Multi-Character Physical and Behavioral Interactions Controller

Joris Vaillant, Karim Bouyarmane, and Abderrahmane Kheddar, *Senior Member, IEEE*

Abstract—We extend the quadratic program (QP)-based task-space character control approach — initially intended for individual character animation — to multiple characters interacting among each other or with mobile/articulated elements of the environment. The interactions between the characters can be either physical interactions, such as contacts that can be established or broken at will between them and for which the forces are subjected to Newton's third law, or behavioral interactions, such as collision avoidance and cooperation that naturally emerge to achieve collaborative tasks from high-level specifications. We take a systematic approach integrating all the equations of motions of the characters, objects, and articulated environment parts in a single QP formulation in order to embrace and solve the most general instance of the problem, where independent individual character controllers would fail to account for the inherent coupling of their respective motions through those physical and behavioral interactions. Various types of motions/behaviors are controlled with only the one single formulation that we propose, and some examples of the original motions the framework allows are presented in the accompanying video.

Index Terms—I.3 Computer Graphics, I.3.7 Three-Dimensional Graphics and Realism, I.3.7.a Animation; I.6 Simulation, Modeling, and Visualization, I.6.8 Types of Simulation, I.6.8.a Animation

1 Introduction

Character animation through physics simulation aims at generating interactive and physically plausible low-level character motions from high-level task objectives. Generally, the controller takes care of figuring out the necessary character's joint torques to realize desired tasks and feeds them to the simulator, that will in turn solve the forward dynamics, collision detection, and contact force problem with the given torques to produce the final motion in real-time.

Our approach builds on the well-studied QP-based method (see Section 2 for a brief review of previous studies). The character locomotion controller is formulated as follows:

$$\begin{aligned} &\min_{\substack{\text{accelerations}\\ \text{joint torques}\\ \text{contact forces}}} \sum \text{quadratic objectives}\\ &\text{subject to} \begin{cases} \text{equation of motion}\\ \text{contact no-slip}\\ \text{friction cone limits}\\ \text{joint torque limits} \end{cases}. \end{aligned} \tag{1}$$

The QP (1) is solved at every simulation time-step, the state of the character (positions and velocities) is updated after a given simulator applies the joint torques resulting from the optimization, and the QP (1) is executed for the next time-step in a new iteration. However, since there might exist multiple solutions to the given simulator's contact force problem, we chose a solution that is independent of the external simulator by integrating directly the accelerations resulting from the optimization to generate the motion. Doing so also allows us to use larger time-steps while preserving simulation stability. Nonetheless, bypassing the external simulator in this fashion does not hinder the targeted physics realism since the QP (1) acts itself as a physics simulator (simulating the dynamics equations of motion) provided that: (i) all the contacts are established, maintained, and released at controlled times, and (ii) unwanted collisions are avoided. Both (i) and (ii) are characteristics of our work.

Previous work has formulated the QP (1) for single character animation problems exclusively. Our contribution is to extend it to systems made of arbitrary numbers of interacting characters and objects, ending up with the QP formulation (2). The rationale behind our idea, instead of simply using and composing independent individual QP character controllers, is to allow coherent *interactions* between the characters and objects in the scene. The motions of the characters are indeed *coupled* through the interactions between them. More specifically, we identified two main categories of interactions:

First the *physical interactions* that occur whenever characters are in physical contact with each other. They generate contact forces in action/reaction pairs according to Newton's third law. In our extension of the QP, we propose an ordering scheme of the components of the systems and their respective forces so as to keep one and only one representation of each action/reaction pair for a minimal set of optimization variables.

The second category of interactions that *implicitly* create a coupling between the motions is what we called *behavioral interactions*: these are the collision avoidance constraints

• *The authors are with CNRS-University of Montpellier LIRMM Interactive Digital Humans group, 161 rue Ada, 34095 Montpellier, France. A. Kheddar is also with CNRS-AIST JRL UMI3218/RL, 1-1-1 Umezono, Tsukuba 305-8568 , Japan*

Manuscript received May 20, 2015; revised December 14, 2015 and March 9, 2016.

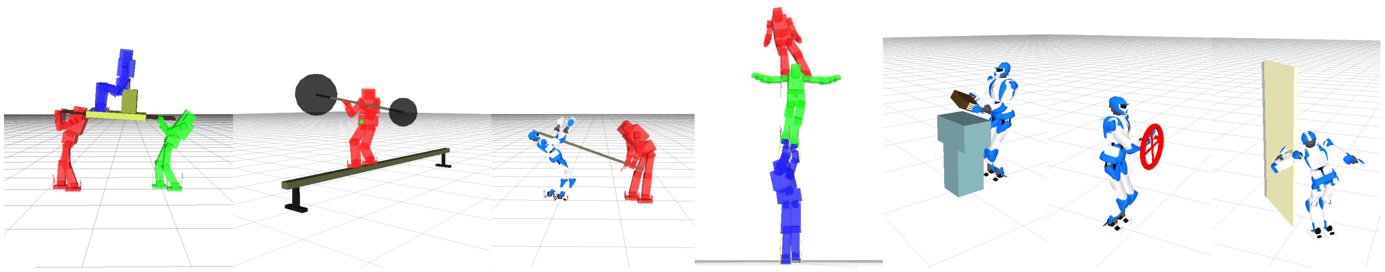


Fig. 1. A selection of example animation scenarios that can be modeled in our framework, screenshots from the accompanying video.

and the collaborative tasks. Collision-avoidance constraints impose for the characters to be "aware" of the presence of each other in their close vicinity and to "predict" each other's motion in order to avoid collisions. The centralized QP accounts for such awareness and inter-predictability, since all the motions are computed together at once by one central controller. Collaboration is also readily encoded in the extension of the QP, by simply writing tasks that are automatically dependent on the motions of the characters involved in the cooperation action. The centralized QP will make the actions *coordinated* in an optimal synergy for performing the task.

Accounting for all these interactions that result in the coupling of the motions, we propose the following compact and easy-to-implement integrated multi-character QP controller:

$$\min_{\substack{\text{(a.1) multi-character accelerations}\\ \text{(a.2) character joint torques}\\ \text{(a.3) minimal set of contact forces}}} \sum \text{quadratic objectives}$$

$$\text{subject to} \begin{cases} \text{(a.4) system of equations of motion} \\ \quad \text{– coupled through Newton's third law} \\ \text{contact no-slip (with the fixed environment)} \\ \text{(a.5) contact no-slip (multi-character interaction)} \\ \text{friction cone limits} \\ \text{joint torque limits} \\ \text{(b.1) joint position and velocity limits} \\ \text{(b.2) collision avoidance} \end{cases} \tag{2}$$

In the QP (2) our technical contributions are explicitly enumerated. The components enumerated (a.1) to (a.5) are contributions pertaining to the *formulation* of the problem as a multi-character system. The components enumerated (b.1) and (b.2) are independent of the multi-character nature of the problem and can as well be incorporated into existing single character controllers. We empirically demonstrate that the centralized brute-force approach consisting in solving one large integrated QP is computationally tractable, by having implemented and executed our framework on a standard laptop computer and generated our motions in real-time or close to real-time.

The presentation of the method is structured in Section 3 as follows: we formulate the multi-character problem (components (a.1) to (a.5) of (2)) with the system of equations of motions in Section 3.1 and the contact no-slip constraint in Section 3.2. Then, we detail the collision-avoidance constraint in Section 3.3 and the joint position and velocity limits among others in Section 3.4. In order to make the paper self-contained, we recall the rest of the components of the QP that we borrow as such from the literature without particular alteration in our method. Those are the friction cone and torque limits in Section 3.4 and the formulation of the quadratic objectives/tasks in Section 3.5. The final form of problem (2) is finally formulated as Equation (35) in Section 3.6. The rest of the paper presents the results in the form of a description of the accompanying video in Section 4, and a discussion and conclusion in Section 5.

2 Literature Review and Related Work

Early work on character physics simulation achieved impressive results using action-specific controllers [1], [2], [3]. In these seminal works a controller is designed for a given human skill (e.g. running, diving, pole-vaulting, biking) and per-joint PD servos track the designed motion in physics simulation. This approach requires the skillful design of a new controller for every new action, and later works would apply the joint-space approach to broader or more parametrizable classes of actions [4], [5], [6]. Task-space approaches have been proposed as an alternative, adapting work done in robotics [7], [8], [9], [10], see e.g. the work by Abe and Popoivić [11].

The task-space formulation is either based on strict lexicographic prioritization — using null-space projectors; or on weighted hierarchy — combining all the tasks in a single QP; or on a mix of both. Our work is mostly inspired by previously proposed QP-based motion controllers [12], [13], [14].

Abe *et al.* [12] initially proposed a framework for achieving standing balance control of physically simulated characters in a given contact configuration with the environment, which allows to either target a static reference posture or to track motion capture data performed from a fixed stance. Based on a similar QP formulation, but with a hybrid lexicographic-weighted policy for the objectives/constraints, De Lasa *et al.* [13] proposed a more general-purpose controller for the locomotion of various biped characters. Momentum objective as proposed by Macchietto *et al.* [15] and later used by Al Borno *et al.* [16] was included and shown to yield "natural-looking" behaviors for walking or jumping. The QP controller was in this work coupled with a finite-state machine (FSM) that builds the appropriate instance of the prioritized-QP problem to control a given phase of the locomotion. In our present work, we opted for the weighted approach and similarly used FSM decompositions of the various phases of our motions. That controller [13] was also used as a low-level controller that realizes a higher-level plan [17], showcasing robust bipedal walking and running simulations.

Jain *et al.* [14] proposed a slightly different formulation of the problem, which is directly in the joint position space. They demonstrate a wide variety of character balancing behaviors, with one of them involving a cooperation between two characters. The capabilities of our framework seem similar in that regard. It is however unclear and not detailed in the paper how the cooperation is effectively achieved and if it was specifically designed for the example motion. See Table 2 for further comparison.

Trajectory optimization approaches, on the other hand, allow to synthesize broader ranges of parametrizable motions at the expense of little or no interactivity and high computational costs which often makes them unadapted to real-time applications, but still achieving a high degree of realism for original highly dynamic motions [18], [19], [20]. These works, however, are mainly about the locomotion of one character in the world and do not integrate along with it a manipulation behavior component [11], [21], [22], [23], [24], a cooperative behavior component [25], [26], [27], [28], a quadruped walking behavior component [29], a dexterous hand component [30], to cite a few examples.

Recently Mordatch *et al.* [31], [32] introduced contact-invariant optimization (CIO) motivated by our same expressed desire of proposing a framework capable of embracing a wide variety of classes of character motions at once. They succeeded in demonstrating that their approach enables to yield (i) locomotion behaviors beyond periodic biped walking, e.g. climbing, crawling, standing-up motions, etc. (ii) various hand dexterous manipulations, and (iii) object-manipulation and multiple character cooperating. Though this was done in an offline trajectory optimization approach which prevents real-time interactive control possibilities and with simplified physics, our present work is inspired by the same philosophy of generality in the targeted character motion instances. We had previously followed such a methodology proposing an inverse kinematics solver [33] and a contact planner [34]. We propose now a real-time controller based on the same philosophy.

As for existing controllers allowing coherent interactions between characters, Ho *et al.* [35] proposed a retargeting method for motions involving characters in *close interactions* (contacts, collision-avoidance). They adapt the close interactions to different characters by using a representation based on spatial relations called *interaction mesh*. Our proposed approach is different in that it does not need a reference motion input.

To sum up this section, our work can be seen as reconciling different aspects of the works reviewed here in what thus constitutes a novel approach. Namely, we fuse the aspects of real-time interactive physics simulation [12], [13] with a general motion planning philosophy [36] [31], or, in other words, we target the same level of generality attained in the latter works [36] [31] using the more flexible, interactive-control-enabling approach of the former works [12], [13].

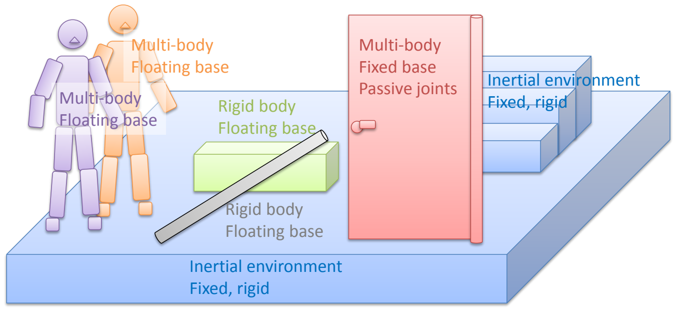


Fig. 2. Various instances of subsystem types that combine together to create the full system controlled as a whole in the animation scene. Each individual subsystem is represented in a different color.

3 Method

3.1 Equation of Motion (EOM)

Our method considers all the interacting characters, objects, and the environment in the scene as one system. Let us denote n the number of all identified independent subsystems in the scene. One such independent subsystem can be a character, a rigid object (e.g. manipulated box), an articulated part of the environment (e.g. a door, a valve, etc.), see Fig. 2. We index them with the variable i in $\{1,\ldots,n\}$, and we use the index $i=0$ for the rest of the rigid inertial environment (ground, walls, stairs, etc.). Every subsystem $i \in \{1,\ldots,n\}$ can be modeled as either a fixed-base or a floating-base articulated kinematic tree with configuration vector $q_i \in \mathbb{R}^{\mu_i}$ (which includes the free-floating base position/orientation if any and the joint angles if any), and behave following their respective EOM[1]

$$M_i(q_i)\ddot{q}_i + N_i(q_i,\dot{q}_i) = J_{\text{all},i}(q_i)^T f_{\text{all},i} + S_i\tau_i\,, \tag{3}$$

where $\tau_i \in \mathbb{R}^{a_i}$ is the vector of torques acting on the actuated DOFs of the subsystem ($a_i = 0$ for a manipulated object and for passive articulated part of the environment) and $S_i \in \mathbf{M}(\mu_i, a_i)$ is the selection matrix that maps the dimension of τ_i to that of q_i by extending τ_i with zeros at the indexes of the non-actuated DOFs (which include the free-floating base DOFs if any). The subsystem is supposed to be subjected to the action of a set of ν_i point contact forces $f_{\text{all},i} \in \mathbb{R}^{3\nu_i}$ with respective Jacobians at the corresponding contact points $J_{\text{all},i} \in \mathbf{M}(3\nu_i, \mu_i)$. M_i and N_i are respectively the mass matrix and the term regrouping the non-linear effects and the gravity. Equation (3) reduces to the Newton-Euler EOM for a rigid body subsystem (e.g. a manipulated object).

Each contact force applied at subsystem i is either applied by the inertial environment or by another subsystem j and thus appears, in the latter case, with an opposite sign

1. The notations of the paper are consistent with the conventional identification of vectors as column matrices (and not as row matrices) $\mathbb{R}^r \equiv \mathbf{M}(r,1)$, meaning that $(\lambda_1,\ldots,\lambda_r) \equiv \begin{pmatrix}\lambda_1 & \cdots & \lambda_r\end{pmatrix}^T$ and in particular that $(\lambda_1,\ldots,\lambda_r) \not\equiv \begin{pmatrix}\lambda_1 & \cdots & \lambda_r\end{pmatrix}$. $\mathbf{M}(\alpha,\beta)$ denotes the set of real matrices of α rows and β columns.

in subsystem j's EOM according to Newton's third law. We thus rewrite all Equations (3) in the following forms:

$$M_i(q_i)\ddot{q}_i + N_i(q_i,\dot{q}_i) = \\ J_{0,i}(q_i)^T f_{0,i} + J_{1,i}(q_i)^T f_{1,i} - J_{2,i}(q_i)^T f_{2,i} + S_i\tau_i\,, \quad (4)$$

where $f_{0,i}$ are the contact forces applied by the environment on subsystem i, $f_{1,i}$ are the contact forces applied by subsystems $j \in \{1,\ldots,i-1\}$ on subsystem i, and $f_{2,i}$ are the forces applied by subsystem i on subsystems $j \in \{i+1,\ldots,n\}$.

Let F_0, F_1, F_2 be respectively the stacked vectors of all the forces $f_{0,i}$'s, $f_{1,i}$'s, and $f_{2,i}$'s, i.e.

$$F_k = (f_{k,i})_{1\le i\le n}\,,\ k = 0,1,2\,. \quad (5)$$

Since, $\forall i \in \{1,\ldots,n\}$, all the forces in $f_{2,i}$ appear at some position in some of the $f_{1,j}$ forces, with j in a subset of $\{1,\ldots,n\}$, we can write $f_{2,i} = \phi_i F_1$ where ϕ_i is a selection matrix that selects the adequate elements in F_1 and reorders them into $f_{2,i}$.

Equations (4) thus take the following forms:

$$M_i(q_i)\ddot{q}_i + N_i(q_i,\dot{q}_i) = \\ J_{0,i}(q_i)^T f_{0,i} + J_{1,i}(q_i)^T f_{1,i} - J_{2,i}(q_i)^T \phi_i F_1 + S_i\tau_i\,. \quad (6)$$

The common variable F_1 binds together all the Equations (6). This binding transcribes the coupling of the motions through the physical interactions among the subsystems. By denoting $q = (q_1,\ldots,q_n)$ and $\tau = (\tau_1,\ldots,\tau_n)$ and by stacking together all the elements of Equations (6):

$$M(q) = \operatorname{diag}\left(M_1(q_1),\ldots,M_n(q_n)\right), \quad (7)$$
$$J_k(q) = \operatorname{diag}\left(J_{k,1}(q_1),\ldots,J_{k,n}(q_n)\right)_{k=0,1,2}, \quad (8)$$
$$S = \operatorname{diag}\left(S_1,\ldots,S_n\right), \quad (9)$$
$$\Phi = \begin{pmatrix}\phi_1\\ \vdots\\ \phi_n\end{pmatrix}, \quad (10)$$
$$N(q,\dot{q}) = \begin{pmatrix}N_1(q_1,\dot{q}_1)\\ \vdots\\ N_n(q_n,\dot{q}_n)\end{pmatrix}, \quad (11)$$

we can rewrite Equations (6) as one EOM of the full system that makes up our animation scene:

$$M(q)\ddot{q}+N(q,\dot{q}) = J_0(q)^T F_0 + \Big(J_1(q)^T - J_2(q)^T\Phi\Big)\,F_1 + S\tau\,. \quad (12)$$

Note: Φ defined in (10) is a square permutation matrix[2], thus in particular an orthogonal matrix $\Phi^T\Phi = I$, since it maps the triple position of every internal contact force of the system in the stacked vector F_1 to its Newton's third law counterpart triple position that uniquely exists in the stacked vector F_2. The relation $F_2 = \Phi F_1$ encodes Newton's third law in the whole system and in (12). F_2 does not appear in this equation anymore and thus (F_0, F_1) is the minimal set of force optimization variables we keep in the formulation. See Fig. 3 for a simple case example.

2. More precisely, if 3κ is the size of F_1, where κ is the total number of point forces from our ordering convention stacked into F_1, then Φ is of the form $\Phi = P \otimes I_3$ where P is a permutation matrix of size $\kappa\times\kappa$ and I_3 is the 3×3 identity matrix. The operator $\otimes$ denotes the Kronecker product.

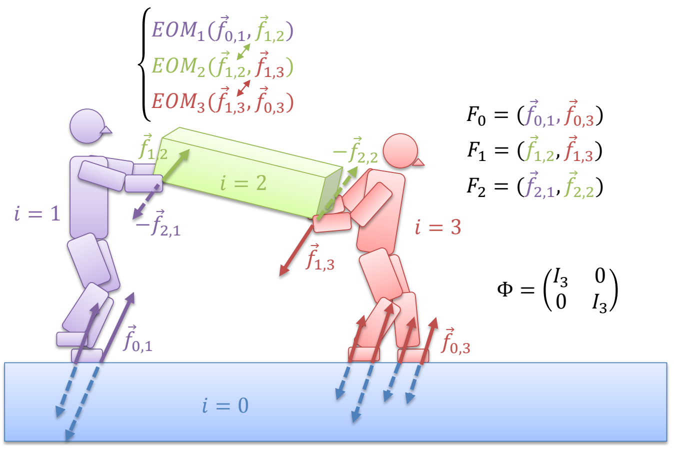

$EOM_1(\vec{f}_{0,1},\vec{f}_{1,2})$
$EOM_2(\vec{f}_{1,2},\vec{f}_{1,3})$
$EOM_3(\vec{f}_{1,3},\vec{f}_{0,3})$
$F_0 = (\vec{f}_{0,1},\vec{f}_{0,3})$
$F_1 = (\vec{f}_{1,2},\vec{f}_{1,3})$
$F_2 = (\vec{f}_{2,1},\vec{f}_{2,2})$
$\vec{f}_{1,2}$
$-\vec{f}_{2,2}$
$i = 1$
$i = 2$
$i = 3$
$-\vec{f}_{2,1}$
$\vec{f}_{1,3}$
$\Phi = \begin{pmatrix}I_3 & 0\\ 0 & I_3\end{pmatrix}$
$\vec{f}_{0,1}$
$\vec{f}_{0,3}$
$i = 0$

Fig. 3. Deriving the entire system's EOM from the individual subsystems respective EOMs. Each subsystem is shown in a different color and the contact forces applied on a subsystem are shown in the same color as the subsystem. Contact forces that are left as optimization variables from our ordering convention and that effectively appear in the QP problem are shown as solid arrows (they make up the variables F_0 and F_1), while dashed arrows represent the corresponding reaction forces. The relation $F_2 = \Phi F_1$ encodes Newton's third law, and the forces $-F_2$, by convention, do not independently appear in the QP. The upper brace EOM shows how the three individual EOMs are coupled through the contact forces $f_{1,2}$ and $f_{1,3}$, i.e. through the QP variable F_1.

3.2 Contact No-Slip

In addition to (12), the consistency of the physical interactions that occur in the scene is ensured by enforcing the following contact no-slip constraints:

$$J_0(q)\,\dot{q} = 0\,, \quad (13)$$
$$J_2(q)\,\dot{q} = \Phi J_1(q)\,\dot{q}\,, \quad (14)$$

Equation (13) is usually written in existing single character QP controllers, encoding the zero-velocity condition of the contact points of the subsystems with the inertial environment. Equation (14) however is exclusive to the multi-character system and encodes the zero-*relative-velocity* condition of all pairs of contact points belonging to pairs of subsystems in contact. The mapping Φ introduced in the previous section allows a very compact encoding of this condition. It expresses that the mapping of the contact forces $F_2 = \Phi F_1$ is conserved for the contact point velocities obtained from the stacked Jacobian matrices through the principle of virtual work. Note that since $\Phi^{-1} = \Phi^T$, Equation (14) is equivalent to

$$\Big(J_1(q)^T - J_2(q)^T\Phi\Big)^T\,\dot{q} = 0\,, \quad (15)$$

and, consequently, F_1 can be interpreted as the Lagrange multiplier associated with this constraint in (12).

Equations (13) and (14) are time-differentiated to obtain constraints on the accelerations compatible with the QP:

$$J_0(q)\,\ddot{q} + \dot{J}_0(q)\,\dot{q} = 0 \quad (16)$$
$$\Big(\Phi J_1(q) - J_2(q)\Big)\ddot{q} + \Big(\Phi\dot{J}_1(q) - \dot{J}_2(q)\Big)\,\dot{q} = 0\,. \quad (17)$$

These formulations are prone to numerical instability since the QP solver cannot ensure a perfect numerical satisfaction of the constraint. It does so only within a certain tolerance threshold. Hence the relative acceleration of the contact links is not perfectly zero, and their relative velocities might diverge in a prolonged contact state. The numerical problem also occurs when the initial velocities are not exactly the same when the contact is established and the constraint activated. Consequently, we replace the formulations (16) and (17) with an empirically more stable formulation as follows.

For every contact between subsystem i and the fixed inertial environment, let us denote $v_{0,i}$ the 6D linear and angular velocity and $\mathcal{J}_{0,i} \in \mathbf{M}(6, \mu_i)$ the corresponding linear and angular Jacobian of the contact link of the subsystem; the no-slip constraint for this contact link is written as:

$$\mathcal{J}_{0,i}\ddot{q}_i + \dot{\mathcal{J}}_{0,i}\dot{q}_i = -\frac{v_{0,i}}{\Delta t}\,, \qquad (18)$$

where Δt is the integration time-step. This differential equation leads to a velocity that converges to zero. The constraints (18) replace the constraint (16).

For a contact between subsystems i and j, let $v_{1,i}$ and ${}^{1,i}v_{2,j}$ denote respectively the 6D velocity of the contact link of subsystem i ("link 1") and the 6D velocity of the contact link of subsystem j ("link 2") transformed in link 1 frame and expressed at the same reference point (in the sequel we denote it for brevity only as $v_{2,j}$). Finally let $\mathcal{J}_{1,i}$ and ${}^{1,i}\mathcal{J}_{2,j}$ (for brevity again denoted $\mathcal{J}_{2,j}$) denote the corresponding 6D Jacobians. The new formulation of (17) is obtained by writing the differential equation on the relative velocity:

$$v_{1,i}+\Big(\mathcal{J}_{1,i}\ddot{q}_i+\dot{\mathcal{J}}_{1,i}\dot{q}_i\Big)\Delta t = v_{2,j}+\Big(\mathcal{J}_{2,j}\ddot{q}_j+\dot{\mathcal{J}}_{2,j}\dot{q}_j\Big)\Delta t\,, \qquad (19)$$

However, this latter formulation might also lead to numerical inaccuracies since the transformation from the frame of link 2 to that of link 1 is not constant over time (small perturbations leading to loss of contact). It results in relative velocites We propose an empirically more stable version of (19) (tested in the examples of Section 4 for integration time-steps Δt ranging between 5ms and up to 33ms) by accounting for the error in the transformation between the frames of the two links in the following law:

$$v_{1,i} + \Big(\mathcal{J}_{1,i}\ddot{q}_i + \dot{\mathcal{J}}_{1,i}\dot{q}_i\Big)\Delta t - v_{2,j} - \Big(\mathcal{J}_{2,j}\ddot{q}_j + \dot{\mathcal{J}}_{2,j}\dot{q}_j\Big)\Delta t = -\frac{\mathrm{Err}\left({}^{2,j}X^{\mathrm{ref}}_{1,i}\,{}^{1,i}X_{2,j}\right)}{\Delta t}\,, \qquad (20)$$

where $\mathrm{Err}\left({}^{2,j}X^{\mathrm{ref}}_{1,i}\,{}^{1,i}X_{2,j}\right)$ expresses the 6D error between:

1) ${}^{1,i}X_{2,j}$: the current transformation between the link 2 frame and the link 1 frame (at the current time-step),
2) ${}^{1,i}X^{\mathrm{ref}}_{2,j}$: the initial "reference" transformation between the link 2 frame and the link 1 frame (at the initial time-step in which the contact was established).[3]

3. If ${}^{B}X_{A}$ denotes a 6D transformation matrix from frame A to frame B, ${}^{B}X_{A} = \begin{pmatrix} R & 0 \\ -(Rr)\times & R \end{pmatrix}$, where R and r denote respectively the rotation matrix and translation vector from frame A to frame B, $\mathrm{Err}({}^{B}X_{A})$ is defined as the 6D vector $\mathrm{Err}({}^{B}X_{A}) = \begin{pmatrix} \ln R \\ r \end{pmatrix}$, where $\ln R$ is defined as the the angular velocity vector that yields R over a unit time, i.e $R = \exp(wt)$ with $t = 1$.

3.3 Collision Avoidance

Collision avoidance is one of the behavioral interactions between the subsystems of the scene that create an implicit coupling of their motions. The collision-avoidance constraint is however not exclusive to the multi-character problem and can also be used in single character applications for avoiding static or moving obstacles.

To the best of our knowledge, no previous QP-based approach proposed in the literature has dealt with this kind of constraint at such low level, and detailing it here would constitutes an original addition to state-of-the-art QP-based controllers. Existing collision-avoidance approaches are usually encoded at higher levels with predefined, known, obstacle trajectories or with predicted obstacle motions, e.g. [14], [28]. Our approach does not need any pre-computation or prediction of trajectories and acts in a reactive fashion to any currently occurring motions. Recent work, that also includes collision-avoidance constraints in a QP-based control, can be found in [37]. Other related work incorporates a reactive collision avoidance scheme similar to the one we use here in an inverse-kinematics-based motion reconstruction from motion capture data [38].

A collision-avoidance constraint in our framework can be written between any pairs of bodies in the scene, whatever subsystem they belong to (including the inertial environment). The distance computation method we use is an implementation of the Gilbert, Johnson and Keerthi (GJK) algorithm, as detailed in [39]. At every configuration of two *convex* bodies, The GJK algorithm computes the two *witness points* belonging to their respective surfaces. They are defined as the pair of points such that the distance between them is equal to the distance between the two convex bodies. These witness points move along the surfaces of the two bodies as their configuration change over time. We apply the positive distance constraint on these two moving points, thus guaranteeing the satisfaction of the collision-avoidance constraint between the two considered convex bodies they belong to, as they move over time. Non-convex bodies are decomposed into convex components (or approximation thereof if no such decomposition exists) and the constraint is applied on the convex components.

The formulation of the collision-avoidance constraint relies on velocity damping initially proposed in robotics applications [40], [41]. Let us consider two bodies of the scene belonging respectively to subsystems i and j for which we would like to write the collision-avoidance constraint. The distance d between the two bodies is

$$d = \sigma||p_{1,i} - p_{2,j}||\,, \qquad (21)$$

where $p_{1,i}$ and $p_{2,j}$ are the two witness points, and $\sigma = +1$ if there is no collision and $\sigma = -1$ if there is collision, in which case d is an inter-penetration distance. A basic velocity damper behaviour is obtained through the following inequality

$$\dot{d} \geq -\xi\frac{d - \delta_s}{\delta_i - \delta_s}\,, \qquad (22)$$

where ξ, δ_s, and δ_i are fixed parameters representing respectively the damping factor, the security distance, and the

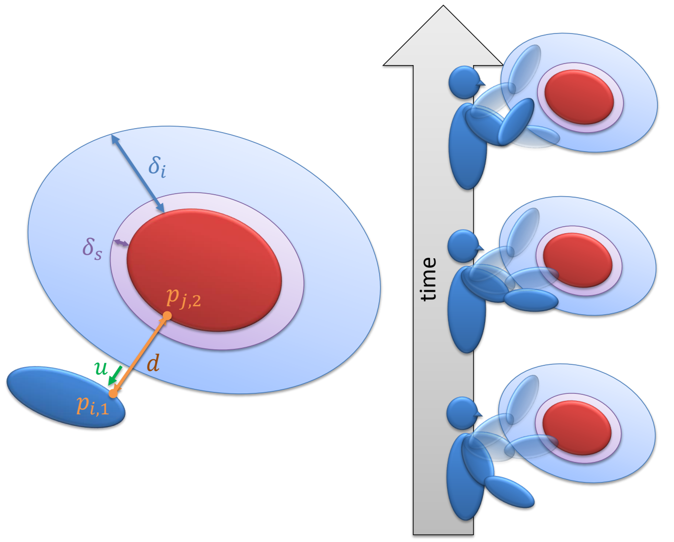


Fig. 4. Illustration of the collision-avoidance method. In the left figure, the blue and red bodies must avoid colliding. The thin purple layer wrapping the red body represents the security "forbidden" zone from which we consider that collision has occurred (this security distance can just be reduced to zero), the light blue zone shows the influence distance from entering which the constraint is activated and starts influencing the motions of the bodies. The right figure is decomposed into three sequential time frames. The bottom time frame shows the initial position and expected motion of the arm performing a reaching task. This expected motion will collide with the red obstacle. Between the bottom and the middle time frames, the motion of the arm occurs outside of the influence zone, so the motion is not affected and still follows the expected path. The middle time frame shows the arm position at the instant when it enters the influence zone, and the expected path that it was following when entering that zone (the same as the previous time frame). At that instant when the arm enters the influence zone the constraint is activated. The top frame shows the actual motion of the arm that was deviated from the expected path after entering the influence zone, this new deviated path avoids the obstacle by avoiding the purple security layer around it.

influence distance below which the constraint is activated. A QP-compatible version can be written as

$$\ddot{d} \geq \frac{1}{\Delta t}\left(-\xi\frac{d-\delta_s}{\delta_i-\delta_s}-\dot{d}\right). \tag{23}$$

Denoting $u = (p_{1,i} - p_{2,j})/d$ the unit vector between $p_{1,i}$ and $p_{2,i}$, the derivative $\dot{d}$ is obtained as $\dot{d} = (\dot{p}_{1,i} - \dot{p}_{2,j})^T u$. Finally denoting $J^{\text{lin}}_{1,i}$ and $J^{\text{lin}}_{2,j}$ respectively the linear (translational) Jacobians of subsystems i and j at $p_{1,i}$ and $p_{2,j}$, equation (23) takes the following final form that we add as a constraint to the QP

$$u^T\left(J^{\text{lin}}_{1,i}\ddot{q}_i + \dot{J}^{\text{lin}}_{1,i}\dot{q}_i - J^{\text{lin}}_{2,j}\ddot{q}_j - \dot{J}^{\text{lin}}_{2,j}\dot{q}_j\right) + \dot{u}^T(\dot{p}_{1,i} - \dot{p}_{2,j}) \geq \frac{1}{\Delta t}\left(-\xi\frac{d-\delta_s}{\delta_i-\delta_s}-\dot{d}\right). \tag{24}$$

See Fig.4 for a schematic illustration of the behavior.

One limitation of this collision-avoidance approach is the possibility of the bodies to get stuck in local minima. The solution we retained for avoiding them is by letting the user specify, in the FSM described below, intermediate way-points to guide the motion away from such local minimum if it occurs. One single way-point is in general sufficient and the user does not need to specify any explicit trajectory.

3.4 Other Constraints of the Motion

The next set of constraints is the unilateral contacts and friction cone constraints. A QP-compatible formulation of those is obtained by linearizing the friction cones into friction pyramids such that the forces F_0 and F_1 can be respectively written as $F_0 = K_0\Lambda_0$ and $F_1 = K_1\Lambda_1$, where K_0 and K_1 are the matrices of unit vectors generators of the pyramid edges, and Λ_0 and Λ_1 the coefficients along these generators. The unilateral and friction cones constraints become:

$$\Lambda_0 \geq 0 \text{ and } \Lambda_1 \geq 0\,. \tag{25}$$

We modeled two types of contacts with this contact framework: planar contacts and grasp contacts. Planar contact areas (e.g. foot sole) are modeled with 4 contact points and contact normals on a plane approximation of the contact area. Grasp contact are modeled with 4 contact points and normal vectors distributed along a cylindrical approximation of the hand palm and fingers contact area. We used 4-edge pyramid approximations of the friction cones, thus each contact of either type contributes with 16 variables in one of the two vectors Λ_0 and Λ_1. See Fig. 5.

The last set of constraints are the position, velocity, and torque limits constraints

$$q_{\text{min}} \leq q \leq q_{\text{max}}\,, \tag{26}$$
$$\dot{q}_{\text{min}} \leq \dot{q} \leq \dot{q}_{\text{max}}\,, \tag{27}$$
$$\tau_{\text{min}} \leq \tau \leq \tau_{\text{max}}\,. \tag{28}$$

The joint limit constraint (26) is necessary for the geometrical consistency of the scene, while the torque limit constraint (28) can be enforced for its physical consistency if desired. The velocity limit constraint (27) is more inherited from robotics applications although not always relevant in a computer animation context. We include it in this

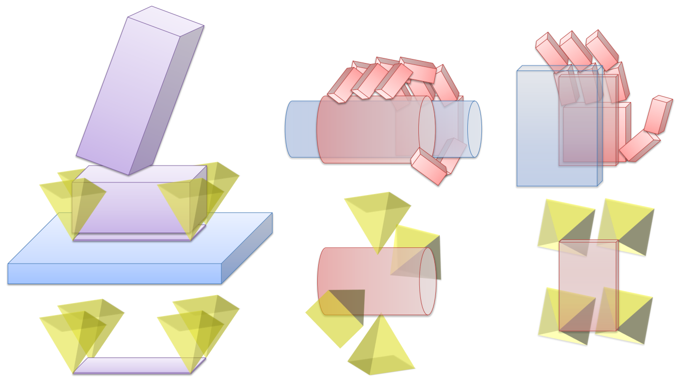


Fig. 5. Example of contact surfaces and contact point modeling. Feet contacts (left) are modeled as 4-point planar contacts. Hand grasp contact surfaces (middle) are modeled as cylindrical surfaces with 4 contact points around the surface (note that the pyramids are oriented to the outside for the hand-attached contact cylinder but are oriented to the inside of the cylinder for the object-attached cylindrical grasp surface, the latter are omitted in the figure for clarity). The hands can also be used for planar non-grasp surface (right) similarly to the feet or other planar contact areas such as the buttocks for sitting.

description for completeness. QP-compatible formulations of inequalities (26) and (27) can be written respectively as

$$\frac{q_{\min} - q - \dot{q}\Delta t}{\frac{1}{2}\Delta t^2} \leq \ddot{q} \leq \frac{q_{\max} - q - \dot{q}\Delta t}{\frac{1}{2}\Delta t^2} \tag{29}$$

$$\frac{\dot{q}_{\min} - \dot{q}}{\Delta t} \leq \ddot{q} \leq \frac{\dot{q}_{\max} - \dot{q}}{\Delta t} \,. \tag{30}$$

However, the formulation (29) leads to strong decelerations when the joint comes close to its limit and to discontinuities in the torque output by the QP. To solve this we introduce a velocity damper similar to the one used for collision avoidance in the previous section. Let $d_{\min} = q - q_{\min}$ and $d_{\max} = q_{\max} - q$, we replace (29) with

$$\frac{-\xi\frac{d_{\min}-\delta_s}{\delta_i-\delta_s} - \dot{q}}{\Delta t} \leq \ddot{q} \leq \frac{\xi\frac{d_{\max}-\delta_s}{\delta_i-\delta_s} - \dot{q}}{\Delta t} \,, \tag{31}$$

which are added as constraints when $d_{\min} \leq \delta_i$ and $d_{\max} \leq \delta_i$ respectively.

3.5 Tasks/Objectives

The tasks/objectives of the motion are expressed in terms of *features* of the system, a feature being any function $x(q)$ such as the position of the hand of the character, the trajectory of the foot of the character, the configuration of a door (opening angle), the position/orientation of a floating object, etc. A feature is associated with a Jacobian J_x such that $\dot{x} = J_x\dot{q}$ and $\ddot{x} = J_x\ddot{q} + \dot{J}_x\dot{q}$. For a number $\mathcal{M}$ of simultaneous objectives $(x_1, \ldots, x_{\mathcal{M}})$ in a given phase of the animation, the quadratic cost function to minimize in the QP is defined as

$$c_{q,\dot{q}}(\ddot{q}) = \sum_{m=1}^{\mathcal{M}} w_m ||\ddot{x}_m - \ddot{x}_m^d||^2 \,, \tag{32}$$

where w_m are the relative weights of the objectives that are tuned by the user depending on which objective they would like to favor and depending on the observed behavior resulting from that choice (e.g. falling down would suggest increasing a COM objective weight); $\ddot{x}^d$ is a desired behavior that we borrow from the previous work [12] as

$$\ddot{x}^d = -k(x - x^{\text{ref}}) - 2\sqrt{k}(\dot{x} - \dot{x}^{\text{ref}}) + \ddot{x}^{\text{ref}} \,, \tag{33}$$

where x^{ref} is a reference trajectory that the user designs and would like to track, or a fixed value around which they would like to regulate the feature. k is a defined stiffness gain for the task. In the example animations of this paper, it was sufficient to use only piecewise constant profiles of x^{ref}, i.e $\dot{x}^{\text{ref}} = 0$ and $\ddot{x}^{\text{ref}} = 0$ and thus

$$\ddot{x}^d = -k(x - x^{\text{ref}}) - 2\sqrt{k}\dot{x} \,, \tag{34}$$

both for regulating the feature x around a constant value x^{reg} ($x^{\text{ref}} = x^{\text{reg}}$) and for steering the feature x to a distant target value x^{tgt} ($x^{\text{ref}} = x^{\text{tgt}}$) (though we also implemented the target objective proposed in [13], we did not use it in our applications). In the coming Figures containing example objective descriptions (Figs. 6, 11, 12 later in the paper), we textually refer to both these kinds of tasks (regulating and targeting) as "Set x to x^{ref}" since (34) is used for both.

A typical phase of the animations we produced in the examples required the design of the following tasks:

• a reference rest pose for the whole configuration of the system. It can be rapidly sketched by the user giving a gross approximation of the expected postures during the motion, or obtained by means of inverse kinematics if the user wants a more refined pose, e.g. [42]. This task is typically low-weight task and used as a "background" task for regulating the values of the DOFs that are not used for the other tasks. In each of the demonstrated animation examples of this paper, only one rest pose was used both to initialize the system and for the whole motion.

• the reference/target COM of a character. We either regulate the COM around its stable static-pose value or its projection at the center of the support polygon, or we steer its projection away from a given contact area and into the reduced support polygon to remove that contact;

• the target 6D position and orientation of the swing foot of the characters in locomotion phases, including both the landing position of the foot and mid-step height,

• a target 3D position of the hand of a character for reaching tasks for example;

• a target 3D or 6D position of a rigid manipulated object, a target configuration/joint angle/position/orientation of an articulated part of the environment with which a character interacts;

• a reference/target COM of a group of characters/objects in contact with each other and moving together in physical interaction.

While all these tasks and features are classically used in existing QP controllers, the main novelty of our present work lies in the latter two tasks which are a specificity of our multi-character multi-object approach. Collaborative behaviors naturally emerge from all the characters present in the scene when controlling a feature that involves these characters. A single feature such as the position of a collaboratively manipulated heavy or bulky object will autonomously drive the behavior of all the characters that are in direct or indirect contact with this object or that are grasping a part of it. The same remark goes for the single COM of the system made of these characters and object. As previously introduced, these types of interactions and synergies among characters are part of the behavioral interactions that implicitly concur in the coupling of their motions.

3.6 Final QP and Finite-State Machine (FSM) Controller

The QP problem that is solved at every time-step of the simulation is formulated as follows

$$\begin{aligned} &\min_{\ddot{q},\tau,\Lambda_0,\Lambda_1} c_{q,\dot{q}}(\ddot{q}) \\ &\text{subject to (12) (18) (20) (24) (25) (28) (30) (31)}\,. \end{aligned} \tag{35}$$

At every control time-step Δt, the problem (35) is solved and the resulting $\ddot{q}$ is integrated to update the state $(q, \dot{q})$ of the system for the next time-step iteration.

An alternative approach would have been to sequentially solve, within each iteration time-step, n "small" QPs, one for each subsystem, rather than our integrated QP for the whole system. The contact forces and the positions of the contact points solved for QPs number 1 to i fed as inputs to QP number $i+1$, and this iteratively for $i \in \{1, \ldots, n-1\}$. This approach would however prove sub-optimal and could lead to unfeasible problems whereas the one-QP approach

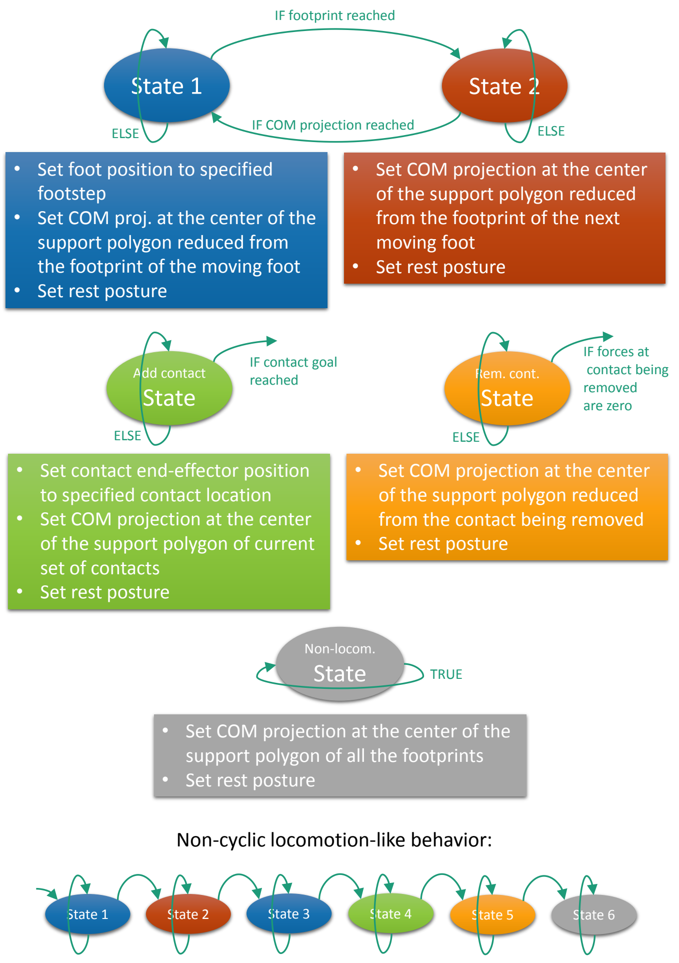

IF footprint reached
State 1
State 2
IF COM projection reached
ELSE
ELSE
Set foot position to specified footstep
Set COM proj. at the center of the support polygon reduced from the footprint of the moving foot
Set rest posture
Set COM projection at the center of the support polygon reduced from the footprint of the next moving foot
Set rest posture
Add contact State
IF contact goal reached
Rem. cont. State
IF forces at contact being removed are zero
ELSE
ELSE
Set contact end-effector position to specified contact location
Set COM projection at the center of the support polygon of current set of contacts
Set rest posture
Set COM projection at the center of the support polygon reduced from the contact being removed
Set rest posture
Non-locom. State
TRUE
Set COM projection at the center of the support polygon of all the footprints
Set rest posture
Non-cyclic locomotion-like behavior:
State 1
State 2
State 3
State 4
State 5
State 6

Fig. 6. FSM building-blocks with minimal sets of objectives. The two top states (blue and red) are the two states for cyclic locmotion alternating left foot and right foot steps. The two middle states (green and orange) are for general contact and grasp events (adding/removing). The bottom state (gray) is for non-locomotion phases (e.g. interactive reaching). Additional objectives can be added to these minimal sets of objectives depending on the particular scenario.

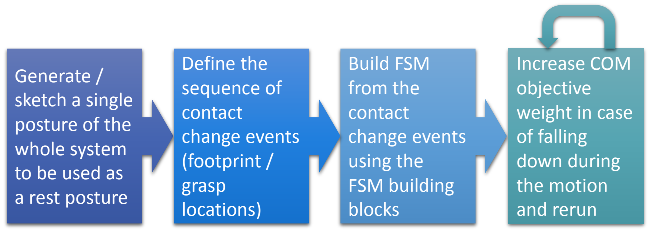

Generate / sketch a single posture of the whole system to be used as a rest posture
Define the sequence of contact change events (footprint / grasp locations)
Build FSM from the contact change events using the FSM building blocks
Increase COM objective weight in case of falling down during the motion and rerun

Fig. 7. Animation authoring workflow.

would find a feasible solution. See Appendix B for a minimal theoretical example comparison between our integrated QP approach versus multi-QP strategies.

To create the final animation, the user needs to decompose the motion in phases within each of which the instance of the QP (35) remains the same, i.e phases with a fixed set of contacts and a fixed set of objectives, though the tracked target value of a given objective can vary in time within a given phase. A finite-state machine (FSM) handles the transitions between the different phases of the motion (states of the FSM) when transition conditions are realized, e.g. foot landed, COM shifted, grasp established, grasp released, contact established, contact broken, etc.

Advanced FSM strategies as proposed in [13], [14] can also be used, though the use of [13] would require an additional effort in adapting its prioritized formulation to our weighted one (for example by assigning weights one order of magnitude higher for every priority level), especially for locomotion phases for which we just contented ourselves in the demonstrated examples with basic quasi-static, slow-gait, locomotion FSMs (alternating swinging feet and shifting COM projection on the new support polygon) for illustration purposes. See Figs. 6 and 7.

4 Results

We assessed our framework with original animation scenarios involving and incorporating multi-character interaction and cooperation, object manipulation, and interaction with the environment, see Fig. 8.

King

Two characters are lifting a third one sitting on a litter vehicle (king carrier). The whole system is made of 4 subsystems: The three characters (floating-base multi-body systems, 6 + 30 DOFs each), and the litter vehicle (free-floating rigid object, 6 DOFs). The FSM for this example is provided in Appendix A. In the interactive mode, the user controls the 3D (x, y, z) position of a point in the scene that the carried character has to reach with her left hand (reaching task). The cooperative behavior of the two carriers in adjusting the position and orientation of the litter vehicle to ease the task for the third character emerges automatically, without any explicit specification.

Funambulist

A character walks along a narrow beam (width of the beam equals that of the character's foot), holding a barbell with randomly time-varying weights at each extremity. This is a locomotion-and-manipulation system). The animation is decomposed into an autonomous locomotion phase and a user-interactive one. The locomotion phase is controlled by a cyclic two-state FSM as in Fig. 6. The collisions between the legs during the swing phase are automatically avoided despite the constrained narrow line walking. When writing the FSM the user only has to worry about the foot landing position and mid-step height without providing collision-free trajectories. In the interactive mode, the arrows displayed on the scene are used to increase/decrease each of the two weights of the barbell. The character reacts in real-time making the adjustments in her posture to keep balance autonomously.

Sword

Two characters engage in an unfair battle with one of them equipped with a sword and the second one bare hands. The second character is however endowed with superior collision-avoidance capabilities that allow her to survive by dodging the swordsman's sword swipes while keeping balance. The animation is decomposed into a scripted phase and a user-interactive phase. In the scripted phase

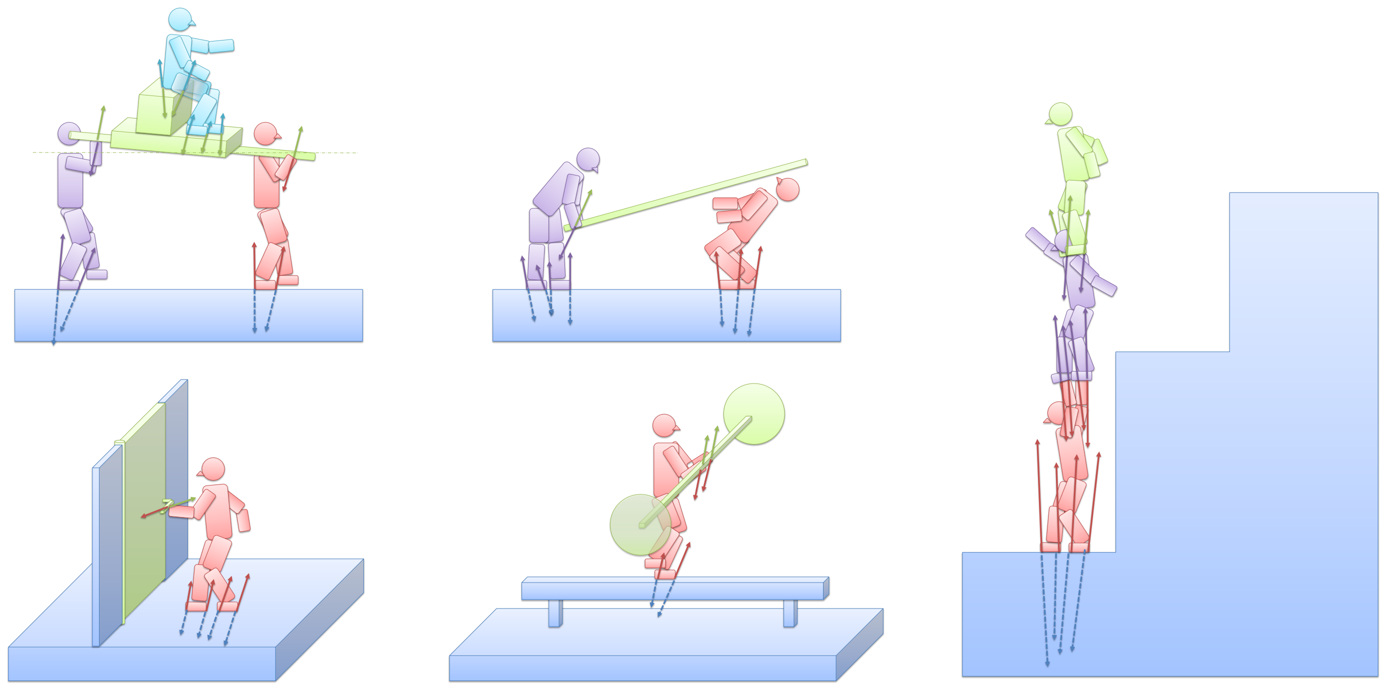


Fig. 8. Schematic representations of the demonstrated example scenarios. In each scenario each subsystem is represented in a different color and the contact forces applied on a given subsystem are represented in the same color as the subsystem (gravity forces are not represented for clarity).

the swordsman follows a sword trajectory pre-designed by the user. In the user-interactive mode the arrows on the scene are used to control the movements of the sword, autonomously driving the movements of the swordsman without explicit control of his posture or his end-effector tasks. The dodging character's movements are fully autonomous and they all emerge from the two constraints: balance and collision avoidance.

Acrobats

Three characters team up in a three-story human tower building enterprise. For the purposes of this animation we dropped the torque limit constraint of the bottom character enabling her with intentional superhuman power. The detailed FSM of the scenario is provided in Appendix A. In the user interactive phase the user controls the 3D position of the COM of the top character, thus shaking the whole tower structure that manages to keep upright and to avoid collapsing by autonomously adjusting the postures and COMs of subsystems of characters.

Door, Box, Valve, Lever

The final animations demonstrate various kinds of interactions with the environment:

Door: The character opens and closes a door. The door is modeled as a 2-DOF fixed base multi-body system: 1 DOF for the hinges of the door and 1 DOF for the knob (both revolute joints, both un-actuated). The animation is decomposed into two phases: 1) rotating the knob and 2) rotating the door.

Valve: The valve has also 2 DOFs, one revolute joint at the handle and one for the valve itself, the two joints having parallel axes this time.

Box: The character operates a box by establishing planar contacts between her hands and two opposite surfaces of the boxes (not grasps). A random 6D trajectory of the box is specified and the character tracks it autonomously accounting for its mass and keeping balance.

Lever: The character operate a 1 DOF non-actuated lever equipped with passive spring-damper at its revolute joint and with an on-purpose voluminous part near its end. The character follows while keeps balance and avoiding collision with the lever.

Tables 1 and 2 and Figures 9 and 10 present experimental figures for all the scenarios. The computation time figures were collected on a laptop Dell Alienware 14 with 7.7GiB memory, Intel Core i7-3720QM@2.60GHzx8, running under Ubuntu 12.04 64bits from a `C++` implementation of the framework. The EOMs, kinematics, and dynamics were computed using the implementations of the algorithms in [43]. Simulations were performed directly by integrating the resulting $\ddot{q}$ of the QP without using an external simulator. The QP solver used is LSSOL [44]. The generic humanoid character model (mass, inertia, link lengths, torque limits, joint angles and velocity limits) we used was an HRP-4 model. Dynamics parameters for the other objects, door, valve, sword... were roughly estimated based on real-life objects and simple geometric model formulas.

5 Discussion and Conclusion

We presented a framework that greatly extends the scopes of applications of QP-based character controllers to animation scenarios beyond simple locomotion. We wrote all the EOMs of the characters, floating objects, articulated environment parts, as particular instances of the general multi-body system dynamics equation. We coupled them together through

TABLE 1
Experimental figures for the example scenarios

	King	Funamb.	Sword	Acrobats	Door
Nb. subsyst.	4	2	3	3	2
DOFs	114	42	78	108	38
Act. DOFs	90	30	60	90	30
Planar ctc. pts.	28	8	16	24	8
Grasp ctc. pts.	16	8	8	0	4
Nb. Col. pairs	6	7	33	6	1
QP size	**380**	**136**	**234**	**294**	**116**
Eq. constr.	246	90	150	180	74
Ineq. constr.	298	99	317	282	93
Min t (ms)	4.45	0.89	0.76	5. 97	0.77
Max t (ms)	84.51	5.10	14.97	34.88	2.33
Med t (ms)	**22.72**	**1.29**	**6.12**	**8.10**	**0.89**
Δt (ms)	33.33	5.00	10.00	10.00	5.00

TABLE 2
Quantitative comparisons between this work (last column) and selected reference prior work: Abe et al. (2007) [12], Jain et al. (2009) [14] and de Lasa et al. (2010) [13]

	[12]	[14]	[13]	**This**
Retained QP output	τ	q	$\ddot{q}$	$\boldsymbol{\ddot{q}}$
Pb. size (char. DOFs)	41	37	41	**38 - 114**
QP solver	MOSEK	SNOPT	MOS./QPC	**LSSOL**
Processor (Intel)	P4	Core 2	Xeon	**Core i7**
QP control freq. (Hz)	30	100	100	**30 - 200**
Time-step (ms)	33	10	10	**5 - 33**
Resol. time (ms)	13 - 19	100 - 500	10 - 20	**0.9 - 22**
Real-time	100%	2 to 10%	50 to 100 %	**100%**
Multi-char. form.	no	unclear	no	**yes**
Reactive coll. av.	no	no	no	**yes**

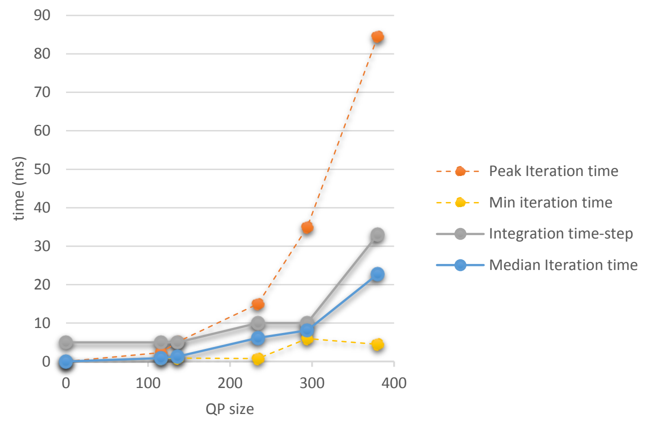


Fig. 9. A representation of the computation time requirements of our framework as a function of the complexity of the scenarios (represented here by the maximum QP size of the scenario, i.e. the total number of scalar variables in the QP of the scenario). The median iteration time (blue curve) is the most significant data as the peak iteration times (red dashed curve) are rarely reached and occur only at isolated points of time during the motion (mainly at the discrete contact change events). The integration time-step is adjusted to the median iteration time to keep real-time interactivity possible. From the profile of the blue curve we can expect that the framework would still have reasonable though non real-time computation times for significantly more complex scenarios if desired.

physical and behavioral interactions in the form of multi-character-specific constraints or task objectives. We could thus adapt the multi-objective feature-based QP control approach to the control of the full system that is made up of all the moving and interacting elements/characters that appear in the scene.

The focus of this work was on the low-level controller. The latter was coupled with FSMs that decompose the scenarios into states with a fixed set of tasks and transition conditions between those states to change/add/remove tasks. Although simple FSMs were used mainly to serve as demonstrators of the performances of the low-level controller, this simplicity might have lead in some cases to over-simplifications that resulted in unrealistic behaviours, such as the perfect coordination of the carriers' feet in the King scenario. In these cases a little more creative effort would be required from the user in the FSM design.

The use of simple FSMs also caused two other limitations. First the balance criterion used throughout the demonstrated scenarios was a quasi-static one, controlling the ground projections of the COMs of the multi-characters system to their statically stable positions with setpoint tasks. Second, the absence of a look-ahead scheme in the controller prevents realizing more dynamic movements while staying balanced. These two limitations can be handled in the future by coupling the local QP controller presented here with a preview controller, in a model-predictive control (MPC) scheme (e.g. [17] or more recently in multi-contact behaviors [45]) on the COM of the full system and/or on the COMs of selected subsystems. A more challenging direction lies in increasing the level of autonomy of the framework by even sparring the user the design of the FSM itself and deriving this FSM from a planning phase, such as the idea in [34] with even higher-level objective specifications.

In this work, the interaction among the multiple characters are instantaneous and perfect, which does not resemble how the humans actually interact with each other. This could lead to unnatural visual results. It can be improved in future work by simulating the communication delays between the characters. Such communication could occur through vision, forces, speech, etc, all of which can be integrated in the framework.

In additional future work, the computation times can be further substantially improved for larger problems by taking advantage of the sparsity of the QPs and hence using a solver that handles this property. We plan to integrate motion-capture data in the framework by replacing the rest pose objective with the reference motion tracking objective.

Finally, this work is currently being applied to real robots. We are controlling through this scheme multiple humanoid robots collaborating for a task and a single humanoid robot that manipulates mobile or articulated parts of the environment such as a door or a drawer. This was made possible since the humanoid robots we are applying the work to, namely the HRP-2 and HRP-4 robots, are position controlled. Thus we are able to use the joint accelerations output by the QP and integrate them to send the resulting position commands to the robots. This control scheme justifies our approach versus a torque-output one which would be suitable for torque-controlled robots.

Appendix A FSM Details

See Figs 11 and 12.

Appendix B Theoretical Case Study

See Fig 13.

Acknowledgments

This work is supported in part by grants from the EU FP7 KoroiBot project www.koroibot.eu. The authors would like to thank Amine Bouyarmane for his help editing the accompanying video.

References

[1] M. H. Raibert and J. K. Hodgins, "Animation of Dynamic Legged Locomotion," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 25, no. 4, pp. 349–358, 1991.

[2] J. K. Hodgins, W. L. Wooten, D. C. Brogan, and J. F. O'Brien, "Animating Human Athletics," *ACM Transactions on Graphics (SIGGRAPH)*, pp. 71–78, 1995.

[3] W. Wooten and J. Hodgins, "Simulating Leaping, Tumbling, Landing and Balancing Humans," in *IEEE International Conference on Robotics and Automation*, 2000, pp. 656–662.

[4] P. Faloutsos, M. van de Panne, and D. Terzopoulos, "Composable Controllers for Physics-based Character Animation," *ACM Transactions on Graphics (SIGGRAPH)*, pp. 251–260, 2001.

[5] K. Yin, K. Loken, and M. van de Panne, "SIMBICON: Simple Biped Locomotion Control," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 26, no. 3, p. 105, 2007.

[6] Y.-Y. Tsai, W.-C. Lin, K. B. Cheng, J. Lee, and T.-Y. Lee, "Real-Time Physics-Based 3D Biped Character Animation Using an Inverted Pendulum Model," *IEEE Transactions on Visualization and Computer Graphics*, vol. 16, no. 2, pp. 325–337, 2010.

[7] A. Liegeois, "Automatic Supervisory Control of the Configuration and Behavior of Multibody Mechanisms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 7, no. 12, pp. 868–871, 1977.

[8] O. Khatib, "A Unified Approach for Motion and Force control of Robot Manipulators: The Operational Space Formulation," *IEEE Journal of Robotics and Automation*, vol. 3, no. 1, pp. 43–53, 1987.

[9] L. Sentis and O. Khatib, "Synthesis of Whole-Body behaviors Through Hierarchical Control of Behavioral Primitives," *International Journal of Humanoid Robotics*, vol. 2, no. 4, pp. 505–518, 2005.

[10] L. Righetti, J. Buchli, M. Mistry, and S. Schaal, "Inverse dynamics control of floating-base robots with external constraints: A unified view," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 1085–1090.

[11] Y. Abe and J. Popović, "Interactive Animation of Dynamic Manipulation," in *Symposium on Computer Animation*, 2006, pp. 195–204.

[12] Y. Abe, M. da Silva, and J. Popović, "Multiobjective Control with Frictional Contacts," in *Symposium on Computer Animation*, 2007, pp. 249–258.

[13] M. de Lasa, I. Mordatch, and A. Hertzmann, "Feature-Based Locomotion Controllers," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 29, no. 3, p. 131, 2010.

[14] S. Jain, Y. Ye, and C. K. Liu, "Optimization-Based Interactive Motion Synthesis," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 28, no. 1, p. 10, 2009.

[15] A. Macchietto, V. Zordan, and C. R. Shelton, "Momentum Control for Balance," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 28, no. 3, p. 80, 2009.

[16] M. Al Borno, E. Fiume, A. Hertzmann, and M. de Lasa, "Feedback Control for Rotational Movements in Feature Space," *Computer Graphics Forum*, vol. 33, no. 2, pp. 225–233, 2014.

[17] I. Mordatch, M. De Lasa, and A. Hertzmann, "Robust Physics-Based Locomotion Using Low-Dimensional Planning," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 29, no. 4, p. 71, 2010.

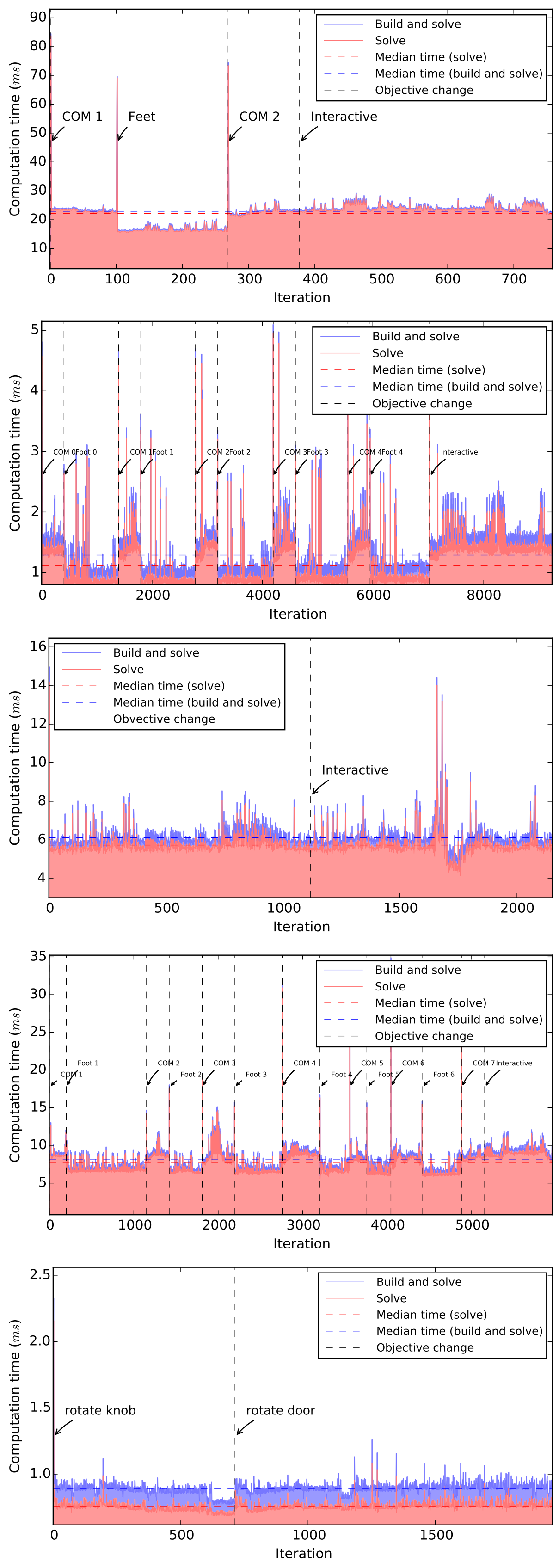
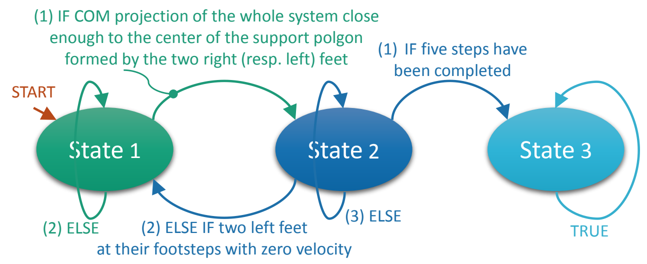


Fig. 10. Tracking of the computation times of each time-step of the example scenarios. From top to bottom: one step of the King scenario, the Funambulist scenario, the Sword scenario, the Acrobats scenarios, and the Door scenario. Vertical dashed lines represent FSM state transition events.


(1) IF COM projection of the whole system close enough to the center of the support polgon formed by the two right (resp. left) feet
(1) IF five steps have been completed
START
State 1
State 2
State 3
(2) ELSE
(2) ELSE IF two left feet at their footsteps with zero velocity
(3) ELSE
TRUE

State 1: Move COM
- Set COM projection of the whole system at the center of the support polygon formed by the two right (resp. left) feet
- Set chair orientation horizontal
- Set rest posture

State 2: Move two feet in parallel
- Set the two left (resp. right) feet to their next footsteps
- Set COM projection of the whole system at the center of the support polygon formed by the two right (resp.left) feet
- Set chair orientation horizontal
- Set rest posture

State 3: Interactive reaching
- Set lifted character's left hand position at user cursor
- Set COM projection of the whole system at the center of the support polygon formed by the four feet
- Set rest posture

Fig. 11. Detailed FSM for the King scenario.

[18] C. K. Liu and Z. Popović, "Synthesis of Complex Dynamic Character Motion from Simple Animations," *ACM Transactions on Graphics*, vol. 21, no. 3, pp. 408–416, 2003.

[19] A. C. Fang and N. S. Pollard, "Efficient Synthesis of Physically Valid Human Motion," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 22, no. 3, pp. 417–426, 2003.

[20] M. Al Borno, M. de Lasa, and A. Hertzmann, "Trajectory Optimization for Full-Body Movements with Complex Contacts," *IEEE Transactions on Visualization and Computer Graphics*, vol. 19, no. 8, pp. 1405–1414, 2013.

[21] Y. Koga, K. Kondo, J. J. Kuffner, and J.-C. Latombe, "Planning Motions with Intentions," *ACM Transactions on Graphics (SIGGRAPH)*, pp. 395–408, 1994.

[22] K. Yamane, J. Kuffner, and J. K. Hodgins, "Synthesizing Animations of Human Manipulation Tasks," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 23, no. 3, pp. 532–539, 2004.

[23] S. Jain and C. K. Liu, "Interactive Synthesis of Human-Object Interaction," in *Symposium on Computer Animation*, 2009, pp. 47–53.

[24] Y. Bai, K. Siu, and C. K. Liu, "Synthesis of Concurrent Object Manipulation Tasks," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 31, no. 6, p. 156, 2012.

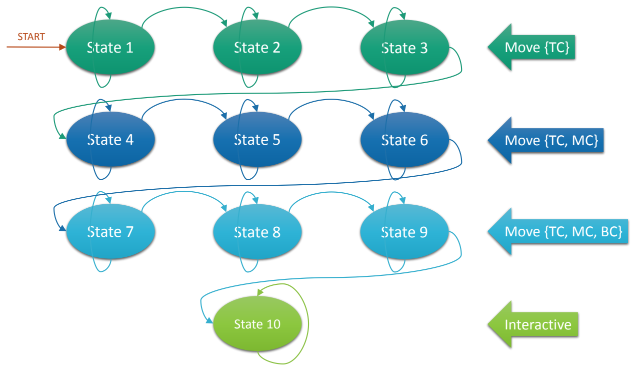
[25] C. Esteves, G. Arechavaleta, J. Pettré, and J.-P. Laumond, "Animation Planning for Virtual Characters Cooperation," *ACM Transactions on Graphics*, vol. 25, no. 2, pp. 319–339, 2006.

[26] K. Wampler, E. Andersen, E. Herbst, Y. Lee, and Z. Popović, "Character Animation in Two-Player Adversarial Games," *ACM Transactions on Graphics*, vol. 29, no. 3, p. 26, 2010.

[27] H. P. H. Shum, T. Komura, and S. Yamazaki, "Simulating Multiple Character Interactions with Collaborative and Adversarial Goals," *IEEE Transactions on Visualization and Computer Graphics*, vol. 18, no. 5, pp. 741–752, 2012.

[28] C. K. Liu, A. Hertzmann, and Z. Popović, "Composition of Complex Optimal Multi-Character Motions," in *Symposium on Computer Animation*, 2006, pp. 215–222.

[29] S. Coros, A. Karpathy, B. Jones, L. Reveret, and M. Van De Panne, "Locomotion Skills for Simulated Quadrupeds," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 30, no. 4, p. 59, 2011.



State 1: Move TC's right foot
Set TC's right foot position on MC's right shoulder
Set COM projection of TC over TC's left foot SP
Set COM projection of MC over MC's two feet SP
Set COM projection of BC over BC's two feet SP
Set rest pose

State 2: Move TC's COM
Set COM projection of TC over TC's right foot SP
Set COM projection of MC over MC's two feet SP
Set COM projection of BC over BC's two feet SP
Set rest pose

State 3: Move TC's left foot
Set TC's left foot position on MC's left shoulder
Set COM projection of TC over TC's rightfoot SP
Set COM projection of MC over MC's two feet SP
Set COM projection of BC over BC's two feet SP
Set rest pose

State 4: Move MC's right foot
Set MC's right foot position on BC's right palm
Set COM proj. of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's left foot SP
Set COM proj. of BC over BC's two feet SP
Set rest pose

State 5: Move {MC,TC}'s COM
Set COM projection of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's right foot SP
Set COM projection of BC over BC's two feet SP
Set rest pose

State 6: Move MC's left foot
Set MC's right left position on BC's left palm
Set COM proj. of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's right foot SP
Set COM proj. of BC over BC's two feet SP
Set rest pose

State 7: Move BC's right foot
Set BC's right foot position to next footstep
Set COM proj. of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's two feet SP
Set COM proj. of {BC,MC,TC} over BC's left foot SP
Set rest pose

State 8: Move {BC,MC,TC}'s COM
Set COM proj. of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's two feet SP
Set COM proj. of {BC,MC,TC} over BC's right foot SP
Set rest pose

State 9: Move BC's left foot
Set BC's left foot position on next footstep
Set COM proj. of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's two feet SP
Set COM proj. of {BC,MC,TC} ov. BC's right foot SP
Set rest pose

State 10: Interactive TC's COM control
Set TC's COM at user target cursor
Set COM proj. of TC over TC's two feet SP
Set COM proj. of {MC,TC} over MC's two feet foot SP
Set COM proj/ of {BC,MC,TC} over BC's two feet SP
Set rest pose

Fig. 12. Detailed FSM for the Acrobats scenario. For clarity we only detail the state objectives and omit the transition conditions. Abbreviations used int he descriptions of objectives: SP Support Polygon, TC: Top Character, MC: Middle Character, BC: Bottom Character.

[30] C. K. Liu, "Dextrous Manipulation from a Grasping Pose," *ACM Transactions on Graphics*, vol. 28, no. 3, p. 59, 2009.

[31] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of Complex Behaviors Through Contact-Invariant Optimization," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 31, no. 4, p. 43, 2012.

[32] ——, "Contact-Invariant Optimization for Hand Manipulation," in *Symposium on Computer Animation*, 2012, pp. 137–144.

[33] K. Bouyarmane and A. Kheddar, "Static Multi-Contact Inverse Problem for Multiple Humanoid Robots and Manipulated Objects," in *IEEE-RAS International Conference on Humanoid Robots*, 2010, pp. 8–13.

[34] ——, "Multi-Contact Stances Planning for Multiple Agents," in *IEEE International Conference on Robotics and Automation*, 2011, pp. 5546–5353.

[35] E. S.-L. Ho, T. Komura, and C.-L. Tai, "Spatial Relationship Preserving Character Motion Adaptation," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 29, no. 4, pp. 33:1–33:8, 2010.

[36] K. Bouyarmane and A. Kheddar, "Humanoid Robot Locomotion and Manipulation Step Planning," *Advanced Robotics*, vol. 26, no. 10, pp. 1099–1126, 2012.

[37] A. Clegg, J. Tan, G. Turk, and C. K. Liu, "Animating human dressing," *ACM Transactions on Graphics (SIGGRAPH)*, vol. 34, no. 4, pp. 116:1–116:9, Jul. 2015.

[38] M. Peinado, D. Meziat, D. Maupu, D. Raunhardt, D. Thalmann, and R. Boulic, "Accurate on-line avatar control with collision anticipation," in *ACM Symposium on Virtual Reality Software and Technology*, 2007, pp. 89–97.

[39] A. Escande, S. Miossec, M. Benallegue, and A. Kheddar, "A Strictly Convex Hull for Computing Proximity Distances with Continuous

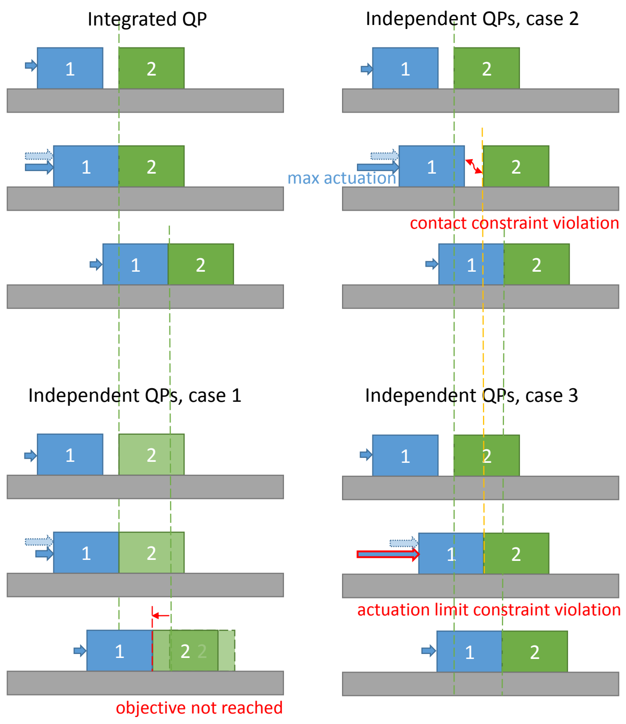
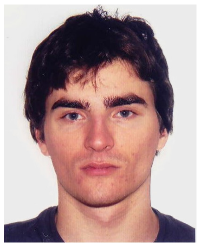


Fig. 13. A theoretical case study to compare the proposed approach with different other approaches. The system is made of object 1 that is actuated through a thrust force (blue arrow) and object 2 that is unactuated. The task is on the final position of object 2, which cannot move by itself and thus needs an interaction with object 1 to realize the task. We compare different approaches in each of the four cases, first the proposed approach based on one integrated QP and then three different approaches based on two independent QPs, one for each object. 1) In the top-left, the proposed integrated QP solves for the actuation force of object 1 to push object 2 to the desired final position. Both equations of motions and dynamic parameters of the two objects, as well as the specified actuation limit constraint on object 1, are taken into account when computing the force. 2) In the bottom-left, an independent QP is formulated for object 1 to reach the same final position as in the previous case. It results in lesser actuation force since the mass of object 2 is not taken into account when computing the force on object 1. As a consequence the actuation force is not sufficient to push both objects 1 and 2 together when they come into contact, and finally the task for object 2 is not realized. 3) In the top-right, an independent kinematic QP is formulated for object 2 to realize its task without accounting for the actuation limit of object 1. Subsequently, an independent QP is formulated for object 1 to follow the resulting motion of object 2. It results in an actuation force on object 1 that reaches its specified limit before being able to match the kinematic motion of object 2, which means that the contact between object 1 and object 2 is never established throughout the motion. 4) In the bottom-right, the contact constraint is enforced, which leads to an actuation force on object 1 that violates its specified limit in order to be able to match the motion of object 2.

Gradient," *IEEE Transactions on Robotics*, vol. 30, no. 3, pp. 666–678, 2014.
[40] B. Faverjon and P. Tournassoud, "A Local Based Approach for Path Planning of Manipulators with a High Number of Degrees of Freedom," in *IEEE International Conference on Robotics and Automation*, 1987, pp. 1152–1159.
[41] F. Kanehiro, F. Lamiraux, O. Kanoun, E. Yoshida, and J.-P. Laumond, "A Local Collision Avoidance Method for Non-Strictly Convex Polyhedra," in *Robotics: Science and Systems IV*, Zurich, Switzerland, June 2008.
[42] P. Baerlocher and R. Boulic, "An Inverse Kinematics Architecture Enforcing an Arbitrary Number of Strict Priority Levels," *The Visual Computer*, vol. 20, no. 6, pp. 402–417, 2004.
[43] R. Featherstone, *Rigid Body Dynamics Algorithms*. Springer-Verlag New York, Inc., 2007.
[44] P. E. Gill, S. J. Hammarling, W. Murray, M. A. Saunders, and M. H. Wright, "User's Guide For LSSOL (Version1.0): A Fortran Package for Constrained Linear Least-Squares and Convex Quadratic Programming," Stanford University, Department of Operations Research, Tech. Rep., 1986.
[45] H. Audren, J. Vaillant, A. Kheddar, A. Escande, K. Kaneko, and E. Yoshida, "Model Preview Control in Multi-Contact Motion Application to a Humanoid Robot," in *IEEE/RSJ International Conference on Intelligent Robots and Systems*, 2014, pp. 4030–4035.



Joris Vaillant received the Master degree in interactive systems at Paul Sabatier University, Toulouse, France, in 2010 and the PhD degree in robotics from the University of Montpellier in 2015. His research interest is contact planning and whole body motion for humanoids and virtual avatars.

Karim Bouyarmane received the double Ingenieur diploma from Ecole Polytechnique in Palaiseau in 2007 and from Ecole Nationale Superieure des Mines de Paris in 2008, and the PhD degree from the University of Montpellier in 2011. He completed the PhD program in the Joint Robotics Laboratory (JRL) at the National Institute of Advanced Industrial Science and Technology (AIST) in Tsukuba, Japan. He then held a Japan Society for the Promotion of Science (JSPS) post-doctoral fellowship at the Advanced Telecommunications Research Institute International (ATR) in Kyoto, in the Computational Neuroscience Laboratories working on brain-robot interfaces. He is currently a CNRS research associate.

Abderrahmane Kheddar received the BSCS degree from the Institut National dInformatique (ESI), Algiers, the MSc and PhD degrees in robotics, both from the University of Pierre and Marie Curie, Paris 6. He is presently Directeur de Recherche at CNRS. He is the Director of the CNRS-AIST Joint Robotic Laboratory (JRL), UMI3218/RL, Tsukuba, Japan; and the leader of the Interactive Digital Humans (IDH) team at CNRS-UM LIRMM at Montpellier, France. His research interests include humanoid robotics, haptics, and thought-based control. He is a founding member of the IEEE/RAS chapter on haptics (now acting as a senior advisor), the co-chair and co-founding member of the IEEE/RAS Technical committee on model-based optimization. He is presently Editor of the IEEE Transactions on Robotics, and the Journal of Intelligent and Robotic Systems. He is a founding member of the IEEE Transactions on Haptics and served in its editorial board during three years (2007-2010), he also served as associate editor in the MIT Press PRESENCE journal. He coordinated or acted as a PI for several EU projects. He is titular (full) member of the National Academy of Technologies of France (NATF) and a Senior Member of the IEEE Society.