A Dedicated Solver for Fast Operational-Space Inverse Dynamics

Nicolas Mansard

A Dedicated Solver for Fast Operational-Space Inverse Dynamics

N. Mansard

***Abstract*— The most classical solution to generate whole-body motions on humanoid robots is to use the inverse kinematics on a set of tasks. It enables flexibility, repeatability, sensor-feedback if needed, and can be applied in real time on-board the robot. However, it cannot comprehend the whole complexity of the robot dynamics. Inverse dynamics is then a mandatory evolution. Before application as a generic motion generator, two important concerns need to be solved. First, when including in the motion-generation problem the forces and torques variables, the numerical conditioning can become very low, inducing undesired behaviors or even divergence. Second, the computational costs of the problem resolution is much more important than when considering the kinematics alone. This paper proposes a complete reformulation of the inverse-dynamics problem, by cutting the ill-conditioned part of the problem, solving in a same way the problem of numerical stability and of cost reduction. The approach is validated by a set of dynamic whole-body movements of the HRP-2 robot.**

I. Introduction

Designing the whole-body motion of a humanoid robot directly in the joint space is a very tedious task, requiring a lot of trials and errors. On the opposite, the task-function approach provides an elegant way to formulate the motion objectives [1]. A dedicated task space (also called operational space [2]) is chosen, in which the control law driving the robot to the objective is easy to write. Transposing the control law from the task space to the whole-body joint space is then simply a matter of resolving a linear system [3].

On humanoid robots, there is typically several objectives to be accounted at the same time. Two approaches can be considered to fuse several tasks in a single control law. It is possible to sum several tasks in a single task, by performing a Cartesian product of the task spaces [4]. The tasks are generally weighted to give some respective importance between objectives when it is not possible to fulfill all of them. On the opposite, it is possible to define a strict hierarchy between tasks (called a stack of tasks [5]), by using the redundancy projectors [6], [7]. The coupling between concurrent tasks is then artificially nullified. It can be shown that a stack of tasks is the limit of a weighted tasks composition, when the weight magnitude orders reach the infinity [8]. In the following, we focus on hierarchy, as weighted composition is a special case of it.

The task-function approach is valid when considering only the robot kinematics [9], [10], but can be extended similarly to the robot dynamics [2], [11]. Hierarchy of tasks can also be generalized to inverse dynamics [12]. Connections between the two inverses can be found in [13]. Basically, inverse kinematics consists in finding the robot joint velocity that fits with the references velocity in the given task spaces. On the opposite, inverse dynamics consists in finding the joint accelerations and torques along with the external contact forces, that fit the reference accelerations and forces in the given task spaces. If mathematically, both problems can be written under a similar shape, the number of parameters in the second case is much more important: two times more variables in the case of a robot without contact, and up to four times more for classical cases on a humanoid robot. Since the inversion methods have a quadratic cost in the number of parameters, inverse dynamics involves much more computation than inverse kinematics [14].

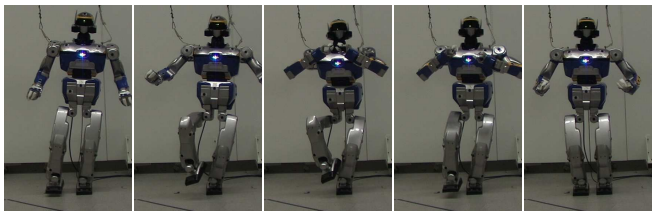

Fig. 1. Reaching "Standing Lotus": dynamic motion from motion-capture

The computation cost is composed on one side of the cost of the numerical resolution, and on the other side, of the cost of the problem formulation (typically, computing the numerical quantities, Jacobians, inertia matrices, etc). A complete resolution of the inverse dynamics requires to compute explicitly all the variables (accelerations, torques and forces). However, at the end, only the control variables are needed. A first solution to reduce the cost is to reformulate the problem to hide the unnecessary variables and avoid their computations. This is typically what is done in general in the work of Khatib *et al.* [2], [12], [15] (detailed in Section II-B). This solution enables to reduce the cost of the inversion. However, the formulation of the numerical quantities of the reduce problem induces a supplementary computation cost, and a supplementary design cost as the reduction is not automatic. Moreover, it is not possible to express constraints that requires explicitly the hidden force and acceleration variables (e.g. center of pressure).

A very appealing solution is to reduce together the formulation and inversion costs, as done for non-contacting manipulator robot [16] (detailed in Section II-A). However, such a solution seems impossible to generalize.

On the opposite, in [4], [13] (detailed in Section II-C), a quadratic-problem (QP) solver is used that can solve literally the whole set of variable. In that case, no reformulation at all is needed. The interest of this method is its genericity:

N. Mansard is with CNRS, LAAS, University of Toulouse (UPS), France. nicolas.mansard@laas.fr

the whole dynamics is taken into account, and constraints (equality or inequality) can be expressed on any part, acceleration, torque or forces. However, computation-cost overhead is very important and do not allow to obtain real-time control. Moreover, since there is no artificial decoupling between the spaces of actuation and movement, low condition number can appear, that are not acceptable on a control scheme.

In this paper, we propose to keep this last approach by enabling a generic solver working on all the variables of the system. However, following the spirit of [12], we propose to automatically reduce the variables spaces along the null or ill-conditioned spaces, in order to both save computation cost and improve the condition number of the numerical problem. We first recall the previous approaches in Section II. Our original solution is presented in Section III. The method is validated on the humanoid robot HRP2 in Section IV.

II. Operational-space control

The robot configuration is denoted by q[1]. The joint torque is denoted by τ. The dynamic equation of the system is then:

$$A\ddot{q} + b = \tau \tag{1}$$

where A is the generalized inertia matrix, and b is the dynamic drift (gravity and Coriolis forces summed). The robot objective is given under the shape of a task function $e_1(q)$. The Jacobian of this function is denoted by $J_1 = \frac{\partial e_1}{\partial q}$:

$$\dot{e}_1 = J_1\dot{q} \tag{2}$$

The image space of $e_1(q)$ is called the task space. In this task space, a control law that bring the system to an stable position is given by $\ddot{e_1}^*$[2]. The operational-space inverse dynamics is to find $\ddot{q}$ and τ such that $\ddot{e_1} = \ddot{e_1}^*$. The size of τ is denoted by n_τ, while the size of e_1 is denoted by n_1 (generally, $n_1 < n_\tau$).

A. Basic operational-space inverse dynamics

A first solution is to find $\ddot{q}$ from the derivative of (2):

$$\ddot{e_1} = J_1\ddot{q} + \dot{J}_1\dot{q} \tag{3}$$

Inverting (3) gives a minimal $\ddot{q}$ respecting $\ddot{e_1}^*$. Then, using (1), τ is obtained:

$$\tau = AJ_1{}^{\#}(\ddot{e_1}^* - \dot{J}_1\dot{q}) + b \tag{4}$$

where $.^{\#}$ is any generalized inverse [14], typically the pseudo inverse $.^{+}$ or a left-weighted inverse $.^{\#W}$ defined by

$$X^{\#W} = WX^\top(XWX^\top)^+ \tag{5}$$

where W is any user-defined positive definite symmetrical matrix. This solution only results in a minimal $\ddot{q}$, while minimal τ is generally preferred. For that, a torque-based equation is obtained by multiplying (1) by J_1A^{-1}, and inserting (3):

$$\ddot{e_1} - \dot{J}_1\dot{q} + J_1A^{-1}b = J_1A^{-1}\tau \tag{6}$$

[1]For mobile robots, the first part of the configuration is the free-floating configuration x_f. The first part of $\dot{q}$ is the free-floating velocity ν_f, which is not integrable. The vector $\dot{q}$ is abusively used in the paper.

[2]The notation x^* will always designate the reference value of x.

The control law is obtained by inverting J_1A^{-1}:

$$\tau = (J_1A^{-1})^{\#}(\ddot{e_1}^* - \dot{J}_1\dot{q} + J_1A^{-1}b) \tag{7}$$

Any weight can be used to define the inverse $.^{\#}$ [11]. In [2], it is generally chosen to weight by the inertia matrix. In that case, unrolling (5) gives:

$$\tau = J_1{}^\top\Lambda_1(\ddot{e_1}^* - \dot{J}_1\dot{q} + J_1A^{-1}b) \tag{8}$$

with $\Lambda_1 = (J_1AJ_1{}^\top)^{\#}$ the operational-space inertia matrix. In general, it is better for the numerical algorithm to compute directly the pseudo inverse (e.g. using a singular value decomposition, or an extension for weighted inverse [12]), rather than inverse the quadratic shape $XWX^\top$ whose condition number is very low when X is ill conditioned.

In [16], it is proposed to compute analytically the inertia Λ_1, directly from the robot model. The computation cost of the problem formulation is then reduced, since instead of computing A, only Λ_1 of smaller size is needed. Moreover, the inversion cost is also reduced, since no explicit inversion algorithm is called, and the cost is finally reduced to the multiplication of two matrices. This approach has only been applied for controlling the position and orientation of a Cartesian point of the robot. Its generalization to other types of tasks (controlling the robot gaze for example) is not immediate. Moreover, a dedicated algorithm to compute Λ_1 is needed for each type of task, which is not tractable for humanoids with a wide range of possible tasks.

B. Projected inverse dynamics

Eq. (1) is limited to non-contacting robots. For humanoid robots, two other constraints have to be accounted: the robot is in contact with the world, and it cannot actuate its free-floating basis. In general, the contact study is limited to rigid contacts. The dynamic equations are thus:

$$A\ddot{q} + b + J^\top f = S^\top\tau \tag{9}$$

$$J\ddot{q} + \dot{J}\dot{q} = 0 \tag{10}$$

with f the contact forces, J the Jacobian of the contact point. In case of multiple contacts, the forces are simply stacked in the vector f. The size of f is denoted by n_f. Due to the under actuation, the size of $\ddot{q}$ is now $n_\tau + 6$.

1) For one task: Now, the problem is to find the parameter $x = (\ddot{q}, \tau, f)$ such that the task constraint $\ddot{e_1} = \ddot{e_1}^*$ is fulfilled. However, as previously, only the control input τ is needed, f and $\ddot{q}$ do not have to be explicitly computed in general. A first solution was proposed in [12] to reduce the whole dynamics to a single link between τ and $\ddot{e_1}$. First, the contact forces can be computed from $\ddot{q}$ and τ, by multiplying (9) by JA^{-1} and replacing the result by (10):

$$(JA^{-1}J^\top)f = JA^{-1}S^\top\tau - JA^{-1}b + \dot{J}\dot{q} \tag{11}$$

If $(JA^{-1}J^\top)$ is invertible, (9) can be projected into the null space of (10) by replacing (11) in (9). This produces a force-free dynamic equation with a pattern similar to (1):

$$A\ddot{q} + Nb + b_c = NS^\top\tau \tag{12}$$

where $N = I - (JA^{-1})^{\#}JA^{-1}$ is the projector in the null space of JA^{-1}, and $b_c = (JA^{-1})^{\#}\dot{J}\dot{q}$. Using the same method as in the non-contact case, the dynamics is then projected in the task space to hide the acceleration variable:

$$\ddot{e_1} + \mu_1 = J_1 A^{-1} N S^\top \tau \tag{13}$$

with $\mu_1 = -\dot{J}_1\dot{q} + J_1 A^{-1}(Nb + b_c)$. The minimal torque is obtained by inverting $J_1 A^{-1} N S^\top$. In [15], it is proposed to use $W = SA^{-1}NS^\top$ to weight this last inverse:

$$\tau = (J_1 A^{-1} N S^\top)^{\#W}(\ddot{e_1} + \mu_1) \tag{14}$$

The pattern of this last solution is similar to (7). It is even possible to rewrite the equation to obtain a shape with a pseudo inertia matrix Λ. However, it is not possible to obtain a nice simplification of it that allows an analytical computation of the inertia, or of any part of the inverse.

Consequently, this solution involves the cost of the inversion of the size of τ ($o(n_\tau n_1^2)$). However, the corresponding formulation implies a cost: computing N implies a cost of $o((n_\tau + 6)^3)$, with similar costs for W and $J_1 A^{-1} N S^\top$.

2) Extension to a stack of tasks: Until now, we have only considered a single task. The equations can be easily generalized to a hierarchized set of tasks $(e_1, ...e_m)$. Indeed, (14) is the minimal W-norm solution. The generic solution involves a secondary input τ_2:

$$\tau = (J_1 A^{-1} N S^\top)^{\#W}(\ddot{e_1} + \mu_1) + P_1\tau_2 \tag{15}$$

with P_1 the projector in the null space of $J_1 A^{-1} N S^\top$ along W, and τ_2 any secondary input. This secondary input can be used to fulfill a second task, by replacing τ in (12), and projecting the resulting dynamics in the second task space e_2. The resolution of the secondary task is:

$$\tau_2 = (J_2 A^{-1} P_1 N S^\top)^{\#W}(\ddot{e_2} + \mu_{2|1}) + P_2\tau_3 \tag{16}$$

where $\mu_{2|1}$ is the task drift due to the dynamics of the task e_1, P_2 is the projector corresponding to the inverse, and τ_3 is a third input used to propagate the recurrence to m tasks.

This solution costs the inversion of a stack of tasks ($o(n_\tau n^2 m) \approx o(n_\tau^3)$. There is no additional costs of formulation compared to the solution for only one task.

C. Explicit QP formulation

1) For one task: Formulating the whole dynamics into the task space by (13) is appealing, first by its similarity with the classical inverse kinematics, and by its conciseness that spares computation during the inversion. However, the formulation is tedious and costly. On the opposite, it is possible to formulate directly the inverse-dynamics problem as a QP, that is straight forward to formulate and invert. Basically, the problem can be written: find the variables $\ddot{q}, \tau, f$ that are consistent with the dynamic equations and minimize the distance to the task reference:

$$\min_{\ddot{q},\tau,f} ||J_1\ddot{q} + \dot{J}_1\dot{q} - \ddot{e_1}^*||^2 \tag{17}$$

$$\text{subject to } A\ddot{q} + b + J^\top f = S^\top\tau$$

$$J\ddot{q} + \dot{J}\dot{q} = 0$$

Another interest of the QP formulation is that it enables directly the use of inequality constraints, which was not possible with the solutions based on the pseudo-inverse described above. For example, it is easy to add the joint-limit constraints in the problem (17) [4].

The contact model (10) is partial. Indeed, when considering a point contact where the robot position along the contact normal is denoted by z, the entire model is:

$$\ddot{z} \geq 0 \quad \perp \quad f^\perp \geq 0 \tag{18}$$

The motion along the normal is null or the force is null. A similar constraint can be written for the tangential directions, which will not be considered here. By writing (10), we implicitly chose the solution to be in the second case: positive normal forces. This constraint is never explicitly checked in the previous methods, while it can be directly taken into account by the QP formulation:

$$f^\perp \geq 0 \tag{19}$$

where $f^\perp = S^\perp f$ are the normal components of f.

Another difference between QP and pseudo-inverse-based methods is the management of the redundancy. Among the possible solution (14) will chose the one that minimizes the torque variable. This choice acts as a third level of constraints, the dynamic equations being the first level and the task reference at the second level. Such a behavior is not possible with a standard QP. Similarly, it is not possible to account for a secondary task e_2, except by fusing it with e_1, loosing the notion of hierarchy.

2) Hierarchized QP: In [13], we have proposed to use an extended hierarchical QP (HQP) solver to solve the operational-space inverse dynamics. A classical QP can be seen as a hierarchy of two levels, the first one, having priority, used to constraint the dynamic consistency, the second one being used for the task reference. A HQP is simply a generalization of the classical QP to an arbitrary number of hierarchy levels. The task hierarchy is denoted by $\prec$. A dynamic stack of m tasks is (9) $\prec$ (10) $\prec$ (19) $\prec$ (3.1) $\prec$ ... $\prec$ (3.m) $\prec$ (20), where the last constraint level

$$\tau = 0 \tag{20}$$

enforces a minimization of the motor torques.

This control scheme generalizes the QP-based inverse-dynamics solver, and has been proved equivalent to the pseudo-inverse based solvers in [13] when not considering any inequalities. Its formulation is free of overhead, needing only the direct quantities (Jacobians, inertia matrix). However, its resolution for a full stack (when all the degrees of freedom -DOF- of the robot are used) is $o((2n_\tau + 6 + n_f)^3)$

3) Ill-condition of motion-actuation coupling: Apart from its cost, a second problem can be noticed with the explicit HQP formulation. Due to the dynamic equations, the variable space $x = (\ddot{q}, f, \tau)$ can be divided theoretically in two sub spaces: the motion space, where the acceleration can be chosen freely (setting accordingly the necessary force and torque variables), and the actuation space, where the

acceleration is fixed (practically 0) and only forces can be chosen. However, in practice, the distinction between motion and actuation is equivalent to the problem of deciding if a value is null: it relies on a fixed threshold. This is a generic problem, that has to be solved explicitly when computing the projector (12) for pseudo-inverse based methods. This problem is linked to the decision of the singularity of a task.

However, when accounting for inequality constraint in a QP solver, the threshold cannot be arbitrarily selected: it is fixed and depends on the precision of the solver, generally $1e^{-8}$. The consequence is that three subspaces of the variables have to be considered: the motion and actuation spaces, and a ϵ-motion space, where motion variables are free in theory, but in exchange of impractical force values.

D. Conclusion

Two classes of solutions for operational-space inverse dynamic have been described. The first class is based on the pseudo inverse. The cost for a full stack of tasks is in $o(n_\tau^3)$. The drawbacks are the impossibility to account for inequalities, and the specific treatment to formulate the problem, tedious to write and costly. On the other hand, QP are easier to formulate, and can account for more generic constraints, involving *hidden* variables and inequalities. However, the cost is more important and the simultaneous treatment of contact and tasks can involve ill-conditioned subspaces.

In the next section, we will propose to automatically formulate of a reduced HQP, where the actuation space is explicitly distinguished from the motion space. This formulation keeps the possibility to have generic constraints, but for a cost lower than both classes of solution, and with an explicit and arbitrary condition threshold.

III. Decoupled dynamics

The idea is to explicitly separate the variable spaces between motion and actuation. Then, instead of keeping the explicit variables we will rely on the basis of the two decoupled, that are of lower dimension than the coupled original variables. However, due to the consistency with the inertia matrix, the decoupling is not trivial.

A. Motion-force decoupling

The inertia matrix is decomposed into an inverse Choleski:

$$A^{-1} = BB^\top \qquad A = B^{-\top}B^{-1} \tag{21}$$

where B, the square-root of A^{-1}, is invertible triangular (no zero term on the diagonal). Using the decomposition, the dynamics can be rewritten:

$$B^{-1}\ddot{q} + B^\top J^\top f + B^\top b = B^\top S^\top \tau \tag{22}$$

For shortcuts, we denote $G = JB$. The projector into the null space of G is denoted by P. Using the pseudo-inverse constructive definition:

$$P = I - G^+G \tag{23}$$

Multiplying (22) by P gives:

$$B^{-1}\ddot{q} - G^+J\ddot{q} + PG^\top f + PB^\top b = PB^\top S^\top \tau \tag{24}$$

The second term is constant with respect to the variables $(\ddot{q}, \tau, f)$ since $J\ddot{q} = \dot{J}\dot{q}$. The third term is null by definition of P. The dynamic evolution under constraint of contact is then:

$$B^{-1}\ddot{q} + PB^\top b + G^+\dot{J}\dot{q} = PB^\top S^\top \tau \tag{25}$$

or, by multiplying by $B^{-\top}$ and using the classical notations:

$$A\ddot{q} + Nb + B^{-\top}G^+\dot{J}\dot{q} = NS^\top \tau \tag{26}$$

with $N = B^{-\top}PB^\top$ the projector (12).

B. Contact constraint

Alternatively, the contact constraint can be rewritten by multiplying (22) by G:

$$GG^\top f + GB^\top b - \dot{J}\dot{q} = Gz \tag{27}$$

with $z = B^\top S^\top \tau$. Computing τ is equivalent to computing the variable z. When considering the last equation on z, it clearly appears that a sub manifold of possible z values has no effect on the motion, but only on the internal forces. This space is then useless since it is already considered in the f variables. To reduce the space where z is searched, we simply have to search it under a shape that is consistent with (27):

$$z = G^\top f + Vu + B^\top b + \delta_c \tag{28}$$

where $\delta_c = -G^+\dot{J}\dot{q}$, V is a basis of the null space of G ($P = VV^\top$) and u is any vector in this kernel. Introducing z in (22) gives:

$$B^{-1}\ddot{q} = Vu + \delta_c \tag{29}$$

This very simple equation is the reduced motion dynamics implied by contact constraints. It is fully decoupled from the force dynamics, and reduced to the lower dimension V space. The possible motions are then directly given by the subspace VB^{-1}. The projection of the reference motion into the subspace of possible motion is simply:

$$u = V^\top(B^{-1}\ddot{q} - \delta_c) \tag{30}$$

C. Forces resolution

Consider that a reference feasible motion is given under the shape of a reduced u^*. It is now necessary to compute the actual necessary torques and forces. As said upper, in most cases, there is a redundancy of actuation. Therefore, there is not a single possible pair of (f, τ), but a set of possible values. However, the necessary external forces can be explicitly expressed in the null space of the selection matrix. The selection of the floating part is denoted by $\bar{S}$ ($\bar{S} = [I\ 0]$, $\bar{S}S^\top = 0$). Multiplying (28) by $\bar{S}B^{-\top}$ gives:

$$\bar{S}J^\top f + \bar{S}b + \bar{S}B^{-\top}\delta_c = -\bar{S}B^{-\top}Vu \tag{31}$$

In the case of a single contact with reduced forces, $\bar{S}J^\top$ is 6x6 invertible, and the contact forces can be computed uniquely. Otherwise, any forces satisfying the previous equality are admissible. The torques are obtained by subtraction:

$$\tau = S(J^\top f + B^{-\top}Vu + b + B^{-\top}\delta_c) \tag{32}$$

If searching for a particular $\ddot{q}$ respecting some linear constraints J_t, e_t, under inequality constraint on f, the optimization problem will be written:

$$\min_{u,f} \quad ||J_t B V u - e_t||^2 \tag{33}$$

$$s.t. \quad \bar{S}J^\top f + \bar{S}b + \bar{S}B^{-\top}\delta_c = -\bar{S}B^{-\top}Vu \tag{34}$$

$$f^\perp \geq 0 \tag{35}$$

The problem of this formulation is that there is still many 0 on the cost function due to the invariance to f, and many 0 on the inequality part due to the invariance to u. Moreover, f is possibly still of large size, while only a reduced space acts on the problem.

D. Reduction of the force variable

Due to the underactuation, the only constraint on the force is to ensure the full actuation of the floating DOF. The constraint can be written:

$$\bar{S}J^\top f = -\bar{S}(B^{-\top}Vu + b + B^{-\top}\delta_c) \tag{36}$$

The variable f can be searched under the following shape:

$$f = -(\bar{S}J^\top)^+\bar{S}(B^{-\top}Vu + b + B^{-\top}\delta_c) + K\psi \tag{37}$$

where K is a basis of the null space of matrix $\bar{S}J^\top$. Because B is triangular, the previous equality can be reduced to:

$$f = -(\bar{S}G^\top)^+\bar{S}(Vu + B^\top b + \delta_c) + K\psi \tag{38}$$

The optimization problem can be reformulated consequently:

$$\min_{u,\psi} \quad ||J_t B V u - e_t||^2 \tag{39}$$

$$s.t. \quad -(\bar{S}G^\top)^+\bar{S}Vu + K\psi \geq f_0 \tag{40}$$

with f_0 any solution of $\bar{S}G^\top f_0 = B^\top b + \delta_c$, for example $f_0 = (\bar{S}G^\top)^+(B^\top b + \delta_c)$.

The problem is directly generalized to a stack of tasks using the following HQP: (40) $\prec$ (3.1) $\prec$ ... $\prec$ (3.m). A last stage can be added to minimize the torques. However, it is generally better for the system stability to add sufficient tasks to fill up the stack. Then, there is no more redundancy to minimize the torques. In the case where the stack is not full, a last task of friction or posture (see Section IV) is generally added to complete it.

E. Spatial-force reduction

In practice, each contact is defined by a finite set of convex points where 3D point contacts occurs. The points are fixed with respect to one of the robot body. The contact forces summed into a 6D spatial force (linear and angular components), expressed in the body coordinates system at the central point. From the forces at the contact points denoted f_i, the force $f = \sum_{i=1}^n f_i$ and torque $\tau = \sum_{i=1}^n p_i \times f_i$ at the central points can be computed. The Jacobian of the contact can by written:

$$J = XJ_6$$

where J_6 is the 6D Jacobian of the frame attached to the body, and X is a $3n \times 6$ matrix involving the cross product with p_i. When the contact happens on three points or more of the same body, X is full column rank. This property is then used to reduce the computation cost of $V = ker(J_6)$.

F. Conclusion

Finally, the solver is reduced to the variable $x = [u, \psi]$, whose size is typically equal to $n_\tau + 6$. The computation cost for a full stack of tasks is thus in $o((n_\tau + 6)^3)$, similar to the cost of the pseudo-inverse based solution. If using a proper solver, like the one proposed in [17], it should even be faster in practice, as shown in the experiments.

The formulation of the problem is reduced to the computation of V and δ_c. The two quantities are computed from a QR decomposition of the matrix J_p, for a total cost of $o(k^2)$, with k the number of joints of the robot that are linked with a contact (typically, $k = c * 6 + 6$, each contacts occurring on an end-effector after a 6-joint limb, plus 6 for the free flyer). The cost of K can be neglected, since this computation is only needed once in a while. Having the decomposition corresponding to K, the computation of f_0 is negligible.

The proposed solution offers a versatile implementation of the dynamic stack of tasks. The dynamic balance of the robot is ensured by (40), that can be shown to be equivalent to the classical zero-momentum point (ZMP) condition in the case of coplanar contacts. The classical task functions can then be added to manipulate the robot motion.

IV. Experiments

The control law presented above is theoretically strictly equivalent to the explicit HQP proposed in [13], that was previously shown equivalent to [15] if limited to equalities. The objective is thus not to validate the motions that it is possible to generate, but to compare the computation times. The motions presented below consist in dynamic reaching of a robot pose that is defined by various set of tasks on the end-effector positions and orientation, a set of contacts, the robot gaze, or some joint positions. The tasks are the same as those used in [18]. The task sequence is defined by hand.

Three motions are presented below. They all have been designed for a dissemination event, where the robot is interacting with dancers. They are dynamic motions, in the sense that considering only the center of mass position is not sufficient to ensure the feasibility by the real robot: due to large accelerations or quantities of motion, the robot can loose contact if the ZMP constraint is not properly satisfied. The first motion is a simple but fast motion: with the two feet on the ground, the robot moves mainly its waist while keeping its hands fixed at a given position. The second movement requires to balance the robot body while making large upper-body movements standing on one foot. The last motion involves a large step while keeping the balance using a contact of the hand. The two last motions involves variations of the contact set during the sequence. The movies corresponding to each motion can be found at `homepages.laas.fr/nmansard/icra2012` .

For each motion, the three solvers, *PINV* based on the pseudo inverse (Sec. II-B), *HQP* based on the explicit HQP

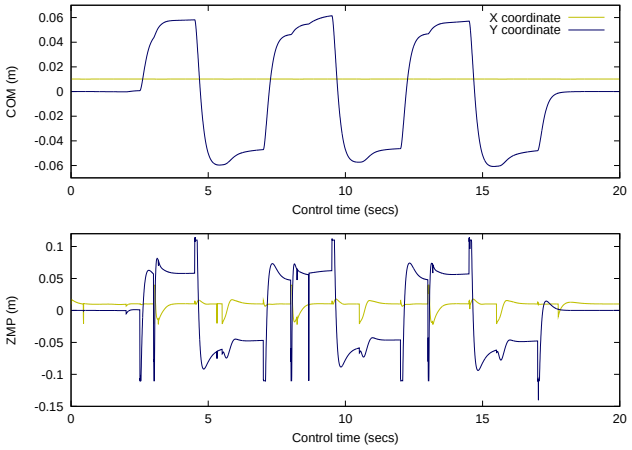

Fig. 2. Experiment A: COM and ZMP position: the COM fluctuates along the Y axis, while the ZMP is limited in the defined support polygon.

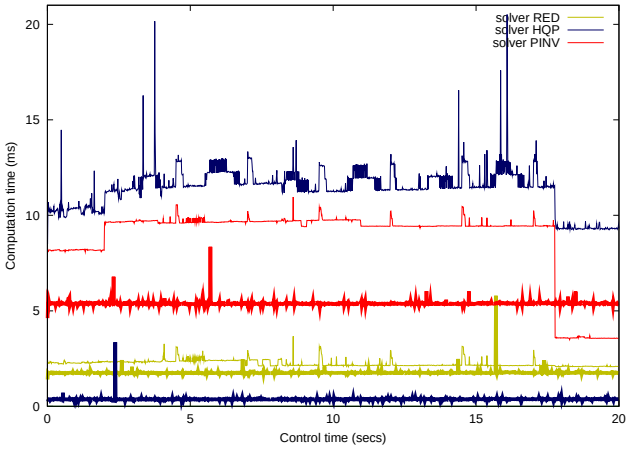

Fig. 3. Experiment A: formulation time (bold) and solver time (non-bold)

formulation (Sec. II-C), and *RED* based on the reduced dynamic HQP (Sec. III-D) are compared. Since the *PINV* solver cannot handle inequalities, the ZMP can not be constrained in *PINV*: the resulting motion can be dynamically inconsistent. In simulation, it is then necessary to artificially enforce the foot contact, in order to compare the computation times of each method. For the same reasons, in *PINV*, the joint-limit constraint has been removed, and the eventual floating COM transform to a fix attractor. To prove the validity of the obtained dynamics, the motion generated by *RED* for the experiments A and B has been realized by the robot. However, all the time-computation comparisons have been run on the same computer in simulation.

For each execution, two computation costs are measured: on the one side the cost of the solver, i.e. the time spent by the numerical resolution (call to the QP program for *HQP* and *RED*, call to the pseudo inverse operator for *PINV*), and on the other side the formulation cost, i.e. all the other computation (Jacobian, projectors, etc).

A. Experiment A: waist rotation

The robot hands are controlled to reach a fixed point in front of the robot shoulders. They are then constrained to keep this position, and to keep the rotation along the robot X axis (front axis). Another task is then added to move the robot waist position. The waist is successively driven to the four corners of a rectangle set along the Y-Z plane (perpendicular to the front axis), whose coordinates are top-left corner: (0,0.1,0.65) to bottom-right corner: (0,-0.1,0.35). The robot COM is constrained to stay along the segment linking the two robot feet. Finally, the friction task is used to limit the robot velocity without constraining the posture. The set of contact is constant during all the motion: the two feet are flat on the ground. To account for the flexibility in the ankles, the ZMP should be kept close to the center of the foot. The support polygon is then a rectangle of $5cm \times 3cm$ centered on the vertical the last ankle motor. The stack of tasks is (40) $\prec$ (joint-limits) $\prec$ (right-hand) $\prec$ (left-hand) $\prec$ (COM) $\prec$ (waist) $\prec$ (friction).

The COM and ZMP position are displayed on Fig 2. The computation times are shown on Fig. 3. The computation cost of the *PINV* solver only depends on the number of tasks in the stack: it is very stable through the time. On the opposite, the cost of the *HQP* varies each time a limit is reached. Due to the mentioned numerical problems (here caused by the leg closed kinematic chain), limits are very often hit, that causes some variations in the computation time. The variations when the stack size varies is also visible. In average, the *HQP* solver costs 30% more than the *PINV*. The cost of the *RED* solver is around four times less. The major costs is due to the friction task, that implies a large multiplication by BV. Therefore, its computation cost is nearly constant.

The computation cost of the formulation of *HQP* is null as expected. The size of the formulation of *RED* is mainly due to the computation of V. It is thus constant in this motion, since the contact set do not vary. Similarly for *PINV*, the computation cost is nearly constant. In both cases, the cost of the formulation is equivalent to the cost of the solver itself.

B. Experiment B: standing lotus

The robot starts with the two feet on the ground, then ups its right foot and moves it near its left knee, while bringing both hand close to contact in front of its shape, reaching thus the position *standing lotus*. The position is maintained during 2 secs, and the robot finally moves back to the initial position. The arms and right leg trajectories are captured from a real human demonstration, as well as the overall posture, as proposed in [18]. During the control, the trajectories are simply tracked by the classical tasks. The stack is (40) $\prec$ (right-hand) $\prec$ (left-hand) $\prec$ (COM) $\prec$ (waist) $\prec$ (gaze) $\prec$ (friction).

Snapshots of the motion on the robot are given on Fig. 1. The times are given in Fig. 4. The same observation as previously can be noticed. Additionally, the change in the contact set appears: the cost of the formulation for both *RED* and *PINV* diminish. The cost of the *RED* solver increases since the variable space is bigger during simple support. The cost of both *PINV* and *HQP* solvers remains invariant while the contact set changes.

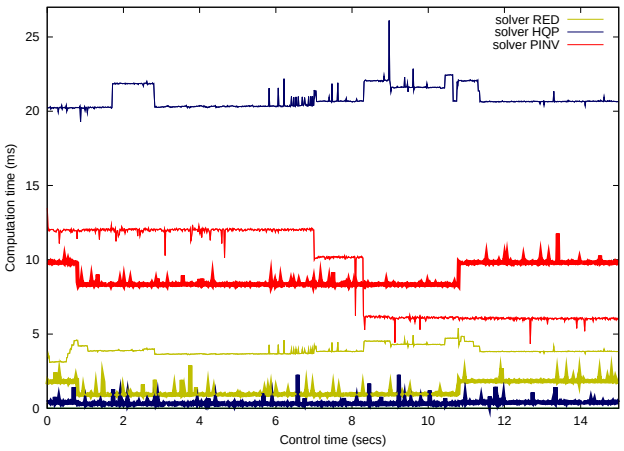

Fig. 4. Experiment B: formulation time (bold) and solver time (non-bold)

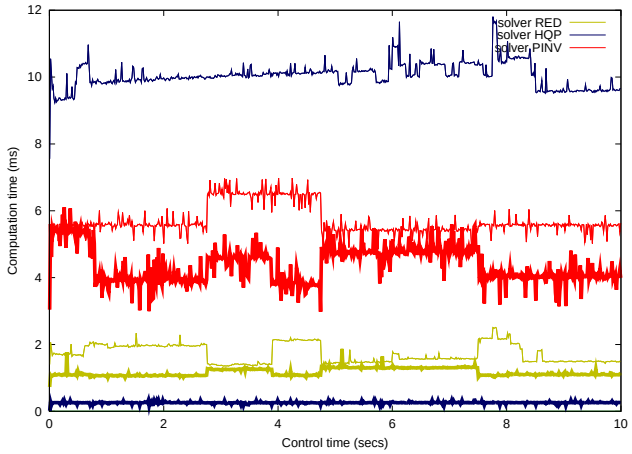

Fig. 5. Experiment C: formulation time (bold) and solver time (non-bold)

C. Experiment C: arm-contact step

This last motion is a typical example of the interest of considering the dynamics. It was already demonstrated in [17], [19]. The robot first contacts its left arm on a table. Then, it performs a large step with the right leg. The step is sufficiently small to be realized by the robot kinematics, but not when considering ZMP and joint limits. When the arm is in contact with the table, the ZMP is not defined anymore, but the constraint (40) is still valid. This motion presents several variations of the contact set (from 1 to 3 bodies).

The computation times are given in Fig. 5. The same observations as previously can be made. During this motion, ill-conditioned spaces appears during the triple contact phase, due to the quasi-null respective orientation of the feet. The contact forces estimated by the *HQP* solver reaches impossible values ($1e^7$). The forces chosen by the *RED* solver remain consistent.

V. Conclusion

In this paper, we have first compared QP- and pseudo-inverse- based dynamic motion generators, in term of formulation and computation cost. QP are more generic, and easier to set up, but more expensive and with possible numerical problems. On the opposite, pseudo-inverse methods are cheaper, but more complex to set up, and cannot comprehend inequalities, in particular the classical ZMP constraint.

We have then proposed a new formulation of the problem, to reduce its dimensionality, while keeping the capabilities of the QP solver. This solution allows fast resolution (around 4ms for typical HRP2 problems), and is numerically stable. The control scheme can comprehend the whole robot dynamics with complex set of equalities and inequalities constraints in real time at 200Hz. The solution was tested to generate dynamic movements on the real robot.

References

[1] C. Samson, M. Le Borgne, and B. Espiau, *Robot Control: the Task Function Approach*. Clarendon Press, Oxford, United Kingdom, 1991.

[2] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *International Journal of Robotics Research*, vol. 3, no. 1, pp. 43–53, Feb. 1987.

[3] A. Liégeois, "Automatic supervisory control of the configuration and behavior of multibody mechanisms," *IEEE Trans. on Systems, Man and Cybernetics*, vol. 7, no. 12, pp. 868–871, Dec. 1977.

[4] C. Collette, A. Micaelli, C. Andriot, and P. Lemerle, "Dynamic balance control of humanoids for multiple grasps and non coplanar frictional contacts," in *IEEE-RAS International Conference on Humanoid Robots (Humanoid'07)*, Dec. 2007.

[5] N. Mansard and F. Chaumette, "Task sequencing for sensor-based control," *IEEE Trans. on Robotics*, vol. 23, no. 1, pp. 60–72, Feb. 2007.

[6] J. Rosen, "The gradient projection method for nonlinear programmimg, part ii, nonlinear constraints," *SIAM Journal of Applied Mathematics*, vol. 9, no. 4, pp. 514–532, Dec. 1961.

[7] B. Siciliano and J.-J. Slotine, "A general framework for managing multiple tasks in highly redundant robotic systems," in *IEEE Int. Conf. on Advanced Robot. (ICAR'91)*, Pisa, Italy, June 1991.

[8] C. Van Loan, "On the method of weighting for equality-constrained least-squares problems," *SIAM Journal on Numerical Analysis*, pp. 851–864, 1985.

[9] B. Espiau, F. Chaumette, and P. Rives, "A new approach to visual servoing in robotics," *IEEE Trans. on Robotics and Automation*, vol. 8, no. 3, pp. 313–326, June 1992.

[10] P. Baerlocher, "Inverse kinematics techniques for the interactive posture control of articulated figures," Ph.D. dissertation, EPFL, 2001.

[11] J. Nakanishi, R. Cory, M. Mistry, J. Peters, and S. Schaal, "Operational space control: A theoretical and empirical comparison," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 737–757, June 2008.

[12] J. Park, "Control strategies for robots in contact," Ph.D. dissertation, Stanford University, California, USA, March 2006.

[13] L. Saab, N. Mansard, F. Keith, J. Fourquet, and P. Soueres, "Generation of dynamic motion for anthropomorphic systems under prioritized equality and inequality constraints," in *IEEE Int. Conf. on Robot.& Automation (ICRA'11)*, May 2011.

[14] G. Golub and C. Van Loan, *Matrix computations*. John Hopkins University Press, 1996.

[15] L. Sentis, "Synthesis and control of whole-body behaviors in humanoid systems," Ph.D. dissertation, Stanford University, USA, July 2007.

[16] K. Chang and O. Khatib, "Efficient algorithm for extended operational space inertia matrix," in *IEEE/RSJ Int. Conf. on Intelligent Robots and Systems (IROS'99)*, Kyongju, South Korea, Oct. 1999.

[17] A. Escande, S. Miossec, and A. Kheddar, "Continuous gradient proximity distances for humanoids free-collision optimized-postures generation," in *IEEE/RAS International Conference on Humanoid Robots*, Pittsburgh, USA, Nov. 2007.

[18] O. Ramos, L. Saab, S. Hak, and N. Mansard, "Dynamic motion capture and edition," in *IEEE-RAS International Conference on Humanoid Robots (Humanoid'11)*, Bled, Slovenia, Nov. 2011.

[19] S. Lengagne, A. Kheddar, and E. Yoshida, "Considering floatting contact and un-modeled effects for multi-contact motion generation," in *Workshop, IEEE Humanoids'11*, Bled, Slovenia, Nov. 2011.