

# Simultaneous state and dynamics estimation in articulated structures

Francesco Nori<sup>1</sup>, Naveen Kuppuswamy<sup>1</sup> and Silvio Traversaro<sup>1</sup>

**Abstract**—Given an articulated rigid body, we define the problem of estimating its dynamics as the problem of computing all the forces and accelerations acting on the bodies which constitute the articulated system. Similarly, we define the state estimation problem as the problem of computing the system positions and velocities. In the present paper we propose a framework for simultaneous state and dynamics estimation. The estimation is framed in a Bayesian framework and a suitable Bayesian prior is defined to guarantee the physical consistency of the obtained estimation. The Bayesian posterior makes use of all available measurements which include encoders, gyroscopes, accelerometers, force and torque sensors. The proposed theoretical framework is validated both on simulation and on the iCub humanoid. The software that implements the theoretical framework is realised with an open-source license.

## I. INTRODUCTION

In the field of humanoid robotics, computing the robot dynamics is a crucial component for modelling, estimation and control. Depending on the available data, computations might take the form of inverse dynamics, forward dynamics or hybrid dynamics. Computationally efficient solutions have been proposed in literature [1]: recursive Newton-Euler algorithm for solving inverse dynamics, articulated-body algorithm for forward dynamics and articulated-body hybrid dynamics for hybrid dynamics.

All these approaches have a rigid structure in the way inputs (i.e. measurements) and outputs (variables to be computed) are defined. As a first example, the inverse dynamics problem is about computing joint torques given joint positions, velocities and accelerations. As a second example, the forward dynamics problem consists in computing joint accelerations given joint positions, velocities and torques. Finally, the hybrid dynamics problem assumes that torques are known at some joints, accelerations at the rest, and the problem is to calculate the unknown torques and accelerations given joint positions and velocities.

Within this context, no appropriate method for computing robot dynamics in presence of redundant measurements is available. In this paper we propose the definition of a new problem, consisting in simultaneously estimating robot dynamics and robot state in presence of multiple, possibly redundant sensors. Considered types of sensors include accelerometers, gyroscopes, force/torque sensors and encoders.

This paper was supported by the FP7 EU projects CoDyCo (No. 600716 ICT 2011.2.1 Cognitive Systems and Robotics), and Koroibot (No. 611909 ICT-2013.2.1 Cognitive Systems and Robotics).

<sup>1</sup>Francesco Nori, Naveen Kuppuswamy and Silvio Traversaro are with the Robotics, Brain and Cognitive Sciences department at the Istituto Italiano di Tecnologia, 16163 Genoa, Italy. Contact email: name.surname@iit.it

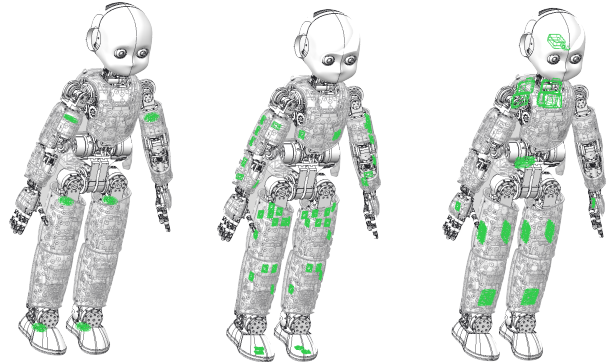


Fig. 1. Mechanical schemes of the humanoid robot iCub with gyroscopes, accelerometers and force/torque (F/T) sensors highlighted in green. **Left**: the position of the six proximal six-axis F/T sensors (legs and arms). **Centre**: the position of the skin micro-controllers, which have a 3D accelerometer embedded. **Right**: the position of the motor micro-controllers and the commercial inertial sensor which have a 3D gyroscope and a 3D accelerometer embedded.

A computationally efficient solution of the problem is proposed leveraging on some recent results presented in [16].

The paper is organised as follows: the first two sections give the necessary background in terms of previous literature (Section II) and notation (Section III). Section IV defines the dynamics estimation problem and Section V gives its solution. Section VI deals with the extension of previous sections to the case of simultaneous state and dynamics estimation. Finally, Section VII-A and Section VII-B present the simulation and experimental results.

## II. PREVIOUS WORKS

Different approaches have been proposed in literature to estimate position, velocity and acceleration of free-floating articulated rigid bodies. For sake of reducing computational complexity, previous approaches have dealt either with a simplified model of the system (e.g. linear inverted pendulum model, LIPM) or with a reduced (if not minimal) set of coordinates (e.g. joint coordinates).

In the present literature review we restrict our attention to the field of humanoids. Simplified linear inverted pendulum models have been addressed by [2] and [3]. In particular, [2] have proposed a LIPM to estimate some kinematic (centre of mass position and velocity) and some dynamic (centre of pressure and external forces) variables. The proposed estimation strategy deals with modelling errors such as measurement offsets and external disturbances. Focusing on estimating the centre of mass position and velocity, [3] proposes a quantitative evaluation of the estimation errors

induced by choosing a LIPM model as opposed to a full humanoid planar model.

More recent approaches have considered full-body models which guarantee the physical consistency of the resulting estimation. In this sense an interesting approach was proposed in [4]. The novelty of the approach consists in adding a penalty function for non-physically consistent predictions. Along the same line, [5] and [6] proposed algorithms to include kinematic constraints (such as footholds) to infer base motions given joint motions. Force sensors at the foot are typically used to detect contacts, assumed rigid and slipless. Another interesting approach is described in [7] where the estimation problem is framed in a quadratic programming framework which allows the inclusion of inequality constraints. Only recently, computationally efficient solutions have been proposed. As to this concern, in [8] the global estimation problem is decoupled into three sub-problems: base estimation, joint position estimation and joint velocity estimation. This is proven to simplify the computation burden of a full-body model estimation.

Recently, the use of gyroscopes and accelerometers have been proposed as a possible source of information to achieve reliable estimations. In [10], MEMS gyros and linear accelerometers have been used for angular acceleration estimation. Yet another approach to the problem of full-body kinematic estimation has been recently described in [9], where gyroscopes and accelerometers have been successfully used to estimate the deformations due to the robot flexibility at specific locations.

Differently from previous approaches such as [2], [3], [6], [8], we choose full coordinates as opposed to joint coordinates to describe the articulated rigid body chain. Despite the huge problem dimensionality, computational efficiency is maintained low by exploiting the problem sparsity [16]. Differently from all previous approaches, estimated variables includes both kinematic (joint positions and velocities) and dynamic (forces, torques and accelerations) variables. Similarly to [8] and [4] physical consistency is used to improve the estimation accuracy. As in [9], gyroscopes and accelerometers are included in the estimation but differently from all previous approaches we also use force/torque sensing. Experiments have been performed both in simulation and on the iCub, an ideal platform thanks to its whole-body distributed sensor modalities (see Fig. 1).

### III. NOTATION

In this paper we follow the same notation used in [1] (readers already familiar with the notation can jump to Section IV):  $\mathbf{v}$  denotes the spatial velocity (a six dimensional vector including angular velocities in the first three components and the rest as linear velocities),  $\mathbf{a}$  denotes the spatial acceleration (again angular and then linear),  $\mathbf{f}$  denotes the spatial force (couples in the first three components and forces in the remaining),  ${}^B\mathbf{X}_A$  is the motion vector transformation from  $A$  to  $B$  coordinates and  ${}^B\mathbf{X}_A^*$  is the analogous transformation for a force vector. Given a vector  $\mathbf{v}$  and its coordinates  ${}^A\mathbf{v}$  in  $A$  we denote with  ${}^A\dot{\mathbf{v}}$  its temporal

derivative in the  $A$  coordinates. The Euclidean cross operator  $\times$  (on  $\mathbb{R}^3$ ) is defined as the usual vector product, while for a spatial velocity  $\mathbf{v}$  the cross spatial cross operator and its dual  $\times^*$  are defined as follows:

$$\mathbf{v} \times = \begin{bmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{bmatrix} \times = \begin{bmatrix} \boldsymbol{\omega} \times & \mathbf{0} \\ \dot{\mathbf{p}} \times & \boldsymbol{\omega} \times \end{bmatrix}, \mathbf{v} \times^* = \begin{bmatrix} \boldsymbol{\omega} \\ \dot{\mathbf{p}} \end{bmatrix} \times^* = \begin{bmatrix} \boldsymbol{\omega} \times & \dot{\mathbf{p}} \times \\ \mathbf{0} & \boldsymbol{\omega} \times \end{bmatrix}$$

In the following sections, we restrict the analysis to robots described by kinematic trees (i.e. robots with no kinematic loops) composed of  $N_B$  rigid links numbered from 0 to  $N_B$ , 0 being the selected fixed base (world reference frame) and 1 being the selected floating base (reference rigid body in the articulated chain). Links numbering is obtained by defining a suitable spanning tree where each rigid link is associated to a unique node in the tree. Numbers can be always selected in a topological order so that each node  $i$  has a higher number than its unique parent  $\lambda(i)$  and a smaller number than all the nodes in the set of its children  $\mu(i)$ . Links  $i$  and  $\lambda(i)$  are coupled with the joint  $i$  whose joint motion constraints are modelled with  $\mathbf{S}_i \in \mathbb{R}^{6 \times 1}$  (therefore without loss of generality we are assuming that all joints have a single degree of freedom). For each rigid link  $i$ , the system is modelled also by supplying the spatial inertia tensor<sup>1</sup>  $\mathbf{I}_i$  and the motion-vector transform from the reference frame of the rigid link  $i$  to the reference frame of the rigid link  $j$ , denoted  ${}^j\mathbf{X}_i$ . For each link  $i$  the considered kinematic variables are:

- $\mathbf{v}_i$ : the link spatial velocity,
- $\mathbf{q}_i$ : the joint  $i$  position,
- $\dot{\mathbf{q}}_i$ : the joint  $i$  velocity,

Similarly, the dynamic variables associated to the link  $i$  are:

- $\mathbf{a}_i$ : the link spatial accelerations,
- $\ddot{\mathbf{q}}_i$ : the joint  $i$  acceleration,
- $\boldsymbol{\tau}_i$ : the joint  $i$  torque,
- $\mathbf{f}_i$ : the spatial force transmitted to body  $i$  from  $\lambda(i)$ ,
- $\mathbf{f}_i^x$ : external forces acting on body  $i$ .

All these variables are expressed in body  $i$  coordinates including  $\mathbf{f}_i^x$  which is more often expressed in absolute (i.e. body 0) coordinates (see also the appendix).

Within the stochastic context we adopted the following notation. Given a stochastic vector  $\mathbf{x}$  we denote with  $p(\mathbf{x})$  its probability density and with  $p(\mathbf{x}|\mathbf{y})$  the probability density of  $\mathbf{x}$  conditioned on the stochastic vector  $\mathbf{y}$ . With  $E_{\mathbf{x}}[f(\mathbf{x})]$  we denoted the expected value of  $f(\mathbf{x})$  with respect to the probability distribution  $p(\mathbf{x})$ . With  $\mu_{\mathbf{x}}$ ,  $\Sigma_{\mathbf{x}}$  we denote the mean and covariance of  $\mathbf{x}$ , i.e.  $\mu_{\mathbf{x}} = E[\mathbf{x}]$  and  $\Sigma_{\mathbf{x}} = E[\mathbf{x}\mathbf{x}^T]$ . The probability density function of multivariate Gaussian distribution  $\mathbf{x} \in \mathbb{R}^n$  is indicated  $p(\mathbf{x}) \sim \mathcal{N}(\mu_{\mathbf{x}}, \Sigma_{\mathbf{x}})$ :

$$p(\mathbf{x}) = \frac{|\Sigma_{\mathbf{x}}|^{-\frac{1}{2}}}{(2\pi)^{\frac{n}{2}}} \exp \left\{ -\frac{1}{2} (\mathbf{x} - \mu_{\mathbf{x}})^T \Sigma_{\mathbf{x}}^{-1} (\mathbf{x} - \mu_{\mathbf{x}}) \right\},$$

<sup>1</sup>The spatial inertia tensor of the rigid link  $i$  has the following form in the link reference frame:

$$\mathbf{I}_i = \begin{bmatrix} \mathbf{I}_{C,i} + m_i \mathbf{c}_i \times \mathbf{c}_i^T & m_i \mathbf{c}_i \times \\ m_i \mathbf{c}_i \times^T & m_i \mathbf{I}_{3 \times 3} \end{bmatrix},$$

where  $\mathbf{I}_{C,i}$  is the spatial inertia tensor with respect to the body's centre of mass,  $m_i$  is the total mass,  $\mathbf{c}_i$  is the relative displacement between the centre of mass and the origin of the link reference frame.

where  $|\Sigma_x|$  denotes the determinant of the matrix  $\Sigma_x$ . Finally, with  $\mathbf{1}$  and  $\mathbf{0}$  we will denote the identity and the null matrices respectively with suitable dimensions depending on the context.

#### IV. PROBLEM FORMULATION

In this section we revise the classical inverse and forward dynamics problems. Similar redefinitions can be used in the hybrid dynamics problem but for sake of simplicity we leave this case out. In this section we assume that the robot state (position  $\mathbf{q}$  and velocity  $\dot{\mathbf{q}}$ ) are given, i.e. measured without uncertainty. The extension beyond this hypothesis will be given in the Section VI.

##### A. Forward and inverse dynamics

Let's recall the inverse and forward problems definitions.

*Problem 1 (Inverse dynamics):* Compute the joint torques  $\boldsymbol{\tau}$  of a kinematic tree given its joint accelerations  $\ddot{\mathbf{q}}$  and applied external forces  $\mathbf{f}_1^x, \dots, \mathbf{f}_{N_B}^x$ .

*Problem 2 (Forward dynamics):* Compute joint accelerations  $\ddot{\mathbf{q}}$  of a kinematic tree given its joint torques  $\boldsymbol{\tau}$  and applied external forces  $\mathbf{f}_1^x, \dots, \mathbf{f}_{N_B}^x$ .

A classical and computationally efficient solution to the inverse dynamics problem (*Problem 1*) is the recursive Newton-Euler algorithm (RNEA) which also computes  $\mathbf{f}_1, \dots, \mathbf{f}_{N_B}$  and  $\mathbf{a}_1, \dots, \mathbf{a}_{N_B}$  as a by-product of the algorithm. It makes therefore sense to include those variables in the above problem formulations. In order to do so, let's define the vector  $\mathbf{d}_i \in \mathbb{R}^{d_i}$  and  $\mathbf{d} \in \mathbb{R}^d$  that includes all relevant variables:

$$\mathbf{d}_i = [\mathbf{a}_i^\top \quad \boldsymbol{\tau}_i^\top \quad \mathbf{f}_i^\top \quad \mathbf{f}_i^{x\top} \quad \ddot{\mathbf{q}}_i^\top]^\top, \quad (1)$$

$$\mathbf{d} = [\mathbf{d}_1^\top \quad \mathbf{d}_2^\top \quad \dots \quad \mathbf{d}_{N_B}^\top]^\top, \quad (2)$$

whose dimensionality is  $d_i = 20$  and  $d = 20N_B$  respectively. Remarkably,  $\mathbf{d}$  should be physically consistent, i.e. it should satisfy a set of constraints which derive from the structure of the robot. These constraints take the following form (see the appendix):

$$\mathbf{D}(\mathbf{q})\mathbf{d} + \mathbf{b}_D(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}, \quad (3)$$

with  $\mathbf{D} \in \mathbb{R}^{13N_B \times d}$  and  $\mathbf{b}_D \in \mathbb{R}^{13N_B}$ .

In the following, we slightly modify the inverse and forward dynamic problems as defined above. Rather than focusing on computing  $\boldsymbol{\tau}$  (inverse dynamics) or  $\ddot{\mathbf{q}}$  (forward dynamics) we reinterpret the inverse and forward problems as particular instances of the general problem of computing a physically consistent  $\mathbf{d}$  which also satisfies a set of measurement equations defined as follows:

$$\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{d} + \mathbf{b}_Y(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{y}. \quad (4)$$

In the inverse problem, the measurement  $\mathbf{y}$  has the following form:  $\mathbf{y} = [\ddot{\mathbf{q}}, \mathbf{f}_1^x, \dots, \mathbf{f}_{N_B}^x]$ . In a forward problem instead  $\mathbf{y} = [\boldsymbol{\tau}, \mathbf{f}_1^x, \dots, \mathbf{f}_{N_B}^x]$ . Both measurements can be expressed

as in (4) choosing  $\mathbf{b}_Y = \mathbf{0}$  and defining suitable matrices  $\mathbf{Y}_{inv} \in \mathbb{R}^{7N_B \times d}$  and  $\mathbf{Y}_{fwd} \in \mathbb{R}^{7N_B \times d}$  such that:

$$\mathbf{Y}_{inv}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{d} = \begin{bmatrix} \ddot{\mathbf{q}} \\ \mathbf{f}_1^x \\ \vdots \\ \mathbf{f}_{N_B}^x \end{bmatrix}, \quad \mathbf{Y}_{fwd}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{d} = \begin{bmatrix} \boldsymbol{\tau} \\ \mathbf{f}_1^x \\ \vdots \\ \mathbf{f}_{N_B}^x \end{bmatrix}.$$

Within this context the forward and inverse problems can be reformulated as follows.

*Problem 3 (Inverse dynamics):* Given  $\mathbf{y}$ , compute  $\mathbf{d}$  which satisfies (3) and (4) with  $\mathbf{Y} = \mathbf{Y}_{inv}$  and  $\mathbf{b}_Y = \mathbf{0}$ .

*Problem 4 (Forward dynamics):* Given  $\mathbf{y}$ , compute  $\mathbf{d}$  which satisfies (3) and (4) with  $\mathbf{Y} = \mathbf{Y}_{fwd}$  and  $\mathbf{b}_Y = \mathbf{0}$ .

*Note:* originally conceived for solving Problem 1, the RNEA is also a way to solve Problem 3 since it computes a physically consistent  $\mathbf{d}$  as by-product of the procedure for computing  $\boldsymbol{\tau}$ . On the other side, any solution to Problem 2 can be extended to a solution of Problem 4. As a matter of fact, the outcome  $\ddot{\mathbf{q}}$  of the solution to Problem 2 can be used in the RNEA to obtain a physically consistent  $\mathbf{d}$ .

##### B. Maximum-a-posteriori dynamics

In the previous section we have seen that the inverse and forward dynamics problems can be seen as specific instances of a more general problem consisting in estimating a vector  $\mathbf{d}$  of dynamic variables which simultaneously satisfies (3) and (4):

$$\begin{bmatrix} \mathbf{D}(\mathbf{q}) \\ \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}) \end{bmatrix} \mathbf{d} + \begin{bmatrix} \mathbf{b}_D(\mathbf{q}, \dot{\mathbf{q}}) \\ \mathbf{b}_Y(\mathbf{q}, \dot{\mathbf{q}}) \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{y} \end{bmatrix}. \quad (5)$$

In the inverse and forward cases the matrix  $\mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}})$  has a specific structure. In particular, the structure is such that there exists a unique solution  $\mathbf{d}$  of (5) for any given  $\mathbf{q}$ ,  $\dot{\mathbf{q}}$  and  $\mathbf{y}$ . In practice, it can be shown that the matrices  $[\mathbf{D}; \mathbf{Y}_{inv}]$  and  $[\mathbf{D}; \mathbf{Y}_{fwd}]$  are both square and invertible. Procedures like the recursive Newton-Euler algorithm and the articulated-body algorithm rely on analytic *LU* factorisations of  $[\mathbf{D}; \mathbf{Y}]$  to simplify the computational cost of computing the solution of (5) in the forward and inverse cases respectively.

In this section we define a different problem by relaxing the assumptions on the structure of the matrices in (5). In particular we are interested in situations which involve redundant measurements. Therefore, instead of assuming  $[\mathbf{D}; \mathbf{Y}]$  square, we require its full-column rankness. Under this new hypothesis,  $\mathbf{d}$  is over constrained and an exact solution of (5) does not always exist. An ‘‘approximated’’ solution can be obtained only by making some additional hypotheses. If we assume that all the constraints have equal relevance and all measurements the same accuracy, we can use the Moore-Penrose pseudo-inverse to obtain the least squares solution. If we have good reasons for weighting differently constraints and measurements, we can use the weighted pseudo-inverse to obtain the weighted least squares solution. We propose a different solution which consists in framing the estimation of  $\mathbf{d}$  in a Gaussian Bayesian framework. Within the proposed context, computational efficiency can be maintained by representing the sparsity in  $\mathbf{D}$  and

$\mathbf{Y}$  either with a Bayesian network or with a sparse matrix representation [16].

The idea is to build a joint probability density distribution for the  $\mathbf{d}$  and  $\mathbf{y}$ . This probability distribution should embed the structure represented in (5). We specify this probability using the factorisation  $p(\mathbf{d}, \mathbf{y}) = p(\mathbf{d})p(\mathbf{y}|\mathbf{d})$ . Let's first give an expression for  $p(\mathbf{y}|\mathbf{d})$ :

$$p(\mathbf{y}|\mathbf{d}) \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y), \quad \boldsymbol{\mu}_y = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{d} + \mathbf{b}_Y(\mathbf{q}, \dot{\mathbf{q}}), \quad (6)$$

which implicitly makes the assumption that the measurement equation  $\mathbf{y} = \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}})\mathbf{d} + \mathbf{b}_Y(\mathbf{q}, \dot{\mathbf{q}})$  is affected by a Gaussian noise with zero mean and covariance  $\boldsymbol{\Sigma}_y$ . The expression for  $p(\mathbf{d})$  is a little bit more complex. Ideally, we would like to have:

$$p(\mathbf{d}) \propto \exp \{ \mathbf{e}(\mathbf{d})^\top \boldsymbol{\Sigma}_D^{-1} \mathbf{e}(\mathbf{d}) \}, \quad \mathbf{e}(\mathbf{d}) = \mathbf{D}(\mathbf{q})\mathbf{d} + \mathbf{b}_D(\mathbf{q}, \dot{\mathbf{q}}),$$

where the matrix  $\boldsymbol{\Sigma}_D$  defines how much the dynamical model (16) is reliable. Such a choice for  $p(\mathbf{d})$  would lead to a degenerate Gaussian distribution<sup>2</sup> unless we add some prior on  $\mathbf{d}$ . Assuming a prior  $\mathcal{N}(\boldsymbol{\mu}_d, \boldsymbol{\Sigma}_d)$  we therefore define:

$$p(\mathbf{d}) \propto \exp -\frac{1}{2} \left\{ \mathbf{e}(\mathbf{d})^\top \boldsymbol{\Sigma}_D^{-1} \mathbf{e}(\mathbf{d}) + (\mathbf{d} - \boldsymbol{\mu}_d)^\top \boldsymbol{\Sigma}_d^{-1} (\mathbf{d} - \boldsymbol{\mu}_d) \right\}.$$

Formally, the previous definitions lead to a proper probability density  $p(\mathbf{d}) \sim \mathcal{N}(\bar{\boldsymbol{\mu}}_D, \bar{\boldsymbol{\Sigma}}_D)$  with:

$$\begin{aligned} \bar{\boldsymbol{\Sigma}}_D &= (\mathbf{D}^\top \boldsymbol{\Sigma}_D^{-1} \mathbf{D} + \boldsymbol{\Sigma}_d^{-1})^{-1}, \\ \bar{\boldsymbol{\mu}}_D &= \bar{\boldsymbol{\Sigma}}_D (\boldsymbol{\Sigma}_d^{-1} \boldsymbol{\mu}_d - \mathbf{D}^\top \boldsymbol{\Sigma}_D^{-1} \mathbf{b}_D). \end{aligned} \quad (7)$$

With the definitions above we are now ready to give a new formulation of the estimation problem. In practice, we base the new formulation on the probability distribution  $p(\mathbf{d}, \mathbf{y})$  which we just derived from (6) and (7).

*Problem 5 (Maximum-a-posteriori dynamics):* Compute  $\mathbf{d}_{map}$ , the maximum a posteriori estimation of  $\mathbf{d}$  given the measurements  $\mathbf{y}$ .

$$\mathbf{d}_{map} = \arg \max_{\mathbf{d}} p(\mathbf{d}|\mathbf{y}).$$

## V. MAXIMUM-A-POSTERIORI DYNAMICS ESTIMATION

In the proposed Gaussian context,  $\mathbf{d}_{map}$  can be computed analytically. The idea is to first compute the probability distribution of  $\mathbf{d}|\mathbf{y}$ , which can be proven to be Gaussian itself. In particular we have  $p(\mathbf{d}|\mathbf{y}) \sim \mathcal{N}(\boldsymbol{\mu}_{d|y}, \boldsymbol{\Sigma}_{d|y})$  with:

$$\boldsymbol{\Sigma}_{d|y} = (\boldsymbol{\Sigma}_D^{-1} + \mathbf{Y}^\top \boldsymbol{\Sigma}_y^{-1} \mathbf{Y})^{-1}, \quad (8a)$$

$$\boldsymbol{\mu}_{d|y} = \boldsymbol{\Sigma}_{d|y} [\mathbf{Y}^\top \boldsymbol{\Sigma}_y^{-1} (\mathbf{y} - \mathbf{b}_Y) + \boldsymbol{\Sigma}_D^{-1} \bar{\boldsymbol{\mu}}_D]. \quad (8b)$$

Moreover, in the Gaussian case the maximum a posteriori coincides with the mean of the posterior distribution  $p(\mathbf{d}|\mathbf{y})$  so that we have:

$$\mathbf{d}_{map} = \boldsymbol{\mu}_{d|y}. \quad (9)$$

Computations can be performed in various ways but computationally efficient solutions need to exploit the sparsity in the matrices  $\mathbf{D}$  and  $\mathbf{Y}$  as discussed in [16].

<sup>2</sup>[http://en.wikipedia.org/wiki/Multivariate\\_normal\\_distribution#Degenerate\\_case](http://en.wikipedia.org/wiki/Multivariate_normal_distribution#Degenerate_case).

## VI. MAXIMUM-A-POSTERIORI DYNAMICS AND STATE ESTIMATION

In the previous sections we focused on estimating  $\mathbf{d}$  while assuming  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  measured without uncertainty. In this section we discuss how to include  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  in the estimation. The motivation behind this extension are twofold. On one side, it is now always true that  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  can be measured. On the other side, some measurements are functions of  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  but do not depend explicitly on  $\mathbf{d}$ . Gyroscopes are an example of sensors which can not be fruitfully integrated in the estimation of  $\mathbf{d}$  proposed in the previous sections.

Including  $\mathbf{x} = [\mathbf{q}; \dot{\mathbf{q}}]$  in the estimation is non-trivial because it appears non-linearly in the equations (5). The idea is to first order approximate these equations around the mean of prior distributions on  $\mathbf{d}$  and  $\mathbf{x}$ , denoted  $\bar{\mathbf{d}}$  and  $\bar{\mathbf{x}}$ . We have:

$$\begin{bmatrix} \mathbf{b}_D(\bar{\mathbf{x}}) \\ \mathbf{b}_Y(\bar{\mathbf{x}}) \end{bmatrix} + \begin{bmatrix} \mathbf{D}(\bar{\mathbf{x}}) \\ \mathbf{Y}(\bar{\mathbf{x}}) \end{bmatrix} \mathbf{d} + \begin{bmatrix} \partial \mathbf{b}_D(\bar{\mathbf{d}}, \bar{\mathbf{x}}) \\ \partial \mathbf{b}_Y(\bar{\mathbf{d}}, \bar{\mathbf{x}}) \end{bmatrix} (\mathbf{x} - \bar{\mathbf{x}}) = \begin{bmatrix} \mathbf{0} \\ \mathbf{y} \end{bmatrix}, \quad (10)$$

where we defined the following quantities:

$$\partial \mathbf{b}_D(\bar{\mathbf{d}}, \bar{\mathbf{x}}) = \frac{\partial}{\partial \mathbf{x}} [\mathbf{D}(\mathbf{x})\bar{\mathbf{d}} + \mathbf{b}_D(\mathbf{x})]_{\mathbf{x}=\bar{\mathbf{x}}}, \quad (11)$$

$$\partial \mathbf{b}_Y(\bar{\mathbf{d}}, \bar{\mathbf{x}}) = \frac{\partial}{\partial \mathbf{x}} [\mathbf{Y}(\mathbf{x})\bar{\mathbf{d}} + \mathbf{b}_Y(\mathbf{x})]_{\mathbf{x}=\bar{\mathbf{x}}}. \quad (12)$$

Rearranging (10), we can give it the same structure of (5):

$$\begin{bmatrix} \mathbf{D}(\bar{\mathbf{x}}) & \partial \mathbf{b}_D(\bar{\mathbf{d}}, \bar{\mathbf{x}}) \\ \mathbf{Y}(\bar{\mathbf{x}}) & \partial \mathbf{b}_Y(\bar{\mathbf{d}}, \bar{\mathbf{x}}) \end{bmatrix} \begin{bmatrix} \mathbf{d} \\ \mathbf{x} \end{bmatrix} + \begin{bmatrix} \mathbf{b}_D(\bar{\mathbf{x}}) - \partial \mathbf{b}_D(\bar{\mathbf{d}}, \bar{\mathbf{x}})\bar{\mathbf{x}} \\ \mathbf{b}_Y(\bar{\mathbf{x}}) - \partial \mathbf{b}_Y(\bar{\mathbf{d}}, \bar{\mathbf{x}})\bar{\mathbf{x}} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{y} \end{bmatrix}$$

In practice, all considerations in Section IV-B and Section V can be repeated, this time the vector  $\mathbf{d}$  being substituted by  $[\mathbf{d}; \mathbf{x}]$ . A probability density  $p(\mathbf{y}|\mathbf{d}, \mathbf{x})$  can be defined as in (6) by replacing:

$$\begin{aligned} \mathbf{Y}(\mathbf{q}, \dot{\mathbf{q}}) &\leftrightarrow [\mathbf{Y}(\bar{\mathbf{x}}) \quad \partial \mathbf{b}_Y(\bar{\mathbf{d}}, \bar{\mathbf{x}})], \\ \mathbf{b}_Y(\mathbf{q}, \dot{\mathbf{q}}) &\leftrightarrow [\mathbf{b}_Y(\bar{\mathbf{x}}) - \partial \mathbf{b}_Y(\bar{\mathbf{d}}, \bar{\mathbf{x}})\bar{\mathbf{x}}]. \end{aligned}$$

A probability density  $p(\mathbf{d}, \mathbf{x})$  can be defined as in (7) by replacing:

$$\begin{aligned} \mathbf{D}(\mathbf{q}) &\leftrightarrow [\mathbf{D}(\bar{\mathbf{x}}) \quad \partial \mathbf{b}_D(\bar{\mathbf{d}}, \bar{\mathbf{x}})], \\ \mathbf{b}_D(\mathbf{q}, \dot{\mathbf{q}}) &\leftrightarrow [\mathbf{b}_D(\bar{\mathbf{x}}) - \partial \mathbf{b}_D(\bar{\mathbf{d}}, \bar{\mathbf{x}})\bar{\mathbf{x}}], \\ \boldsymbol{\mu}_d &\leftrightarrow [\boldsymbol{\mu}_d; \boldsymbol{\mu}_x], \\ \boldsymbol{\Sigma}_d &\leftrightarrow \begin{bmatrix} \boldsymbol{\Sigma}_d & \mathbf{0} \\ \mathbf{0} & \boldsymbol{\Sigma}_x \end{bmatrix}, \end{aligned}$$

being  $\mathcal{N}(\boldsymbol{\mu}_x, \boldsymbol{\Sigma}_x)$  the prior distribution on  $\mathbf{x}$ . The resulting joint probability distribution  $p(\mathbf{d}, \mathbf{x}, \mathbf{y})$  can be used to define a new maximum a posteriori estimation problem.

*Problem 6 (Maximum-a-posteriori state and dynamics):* Compute  $\mathbf{d}_{map}$ ,  $\mathbf{x}_{map}$  the maximum a posteriori estimation of  $\mathbf{d}$  and  $\mathbf{x}$  given the measurements  $\mathbf{y}$ .

$$[\mathbf{d}_{map}, \mathbf{x}_{map}] = \arg \max_{\mathbf{d}, \mathbf{x}} p(\mathbf{d}, \mathbf{x}|\mathbf{y}).$$

Also in this case the solution is given by (8) applying all the substitution above.

### A. Analytic derivative computations

In this section we describe how to analytically compute the matrix  $\partial \mathbf{b}_D$ . For convenience we compute the derivative one component at a time, by deriving the  $i$ -th element of the vector  $\mathbf{D}(\mathbf{x})\bar{\mathbf{d}} + \mathbf{b}_D(\mathbf{x})$ , denoted  $[\mathbf{D}(\mathbf{x})\bar{\mathbf{d}} + \mathbf{b}_D(\mathbf{x})]_i$ . Similarly  $\mathbf{D}_{ij}$  is used to denote the element  $(i, j)$  of  $\mathbf{D}$  and  $\mathbf{b}_{D,i}$  to denote the  $i$ -th element of  $\mathbf{b}_D$ . With this notation we have:

$$\frac{\partial}{\partial q_h} [\mathbf{D}(\mathbf{x})\bar{\mathbf{d}} + \mathbf{b}_D(\mathbf{x})]_i = \sum_{j=1}^{N_B} \frac{\partial \mathbf{D}_{ij}}{\partial q_h} \bar{d}_j + \frac{\partial \mathbf{b}_{D,i}}{\partial q_h}.$$

For the joint velocity derivatives we can exploit the fact that  $\mathbf{D}$  only depends on  $\mathbf{q}$  to obtain:

$$\frac{\partial}{\partial \dot{q}_h} [\mathbf{D}(\mathbf{x})\bar{\mathbf{d}} + \mathbf{b}_D(\mathbf{x})]_i = \frac{\partial \mathbf{b}_{D,i}}{\partial \dot{q}_h}.$$

The derivation is therefore translated into deriving the elements  $\mathbf{D}_{ij}$  and  $\mathbf{b}_{D,i}$  with respect to  $q_h$  and  $\dot{q}_h$ . Given the structure described in the appendix, derivations boil down to the problem of computing the derivatives of  ${}^i\mathbf{X}_{\lambda(i)}$ ,  ${}^i\mathbf{X}_j^*$  with  $j \in \mu(i)$ ,  $\mathbf{v}_i \times \mathbf{S}_i \dot{\mathbf{q}}_i$  and  $\mathbf{v}_i \times {}^i\mathbf{I}_i \mathbf{v}_i$ . In the following sections we describe how to analytically compute these derivatives.

1)  ${}^i\mathbf{X}_{\lambda(i)}$  derivatives: in this section we compute the derivative of  ${}^i\mathbf{X}_{\lambda(i)}$  with respect to  $q_h$  and  $\dot{q}_h$ . In general, this transformation takes a different form depending on the type of joint (see [1] Table 4.1). For simplicity we limit our notation to revolute joints around the  $z$ -axis. In this case we have:

$${}^i\mathbf{X}_{\lambda(i)} = \begin{bmatrix} R_z(q_i) & \mathbf{0} \\ \mathbf{0} & R_z(q_i) \end{bmatrix}, R_z(q_i) = \begin{bmatrix} c_{q_i} & s_{q_i} & 0 \\ -s_{q_i} & c_{q_i} & 0 \\ 0 & 0 & 1 \end{bmatrix},$$

being  $s_{q_i}$  and  $c_{q_i}$  the sine and cosine functions respectively. Clearly, the derivative of  ${}^i\mathbf{X}_{\lambda(i)}$  with respect to  $q_h$  is non-null only if  $i = h$ . In this case we have:

$$\frac{\partial {}^i\mathbf{X}_{\lambda(i)}}{\partial q_i} = \begin{bmatrix} \partial R_z(q_i) & \mathbf{0} \\ \mathbf{0} & \partial R_z(q_i) \end{bmatrix},$$

$$\partial R_z(q_i) = \begin{bmatrix} -s_{q_i} & c_{q_i} & 0 \\ -c_{q_i} & -s_{q_i} & 0 \\ 0 & 0 & 1 \end{bmatrix}.$$

2)  ${}^i\mathbf{X}_j^*$  derivatives: in computing the derivative of  ${}^i\mathbf{X}_j^*$  we exploit the fact that  $j \in \mu(i)$ , i.e.  $j$  is a children of  $i$ . In other terms  $i$  is the parent of  $j$ , i.e.  ${}^i\mathbf{X}_j^* = {}^{\lambda(j)}\mathbf{X}_j^*$ . Recalling that  ${}^i\mathbf{X}_j^* = {}^i\mathbf{X}_j^{-\top} = {}^j\mathbf{X}_i^\top$ , we have:

$${}^i\mathbf{X}_j^* = {}^j\mathbf{X}_{\lambda(j)}^\top \Rightarrow \frac{\partial {}^i\mathbf{X}_j^*}{\partial q_j} = \left[ \frac{\partial {}^j\mathbf{X}_{\lambda(j)}}{\partial q_j} \right]^\top.$$

3)  $\mathbf{v}_i$  derivatives: the following computations make use of the set  $\kappa(i)$  which contains the ancestors of node  $i$  in the spanning tree used to describe the robot. Looking at the structure of equation (14) which recursively defines  $\mathbf{v}_i$ , it can be concluded that:

$$\frac{\partial \mathbf{v}_i}{\partial q_h} = \begin{cases} \mathbf{0} & h \notin \kappa(i) \\ {}^i\mathbf{X}_h \frac{\partial {}^h\mathbf{X}_{\lambda(h)}}{\partial q_h} \mathbf{v}_{\lambda(h)} & h \in \kappa(i) \end{cases},$$

and:

$$\frac{\partial \mathbf{v}_i}{\partial \dot{q}_h} = \begin{cases} \mathbf{0} & h \notin \kappa(i) \\ {}^i\mathbf{X}_h \mathbf{S}_h & h \in \kappa(i) \end{cases}.$$

4) *Cross product derivatives*: taking advantage of the derivatives computed above we are left with computing the following cross products:

$$\begin{aligned} \frac{\partial}{\partial q_h} [\mathbf{v}_i \times \mathbf{S}_i \dot{\mathbf{q}}_i] &= \frac{\partial \mathbf{v}_i}{\partial q_h} \times \mathbf{S}_i \dot{\mathbf{q}}_i, \\ \frac{\partial}{\partial q_h} [\mathbf{v}_i \times {}^i\mathbf{I}_i \mathbf{v}_i] &= \frac{\partial \mathbf{v}_i}{\partial q_h} \times {}^i\mathbf{I}_i \mathbf{v}_i + \mathbf{v}_i \times {}^i\mathbf{I}_i \frac{\partial \mathbf{v}_i}{\partial q_h}, \\ \frac{\partial}{\partial \dot{q}_h} [\mathbf{v}_i \times \mathbf{S}_i \dot{\mathbf{q}}_i] &= \frac{\partial \mathbf{v}_i}{\partial \dot{q}_h} \times \mathbf{S}_i \dot{\mathbf{q}}_i + \mathbf{v}_i \times \mathbf{S}_i \delta_{i,h}, \\ \frac{\partial}{\partial \dot{q}_h} [\mathbf{v}_i \times {}^i\mathbf{I}_i \mathbf{v}_i] &= \frac{\partial \mathbf{v}_i}{\partial \dot{q}_h} \times {}^i\mathbf{I}_i \mathbf{v}_i + \mathbf{v}_i \times {}^i\mathbf{I}_i \frac{\partial \mathbf{v}_i}{\partial \dot{q}_h}, \end{aligned}$$

where  $\delta_{i,h}$  is the Kronecker delta.

## VII. RESULTS

We tested the proposed estimation algorithm both in simulation and on a real robot. As it was previously pointed out, the estimation relies on a linear approximation of the non-linear constraints. Therefore, even in simulation where the noise level can be arbitrarily reduced, no exact estimation will be obtained in a single step. Exact estimation would require an iterative procedure which is outside the scope of the present paper. As to this concern, purpose of the simulation and real robot experiments was to validate the accuracy of the simultaneous dynamics ( $\mathbf{d}$ ) and state ( $\mathbf{x}$ ) estimation. It is worth noticing here that the MATLAB software for performing all the computations described above and the experiments presented below is freely available under an open source license<sup>3</sup>.

### A. Simulation results

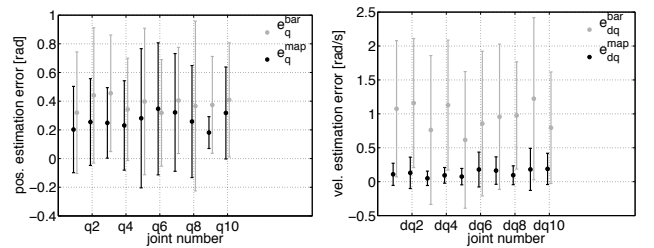


Fig. 2. The figure shows with error bars the mean and standard deviation of errors in estimating  $\mathbf{q}$  and  $\dot{\mathbf{q}}$ . In grey the initial errors:  $e_q^{bar} = \bar{\mathbf{q}} - \mathbf{q}$  (left) and  $e_{\dot{q}}^{bar} = \bar{\dot{\mathbf{q}}} - \dot{\mathbf{q}}$  (right). In black the errors after the maximum-a-posteriori estimation:  $e_q^{map} = \mathbf{q}_{map} - \mathbf{q}$  (left) and  $e_{\dot{q}}^{map} = \dot{\mathbf{q}}_{map} - \dot{\mathbf{q}}$  (right).

In the proposed simulations we considered a ten degrees of freedom ( $N_B = 10$ ,  $\mathbf{x} \in \mathbb{R}^{20}$ ,  $\mathbf{d} \in \mathbb{R}^{200}$ ) planar chain composed of rotational joints and subject to a gravitational field. The system dynamics were randomly generated using the MATLAB toolbox released by Roy Featherstone<sup>4</sup>. The

<sup>3</sup>[https://github.com/iron76/bnt\\_time\\_varying](https://github.com/iron76/bnt_time_varying)

<sup>4</sup>Spatial vector and rigid-body dynamics software toolbox available at <http://royfeatherstone.org/spatial/index.html>.

sensors were also randomly generated in a number of  $4N_B$ , with the constraint of having at least one gyroscope and one linear accelerometer at each link. Possible measurements include internal forces and torques ( $\mathbf{f}_i$ ), link accelerations ( $\mathbf{a}_i$ ), link velocities ( $\mathbf{v}_i$ ) and joint torques ( $\tau_i$ ). In the proposed example, no direct measurement of  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  is available but gyroscopes and linear accelerometers provide an indirect measure of the system state. Before estimation, the values of  $\bar{\mathbf{d}}$  and  $\bar{\mathbf{x}}$  were chosen by corrupting the real values ( $\mathbf{x}$  and  $\mathbf{d}$ ) with some noise:  $\bar{\mathbf{x}} = \mathbf{x} + \mathbf{e}_x$ ,  $\bar{\mathbf{d}} = \mathbf{d} + \mathbf{e}_d$ . Noise  $\mathbf{e}_x$  was chosen as a uniformly distributed random vector in the interval  $[0, 0.8][rad]$  for joint positions and  $[0, 1.8][rad/s]$  for joint velocities. The vector  $\mathbf{e}_d$  was also uniformly distributed in the interval  $[0, 100]$  in the suitable unit of measurement. Estimations have been performed ten times, and errors computed according to the following definitions: (1) error on  $\mathbf{q}$  before ( $e_q^{bar} = \bar{\mathbf{q}} - \mathbf{q}$ ) and after ( $e_q^{map} = \mathbf{q}_{map} - \mathbf{q}$ ) the maximum a posteriori estimation; (2) error on  $\dot{\mathbf{q}}$  before ( $e_{\dot{q}}^{bar} = \bar{\dot{\mathbf{q}}} - \dot{\mathbf{q}}$ ) and after ( $e_{\dot{q}}^{map} = \dot{\mathbf{q}}_{map} - \dot{\mathbf{q}}$ ) the maximum a posteriori estimation; (3) error on  $\mathbf{d}$  before ( $e_d^{bar} = \bar{\mathbf{d}} - \mathbf{d}$ ) and after ( $e_d^{map} = \mathbf{d}_{map} - \mathbf{d}$ ) the maximum a posteriori estimation. Results with the associated mean ( $\mu$ ) and standard deviation ( $\sigma$ ) are reported in Figure 2 and in the following table.

	$\ e_q^{map}\ $	$\ e_q^{bar}\ $	$\ e_{\dot{q}}^{map}\ $	$\ e_{\dot{q}}^{bar}\ $	$\ e_d^{map}\ $	$\ e_d^{bar}\ $
$\mu$	0.99	1.37	0.51	3.44	158.39	928.07
$\sigma$	0.21	0.25	0.14	0.39	44.76	20.61

### B. Experimental results

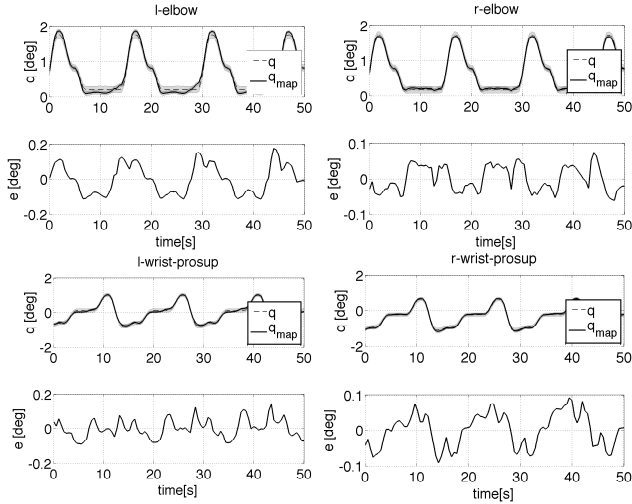


Fig. 3. The figure shows the accuracy in estimating the elbow (two top rows) and the wrist (two bottom lines) positions. Maximum-a-posteriori estimation (see Section VI) were obtained without using the associated encoders, which have been used only as a ground-truth for determining the estimation accuracy.

We tested the proposed estimation procedure on the iCub humanoid. The robot has 53 degrees of freedom, 25 of which were used in the present experiment ( $N_B = 25$ ,  $\mathbf{d} \in \mathbb{R}^{500}$ ). Considered joints are those of the arms (5), legs (6) and pelvis (3). The iCub is equipped with whole-body distributed sensors (see Fig. 1) but in the proposed experiment we used

a subset of them: some linear accelerometers (8 in total: one per foot, one per hand, one in the head, one in the torso, one on each upper arm), gyroscopes (3 in total: one per hand and one in the head), all the available encoders (25) and all the force/torque sensors (6 in total: one per foot, one per upper arm and one per thigh)<sup>5</sup>. Considering that during the experiment no external force was applied to the robot, all  $\mathbf{f}_i^x$  were null and this was simulated by associating a null measurement with small variance for all  $\mathbf{f}_i^x$ . The resulting measurement vector  $\mathbf{y}$  consists of 244 measurements, i.e.  $\mathbf{y} \in \mathbb{R}^{244}$ . Computations of the maximum a posteriori dynamics  $p(\mathbf{d}, \mathbf{x} | \mathbf{y})$  took less than 100ms on a 3GHz Intel Core i7. The MATLAB code used for data processing is available on <https://github.com> and released with an open source license.

The dataset used in the experiment consists of 50 seconds of repetitive movements<sup>6</sup>. Data were dumped at 100Hz. In order to validate the accuracy of the proposed estimation we used some of the encoders as validation data. In particular, even if their value was measured during the experiment, the associated measurements were not used in the estimation but simply used a ground-truth. In Fig. 3 we report the results and the errors in estimating the elbow (two top rows) and the wrist (two bottom lines) joint coordinates without direct measurements. Compared with the ground-truth the achieved estimation gave a maximum error below  $0.1[rad] \cong 5.7[deg]$ .

## VIII. CONCLUSIONS

We have presented a framework for simultaneously estimating the dynamics and the state of an articulated rigid body. Dynamics are represented in full coordinates, i.e. for each rigid body we estimate internal and external forces/torques, linear/angular accelerations, joint torque and joint accelerations. The computed estimation is physically consistent since it takes into account the dynamic model of the articulated rigid body. The a-posteriori estimation includes all available measurements such as encoders, gyroscopes, accelerometers, force and torque sensors. Simulations and experiments have shown that the proposed approach can obtain reliable estimations.

## APPENDIX

In this section we derive the set of constraints that the vector  $\mathbf{d}$  defined by (1) and (2) should satisfy in order to be physically consistent. These constraints derive from the Newton-Euler equations of an articulated rigid body. Derivation can be found in [1] (section 5.3) the only difference being our choice of expressing  $\mathbf{f}_i^x$  in local coordinates:

$$\mathbf{a}_i = {}^i\mathbf{X}_{\lambda(i)}\mathbf{a}_{\lambda(i)} + \mathbf{S}_i\ddot{\mathbf{q}}_i + \mathbf{v}_i \times \mathbf{S}_i\dot{\mathbf{q}}_i, \quad (13a)$$

$$\mathbf{f}_i = \mathbf{I}_i\mathbf{a}_i - \mathbf{f}_i^x + \sum_{j \in \mu(i)} {}^i\mathbf{X}_j^*\mathbf{f}_j + \mathbf{v}_i \times {}^i\mathbf{I}_i\mathbf{v}_i \quad (13b)$$

<sup>5</sup>It has to be noticed that force/torque sensors are embedded in the links and can be used to measure internal wrenches, i.e. spatial forces exchanged between links (denoted  $\mathbf{f}_i$  in the previous sections).

<sup>6</sup><https://www.youtube.com/watch?v=q1Esrqze50>

which should hold for any choice of  $i$  in the set  $1, \dots, N_B$  with the definition  $\mathbf{a}_{\lambda(1)} = -\mathbf{a}_g$  (the gravitational spatial acceleration vector expressed in the inertial frame, null in its first three components and equal to the gravitational acceleration in the last three). Equation (13a) is the equation for propagating accelerations across a kinematic tree. Equation (13b) represents the Newton-Euler equations in spatial notation. The quantities  $\mathbf{v}_i$  which depend on  $\mathbf{q}$  and  $\dot{\mathbf{q}}$  only, are computed iterating the following equations from 1 to  $N_B$  with initial condition  $\mathbf{v}_{\lambda(1)} = \mathbf{0}$ .

$$\mathbf{v}_i = {}^i\mathbf{X}_{\lambda(i)}\mathbf{v}_{\lambda(i)} + \mathbf{S}_i\dot{\mathbf{q}}_i, \quad (14)$$

As a final constraint on  $\mathbf{d}$ , the spatial force  $\mathbf{f}_i$  projected on the joint motion constraints  $\mathbf{S}_i$  should equal the joint torque  $\tau_i$  for  $i = 1, \dots, N_B$ :

$$\tau_i = \mathbf{S}_i^\top \mathbf{f}_i, \quad (15)$$

Remarkably, the constraints (13) and (15) are all linear on the components of  $\mathbf{d}$  and can therefore be rearranged as follows:

$$\mathbf{D}(\mathbf{q})\mathbf{d} + \mathbf{b}_D(\mathbf{q}, \dot{\mathbf{q}}) = \mathbf{0}. \quad (16)$$

The matrix  $\mathbf{D}$  and the vector  $\mathbf{b}_D$  have the following structure:

$$\mathbf{D} = \begin{bmatrix} \mathbf{A}_{1,1} & \dots & \mathbf{A}_{1,N_B} \\ \vdots & \ddots & \vdots \\ \mathbf{A}_{N_B,1} & \dots & \mathbf{A}_{N_B,N_B} \end{bmatrix}, \quad \mathbf{b}_D = \begin{bmatrix} \mathbf{b}_1 \\ \vdots \\ \mathbf{b}_{N_B} \end{bmatrix}, \quad (17)$$

Let's also define the matrices:

$$\mathbf{A}_{i,i} = \begin{bmatrix} -\mathbf{1} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{S}_i^\top \\ \mathbf{I}_i & \mathbf{0} & -\mathbf{1} & -\mathbf{1} & \mathbf{0} \\ \mathbf{0} & -\mathbf{1} & \mathbf{S}_i^\top & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad (18)$$

$$\forall j \in \mu(i) \quad \mathbf{A}_{i,j} = \begin{bmatrix} \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & {}^i\mathbf{X}_j^* & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad (19)$$

$$i = 2 \dots N_B \quad \mathbf{A}_{i,\lambda(i)} = \begin{bmatrix} {}^i\mathbf{X}_{\lambda(i)} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \\ \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} & \mathbf{0} \end{bmatrix}, \quad (20)$$

$$\text{if } \lambda(i) = 0 \quad \mathbf{b}_i = \begin{bmatrix} {}^i\mathbf{X}_0\mathbf{a}_0 + \mathbf{v}_i \times \mathbf{S}_i\dot{\mathbf{q}}_i \\ \mathbf{v}_i \times {}^*\mathbf{I}_i\mathbf{v}_i \\ \mathbf{0} \end{bmatrix}, \quad (21)$$

$$\text{if } \lambda(i) \neq 0 \quad \mathbf{b}_i = \begin{bmatrix} \mathbf{v}_i \times \mathbf{S}_i\dot{\mathbf{q}}_i \\ \mathbf{v}_i \times {}^*\mathbf{I}_i\mathbf{v}_i \\ \mathbf{0} \end{bmatrix}. \quad (22)$$

## REFERENCES

- [1] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer, 2008.
- [2] B. J. Stephens, "State estimation for force-controlled humanoid balance using simple models in the presence of modeling error," in *Robotics and Automation (ICRA), 2011 IEEE International Conference on*, May 2011, pp. 3994–3999. [Online]. Available: <http://dx.doi.org/10.1109/ICRA.2011.5980358>
- [3] B. Xinjilefu and C. G. Atkeson, "State estimation of a walking humanoid robot," in *IROS*. IEEE, 2012, pp. 3693–3699. [Online]. Available: <http://dblp.uni-trier.de/db/conf/iros/iros2012.html#XinjilefuA12>
- [4] K. Lowrey, S. Kolev, Y. Tassa, T. Erez, and E. Todorov, "Physically-consistent sensor fusion in contact-rich behaviors," *manuscript under review*, 2014.
- [5] M. Bloesch, M. Hutter, M. Hoepfner, C. D. Remy, C. Gehring, and R. Siegwart, "State Estimation for Legged Robots - Consistent Fusion of Leg Kinematics and IMU," in *Robotics: Science and Systems VIII*, 2012. [Online]. Available: <http://roboticsproceedings.org/rss08/p03.pdf>
- [6] N. Rotella, M. Bloesch, L. Righetti, and S. Schaal, "State estimation for a humanoid robot," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2014, pp. 952–958.
- [7] B. Xinjilefu, F. Siyuan, and C. G. Atkeson, "dynamic state estimation using quadratic programming," in *2014 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2014, pp. 989–994.
- [8] X. Xinjilefu, S. Feng, W. Huang, and C. Atkeson, "Decoupled state estimation for humanoids using full-body dynamics," in *IEEE/RAS 2014 IEEE International Conference on Robotics and Automation (ICRA 2014)*, 2014.
- [9] M. Benallegue and F. Lamiraux, "Humanoid flexibility deformation can be efficiently estimated using only inertial measurement units and contact information," in *IEEE International Conference on Humanoid Robots (Humanoids)*, Madrid, Spain, 2014.
- [10] J. Vihonen, J. Honkakorpi, J. Mattila, and A. Visa, "Geometry-aided angular acceleration sensing of rigid multi-body manipulator using mems rate gyros and linear accelerometers," in *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2013.
- [11] M. J. Wainwright, M. J. Wainwright, M. I. Jordan, and M. I. Jordan, "Graphical Models, Exponential Families, and Variational Inference," *Foundations and Trends® in Machine Learning*, vol. 1, pp. 1–305, 2007. [Online]. Available: <http://www.nowpublishers.com/product.aspx?product=MAL&doi=2200000001>
- [12] O. Shental, P. Siegel, J. Wolf, D. Bickson, and D. Dolev, "Gaussian belief propagation solver for systems of linear equations," *2008 IEEE International Symposium on Information Theory*, 2008.
- [13] T. Wu, Y. Tassa, V. Kumar, J. Movellan, and E. Todorov, "Stac: Simultaneous tracking and calibration," in *IEEE/RAS International Conference on Humanoid Robots (Humanoids)*, 2013.
- [14] M. Fumagalli, S. Ivaldi, M. Randazzo, L. Natale, G. Metta, G. Sandini, and F. Nori, "Force feedback exploiting tactile and proximal force/torque sensing. Theory and implementation on the humanoid robot iCub," *Autonomous Robots*, vol. 33, no. 4, pp. 381–398, 2012.
- [15] D. Koller and N. Friedman, *Probabilistic Graphical Models: Principles and Techniques*. MIT Press, 2009.
- [16] F. Nori, "Berdy: Bayesian estimation for robot dynamics. a probabilistic estimation of whole-body dynamics with redundant measurements," *International Journal of Robotics Research (submitted)*, 2014.