Multi-Character Physical and Behavioral Interactions Controller
Joris Vaillant, Karim Bouyarmane, Abderrahmane Kheddar

To cite this version:
Joris Vaillant, Karim Bouyarmane, Abderrahmane Kheddar. Multi-Character Physical and Behavioral Interactions Controller. IEEE Transactions on Visualization and Computer Graphics, Institute of Electrical and Electronics Engineers, 2017, 23 (6), pp.1650-1662. 10.1109/TVCG.2016.2542067. hal-01246357

HAL Id: hal-01246357
https://hal.archives-ouvertes.fr/hal-01246357
Submitted on 18 Dec 2015

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
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:

$$\begin{align*}
\text{min} & \quad \sum \text{quadratic objectives} \\
\text{subject to} & \quad \begin{cases}
\text{equation of motion} \\
\text{contact no-slip} \\
\text{friction cone limits} \\
\text{joint torque limits}
\end{cases}
\end{align*} \tag{1}$$

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 shall prefer a more stable and replicable behavior, independent from the chosen external simulator, by integrating directly the accelerations resulting from the optimization. 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.

In the previous works, the QP (1) was formulated exclusively for single character animation problems. Our contribution is to extend it to systems made of arbitrary numbers of interacting characters and objects, all in all 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 and ensure 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, resulting in the generation of 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 representative 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 behav-
Our technical contributions are highlighted 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. In these latter two sections we only reproduce existing works from the literature. 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 human 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. [11].

The task-space formulation is either based on strict hierarchy 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 from previously proposed QP-based motion controllers, in particular the two works [12], [13], and to a lesser extent [14]. [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. This work was followed by [15], [16] that extended it to periodic walking.

Based on a similar QP formulation, but with a hybrid priority-weighted policy for the objectives/constraints, [13] proposed a more general-purpose controller for the locomotion of various biped characters. Momentum objective as proposed in [17] and later used in [18] was included and shown to yield “natural-looking” behaviors for walking or jumping. The QP controller was in this work coupled

\[
\begin{align*}
\text{min} & \quad \sum \text{quadratic objectives} \\
\text{subject to} & \quad \text{system of equations of motion} \\
& \quad \text{– coupled through Newton’s third law} \\
& \quad \text{contact no-slip (with the fixed environment)} \\
& \quad \text{contact no-slip (multi-character interaction)} \\
& \quad \text{friction cone limits} \\
& \quad \text{joint torque limits} \\
& \quad \text{joint position and velocity limits} \\
& \quad \text{collision avoidance}
\end{align*}
\]

In the QP (2) our technical contributions are highlighted. The red-colored components are contributions pertaining to the formulation of the problem as a multi-character system. The blue-colored components are independent of the multi-character nature of the problem and can as well be incorporated into existing single character controllers. We experimentally confirm 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 (red-colored parts 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. In these latter two sections we only reproduce existing works from the literature. 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.

Fig. 1. A selection of example animation scenarios that can be modeled in our framework, screenshots from the accompanying video.
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 in [13] was also used as a low-level controller that realizes a higher-level plan in [19], showcasing robust bipedal walking and running simulations.

The authors in [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 and those of [14] 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. Instead, we focus on the systematic modeling of arbitrary systems in the most general setting, e.g. object manipulation, and on our novel behavioral interactions. Another advantage appears in comparison to the non real-time aspect of [14]. Yet [14] provides more advanced FSM design methodologies than ours since we mainly focus on the low-level controller. Our controller can for example be used with the FSMs of [14].

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 [20], [21], [22]. All these works however are mainly about the locomotion of one character in the world and do not integrate, with that locomotion behavior, a manipulation behavior component [11], [23], [24], [25], [26], a cooperative behavior component [27], [28], [29], [30], a quadruped walking behavior component [31], a dexterous hand component [32], to cite a few examples among a vast body of existing works and approaches in these fields.

Recently [33], [34] 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 [35]. 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 dextrous 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 proposed such a unification approach of humanoid behaviors in non real-time motion generation contexts. Those are the static single posture generation problem for multiple robot systems in [36], formulated as an inverse kinematics problem integrating all the robots and objects at once, and the planning problem of a sequence of such static postures in [35]. We propose now a real-time controller based on the same philosophy.

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 fusion the aspects of real-time interactive physics simulation proposed in [12], [13] with the general motion planning philosophy adopted in [35] [33], or, in other words, we target the same level of generality attained in the latter using the more flexible, interactive-control-enabling approach of [12], [13].

3 Method

3.1 Equation of Motion

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 \( q_i \in \mathbb{R}^{n_i} \) (which includes the free-floating base position/orientation if any and the joint angles if any), and behave following their respective EOM1

\[
M_i(q_i)\ddot{q}_i + N_i(q_i, \dot{q}_i) = J_{all,i}(q_i)^T f_{all,i} + S_i \tau_i ,
\]

(3)

where \( \tau_i \in \mathbb{R}^{n_i} \) 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 \( S_i \in M(\mu_i, a_i) \) is the selection matrix that maps the dimension of \( \tau_i \) to that of \( q_i \) by extending \( \tau_i \) 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 \( \nu_i \) punctual contact forces \( f_{all,i} \in \mathbb{R}^{3n_i} \) with respective Jacobians at the corresponding contact points \( J_{all,i} \in M(3\nu_i, \mu_i) \). \( M_i \) 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).

1. The notations of the paper are consistent with the conventional identification of vectors as column matrices (and not as row matrices) \( \mathbb{R}^r \equiv M(r, 1) \), meaning that \((\lambda_1, \ldots, \lambda_r) \equiv (\lambda_1 \cdots \lambda_r)^T \) and in particular that \((\lambda_1, \ldots, \lambda_r) \neq (\lambda_1 \cdots \lambda_r) \). \( M(\alpha, \beta) \) denotes the set of real matrices of \( \alpha \) rows and \( \beta \) columns.
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 in subsystem \( j \)'s EOM according to Newton’s third law. We thus rewrite all Equations (3) in the following forms:

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

where \( f_{0,j} \) 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.

\[
F_k = (f_{k,i})_{1 \leq i \leq n}, \quad k = 0, 1, 2. \tag{5}
\]

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 \( \phi_i \) 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:

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

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):

\[
\begin{align*}
M(q) &= \text{diag} (M_1(q_1), \ldots, M_n(q_n)), \\
J_k(q) &= \text{diag} (J_{k,1}(q_1), \ldots, J_{k,n}(q_n))_{k=0,1,2}, \\
S &= \text{diag} (S_1, \ldots, S_n), \\
\Phi &= \begin{pmatrix} \phi_1 & \\ \vdots & \\ \phi_n \end{pmatrix}, \\
N(q, \dot{q}) &= \begin{pmatrix} N_1(q_1, \dot{q}_1) \\ \vdots \\ N_n(q_n, \dot{q}_n) \end{pmatrix},
\end{align*}
\]

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

\[
M(q)\ddot{q} + N(q, \dot{q}) = J_0(q)^T F_0 + \left(J_1(q)^T - J_2(q)^T \Phi \right) F_1 + S\tau. \tag{12}
\]

Note: \( \Phi \) defined in (10) is a square permutation matrix\( ^r \), 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.

### 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:

\[
\begin{align*}
J_0(q) \dot{q} &= 0, \\
J_2(q) \dot{q} &= \Phi J_1(q) \dot{q}.
\end{align*}
\]

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 \( \Phi \) introduced in the previous section appears to be helpful here and 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

\[
\begin{pmatrix} J_1(q)^T - J_2(q)^T \Phi \end{pmatrix}^T \dot{q} = 0,
\]

and, consequently, \( F_1 \) appears to be the Lagrange multiplier associated with this constraint in (12), which can thus be interpreted as the Lagrange’s equation of the whole system.
Equations (13) and (14) are time-differentiated to obtain constraints on the accelerations compatible with the QP:

$$ J_0(q) \ddot{q} + \dot{J}_0(q) \dot{q} = 0 $$

(16)

$$ \left( \Phi J_1(q) - J_2(q) \right) \ddot{q} + \left( \Phi \dot{J}_1(q) - \dot{J}_2(q) \right) \dot{q} = 0. $$

(17)

These formulations are prone to numerical instability (also reported in [13]) so we replace them in our implementation with a more stable behavior-yielding formulation as follows. For every contact between subsystem $i$ and the fixed inertial environment, let us denote $\nu_{0,i}$ the 6D linear and angular velocity and $J_{0,i} \in M(6, \mu_i)$ the corresponding linear and angular Jacobian of the contact link of the subsystem; the no-slip constraint for this contact link is written as:

$$ J_{0,i} \ddot{q}_i + \dot{J}_{0,i} \dot{q}_i = -\nu_{0,i}, $$

(18)

where $\Delta t$ is the integration time-step. These constraints (18) replace the constraint (16).

For a contact between subsystems $i$ and $j$, let $v_{1,i}$ and $\nu_{1,i}$, and $v_{2,j}$ and $\nu_{2,j}$ 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 $J_{1,i}$ and $J_{2,j}$ (for brevity again denoted $J_{i,j}$) denote the corresponding 6D Jacobians. A more stable behaviour than that of (17) is obtained by writing:

$$ \nu_{1,i} + \left( J_{1,i} \ddot{q}_i + \dot{J}_{1,i} \dot{q}_i \right) \Delta t = v_{2,j} + \left( J_{2,j} \ddot{q}_j + \dot{J}_{2,j} \dot{q}_j \right) \Delta t. $$

(19)

Yet, even this latter formulation might lead to numerical inaccuracies since the transformation from the frame of link 2 to that of link 1 is not constant over time (due to small perturbations leading to loss of contact), so we propose a refined version of (19) that ultimately proves stable in our experience (for integration time-steps $\Delta t$ ranging between 5ms and up to 33ms)

$$ \nu_{1,i} + \left( J_{1,i} \ddot{q}_i + \dot{J}_{1,i} \dot{q}_i \right) \Delta t - v_{2,j} - \left( J_{2,j} \ddot{q}_j + \dot{J}_{2,j} \dot{q}_j \right) \Delta t = \frac{\text{Err}(2J \tilde{X}_{1,i}^{\text{ref}} - 1J \tilde{X}_{2,j})}{\Delta t}, $$

(20)

where $\text{Err}(2J \tilde{X}_{1,i}^{\text{ref}} - 1J \tilde{X}_{2,j})$ expresses the 6D error between (a) $1J \tilde{X}_{2,j}$, the current transformation of the link 2 frame in the link 1 one, and (b) $1J \tilde{X}_{2,j}^{\text{ref}}$ that same transformation in the reference ideal situation where the contact is perfectly established (initial state of the contact).

### 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 dealt with this kind of constraint at such low level, and we believe that detailing it here would constitute an original addition to state-of-the-art QP-based controllers. Existing collision-avoidance approaches are usually encoded at higher levels with pre-defined, known, obstacle trajectories or with predicted obstacle motions, e.g. [14], [30]. 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]. The GJK algorithm is based on so-called support functions that allow, at a given configuration of two convex bodies, to return two witness points belonging to the surfaces of each body for which the distance is equal to the distance between the two bodies. These witness points move along the surfaces of the two bodies as their configuration change over time. We apply the strictly-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. 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

$$ d = \sigma ||p_{1,i} - p_{2,j}||, $$

(21)

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

$$ \ddot{d} \geq -\xi \frac{d - \delta_s}{\delta_t - \delta_s}, $$

(22)

where $\xi$, $\delta_s$, and $\delta_t$ are fixed parameters representing respectively the damping factor, the security distance, and the influence distance below which the constraint is activated. A QP-compatible version can be written as

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

(23)

Denoting $u = (p_{1,i} - p_{2,j})/d$ the unit vector between $p_{1,i}$ and $p_{2,j}$, the derivative $d$ is obtained as $\dot{d} = (\dot{p}_{1,i} - \dot{p}_{2,j})/(d^2).$
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 $\Lambda_0$ and $\Lambda_1$. See Fig. 5.

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

$$q_{\text{min}} \leq q \leq q_{\text{max}} \quad (26)$$

$$\dot{q}_{\text{min}} \leq \dot{q} \leq \dot{q}_{\text{max}} \quad (27)$$

$$\tau_{\text{min}} \leq \tau \leq \tau_{\text{max}} \quad (28)$$

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

$$\frac{q_{\text{min}} - \dot{q}}{\Delta t} \leq \dot{q} \leq \frac{q_{\text{max}} - \dot{q}}{\Delta t} \quad (29)$$

$$\frac{q_{\text{min}} - q - \dot{q} \Delta t}{\frac{1}{2} \Delta t^2} \leq \dot{q} \leq \frac{q_{\text{max}} - q - \dot{q} \Delta t}{\frac{1}{2} \Delta t^2} \quad (30)$$

However, the formulation (30) leads to strong deccelerations 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 $d_{\text{min}} = q - q_{\text{min}}$ and $d_{\text{max}} = q_{\text{max}} - q$, we replace (30) with

$$\frac{-\xi \cdot \Delta t d_{\text{min}} - \delta_{s} - \dot{q}}{\delta_{s} - \delta_{s} - \dot{q}} \leq \dot{q} \leq \frac{\xi \cdot \Delta t d_{\text{max}} - \delta_{s} - \dot{q}}{\delta_{s} - \delta_{s} - \dot{q}} \quad (31)$$

### 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 $F_0 = K_0 \delta_0$ 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 $\Lambda_0$ and $\Lambda_1$ the coefficients along these generators. The unilateral and friction cones constraints become:

$$\Lambda_0 \geq 0 \text{ and } \Lambda_1 \geq 0 \quad (25)$$

### 3.4.1 Example of Contact Surfaces and Contact Point Modeling

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.

---

**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 safety "forbidden" zone from which we consider that collision has occurred (this security distance can be reduced to zero), the light blue zone shows the influence distance from entering which the constraint is activated and starts influencing the motion of the bodies. The right figure is decomposed in three sequential frames. The bottom 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 frame, the motion is outside of the influence zone, so the motion is not affected. In the middle frame, the arm enters the influence zone and the constraint is activated, leading to the deviation of the motion that is shown in the top frame, in which the arm avoids collision by avoiding the purple security layer around the red obstacle.

$$\dot{p}_{2,j}^T u.\text{ Finally denoting } J_{1,i}^\text{lin} \text{ and } J_{2,j}^\text{lin} \text{ respectively the linear (translational) Jacobians of subsystems } i \text{ and } j \text{ at } p_{1,i} \text{ and } p_{2,j}, \text{ equation } (23) \text{ takes the following final form that we add as a constraint to the QP:}$$

$$u^T \left( J_{1,i}^\text{lin} \dot{q}_i + J_{2,j}^\text{lin} \dot{q}_j - J_{2,j}^\text{lin} \dot{q}_j - J_{2,j}^\text{lin} \dot{q}_j \right) +$$

$$\dot{u}^T \left( p_{1,i} - \dot{p}_{2,j} \right) \geq \frac{1}{\Delta t} \left( -\xi \frac{d_1 - \delta_1}{s_1 - \delta_1} \right). \quad (24)$$

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 waypoints 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.
which are added as constraints when \( d_{\text{min}} \leq \delta_i \) and \( d_{\text{max}} \leq \delta_i \) 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 \( J_q \) such that \( \dot{x} = J_q \dot{q} \) and \( \ddot{x} = J_q \ddot{q} + J_\dot{q} \dot{q} \). For a number \( M \) of simultaneous objectives \((x_1, \ldots, x_M)\) in a given phase of the animation, the quadratic cost function to minimize in the QP is defined as

\[
c_q,q(q) = \sum_{m=1}^{M} w_m ||\ddot{x}_m - \ddot{x}_d||^2, \tag{32}
\]

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

\[
\ddot{x}_d = -k(x - x^{\text{ref}}) - 2\sqrt{k(\ddot{x} - \ddot{x}^{\text{ref}})} + \ddot{x}^{\text{ref}}, \tag{33}
\]

where \( x^{\text{ref}} \) 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 \( x^{\text{ref}}, \) i.e \( \ddot{x}^{\text{ref}} = 0 \) and \( \dddot{x}^{\text{ref}} = 0 \) and thus

\[
\dddot{x} = -k(x - x^{\text{ref}}) - 2\sqrt{k}\dddot{x}, \tag{34}
\]

both for regulating the feature \( x \) around a constant value \( x^{\text{eq}} \) (\( x^{\text{ref}} = x^{\text{eq}} \)) and for steering the feature \( x \) to a distant target value \( x^{\text{eq}} \) (\( x^{\text{ref}} = x^{\text{eq}} \)) (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. 7, 12, 13 later in the paper), we textually refer to both these kinds of tasks (regulating and targeting) as “Set \( x \) to \( x^{\text{ref}}, \) ” 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-configuration 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 behav-

---

**Fig. 6.** The objective in this minimal theoretical case study is to use the multi-QP approach to manipulate the un-actuated object 2 with the actuated object 1. The desired task by the user is on the final position of object 2. The integrated QP (top left) takes into account the coupling between the two objects and the inability of object 2 to move “by itself” to fulfill the desired position task. Various decoupled QP strategies with different orderings are compared in the three other cases. In case 1 (bottom left), a QP is first formulated for object 1, that is not aware of (“doesn’t see”) the mass added by object 2. The QP for object 1 computes its motion to try to reach the same final position for itself as in the integrated QP case, but leading to a lesser actuation force (since the mass of object 2 is not accounted for) and to the final objective not reached in the physics simulation where the mass of object 2 slows down the motion. In the decoupled strategy of case 2 (top right), a kinematic QP is formulated for object 2 to follow the user-input desired task, and a dynamic QP is formulated for object 1 to try to follow the motion of object 2 in a physically-consistent manner. That QP for object 1 however reaches the actuation limit before reaching the contact, hence leading to the physically unrealistic aerial phase with the contact loss. Enforcing the kinematic contact constraint on the QP for object 1 to follow the output of the QP for object 2 leads to case 3 (bottom right) resulting in an acceleration and actuation force that violates the actuation limit.
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 would find a feasible solution. See Fig. 6 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. 7 and 8.

### 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. 9.

**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), adding up to a total of 114 DOFs for the system, of which 90 are actuated. The animation is decomposed into two phases: an autonomous locomotion and a user-interactive animations. The locomotion phase is decomposed into sub-phases representing the cyclic transitions between the two FSM states that are: 1) taking a step...
Fig. 9. 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).

by swinging the feet while keeping the projection of the COM of the whole system over the opposite feet support polygon, and 2) switching the COM of the whole system over the landed feet support polygon. 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 made of two subsystems: the character (6+30 DOFs) and the barbell (6 DOFs). 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.7. 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 trajectory. 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 system is composed of three subsystems: the two characters (30+6 DOFs each) and the sword (6 DOFs). The animation is decomposed into a scripted phase and a user-interactive phase. In the scripted phase the swordsman follows a sword trajectory pre-designed by the user, aiming at his opponent, then going back to the initial position, then aiming at the shoulder. 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. For this scene, a total of 33 collision-avoidance constraint pairs were added: 5 between the swordsman and his sword, 19 between his opponent and the sword, and 9 self-collision avoidance pairs for the dodging character.

**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 system is made of the three characters (30+6 DOFs each). The animation is decomposed into four meta-phases: 1) the top character climbs on the middle character, 2) the middle character carrying the top one climbs on the bottom character 3) the bottom character takes two steps while carrying the other two, and finally 4) a user-interactive phase. See Fig. 13 for the detailed FSM of the scenario. In the user interactive phase the user controls the 3D position of the COM of the top character, thus shaking the whole
We presented a framework that greatly extends the scopes of objects and simple geometric model formulas. Valve, sword... were roughly estimated based on real-life HRP-4. Dynamics parameters for the other objects, door, HRP-4 kinematics and dynamics model \cite{45} with different limits, joint angles and velocity limits we used was a humanoid character model (mass, inertia, link lengths, torque translator. The QP solver used is LSSOL \cite{44}. The generic humanoid figure \cite{43}. Simulations were performed directly by integrating C++
7.7GiB memory, Intel Core i7-3720QM@2.60GHzx8, running figures were collected on a laptop Dell Alienware 14 with mental figures for all the scenarios. The computation time balance and avoiding collision with the lever.

All these latter transformations that resulted in unrealistic behaviours. Such 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 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. This systematic approach unlocks various horizons of possibilities offered by all imaginable combinations between characters and objects that it allows, creating animation scenarios that simultaneously and seamlessly integrate locomotion components, manipulation components, cooperative behaviors components, interaction-among-characters components, and interactions-with-the-environment components.

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 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 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. This systematic approach unlocks various horizons of possibilities offered by all imaginable combinations between characters and objects that it allows, creating animation scenarios that simultaneously and seamlessly integrate locomotion components, manipulation components, cooperative behaviors components, interaction-among-characters components, and interactions-with-the-environment components.

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
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.

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. 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.

a one can be noted for example in the perfect coordination of the carriers’ feet in the King scenario. In these cases a little more creativity 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. [19] or more recently in multi-contact behaviors [46]) 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 [35] with even higher-level objective specifications.

In this work, cooperating characters instantaneously communicate each other’s “will” through the centralized QP rather than through vision, conversation, forces, etc. One additional improvement of the framework can be made by encoding the simulation of these realistic communication mean delays between the cooperating characters.

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. Lastly, we plan to integrate motion-capture data in the framework by replacing the rest pose objective (reference posture) with the reference motion data tracking objective.
Appendix
FSM Details
See Figs 12 and 13.

Fig. 12. Detailed FSM for the King scenario.

Fig. 13. Detailed FSM for the Acrobat scenario. For clarity we only detail the state objectives and omit the transition conditions. Abbreviations used in the descriptions of objectives: SP Support Polygon, TC: Top Character, MC: Middle Character, BC: Bottom Character.

References


