2011 11th IEEE-RAS International Conference on Humanoid Robots
Bled, Slovenia, October 26-28, 2011

Accelerometer based robotic Joint Orientation Estimation

Erhard Wieser, Philipp Mittendorfer, and Gordon Cheng

***Abstract*—In this paper, we present a new system to automatically estimate the *rotational axis (orientation)* of *robotic joints* relative to *distributed accelerometers*. We designed, implemented, and tested a method for the estimation of joint orientation. The method takes advantage of basic *movement patterns* of a robotic segment. The method uses considerably less input data compared to related methods for the estimation of joint orientation. As sensor input, it only needs the *gravitational acceleration* measured before and after a commanded joint rotation, dynamic acceleration components are not needed. We evaluated the implementation of the method on a Bioloid robot equipped with three Tactile Module prototypes. Our Tactile Modules are multimodal sensor systems and also feature a triaxial accelerometer. The robot successfully estimated the rotation axes of each DOF of its shoulder and elbow joints relative to the accelerometer frames of the Tactile Modules that are randomly distributed on the corresponding segments.**

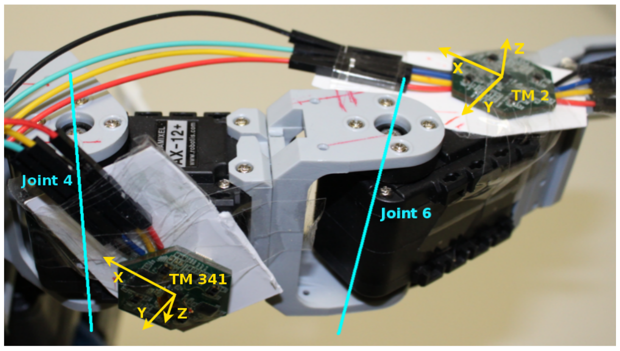


Fig. 1. The left arm of a Bioloid robot equipped with two of our Tactile Modules (TM 341 and TM 2). The robot moves the joints 4 and 6 (one joint at a time). Our system estimates the orientation of the joint axes (cyan) relative to the accelerometer frames (orange) of the Tactile Modules that are randomly distributed on the segments.

I. INTRODUCTION

A. Body schema, automatic kinematic calibration, and joint reconstruction

Humans are aware of their limbs and joints, they are aware of the shape of their entire body. Based on this awareness, the brain can exactly control each single human movement. The sensory representation of the 3-D body shape is termed as a *body schema*. In [1], body schema is defined as: "(a) the mapping from proprioception and efferent copy (copy of motor command) to body posture and motion, and (b) the mapping from tactile sensation to its originating position on the body surface". This definition is derived from Head and Holmes [2] who initially proposed the term body schema. The autonomous acquisition of body schema is also one of the important issues in embodied artificial intelligence. According to [3] and [4], the physical body of an agent specifies the constraints on the interaction between the agent and its environment. A physical body also helps to form cognition and action of an agent.

Body schema is strongly connected to the problem of *automatic kinematic calibration* in robotics [5]. Kinematic calibration acquires relative joint positions/distances and orientations, and uses this data to obtain the kinematic function, e.g., forward kinematics. It is highly desirable to automate the calibration process, so that a robot finds its kinematic parameters by itself, especially without relying on external sensors. By external sensors, we mean measurement systems, which are neither a part of the robot nor attached to its surface. Automatic calibration contrasts with traditional industry robotics, where the kinematic parameters are either already provided by the manufacturer of the robot or gathered by a calibration procedure with a human in the processing loop utilising external measurement systems.

Institute for Cognitive Systems, Technische Universität München, Karlstrasse 45/II, 80333, München, Germany, Email: see `http://www.ics.ei.tum.de/`

In order to solve the problem of autonomous kinematic calibration and to provide a body schema, our overall approach is to create an autonomous robot reconstruction system. To this end, the surface of the robot is covered by Tactile Modules [6]. The Tactile Modules are multimodal sensor systems, which can be connected to each other to form a network. A Tactile Module Network emulates the human skin and approaches multimodal whole body touch sensation for humanoid robots [6]. An important application of the Tactile Modules is full spatial calibration, also called 3-D shape reconstruction [6], by using their integrated BMA 150 acceleration sensors. Thus, a robot would reconstruct the 3-D shape of its segments, which are fully covered by the Tactile Modules, but even then one problem still remains: the segments are connected by joints. So if the entire 3-D shape and also the kinematic chain of the robot have to be reconstructed, the joints connecting two segments will have to be reconstructed as well. This can be achieved by acquiring the distance and orientation of the joint axis relative to a Tactile Module on the corresponding segment, as it is pointed out in [6].

Joint reconstruction can be split up into two problems, the estimation of *joint position/distance* on the one hand, and the estimation of *joint orientation* on the other. This paper investigates the latter one, the estimation of joint orientation. By joint orientation, we mean the vector describing the rotation axis of a joint relative to a specific reference frame. It is crucial to have data on how the joint axes are oriented to each other. This information is important to gather the forward kinematics and to match reconstructed segments to each other.

Here, we present a system, which automatically estimates the rotation axis of a joint relative to randomly distributed accelerometers. We use the triaxial accelerometer embedded into each of our Tactile Modules. In order to estimate the axis of a selected joint, our system first commands that joint to perform a simple movement pattern (only one joint moves at a time) and then it deduces the joint axis.

B. Related Works

Here, we focus on related works on the estimation of joint orientation/axis as a crucial part of joint reconstruction and kinematic calibration.

Hersch, Sauser, and Billard presented a system which estimates robotic joint orientation and position in order to acquire a body schema represented by the kinematic function [5]. The system uses a learning mechanism, which combines information from the proprioception (motor encoders), the stereo vision, and tactile sensors. An iterative gradient-descent method calculates joint orientation and position, but it also needs visual or tactile input for this purpose.

Different possibilities exist to describe the orientation of a joint axis. In [5], the orientation of a joint axis is described as a vector. As it is shown in [7], the Denavit-Hartenberg convention uses angles to describe the orientation of a joint axis i relative to the previous joint axis $i - 1$.

The orientation of the joint axis is necessary to acquire the kinematic chain and it is one of the parameters, which *kinematic calibration* should find out.

According to the categorization in [8], kinematic calibration can be split up into three approaches: 1) *closed-loop calibration*; 2) *open-loop calibration*; and 3) *screw axis measurement*.

Closed-loop calibration uses movement patterns (self motions) and only joint angle sensing, without any external metrology system. The method requires the robot to attach its end effector to the ground and to form a mobile closed kinematic chain [8]. Works on closed-loop calibration are reported in [9], [10], [11], and [12].

Open-loop calibration requires an external metrology system to measure the pose of the end effector of a robot [8]. Many approaches exist to measure the pose of the end effector. For this purpose, Goswami, Quaid, and Peshkin use a radial-distance linear transducer (LVDT) [13]. It is also possible to use a calibration plate with index marks, which provide accurate positions of the end effector, as reported in [14]. Stone, Sanderson, and Neuman use external ultrasonic range sensors [15].

Screw axis measurement identifies rotational joints of a robot as a screw, i.e., as a line in space [8]. The method extracts the kinematic parameters from the knowledge of all of the joint screws found. Screw axis measurement splits up into two sub-methods: 1) *circle point analysis (CPA)*; and 2) *Jacobian measurement method*.

In *circle point analysis*, each joint moves in a circle [8], [16], while an external measurement unit records the positions of the end effector, similar to open-loop calibration. The method uses these recorded positions to approximate a plane, in which the circle lies [16]. The normal of this plane and the center of the circle define the joint screw. Instead of external position sensors to obtain the location of the end point of a robotic link, Canepa, Hollerbach, and Boelen used an onboard accelerometer and proceeded similarly to open-loop methods with a nonlinear optimization to derive the parameters, see [17]. Their system calibrated a 7 DOF robotic arm. The researchers mounted a triaxial accelerometer on the robot endpoint. Their method calculates kinematic parameters (joint axis orientation, center of rotation). Concerning the joint orientation, they report a very high accuracy, which is partly due to their high precision accelerometer used. However, they do not clearly mention how they derive that reported accuracy, i.e., it is not clear how they know the ideal axis *relative* to the *accelerometer frame*. When the accelerometer frame is mounted randomly somewhere on the robot segment, it is important to first gather the ideal axis orientation relative to the accelerometer frame, before one can compare that ideal orientation with the orientation estimated by the system. Since we distributed our Tactile Modules randomly somewhere on the segments and measured each axis orientation relative to our accelerometer frame manually by using a triangle and goniometer, the ideal axis alone can have uncertainties of a few degrees. So when obtaining the ideal axis manually by such simple instruments, it is unlikely that the estimation accuracy is less than 1°, even in case of a correct estimation. The precision of the actuators is also an important factor influencing the estimation result.

The *Jacobian measurement method* determines the joint screws simultaneously by measuring the Jacobian matrix [8]. This method works *velocity based* on the one hand, or *force and torque based* on the other hand [18]. As it is described in [18], the *velocity based approach* needs an external measurement unit, which first gathers the endpoint linear and angular velocities. Then the method estimates the kinematic parameters. The *force and torque based approach* requires a manipulator to have joint torque sensing and endpoint 6-axis force/torque sensing [18]. During the calibration procedure, the manipulator rigidly attaches its endpoint to the environment. The method systematically exerts the joint torques and the corresponding endpoint forces. In the identification step, the method estimates the parameters. Hollerbach, Giugovaz, Buehler, and Xu apply screw axis measurement based on the Jacobian method in order to calibrate the Sarcos Dextrous Arm, see [19]. They also compared the method to open-loop calibration and CPA.

In sum, the related works show that joint orientation is one of the key parameters used to extract kinematic data. However, the methods of the related works have their drawbacks when applying them on autonomous robot systems. They either need external measuring devices (like standard open-loop approaches and CPA) or require the robot to form a closed-loop with its environment (like closed-loop approaches). When using a closed-loop approach for example, a humanoid robot would always have to attach its arms to some part of the environment in order to automatically calibrate its arms.

C. Our Approach

The idea of our work was to acquire the joint orientation with a *minimum of input data*. In contrast to open-loop approaches and especially CPA, our approach does not need external measuring devices. Unlike the closed-loop approaches, a robot using our method does not need to form a closed kinematic chain with its environment. We investigated the capability of the estimation of joint axis by using only acceleration as sensor modality. We decided to take a 'static' estimation approach. In contrast to CPA as a common method for joint orientation estimation, our approach for orientation estimation relies only on the *gravitational acceleration* measured at different static segment positions and the commanded *joint rotation angle*. An accelerometer A is mounted on a segment S (see Fig. 2). Here, the expression 'static' indicates that the segment S does not move while the accelerometer is measuring the gravity vector (G-vector). After the system has measured the G-vector, it moves the segment S by the corresponding rotational joint J, which axis is to be estimated. Our approach does not need dynamic acceleration components, no accelerometer recordings are captured during segment motion. Note that only one joint is allowed to move at a time. After the movement has stopped, the system measures the G-vector again. Then it deduces the joint orientation from the different G-vectors and the known joint rotation angle. The rotation angle is relatively small (ca. $17°$) compared to the one of CPA (step by step throughout the *entire* joint range, e.g., $180°$, according to [19]). The smaller the joint rotation angle is, the smaller the difference between the recorded G-vectors gets. In general, the rotation angle should not be too small, e.g., below $10°$, since the G-vectors are nearly one and the same then, making the estimation of joint orientation less precise. The rotation angle should also not be too big, since the robot requires more space and time to move the corresponding segment. We found out that an angle between $15°$ and $20°$ is a good tradeoff. Our approach is limited to rotational joints, so it estimates the orientation of each rotation axis representing a DOF. Our approach has the advantage of using *flexible reference frames*. This means that a Tactile Module (TM) can be placed anywhere on a robotic segment. The location is *not* relevant, the module should just stay fixed somewhere on the segment moved by the joint, which is to be estimated. One Tactile Module on segment S is enough to estimate the corresponding joint axis J. If two or more TMs are attached on the segment, e.g., a network of TMs, one of them can be arbitrarily selected to be the reference TM providing the reference frame. Our estimation method is based on a global optimization of a quaternion rotation and uses a simple movement pattern of the joint that is to be estimated. Closed-loop and open-loop approaches use nonlinear optimization techniques, however their mathematical and algorithmic approach, e.g., in [17], is different to ours, and none of the related work was found utilises quaternions, which are robust to singularities.

As input data, our method needs gravitational acceleration and the rotation angle commanded to the robotic joint. As output data, our method delivers the orientation/direction vector of a joint axis relative to the frame of a selected accelerometer.

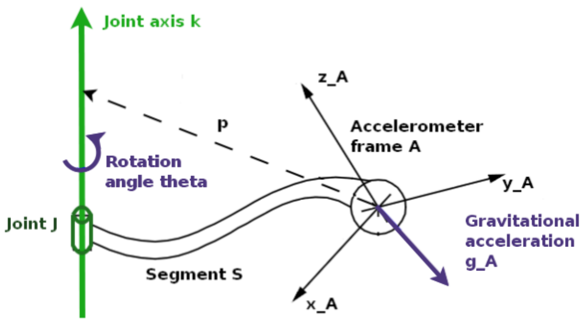


Fig. 2. Model of our joint estimation approach (original image [17] was modified): Inputs (purple) of the estimation system are the G-vectors recorded in multiple segment poses and the rotation angle commanded to joint J. The output (light green) is the unit vector $\boldsymbol{\kappa}$ describing the orientation of J relative to the accelerometer frame. The estimation of the vector $\boldsymbol{p}$ (dashed arrow) describing the position of J also belongs to joint reconstruction but is *not* the focus of this paper. When using our Tactile Modules, the accelerometer frame A is also the module frame.

II. SYSTEM DESCRIPTION

A. Hardware – Tactile Modules

Our *Multimodal Tactile Modules (HEX-O-SKINs)* [6] are an approach to multimodal humanoid tactile sensing. They form an artificial robotic sensor skin, emulating the functions of human skin. Every HEX-O-SKIN is a small, rigid *printed circuit board (PCB)*, which incorporates different sensor modalities together with a local controller, as it is described in [6]. The PCB has a hexagonal shape of 1.4 cm edge length (see Fig. 3). An important feature is that four of six edges have ports, which can connect neighbouring modules. Thus, we can form a *Tactile Module Network (TMN)*. For joint orientation estimation, we only need the integrated triaxial acceleration sensor BMA 150. Note that the frame of the BMA 150 is the frame of the corresponding Tactile Module as well.

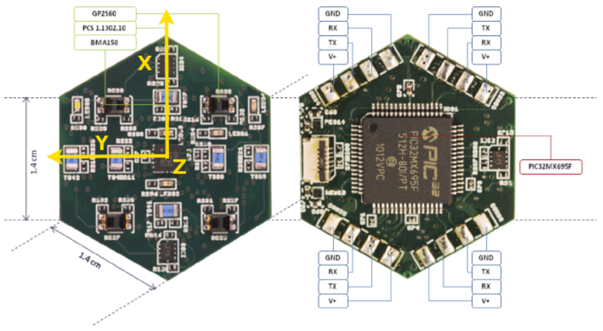


Fig. 3. HEX-O-SKIN. Top view (left) and bottom view (right). The BMA 150 acceleration sensor is located in the center. The vector describing joint orientation is estimated relative to the right handed Cartesian frame (orange) of the BMA 150.

B. Conditions

At a certain processing step, our method estimates the axis of rotation $\boldsymbol{k}$ described in frame A. Note that the axis of rotation $\boldsymbol{k}$ is actually identical with the axis $\boldsymbol{\kappa}$ of joint J (joint orientation) when accelerometer A is mounted on the segment S (see Fig. 2). This is true for the following conditions:

- Segment S is rigid, e.g., not made of a material which can deform during movement.
- Only joint J is allowed to move during the estimation process, all the other joints have to stand still (one joint at a time).
- The orientation of the whole robot does not change during the estimation process.

These conditions have to be met when running the implementation of our method on a robot. In the following, one can consider these conditions to be fully met. This is also the case when our system performs on different robotic platforms, like the Bioloid or a KUKA Lightweight arm.

C. Our Method

The goal is that the joint orientation has to be estimated relative to the accelerometer frame, no matter how the accelerometer is mounted on the corresponding segment. So the orientation of the Tactile Module in space and its orientation relative to the joint axis should *not* be constrained.
We observed that a method using roll-pitch-yaw angles as orientation descriptors is not the best choice due to singularities. Thus, our method works with quaternions. Our method delivers the vector $\boldsymbol{\kappa}$ describing joint orientation according to the right hand rule of rotation. The main idea is to extract this vector from a quaternion rotation:

$$\hat{\boldsymbol{q}}_2 = \hat{\boldsymbol{q}}_R * \hat{\boldsymbol{q}}_1 * \hat{\boldsymbol{q}}_R^{-1} \quad . \tag{1}$$

where $\hat{\boldsymbol{q}}$ denotes a quaternion and $*$ symbolizes quaternion multiplication. Generally, Eq. (1) rotates a 3-D vector represented by $\hat{\boldsymbol{q}}_1$ into the vector represented by $\hat{\boldsymbol{q}}_2$. The vectors are described in the same frame and the rotation quaternion $\hat{\boldsymbol{q}}_R$ encodes the axis of rotation. Applying Eq. (1) on the estimation of joint axis, the 3-D vectors represented by $\hat{\boldsymbol{q}}_1$ and $\hat{\boldsymbol{q}}_2$ are the G-vectors in frame A which the accelerometer measures before and after rotation of J, i.e., from servo position 1 to servo position 2 (see Fig. 4).

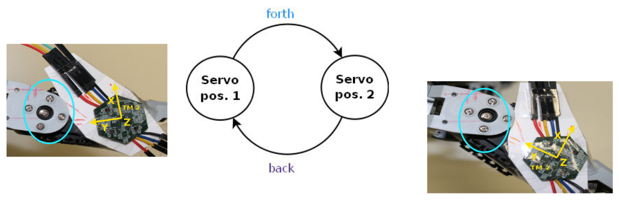


Fig. 4. Basic movement pattern (top view on the joint). Here, the orientation of the elbow joint (cyan) is estimated relative to the TM having ID 2. Our system chooses a movement direction (back or forth) and then it estimates joint orientation according to the right hand rule of rotation.

$$\hat{\boldsymbol{q}}_1 = \begin{bmatrix} 0 & g_{1_x} & g_{1_y} & g_{1_z} \end{bmatrix}^T \tag{2}$$

$$\hat{\boldsymbol{q}}_2 = \begin{bmatrix} 0 & g_{2_x} & g_{2_y} & g_{2_z} \end{bmatrix}^T \tag{3}$$

Given $\hat{\boldsymbol{q}}_1$ and $\hat{\boldsymbol{q}}_2$, now the aim is to find out $\hat{\boldsymbol{q}}_R$, since it contains the axis of rotation and its angle.

$$\hat{\boldsymbol{q}}_R = \begin{bmatrix} q_{R_1} & q_{R_2} & q_{R_3} & q_{R_4} \end{bmatrix} \tag{4}$$

$$\hat{\boldsymbol{q}}_R = \begin{bmatrix} \cos(\frac{\Delta\theta_{cmd}}{2}) \\ k_x \cdot \sin(\frac{\Delta\theta_{cmd}}{2}) \\ k_y \cdot \sin(\frac{\Delta\theta_{cmd}}{2}) \\ k_z \cdot \sin(\frac{\Delta\theta_{cmd}}{2}) \end{bmatrix} \tag{5}$$

We transform Eq. (1) into a compact form written as

$$\boldsymbol{0} = \boldsymbol{G} \cdot \hat{\boldsymbol{q}}_R \quad . \tag{6}$$

We call the matrix of Eq. (6) the 'gravity-matrix' $\boldsymbol{G}$, since it consists of the G-vectors before and after rotation of J. Equation (6) is determined analytically, and it is a null space problem, which is solved by *singular value decomposition (SVD)*. The solution is the right singular vector of $\boldsymbol{G}$ corresponding to the smallest singular value. However, we observed that the solution found was not unique. After every run, two equal smallest singular values existed, which means that any linear combination of the corresponding right singular vectors is a valid solution. For this reason, we took another approach: we expected the left side of Eq. (6) not to be exactly zero due to noise or other irregularities.

$$\boldsymbol{e} = \boldsymbol{G} \cdot \hat{\boldsymbol{q}}_R \tag{7}$$

This led to an optimization problem. Besides the given G-vectors, we used the commanded angle of joint rotation $\Delta\theta_{cmd}$ as additional information. Our method minimizes Eq. (7) along with the additional constraint that $\boldsymbol{k}$ has to be a unit vector. The method minimizes with respect to the rotation axis $\boldsymbol{k}$. We decided to use the *Nelder-Mead algorithm (NMA)* in order to minimize this error function. The advantages of NMA are that it is robust and the function, which is to be minimized does not need to have a derivative. In general, optimization methods often converge into local minima, strongly dependent on the initial guess, so they do not guarantee a global solution. A common problem is the false convergence at a point other than the minimum. This difficulty was also found when using the NMA [20]. In order to find a global minimum, we use a finite set of start vectors as input for the Nelder-Mead algorithm. Each of these start vectors is a point located on the surface of a sphere with the radius $r = 1$. The number of start vectors depends on the step width δ_s of the discretization of the sphere surface. A step width of $\delta_s = 0.1$ was enough to obtain a high number of initial vectors. This resulted into 2016 different points on the sphere surface, $\boldsymbol{k}_{i_1}, \boldsymbol{k}_{i_2}, \ldots, \boldsymbol{k}_{i_{2016}}$.
Our overall system architecture is depicted in fig. 5. Our system calculates one solution $\boldsymbol{k}_s$ for each initial guess $\boldsymbol{k}_i$ on the sphere surface. A large set $\{S\}$ of solutions $\boldsymbol{k}_s$ (symbolized by $\{S\}_{\boldsymbol{k}_s}$) is available to our system *before* it enters the processing steps marked by a green dashed line in Fig. 5. We call each $\boldsymbol{k}_s$ a *single solution*. The vector $\boldsymbol{\kappa}_{axis}$ describing the joint

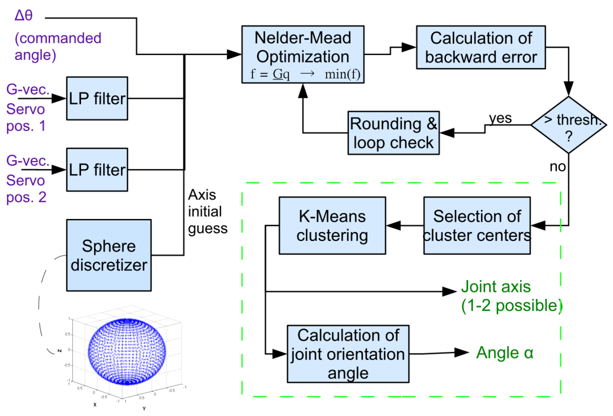
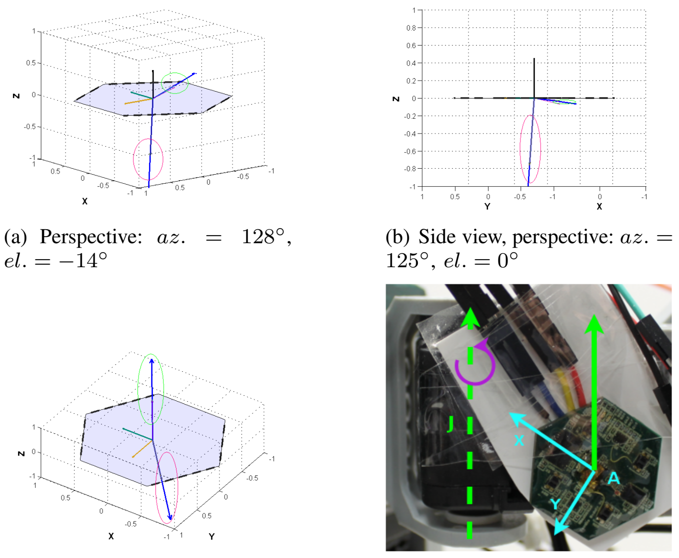


Fig. 5. System architecture: estimation by global optimization of a quaternion rotation. Input data in purple, output data in dark green. The visual result of the process of sphere discretization is shown below the box 'Sphere discretizer'. Each point (blue) on the sphere surface is sequentially put into the NM optimization step. For each solution, the system calculates the backward error. It is the Euclidian distance between the estimated $\boldsymbol{g}_2$ and the measured $\boldsymbol{g}_2$. This ensures the correctness of each estimated solution $\boldsymbol{k}_s$ of the optimization step. A K-Means algorithm with automatic selection of suitable cluster centers deduces 1 up to 2 final vectors $\boldsymbol{\kappa}_{axis_1}$, $\boldsymbol{\kappa}_{axis_2}$ (joint axis) describing the orientation of the joint. In addition, the system also calculates the joint orientation angle α, which is the angle enclosed by the TM-plane and the estimated vector $\boldsymbol{\kappa}_{axis}$.



(a) Perspective: $az. = 128°$, $el. = -14°$

(b) Side view, perspective: $az. = 125°$, $el. = 0°$

(c) Perspective similar to the photo on the right: $az. = 146°$, $el. = -62°$

(d) The joint axis (light green) in reality, according to the rotation direction (purple)

Fig. 6. This visual output of our system shows the solution manifold problem in the A-frame of a TM (x-axis in dark green, y-axis in dark orange, z-axis in black, TM-ports represented by dashed black edges). Here, two solutions exist. They are the average vectors displayed in blue color, each of them represents one group of single solutions. All of them have a small backward error (here, $d_{BE} = 0.0037$), so they satisfy Eq. (7), but the picture 6(d) shows that only one joint axis (light green) is correct according to the right hand rule of rotation, which corresponds to one of the blue average vectors. A green ellipse marks the right one, a red ellipse the wrong one.

axis has to be 'filtered out' of $\{S\}_{\boldsymbol{k}_s}$. If the whole set consists of single solution vectors which have very little numerical difference to each other (e.g., numerical discrepancy after the third decimal place), it is clear that the set will represent one solution which can be described by the average of all $\boldsymbol{k}_s$. So in this case, our optimization method has converged into one final solution (one average vector). Nevertheless, experiments showed that $\{S\}_{\boldsymbol{k}_s}$ can sometimes be divided into more than one group, each group again represented by its average vector. This means that our optimization method has converged into multiple final solutions. In such case, a single solution of one group has a mentionable numerical difference to a single solution of another group. This difference results into a certain Euclidian distance between the corresponding solution vectors. Our system has to obtain the correct average vector, which describes the physical joint axis according to the right hand rule of rotation. All the other average vectors are correct in so far that they minimize the error function, but they are *not* the joint axis in reality. In the following, we term this phenomena as the *problem of solution manifolds*. A typical case is shown in Fig. 6. For the clustering process, the number of cluster centers is two because of the *worst case configuration*, in which the G-vector obtained in servo position 1 is (nearly) the same as in servo position 2. So both estimated axes (one axis points up and the other one points down, both aligned with the G-vector) are valid, since each of them turns $\boldsymbol{g}_1$ into $\boldsymbol{g}_2$ according to the right hand rule of rotation. The main task of the clustering process is to divide the single solutions into one up to two groups, so that the system can easier select the correct group (corresponding to the physical joint axis) later.

D. Final algorithm

As output, our method delivers either one joint axis $\boldsymbol{\kappa}_{axis}$ on the one hand, or two joint axes $\boldsymbol{\kappa}_{axis_1}$ and $\boldsymbol{\kappa}_{axis_2}$ on the other hand (see output of Fig. 5). In the second case, only one of the estimated axes is physically correct, $\boldsymbol{\kappa}_{axis_1}$ *or* $\boldsymbol{\kappa}_{axis_2}$. Experimental results showed that in contrast to the wrong estimation, it is very likely that the correct estimation repeats itself after the system has moved the joint J once again in the same direction, i.e., from servo position 2 to position 3 (see Fig. 7). Based on this discovery, we created

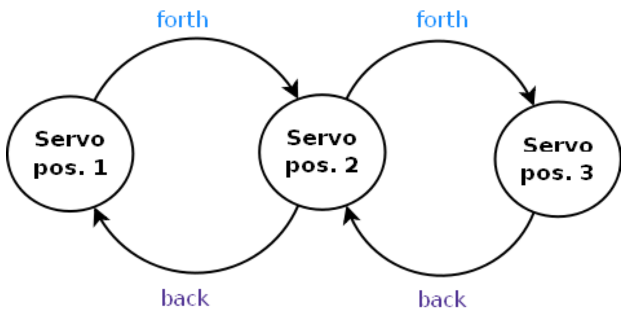
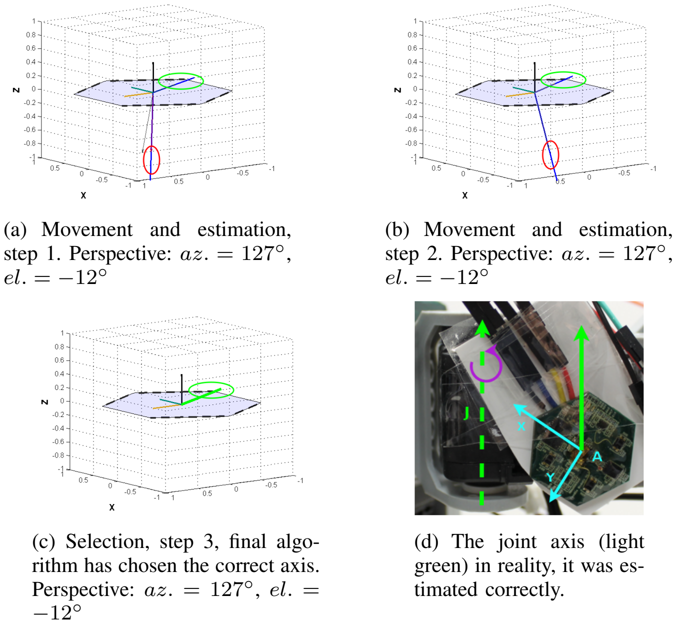


Fig. 7. Schema of the extended movement pattern.

the final algorithm for joint estimation. After measuring the G-vector in servo position 2, the robot is commanded to move its joint from position 2 to position 3 in the same direction with the same amount of joint rotation $\Delta\theta_{cmd}$. Our final algorithm calls the implementation of our method (Fig. 5) *twice*: the first time with the G-vectors of servo position 1 and 2 (*movement and estimation, step 1*); and again with the G-vectors of servo position 2 and 3 (*movement and estimation, step 2*). Then, it selects the correct joint axis from the results obtained (*selection, step 3*). In this selection step, our algorithm calculates the average of the two axes (one of movement step 1 and one of movement step 2), which have the *minimum* Euclidian distance to each other. This distance is very small, since the corresponding axes are actually the same in reality (correct estimation repeats itself from movement step 1 to step 2). The principle of the solution repeating itself after a second joint movement is visualized in Fig. 8.


(a) Movement and estimation, step 1. Perspective: $az. = 127°$, $el. = -12°$

(b) Movement and estimation, step 2. Perspective: $az. = 127°$, $el. = -12°$

(c) Selection, step 3, final algorithm has chosen the correct axis. Perspective: $az. = 127°$, $el. = -12°$

(d) The joint axis (light green) in reality, it was estimated correctly.

Fig. 8. The correct estimation repeats itself after an additional joint movement in the same direction. As one can clearly see in 8(a) and 8(b), the Euclidian distance between the wrong estimations (red ellipses) are greater than between the correct ones (green ellipses). The Euclidian distance between the correct estimations of step 1 and 2 tends towards zero.

III. EXPERIMENTS

A. Setup

We tested our system on the Bioloid robot with three Tactile Modules randomly mounted on the segments (Fig. 9). Compared to other robots like KUKA, the motor encoders of the Bioloid are less precise. We wanted to show that our joint orientation estimator also performs on low-cost robots. For our distribution of Tactile Modules on the segments as shown in Fig. 9, the robot has the following possibilities to estimate the corresponding joint axes, see the schematics in Fig. 10. We tested 15 'joint to sensor' configurations where the Tactile Modules had various inclinations and poses relative to the joint axes, which were to be estimated (Fig. 11). For each of the 15 'joint to sensor' configurations, we let the robot do five repetitions of the estimation process. Note that the estimation accuracy or estimation error is the angle between the axis estimated by our system and the ideal axis. For each 'arm/joint to sensor' configuration, we first measured the orientation of the ideal axis on the robot manually by using a triangle and goniometer, and then let the system estimate the corresponding axis orientation.

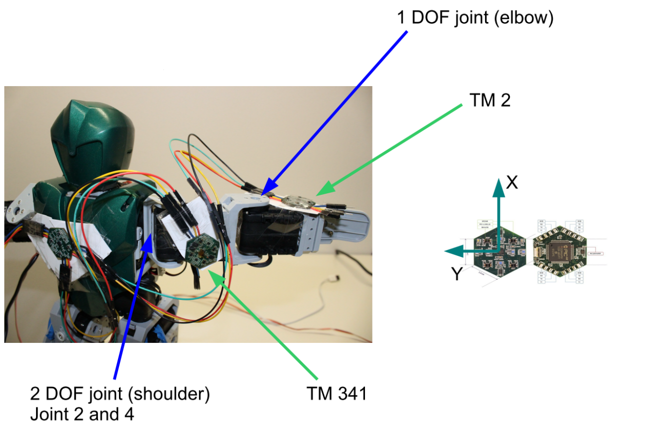
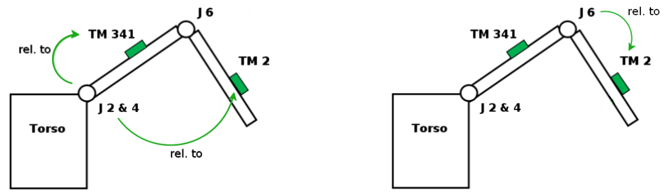


Fig. 9. Experimental setup on the Bioloid robot: the blue arrows mark the joints, which axes were to be estimated. The green arrows mark the Tactile Modules attached to the corresponding segments. The joint axes of the shoulder and elbow provided sufficient various arm configurations in order to test and validate our system.



(a) Joint 2 can be estimated relative to the frame of TM 341 and TM 2. Joint 4 can be estimated relative to the frame of TM 341 and TM 2. Joint 2 and joint 4 move upper arm and lower arm with joint 6 staying fixed.

(b) Joint 6 can be estimated relative to the frame of TM 2. Joint 6 can only move the lower arm.

Fig. 10. Estimation possibilities. Note that the shoulder joint has two DOFs (joint axis 2 and 4) and the elbow joint has one DOF (joint axis 6).

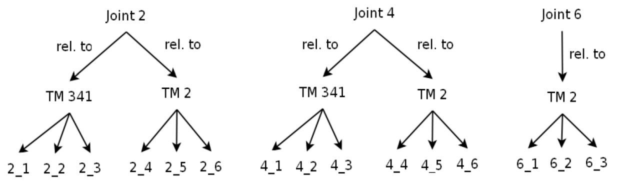


Fig. 11. Tree structure showing the 15 'joint to sensor' configurations we tested. In each of these configurations, the robot did five consecutive axis estimations per back or forth direction.

B. Results

1) Mean error: In each direction of rotation (back or forth), the mean error was calculated after five estimations.

An overview is shown in Fig. 12.

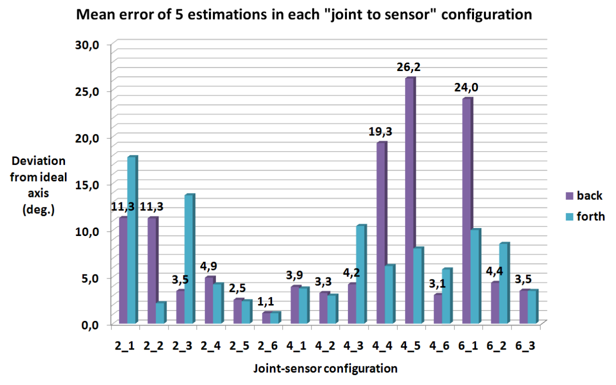


Fig. 12. Mean error in all the tested 'joint to sensor' configurations. Outliers in some cases lead to greater mean errors over five estimation results. Only one single estimation alone might not be enough (danger of an outlier). Thus, the robot should do five estimations one after the other and take the *median* of these estimation results to be the final joint axis (see Fig. 13).

2) Median error: In each direction of rotation (back or forth), the median error was calculated after five estimations. An overview is shown in Fig. 13.

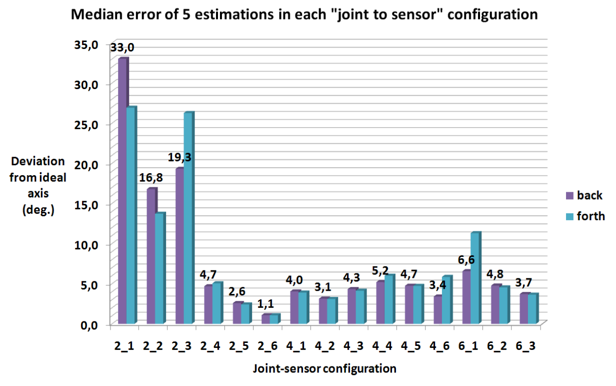


Fig. 13. Median error in all the tested 'joint to sensor' configurations. Our system achieves a good accuracy (ca. $4^\circ - 5^\circ$) for all 'joint to sensor' configurations tested except for the cases 2_1 – 2_3. In those cases, the accuracy was also depended on the choice of internal system parameters used for the automatic selection of cluster centers. Unfortunately, there is no 'golden' choice of parameters.

IV. CONCLUSION

A. Discussion

The overall system was developed in a bottom-up manner. At the beginning, it was not clear how our optimization method will behave, e.g., convergence into one or more solutions. This could only be discovered experimentally. Therefore, it was helpful to reduce and to specify the amount of unknown parameters at the start of the design process. We explicitly decided to use the commanded angle of joint rotation as additional input data, otherwise our method would have to minimize the error function with respect to four unknown parameters (k_x, k_y, k_z, $\Delta\theta_{cmd}$) instead of three (k_x, k_y, k_z). The control system specifies the commanded joint angles anyway.

We expect a better estimation accuracy on robots having more precise actuators, e.g., the KUKA arm. Note again that the *ideal* axes, which we used to compare to the estimated ones in order to obtain the accuracy, were *not* pre-given like sometimes kinematic parameters are on datasheets. We measured the ideal axis orientation manually relative to the accelerometer frame by using simple instruments like a triangle. This can already imply some inaccuracies. The estimation accuracy also depends heavily on the precision of the accelerometers, which are influenced by accelerometer axis calibration and offsets. In general, any other type of triaxial accelerometer can be used to work with our method.

B. Summary

In this paper, we presented a system, which automatically estimates the orientation of a robotic joint. As input data, our system uses the commanded rotation angle and the gravity vector measured by the triaxial accelerometers. We designed our system based on quaternion calculations, which are not affected by singularities. Our estimation method uses an optimization approach to acquire the joint axis. It turned out that this approach leads to solution manifolds. Our method uses K-means clustering with automatic selection of cluster centers to group the solutions found. The final algorithm of our method selects the solution, which is physically correct. Except for a few cases, the estimation accuracy is about four degrees in the median on the Bioloid robot.

C. Contribution

A notable advantage of our axis estimation method is that it really relies on minimum input data. Only gravitational acceleration is used along with the commanded rotation angle. Related methods for the estimation of joint axis like the circle point analysis of screw axis measurement are dependent on external sensors. Sometimes they also suffer from the drift of accelerometer integration. In addition, our system commands relatively small rotation angles to the joint, compared to related methods like CPA where the joint is often moved through its entire range.

With the obtained joint orientations, combined with data on surface geometries (3-D reconstruction [6]) and kinematic tree, the robot has a large collection of useful data for its overall shape reconstruction, which is helpful to acquire a body schema.

Acknowledgment

This work was supported (in part) by the DFG cluster of excellence 'Cognition for Technical Systems CoTeSys' of Germany.

REFERENCES

[1] C. Nabeshima, M. Lungarella, and Y. Kuniyoshi, "Timing-based model of body schema adaptation and its role in perception and tool use: a robot case study," *Proceedings of the 4th IEEE International Conference on Development and Learning*, pp. 7–12, 2005.

[2] H. Head and G. Holmes, “Sensory disturbances from cerebral lesions,” *Brain*, vol. 34, no. 2-3, pp. 102–254, 1911.

[3] Y. Kuniyoshi, Y. Yorozu, S. Suzuki, S. Sangawa, Y. Ohmura, K. Terada, and A. Nagakubo, “Emergence and development of embodied cognition: A constructivist approach using robots,” *Progress in Brain Research*, vol. 164, pp. 425–445, 2007.

[4] M. Asada, K. Hosoda, Y. Kuniyoshi, H. Ishiguro, T. Inui, Y. Yoshikawa, M. Ogino, and C. Yoshida, “Cognitive developmental robotics: a survey,” *IEEE Transactions on Autonomous Mental Development*, vol. 1, no. 1, pp. 12–34, 2009.

[5] 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.

[6] P. Mittendorfer and G. Cheng, “Humanoid multimodal tactile-sensing modules,” *IEEE Transactions on Robotics*, vol. 27, no. 3, pp. 401–410, 2011.

[7] P. Allard, I. A. F. Stokes, and J.-P. Blanchi, *Three-dimensional analysis of human movement*. Human Kinetics, 1995.

[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. 15, no. 6, pp. 573–591, 1996.

[9] D. J. Bennett and J. M. Hollerbach, “Autonomous calibration of single-loop closed kinematic chains formed by manipulators with passive endpoint constraints,” *IEEE Transaction on Robotics and Automation*, vol. 7, no. 5, pp. 597–606, 1991.

[10] ——, “Identifying the kinematics of robots and their tasks,” *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 1, pp. 580–586, 1989.

[11] ——, “Self-calibration of single-loop, closed kinematic chains formed by dual or redundant manipulators,” *Proceedings of the 27th Conference on Decision and Control*, vol. 1, pp. 627–629, 1988.

[12] D. J. Bennett, D. Geiger, and J. M. Hollerbach, “Autonomous robot calibration for hand-eye coordination,” *The International Journal of Robotics Research*, vol. 10, no. 5, pp. 550–559, 1991.

[13] A. Goswami, A. Quaid, and M. Peshkin, “Identifying robot parameters using partial pose information,” *IEEE Control Systems Magazine*, vol. 13, no. 5, pp. 6–14, 1993.

[14] S. Hayati, K. Tso, and G. Roston, “Robot geometry calibration,” *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 2, pp. 947–951, 1988.

[15] H. W. Stone, A. C. Sanderson, and C. P. Neuman, “Arm signature identification,” *Proceedings of the IEEE International Conference on Robotics and Automation*, vol. 3, pp. 41–48, 1986.

[16] H. W. Stone, *Kinematic modeling, identification, and control of robotic manipulators*. Springer, 1987, vol. 29.

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

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

[19] J. M. Hollerbach, L. Giugovaz, M. Buehler, and Y. Xu, “Screw axis measurement for kinematic calibration of the sarcos dextrous arm,” *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, vol. 3, pp. 1617–1621, 1993.

[20] J. A. Nelder and R. Mead, “A simplex method for function minimization,” *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.