Open-loop Self-calibration of Articulated Robots with Artificial Skins

Philipp Mittendorfer and Gordon Cheng

***Abstract*— In this paper, we present a twofold, open-loop method to explore, model and calibrate articulated robots equipped with artificial skin. We do so, using a 3-axis accelerometer per artificial sensor skin unit (SU) and special excitation pattern on every actuated degree of freedom (DoF) of the robotic joints. The first algorithm extracts the kinematic dependencies in between segments, equipped with artificial skin units, and joints, featuring one or multiple rotatory DoFs. A second algorithm uses this structural knowledge to automatically build and estimate kinematic models in between a static reference and an end effector segment. We show experimental results for the structural exploration with a KUKA light weight robotic arm equipped with our own SU prototypes. Additional simulation results, supporting our approach on estimating the kinematic parameters of the robot, are also presented.**

I. INTRODUCTION

*1) **Motivation:*** In comparison to a robot purely relying on joint information (position and/or force), a robot equipped with sensitive skin has a much richer information set. It can not only classify different touch events, e.g. temperature changes or impacts, but also accurately localize the origin of the sensations and act meaningfully. In order to do so, integrative control algorithms need information on the relative location and orientation of every sensor and actuator. Manually providing this information is infeasible, due to the possible high number of sensors and actuators. Such a process would also be erroneous and exclude non-specialist users. To overcome this, automated calibration routines can be realized on robots equipped with artificial sensor skins. The robot can utilize its own sensors and actuators to explore, model and calibrate itself. This is especially useful when a sensory system is not designed for a single robot, but applicable across multiple robotic platforms. The system should then only use general physical means for its calibration, instead of robot specific a-priori knowledge, and apply simple constraints that a non-specialist user can understand and establish. An easy re-exploration mechanism is also useful to accommodate hardware failure or hard resets with autonomous robots. Here we are motivated to handle the available sensors and actuators in a single approach, to cope with sensor, actuator and structural changes at once. We are also specifically interested in a contact free, open loop method, as this provides a fast, safe and unambiguous way to generate the necessary actuator to sensor excitation.

*2) **Related Works:*** In [1] we introduced a new artificial skin which we had to manually provide with information in order to react on touch stimulation. We then presented a method [2] to automatically acquire and store local reaction behaviors for a varying number of distributed sensors and actuators. This approach is only feasible with a limited number of actuators and/or limited workspace, as it has to a-priori explore every pose it needs to react in, or utilize a 'similar' one from memory. In this paper, we focus on the next step, building and estimating a more general model.

Philipp Mittendorfer and Gordon Cheng are with the Institute for Cognitive Systems, Technische Universität München, Karlstrasse 45/II, 80333 Munich, Germany `Email: see http://www.ics.ei.tum.de/`

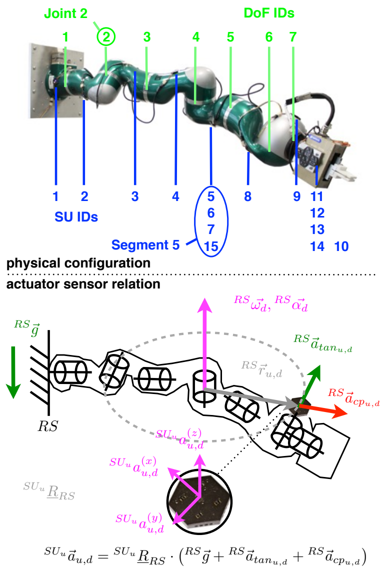

Joint 2
DoF IDs
1
2
3
4
5
6
7
1
2
3
4
5
6
7
15
8
9
11
12
13
14
10
SU IDs
Segment 5
physical configuration
actuator sensor relation
$^{RS}\vec{g}$
RS
$^{RS}\vec{\omega}_d, {}^{RS}\vec{\alpha}_d$
$^{RS}\vec{a}_{tan_{u,d}}$
$^{RS}\vec{r}_{u,d}$
$^{RS}\vec{a}_{cp_{u,d}}$
$^{SU_u}a^{(z)}_{u,d}$
$^{SU_u}a^{(x)}_{u,d}$
$^{SU_u}a^{(y)}_{u,d}$
$^{SU_u}\underline{R}_{RS}$
$^{SU_u}\vec{a}_{u,d} = {}^{SU_u}\underline{R}_{RS} \cdot ({}^{RS}\vec{g} + {}^{RS}\vec{a}_{tan_{u,d}} + {}^{RS}\vec{a}_{cp_{u,d}})$

Fig. 1. Problem set - Unknown physical distribution of artificial sensor skin units (SU) and degrees of freedom (DoF) with an articulated robotic arm. Accelerations acting on a SU when a single, rotatory DoF is being excited, while one reference segment (RS) is kept static.

Related approaches, such as motion capture systems, have been largely used to automatically build and estimate kinematic models. Most systems track active [3] or passive [4] visual markers/features. Such approaches can be extremely fast and can robustly adapt to structural changes [5]. Magnetic motion capture systems perform better when occlusion is a problem but have problems when utilized on metal frame robots [6]. All of these systems rely on a globally accurate, calibrated external sensor system and the availability of robust markers/features to track. Wearable motion capture systems in contrast, like the XSens MVN [7], come with a specific underlying body model of which several parameters have to be given manually and others are found, matching known postures. Classic robot calibration methods, where the robot model is known, but subject to imperfections in fabrication and/or wear and tear, split into closed and open loop approaches [8]. Most open loop approaches require a calibrated external metrology system, of which the noise plays a significant role [8]. Closed loop measurements profit from the accuracy of the fixation point to the environment or another manipulator. Their precision stands and falls with this fixation accuracy. With over constrained closed loops additional sensors have to be provided, like a force torque sensor at the endpoint in [9].

Different methods of sensor skin based self-exploratory learning have been reported in simulations [10], as well as practical implementations [11]. Visual information is commonly used, along with a strongly constrained and time consuming probing of each individual sensor.

A special form of open loop calibration, which is closely related to our work, has been presented in [12]. Canepa *et al.* use a 3-axis accelerometer on the end effector to conduct a circle point analysis [13]. However, their approach is based on a first order integration of the accelerometer data and large movements. This requires high integrity of the accelerometer data, which can normally only be achieved with large and expensive high grade devices. Here, we make use of many small, low-cost devices distributed over the robot, in order to estimate both, structure and kinematic parameters without any integration of the signals.

We concur with a statement of Hoffmann *et al.* in [14] that the work on models in robotics is heavily biased toward manipulator arms observed by a camera, which is lacking the integration of multiple modalities as demonstrated by biological agents. In contrast to biological agents using implicit models, we focus here on an explicit model. Explicit models according to Hoffmann *et al.* are easier to debug, to link to common control theory and provide valid data also in previously unseen situations.

3) **Our Approach:** In this paper, we utilize one of the multiple sensing modalities located on every unit cell that our artificial skin consists of – a three-axis accelerometer. Given that a central segment of the robot can remain static during exploration, it is possible to directly evaluate motion pattern of the robot with these distributed accelerometers.

Algorithm 1 Structural Exploration (whole robot)

```
1: Detect number of available SUs (U) and DoFs (D)
2: Move one DoF a time, sampling all SU gravity vectors
3: Create activity matrix $\underline{A}^{U\times D}$, thresholding the samples
4: Merge SUs with similar row vectors in $\underline{A}$ to segments
5: Merge DoFs with similar column vectors in $\underline{A}$ to joints
6: Extract connection sequences from row vectors
7: Detect the reference segment (RS)
8: Determine available end-effector segments
9: Assemble a serial sequence for a robotic limb
```

The first step of our algorithm (see Alg.1) analyses the structural dependency of the robot. By applying excitation pattern to all degrees of freedom (DoFs), one after the other, the robot detects which sensor unit (SU) reacts towards which DoF. Given that each segment of the robot is equipped with at least one SU, our new method is then able to merge DoFs to joints, SUs to segments and discriminate the kinematic dependency of joints and segments. This method provides us with information on how many body parts there are, which SUs belong to which body part, how joints, combining one or more DoFs, connect the body parts and which body parts are end effectors, located at the end of a serial chain.

Algorithm 2 Kinematic Estimation (selected limb)

```
1: Instantiate serial chain model of selected limb (Alg.1)
2: Generate training data in (P) randomized poses
3: Optimize parameters of model to best fit training set
```

In the second step, our algorithm (see Alg.2) uses this structural knowledge to build a kinematic model for the DoFs and SUs located on the same limb. What we refer to as robotic limb is the connection of joints and segments in between the stationary segment and one end effector segment. The robot then generates a training data set for every limb, moving and sampling the related DoFs and SUs. Every training set contains measurements of the static gravity vector, as well as the dynamic acceleration on isolated DoF motion pattern, for multiple random DoF poses. A global optimization algorithm then minimizes the error in between readings generated by the kinematic model and the training data set.

The advantage of our approach is that we require only minimal, robot independent a-priori knowledge – a physical model relating actuation and sensation. Only few constraints have to be taken into account: one SU per segment, one static segment, no closed loop is required during calibration. Using an internal observer we are not limited by occlusion or any external components. Our method is able to acquire the information in a short time, which is especially useful for ongoing autonomous re-calibration.

II. SYSTEM DESCRIPTION

A. Robotic System

In this section, we first describe our artificial sensor skin. We then summarize the constraints our method assumes. Finally, we describe the physical effects every accelerometer is exposed to, given these constraints (see Fig. 1).

1) **Artificial Skin:** What we refer to as HEX-O-SKIN sensor unit, is a rigid, hexagonal shaped printed circuit board (PCB), with multiple sensors on the front side, emulating human cues of touch, and a local controller on the back side, converting, processing and forwarding sensor signals in an arbitrary configuration of units (refer to [1]).

In this paper, we make use of the 3-axis accelerometer available on every SU. Currently this is a BMA150 from BOSCH with 10 bit resolution, set at $\pm 2g$ range and a sample rate of 1 kHz. The alignment of the accelerometer coordinate system towards all other modalities, located on the same SU, is known and static. Consequently, it is sufficient to estimate the kinematic relations towards the accelerometer coordinate systems.

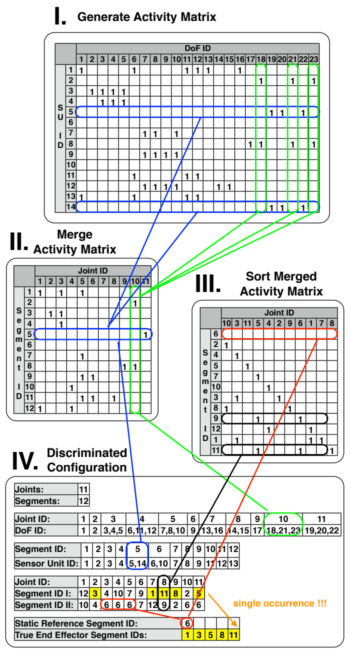

I. Generate Activity Matrix
DoF ID
SU ID
II. Merge Activity Matrix
Joint ID
Segment ID
III. Sort Merged Activity Matrix
IV. Discriminated Configuration
Joints:
Segments:
Joint ID:
DoF ID:
Segment ID:
Sensor Unit ID:
Segment ID I:
Segment ID II:
single occurrence !!!
Static Reference Segment ID:
True End Effector Segment IDs:

Fig. 2. Steps of the structure exploration algorithm for an activity matrix representing a "simulated" humanoid robot (our test case). Available SUs and DoFs, as well as detected body segments and joints are automatically labeled with consecutive identification numbers (IDs).

2) **Constraints:** First of all we assume rigid body kinematics, joints with one or more DoFs are connected by non-deformable segments. During the whole calibration process one reference segment on the robot has to remain static in world coordinates. With a robotic arm or wheeled robotic platform this constraint is naturally given by the base or chassis. For a humanoid robot the torso is considered best, as it is a casual point to fix a humanoid, is located close to the center of mass and is a relatively central point in the kinematic tree. Choosing the static segment has a large influence on the outcome of our method, as it serves as root to the structural exploration as well as the kinematic models. It is necessary that every degree of freedom (DoF) can be actuated freely. Impacts with the DoF limits and any objects surrounding the robot would interfere with the motion generation and sensory sampling. Closed loop and under-actuated kinematics can not be handled with our method. We also assume that we can control the robot with velocity commands and the position sensors are calibrated. The robot needs to be equipped with at least one SU per segment. Only then is it possible to unambiguously discriminate the sequence of joints and segments of the robot's kinematic tree. So far the kinematic model estimation only features joints with a single DoF. Multi-DoF joints in robots can be built either by a close sequence of DoFs or a complex driving mechanism. Identifying and parameterizing these is part of our future work.

3) **Sensor Actuator Relation:** Neglecting skin deformations, every SU follows the acceleration of its mounting point. Given a static reference segment like in Fig. 1, the acceleration of all mounting points can be directly controlled by actuating the related DoFs. A change in velocity ($\frac{d}{dt}{}^{RS}\omega_d(t) = {}^{RS}\alpha_d(t)$) of a revolute DoF ($d$) has a direct influence on the acceleration of the SU (u), which composes from three different effects:

(a) The tangential acceleration (${}^{RS}\vec{a}_{tan_{u,d}}$), which is dependent on the rotatory acceleration of the DoF (${}^{RS}\vec{\alpha}_d$) and the vector (${}^{RS}\vec{r}_{u,d}$), in between the DoF axis (d) and the mounting point of SU (u):

$$ {}^{RS}\vec{a}_{tan_{u,d}} = {}^{RS}\vec{\alpha}_d \times {}^{RS}\vec{r}_{u,d} \tag{1} $$

(b) The centripetal acceleration ${}^{RS}\vec{a}_{cp_{u,d}}$, which is dependent on the angular velocity ${}^{RS}\vec{\omega}_d$ as well as the vector ${}^{RS}\vec{r}_{u,d}$:

$$ {}^{RS}\vec{a}_{cp_{u,d}} = {}^{RS}\vec{\omega}_d \times \left({}^{RS}\vec{\omega}_d \times {}^{RS}\vec{r}_{u,d}\right) \tag{2} $$

(c) And the gravity vector ${}^{RS}\vec{g}$.

An accelerometer does not distinguish, but senses all effects at the same time, in its local coordinate system:

$$ {}^{SU_u}\vec{a}_{u,d} = {}^{SU_u}\underline{R}_{RS} \cdot \left({}^{RS}\vec{g} + {}^{RS}\vec{a}_{tan_{u,d}} + {}^{RS}\vec{a}_{cp_{u,d}}\right) \tag{3} $$

The rotation matrix ${}^{SU_u}\underline{R}_{RS}$ in between the static reference frame and the SU, as well as the vector ${}^{RS}\vec{r}_{u,d}$, are dependent on the unknown kinematics of the robot.

B. Structural Exploration

1) **Activity Matrix Generation:** What we refer to as activity matrix is a binary matrix with the following entries: 1) a '1' when a SU can be moved by a DoF during calibration; 2) a '0' when no or only minor motion is detected. In II-A.3 we summarized the three effects we are able to measure with the accelerometer on moving a rotatory DoF. Both dynamic effects are dependent on the radial distance (${}^{RS}\vec{r}_{u,d}$). When the absolute of this vector is close to zero, both effects vanish. Dynamic motions also induce other sources of noise, like vibrations on the robot and coupling effects in between DoFs. These would make the separation in between the two binary states a very difficult task. However, the correct generation of the activity matrix is crucial to the structural exploration as well as the kinematic estimation that builds on it. This is why the third static effect is utilized. Changing a DoF from one to another position changes the orientation of all related segments in the same way (rigid body kinematics). It thus also rotates the gravity vector (${}^{SU_u}\vec{g}$) in accelerometer coordinates in the same way. Fixed thresholds can thus be applied for every SU. Since the gravity vector can be measured statically in both poses, no additional care has to be taken regarding dynamic problems like DoF coupling or vibrations on the robot. Problems arise when some of the DoF axes are aligned close to or exactly vertical, as the gravity vector then only rotates around its own axis and the accelerometer values do not change. An attribute of the activity matrix can be used to prevent this case. Multiple activity matrices of the same robot can be easily combined by an element wise logical 'or' function. The most secure approach to generate two complementary activity matrices is to rotate the static segment of the robot 90 degrees around one of the horizontal axes. This ensures that no rotation axis is aligned close to vertical with both matrices and a matrix entry is falsely set '0'.

2) **Segment and Joint Merging:** SUs that are located on the same segment encounter the same 'activity' (refer to Section II-B.1) and should thus ideally have identical row vectors in the activity matrix. Consequently, all identical row vectors of the activity matrix can be merged into segments (see Fig. 2). A similar principle applies to DoFs, which are located closely – for example the three DoFs in a shoulder joint. DoFs that are located closely have identical column vectors in the activity matrix. Consequently, all identical column vectors of the activity matrix can be merged into joints. This finally leads to a merged activity matrix, setting joint and segment activities into a context. The number of rows gives the number of detected segments while the number of columns the number of joints. Here the number of segments must always be one higher than the number of joints. This is natural for a tree like robotic structure, where an additional joint always connects an already existing body segment with a new one.

3) **Joint Segment Connectivity:** The merged activity matrix can now to be sorted to a lower triangular form. Only a strictly lower triangular form, the secondary diagonal ($l_{(n+1)(n)}, n \in \mathbb{N}$) must only contain '1' as entries and the dimension of the matrix must be ($(n) \times (n+1)$), provides the necessary input for the following algorithm step. If this should not be the case, one of the given constraints has been violated (refer to II-A.2). This could be for example a missing static reference segment or end effector segments without SUs, leading to (super-) diagonal elements. The algorithm now progresses along the secondary diagonal ($l_{(n+1)(n)}, n \in \mathbb{N}$) – going from low to high indices. While stepping through the merged and sorted activity matrix (see Fig. 2) the algorithm searches rows that are identical except for the current secondary diagonal element. This current diagonal element is in fact a joint, related to the current column index, which connects the two segments related to the row indices.

4) **Segments and Limbs:** The joint segment connectivity extracted in Section II-B.3 represents a hierarchical kinematic tree. The static reference segment (root) of the robot is the null row vector of the merged activity matrix (see Fig.2). 'True' end effector segments (leaves), like the finger tips of a humanoid, exactly connect to a single joint. Segments that connect to more than two joints, like the palm of a humanoid, are 'false' end effectors (inner nodes) and could be used as reference segments for sub-limbs, like the fingers attached to the palm of a humanoid hand. What we actually refer to as limb, is a sequence of segments connected by joints, starting from a reference segment and ending at a 'true' or 'false' end effector. With the KUKA light weight robotic arm the kinematic tree is trivial, featuring a single limb (one branch and one leaf) (see Fig.1).

C. Kinematic Estimation

In this section, we describe how we automatically build and estimate kinematic models for robotic limbs (refer to Section II-B.4). Due to the explicit model and the type of sensor we are using, a 3-axis accelerometer, in comparison to the result we wish to achieve, a kinematic model, we need to provide additional information to the system. Here, we provide additional information in the form of an error function, part of which is an explicit serial chain model. With a global optimization algorithm we then minimize the error in between a training data set and the predictions from the parameterized model.

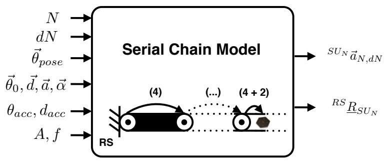

N
dN
$\vec{\theta}_{pose}$
$\vec{\theta}_0, \vec{d}, \vec{a}, \vec{\alpha}$
θ_{acc}, d_{acc}
A, f
Serial Chain Model
(4)
(...)
(4 + 2)
RS
${}^{SU_N}\vec{a}_{N,dN}$
${}^{RS}\underline{R}_{SU_N}$

Fig. 3. I/O Schema of the serial chain model.

1) **Serial Chain Model:** The serial chain model (see Fig. 3) describes the sensor/actuator relationship in between the static reference segment and one accelerometer coordinate system on the (N_{th}) segment within the serial chain of a robotic limb. Here we use Denavit-Hartenberg (DH) parameters to specify the transformations in between DoF axes along the way from the reference, as well as the last transformation to the accelerometer coordinate system. The serial chain model always refers to the static segment, as it is the only known (static) reference point in the system. DH parameters represent axis-axis transformations with a minimal number of four parameters (θ, d, a, α), which is especially useful when parameters need to be reduced for a global optimization algorithm (refer to Section II-C.4). With only four parameters it is however not possible to represent the free position and orientation of the accelerometer coordinate system. Also with other representations, e.g. screw theory – using 4 independent plucker vector components along with 1 rotation angle and 1 linear slide, 6 parameters are necessary to describe a free translation and rotation. We therefore introduced two additional parameters, a rotation (θ_{acc}) and translation (d_{acc}) along the z-axis for the last transformation in between a virtual DoF axis (accelerometer z-axis) and the free accelerometer coordinate system. Our model takes a variable array of (N) DH parameters ($\vec{\theta}_0, \vec{d}, \vec{a}, \vec{\alpha}$), the two additional parameters (θ_{acc} and d_{acc}) and the partial robotic limb pose ($\vec{\theta}_{pose}$) as input. From this input it assembles a transformation matrix in between the reference and accelerometer coordinate system:

$$ {}^{SU_N}\underline{T}_{RS} = {}^{SU_N}\underline{T}_{DoF_{N+1}} \cdot \left(\prod_{d=1}^{N} {}^{DoF_{(d+1)}}\underline{T}_{DoF_{(d)}} \right) {}^{DoF_1}\underline{T}_{RS} \quad (4) $$

The transformation (${}^{DoF_1}\underline{T}_{RS}$) in between the static reference coordinate system and the first DoF axis can not be found (refer to Section II-C.5). Here we initialize it with the identity matrix ($\underline{I}$). The following matrices are DH transformations in between ($N+1$) consecutive DoFs, including the final virtual DoF axis aligned with the accelerometer z-axis:

$$^{DoF_{(d+1)}}\underline{T}_{DoF_{(d)}} = \begin{pmatrix} c\theta_d & -s\theta_d c\alpha_d & s\theta_d sin\alpha_d & a_d c\theta_d \\ s\theta_d & c\theta_d c\alpha_d & -c\theta_d sin\alpha_d & a_d s\theta_d \\ 0 & s\alpha_d & c\alpha_d & d_d \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (5)$$

The parameter (θ_d) hereby composes of the DH offset (θ_{0_d}), the DoF position (θ_{pose_d}) and a motion pattern (θ_{patt_d}):

$$\theta_d = \theta_{0_d} + \theta_{pose_d} + \theta_{patt_d} \quad (6)$$

The last transformation enables the free placement and orientation of the (N_{th}) accelerometer frame relative to the virtual DoF axis:

$$^{SU_N}\underline{T}_{DoF_N} = \begin{pmatrix} c\theta_{acc} & -s\theta_{acc} & 0 & 0 \\ s\theta_{acc} & c\theta_{acc} & 0 & 0 \\ 0 & 0 & 1 & d_{acc} \\ 0 & 0 & 0 & 1 \end{pmatrix} \quad (7)$$

The serial chain model also contains a model of the motion pattern (see Section II-C.2), we excite on every DoF during exploration. Here we can control the amplitude (A) and frequency (f), as well as the DoF (dN) the pattern is generated on. One output of the model is the rotation matrix ($^{RS}\underline{R}_{SU_N}$) in between the accelerometer coordinate system and the static reference segment for the current pose ($\vec{\theta}_{pose}$), which is the transposed rotation matrix ($^{SU_N}\underline{R}_{RS}$), contained in ($^{SU_N}\underline{T}_{RS}$). The second output is the maximum acceleration ($^{SU_N}\vec{a}_{N,dN}$) of the accelerometer on the (N_{th}) segment, measured in accelerometer coordinates, due to an exploration pattern on the selected DoF (dN). The model calculates this value, rotating the second numerical differentiation of the accelerometer position ($^{RS}\vec{p}_{N,dN}(t)$) in ($^{SU_N}\underline{T}_{RS}$), in reference coordinates, back to accelerometer coordinates:

$$^{SU_N}\vec{a}_{N,dN} = {}^{SU_N}\underline{R}_{RS} \cdot {}^{RS}\vec{a}_{N,dN} \quad (8)$$

$$^{SU_N}\vec{a}_{N,dN} = \left({}^{RS}\vec{p}_{N,dN}(h) + {}^{RS}\vec{p}_{N,dN}(-h) - 2 \cdot {}^{RS}\vec{p}_{N,dN}(0)\right)/h^2 \quad (9)$$

The value is taken around the expected maximum acceleration time point ($t = 0$) of the excitation pattern (refer to II-C.2) with a differentiation time step of $h = 1/(1000 \cdot f)$.

2) **Automated Training Data Generation:** In order to find significant model parameters it is necessary to generate a descriptive set of measurements, in a number (P) of different poses ($\vec{\theta}_{pose_p}$) of the (D) DoFs of a robotic limb. We propose to randomly generate the DoF positions for every pose within the rotatory range $[-\pi, \pi]$. With a real robot the available workspace is of course constrained by self-collision and motor range. As described in Section II-A.3 we can measure static and dynamic effects with every accelerometer in the current pose (p). Here we statically sample the gravity vector ($^{SU_u}\vec{g}_{u,p}$) of every accelerometer (u) in every pose (p). In order to measure a dynamic effect it is necessary for the robot to apply isolated motion pattern to all of its DoFs (d). It is important to only move one DoF at a time, in order to be able to directly separate the information. Since we use an accelerometer, the robot must generate DoF motion pattern ($\theta_{patt_d}(t)$) that the sensor can measure. Here we optimized the generation towards the tangential acceleration ($^{SU_u}\vec{a}_{tan_{u,d}}$) which is directly dependent on the angular acceleration ($\alpha_{patt_d}(t)$). The centripetal acceleration ($^{SU_u}\vec{a}_{cp_{u,d}}$) in contrast asks for a high angular velocity ($\omega_{patt_d}(t)$). With a natural limit in angular acceleration, the robot can only achieve higher velocities after some time, leading to dangerous high speed motions with large angular range ($\triangle\varphi_{patt_d}$). In order to maximize the tangential acceleration, the angular acceleration ($\alpha_{patt_d}(t)$) has to be high. To avoid oscillations, it is advisable to smooth the commanded angular velocity ($\omega_{patt_d}(t)$) as well as the induced acceleration ($\alpha_{patt_d}(t)$) and jerk ($\frac{d}{dt}\alpha_{patt_d}(t)$). At the same time we need to return to the same DoF position ($\theta_{patt_d}(T)$), once the exploration pattern on the particular DoF (d) stops and is shifted to the next DoF ($d+1$). One possible DoF pattern ($\theta_{patt_d}(t)$) that fulfills all requirements is a sine wave with the first and second derivatives:

$$\theta_{patt_d}(t) = \frac{A}{2\pi f}\left(1 - cos(2\pi f t)\right) \quad (10)$$

$$\omega_{patt_d}(t) = A sin(2\pi f t) \quad (11)$$

$$\alpha_{patt_d}(t) = 2\pi f A cos(2\pi f t) \quad (12)$$

These equations give us a guideline how to dimension the pattern parameters (f and A). The selection of (A) is limited by the maximum DoF velocity of the robot. ($2\pi f A$) has to be lower than the maximum DoF acceleration and below a value that shows excessive coupling between the DoFs due to the distributed mass. ($\frac{A}{2\pi f}$) has to be small enough to be able to neglect the influence of the rotating gravity vector. Still ($2\pi f A$), has to be sufficiently large so that the effect of a DoF pattern towards the measurement of the accelerometer ($^{SU_u}\vec{a}_{u,d}$) stands out from the sensor noise. Since endless sinusoidal DoF pattern would not allow to switch the pattern between DoFs, we additionally need a windowing function ($F(t)$). The actual commanded motor pattern ($\theta_{patt_{com_d}}(t)$) thus is:

$$\theta_{patt_{com_d}}(t) = \theta_{patt_d}(t) \cdot F(t) \quad (13)$$

Actually this commanded pattern and the related dervivatives have to meet the desired constraints with a real robot. In this paper we focus on simulated measurements generated by the serial chain model itself. Neglecting the influence of the window function, the simulator numerically provides the maximum of the dynamic acceleration ($^{SU_u}\vec{a}_{dyn_{u,d,p}}$) for every SU ($u$), in every pose ($p$) for a pattern ($\theta_{patt_d}(t)$) on every DoF ($d$) at ($t = 0$).

3) **Error Function:** The error function has access to the training data (refer to Section II-C.2) as well as the serial chain model (refer to Section II-C.1). On every call of the error function, the optimizer passes a current set of serial chain model parameters (refer to Section II-C.1) for a partial chain of ($N \leq D$) DoFs. The error function then evaluates the serial chain model in every partial pose ($\vec{\theta}_{pose_p}$) of the (P) poses of the training set. We only evaluate and compare acceleration values on DoF pattern for up to three DoFs directly before the accelerometer (refer to Section II-C.4). Finally, the error function returns a positive, single dimensional total error (e_T) which builds from two parts:

$$e_T = e_1 + e_2 \ , \quad e_T \geq 0 \tag{14}$$

The first part (e_1) is a measure of deviation of the (P) gravity vectors (${}^{SU_N}\vec{g}_{N,p}$) rotated back into the static reference coordinates by the rotation matrix (${}^{RS}\underline{R}^{mod,p}_{SU_N}$) of the current serial chain model:

$$e_1 = \sum_{p=1}^{P} |{}^{RS}\vec{g}_{N,p} - \frac{1}{P}\sum_{p=1}^{P} {}^{RS}\vec{g}_{N,p}|^2 \tag{15}$$

$${}^{RS}\vec{g}_{N,p} = {}^{RS}\underline{R}^{mod,p}_{SU_N} \cdot {}^{SU_N}\vec{g}_{N,p} \tag{16}$$

The second part (e_2) is a mismatch in between the accelerations (${}^{SU_N}\vec{a}^{\,mod}_{dyn_{N,d,p}}$) calculated by the serial chain model with motion pattern on the last three DoFs, in comparison to the accelerations (${}^{SU_N}\vec{a}^{\,train}_{dyn_{N,d,p}}$) sampled during training:

$$e_2 = \sum_{p=1}^{P} \sum_{\substack{d>0 \\ d=N-3}}^{N} |{}^{SU_N}\vec{a}^{\,mod}_{dyn_{N,d,p}} - {}^{SU_N}\vec{a}^{\,train}_{dyn_{N,d,p}}|^2 \tag{17}$$

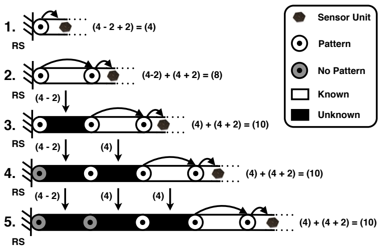


Fig. 4. Progressive estimation of serial chain parameters from static reference to end effector segment, showing the number of re-utilized and newly estimated parameters, as well as the DoFs to be excited in each step.

4) **Parameter Optimization:** The parameter optimization algorithm minimizes the value (e_T) of the error function (refer to Section II-C.3) by tuning (D) DH parameter sets ($\vec{\theta}_0, \vec{d}, \vec{a}, \vec{\alpha}$). Since we can not provide a derivative of the algorithmic error function, we use a randomized global optimization algorithm (MLSL_LDS)[1] together with a derivative free local optimizer (LN_NELDERMEAD)[2] of the NLopt library[3]. Finding the global minimum of a function is a difficult task, becoming exponentially harder with the number of parameters. We can be certain on finding a true global minimum when (e_T) is close to zero (refer to Section II-C.3). However, it turned out that the error function is badly conditioned, in the sense that the local minimizer easily gets stuck in a local minimum. This made a one step approach, estimating all kinematic parameters of a serial chain at once, impossible. We explicitly make use of the distributed accelerometer located on every segment in order to reduce the number of parameters that need to be estimated at the same time. In Fig. 4 we show how we reuse already estimated kinematic parameter in between real DoF axes. Starting from the static reference segment, the optimization algorithm only has to estimate the parameter for new transformations. These are the transformation in between the previous and new real DoF axis, as well as the virtual DoF axis and the additional parameters we use for the accelerometer coordinate system (refer to Section II-C.1). For the first two DoFs the amount of parameters is further reduced due to additional constraints (refer to Section II-C.5). From three DoFs on, the amount of new parameters stays a constant of (10), (4) for the real DH transformation and $(4 + 2)$ for the accelerometer coordinate system. Another advantage of the incremental approach is that we do not need to excite motion pattern with all DoFs on every call of the error function. Actuating the last three DoFs before the current accelerometer is enough for most robot configurations. Two additional, non-trivial axes help to align the estimation of the third axis, under evaluation, with an already existing structure. This significantly reduces the function call time and seems to also reduce the amount of local minima. Actuating all DoFs every time also makes the global termination criterion difficult, as estimation errors accumulate with a growing chain. We use a global stop criterion ($e_T \leq e_{T_{stop}}$) and a local relative stop criterion ($\triangle e_T / e_T \leq e_{T_{ftol_{rel}}}$) to prevent time wasted in local minima. Boundaries and start values have to be given to the global optimization algorithm. We defined the boundaries as follows and initialized the start value to the middle of each boundary:

$$\theta_{0_d} \in [-\pi;\pi], d_d \in [-1;1], a_d \in [0;1], \alpha_d \in [-\pi;\pi] \tag{18a}$$
$$\theta_{acc} \in [-\pi;\pi], d_{acc} \in [-1;1] \tag{18b}$$

5) **Limitations:** The approach presented in Section II-C is limited to the following extent. It is only possible to independently estimate the kinematic parameters for each identified robotic limb. The transformations in between the first DoF of a limb and an unique coordinate system on the reference segment (${}^{DoF_1}\underline{T}_{RS}$) can not be estimated as the static segment can not be moved. Due to the same reason, two parameters (θ_{0_1} and d_1) of the first transformation (${}^{DoF_2}\underline{T}_{DoF_1}$) are arbitrary. In certain robotic configurations the true Denavit-Hartenberg (DH) parameters (d_d and a_d), as well as the extension (d_{acc}), can not be determined by purely analyzing accelerometer data. This is for example the case when all DoF axes of a limb are aligned.

III. EXPERIMENTS

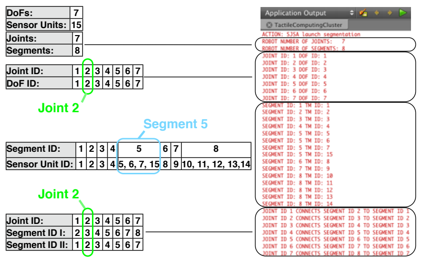
In this section, we present experimental results for the structural exploration and supporting simulation results for the kinematic estimation.

Structural Exploration: Fig. 5 shows results we maintain when performing our structure exploration algorithm on the real KUKA light weight robotic arm shown in Fig. 1. As explained in Section II-B.1 we need to generate a valid activity matrix. We subtracted 1000 accelerometer values, sampled at 1 kHz in a static pose of the robot, from 1000 samples taken at 1 kHz during a commanded motion ($\omega_d(t) = 0.5\frac{rad}{s}$) on every DoF ($d$). The robot excites every known DoF after the

[1]A. H. G. Rinnooy Kan and G. T. Timmer, "Stochastic global optimization methods", Mathematical Programming, vol. 39, p. 27-78 (1987)

[2]J. A. Nelder and R. Mead, "A simplex method for function minimization", The Computer Journal 7, p. 308-313 (1965)

[3]Steven G. Johnson, NLopt V2.2.4, http://ab-initio.mit.edu/nlopt



DoFs:	7
Sensor Units:	15
Joints:	7
Segments:	8

Joint ID:	1	2	3	4	5	6	7
DoF ID:	1	2	3	4	5	6	7

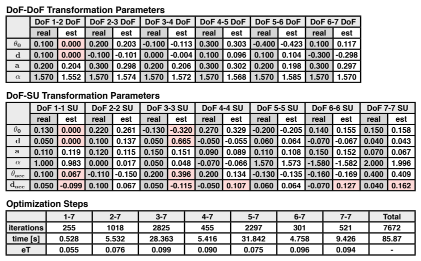
Segment ID:	1	2	3	4	5	6	7	8
Sensor Unit ID:	1	2	3	4	5, 6, 7, 15	8	9	10, 11, 12, 13,14

Joint ID:	1	2	3	4	5	6	7
Segment ID I:	2	3	4	5	6	7	8
Segment ID II:	1	2	3	4	5	6	7

```
ACTION: SJSA launch segmentation
ROBOT NUMBER OF JOINTS:   7
ROBOT NUMBER OF SEGMENTS: 8
JOINT ID: 1 DOF ID: 1
JOINT ID: 2 DOF ID: 2
JOINT ID: 3 DOF ID: 3
JOINT ID: 4 DOF ID: 4
JOINT ID: 5 DOF ID: 5
JOINT ID: 6 DOF ID: 6
JOINT ID: 7 DOF ID: 7
SEGMENT ID: 1 TM ID: 1
SEGMENT ID: 2 TM ID: 2
SEGMENT ID: 3 TM ID: 3
SEGMENT ID: 4 TM ID: 4
SEGMENT ID: 5 TM ID: 5
SEGMENT ID: 5 TM ID: 6
SEGMENT ID: 5 TM ID: 7
SEGMENT ID: 5 TM ID: 15
SEGMENT ID: 6 TM ID: 8
SEGMENT ID: 7 TM ID: 9
SEGMENT ID: 8 TM ID: 10
SEGMENT ID: 8 TM ID: 11
SEGMENT ID: 8 TM ID: 12
SEGMENT ID: 8 TM ID: 13
SEGMENT ID: 8 TM ID: 14
JOINT ID 1 CONNECTS SEGMENT ID 2 TO SEGMENT ID 1
JOINT ID 2 CONNECTS SEGMENT ID 3 TO SEGMENT ID 2
JOINT ID 3 CONNECTS SEGMENT ID 4 TO SEGMENT ID 3
JOINT ID 4 CONNECTS SEGMENT ID 5 TO SEGMENT ID 4
JOINT ID 5 CONNECTS SEGMENT ID 6 TO SEGMENT ID 5
JOINT ID 6 CONNECTS SEGMENT ID 7 TO SEGMENT ID 6
JOINT ID 7 CONNECTS SEGMENT ID 8 TO SEGMENT ID 7
```

Fig. 5. Result of the structure exploration algorithm for the configuration shown in Fig. 1. Joint 2 and Segment 5 are highlighted in both figures.

other and automatically returns to the initial position after every DoF motion. We then compared the absolute difference of the mean values of the three accelerometer axes with a constant threshold of $(0.05g)$, setting the activity value to '1' when the difference was larger. This method gave reliable results for most of the robot poses.



DoF-DoF Transformation Parameters

	DoF 1-2 DoF		DoF 2-3 DoF		DoF 3-4 DoF		DoF 4-5 DoF		DoF 5-6 DoF		DoF 6-7 DoF	
	real	est	real	est	real	est	real	est	real	est	real	est
θ_0	0.100	0.000	0.200	0.203	-0.100	-0.113	0.300	0.303	-0.400	-0.423	0.100	0.117
d	0.100	0.000	-0.100	-0.101	0.000	-0.004	0.100	0.096	0.100	0.104	-0.300	-0.298
a	0.200	0.204	0.300	0.298	0.200	0.206	0.300	0.302	0.200	0.198	0.300	0.297
α	1.570	1.552	1.570	1.574	1.570	1.572	1.570	1.568	1.570	1.585	1.570	1.570

DoF-SU Transformation Parameters

	DoF 1-1 SU		DoF 2-2 SU		DoF 3-3 SU		DoF 4-4 SU		DoF 5-5 SU		DoF 6-6 SU		DoF 7-7 SU	
	real	est	real	est	real	est	real	est	real	est	real	est	real	est
θ_0	0.130	0.000	0.220	0.261	-0.130	-0.320	0.270	0.329	-0.200	-0.205	0.140	0.155	0.150	0.158
d	0.050	0.000	0.100	0.137	0.050	0.665	-0.050	-0.055	0.060	0.064	-0.070	-0.067	0.040	0.043
a	0.110	0.119	0.120	0.115	0.150	0.151	0.090	0.089	0.110	0.108	0.150	0.152	0.070	0.067
α	1.000	0.983	0.000	0.017	0.050	0.048	-0.070	-0.066	1.570	1.573	-1.580	-1.582	2.000	1.996
θ_{acc}	0.100	0.067	-0.110	-0.150	0.200	0.396	0.200	0.134	-0.130	-0.135	-0.160	-0.169	0.400	0.409
d_{acc}	0.050	-0.099	0.100	0.067	0.050	-0.115	-0.050	0.107	0.060	0.064	-0.070	0.127	0.040	0.162

Optimization Steps

	1-7	2-7	3-7	4-7	5-7	6-7	7-7	Total
iterations	255	1018	2825	455	2297	301	521	7672
time [s]	0.528	5.532	28.363	5.416	31.842	4.758	9.426	85.87
eT	0.055	0.076	0.099	0.090	0.075	0.096	0.094	-

Fig. 6. Results of the kinematic parameter estimation based on a training set, compared with the true simulation parameters. We highlighted deviations larger than ±0.1 (radian or meter) in the estimation.

Kinematic Estimation: In order to validate our kinematic estimation approach we used the serial chain model to generate simulated training sets for different robot configurations. Here we present one case with the number of $(D = 7)$ DoFs like with a KUKA light weight robotic arm. The simulator automatically creates $(P = 20)$ pseudo random poses $(\vec{\theta_{pose}})$ with a range of $(\pm 2.62rad)$. The serial chain model then generates the static gravity measurements $({}^{SU_u}\vec{g}_{u,p})$, as well as the dynamic acceleration measurements $({}^{SU_u}\vec{a}_{dyn_{u,d,p}})$ in every pose (p). In order to simulate sensor noise and other imperfections like offsets, we add pseudo random values with an amplitude of $(\pm 0.01g)$ to every measurement. The training set is then passed to the optimization algorithm. We used a global stop criterion of $(e_{T_{stop}} = 0.1)$ and a local relative stop criterion of $(e_{T_{ftol_{rel}}} = 1^{-6})$. After a total of 7672 iterations in 86 seconds[4], the algorithm returned the estimated values, listed and compared with the originals in Fig.6. It is clearly visible that there are no major deviations with the DoF-DoF transformation parameters besides the known limitations with $(\theta_{0_0}$ and $d_0)$. Deviations in the DoF-SU transformation parameters show the presence of ambiguity with the notation chosen to describe the accelerometer coordinate system with extended DH parameters.

[4]MacBook Pro - 2.26 GHz Core 2 Duo, 4GB RAM, Lion

IV. CONCLUSION

In this paper we demonstrated that it is possible to discriminate the structural dependencies of an open loop, articulated robot with a minimum of one 3-axis accelerometer per segment. We presented results that support our approach, to utilize structural and additional physical knowledge, in order to build and estimate kinematic models of robotic limbs based on distributed accelerometers.

6) **Contribution:** Our contribution is towards the facilitation of using large numbers of distributed SUs on articulated robots – especially with artificial sensor skins. The presented open-loop method provides a synergistic calibration of sensor and actuator kinematics, within a short time, and applies only few controllable constraints. The presented method works as an internal observer and thus does not suffer from occlusion, like vision based systems and or is in need of external components.

Acknowledgment: This work was supported (in part) by the DFG cluster of excellence 'Cognition for Technical systems - CoTeSys'.

REFERENCES

[1] P. Mittendorfer and G. Cheng, "Humanoid multi-modal tactile sensing modules," *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 401–410, June 2011.

[2] ——, "Self-organizing sensory-motor map for low-level touch reactions," *11th IEEE-RAS International Conference on Humanoid Robots*, pp. 59–66, October 2011.

[3] A. Ude, C. Man, M. Riley, and C. G. Atkeson, "Automatic generation of kinemtaic models for the conversion of human motion capture data into humanoid robot motion," *Proceedings of the 1st IEEE-RAS International Conference on Humanoid Robotics*, September 2000.

[4] J. Yan and M. Pollefeys, "Automatic kinematic chain building from feature trajectories of articulated objects," *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 712–719, July 2006.

[5] J. Sturm, C. Plagemann, and W. Burgard, "Adaptive body scheme models for robust robotic manipulation," *International Conference on Robotics: Science and Systems IV*, June 2008.

[6] J. F. O'Brien, R. E. Bodenheimer, G. J. Brostow, and J. K. Hodgins, "Automatic joint parameter estimation from magnetic motion capture data," *Proceedings of Graphics Interface*, pp. 53–60, 2000.

[7] D. Roetenberg, H. Luinge, and P. Slycke, "Moven: Full 6dof human motion tracking using miniature inertial sensors," *XSENS TECHNOLOGIES*, 2008.

[8] J. M. Hollerbach and C. W. Wampler, "The calibration index and taxonomy for robot kinematic calibration methods," *The International Journal of Robotics Research*, vol. 14, pp. 573–591, 1996.

[9] D. J. Bennett, J. M. Hollerbach, and P. D. Henri, "Kinematic calibration by direct estimation of the jacobian matrix," *IEEE International Conference on Robotics and Automation*, vol. 1, pp. 351–357, May 1992.

[10] Y. Kuniyoshi, Y. Yorozu, Y. Ohmura, K. Terada, T. Otani, A. Nagakubo, and T. Yamamoto, "From humanoid embodiment to theory of mind," *Lecture Notes in Computer Science*, pp. 202–218, 2004.

[11] M. Hersch, E. Sauser, and A. Billard, "Online learning of the body schema," *International Journal of Humanoid Robotics*, vol. 5, no. 2, pp. 161–181, 2008.

[12] G. Canepa, J. M. Hollerbach, and A. J. M. A. Boelen, "Kinematic calibration by means of a triaxial accelerometer," *IEEE International Conference on Robotics and Automation*, vol. 4, pp. 2776–2782, May 1994.

[13] B. Mooring, M. Driels, and Z. Roth, *Fundamentals of Manipulator Calibration*. John Wiley and Sons Inc, April 1991.

[14] M. Hoffmann, H. G. Marques, A. H. Arieta, H. Sumioka, M. Lungarella, and R. Pfeifer, "Body schema in robotics: A review," *IEEE Transactions on Autonomous Mental Development*, vol. 2, no. 4, pp. 304–324, December 2010.