Implementing Torque Control with High-Ratio Gear Boxes and without Joint-Torque Sensors

Andrea del Prete, Nicolas Mansard, Oscar Efrain Ramos Ponce, Olivier Stasse, Francesco Nori

Implementing Torque Control with High-Ratio Gear Boxes and without Joint-Torque Sensors

Andrea Del Prete*, Nicolas Mansard, Oscar E. Ramos,
Olivier Stasse, Francesco Nori†

September 17, 2015

Abstract

This paper presents a complete framework (estimation, identification and control) for the implementation of joint-torque control on the humanoid robot HRP-2. While torque control has already been implemented on a few humanoid robots, this is one of the first implementations of torque control on a robot that was originally built to be position controlled (iCub[1] and Asimo[2] being the first two, to the best of our knowledge). The challenge comes from both the hardware, which does not include joint-torque sensors and presents large friction due to the high-ratio gear boxes, and the software interface, which only accepts desired joint-angle commands (no motor current/voltage control). The contribution of the paper is to provide a complete methodology that is very likely to be reproduced as most robots are designed to provide only position control capabilities. Additionally, the method is validated by exhaustive experiments on one leg of the robot, including a comparison with the original position controller. We tested the torque controller in both motion control and cartesian force control. The torque control can track better a reference trajectory while using lower values for the feedback gains (up to 25%). Moreover, we verified the quality of the identified motor models by analyzing the contribution of the feedforward terms of our torque controller, which dominate the feedback terms.

1 Introduction

With respect to position control, torque control provides a number of well-known benefits, especially for humanoid robots. Torque control is a necessary requirement for the implementation of rigid-body inverse-dynamics

*Del Prete, Mansard, Ramos and Stasse are with LAAS, CNRS, Univ. Toulouse, Toulouse, France adelpret@laas.fr, nmansard@laas.fr, oramos@laas.fr, ostasse@laas.fr

†Nori is with IIT, Genova, Italy, francesco.nori@iit.it

control[3] (i.e. a feedback linearization technique). This class of control algorithms compensate for the dynamics of the system, that is they linearize the dynamics of the state (or a linear combination of the state) of the system. Once the system is linearized standard linear control techniques can be applied. The higher complexity of inverse-dynamics control with respect to position control is justified by the improved trajectory tracking capabilities, which are crucial for dynamic locomotion[4] and whole-body manipulation. Moreover a relatively recent trend of inverse-dynamics controllers[5, 6] exploit the efficiency of quadratic programming solvers to ensure the satisfaction of a number of limits affecting humanoid robots, such as torque bounds, force friction cones and center of pressure limits. This is not possible with standard position control.

Given the improved tracking performances, lower feedback gains can be used with inverse-dynamics controller, resulting in higher compliance of the system. Higher compliance brings two main advantages: automatic adaptation to uncertain environment (e.g. walking on uneven terrain[7]) and safer human-robot interaction.[8]

Finally, another benefit of inverse-dynamics control is the clean integration of motion and force control in a unified framework.[9, 10] While contact forces can also be regulated with position-based admittance control[11], this strategy lacks a clean integration with motion control: since reference joint positions are computed based on the force feedback they cannot accomodate a desired motion at the same time. Moreover admittance control is based on force feedback only, whereas in inverse dynamics the feedforward terms can contribute to improving the force tracking. Finally it is not clear how to deal with underactuation in admittance control, which makes its application to humanoid robots not straightforward.

The price to pay for all these benefits boils down to i) a more complex control algorithm, which needs a model of the dynamics of the system, and ii) the need of a measurement device to reconstruct the torque at every joint. The need for torque measurements comes from the large joint friction[12] (especially static friction) introduced by the high-ratio gear boxes used by most humanoid robots. The lack of torque measurements is what prevents the implementation of torque control on old-generation robots, such as our platform HRP-2.[13] This paper discusses a complete framework (identification, estimation and control) to implement torque control on robots despite the lack of joint-torque sensors. We rather use a combination of sensors that more classically equip humanoid robots nowadays: 6-axis force/torque (F/T) sensors, an inertial measurement unit (IMU) and joint encoders.

It should be clear that we do not advocate against the use of joint-torque sensors; on the contrary we think that measuring directly the joint torques is the best way to achieve good torque tracking. We instead estimate joint torques using the robot inverse-dynamics model, which adds also a little computational overhead (about 0.01 ms) with respect to torque-sensor based

controllers. However, in case torque sensing is not available (as it is the case for HRP-2, iCub, HRP-4, Asimo, ...), the presented method is probably the only way to implement torque control and inverse-dynamics control.

1.1 Paper overview

The key idea is to estimate the joint torques by using the procedure proposed for the iCub robot,[14] which we recall in Section 2. We propagate the wrenches measured by the F/T sensors along the kinematic chain using a model of the dynamics of the robot and an estimation of the velocities and accelerations of the robot bodies, reconstructed using the joint encoders and the IMU. This joint-torque estimation is used in the control as a feedback; it is also used off-line to identify the relationship between the motor input and the associated joint torque. We discuss the selected actuator model and its identification in Section 3. In brief, the selected model neglects the elasticity of the harmonic drive and the electric pole of the motor transfer function, which results in an instantaneous relationship between motor input and joint torque. While the model has experimentally proved to achieve a reasonable accuracy, it remarkably simplifies the identification procedure: in particular, the identification does not require to excite the robot at high frequencies. In Section 4 we then discuss a torque-control law that is the superposition of a feedforward term (given by the identified motor model) and a feedback term (based on the estimated joint torque). Finally we validated the whole framework by implementing an inverse-dynamics controller on one leg of HRP-2, which has 6 joints. The results presented in Section 5 show that, in comparison to the closed-source position control of HRP-2, we get a better position tracking while using lower feedback gains. We also show the performances of our framework on a force-tracking task, which is easily integrated in the inverse-dynamics controller.

1.2 Contributions

The paper presents a complete methodology to effectively implement torque control on a stiff robot without joint-torque sensors, along with an exhaustive experimental analysis of the implementation of the actuator identification, the state estimation, and the control on one leg of HRP-2. The major contribution is the proof of concept (HRP-2 was effectively transformed into a torque-controlled robot) and the report of the experimental results, which we tried to make as exhaustive as possible. While the torque-estimation procedure is taken from previous work on iCub[14], the present paper introduces several technical contributions: i) the simple actuator model used in the identification procedure (neglecting gear-box elasticity and including the low-level PD position controller), ii) the asymmetric-penalty fitting optimization, iii) the piecewise-linear fitting algorithm, iv) a fair comparison

with the standard position control, v) an analysis of the contribution of the different feedforward/feedback components of the controller for the two cases of motion tracking and force tracking.

1.3 Related Works

Despite being an essential component for the implementation of rigid-body inverse-dynamics control, the problem of regulating the joint torques is still subject of ongoing research. The difference between the various works mainly lies in the type of actuator (rigid vs elastic, electric vs hydraulic) and the chosen actuator model (i.e. whether it includes the gear-box elasticity and/or the electric motor pole).

The first industrial torque controller has been designed by DLR for their light-weight robot, by modeling the actuator flexibility and adding joint-torque sensing capabilities at the lowest level.[15] Both the measured joint torque and its derivative are used as feedback, which requires an excellent and expensive hardware, while the identification of the joint flexibility requires a difficult experimental process. A similar approach can be applied to hydraulic actuation, also exploiting measurements of joint torques,[16] by compensating for the natural velocity feedback between the load and the actuator. Other works have focused on improving the performances of these controllers by identifying/observing and compensating for friction [17, 18].

Even if the flexibility was modeled in the previously cited works, the considered robots were very stiff. However, the approach also applies to series-elastic actuators.[19] The control action is given by the superposition of a feedforward term (given by the inverse actuator model) and a feedback PID term (given from the torque error measured through an integrated torque sensor). Moreover, a disturbance observer was used to improve the torque-tracking capabilities of the control system.[20] Extensions have been proposed to automatically tune the gains of the controller by means of an LQR control.[21]

When the identification of the system model is not accurate, a classical solution is to rely on time-delay estimation.[22] Also in this work, the authors model the elasticity introduced by the gear-box; moreover, they compensate for viscous and Coulomb friction and propose a heuristic to compensate for static friction. More complex observers, like disturbance observers[23] have also been explored to compensate during the motion for the imperfection of the controller model.

A comparison of several torque-control schemes has been presented,[24] which focuses on passivity, an important property when exploiting force measurements in a control loop.

All the above-mentioned works rely on a direct measure of the joint torques. Khatib *et al.*[2] first proposed to only rely on feedforward. Their work mostly focused on the identification of the actuator transfer function

using high-order polynomials. The identification is then particularly delicate to implement, due to the sensitivity of the model to observations and numerical errors. The humanoid robot Atlas (built from the prototype Petman[25]) is likely using a similar strategy, with a good actuation model, but without a direct measure of joint torques.

With respect to the related works our approach is characterized by two facts: i) our platform is not equipped with joint-torque sensors and ii) our motor model neglects the gear-box elasticity. Not relying on torque sensors for the torque feedback makes this framework applicable to a large number of robots that are only equipped with 6-axis F/T sensors. Moreover, the chosen actuation model results in a simpler identification procedure, yet reasonably accurate and experimentally leading to good control performances.

2 Torque Estimation

Before delving into the identification and the control, we need to know how to estimate the joint torques by using the 6-axis F/T sensors mounted at the wrists and ankles of the robot. Consider the equations of motion of an n-joint floating-base robot:

$$M(q)\dot{v} + h(q, v) - J(q)^\top f = S^\top \tau, \tag{1}$$

where $q^\top = \begin{bmatrix} x_b^\top & q_j^\top \end{bmatrix} \in \mathbb{R}^{n+6}$ and $v^\top = \begin{bmatrix} v_b^\top & \dot{q}_j^\top \end{bmatrix} \in \mathbb{R}^{n+6}$ contain respectively the position and the velocity of the base and the joints of the robot, $M(q) \in \mathbb{R}^{(n+6)\times(n+6)}$ is the mass matrix, $h(q, v) \in \mathbb{R}^{n+6}$ contains the Coriolis, centrifugal and gravity generalized forces, $J(q) \in \mathbb{R}^{k\times(n+6)}$ is the contact Jacobian, $f \in \mathbb{R}^k$ are the contact forces, $\tau \in \mathbb{R}^n$ are the joint torques, and $S \in \mathbb{R}^{n\times(n+6)}$ is the joint-selection matrix. The joint torques can be seen a function of $q, v, \dot{v}$ and f, so we can translate the problem of estimating τ into the problem of estimating all these quantities. In the next subsections we discuss how to estimate $q, v, \dot{v}$ and f from the available sensors in HRP-2: one encoder at each joint, one IMU located in the torso, and four F/T sensors located at both wrists and ankles.

2.1 Estimating Positions, Velocities, and Accelerations of the Joints

Since our robot is only equipped with encoders, we can only directly measure q_j. However, from the position measurements we can estimate $\dot{q}_j$ and $\ddot{q}_j$ by numerical differentiation. We used a Savitzky-Golay filter,[26] which is a type-I finite impulse response (FIR) low-pass filter. These filters are based on the idea of fitting a low-order polynomial to a fixed-length moving window. An important feature is that they provide a smooth version of the signal together with an estimate of its derivatives. Moreover, contrary

to the Kalman filter, Savitzky-Golay filters are model free, so they are not biased by a model (e.g. constant acceleration) that could become extremely inaccurate in case of physical interaction with the user — which is one of the use cases of our method. Since the sampling time is constant in our setup, the matrix pseudo-inverse appearing in the fitting procedure can be precomputed, so that only a matrix-vector multiplication is needed to get the estimation, making the filter computationally cheap.

2.2 Estimating the End-Effector External Wrenches

In HRP-2, the F/T sensors are attached to the end-effectors of the robot. They then measure two dynamic effects: the end-effector body dynamics (weight and inertia) and the applied external wrench. The wrench is estimated by compensating the weight and inertia of the end-effector body. This needs the angular velocity and both linear and angular accelerations of this body, which are estimated using the methodology described in the next paragraph. Additionally, the estimation requires the inertial parameters of the end-effector body, which can be simply identified off-line using directly the sensor measurements at several static configurations.

When the F/T sensors are located inside the kinematic chain (like on iCub, where they are positioned between shoulder and elbow in the arms, and between hip and knee in the legs), the dynamics of the lower part of the chain must be compensated. The idea is just the same as when the sensor is attached to the end-effector: estimate the velocity and acceleration of the lower bodies, then compute the wrench corresponding to these movements and subtract it from the F/T measurements to get the external wrench.

2.3 Estimating the Joint Torques from the Floating-Base Pose, Velocity, and Acceleration

From the fundamental laws of mechanics, joint torques τ depend on neither the pose of the floating base nor its linear velocity (the robot base — i.e. the arbitrary first body of the kinematic tree — is a Galilean referential regardless of its pose and linear velocity). In fact, joint torques are only affected by the linear/angular accelerations (including gravity acceleration) and the angular velocity of the floating base. Both the linear acceleration (combined with gravity) and the angular velocity of the base are given by the IMU.[1] The angular acceleration can instead be either computed by numerical differentiation of the angular velocity or neglected (since the gyroscope measurements

[1] Actually, the waist represents the base in the model of HRP-2, whereas the IMU is located in the torso. However, we can either move the base in the model (using some dynamic library) or compute the acceleration/velocity of the waist from the acceleration/velocity of the torso and the pose/velocity/acceleration of the interconnecting joint.

are typically noisy, in practice their numerical derivative could be too noisy to be used).

The joint torques are then recursively computed from the robot-base body to its end-effectors by going along the kinematic tree, following the Recursive Newton-Euler Algorithm (RNEA).[27] Pose and velocity of each body of the robot is computed with respect to the base. Accelerations are propagated along the chain from the combination of linear acceleration and gravity measured by the IMU accelerometer. At the end of the forward pass, the algorithm returns the velocity of every body and a combination of acceleration and gravity effects. This quantity is used to cancel out the inertial effect of the end-effector bodies measured by their respective F/T sensor, as explained in the previous section. The algorithm then computes the joint torques by recursively propagating the forces backward along the kinematic tree.

Since the IMU measurements are subject to high-frequency noise, we filter them with low-pass filters. As for the estimation of joint velocities and accelerations, we used Savitzky-Golay filters.[26] Algorithm 1 summarizes the torque estimation procedure.

Algorithm 1 Torque Estimation

Read encoders, accelerometer, gyroscope, ftSens
$\{q_j, \dot{q}_j, \ddot{q}_j\} \leftarrow$ SAVITZKY_GOLAY_2(encoders)
$\{v_{imu}, \dot{v}_{imu}\} \leftarrow$ SAVITZKY_GOLAY_1(gyroscope, accelerometer)
$\{v_{torso}, \dot{v}_{torso}\} \leftarrow$ IMUTOTORSOKIN$(v_{imu}, \dot{v}_{imu})$
$\{v_b, \dot{v}_b\} \leftarrow$ TORSOTOBASEKIN$(v_{torso}, \dot{v}_{torso}, q_j, \dot{q}_j, \ddot{q}_j)$
ftSens $\leftarrow$ SAVITZKY_GOLAY_0(ftSens)
ftSens $\leftarrow$ COMPENSATEFORINERTIALWRENCHMEASUREDBYFTSENS$(q_j, \dot{q}_j, \ddot{q}_j, v_b, \dot{v}_b$,ftSens)
$\tau \leftarrow$ RNEA$(q_j, \dot{q}_j, \ddot{q}_j, v_b, \dot{v}_b$, ftSens)
return τ

2.4 Estimating Additional External Wrenches

An additional external wrench exerted above the F/T sensors on the kinematic chain (e.g. on the torso) can also be estimated. Consider again the equations of motion of the robot:

$$M\dot{\nu} + h - J_m^\top f_m = \underbrace{\begin{bmatrix} J_u^\top & S^\top \end{bmatrix}}_{A} \begin{bmatrix} f_u \\ \tau \end{bmatrix},$$

where f_m are the measured wrenches (with J_m being the associated Jacobian) and f_u are the unknown wrenches (with J_u being the associated Jacobian). As long as A is full-column rank, there is enough information

to estimate τ using a least-squares procedure. In practice this means that we can only estimate one additional external wrench, namely $\text{rank}(J_u) \leq 6$. Altogether five external wrenches can be applied on the robot: one for each end-effector plus one on the kinematic tree delimitated by the four F/T sensors. In most classical humanoid scenarios such as walking, push recovery and manipulation this assumption is verified. Complex scenarios with multiple contacts on the robot's body (e.g. both knees on the ground, both elbows in contact) could not be properly handled by our method. Adding more F/T sensors inside the robot's limbs (similarly to the iCub robot) would increase the number of contact scenarios handled by the method.[14]

The critical point is to know on which link the additional external wrench is applied, because this affects on which joints that wrench is creating torque. This problem can be solved by using tactile sensors[28], but at the moment HRP-2 does not have any. Alternatively, momenta/residual methods could be used [29, 8], but it is not clear whether they could work without knowing the motor current. We decided to always assume to have an external wrench on the torso, so that the robot can properly estimate the joint torques in case it is pushed on the torso. In case of no external wrench, the estimated wrench should be close to zero. Alternatively, we can assume no external wrench at all, in which case it is possible to use the $n+6$ equations of motion to estimate the n torque variables as:

$$\hat{\tau} = \underset{\tau}{\text{argmin}} \, ||M\hat{\dot{v}} + \hat{h} - J_m^\top \hat{f}_m - S^\top \tau||^2 = S(M\hat{\dot{v}} + \hat{h} - J_m^\top \hat{f}_m) \qquad (2)$$

where $\hat{.}$ represents the estimated values. This is equivalent to assuming that the unknown wrench is applied to the floating base (i.e. J_u is a matrix that selects only the base-link coordinates).

In any case, the torque estimation can be computed using a modified RNEA:[14] the measured wrenches have to be propagated from the four end-effectors back to the unknown-wrench link. This will automatically result in the estimation of the joint torques, without any need to directly estimate the unknown wrench. However, it is useful to monitor the unknown wrench to check that, in case of no external contact, it remains small in magnitude. The unknown wrench f_u can be computed as the wrench that makes the net wrench on the associated link be equal to the time derivative of its momentum:

$$\sum_i f_i + f_u = I\dot{v}_l + \dot{I}v_l,$$

where I represents the spatial inertia of the link, v_l is its spatial velocity and f_i are the wrenches applied on the link by all the connected links.

3 Motor Identification

The previous section briefly recalled how joint torques are estimated from the classical sensors of a humanoid robot without direct joint-torque measurements; this estimation will be used to identify a model of the actuation of the robot. This section starts by describing and discussing the selected actuation model, which is kept simple to allow a robust identification procedure. Then, the identification is formulated as an optimization problem, which is solved by numerical methods. We finally briefly discuss practical issues related to the data collection to ensure a good identification.

3.1 Linear Model

3.1.1 Generic rigid model

Considering DC motors as actuators and assuming a rigid transmission (i.e. no elasticity in the gear box), the joint torques τ are given by the difference between the motor torques τ_m and the friction torques τ_F:

$$\tau = \underbrace{K_1 i}_{\tau_m} - \underbrace{(K_2 \dot{q}_j + K_3 \text{sign}(\dot{q}_j))}_{\tau_F}, \tag{3}$$

where $K_1 \in \mathbb{R}^{n \times n}$ contains the motor drive gains, $i \in \mathbb{R}^n$ are the motor currents, $K_2 \in \mathbb{R}^{n \times n}$ contains the viscous-friction coefficients, and $K_3 \in \mathbb{R}^{n \times n}$ contains the Coulomb-friction coefficients. The same model also applies if we can control the motor voltage V, rather than the current i. Neglecting the electrical dynamics, which is a reasonable assumption provided that the dynamics of the current amplifiers have much higher bandwidth than the motors, we have:

$$i = \frac{1}{R} V + \underbrace{\frac{k_b}{R} \dot{q}_j}_{i_b}, \tag{4}$$

where i_b is proportional to the back electromotive torque, R is the motor coil resistance, and k_b is a constant comprising the effect of the flux generated by the coil. If we substitute (4) in (3), the equation maintains the same form because the back electromotive torque can be included inside the viscous-friction torque, being both terms proportional to $\dot{q}_j$. For this reason, in the following we do not make any distinction between voltage and current control.

Overall our motor model is based on two simplifying assumptions: i) the electrical dynamics is negligible, and ii) the transmission is perfectly rigid. While the first assumption is common in the literature [15, 19], the second one constitutes one of the specific features of this work. Our experiments show that even using this simplified model we get significant improvements over classic position control.

3.1.2 Model for HRP-2

HRP-2 is a high-quality industry-built robot which comes with the drawback that the low-level control implementation is closed source and thus prevents the direct command of the current/voltage of the motors. However, we can specify the desired joint angles $q_j^d \in \mathbb{R}^n$, which are then used by the low-level position controller to compute the desired motor currents using a simple proportional-derivative (PD) control law (with marginal modifications, as we will see later):

$$i = K_4 \underbrace{(q_j^d - q_j)}_{\Delta_q} - K_5 \dot{q}_j. \tag{5}$$

Substituting (5) in (3) and solving for Δ_q, we get the following relationship between the position tracking error Δ_q and the joint torques τ:

$$\Delta_q = K_4^{-1} \left(K_1^{-1} \tau + (K_1^{-1} K_2 + K_5) \dot{q}_j + K_1^{-1} K_3 \, \text{sign}(\dot{q}_j) \right),$$

which we can rewrite as:

$$\Delta_q = K_\tau \tau + K_v \dot{q}_j + K_c \text{sign}(\dot{q}_j). \tag{6}$$

This implies that having the motor current as control input is actually equivalent to having the desired joint angles as control input: in both cases there is a linear relationship between the τ, $\dot{q}_j$, $\text{sign}(\dot{q}_j)$ and the control input.

3.1.3 Linear identification

Having a linear model makes the identification straightforward because we can use a least-squares fitting. Setting $x^\top = \begin{bmatrix} K_\tau & K_v & K_c \end{bmatrix}$ as the decision variable, and assuming m collected samples, we can solve the following optimization problem for each motor:

$$\min_{x \in \mathbb{R}^3} ||Ax - b||^2 \tag{7}$$

where

$$A = \begin{bmatrix} \tau_1 & \dot{q}_{j_1} & \text{sign}(\dot{q}_{j_1}) \\ \vdots & \vdots & \vdots \\ \tau_m & \dot{q}_{j_m} & \text{sign}(\dot{q}_{j_m}) \end{bmatrix} \quad \text{and} \quad b = \begin{bmatrix} \Delta_{q_1} \\ \vdots \\ \Delta_{q_m} \end{bmatrix}.$$

The optimal estimate is obtained by solving this quadratic problem (e.g. computing the pseudo-inverse of A, whose implementation is available in most mathematic programming frameworks).

During the experiments on the robot, we observed that the Coulomb-friction term was not significantly improving the fitting. We therefore neglected it in practice. For the sake of clarity, we removed this term from the model in the remaining of the paper.

3.2 Adapting the Model to the Data

As soon as we started collecting data on the robot we realized that the model (6) did not fit on HRP-2. For instance, Fig. 1 depicts the relationship between Δ_q and τ. Clearly, the relationship is not linear as we could have expected. One may wonder whether this nonlinearity is real or due to other estimation errors, in particular the discretization error of the encoders, and the error in the inertial parameters of the system that we use to estimate the joint torques. However, even if Δ_q is small (in the order of 10^{-3} rad), the discretization error of the encoder is much smaller (in the order of 10^{-5} rad), therefore we do not think that the nonlinearity could be due to this effect. Regarding the torque estimation, given that the data have been collected keeping the joints fixed, the joint torques depend only on gravity (which is basically constant) and external wrenches. We can then infer that any error in the inertial parameters of the robot would create a simple offset in the torque estimation (i.e. in the gravity torque), which would not explain the nonlinearity.

We then proposed an augmented empirical model and a procedure to identify its parameters.

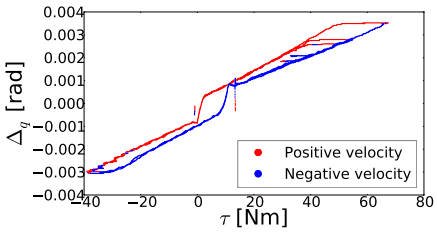
3.2.1 Piecewise model

The mismatch observed between model and data is partially due to a *dead zone* in the current control (implemented at the closed-source low level of the robot), which means that errors with magnitude smaller than a certain threshold are considered as zero. Moreover, Fig. 1 shows that τ is not a function of Δ_q only. We experimentally discovered that τ seems to depend also on the sign of the joint velocity: this is likely due to the fact that the low-level position controller is not just a PD as we initially hypothesized. Contrary to the Coulomb friction, this results in a change of the model (different affine parameters) triggered by the change of sign. By splitting the data into two groups depending on $\text{sign}(\dot{q})$ (see Fig. 1a), the relationship is well described by a piecewise-linear model:

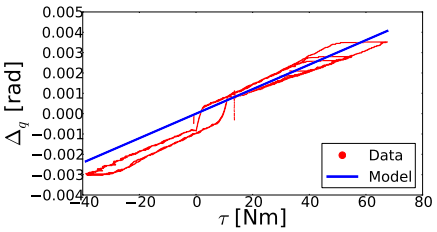
$$\Delta_q = f(\tau, \text{sign}(\dot{q})) = \begin{cases} f_p(\tau), & \text{if } \text{sign}(\dot{q}) > 0 \\ f_n(\tau), & \text{if } \text{sign}(\dot{q}) < 0 \end{cases}$$

$$f_p(\tau) = \begin{cases} k_{\tau 1p}\tau + q_{1p}, & \text{if } \tau \leq \tau_{1p} \\ k_{\tau 2p}\tau + q_{2p}, & \text{if } \tau_{1p} \leq \tau \leq \tau_{2p} \\ k_{\tau 3p}\tau + q_{3p}, & \text{if } \tau \geq \tau_{2p} \end{cases} \quad f_n(\tau) = \begin{cases} k_{\tau 1n}\tau + q_{1n}, & \text{if } \tau \leq \tau_{1n} \\ k_{\tau 2n}\tau + q_{2n}, & \text{if } \tau_{1n} \leq \tau \leq \tau_{2n} \\ k_{\tau 3n}\tau + q_{3n}, & \text{if } \tau \geq \tau_{2n} \end{cases}$$

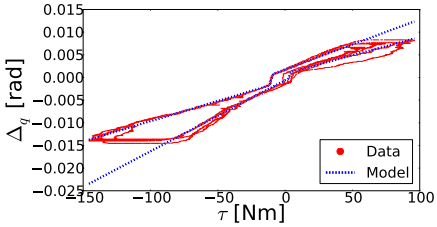
Moving from a linear to a piecewise-linear model seems to be a small change, but it increases the complexity making the identification problem nonlinear, and more importantly, nonconvex.[30] However, given a good-enough initial guess, the algorithm proposed in [30] is often able to find a good fitting for



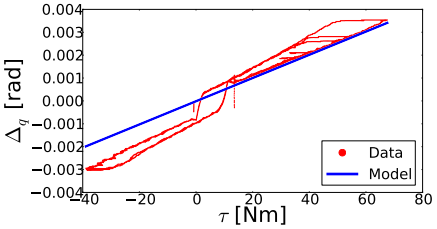
(a) Data separated depending on the sign of the joint velocity.



(b) The linear model identified with a symmetric penalty function (i.e. least-squares).



(c) The two piecewise-linear models (for positive and negative velocities).



(d) The linear model identified with an asymmetric penalty function.

Figure 1: Torque τ vs. position error Δ_q for the right knee (a, b, d) and right hip-pitch (c) joints of HRP-2. These data have been collected on one position-controlled joint keeping q_j^d constant and applying external forces. The joint is nearly static, so that $\dot{q}_j \approx 0$. From (6), we expect $\Delta_q = K_\tau \tau$, but the data clearly do not fit.

the data. This algorithm only works for convex piecewise-linear functions, whereas our function is nonconvex.[2] For this reason, we slightly modified the algorithm so that it could work for our special nonconvex case.

3.2.2 Piecewise-Linear Fitting

The algorithm is rather simple. We start with an initial guess on the two switching points τ_1, τ_2 between the three linear models (we will hereafter drop the p and n sub-indices for notation simplicity; therefore, τ_1 represents either τ_{1p} or τ_{1n}, and so forth). Then, we repeat the following procedure until convergence or until we reach the maximum number of iterations:

- update $k_{\tau 1}, k_{\tau 2}, k_{\tau 3}, q_1, q_2, q_3$ by separately fitting the three linear models to the three datasets
- update τ_1 to the intersection point between line 1 and line 2
- update τ_2 to the intersection point between line 2 and line 3

The algorithm converges when τ_1 and τ_2 do not change between two successive iterations. Remarkably, the resulting model is continuous by construction. We empirically noticed that the algorithm works better if we bound the slope of the central line (k_2), which is reasonable because we know, from observing the data, that it always has a large positive slope.

For identifying $f(.)$ on the leg joints, we set the initial guess to $\tau_1 = -0.5$ and $\tau_2 = 0.5$, and we set the maximum number of iterations to 10. Fig. 1c shows the result for the right hip-pitch joint. Most of the time, the algorithm converged before reaching the maximum number of iterations. Even if not perfect, the results seem satisfactory for the scope of the application.

3.2.3 Limits of the Model

Despite the good fitting of the model to the data, this approach proved unsuccessful for a number of reasons, so we decided to pursue another strategy (see next subsection). The main reason that led us to this decision was that the resulting torque control was unstable in open-loop (i.e. with zero feedback gains). We believe that this was mainly due to the fact that the two models are not exactly separated by the sign of the joint velocity. In particular, the switch between the two models (i.e. when the velocity changes sign) is critical and it is hard to identify. Having access to the code of the low-level control of HRP-2 would drastically simplify the understanding of this effect and of how we could compensate for it, but for now this is not the case. Since our piecewise-linear model did not perfectly fit the data, in

[2] A piecewise-linear function is convex if and only if it can be expressed as the maximum over a set of linear function (see [31], Example 3.5). This is not the case for our model.

some regions it was over-estimating the value of Δ_q necessary to generate a certain τ, triggering instability.

As secondary reasons, the extended model made both the identification and the control more complex. The piecewise-linear fitting algorithm does not always converge to a good solution, so it requires the user to provide an initial guess that may be different for each joint. The controller with the extended model also requires an estimation of the sign of the acceleration, which needs to be approximated with a smooth function to avoid discontinuities in the control input. This introduces an additional parameter to tune the smoothing of the sign function.

3.3 Asymmetric-Penalty Identification

Given the failure of the approach described in the previous subsection, we decided to go back to the original linear model. To account for the fact that the model could not exactly fit the data, we modified the identification problem. Rather than trying to find the best fit in the least-squares sense, we now look for the best fit that almost never over-estimates Δ_q in absolute value. This conservative approach guarantees that the torque controller will be stable in open-loop. In other words, we penalize more over-estimating than under-estimating. The result of this asymmetric penalty function can be seen in Fig. 1d: the blue line is almost never higher than the red dots in absolute value.

To formulate the new identification problem, the data $\{A, b\}$ are separated into two sets $\{A^{pos}, b^{pos}\}$ and $\{A^{neg}, b^{neg}\}$ depending on the sign of Δ_q, where each set has m^{pos} and m^{neg} pair elements, respectively. The optimization problem is then formulated as:

$$\min_{x \in \mathbb{R}^3} \sum_{i=1}^{m^{pos}} \Phi(a_i^{pos} x - b_i^{pos}) + \sum_{i=1}^{m^{neg}} \Phi(-a_i^{neg} x + b_i^{neg}) \tag{8}$$

with

$$\Phi(z) = \begin{cases} wz^2, & \text{if } z > 0 \\ z^2, & \text{otherwise} \end{cases}$$

where a_i^{pos}/a_i^{neg} is the i-th row of A^{pos}/A^{neg}, b_i^{pos}/b_i^{neg} is the i-th element of b^{pos}/b^{neg}, and $w > 1$ is a user-defined parameter that weighs how much over-estimation is penalized more with respect to under-estimation.

This problem is convex, so we can solve it with, for instance, Newton's method. Alternatively, we can reformulate it as a Quadratic Program (QP). To this end, we introduce two auxiliary variables y^p, y^n and we reformulate

(8) as:

$$\begin{aligned} \min_{x\in\mathbb{R}^3, y^p\in\mathbb{R}^m, y^n\in\mathbb{R}^m} \quad & w||y^p||^2 + ||y^n||^2 \\ \text{s. t.} \quad & y^p \geq \bar{A}x - \bar{b} \\ & y^n \geq -\bar{A}x + \bar{b} \\ & y^p, y^n \geq 0, \end{aligned} \tag{9}$$

where:

$$\bar{A} = \begin{bmatrix} A^{pos} \\ -A^{neg} \end{bmatrix} \qquad \bar{b} = \begin{bmatrix} b^{pos} \\ -b^{neg} \end{bmatrix}$$

At the optimum, the elements of y^p corresponding to under-estimation (i.e. $\bar{A}x < \bar{b}$) are zero, while those corresponding to over-estimation are exactly equal to $\bar{A}x - \bar{b}$ (which is then penalized with the weight w in the cost function). Similarly, at the optimum the elements of y^n corresponding to over-estimation are zero, while those corresponding to under-estimation are equal to $-\bar{A}x+\bar{b}$ (which is penalized without any weight in the cost function). In other words, the optimum values of y^p and y^n are complementary, that is ${y^p}^\top y^n = 0$. The advantage of this new formulation is that we can solve it with any QP solver. The disadvantage is that we have introduced $2m$ new variables and $2m$ inequality constraints, which could make the problem intractable if m — the number of data samples — is too large (e.g. both *CVX*[32] and *qpOases*[33] failed for $m > 10^4$). In practice we saw that 10^3 samples (obtained by downsampling the original data taken at 1 kHz) are enough to identify the model, which can then be validated on a larger number of samples.

3.3.1 Two-Stage Identification

We noticed that, especially for some joints, it was beneficial to split the identification problem into two parts. First, only use the zero-velocity (i.e. below some threshold) samples to identify k_τ; then, only use the nonzero-velocity samples, together with the previously identified value of k_τ, to identify k_v. In some cases, this procedure improves the identification of k_τ. The reason is that k_v is always 1 or 2 orders of magnitude greater than k_τ, so when the joint velocity is not zero most of Δ_q is due to friction, making the identification of k_τ poorly conditioned.

3.4 Data Collection

The collection of the data used for identification is crucial to the final result. The fact that the linear model cannot fit the data well (e.g. because of the deadzone) implies that the identification will not easily generalize outside of the observed range. It is then important to make sure that the collected data properly cover the whole operative region of τ and $\dot{q}$. A direct consequence is that we cannot identify the model using only low or

medium velocities/torques and expect the model to work well for large velocities/torques. To get both large velocities and large torques, we split the data collection into two parts. The first part samples large torques and zero velocities, so that we can identify k_τ in (6). In the second part we use large velocities, so that we can identify k_v.

3.4.1 High-torque data collection

In this phase we position the robot in a predefined set of configurations using the standard high-gain position control of HRP-2. The user can then apply torque on all the joints by exerting forces on the robot end-effectors. The aim is to create as large as possible torques. For the identification of the leg joints we can lift the robot and apply forces on its feet. Different configurations are necessary to generate large torques on all the joints.

3.4.2 High-velocity data collection

The identification of frictions can be automatized because we do not need the user to apply external forces. We commanded to the position controller an increasing-amplitude sinusoid:

$$q_j^d(t) = (a_0 + at)\sin(2\pi f t)$$

where a_0 is the initial amplitude, a regulates the speed of increase of the sinusoid amplitude, and f is the sinusoid frequency.

3.4.3 Noncausal Estimation

Since the identification is based on the estimation of the joint velocities and accelerations, we can expect to get better results by using noncausal estimations. This means that at time t we do not estimate $\dot{q}_j(t)$, but we rather estimate $\dot{q}_j(t-t_d)$, where t_d is the estimation delay. The same applies for $q, \dot{v}$ and the force/torque measurements.

4 Control

The control is made up of two terms that we describe in the first subsection: a feedforward term to compensate for the dynamics of the actuator, and a feedback term to reject noise and unmodeled dynamics. We then present an inverse dynamics scheme, which we used in the experiments to generate the reference joint torques. In the second subsection, we reformulate the control law in order to ease its experimental comparison with the position controller.

4.1 Control law

4.1.1 Feedforward

The feedforward comes from the identification of the model equation (6), carried out in the previous section. The feedforward control law is:

$$\Delta_q = K_\tau \tau^* + K_v \dot{q}_j, \tag{10}$$

where τ^* are the commanded joint torques.

4.1.2 Torque feedback

The feedback consists in a proportional-integral control law:

$$\tau^* = \tau^d + K_p e_\tau + K_i \int e_\tau dt \tag{11}$$

where $e_\tau = (\tau^d - \hat{\tau}) \in \mathbb{R}^n$ is the torque tracking error, $K_p, K_i \in \mathbb{R}^{n \times n}$ are respectively the proportional and integral gain matrices (diagonal positive-definite), $\tau^d \in \mathbb{R}^n$ are the desired joint torques and $\hat{\tau} \in \mathbb{R}^n$ are the estimated joint torques (2).

In practice, we did not observe any improvement of the control behavior when modifying the integral term. Since the position control does not comprehend an integral term, we decided not to include it and consider $K_i = 0$ in the remaining of the paper. We could possibly add a disturbance observer, which has been already used for joint-torque control.[19]

4.1.3 Inverse dynamics

In the experiments, the desired joint torques are computed from an inverse-dynamics control law[34] tracking a reference trajectory of the actuated degrees of freedom.[3] Given a desired joint trajectory $\{q_j(t)^d, \dot{q}_j(t)^d, \ddot{q}_j(t)^d\}$ and a desired contact force f^d, the desired joint torques are:

$$\tau^d = M_j \ddot{q}_j^d + \hat{h}_j - J_j^\top f^* + K_s e_q + K_d \dot{e}_q \tag{12}$$

with

$$f^* = f^d + K_f e_f$$

where $M_j \in \mathbb{R}^{n \times n}$ is the joint-space mass matrix (i.e. the bottom-right corner of M), $\hat{h}_j \in \mathbb{R}^n$ are the last n elements of h, J_j is the part of the Jacobian corresponding to the actuated joints, $e_q = q_j^d - \hat{q}_j$ is the position

[3] When discussing estimation in Section 2 we considered a floating-base model of the robot. However, for the sake of simplicity, the controller rather considers a fixed-base model. This is justified by the fact that, being the robot attached to a lift device, the motion of its base was limited. In the future we plan to switch to a floating-base inverse-dynamics controller such as the one presented in [6].

tracking error, $e_f = f^d - \hat{f}$ is the force tracking error, and $K_s \in \mathbb{R}^{n\times n}$, $K_d \in \mathbb{R}^{n\times n}$, $K_f \in \mathbb{R}^{k\times k}$ are the stiffness, damping and force gain matrices (diagonal positive-definite), respectively.

4.2 Gain Comparison

The control law can be parametrized by the user-defined gains (K_p, K_s, K_d, K_f) and by the gains identified from the motor characteristics (K_τ, K_v). Substituting (11) in (10), using (12) and the estimates $\hat{\dot{q}}_j = \dot{q}_j$ and $\hat{\tau} = M_j\hat{\ddot{q}}_j + \hat{h}_j - J_j^\top \hat{f}$, the control law can be finally separated in two main parts, corresponding to the feedforward and feedback components, as:

$$\Delta_q = K_\tau(M_j\ddot{q}_j^d + \hat{h}_j - J_j^\top f^d) + K_v\hat{\dot{q}}_j \tag{13}$$

$$+ K_6 e_q + K_7\dot{e}_q + K_8\ddot{e}_q + K_9 e_f \tag{14}$$

where:

$$\begin{aligned} K_6 &= K_\tau(I + K_p)K_s \\ K_7 &= K_\tau(I + K_p)K_d \\ K_8 &= K_\tau K_p M_j \\ K_9 &= -K_\tau(K_p J_j^\top(I + K_f) + J_j^\top K_f). \end{aligned}$$

The terms in (13) correspond to the feedforward action. More precisely, the first term decouples the articulated dynamics, while the second term compensates for the friction. The terms in (14) correspond to the feedback action on the position, velocity, acceleration and force. The feedforward part (13) is independent of the position and force tracking errors.

Since K_τ is given by the motor identification (it is not defined by the user), selecting one of the two sets of gains $\{K_6, K_7, K_8, K_9\}$ or $\{K_p, K_s, K_d, K_f\}$ implies an immediate correspondence in the other set.

There is a direct correspondence with the position-based PD controller originally implemented on the robot. The errors in position e_q appears in both controllers, while the other errors $\dot{e}_q$, $\ddot{e}_q$ and e_f as well as the feedforward are only considered in our proposed torque control. It is now possible to compare the tuning of both controllers. When the diagonal components of K_6 are greater than one, it implies that the torque controller is using feedback gains larger than those used by the native low-level position controller of HRP-2. Conversely, diagonal components of K_6 lower than one imply a smaller position feedback for the torque controller than for the position controller.

(a) Force tracking experiment.

(b) Motion tracking experiment.

Figure 2: Experimental setup.

5 Experimental Results

5.1 Experimental Setup

We carried out all the experiments on the 6 joints of the right leg of HRP-2. In the motion-control experiment HRP-2 was hanging in the air, attached to a lift device by means of two strings tied to its shoulders (see Fig. 2b). In the force-control experiment it was instead standing on both feet (see Fig. 2a), while the strings prevented the robot from falling. As far as the estimation is concerned, we used a window of 80 samples. Since the sampling time was 1 ms, this resulted in an estimation delay of 40 ms (i.e. half the window size). We tested the controller also with smaller/larger window sizes, but 80 seemed to be the best trade-off between stability and performance. We used first-order polynomials to filter the F/T sensor and the IMU measurements, and second-order polynomials for the encoders. We set the torque-feedback gains K_p to 2 for all the 6 joints.

The tuning of the estimation and control parameters (i.e. window length and feedback gains) has been empirically performed. To keep things simple, we decided to use the same gain for all the joints and the same delay (and cut-off frequency) for all the sensors. However, we believe that there is room for improvement in this direction. We saw that by increasing the estimation window we got smoother feedback signals, which allowed us to use higher gains, so improving the performance of the controller. On the other side, increasing the estimation window also increases the estimation delay, which

Table 1: Identified motor parameters.

Joint	$10^3 k_\tau$	$10^3 k_v$
Hip yaw	0.212	13.0
Hip roll	0.030	6.332
Hip pitch	0.12	7.0
Knee	0.051	6.561
Ankle pitch	0.177	7.698
Ankle roll	0.240	6.0

at a certain point outweighs the improvement coming from using higher gains. Finding the best trade-off between estimation smoothing/delay and feedback gains is still an open problem, which might be an interesting subject for future work.

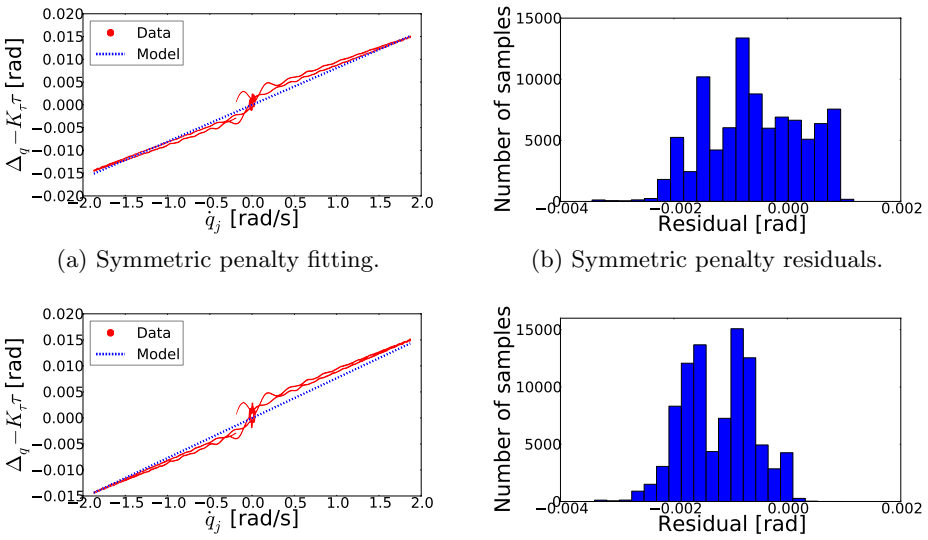
5.2 Motor Identification

We collected the data following the procedure described in Section 3.4 and we then used them to identify the motor parameters, as described in Section 3.3. We set the weight of the asymmetric penalty function (8) to $w = 100$. Table 1 lists the identified parameters. Fig. 3 depicts how the model fits the data for the ankle-pitch joint. Using the symmetric penalty function, the model tends to over-estimate the data for large velocities. In practice this resulted in an unstable controller (i.e. the joint accelerating as soon as a certain velocity was reached, even when lowering the feedback gains to zero). With the asymmetric penalty, the model no longer over-estimates the data, while the quality of the fit for small velocity remains of similar quality.

5.3 Motion Control

Table 2: Average squared tracking error (in rad 10^3) for the stairs-climbing trajectory: $N^{-1}\sqrt{\sum_{i=1}^{N}(q^d(t_i) - q(t_i))^2}$, where N is the number of samples.

Joint	Pos. ctrl.	Torque ctrl $K_6 = 1$	Torque ctrl $K_6 = 0.5$	Torque ctrl $K_6 = 0.25$	Torque ctrl $K_6 = 0.1$
Hip roll	0.020	0.005	0.008	0.016	0.040
Hip pitch	0.056	0.018	0.035	0.064	0.139
Knee	0.115	0.060	0.061	0.067	0.094
Ankle pitch	0.064	0.017	0.034	0.063	0.148
Ankle roll	0.029	0.015	0.029	0.059	0.136



(a) Symmetric penalty fitting.

(b) Symmetric penalty residuals.

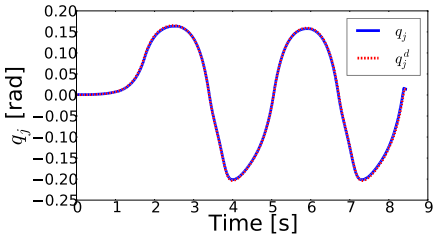
(c) Asymmetric penalty fitting.

(d) Asymmetric penalty residuals.

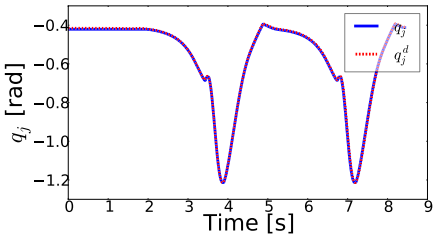
Figure 3: Friction identification for the ankle-pitch joint: comparison of the two penalty functions: [top] symmetric penalty (7) [bottom] asymmetric penalty (8). The two plots on the left display the friction part of the model ($\Delta_q - K_\tau \tau$) with respect to the velocity ($\dot{q}_j$). The data have been collected tracking a sinusoidal reference with increasing amplitude. The two plots on the right display the distribution of the residuals ($|K_\tau \tau + K_v \dot{q}_j| - |\Delta_q|$): the symmetric penalty tends to overestimate the friction for high velocities (indeed we have positive residuals), while the asymmetric penalty mostly underestimates while keeping a fit of similar quality.

Table 3: Maximum tracking error (in rad 10^3) for the stairs-climbing trajectory.

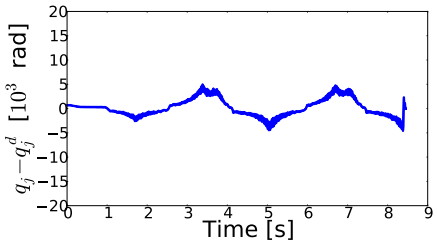
Joint	Pos. ctrl.	Torque ctrl $K_6 = 1$	Torque ctrl $K_6 = 0.5$	Torque ctrl $K_6 = 0.25$	Torque ctrl $K_6 = 0.1$
Hip roll	4.79	2.35	2.95	2.85	5.76
Hip pitch	17.01	7.87	7.83	10.40	22.78
Knee	61.86	39.64	41.55	41.19	49.33
Ankle pitch	19.56	4.84	5.92	10.39	22.43
Ankle roll	6.11	2.55	3.69	6.80	16.44



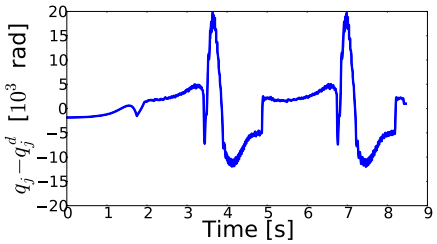
(a) Hip-roll position tracking



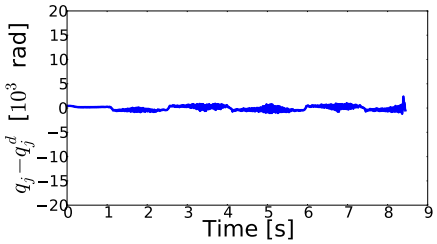
(b) Ankle position tracking



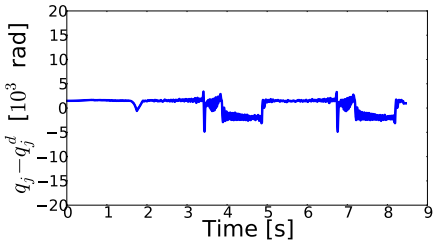
(c) Hip-roll error, position control



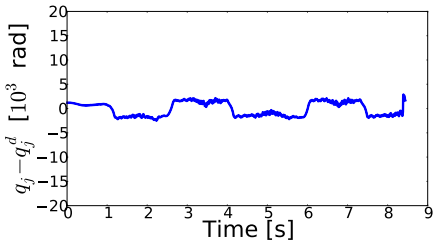
(d) Ankle error, position control



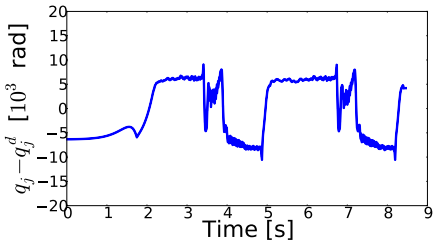
(e) Hip-roll error, torque control $K_6 = 1$



(f) Ankle error, torque control $K_6 = 1$



(g) Hip-roll error, torque control $K_6 = 0.25$



(h) Ankle error, torque control $K_6 = 0.25$

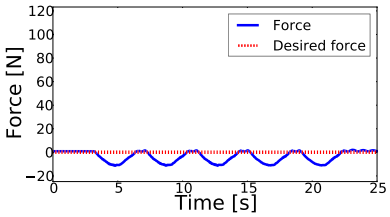
Figure 4: Comparison of the motion tracking accuracy for the position controller and for the torque controller. The results from two joints are displayed (hip roll and ankle pitch). Three different controllers are exhibited: original low-level position controller, inverse dynamics with the same gains than the position controller ($K_6 = 1$) and inverse dynamics with 25% feedback gains ($K_6 = 0.25$). At large scale, the position trajectories are similar for the three controllers (the top row shows the results for the position controller, but the other two controllers would look just the same). Compared to the position controller, the torque controller with similar gain ($K_6 = 1$) is much more accurate (lower tracking error). The torque controller with lower gain ($K_6 = 0.25$) has a comparable accuracy.

We compared the capabilities of tracking a joint-space trajectory of our new controller with those of the standard low-level position controller of HRP-2. The trajectory is the swinging motion of the right leg computed to make the robot climb some stairs. This motion is very demanding with respect to the capabilities of the motors, by asking a large displacement (of the swing leg) in a short time. We experimentally know that this movement is close to the dynamic limits of the robot.

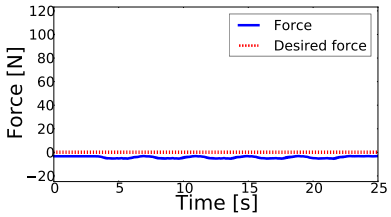
We tested the position controller against four different gain tuning of inverse-dynamics controller, corresponding to 100%, 50%, 25% and 10% of the standard position-control gains (i.e. K_6 was equal to $1, 0.5, 0.25$ and 0.1, respectively). Table 2 and 3 respectively report the average and maximum tracking error for the each joint and each controller. Fig. 4 shows the tracking error trajectory for two joints (hip roll and ankle pitch) and three controllers, to visualize the different shapes of the errors. With $K_6 = 1$ the inverse-dynamics controller performs significantly better than the position controller, obtaining lower maximum and average tracking errors on all joints. With $K_6 = 0.5$ the inverse-dynamics controller still performs better than the position controller on all joints (except the average error of the ankle roll, which is equivalent). With $K_6 = 0.25$ the average errors on most joints are almost equivalent (except ankle roll/knee, which are significantly worse/better), but the inverse-dynamics controller still results in lower maximum errors (except for the ankle roll). Finally, with $K_6 = 0.1$ the position controller performs better than the inverse-dynamics controller on all joints but the knee.

5.4 Force Control

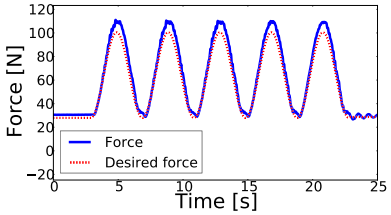
This experiment tests the capabilities of the torque controller to track a reference Cartesian force. The right foot of HRP-2 is positioned in contact with a rigid fixed object (a small pile of bricks, see Fig. 2). We then commanded to the inverse-dynamics controller a sinusoidal force reference on the z axis (vertical direction), while maintaining the force on the other axes to zero. The position gains were set to 10% of the standard position-control gains (i.e. $K_6 = 0.1$), which corresponds to the lowest gain with motion-tracking accuracy similar to the position controller, obtained from the previous experiment. The reference joint angles were set to be compatible with the force task. By keeping significant position-feedback gains, some slight motion occurring during the experiment would likely cause interferences between the joint-position and the force tracking. However, it does not seem relevant to set these gains to zero, mainly for security reasons (in case the robot lost the contact the position feedback would have prevented it from moving too far away from the initial configuration). We empirically set all the force feedback gains K_f to 1, which is the higher values before observing instability in the control. Fig. 5 shows the results. The robot is



(a) Force tracking on the x axis.



(b) Force tracking on the y axis.



(c) Force tracking on the z axis.

Figure 5: Tracking of a sinusoidal force reference on the z axis of the right foot while keeping other force directions to zero. An overshoot of maximum 10% is observed at the sinusoid apex. The same 10% are observed as a coupling on the X axis. No significant delay affects the tracking.

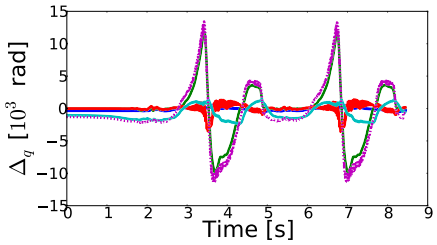
able to track the desired force sinusoid; the $\approx 10\%$ overshoot is likely due to the above-mentioned conflict between force and position tracking.

5.5 Analysis of Different Control Components

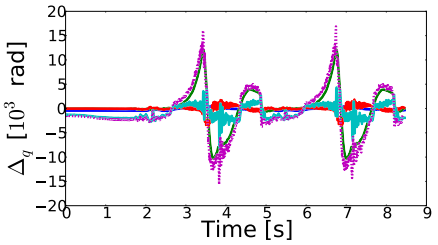
The final torque control is a complex mix of feedforward and feedback terms. This section analyses the different components of the control input Δ_q in the two previous experiments (i.e. motion control and force control) to provide some intuitions regarding the effect of each component on the quality of the presented results. Our goal is to understand to what extent each component is contributing to the final control input. Recall from Section 4.2 that we can decompose the control input in six different parts, two feedforward components and four feedback components:

$$\Delta_q = \underbrace{K_\tau(M_j\ddot{q}_j^d + \hat{h} - J^\top f^d)}_{\text{feedforward torque}} + \underbrace{K_v\hat{\dot{q}}_j}_{\text{feedforward friction}} + \underbrace{K_6 e_q}_{\text{feedback position}} + \underbrace{K_7\dot{e}_q}_{\text{feedback velocity}} + \underbrace{K_8\ddot{e}_q}_{\text{feedback acceleration}} + \underbrace{K_9 e_f}_{\text{feedback force}}$$

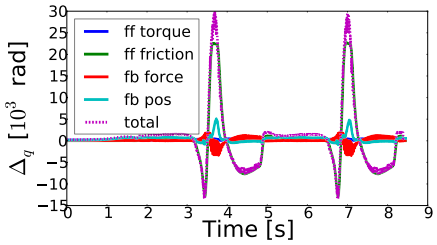
We reproduced the two previous experiments and measured the importance of each of these six terms in the control. Fig. 6 shows the values of the two



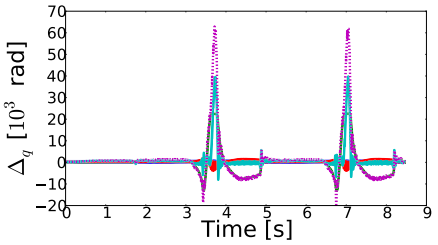
(a) Hip-pitch joint, inverse-dynamics control with $K_6 = 0.1$.



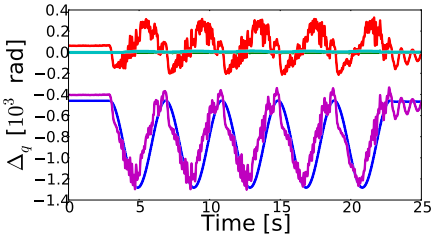
(b) Hip-pitch joint, inverse-dynamics control with $K_6 = 1$



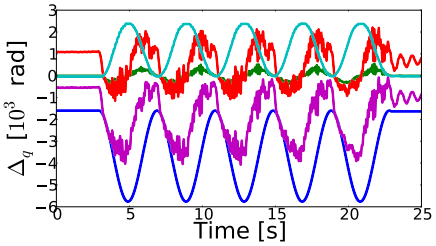
(c) Knee joint, inverse-dynamics control with $K_6 = 0.1$.



(d) Knee joint, inverse-dynamics control with $K_6 = 1$.



(e) Knee joint, force control.



(f) Ankle-pitch joint, force control.

Figure 6: Different components of the control input Δ_q in the "motion control" (a, b, c and d) and "force control" experiments (e and f).

feedforward components and the position and force feedback components for two different joints during the motion-control experiment (with both the maximum and the minimum feedback gains) and force-control experiment. By looking at these data we can draw that:

- during motion control the "feedforward friction" is contributing the most, followed by the "feedback position";
- during force control the "feedforward torque" is contributing the most, followed by the "feedback force" (and the "feedback position" for the ankle-pitch joint).

This means that in motion control the proposed torque controller is approximately acting as a standard position controller plus a friction compensation. On the contrary, in force control the "feedforward torque" component is playing an important role in the control, which justifies the existence of that part in the control law. Most probably it is the term $J^\top f^d$ the one contributing the most between the three terms composing the feedforward torque, given that inertial and bias torques are negligible in this scenario.

From Fig. 6f it can also be noticed that the "feedback position" significantly contributed to the force control for certain joints. This is due to the fact that the contact was not perfectly rigid, so some joints slightly moved during the experiment affecting (likely negatively) the force tracking.

6 Conclusions and Future Work

This paper discussed the implementation of torque control on the humanoid robot HRP-2. Contrary to most previous works on the subject — but like 90% of nowadays robots — our platform does not have joint-torque sensors and its software interface does not allow the user to directly control the motor currents/voltages. This makes the control of the joint torques more challenging on this hardware than on robots that were built to be torque controlled. We list hereafter the main contributions of this work.

- Our experimental setup shows that it is possible to implement torque control on a robot that was originally built to be position controlled, and we can use it to control its motion and its contact forces.
- We described the implementation on the physical robot of the complete framework, composed by estimation, identification and control.
- Torque control, together with inverse dynamics, has been experimentally proved to produce better motion tracking than position control (i.e. improved accuracy for similar gains, or reduced gains while preserving accuracy). While this was well-known for torque-sensor based torque control, it is a nontrivial result for the torque control proposed in this paper.

- Our analysis shows that in motion control the most important part of our control framework is the feedforward friction compensation, whereas in force control the most important part is the torque feedforward component. The dominance of the feedforward terms in the control action validates our choice to use a simplified actuator model and our identification procedure.

- We proposed an asymmetric-penalty identification that tends to preserve the stability of the controller (e.g. by avoiding over-compensating friction), along with two numerical formulations (either as a convex optimization problem, or as a QP).

While the presented results are very promising, there is still room for improvement, especially in the identification and the estimation. Using an approximated model for the relationship between the joint torques and the position is interesting because it simplifies the identification and avoids instability. However, the simplified model is arbitrary (we selected the relevant terms from our subjective observations), and some unidentified terms are now missing to obtain a perfect fit. Another important point is that the identified model is currently only used by the controller. The state estimation algorithm could also exploit it to predict the future state and so nullify estimation delays. We also plan to experiment with Disturbance Observers, as they have already been used to improve the performances of other torque-control architectures.[19]

To conclude, this work opens a new interesting direction for the control of "position-controlled" robots (i.e. stiff robots without joint-torque sensors), which could benefit from the presented torque-control architecture to improve their performances, both in motion-tracking and in force-tracking tasks.

Acknowledgements

This work was supported by Euroc (FP7 Grant Agreement 608849), Entracte (ANR Grant Agreement 13-CORD-002-01) and Koroibot (ICT-2013-10 project number 611909).

References

[1] Francesco Nori, Silvio Traversaro, Jorhabib Eljaik, Francesco Romano, Andrea Del Prete, and Daniele Pucci. iCub Whole-Body Control through Force Regulation on Rigid Non-Coplanar Contacts. *Frontiers in Robotics and AI*, 2, 2015.

[2] Oussama Khatib, Peter Thaulad, and Jaeheung Park. Torque-position transformer for task control of position controlled robots. *2008 IEEE*

International Conference on Robotics and Automation, pages 1729–1734, May 2008.

[3] Jun Nakanishi, Michael Mistry, and Stefan Schaal. Inverse dynamics control with floating base and constraints. In *Robotics and Automation, 2007 IEEE International Conference on*, number April, pages 10–14, 2007.

[4] Mrinal Kalakrishnan, Jonas Buchli, Peter Pastor, Michael Mistry, and Stefan Schaal. Learning, planning, and control for quadruped locomotion over challenging terrain. *The International Journal of Robotics Research*, 30(2):236–258, November 2010.

[5] Layale Saab, Oscar E. Ramos, Nicolas Mansard, Philippe Soueres, and Jean-yves Fourquet. Dynamic Whole-Body Motion Generation under Rigid Contacts and other Unilateral Constraints. *IEEE Transactions on Robotics*, 29(2):346–362, 2013.

[6] Andrea Del Prete and Nicolas Mansard. Addressing Constraint Robustness to Torque Errors in Task-Space Inverse Dynamics. In *Robotics, Science and Systems (RSS)*, Rome, 2015.

[7] Oscar E. Ramos, Nicolas Mansard, Olivier Stasse, Jean-bernard Hayet, and Philippe Soueres. Towards reactive vision-guided walking on rough terrain: an inverse-dynamics based approach. *International Journal of Humanoid Robotics*, 2, 2014.

[8] Sami Haddadin, Alin Albu-Schäffer, Alessandro De Luca, and Gerd Hirzinger. Collision detection and reaction: A contribution to safe physical Human-Robot Interaction. *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3356–3363, September 2008.

[9] Andrea Del Prete, Francesco Nori, Giorgio Metta, and Lorenzo Natale. Prioritized Motion-Force Control of Constrained Fully-Actuated Robots: "Task Space Inverse Dynamics". *Robotics and Autonomous Systems*, 63:150–157, 2015.

[10] Oussama Khatib. A unified approach for motion and force control of robot manipulators: The operational space formulation. *IEEE Journal on Robotics and Automation*, 3(1):43–53, February 1987.

[11] R. Volpe and P. Khosla. A theoretical and experimental investigation of explicit force control strategies for manipulators. *IEEE Transactions on Automatic Control*, 38(11):1634–1650, 1993.

[12] Brian Stewart Randall Armstrong. *Dynamics for Robot Control: Friction Modeling and Ensuring Excitation During Parameter Identification*. PhD thesis, Stanford University, 1988.

[13] Kenji Kaneko and Fumio Kanehiro. Design of prototype humanoid robotics platform for HRP. In *Intelligent Robots and Systems, 2002. IEEE/RSJ International Conference on.*, 2002.

[14] Matteo Fumagalli, Serena Ivaldi, Marco Randazzo, Lorenzo Natale, Giorgio Metta, Giulio Sandini, and Francesco Nori. Force feedback exploiting tactile and proximal force/torque sensing. *Autonomous Robots*, 33(4):381–398, April 2012.

[15] Christian Ott, Alin Albu-Schäffer, and Gerd Hirzinger. Comparison of adaptive and nonadaptive tracking control laws for a flexible joint manipulator. In *Intelligent Robots and Systems (IROS), 2002 IEEE/RSJ International Conference on*, 2002.

[16] Thiago Boaventura, Michele Focchi, Marco Frigerio, Jonas Buchli, Claudio Semini, Gustavo A. Medrano-Cerda, and Darwin G. Caldwell. On the role of load motion compensation in high-performance force control. In *Intelligent Robots and Systems, IEEE/RSJ International Conference on*, pages 4066–4071. Ieee, October 2012.

[17] Mehrdad R. Kermani, Rajnikant V. Patel, and Mehrdad Moallem. Friction identification and compensation in robotic manipulators. *IEEE Transactions on Instrumentation and Measurement*, 56(6):2346–2353, 2007.

[18] Luc Le Tien and Alin Albu-Schäffer. Friction Observer and Compensation for Control of Robots with Joint Torque Measurement. In *2008 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 22–26, 2008.

[19] Nicholas Paine, Sehoon Oh, and Luis Sentis. Design and Control Considerations for High-Performance Series Elastic Actuators. *IEEE/ASME Transactions on Mechatronics*, 19(3):1080–1091, 2014.

[20] Kyoungchul Kong, Joonbum Bae, and Masayoshi Tomizuka. A Compact Rotary Series Elastic Actuator for Human Assistive Systems. *IEEE/ASME Transactions on Mechatronics*, 17(2):288–297, 2012.

[21] Kyoungchul Kong and Joonbum Bae. Control of Rotary Series Elastic Actuator for Ideal Force-Mode Actuation in HumanRobot Interaction Applications. *IEEE/ASME Transactions on Mechatronics*, 14(1):105–118, 2009.

[22] Sung-moon Hur, Sang-rok Oh, and Yonghwan Oh. Joint Space Torque Controller Based on Time-Delay Control with Collision Detection. In *Intelligent Robots and Systems (IROS), IEEE International Conference on*, pages 4710–4715, 2014.

[23] Min Jun Kim and Wan Kyun Chung. Robust Control of Flexible Joint Robots Based On Motor-side Dynamics Reshaping using Disturbance Observer (DOB). In *Intelligent Robots and Systems (IROS 2014), 2014 IEEE/RSJ International Conference on*, pages 2381–2388, 2014.

[24] Heike Vallery, Ralf Ekkelenkamp, Herman van der Kooij, and Martin Buss. Passive and accurate torque control of series elastic actuators. In *Intelligent Robots and Systems (IROS), IEEE International Conference on*, 2007.

[25] Gabe Nelson, Aaron Saunders, Neil Neville, Ben Swilling, Joe Bondaryk, Devin Billings, Chris Lee, Robert Playter, and Marc Raibert. PETMAN: A Humanoid Robot for Testing Chemical Protective Clothing. *Journal of the Robotics Society of Japan*, 30(4):372–377, 2012.

[26] Ronald W. Schafer. What is a Savitzky-Golay filter? *Signal Processing Magazine, IEEE*, (July):111–117, 2011.

[27] Roy Featherstone. *Rigid body dynamics algorithms*, volume 49. Springer Berlin:, 2008.

[28] Andrea Del Prete, Simone Denei, Lorenzo Natale, Fulvio Mastrogiovanni, Francesco Nori, Giorgio Cannata, and Giorgio Metta. Skin Spatial Calibration Using Force/Torque Measurements. In *Intelligent Robots and Systems (IROS), 2011 IEEE/RSJ International Conference on*, 2011.

[29] Alessandro De Luca and Raffaella Mattone. Sensorless robot collision detection and hybrid force/motion control. *Proceedings - IEEE International Conference on Robotics and Automation*, 2005(April):999–1004, 2005.

[30] Alessandro Magnani and Stephen Boyd. Convex piecewise-linear fitting. *Optimization and Engineering*, 10(1):1–17, March 2008.

[31] Stephen Boyd and Lieven Vandenberghe. *Convex Optimization*, volume 98. 2004.

[32] Michael Grant and Stephen Boyd. CVX: Matlab software for disciplined convex programming, version 2.1. http://cvxr.com/cvx, 2014.

[33] H. J. Ferreau, C. Kirches, and A. Potschka. qpOASES: A parametric active-set algorithm for quadratic programming. *Mathematical Programming Computation*, 2013.

[34] Richard M. Murray, Zexiang Li, and S Shankar Sastry. *A Mathematical Introduction to Robotic Manipulation*, volume 29. 1994.