Multi-Objective Control of Robots

Dimitar Dimitrov, Pierre-Brice Wieber, and Adrien Escande

"Prior to linear programming it was not practical to explicitly state general goals and so objectives were often confused with the ground rules for the solution. (...) Thus the means to attain the objective becomes the objective in itself, which in turn spawns new ground rules as to how to go about attaining the means (...). These means in turn become confused with goals."

G. B. Dantzig

Even though over 20 years old, the above quote still reflects well a common practice in the field of robotics. That is, not establishing a clear separation between: (i) what one wants to achieve, and (ii) how this must be done. Using high-level goals for posing real-world problems in mathematical terms can be advantageous since, at the level of modeling, one need not consider the particular technique for approaching their solution. In fact, being able to abstract the problem formulation can be viewed as a part of the revolutionary development that followed the birth of the field of linear programming [1], because practitioners could be trained to cast situations (of potentially great complexity) in terms of a set of general goals and rely on available solvers developed by specialists in the fields of numerical analysis and optimization. Such separation leads to a reliable solution process.

In this note we argue that the ability to define typical robotic problems in terms of lucid goals is beneficial not only from the point of view of clarity of presentation, but also for efficiency of computations. Our main point is that this can be achieved through the explicit use of multi-objective formulations. By means of several examples, we illustrate the advantages of this modeling approach and suggest that, by leveraging standard tools from the field of multi-objective optimization, such formulations can be resolved reliably and efficiently.

We summarize some recent developments of (the already classical in the field of robotics) task prioritization, among which the fact that hierarchical problems can be interpreted as a particular type of multi-objective models.

I. Multi-criteria decision making

Multi-criteria decision making has been popular for many decades and a variety of optimization techniques for facilitating it has been developed. The ability to handle multiple criteria provides more expressive models, leading to an increase in flexibility and reliability of design, control, and estimation schemes. The key question in many areas is how to define what is a good/desirable behavior of a given process and often the answer involves the specification of multiple conflicting objectives. Conflicting objectives occur naturally in typical robotics problems. For example, consider a manipulator arm mounted on a free-floating base. Then, due to the momentum conservation, the objective of tracking a given end-effector profile would conflict with the objective of preserving the posture of the base.

Resolving conflicts between objectives usually requires the participation of a human decision-maker who can express preference relations between alternative solutions. The involvement of a decision-maker can come at different stages of the solution process. In some applications, solving a multi-objective optimization problem is defined as the generation of the pareto-optimal surface, at which point a decision-maker can analyze the results and choose the most appropriate pareto-optimal point. This technique, however, might be inadequate to the needs of other applications *e.g.,* real-time robot control. An alternative approach, which we adopt, is based on including the preference information directly in the definition of the problem. In this way, during the modeling phase, one can ensure that only solutions with desired properties would be generated.

A. Goal programming

In this note we focus on one particular variant of multi-objective optimization referred to as *goal programming* (GP). This is one of the first formulations specifically dedicated to handling multiple criteria and still is one of the most widely used decision making techniques (a recent account can be found in [2]). Let $f_k(x)$, $k = 1, \dots, P$ denotes the k-th (scalar-valued) objective function. GP models can be classified into two major categories. In the first category, each objective function is assigned a non-negative weight according to its relative importance and the following single-objective optimization problem is defined

$$\underset{x \in \mathcal{X}}{\text{minimize}} \quad \lambda_k f_1(x) + \cdots + \lambda_P f_P(x), \tag{1}$$

where $\mathcal{X}$ is a set of constraints on the decision variables. In general, this technique is referred to as *scalarization* of a multi-objective problem. Note that different pareto-optimal points can be generated by varying the weights.

The second class of GP models addresses the case where the decision-maker cannot (meaningfully) assign finite trade-offs among the goals. That is, when the minimization of some objectives is infinitely more important that the minimization of others. In order to reflect this during the modeling stage, each objective is assigned a *priority level* (instead of a weight). This can be expressed as the following optimization problem

$$\underset{x \in \mathcal{X}}{\text{lex minimize}} \quad f = \left(f_1(x), \dots, f_P(x)\right), \tag{2}$$

where the vector-valued objective function f is minimized according to a lexicographic order.

An interesting statistics presented in [3] shows that the lexicographic approach is used in approximately 80% of the reported applications. Hence, lexicographic optimization is not only an attractive theoretical formulation but a widely used tool in practice.

B. Task prioritization: a lexicographic interpretation

Here, we draw parallel between the lexicographic optimization problem (2) and task prioritization. Recall that a common setting in robotics is to be given P tasks/goals of the form $A_k x = y_k$, $k = 1, \ldots, P$ with decreasing levels of priority, that is, solving the i-th system of linear equations (in a least-squares sense) is infinitely more important that solving the j-th one for $i < j$. Note how this problem can be cast directly in the form (2) by using $f_k(x, r) = \|r_k\|_2^2$, where $\mathcal{X}$ is defined as the set of pairs (x, r) that satisfy

$$\begin{bmatrix} A_1 \\ \vdots \\ A_P \end{bmatrix} x - \underbrace{\begin{bmatrix} r_1 \\ \vdots \\ r_P \end{bmatrix}}_{r} = \begin{bmatrix} y_1 \\ \vdots \\ y_P \end{bmatrix}.$$

The variable r_k can be interpreted as the constraint violation (or residual) of the k-th system of linear equations. Observe that, for this choice of the objective functions f_k, (1) is simply a weighted least-squares problem, and as demonstrated (both in theory and in practice) in [4] it is more expensive to solve compared to the lexicographic variant (2). The above equivalence can be taken one step further and introduce inequality constraints in the formulation by redefining $\mathcal{X}$ to be the set of pairs (x, r) that satisfy $b_k \leq A_k x - r_k \leq u_k$, $k = 1, \ldots, P$. Clearly, the equality-constrained problem is recovered by using $u_k = b_k$.

The fact that task prioritization can be interpreted as a lexicographic optimization problem was first pointed out in [5]. Such a recognition is important since classical approaches (developed in the field of multi-objective optimization) can be leveraged. One such approach, which is based on the solution of a sequence of inequality-constrained optimization problems, was rediscovered in the field of robotics only relatively recently in [6].

Beyond efficiently solving a hierarchical problem, casting it in the form (2) provides the possibility to consider some of the, already available, analysis related to good modeling practices and common pitfalls. It is interesting to observe that many of the issues discussed in [7] are relevant to robotics applications. In addition, there are existing numerical tools for handling hierarchies with objectives involving ℓ_1 and ℓ_∞ norms, which lead to solutions with different properties (that might be desirable in some situations). The fact that formulation (2) is rather abstract can be very convenient during the design phase of control schemes (as we discuss next).

II. Controller design

Modeling problems in a hierarchical framework can be very beneficial. Here we illustrate this on an example from the field of space robotics.

The equation of motion of a system with flexible base can be put in the following form [9]

$$\underbrace{\begin{bmatrix} H_m & H_{bm}^T \\ H_{bm} & H_b \end{bmatrix}}_{H} \underbrace{\begin{bmatrix} \ddot{\theta} \\ \ddot{x}_b \end{bmatrix}}_{\ddot{q}} + \underbrace{\begin{bmatrix} h_m \\ h_b \end{bmatrix}}_{h} = \underbrace{\begin{bmatrix} I \\ 0 \end{bmatrix}}_{S} \tau + \begin{bmatrix} 0 \\ d_b \end{bmatrix}, \qquad (3)$$

where $\ddot{\theta} \in \mathbb{R}^n$ and $\ddot{x}_b \in \mathbb{R}^6$ denote the generalized accelerations of the manipulator joints and base, respectively. The torques $\tau \in \mathbb{R}^n$ delivered by joint motors constitute the control input of the system, while the disturbance $d_b = -D_b \dot{x}_b - K_b \Delta x_b$ excites the base dynamics, where D_b and K_b are positive-definite (symmetric) matrices and Δx_b is the offset of the base from its equilibrium posture. For the motivation behind such a model refer to [10].

One of the typical problems that should be addressed in the context of systems with flexible components is the occurrence of vibrations. Assuming that such vibrations have already been excited, we are interested in finding manipulator motions that would facilitate their suppression. A controller that achieves this can be designed by finding a feasible pair $(\ddot{q}, \tau)$ such that $\ddot{x}_b = -G_b \dot{x}$ with G_b being an appropriately chosen positive-definite (symmetric) matrix. Intuitively, the aim is to produce base accelerations that damp the base velocity (and thus the vibrations). In [9] this has been demonstrated to lead to a damped vibration closed-loop system (with desired properties). In a hierarchical form, such a controller can be described as

$$(H\ddot{q} = S\tau - h) \succ (\ddot{x}_b = -G_b \dot{x}) \succ (\ddot{\theta} = 0), \qquad (4)$$

where the decision variables are $(\ddot{q}, \tau)$. We will adopt the above notation as a convenient alternative to explicitly stating a lexicographic least-squares problem in the form (2) (a similar approach is taken in [8]). Solving (4) implies finding a pair $(\ddot{q}, \tau)$ that satisfies the dynamics of the system, and $\ddot{x}_b$ damps the base velocity as much as possible (in a least-squares sense). The terminal objective is optional and ensures obtaining a solution such that the Euclidean norm of the joint accelerations is minimized (the use of other terminal objectives could be envisioned). Note that d_b is assumed to be unmodeled dynamics.

It is worth pointing out that one can reformulate (4) in a variety of ways and still achieve the vibration suppression task $\ddot{x}_b = -G_b \dot{x}$. Probably the simplest reformulation would be to eliminate τ from the decision variables and modify the first objective as $H_{bm}\ddot{\theta} + H_b \ddot{x}_b = -h_b$. Once a solution $\ddot{q}^\star$ is obtained, one can compute the corresponding torques by substituting $\ddot{q}^\star$ in the upper part of (3). This procedure leads to the same result as (4). Another possibility would be to eliminate $\ddot{q}$ by expressing $\ddot{\theta}$ as a function of τ and using $\ddot{x}_d = -G_b \dot{x}$. After substituting the resultant expression into the lower part of (3) one can solve for the joint torques. If the Moore-Penrose pseudo-inverse is used for obtaining $\tau^\star$, this method would be equivalent to the hierarchy (4) but with a terminal objective $\tau = 0$. Out of various alternatives, the authors of [9] choose to make yet another elimination which leads to the same result as (4).

Performing different variable eliminations puts emphasis on **how to solve** rather than **what to solve**. In many cases

this can obscure the controller design. For example, in [9] it is not emphasized that the actual vibration suppression task is $\ddot{x}_b = -G_b\dot{x}$. This is a very common practice in robotics. The urge to eliminate variables has led to a variety of formulations of equivalent control schemes – a similar observation can be found in [11]. Contrary to common beliefs, however, reducing the number of variables (by using various elimination techniques) does not necessarily lead to faster computations [12]. In fact, if not properly designed, such problem reformulations may lead to a much slower or even unreliable solution process.

Keeping things at a slightly more abstract level (*e.g.,* formulation (4)) emphasizes the actual tasks and not a particular approach for their resolution. In this way, for example practitioners can easily understand the motivation behind a given approach instead of being lost in derivations that are often not directly related to it. Hence, they would be able to actively contribute during the model development, given their hands-on experience. The above argument is very close to the original rationale behind the operational-space formulation, which aims at "The description, analysis, and control of manipulator systems with respect to the dynamic characteristics of their end-effectors" [13]. The main difference is that we do not insist on obtaining a formulation in minimal number of coordinates. With reference to hierarchy (4), note that the motion profile of the reference point on the base body, appearing in the second task, strictly inherits (by virtue of being with lower priority) the dynamical characteristics imposed by the system at the first hierarchical level. Thus, one can think of a lower-level task, that might involve only "kinematic quantities", related to a given end-effector as an implicit operational space formulation.

Choosing to reduce the number of levels of a given hierarchy to the point that only a minimal number of coordinates is left might, in some cases, be important for the purposes of analysis. What we argue above is that implementing literally such "analytical" formulas should be avoided since often it results in inefficient computations (Section III includes some further remarks). Moreover, the benefits of avoiding preliminary problem condensing becomes especially evident when inequality constraints are considered. Our general conclusion here is that one should not be "afraid" to formulate problems with large number of variables since it is not the size but the structure of a problem that counts – and a hierarchy leads to such structure which, if exploited properly, can result in very fast and reliable computations (see [5], [4]).

III. Analytical expressions

Here we include some remarks about the direct use of analytical expressions in numerical computations. We motivate the discussion with an example. Assuming zero initial momentum and no external forces acting on the system (*e.g.,* $d_b = 0$), the lower part of (3) can be integrated to obtain the momentum conservation equation

$$\underbrace{\begin{bmatrix} H_{bm} & H_b \end{bmatrix}}_{H_c} \dot{q} = 0 \tag{5}$$

Furthermore, let the relation between $\dot{q}$ and the velocity $\dot{p}$ of some end-effector be defined as

$$\underbrace{\begin{bmatrix} J_m & J_b \end{bmatrix}}_{J} \dot{q} = \dot{p}. \tag{6}$$

The hierarchy (5) $\succ$ (6) (with decision variable $\dot{q}$) is popular in the field of space robotics. If $\dot{x}_b$ is eliminated from it, one recognizes the Schur complement of H_b

$$J_G = J_m - J_b H_b^{-1} H_{bm} = J \underbrace{\begin{bmatrix} I \\ -H_b^{-1} H_{bm} \end{bmatrix}}_{N_c}, \tag{7}$$

which has been labeled as the "generalized Jacobian" matrix [14]. Note that the columns of N_c form a basis for the null-space of H_c.

Matrices of the type JN occur naturally after the elimination of equality constraints. The machinery of such elimination, however, permits choosing matrices N of various properties and sizes, *e.g.,* N could be chosen to be a projection matrix, in which case JN is sometimes referred to as a "restricted Jacobian" [15]. Depending on the specific application, various names can be encountered. For example several elimination steps on the hierarchy $(\omega = 0) \succ$ (5) $\succ$ (6) lead to the so called "fixed-attitude restricted Jacobian" [16]. In [17], JN has been called a "task-consistent posture Jacobian" etc.

It is worth pointing out that, in the robotics literature, the motivation behind choosing a specific N is often missing or incomplete. Furthermore, this choice is rarely made bearing in mind actual numerical computations. While in certain cases this might be of smaller importance, *e.g.,* when analyzing the singularities of J_G [18], in others, neglecting to consider the computational burden might lead to discarding the use of perfectly viable approaches (due to high computational demand). For example, often in the definition of N appears a generalized inverse of a given matrix. As a result, many practitioners and researchers assume that forming such generalized inverse explicitly is a natural step towards the solution of the problem, while this is simply a step towards the implementation of the specific formulation (refer to Dantzig's quote). Further issues related to the choice of N, its implications and related numerical analysis are addressed in [4].

IV. Numerical example

Here we present an example including some typical tasks for the system (3). A planar manipulator with four revolute joints mounted on a flexible base is considered. Each of the five links is 1 meter long with mass 1 kg and an identity inertia matrix. The initial offset of the base from its equilibrium posture (the origin) is $\Delta x_b = (0.01, 0.01, 0, 0, 0, 0)$ and vibrates according to $K_b = 1000E$, $D_b = 0.1E$, where $E = diag(1, 1, 0, 0, 0, 0)$.

The controller is defined in terms of the following hierarchy (with decision variables $(\ddot{q}, \tau)$)

$$(\tau^b \leq \tau \leq \tau^u) \succ (H\ddot{q} = S\tau - h) \succ (\dot{\theta}^b \leq \dot{\theta} \leq \dot{\theta}^u) \succ$$
$$(\theta^b \leq \theta \leq \theta^u) \succ (J_e\ddot{q} + \dot{J}_e\dot{q} = -D_e J_e \dot{q}) \succ$$
$$(\dot{\omega}_z = -D_\omega w_z - K_\omega a_z) \succ (\dot{v}_{x,y} = -D_v v_{x,y}) \succ (\ddot{\theta} = 0),$$

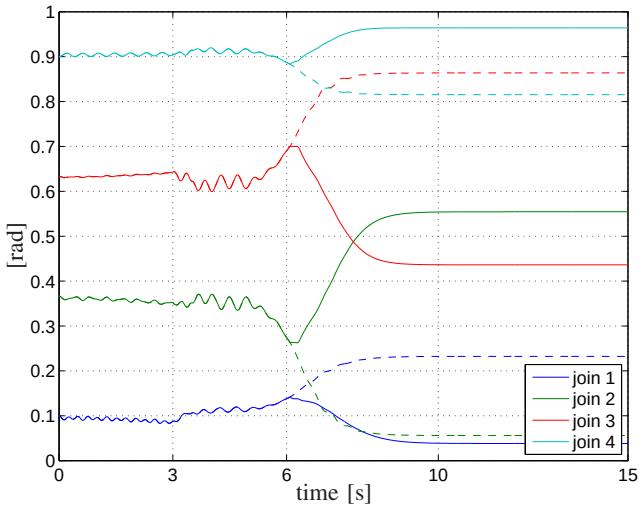

Fig. 1. Joint angles. Dashed lines represent the profiles when constraints on joint angles have not been imposed.

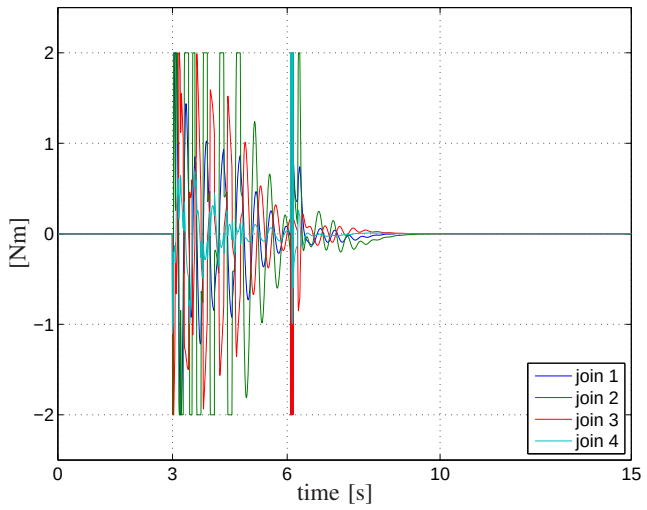

Fig. 2. Joint torques (bounded between $[-2, 2]$).

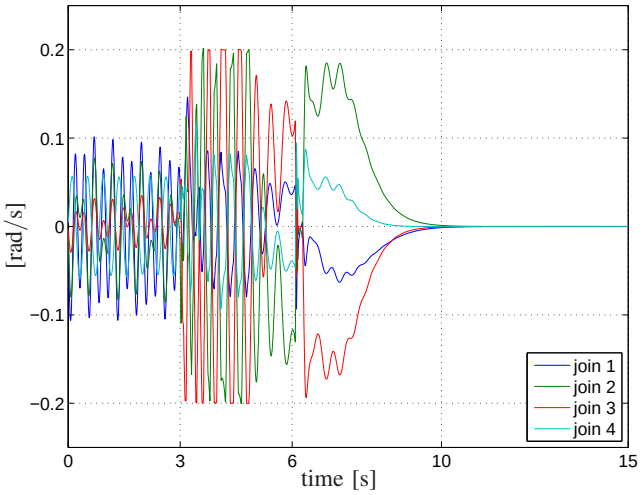

Fig. 3. Joint angular velocities ("bounded" between $[-0.2, 0.2]$).

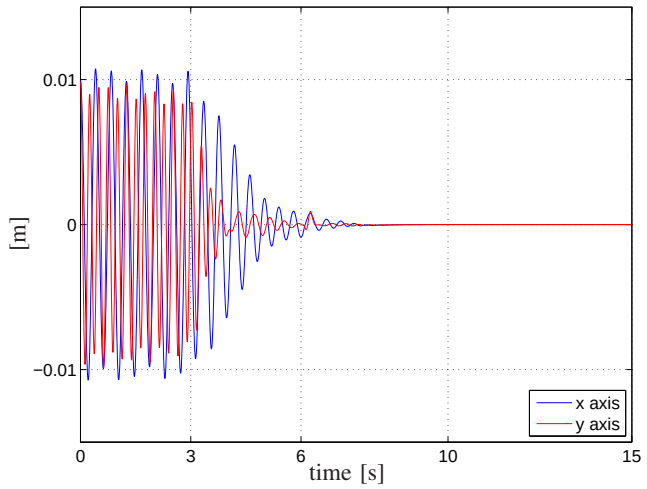

Fig. 4. Base position (x and y axis).

where $\ddot{x}_b = (\dot{v}, \dot{w})$, a_z is the angle (around the z-axis) of the base and $K_\omega = 40$, $D_\omega = 10$, $D_v = 30$, $D_e = 10$, $D_m = 5$ are gains. J_e denotes the Jacobian matrix of the "end-effector". Following the approach in [9], the torque $\tau^\star$ (obtained after resolving the above hierarchy) is modified to $\tau^\star - D_m\dot{\theta}$ before applying it to the system (such joint damping can be achieved in alternative ways).

The first, third and fourth hierarchical levels define tasks involving inequality constraints (however, the only "hard-constraints" are the simple bounds on τ, while the others can be violated depending on the conflicts in the hierarchy). The dynamics of the system is imposed at the second level (and, by design, it does not conflict with the torque limitations). The fifth and seventh task damp the motion of the end-effector and base (linear motion), respectively. The sixth task is dedicated to preserving zero base orientation, while the terminal objective states that obtaining a solution with minimal Euclidean norm of $\ddot{\theta}$ is preferred. The bounds for $\dot{\theta}$ and θ are formed in a standard way [8] with the only difference that the feed-forward joint damping $-D_m\dot{\theta}$ has to be accounted for (the same applies for the bounds on τ).

The results are depicted in figures 1 to 6. During the first three seconds of the simulation the system is uncontrolled. After control is applied, the base vibrations starts to attenuate with a fixed rate (the joint torques are saturated). Just before the 6-th second, the upper bound 0.7 for the third joint angle is reached. The effects of the unmodeled dynamics (d_b) can be seen in Fig. 3 where small violations of joint velocity bounds (not due to conflicts with higher priority tasks) occur.

The simulations were performed on an Intel Core 2 Duo CPU (2.26 GHz, P8400) using `g++ 4.6.3` with `-O3` optimization, the mean computation time is 61μs (*i.e.,* microseconds), while the maximum one is 216μs.

The arrangement of the hierarchy could be modified to obtain alternative prioritizations. For example, one might be interested in assigning higher priority to the base vibration task (in which case, there would be no need to re-derive expressions). If the tasks involving inequality constraints are

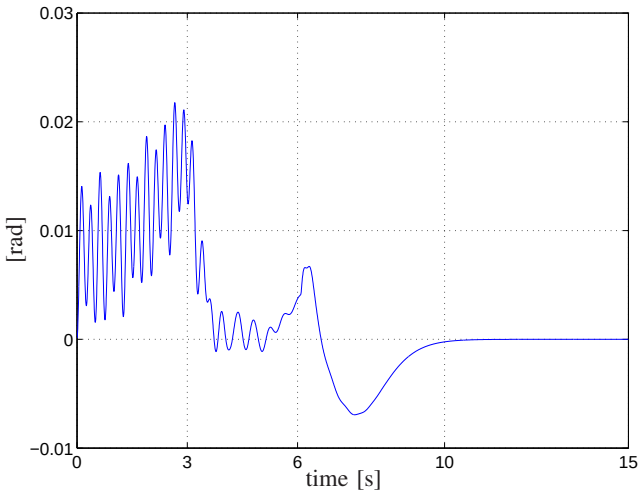

Fig. 5. Base orientation (z axis).

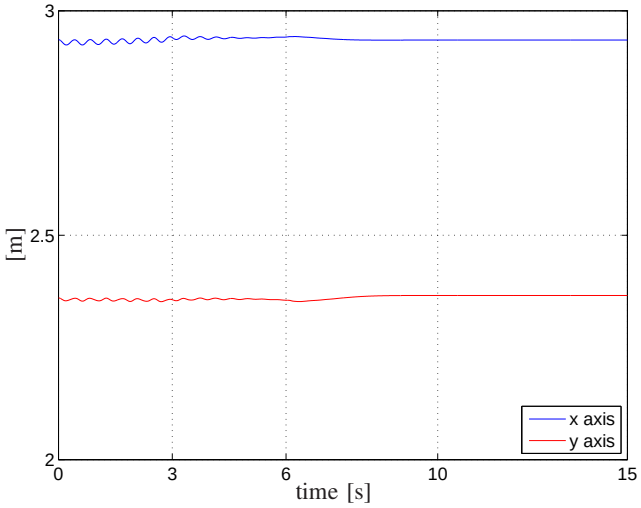

Fig. 6. End-effector position (x and y axis).

removed, the vibrations of the base are suppressed faster, however, this comes at the expense of much higher joint angular rates. The computation time in this case (*i.e.,* 5 objectives involving only equality constraints) is 9μs.

V. Conclusion

In this note we emphasized that the separation of the definition of a problem from considerations related to its solution is important because: (i) it improves the clarity of presentation, and (ii) efficiency of computations. Our main point is that such separation can be achieved through the explicit use of multi-objective formulations.

References

[1] G. B. Dantzig, "Linear programming," *in History of Mathematical Programming: A Collection of Personal Reminiscences,* 1991.

[2] M. Tamiz, and D. Jones, "Practical Goal Programming," *Springer,* 2010.

[3] M. Tamiz, D. Jones, and E. El-Darzi, "A review of Goal Programming and its applications," *Annals of Operations Research,* 58(1), p. 39-53, 1995.

[4] D. Dimitrov, P.-B. Wieber, and A. Sherikov, "Lexicographic least-squares: fast resolution and applications in robotics," to be submitted.

[5] A. Escande, N. Mansard, and P-B. Wieber, "Hierarchical quadratic programming: Fast online humanoid-robot motion generation," *Int. J. Robot. Res.,* 2014.

[6] O. Kanoun, F. Lamiraux, P.-B. Wieber, F. Kanehiro, E. Yoshida, and J.-P. Laumond "Prioritizing linear equality and inequality systems: application to local motion planning for redundant robots," *Proc. of ICRA,* 2009.

[7] M. Tamiz, D. Jones, and C. Romero, "Goal programming for decision making: An overview of the current state-of-the-art," *European Journal of Operational Research,* 111(1), p. 569-581, 1998.

[8] L. Saab, O. E. Ramos, F. Keith, N. Mansard, P. Souères, and J.-Y. Fourquet, "Dynamic Whole-Body Motion Generation Under Rigid Contacts and Other Unilateral Constraints," *IEEE Transactions on Robotics,* 29(2), p. 346-362, 2013.

[9] D. N. Nenchev, K. Yoshida, P. Vichitkulsawat, and M. Uchiyama, "Reaction Null-Space Control of Flexible Structure Mounted Manipulator Systems," *IEEE Transactions on Robotics and Automation,* 15(6), p. 1011-1023, 1999.

[10] S. Abiko, and K. Yoshida, "Adaptive Reaction Control for Space Robotic Applications with Dynamic Model Uncertainty," *Advanced Robotics,* 24(8-9) p. 1099-1126, 2010.

[11] L. Righetti, J. Buchli, M. Mistry, and S. Schaal, "Inverse dynamics control of floating-base robots with external constraints: a unified view," *ICRA,* p. 1085-1090, 2011.

[12] D. Dimitrov, A. Sherikov, and P.-B. Wieber, "A sparse model predictive control formulation for walking motion generation," *Proc. of IROS,* p. 2292-2299, 2011.

[13] O. Khatib, "A unified approach for motion and force control of robot manipulators: The operational space formulation," *IEEE Journal of Robotics and Automation,* 3(1), p. 43-53, 1987.

[14] Y. Umetani, and K. Yoshida, "Resolved motion rate control of space manipulators with generalized Jacobian matrix," *IEEE Transactions on Robotics and Automation,* 5(3), p. 303-314, 1989.

[15] A. Ben-Israel, and T. Greville "Generalized inverses: theory and applications," *Springer,* 2-nd edition, 2003.

[16] D. N. Nenchev, Y. Umetani, and K. Yoshida, "Analysis of a redundant free-flying spacecraft/manipulator system," *IEEE Transactions on Robotics and Automation,* 8(1), p. 1-6, 1992.

[17] O. Khatib, L. Sentis, J. Park, and J. Warren, "Whole-body dynamic behavior and control of human-like robots," *International Journal of Humanoid Robotics,* 1(1), p. 29-43, 2004.

[18] E. Papadopoulos, and S. Dubowsky, "Dynamic Singularities in Free-Floating Space Manipulators," *ASME J. Dyn.Syst., Meas., Contr.,* 115(1), p. 44-52, 1993.

Dimitar Dimitrov is a research scientist at the INRIA research center near Grenoble. He obtained a Ph.D. degree in 2006 from Tohoku University (Japan) and a M.Sc. diploma in 1999 from the Technical University-Sofia (Bulgaria). He has been a lecturer at Örebro University (Sweden). His current research interests include optimal control and estimation as well as convex optimization and its applications.

Pierre-Brice Wieber is Chargé de Recherche at INRIA Grenoble Rhône-Alpes on Optimization methods for robot control. His primary interest is in humanoid robot control. He holds a degree from Ecole Polytechnique and a doctorate from Ecole des Mines de Paris and has been a JSPS fellow in 2005 and a Marie Curie fellow in 2008-2010 at CNRS/AIST JRL in Tsukuba.

Adrien Escande received the MS degree in 2005 from École des Mines de Paris, France and the Ph.D. degree in 2008 in robotics from Université d'Évry Val-d'Essonne, France after spending three years in the CNRS-AIST Joint Robotics Laboratory (JRL), UMI3218/CRT, Tsukuba, Japan. He then worked as a research scientist in CEA-LIST at Fontenay-aux-Roses, France, until the end of 2012 and is now back at JRL. His current research interests include whole-body planning and control for humanoid robots as well as mathematical optimization for robotics.