C, and KUKA-LWR.

skin network is highly redundant (skin cells can be connected to four different neighbors, see Fig. 1 B), besides power and network bandwidth there are no restrictions on how to connect skin cells, patches, and interface boxes together. The skin network is fully self-organizing, it automatically builds up bi-directional communication paths between each skin cell and the PC. It also can find new paths when connections are broken. The self-organization takes places in 4 steps [21]: 1) The skin cells are synchronized with the PC and each skin cell explores which of its 4 communication ports is bi-directionally active and can be used for communication. 2) The PC inserts a path exploration token into the skin network and the skin cells automatically create active communication ports to build paths for the bi-directional communication tree. 3) The PC uses the communication tree for distributing unique skin cell IDs. 4) Using these IDs a neighbor list is created. This structural information is essential for the 3D surface reconstruction step.

C. 3D Surface Reconstruction

The reconstruction algorithm [24] can automatically fully reconstruct the shape of the surface covered with skin cells. This 3D reconstruction is essential as it contains the necessary structural information of the skin needed to synthesize multi-modal controllers (e.g. force, proximity, etc.). The algorithm uses the neighbor lists of the skin cells to create a bidirectional connection graph for skin cells and deletes edges in this graph which represent non-direct connections (i.e. cables). At this point, there might be already several unconnected graphs where each of them represents a skin patch. Then, we use the Procrustes algorithm and the gravitational vectors measured at each skin cell to estimate the rotation for all edges in the graph. These rotations represent the relative orientations between neighboring skin cells. To avoid singularities in the rotation estimation, the gravitational vectors should be measured on different robot poses, see Fig .1 C). The rotations and geometric information are combined to generate homogeneous transformations between neighboring skin cells. The algorithm searches for the skin cell which has the shortest accumulated path to all the other skin cells in the patch. This skin cell is defined as *root cell*. Finally, we calculate the homogeneous transformations ${}^{Cell_i}T_{Root_k}$ from every skin cell in the patch to this root cell, i.e. we obtain the *intrinsics* calibration of the skin patches.

D. Patch Localization

As each skin patch is rigidly attached to a specific robot link (see Fig. 1 D), the next step is to determine the relative transformation between the root cell and its robot link, namely ${}^{Root_k}T_{Link_j}$. This can be done either automatically or manually (see Fig. 2). The manual calibration uses the robot description (URDF[3]). The 3D model of the skin patch is the result of the reconstruction algorithm. We attach an interactive marker to the root cell and visualize the complete patch and the robot in Rviz[4]. The interactive marker allows

[3]http://wiki.ros.org/urdf
[4]http://wiki.ros.org/rviz

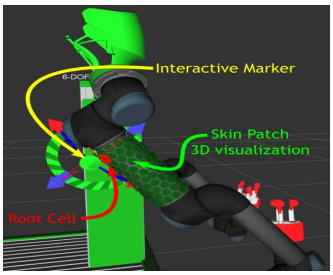

Fig. 2. Interactive skin patch calibration. The user can set the relative pose of the *Root Cell* (${}^{Root_k}T_{Link_j}$) using the interactive marker.

the user to virtually move the patch to the desired robot location. This inherently defines the transformation between robot link and root cell. The automatic calibration uses an external camera and the RGB LED of the skin cells as adaptive markers [25] which can change colors, structure and blink in different patterns. The algorithm uses these markers and the OpenCV library to estimate the transformation between root cell and camera frame. In combination with a known transformation between the camera and the robot base frame, the pose of the root cell in the robot frame can be determined. Finally, the rigid transformation of every skin cell in the robot link is easily computed as:

$$ {}^{Cell_i}T_{Link_j} = {}^{Root_k}T_{Link_j} \left({}^{Cell_i}T_{Root_k} \right). \quad (1) $$

III. Robot Parametric Modeling

Using the *intrinsic* and *extrinsic* skin parameters, and the multi-modal sensor signals, we can automatically obtain robot parameters and models. This is divided in two stages (see Fig. 1 E, F).

A. Full Automatic Kinematic Modeling

For the kinematic model, we developed an algorithm [26] which automatically acquires a DH-like parametrization of the robot and the skin by using skin cell to joint associations, special robot movement patterns and acceleration measurements in every skin cell. The skin cell to joint associations [27] (more specifically: which skin cells are directly and rigidly attached to the moving part of a joint) can be determined through joint wise movements and acceleration changes observed in the skin cells which result in an *activity matrix*. The DH-like parameters are estimated from the *circular points* of the different joints using accelerations measured at a specific skin cell for joint-wise movements. A *circular point* is defined as a joint axis and the radial distance to the skin cell (the center of the measured acceleration). The measured acceleration is a superposition of 3 different linear accelerations: gravitational, tangential and centripetal accelerations. We design the robot motions in such a way that these acceleration components can be separated and calculated step by step. Knowing the acceleration components we can estimate the *circular point* of a skin cell and thus the homogeneous transformation (or DH-like parametrization) between the skin cell and the joint coordinate frame. As a skin cell can usually be moved by several different joints we

can calculate the *circular point* of a skin cell with respect to two different joints. The resulting transformations between a specific skin cell and two different joint coordinate frames can be used to calculate the inter-joint transformations (see Eq. 2). Using the D-H convention, the inter-joint transformations contain only rotations around the joint z-axis such that all the partial homogeneous transformations can be combined to generate the kinematic chain from the robot base frame to each skin cell.

$$^{Link_j}T_{Link_{j+1}} = {}^{Link_j}T_{Cell_i}\left({}^{Link_{j+1}}T^{-1}_{Cell_i}\right). \quad (2)$$

B. Parametric Dynamic Model

The next stage is to use the DH-like parameters to compute the robot Kinematic and Dynamic models (see Fig. 1 F). This process is based on an iterative Euler-Lagrange modeling which is divided into the following steps: a) Compute the relative link transformations $^{Link_j}T_{Link_{j-1}}$ using the symbolic DH-table. b) Compute the global transformation of each link with respect to the robot base ($Link_0$):

$$^{Link_j}T_{Link_0} = {}^{Link_{j-1}}T_{Link_0}\left({}^{Link_j}T_{Link_{j-1}}\right) \quad (3)$$

with $j = 2, 3, ..., n$ where n is the robot DOF. Using the transformations in Eq. (3) we can compute the global pose of each skin cell with respect to the robot base $Link_0$ as:

$$^{Cell_i}T_{Link_0} = {}^{Link_j}T_{Link_0}\left({}^{Cell_i}T_{Link_j}\right). \quad (4)$$

c) Extract the $^{Link_j}z_{Link_0}$ axis and the position vector of each joint $^{Link_j}t_{Link_0} \in \mathbb{R}^3$ from the homogeneous matrices Eq. (3). d) Compute the geometric Jacobian of each joint $^{Link_j}J_{Link_0} \in \mathbb{R}^{6\times n}$. Finally use these Jacobians to compute the symbolic matrices $M(q), C(q,\dot{q}) \in \mathbb{R}^{n\times n}$ and $G(q) \in \mathbb{R}^n$ using the definitions of the Kinetic and Potential energy in the Euler-Lagrange formulation [28]. The robot kinematic models together with the global calibration of the skin cell Eq. (3)-(4) can be used to compute the geometric Jacobian of each skin cell, named $^{Cell_i}J_{Link_0} \in \mathbb{R}^{6\times n}$.

All the stages in Sec. II and III are performed off-line and only once per robot. The obtained models and parameters are exploited to synthesize controllers and command interfaces in the following sections.

IV. Control of Robot Behaviors

We define *robot behaviors* as a collection of controllers running in parallel. In this section, we provide some examples of the controllers that can be implemented and how to integrate the interactions with the environment perceived by the skin in the main control loop of the robot. Depending on the command interface specified by the robot, a *Torque Resolver* transforms these motor commands into a specific interface type. The following subsections further elaborate these components.

A. Skin Joint Control

In order to fuse the information from the artificial skin sensors with different controllers (see Fig. 3), we need to transform the sensors signals (e.g. pre-touch and pressure) to generalized force commands. In this work, we use force

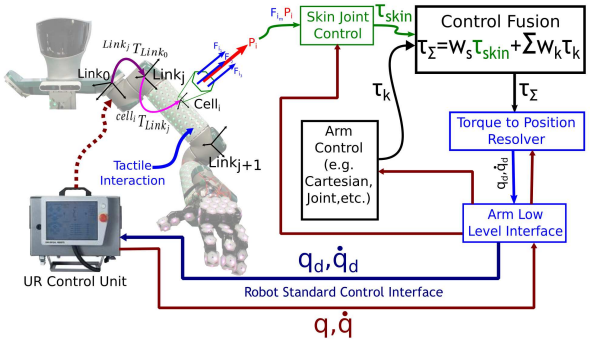

Fig. 3. Transforming skin signals into control signals.

vectors to transform tactile signals into joint torque signals. This is achieved with the following two steps.

1) Multi-modal Tactile Signals to Force Vector: Each $Cell_i$ produces a set of three *pressure signals* $f_{i_m} \in \mathbb{R}, m = 1, 2, 3$ and a single *proximity signal* $p_i \in \mathbb{R}$, see Fig. 3. The first step is to transform these signals into force vectors. By design, both the pressure-signals and the proximity signal are normal to the sensor $Cell_i$, defined by its $z-axis$. Therefore the $Cell_i$ force vectors can be constructed as follows:

$$\overline{P_i} = [0, 0, w_p p_i]^T, \ \overline{F_i} = [0, 0, w_f \sum_{m=1}^{3} f_{i_m}]^T, \quad (5)$$

where $w_p, w_f \in \mathbb{R}$ are weighting gains for the proximity and pressure signals, respectively. The above equations represent the force vectors of each signal with respect to the $Cell_i$ frame. The force vector with respect to the robot base ($Link_0$) is obtained as:

$$^{Cell_i}F_{Link_0} = {}^{Cell_i}R_{Link_0}\left(\overline{F_i} + \overline{P_i}\right) \quad (6)$$

where $^{Cell_i}F_{Link_0} \in \mathbb{R}^3$ represents the total force vector produced by the tactile signals of the $Cell_i$. The rotation matrix $^{Cell_i}R_{Link_0} \in SO(3)$ is extracted from the homogeneous transformation $^{Cell_i}T_{Link_0}$, computed with Eq. (4).

2) Force Vector to Joint Torques: In the second step, the torque $\tau_{Cell_i} \in \mathbb{R}^n$ produced by the tactile signals of each $Cell_i$ is calculated as:

$$\tau_{Cell_i} = {}^{Cell_i}J^T_{Link_0}\left({}^{Cell_i}W_{Link_0}\right) \in \mathbb{R}^n \quad (7)$$

where $^{Cell_i}W_{Link_0} = [{}^{Cell_i}F^T_{Link_0}, 0^{1\times 3}]^T \in \mathbb{R}^6$ is the wrench applied on $Cell_i$[5]. The skin joint torque $\tau_{skin} \in \mathbb{R}^n$ generated by all the *skin cells* on every patch k is computed as:

$$\tau_{skin} = \sum_{k=1}^{p}\sum_{i=1}^{s}\tau_{Cell_{k,i}} \in \mathbb{R}^n \quad (8)$$

with s as the number of skin cells in a skin patch and p as the total number of skin patches on the robot. The joint torque obtained from Eq. (8) can be fused with other controllers to produce specific robot behaviors. In our system, these are some examples of the currently implemented controllers.

[5]We set the moment on $Cell_i = 0 \in \mathbb{R}^{3\times 1}$ since it is physically impossible to apply a pure moment to an individual $Cell_i$ with respect to its own reference frame or even measure it with our skin sensors.

B. Joint Control

Based on [29] we implemented a second order sliding mode controller in the joint space defined as:

$$\tau = -K_d S_q + Y_r \Theta \in \mathbb{R}^n \tag{9}$$

where $K_{d_+} = K_{d_+}^T \in \mathbb{R}^n$ and $Y_r\Theta \in \mathbb{R}^n$ is the robot regressor. The joint error surface S_q is defined as:

$$S_q = \dot{q} - \dot{q}_r \in \mathbb{R}^n \tag{10}$$

where the joint velocity reference $\dot{q}_r \in \mathbb{R}^n$ is given as

$$\dot{q}_r = \dot{q}_d - K_p(t)\,\Delta q + S_d - K_{i_1}\int_{t_0}^{t} S_\delta d\zeta - K_{i_2}\int_{t_0}^{t} \tanh(\mu S_\delta)\, d\zeta \tag{11}$$

with $\tanh(\mu\bullet)$ as a smooth approximation for the function $sign(\bullet)$, and $\mu > 0 \in \mathbb{R}$. $K_{i_j} > 0 \in \mathbb{R}^{n\times n}$, $j = 1, 2$, and $K_p(t)$ is a time-varying proportional gain given by:

$$K_p(t) = \frac{(1+\epsilon)\,\dot{\xi}(t)}{(1-\xi(t)) + \delta} \tag{12}$$

where ϵ and δ are positive small scalars, the function $\xi(t) \in \mathbb{C}^2$ is defined as a spline-function satisfying the following constraints: $\xi(t_0) = \dot{\xi}(t_b) = 0$, with $t_b > 0$ as the convergence time. The joint error manifold S_δ is

$$S_\delta = S - S_d = (\Delta\dot{q} + K_p(t)\,\Delta q) - \left(S(t_0)\, e^{-\kappa t}\right) \tag{13}$$

with the joint position error defined as:

$$\Delta q = q - q_d \in \mathbb{R}^n \tag{14}$$

where $\Delta\dot{q} = \dot{q} - \dot{q}_d$ is the joint velocity error, where q_d and $\dot{q}_d$ stands for the desired joint position and velocity, respectively.

C. Spline Joint Control

For this controller, we use the same definition for the joint velocity reference defined in Eq. (11), where the desired joint position and velocity vectors $q_d, \dot{q}_d$ of Eq. (14) are defined as a time-varying trajectories which can be generated on-line. However, for this controller, the desired joint position is a static goal $q_g \in \mathbb{R}^n$. In this case, the controller generates a smooth trajectory from the initial joint position $q_i \in \mathbb{R}^n$ to the target one q_g. The trajectory generator is defined as:

$$q_d = a_1(q_g - q_i) + q_i,\ \ \dot{q}_d = a_2(q_g - q_i) \tag{15}$$

where $r = t/t_f$ is a time ratio between the current time t and the desired total time t_f. The coefficients a_i are defined as: $a_1 = 10r^3 - 15r^4 + 6r^5$, $a_2 = (30r^2 - 60r^3 + 30r^4)/(t_f)$ and $a_3 = (60r - 180r^2 + 120r^3)/(t_f^2)$. This function guarantees a smooth trajectory that satisfies the constraints $q_d(0) = q_i$, $q_d(t_f) = q_g$ and $\dot{q}(0) = \dot{q}(t_f) = \ddot{q}(0) = \ddot{q}(t_f) = 0 \in \mathbb{R}^n$.

D. G & D Control

The G control is a *Gravity Compensation* control, defined as $\tau = G(q) \in \mathbb{R}^n$, where $G(q)$ is the robot's gravitational torque vector. In the same manner, the D controller is a simple damping control given as $\tau = -D\dot{q}$, where $D_+ = D_+^T \in \mathbb{R}^{n\times n}$ is a *damping matrix*. The above controllers are known as standard controllers. However, the choice of joint torque as the common control output allows the implementation of more sophisticated controllers, e.g. Uncalibrated Image-Based Visual Servoing [30] or Image-Based Position-Force Control [31].

E. Control Fusion

The mapping from tactile signals into joint torques (Eq. (7)-(8)) allows the fusion of multiple controllers that use the same generalized force representation. Thus, a simple normalized *Weighted-Sum* approach to adding the contribution of each individual controller to a total joint torque output τ_Σ can be used:

$$\tau_\Sigma = w_s \tau_{skin} + \sum_{k=1}^{u} w_k \tau_k, \tag{16}$$

where u is the total number of controllers, $w_s, w_k \in \mathbb{R}$ are weighting values and τ_k is the control output of a controller defined by the user. We selected this fusion method to guarantee a deterministic behavior, even when local minimum is present. Nevertheless, a more sophisticated approach can be used in order to select an optimal combination of controls, e.g. [32]. The selection of weights for the controllers depends on the specific robot behavior that we need to generate. Some examples of the different robot behaviors are depicted in Fig. 5. The importance of the *Skin Joint Control* is to generate a compliance behavior on a non-compliant robot.

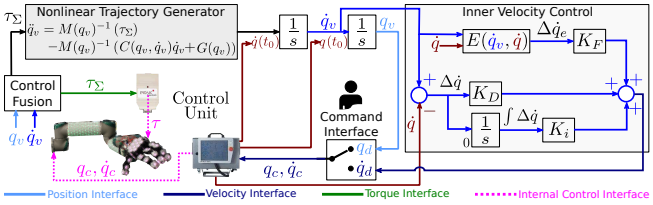

Fig. 4. Torque resolver defined by two principal modules: a) Nonlinear Trajectory Generator which produces desired trajectories based on user-defined dynamic behaviors, b) Inner Velocity Control which generates a desired joint velocity in order to compensate uncertainty in the robot parameters. $q_v, \dot{q}_v$ represent the joint position/velocity of the virtual robot (target position/velocity), $q, \dot{q}$ are the joint position/velocity of the real robot, $q_c, \dot{q}_c$ are the commanded joint position/velocities. $E(\dot{q}_v, \dot{q})$ is a joint velocity estimator.

F. Torque Resolver

The multi-modal controller in Fig. 3 is designed to provide two low-level command interfaces, either *Position/Velocity* interface, available in most of the modern industrial robots, or *Torque* interface, see Fig. 4. In the case of the *Torque* interface, we command directly τ_Σ to the control unit of the robot. In order to control robots with a *Position/Velocity* interface, we need to transform the total commanded joint control τ_Σ into desired joint positions/velocities. To this aim, we have implemented a *Torque Resolver* which uses a *Nonlinear Trajectory Generator* to produce desired joint commands. We obtain the full dynamic model to design the trajectory generator using the kinematic models of the robot in combination with the parametric dynamic model obtained in Sec. III. This parametric model allows to specify user-defined dynamic behaviors, e.g. it can increase the viscous friction, thus generating a slower step response to an external

input (e.g. tactile interaction). The trajectory generator is initialized using the joint positions/velocities of the real robot. The joint torque τ_Σ is obtained using Eq. (16). In the case of *Position* interface, the desired joint position q_v is commanded directly to the robot's Control Unit using its standard control interface. The *Velocity* interface requires a *Inner Velocity Control* to avoid drifting between the real velocity $\dot{q}$ and the virtual velocity $\dot{q}_v$. In our case, we use a *PI* velocity control with Feed-Forward estimation, where $E(\dot{q}_v, \dot{q})$ is a joint velocity estimator, e.g. [33]. This feed-forward estimator is used to compensate the uncertainties in the robot parameters. This velocity control uses two different joint error velocities: $\Delta\dot{q} = \dot{q}_v - \dot{q}$ and $\Delta\dot{q}_e = \dot{q}_v - \dot{q}_e$, where $\dot{q}_e$ is a feed-forward estimation of $\dot{q}$. The generated desired joint velocities $\dot{q}_d$ are also commanded to the Control Unit, see Fig. 4.

V. Experimental Validation

We evaluate or Multi-level control approach in our robotic platform Tactile Omni-directional Mobile Manipulator (TOMM). TOMM is composed of two industrial robot arms (UR-5[6]) covered with our artificial skin (more than 600 skin cells). The UR-5 robots are controlled using Position/Velocity command interface. The controllers and the artificial skin interface are implemented in ROS in a Workstation with Intel Core i7-4702MQ CPU @ 2.20GHz, 16Gb DDR3 RAM. For the experimental validation, we consider three different dynamic behaviors, depicted in Fig. 5. Column 1) Joint Compliant: depicts the fused controllers (1a, 1c, 1d) and the obtained joint trajectories (1b). In this experiment $q_d = q(t_0) = [161, -116, -82, -38, -58, 10]deg$ and $\dot{q}_d = 0 \in \mathbb{R}^6$. When there is an external tactile perturbation (user applies forces to the arm), e.g. $t = [5, 7]s$ (1a), the robot reacts to the tactile event and changes its position (1b), as soon as the external perturbation is removed, e.g. $t = 8s$ the robot returns to its original position smoothly. Column 2) Joint Spline: In this case the goal position of the robot is $q_g = [126, -150, -80, -34, -45, 50]deg$ the initial position is $q_i = q(t_0)$. The desired position q_d is computing using Eq. (15). In the same form 2a, 2c and 2d show the active controllers and 2b depicts the joint position behavior. At $t = 0$ the robot arm follows the spline trajectory q_d (2b), at $t = 2s$ the user interferes the trajectory and it is detected by the skin sensors (proximity and force) (2a), this produces a compliant reactive behavior which forces a change in the robot trajectory (2a). When the user is no longer generating a tactile event $t = 6s$, the robot continues with the tracking of the desired trajectory (without overshoot). The same behavior is repeated at $t = [8, 16]s$. Column 3) Kinesthetic Joint: In this behavior only the *Skin Joint Control* and the *G & D Control* are activated (3a, 3c). At the beginning, the robot is static and remains in that state until the skin sensors detect a tactile event. At $t = 5.5s$ the user interacts with the arm and produces changes in its position (generated by the Skin Control) (3a). Since only the gravity compensation and damping are active, the robot stops (due to the damping) and remains in its current position as soon as the user is no longer touching the robot. The interaction is repeated several times ($t = [11, 26]s$), generating changes reflected in the joint position of the robot. We provide a video[7] to illustrate these behaviors in our robot TOMM using the proposed approach.

VI. Conclusions

In this paper, we presented an *end-to-end* approach to transform multi-modal tactile signals into a compliant control. The approach uses our self -organizing/-calibrating artificial skin and its multi-modal sensors to automatically generate useful kinematic/dynamic models. These models are used to synthesize customized controllers for each robot. The approach has been validated in a real wheeled humanoid robot where three different behaviors were generated. The dynamic behavior can be adapted by the user allowing to change the reactive level of the robot. Thus, enabling a compliant behavior in a standard (stiff) industrial robot.

References

[1] A. D. Santis, B. Siciliano, A. D. Luca, and A. Bicchi, "An atlas of physical human–robot interaction," *Mechanism and Machine Theory*, vol. 43, no. 3, pp. 253–270, 2008.

[2] E. Dean, K. Ramirez-Amaro, F. Bergner, I. Dianov, P. Lanillos, and G. Cheng, "Robotic technologies for fast deployment of industrial robot systems," in *42nd IEEE Industrial Electronics Conference (IEEE IECON2016). [Accepted]*. IEEE, October 2016.

[3] J. Carpentier, S. Tonneau, M. Naveau, O. Stasse, and N. Mansard, "A versatile and efficient pattern generator for generalized legged locomotion," in *2016 IEEE International Conference on Robotics and Automation (ICRA)*, May 2016, pp. 3555–3561.

[4] N. Banerjee, X. Long, R. Du, F. Polido, S. Feng, C. G. Atkeson, M. Gennert, and T. Padir, "Human-supervised control of the atlas humanoid robot for traversing doors," in *IEEE-RAS Humanoid Robots, 2015 15th International Conference on*, Nov 2015, pp. 722–729.

[5] P. Mittendorfer, E. Yoshida, and G. Cheng, "Realizing whole-body tactile interactions with a self-organizing, multi-modal artificial skin on a humanoid robot," *Adv. Robotics*, vol. 1, no. 29, pp. 51–67, 2015.

[6] G. Baetz, B. Weber, M. Scheint, D. Wollherr, and M. Buss, "Dynamic contact force/torque observer: Sensor fusion for improved interaction control," *The International Journal of Robotics Research*, vol. 32, no. 4, pp. 446–457, 2013.

[7] A. Mifsud, M. Benallegue, and F. Lamiraux, "Estimation of contact forces and floating base kinematics of a humanoid robot using only inertial measurement units," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2015, pp. 3374–3379.

[8] A. D. Luca, A. Albu-Schaffer, S. Haddadin, and G. Hirzinger, "Collision detection and safe reaction with the dlr-iii lightweight manipulator arm," in *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct 2006, pp. 1623–1630.

[9] E. Magrini, F. Flacco, and A. D. Luca, "Estimation of contact forces using a virtual force sensor," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 2126–2133.

[10] ——, "Control of generalized contact motion and force in physical human-robot interaction," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, May 2015, pp. 2298–2304.

[11] R. S. Dahiya, G. Metta, M. Valle, and G. Sandini, "Tactile sensing;from humans to humanoids," *IEEE Transactions on Robotics*, vol. 26, no. 1, pp. 1–20, Feb 2010.

[12] G. Cannata, M. Maggiali, G. Metta, and G. Sandini, "An embedded artificial skin for humanoid robots," in *Multisensor Fusion and Integration for Intelligent Systems, 2008. MFI 2008. IEEE International Conference on*, Aug 2008, pp. 434–438.

[6] http://www.universal-robots.com/products/ur5-robot

[7] https://youtu.be/H66vDX3wAZQ

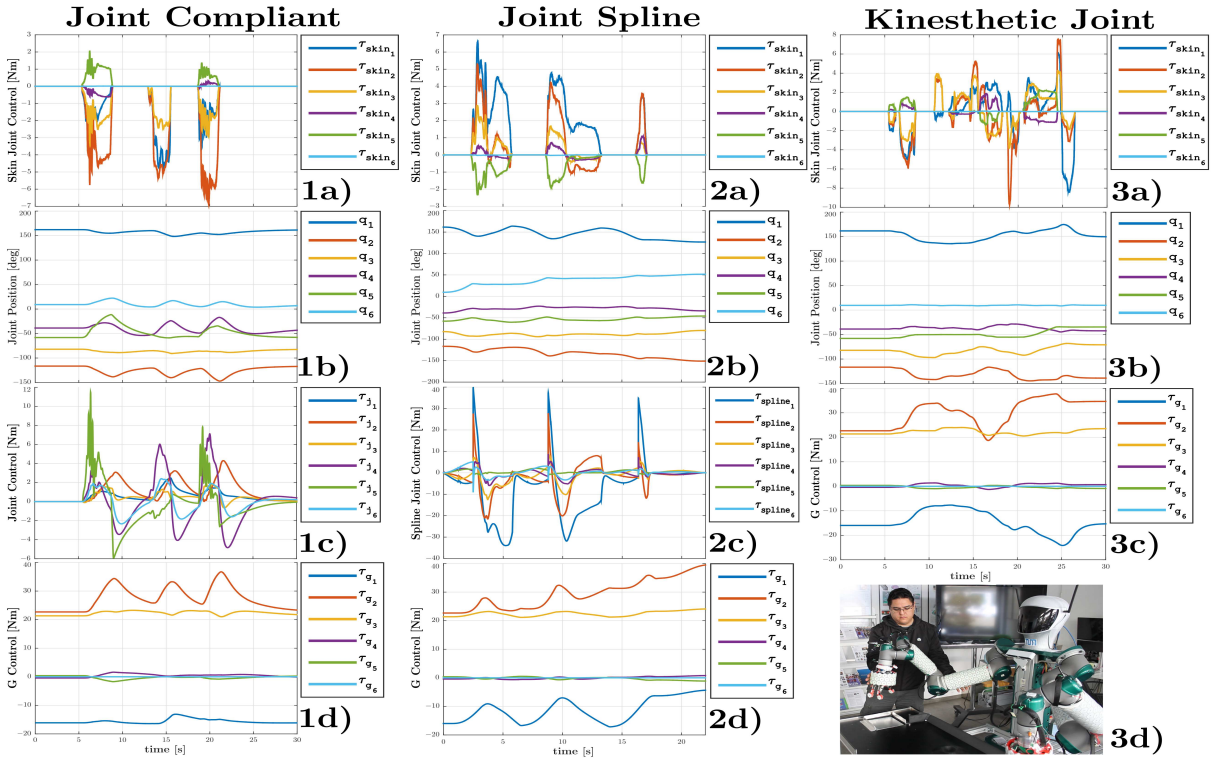

Fig. 5. The first row depicts the joint torque generated by the tactile interactions (proximity and force signals). The second row shows the changes in the joint positions due to the interactions. The last rows illustrate the controllers used to generate the different dynamic behaviors. 3d) shows the robot setup.

[13] V. Duchaine, N. Lauzier, M. Baril, M. A. Lacasse, and C. Gosselin, "A flexible robot skin for safe physical human robot interaction," in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, May 2009, pp. 3676–3681.

[14] M. W. Strohmayr, H. Wrn, and G. Hirzinger, "The dlr artificial skin step i: Uniting sensitivity and collision tolerance," in *Robotics and Automation (ICRA), 2013 IEEE International Conference on*, May 2013, pp. 1012–1018.

[15] Y. Ohmura, Y. Kuniyoshi, and A. Nagakubo, "Conformable and scalable tactile sensor skin for curved surfaces," in *Proceedings 2006 IEEE International Conference on Robotics and Automation, 2006. ICRA 2006.*, May 2006, pp. 1348–1353.

[16] A. Kolker, M. Jokesch, and U. Thomas, "An optical tactile sensor for measuring force values and directions for several soft and rigid contacts," in *Proceedings of ISR 2016: 47st International Symposium on Robotics*, June 2016, pp. 1–6.

[17] A. Cirillo, F. Ficuciello, C. Natale, S. Pirozzi, and L. Villani, "A conformable force/tactile skin for physical human-robot interaction," *IEEE Robot. and Autom. Letters*, vol. 1, no. 1, pp. 41–48, Jan 2016.

[18] T. Asfour, K. Regenstein, P. Azad, J. Schrder, and R. Dillmann, "Armar-iii: A humanoid platform for perception-action," in *Proceedings of 2nd Int. Workshop on Human-Centered Robotic Systems*, 2006.

[19] R. H. Andersen, T. Solund, and J. Hallam, "Definition and Initial Case-Based Evaluation of Hardware-Independent Robot Skills for Industrial Robotic Co-Workers," in *ISR/Robotik 2014; 41st International Symposium on Robotics.*, June 2014, pp. 1–7.

[20] J. F. Gorostiza, R. Barber, A. M. Khamis, M. Malfaz, R. Pacheco, R. Rivas, A. Corrales, E. Delgado, and M. A. Salichs, "Multimodal human-robot interaction framework for a personal robot," in *ROMAN 2006 - The 15th IEEE International Symposium on Robot and Human Interactive Communication*, Sept 2006, pp. 39–44.

[21] P. Mittendorfer and G. Cheng, "Humanoid Multimodal Tactile-Sensing Modules." *IEEE Trans. Robotics*, vol. 27, no. 3, pp. 401–410, 2011.

[22] F. Bergner, P. Mittendorfer, E. C. Dean-Leon, and G. Cheng, "Event-based signaling for reducing required data rates and processing power in a large-scale artificial robotic skin." in *IROS*. IEEE, 2015, pp. 2124–2129.

[23] F. Bergner, E. C. Dean-Leon, and G. Cheng, "Event-based signaling for large-scale artificial robotic skin - realization and performance evaluation," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 2016.

[24] P. Mittendorfer and G. Cheng, "3D surface reconstruction for robotic body parts with artificial skins." in *IROS*. IEEE, 2012, pp. 4505–4510.

[25] P. Mittendorfer, E. Dean, and G. Cheng, "3D spatial self-organization of a modular artificial skin," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 3969–3974.

[26] ——, "Automatic robot kinematic modeling with a modular artificial skin," in *2014 IEEE-RAS International Conference on Humanoid Robots*, Nov 2014, pp. 749–754.

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

[28] E. Dean-Leon, S. Nair, and A. Knoll, "User friendly Matlab-toolbox for symbolic robot dynamic modeling used for control design," in *IEEE International Conference on ROBIO*, Dec 2012, pp. 2181–2188.

[29] V. Parra-Vega, "Chattering-free dynamical tbg adaptive sliding mode control of robot arms with dynamic friction for tracking in finite-time," in *Robotics and Automation, 2001. Proceedings 2001 ICRA. IEEE International Conference on*, vol. 4, 2001, pp. 3471–3476 vol.4.

[30] E. C. Dean-Leon, V. Parra-Vega, A. Espinosa-Romero, and J. Fierro, "Dynamical image-based pid uncalibrated visual servoing with fixed camera for tracking of planar robots with a heuristical predictor," in *Industrial Informatics, 2004. INDIN '04. 2004 2nd IEEE International Conference on*, June 2004, pp. 339–345.

[31] E. C. Dean-Leon, L. G. Garcia-Valdovinos, V. Parra-Vega, and A. Espinosa-Romero, "Uncalibrated image-based position-force adaptive visual servoing for constrained robots under dynamic friction uncertainties," in *2005 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Aug 2005, pp. 2983–2990.

[32] E. Aertbeliën and J. D. Schutter, "eTaSL/eTC: A constraint-based task specification language and robot controller using expression graphs," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Sept 2014, pp. 1540–1546.

[33] M. Barut, S. Bogosyan, and M. Gokasan, "Speed-sensorless estimation for induction motors using extended kalman filters," *IEEE Transactions on Industrial Electronics*, vol. 54, no. 1, pp. 272–280, Feb 2007.