Robot Dynamics Constraint for Inverse Kinematics

Enrico Mingo Hoffman, Alessio Rocchi, Nikos G. Tsagarakis, and Darwin G. Caldwell

Abstract Inverse Kinematics is a fundamental tool of Cartesian/Operational Space control. Recent approaches make use of Quadratic Programming Optimization to obtain desired joint velocities or accelerations from Cartesian references. QP based IK also permits to specify constraints to affect the solution. Constraints are fundamental and necessary when working with real robotic hardware since they prevent possible damages: joint limits, self collision avoidance and joint velocity limits are examples of such constraints. In this work we present a constraint to take into account joint torque limits based on the robot dynamics and force/torque sensor measurements. Despite the robot dynamics can be naturally expressed at acceleration level, our main goal is to specify this constraint in a *resolved motion rate control* IK. For this reason we formulate it also at the velocity level to be used in any IK QP based scheme. Hence, this formulation allows to generate dynamically feasible motions of the robot even in simple IK velocity based schemes. We apply this constraint to our humanoid robot COMAN while performing a Cartesian task which requires high torques in some joints. The constraint is developed inside the OpenSoT library.

Key words: Inverse Kinematics, Quadratic Programming Optimization, Dynamics.

1 Introduction

Inverse Kinematics (IK) is a fundamental step in robots control since it maps high level Cartesian commands into joint space commands. This step is in general highly non-linear, for this reason linearization through the robot Jacobian has been proposed and it is commonly used (named *Differential* IK):

Enrico Mingo Hoffman, Alessio Rocchi, Nikos G. Tsagarakis and Darwin G. Caldwell
DEPARTMENT OF ADVANCED ROBOTICS, ISTITUTO ITALIANO DI TECNOLOGIA, GENOA, ITALY e-mail: {enrico.mingo,alessio.rocchi,nikos.tsagarakis,darwin.caldwell}@iit.it

$$\dot{\mathbf{x}} = \mathbf{J}(\mathbf{q})\dot{\mathbf{q}} \tag{1}$$

where $\dot{\mathbf{q}}$ are joint space velocities, the Jacobian $\mathbf{J}$ (we will skip the dependency on the actual configuration $\mathbf{q}$ from now on) is expressed from a certain *base link* to a certain *distal link*, and operational space velocities $\dot{\mathbf{x}}$ of the *distal link* are expressed in the *base link* reference frame. A general and commonly used solution, for the redundant case of (1), is based on the Jacobian *pseudo-inverse* $\mathbf{J}^{\dagger}$:

$$\dot{\mathbf{q}}_{\mathrm{d}} = \mathbf{J}^{\dagger}\dot{\mathbf{x}}_{\mathrm{d}} + \left(\mathbf{I} - \mathbf{J}^{\dagger}\mathbf{J}\right)\dot{\mathbf{q}}_0 \tag{2}$$

where $\dot{\mathbf{q}}_0$ is an arbitrary joint space velocity.

Recent approaches make use of Quadratic Programming (QP) Optimization that makes also possible to specify linear constraints for the IK to affect the solution:

$$\begin{aligned} \dot{\mathbf{q}}_d = \underset{\dot{\mathbf{q}}}{\operatorname{argmin}} \; & \|\mathbf{J}_n\dot{\mathbf{q}} - \dot{\mathbf{x}}_{\mathrm{n,d}}\| + \lambda \|\dot{\mathbf{q}}\| \\ s.t. \quad & \mathbf{A}_1\dot{\mathbf{q}} = \mathbf{A}_1\dot{\mathbf{q}}_1 \\ & \vdots \\ & \mathbf{A}_{n-1}\dot{\mathbf{q}} = \mathbf{A}_{n-1}\dot{\mathbf{q}}_{n-1} \\ & \mathbf{A}_{c,1}\dot{\mathbf{q}} \leq \mathbf{b}_{c,1} \\ & \vdots \\ & \mathbf{A}_{c,n}\dot{\mathbf{q}} \leq \mathbf{b}_{c,n} \end{aligned} \tag{3}$$

where $\mathbf{A}$ matrices and $\mathbf{b}$ vectors are constraints. In (3), priorities are taken into account considering the previous solutions $\dot{\mathbf{q}}_i$, $i < n$ and constraints of the type $\mathbf{A}_i\dot{\mathbf{q}} = \mathbf{A}_i\dot{\mathbf{q}}_i$, $\forall i < n$, so that the optimality of all higher priority tasks is not changed by the current solution [5]. The second term in the cost function of (3) permits to handle kinematics singularities in order to avoid high joint velocities [6]. A similar structure can be used to solve the IK problem at the acceleration level [9, 11].

Many tasks and constraints have been presented in literature for the framework depicted in (3), examples are: joint limits, joint velocity limits, self collision avoidance [4], Cartesian velocity limits, minimum joint acceleration [2], *Capture Point* [8] and *Momentum Rate* control [3] for humanoid robot balancing.

In this work, a fundamental constraint for the IK step is presented: the robot dynamics. The computed velocities/accelerations in (3) may generate unfeasible motions, in terms of high joint torques, causing the damage of the robot. For this reason it is important to constrain the generated joint torques during the task execution. Despite this constraint is commonly used in *resolved acceleration control* IK schemes, it is not taken into account in simpler *resolved rate control* ones. The main goal of this work is to present the torque limit constraint expressed both at the acceleration and velocity level so that it can be applied to any IK scheme. Furthermore we use this constraint in a stack implementing a high level task in the simulation of a complex humanoid robot. This work follows the basic idea, presented in [2], to have a velocity control scheme that shares (approximately) the characteristics of an acceleration based scheme.

2 OpenSoT

OpenSoT is a framework developed at the *Istituto Italiano di Tecnologia* and aimed to control robots in Operational space [10]. OpenSoT implements the idea of decoupling atomic tasks/constraints descriptions and solvers to execute multiple tasks and achieve complex motion behaviors.

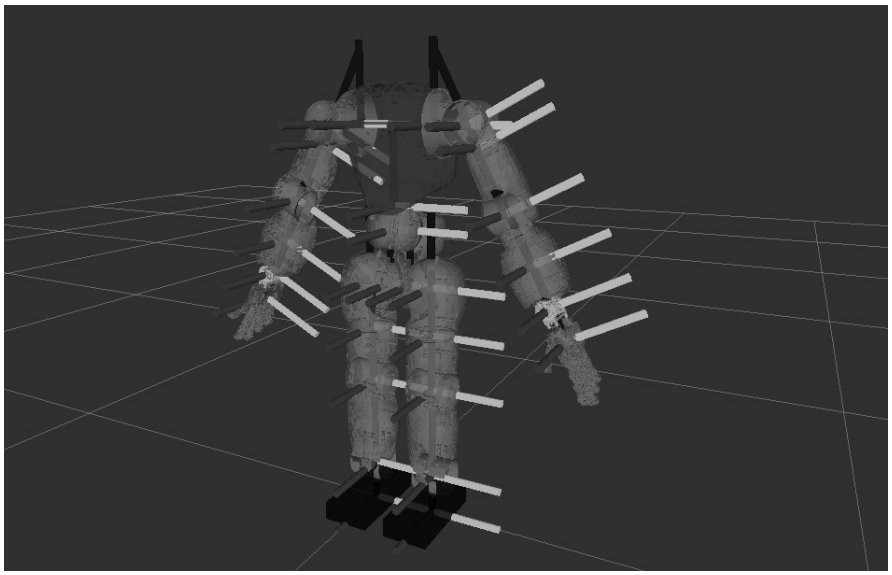


Fig. 1: COMAN robot kinematics and reference frames

It employs a solver, based on the formulation in (3), implementing a cascade of QP problems, and a set of tasks and constraints in velocity space in order to solve a generic hierarchical inverse kinematics problem on a floating or fixed base robot. The IK solver consists of a state machine that hides all the complexity of the underneath QP solver based on a state-of-art library in QP resolution using the *active set* approach: *qpOASES* [1]. This yields the following features that make the implementation of OpenSoT unique and attractive:

- Demonstrates high modularity through the separation of task descriptions, control schemes and solvers maximizing customization, flexibility and expandability.
- Provides user friendly interfaces for defining tasks, constraints and solvers to promote integration and cooperation in the emerging field of whole-body hierarchical control schemes.
- Demonstrates computation efficiency to allow for real time performance implementations.

- Allows ease of use and application with arbitrary robots through the Universal and Semantic Robotic Description Formats (URDF and SRDF).

The architecture of OpenSoT encourages collaboration and helps integration and code maintenance [1]. With all this in mind, we developed a *library* of tasks and constraints and the robot dynamics constraint is part of the latter.

3 Robot Dynamics Constraint

One of the fundamental problem in IK is that some assigned Cartesian reference trajectories might be dynamically unfeasible by the robot. This means that the robot might get damaged since the required joint torques for a certain motion could be too high. Various technique have been presented in the past to avoid this problem, one of the most famous is the *Dynamic Filter* [12]. This technique basically uses an Inverse Dynamics step to filter the generated joint accelerations from the IK solution.

In this work we formulate the *Dynamic Filter* as a constraint. The dynamics of the robot can be written as:

$$\mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} + \mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q}) = \tau - \mathbf{J}_c^T\mathbf{f}_c \tag{4}$$

where $\mathbf{M}(\mathbf{q})$ is the joint space inertia matrix, $\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})$ takes into account centrifugal and Coriolis terms, $\mathbf{G}(\mathbf{q})$ are the gravity torques, τ are the joint torques and $\mathbf{J}_c^T\mathbf{f}_c$ are the torques due to contacts (that we measure from the force/torque sensors).

Considering an acceleration level control and taking into account that each joint can provide $[\underline{\tau},\ \overline{\tau}]$, it is possible to write the constraint as:

$$\mathbf{D}(\mathbf{q},\dot{\mathbf{q}}) + \underline{\tau} \leq \mathbf{M}(\mathbf{q})\ddot{\mathbf{q}} \leq \mathbf{D}(\mathbf{q},\dot{\mathbf{q}}) + \overline{\tau} \tag{5}$$

with $\mathbf{D}(\mathbf{q},\dot{\mathbf{q}},\mathbf{f}_c) = -\left(\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q})\right) - \mathbf{J}_c^T\mathbf{f}_c$.

Despite the constraint is naturally described at the acceleration level, in this work we are considering velocity level control, so it is possible to approximate the joint acceleration $\ddot{\mathbf{q}}$ as:

$$\ddot{\mathbf{q}} \simeq \frac{\dot{\mathbf{q}}^* - \dot{\mathbf{q}}}{\Delta T} \tag{6}$$

then the constraint can be rewritten at the velocity level as:

$$\Delta T\left(\mathbf{D}(\mathbf{q},\dot{\mathbf{q}}) + \underline{\tau}\right) + \mathbf{M}(\mathbf{q})\dot{\mathbf{q}} \leq \mathbf{M}(\mathbf{q})\dot{\mathbf{q}}^* \leq \Delta T\left(\mathbf{D}(\mathbf{q},\dot{\mathbf{q}}) + \overline{\tau}\right) + \mathbf{M}(\mathbf{q})\dot{\mathbf{q}} \tag{7}$$

where $\dot{\mathbf{q}}^*$ are the new joint velocities references.

A similar idea was presented also in [7] but contact forces were not taken in consideration, while they are fundamental when working with floating base robots. Practically speaking, it is useful to have a scaling factor $\sigma \in (0,\ 1]$ in front of

[1] The OpenSoT library is open-source and downloadable at
`https://github.com/robotology-playground/OpenSoT`

the constraint, which allows to smoothen the solution as the robot approaches its dynamic limits:

$$\sigma\left(\Delta T\left(\mathbf{D}(\mathbf{q},\dot{\mathbf{q}})+\underline{\tau}\right)+\mathbf{M}(\mathbf{q})\dot{\mathbf{q}}\right) \leq \mathbf{M}(\mathbf{q})\dot{\mathbf{q}}^{*} \leq \sigma\left(\Delta T\left(\mathbf{D}(\mathbf{q},\dot{\mathbf{q}})+\overline{\tau}\right)+\mathbf{M}(\mathbf{q})\dot{\mathbf{q}}\right) \quad (8)$$

4 Experiments

In this section we will show the application of the robot dynamics Constraint into a complex IK problem to perform a Cartesian task with the simulated model of our humanoid robot COMAN (in Fig 1). The task consists of moving both arms downwards generating a whole body squat motion. To show the effect of the dynamics constraint we highly reduce the available torques at the legs joints of 60%: from 50 $[Nm]$ to 20 $[Nm]$. We will show,in particular, that the joint torque at the knee is bounded in the limits. Apart from the robot dynamics constraint, we consider joint limits, joint velocity limits (up to 0.6 $\left[\frac{rad}{sec}\right]$).

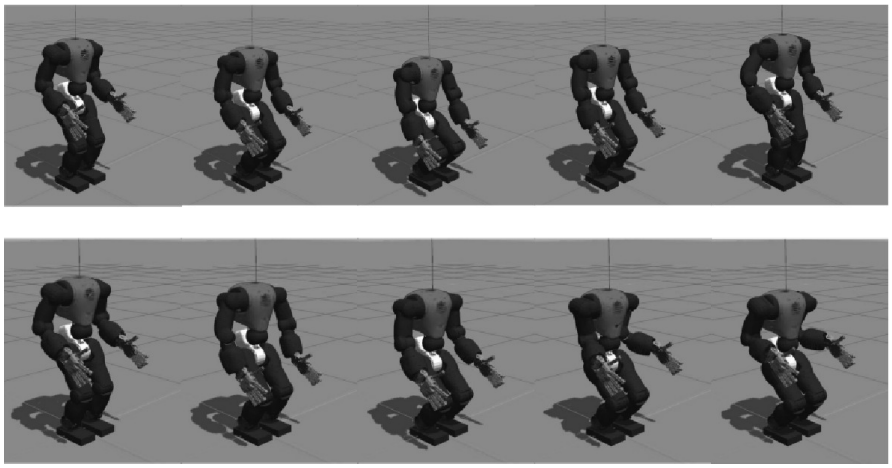


Fig. 2: COMAN performing the squat motion. The upper sequence results in an unfeasible motion considering the imposed torque limits while the second results in a feasible one

For the robot dynamics constraint we are using $\sigma = 0.85$ and we are filtering the sensed (simulated) wrenches at the force/torque sensors using a simple filter:

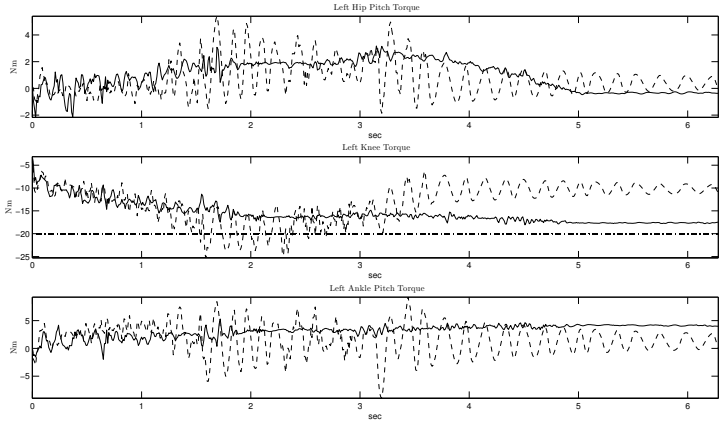
$$\mathbf{w}_{\mathrm{t}} \mathrel{+}= (\mathbf{w}_{\mathrm{t}} - \mathbf{w}_{\mathrm{t\text{-}1}})\,0.9 \quad (9)$$

The Cartesian task consists of a linear trajectory for the left and right hands, from the initial pose, to $0.18\,[m]$ down and then back again. Desired joint trajectories are

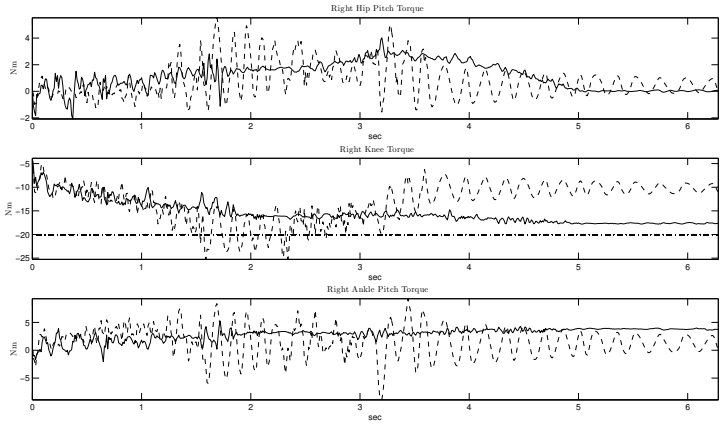
sent to the robot open-loop integrating the results obtained from the IK:

$$\mathbf{q}_{\mathrm{d}} = \mathbf{q} + \dot{\mathbf{q}}\Delta T \tag{10}$$

Measured joint velocities and force/torques at the ankles are used as feedback.



(a) Left leg



(b) Right leg

Fig. 3: Measured torques on the joints of the pitch joints in the legs while performing the task without (dashed lines) and with (continuous lines) the robot dynamics constraint. The constant line shows the limit on the torque of the knee joint

In Fig. 2 it can be observed the final motion performed by the robot when the robot dynamics constraint is not active (upper sequence) and when it is active (lower sequence)

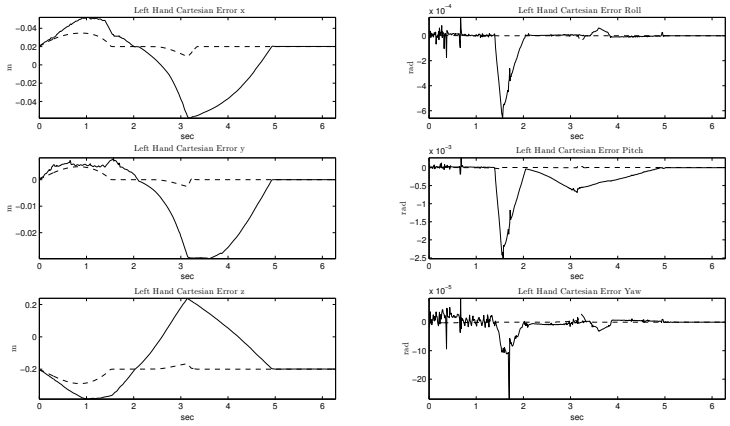


Fig. 4: Cartesian error on the left hand while performing the task without (dashed lines) and with (continuous lines) the robot dynamics constraint

Cartesian errors are shown in Fig. 4. Despite the higher Cartesian errors when using the robot dynamics constraint, the robot exceeds the imposed torque limits, at the joint knees, trying to keep the Cartesian error small when the robot dynamics constraint is not used. Fig. 3a and Fig. 3b shows in particular that the torques at left and right knees, respectively, remains in the imposed limits when using the robot dynamics constraint, while exceeds when not using it.

5 Conclusions

In this paper we have formulated the joints torque limit constraint at the velocity level, for fixed/floating base robots, to filter dynamically unfeasible motions. We presented the theoretical formulation and we showed results in simulation using our humanoid robot COMAN considering a whole body task involving also other constraints such as joint limits and joint velocity limits. We show that the robot dynamics constraint can make the task dynamically feasible and it is able to keep the torque at the knee joint on the given boundary limits. We think this is fundamental when working on real hardware as well as joint limits and joint velocity limits. This constraint is fundamental when Cartesian trajectories references are *aggressive*. Future works will consider the application of such constraint in more complicated

tasks, investigate the effect of the tuning of the σ parameter as well as the interaction with other constraints and the test on the real robot.

6 Acknowledgement

The research leading to these results has received funding from the European Union Seventh Framework Programme [FP7-ICT-2013-10] under grant agreements n.611832 WALKMAN.

References

1. Ferreau, H.J., Kirches, C., Potschka, A., Bock, H.G., Diehl, M.: qpoases: A parametric active-set algorithm for quadratic programming. Mathematical Programming Computation pp. 1–37 (2013)
2. Flacco, F., De Luca, A.: Discrete-time redundancy resolution at the velocity level with acceleration/torque optimization properties. Robotics and Autonomous Systems **70**, 191–201 (2015)
3. Herzog, A., Righetti, L., Grimminger, F., Pastor, P., Schaal, S.: Momentum-based balance control for torque-controlled humanoids. Computing Research Repository **1**, 1–7 (2013)
4. Kanehiro, F., Lamiraux, F., Kanoun, O., Yoshida, E., Laumond, J.P.: A local collision avoidance method for non-strictly convex polyhedra. Proceedings of robotics: science and systems IV (2008)
5. Mansard, N., Stasse, O., Evrard, P., Kheddar, A.: A versatile generalized inverted kinematics implementation for collaborative working humanoid robots: The stack of tasks. In: Advanced Robotics, 2009. ICAR 2009. International Conference on, pp. 1–6. IEEE (2009)
6. Nakamura, Y.: Advanced Robotics: Redundancy and Optimization, 1st edn. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA (1990)
7. Park, K.C., Chang, P.H., Kim, S.H.: The enhanced compact qp method for redundant manipulators using practical inequality constraints. In: Robotics and Automation, 1998. Proceedings. 1998 IEEE International Conference on, vol. 1, pp. 107–114. IEEE (1998)
8. Ramos, O., Mansard, N., Souères, P.: Whole-body motion integrating the capture point in the operational space inverse dynamics control. In: IEEE-RAS International Conference on Humanoid Robots (Humanoid'14). Madrid, Spain (2014)
9. Ramos, O.E., Mansard, N., Stasse, O., Benazeth, C., Hak, S., Saab, L.: Dancing humanoid robots: Systematic use of OSID to compute dynamically consistent movements following a motion capture pattern. IEEE Robot. Autom. Mag. **22**(4), 16–26 (2015)
10. Rocchi, A., Hoffman, E.M., Caldwell, D.G., Tsagarakis, N.G.: Opensot: a whole-body control library for the compliant humanoid robot coman. In: Robotics and Automation (ICRA), 2015 IEEE International Conference on, pp. 1093–1099. IEEE (2015)
11. Saab, L., Ramos, O.E., Keith, F., Mansard, N., Souères, P., Fourquet, J.Y.: Dynamic Whole-Body motion generation under rigid contacts and other unilateral constraints. IEEE Trans. Rob. **29**(2), 346–362 (2013)
12. Yamane, K.: Simulating and generating motions of human figures. STAR / Springer tracts in advanced robotics. Springer, Berlin (2004). URL `http://opac.inria.fr/record=b1119062`. Evolution of the author's PhD