Motion Adaptation with Cascaded Inequality Tasks

Tomohiko Mukai
Tokyo Metropolitan University
tmki@acm.org

Shigeru Kuriyama
Toyohashi University of Technology
sk@tut.jp

Masaki Oshita
Kyushu Institute of Technology
oshita@ces.kyutech.ac.jp

ABSTRACT

A clip of character motion can be adapted to a change in environment or to another character of a different body size via a numerical optimization with several tasks including the objective of movement and physical constraints. Conventional methods, however, lack the design flexibility of such adaptation tasks because of the simple problem formulation. We propose a motion adaptation framework based on a cascaded series of quadratic programs. Our system introduces a layered structure of strictly prioritized tasks, each layer of which comprises arbitrary types of equality and inequality tasks. The cascaded solver identifies the optimal solution in each layer without affecting the fulfillment of the higher layer tasks. The stable computation of the cascaded optimization supports the intuitive design of the spacetime tasks even for novice users. The capability of our method was demonstrated through several experiments of motion adaptation with prioritized inequality tasks, such as environmental adaptation and adaptation of interactive behavior between two characters.

CCS CONCEPTS

• **Computing methodologies → Animation**.

KEYWORDS

motion adaptation, motion retargeting, cascaded quadratic programs

ACM Reference Format:
Tomohiko Mukai, Shigeru Kuriyama, and Masaki Oshita. 2019. Motion Adaptation with Cascaded Inequality Tasks. In *Motion, Interaction and Games (MIG '19), October 28–30, 2019, Newcastle upon Tyne, United Kingdom.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3359566.3360081

1 INTRODUCTION

Motion adaptation is a fundamental technique to deform handcrafted or motion-captured character animations for adapting to environmental change and another character of a different body size. For example, climbing motions can be adapted to a different climbing wall by modifying the whole body movement to reach the different location of graspable portions while preserving the stepping pattern of hands and feet. The adaptation solver deforms the source motion to satisfy several *tasks* and *constraints* of the character kinematics and dynamics. Although these terms are interchangeable depending on the context, *task* can mainly be used as a kinematic or dynamic goal of movement, including trajectory tracking of the end-effectors, a target joint angle at a time instant, and the minimization of kinetic energy. A *constraint*, however, should be a range of joint motion, joint torque limit, and foot position during ground contact.

Many conventional methods formulate the motion adaptation as a constrained spacetime optimization problem. In addition, most only address equality tasks, such as the goal position of an end-effector and the target joint angle, because they are intuitive for designers to interactively edit the character animation. However, inequality tasks also play a significant role in many possible scenarios. Imagine that a reaching motion in which a character reaches its arm into a hole is retargeted to another character of a different size. The conventional techniques require the trajectory of the hand and arm to be precisely planned as equality tasks to avoid collision with obstacles, even if there are many other feasible solutions. Such collision avoidance should be naturally formulated using simple inequality constraints and the adaptation solver must find the optimal motion that simultaneously satisfies such multiple inequality and equality tasks.

Moreover, we need to assign a strict order of priorities among multiple tasks because they are often in conflict with each other. For instance, the highest priority is probably assigned to constraints of the range of joint motion to certainly avoid an invalid pose. Collision avoidance is also important to create a physically valid animation. Other tasks, such as reaching and gazing, might have a relatively lower priority. A common approach uses a weighted combination of task functions for prioritization. However, the weighting strategy frequently causes unstable computation and a certain amount of violation even to the highest priority task because it minimizes the weighted sum of the task errors.

We propose a motion adaptation framework using a cascade of quadratic programs. Our system introduces a layered structure of strictly prioritized tasks, each layer of which is composed of many types of equality and inequality tasks. The quadratic programming solver optimizes motion to satisfy the tasks at each priority layer as much as possible while preserving the fulfillment of the more important higher layer tasks. In addition, we introduce several spacetime tasks that provide intuitive control for motion adaptation. The prioritized spacetime tasks allow novel adaptation methodologies, such as adaptation to the change in environmental geometry and behavioral adaptation in multi-character interactions. Our cascading framework provides a stable computation even for such a complicated scenarios. The technical contributions of this paper are as follows.

- Motion adaptation using a layered structure of strictly prioritized equality and inequality tasks
- Spacetime tasks that allow flexible motion adaptation

The disadvantage of our method is the high computational cost of cascaded quadratic programs, which is inapplicable to interactive

MIG '19, October 28–30, 2019, Newcastle upon Tyne, United Kingdom

computation. Our purely kinematic framework does not incorporate physical laws and cannot modify movement timing and duration. Despite these limitations, our generalized framework provides a stable solution for an arbitrary design of hierarchically prioritized tasks.

2 RELATED WORK

Several methods have been proposed to adapt a motion to a different environment or to another character while maintaining positions or rotational angle of the constrained joints [Choi and Ko 2000; Gleicher 1998; Monzani et al. 2000; Shin et al. 2001]. Semantics-preserving methods enable more aggressive motion retargeting among characters of highly-different topologies [Abdul-Massih et al. 2017; Celikcan et al. 2015; Hecker et al. 2008]. Spatial relationship-preserving motion adaptation [Ho et al. 2010] utilizes as-rigid-as-possible deformation of a volumetric mesh defined by the skeletal joints. This fundamental idea has been employed in several more recent works, such as interactive motion adaptation [Al-Asqhar et al. 2013; Bernardin et al. 2017; Molla et al. 2017; Tonneau et al. 2016] and the interaction-preserving adaptation for skinned characters [Jin et al. 2018; Liu et al. 2018]. Our spacetime formulation is similar to that of the constrained motion adaptation method [Ho and Shum 2013]. This method formulates the motion adaptation as a two-layered least-square problem with so-called soft and hard constraints. Our method uses a more general optimization framework to allow flexible design of the multi-layered structure of strictly prioritized spacetime tasks.

The inverse kinematics (IK) technique is sometimes used as a basic technique for motion adaptation and retargeting [Hecker et al. 2008; Monzani et al. 2000]. Many IK methods in computer animation are used to manipulate an end-effector of the character skeleton [Aristidou and Lasenby 2011]. For example, a real-time application frequently employs an analytical IK method for solving a floor contact constraint to prevent the penetration of feet into the ground. Several Jacobian-based IK methods have been proposed to strictly prioritize equality tasks by utilizing the redundancy of the articulated skeleton [Baerlocher and Boulic 2004; Yamane and Nakamura 2003]. Strict task priority is incorporated into a quadratic-programming-based method [Escande et al. 2010]. The hierarchical quadratic programming (HQP) method was proposed to address an arbitrary number of priority layers consisting of both equality and inequality tasks [Kanoun et al. 2011]. We extend this HQP-based IK solver to a motion adaptation problem.

Motion adaptation can be assumed to be a special case of constrained motion deformation. Per-frame IK techniques have been used to deform a pose in a reference motion at each time frame. For example, a data-driven method synthesizes a manipulation motion using per-frame IK such that the character's hands follow the path planned using a randomized search algorithm [Yamane et al. 2004]. The as-rigid-as-possible motion deformation technique [Kim et al. 2009] modifies the trajectory of joint positions using the gradient-domain curve editing method and deforms the pose at each frame using the per-frame IK. The sketch-based motion deformation technique [Choi et al. 2016] employs a similar approach to adapt the skeletal motion to follow the sketched trajectory of end-effectors. However, these methods allow only using equality tasks on the skeletal configurations, and are not applicable to impose an inequality task on the kinematic relationship between distant time frames. Furthermore, they use a simple weighting strategy to combine multiple tasks with prioritization.

Quadratic programming has been applied to physically-based motion control [da Silva et al. 2008a,b]. These methods generate a physically valid skeleton motion that adapts to the simulation environment while tracking a reference motion sequence. The multi-objective control method [Abe et al. 2007] uses a weighted combination of objective functions to control the importance of multiple tasks. QP-based optimization with weighted task functions is also used for deforming multi-character interactions [Liu et al. 2006]. Another physics-based controller generates several types of locomotion by optimizing strictly prioritized kinematic tasks [de Lasa et al. 2010]. This method assumes that the inequality constraints be imposed only at the highest priority layer. In contrast, our approach uses a cascaded series of QPs [Escande et al. 2014; Kanoun et al. 2011] for a flexible design of the hierarchy of equality and inequality tasks.

Physically based motion adaptation is another approach to retarget motion to new characters or to adapt a motion to different environments [Borno et al. 2018; Hodgins and Pollard 1997; Liu et al. 2005; Popović and Witkin 1999]. These methods provide physically valid results using kinematic and dynamic constraints that consider mass, force, and inertia moment. Our current system can only address kinematic tasks and an extension to such a physically based approach is an interesting future direction.

3 OVERVIEW

Our adaptation method is designed for a skeletal motion of an articulated character. Let $\mathbf{x}_f$ be a vector of the skeleton pose at the f-th timeframe that consists of the joint rotational angles $\boldsymbol{\theta}_{j,f}$ and the global position $\mathbf{p}_{\mathrm{root},f}$ and orientation $\boldsymbol{\theta}_{\mathrm{root},f}$ of the root node, and $\Delta\mathbf{x}_f$ be its variation, where j denotes the index of rotational joint. The skeleton motion is the concatenation of the skeleton pose vectors over all the timeframes as $\mathbf{m} = \mathbf{x}_1^\frown \mathbf{x}_2^\frown \cdots^\frown \mathbf{x}_F$ where F denotes the number of timeframes and $^\frown$ is the symbol of vector concatenation. The motion variation is also defined as $\Delta\mathbf{m} = \Delta\mathbf{x}_1^\frown \Delta\mathbf{x}_2^\frown \cdots^\frown \Delta\mathbf{x}_F$. See Appendix A for details of our implementation of the motion vector and its variation.

Our system requires manual operations to specify the adaptation tasks and assign the priority order to each. For example, to adapt a reaching motion to a different environment, we should design a range of joint motion constraint, a task to reach the goal, and collision avoidance constraints, in which the highest priority might be assigned to the two constraints while the reaching task should have a lower priority. The hierarchical structure of the prioritized tasks is then composed such that each layer contains the tasks of the same priority level. Optionally, we can also assign weights to the tasks to control the relative importance in every single layer. In addition, style preservation layers (§5.4) are added with the lowest priority to guarantee the adaptation result appears similar to the source motion.

Cascaded optimization is then performed to satisfy the tasks and constraints by deforming the source motion $\bar{\mathbf{m}}$. Our method is formulated as a constrained least-squares problem with equality

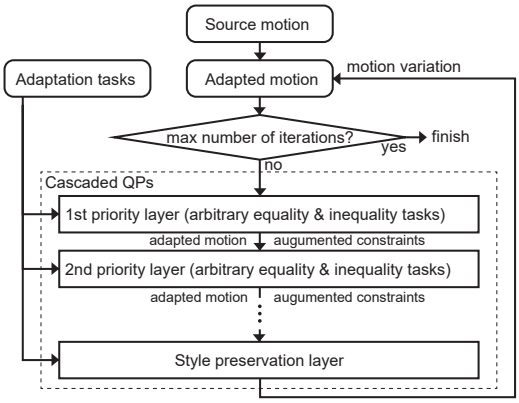


Figure 1: Cascaded optimization procedure

and inequality constraints on the character kinematics. We employ an iterative approach to solve the optimization problem that is nonlinear with respect to the motion variables. At the u-th iteration, the motion vector is updated as $\mathbf{m}_{u+1} = \mathbf{m}_u + \Delta\mathbf{m}_u$ where $\mathbf{m}_1 = \bar{\mathbf{m}}$ and the motion variation $\Delta\mathbf{m}_u$ is optimized via the cascaded QPs. The adaption result $\mathbf{m}^*$ is finally obtained after the manually specified number of iterations U as $\mathbf{m}^* = \mathbf{m}_U + \Delta\mathbf{m}_U$. For example, given an equality task regarding the joint position, each iteration step optimizes the motion variation using the equality task on the joint position such that the constrained joint gradually moves toward the goal every iteration.

In the next section, we explain the formulation of the cascaded optimization procedure; the details of the task designs are explained in §5

4 CASCADED ADAPTATION

The set of adaptation tasks consists of the linear equalities $\mathbf{A}_e\Delta\mathbf{m}_u = \Delta\mathbf{b}_e, e \in \mathcal{E}$ and inequalities $\mathbf{C}_i\Delta\mathbf{m}_u \le \Delta\mathbf{d}_i, i \in \mathcal{I}$ with respect to the motion variation $\Delta\mathbf{m}_u$ at the u-th iteration, where $\mathcal{E}$ and $\mathcal{I}$ denote the set of indices of the equality and inequality tasks; $\mathbf{A}_e$ and $\mathbf{C}_i$ are the Jacobian of each task with respect to the motion vector $\mathbf{m}_u$; and $\Delta\mathbf{b}_e$ and $\Delta\mathbf{d}_i$ are the target variations of task variables, respectively. Note that we discuss only the upper bound constraint because it encompasses the lower by using the simple sign inversion. The quadratic optimization at the u-th iteration is formulated as follows:

$$\min \; E_{\text{mot}}\left(\mathbf{m}_u, \bar{\mathbf{m}}\right) , \tag{1}$$

$$\text{subject to } \; \forall e \in \mathcal{E}, \; \mathbf{A}_e\Delta\mathbf{m}_u = \Delta\mathbf{b}_e , \tag{2}$$

$$\forall i \in \mathcal{I}, \; \mathbf{C}_i\Delta\mathbf{m}_u \le \Delta\mathbf{d}_i , \tag{3}$$

where $\cdot^T$ denotes the matrix transpose, and E_{mot} quantifies the dissimilarlity between the adaptated motion and the source one. Although the optimal solution should ideally satisfy all the tasks, the problem might be ill-conditioned in which some tasks are overconstrained or conflicted with other tasks, which leads to infeasibility or instability in numerical computation.

We extend the HQP framework [Escande et al. 2014; Kanoun et al. 2011] that introduces a hierarchy of strictly prioritized tasks. Figure 1 summarizes the optimization procedure of our framework. The optimzation constaints of Equation 2 and 3 are classified into multiple layers depending on their priorities, and the objective function E_{mot} of Equation 1 corresponds to the style preservation layer at the bottom which is detailed in §5.4. The cascaded series of quadratic optimization ensures that the solutions of higher priority tasks are not influenced by the optimization in the lower layers. The key idea in [Kanoun et al. 2011] is to achieve the tasks in each layer in a least-square sense; the equality tasks are relaxed as the objective function of the least-squares problem and the minimum amount of violation of inequality tasks is tolerated by using slack variables. The augmented constraints are also introduced to transmit the information of all tasks of the higher layers to the lower tasks.

We here briefly describe the procedure of the cascaded optimization; the details are explained in Appendix B. Let $\mathcal{E}^{(l)}$ and $\mathcal{I}^{(l)}$ be sets of indices of equality and inequality tasks assigned at the l-th priority layer where $l = \{1, \cdots, L\}$, $\mathbf{A}^{(l)}$ and $\mathbf{C}^{(l)}$ be the row-wise concatenation of $(\mathbf{A}_e)_{e\in\mathcal{E}^{(l)}}$ and $(\mathbf{C}_i)_{i\in\mathcal{I}^{(l)}}$, and $\Delta\mathbf{b}^{(l)}$ and $\Delta\mathbf{d}^{(l)}$ be the concatenation of $(\Delta\mathbf{b}_e)_{e\in\mathcal{E}^{(l)}}$ and $(\Delta\mathbf{d}_i)_{i\in\mathcal{I}^{(l)}}$, respectively. The optimization problem in the l-th layer at the u-th iteration is formulated as follows:

$$\min_{\Delta\mathbf{m}_u^{(l)}, \boldsymbol{\sigma}_u^{(l)}} \frac{1}{2}\left(E_{\text{eq}}^{(l)} + E_{\text{ie}}^{(l)}\right) , \tag{4}$$

$$E_{\text{eq}}^{(l)} := \left(\mathbf{A}^{(l)}\Delta\mathbf{m}_u^{(l)} - \Delta\mathbf{b}^{(l)}\right)^T \mathbf{W}_{\text{eq}}^{(l)} \left(\mathbf{A}^{(l)}\Delta\mathbf{m}_u^{(l)} - \Delta\mathbf{b}^{(l)}\right) , \tag{5}$$

$$E_{\text{ie}}^{(l)} := \boldsymbol{\sigma}_u^{(l)T} \mathbf{W}_{\text{ie}}^{(l)} \boldsymbol{\sigma}_u^{(l)} , \tag{6}$$

$$\text{subject to } \; \tilde{\mathbf{A}}\Delta\mathbf{m}_u^{(l)} = \Delta\tilde{\mathbf{b}} , \tag{7}$$

$$\tilde{\mathbf{C}}\Delta\mathbf{m}_u^{(l)} \le \Delta\tilde{\mathbf{d}} , \tag{8}$$

$$\mathbf{C}^{(l)}\Delta\mathbf{m}_u^{(l)} \le \Delta\mathbf{d}^{(l)} + \boldsymbol{\sigma}_u^{(l)} , \tag{9}$$

$$\boldsymbol{\sigma}_u^{(l)} \ge \mathbf{0} , \tag{10}$$

where $\boldsymbol{\sigma}_u^{(l)}$ represents the slack variable that corresponds to the violation magnitude and $\mathbf{W}_{\text{eq}}^{(l)}$ and $\mathbf{W}_{\text{ie}}^{(l)}$ are diagonal matrices of weights for equality and inequality tasks, respectively. After the cascaded optimization of the L priority layers, the motion vector $\mathbf{m}_u$ is updated using the optimized variation in the lowest style preservation layer as $\mathbf{m}_{u+1} = \mathbf{m}_u + \Delta\mathbf{m}_u^{(L)}$.

Equation 5 represents the relaxed equality task and Equations 6, 9, and 10 correspond to the relaxed inequality task. These tasks can be assumed to be *soft constraints* because they are imposed in a least-square sense. However, the augmented constraints, denoted by Equations 7 and 8, are imposed as *hard constraints* by using all higher layer tasks. The augmented equality relation between $\tilde{\mathbf{A}}$ and $\Delta\tilde{\mathbf{b}}$ consists of the solutions of the equality tasks of all higher layers. Moreover, inequality tasks that are violated in any layer among the first to the $(l-1)$-th layer are also included to prevent further violation. The augmented inequality relation between $\tilde{\mathbf{C}}$ and $\Delta\tilde{\mathbf{d}}$ is composed of feasible solutions of the other inequality tasks in any higher layer. The size of the augmented constraints increases in the lower layers because the task at every layer is added to either augmented constraint. In spite of a large number of constraints, the optimization problem still has a feasible solution and guarantees

stable computation because the augmented constraints are validated in the higher layers.

5 TASK DESIGN

Each task constrains either the joint angle relative to the parent or the joint position in the global coordinate system. Our iterative solver optimizes the motion variation $\Delta\mathbf{m}_u$ for gradually moving toward the target value of the tasks every iteration. Let $\mathbf{A}_u$ and $\Delta\mathbf{b}^*_u$ be the Jacobian matrix of the constrained variable with respect to the motion vector $\mathbf{m}_u$ and the target variations of the task variable at the u-th iteration, respectively. The equality task is formulated as follows:

$$\mathbf{A}_u\Delta\mathbf{m}_u = \Delta\mathbf{b}^*_u \ , \tag{11}$$

and the inequality task $\mathbf{C}_u\Delta\mathbf{m}_u \leq \Delta\mathbf{d}^*_u$ is similarly formulated by replacing the notations and the relation sign.

In this section, we define two types of primitive tasks as shown in Figure 2, in which the left-hand side Jacobian and the right-hand side variation in Equation 11 will be substituted with task-specific ones. More complex tasks are combined from the primitive tasks as explained in §5.3. Finally, our design of the style preservation layer is derived in §5.4. Note that we focus on the single iteration and omit the iteration index u in the following subsections.

5.1 Unary Tasks

A unary task modifies an independent configuration of the i-th joint in the f-th frame.

Joint angle Given the target angular variation $\Delta\theta^*_{j,f}$, the equality task is formulated as

$$\mathbf{S}_{j,f}\Delta\mathbf{m} = \Delta\theta^*_{j,f} \ .$$

The left-hand side Jacobian is substituted with the matrix $\mathbf{S}_{j,f} \in \mathfrak{R}^{1\times\dim(\mathbf{m})}$ that has only a single identity element at the j-th joint and the f-th frame as follows:

$$\mathbf{S}_{j,f}[1,i] = \begin{cases} 1 & \text{if} \quad i = j + \dim(\mathbf{x})\cdot f \\ 0 & \text{otherwise} \end{cases} \ .$$

When the target joint angle $\theta^*_{j,f}$ is specified, the right-hand side variation is calculated at each iteration as $\Delta\theta^*_{j,f} = \alpha\left(\theta^*_{j,f} - \theta_{j,f}\right)$ where $\theta_{j,f}$ and α are the joint angle and the step size coefficient to control the variation magnitude, respectively. For example, the standing pose shown in Figure 2 (a1) is edited to Figure 2 (a2) by specifying the target joint angle of the right hip as shown by the solid red arrow. Moreover, the inequality task $\mathbf{S}_{j,f}\Delta\mathbf{m} \leq \Delta\theta^*_{j,f}$ is also defined to maintain the angular displacement within the upper bound, as shown by the dashed line segments in Figure 2 (a3).

Joint position Let $\mathbf{J}_{j,f}$ be the Jacobian of the joint position with respect to the motion vector. The equality task is defined as follows:

$$\mathbf{J}_{j,f}\Delta\mathbf{m} = \Delta\mathbf{p}_{j,f} \ ,$$

Given the target location $\mathbf{p}^*_{j,f}$, the target variation is calculated based on the displacement from the current position at each iteration, as $\Delta\mathbf{p}_{j,f} = \alpha\left(\mathbf{p}^*_{j,f} - \mathbf{p}_{i,f}\right)$. For example, the standing pose shown in Figure 2 (a1) is edited to Figure 2 (a3) by specifying the target position of the waist as shown by the red circle.

The inequality position task is also defined to maintain the joint position within a specified workspace, as shown by the dashed red line in Figure 2 (a3). Furthermore, we can impose the positional constraint only on selected components. For instance, if we impose the constraint on the Y component of the joint position as $\mathbf{J}_{j,f}|_Y\Delta\mathbf{m} = \alpha\left(0 - \mathbf{p}_{j,f}|_Y\right)$, the constrained joint is expected to be on the X-Z plane.

5.2 Binary Relational Tasks

A binary relational task controls the spacetime relation between two different joint configurations which are noted by the j-th joint in the f-th frame and the k-th joint in the g-th frame. This type of task is defined for angular displacement, positional displacement, and distance. For example, the angular displacement of the same joint between the different time frames is controlled if $j = k$ holds and the positional displacement between different joints in the same frame is modified if $f = g$ holds.

Inter-joint angle Given the target angular variation $\Delta\delta^*_\theta$, the equality task is formulated as follows:

$$\left(\mathbf{S}_{j,f} - \mathbf{S}_{k,g}\right)\Delta\mathbf{m} = \Delta\delta^*_\theta \ .$$

This task, in the case where $j \neq k$ and $f \neq g$, is illustrated by the solid green arcs in Figure 2 (b1) and (b2). In this example, the specified task is used to decrease the angular displacement between the knee angle in the first frame and the elbow angle in the last frame. The target variation is calculated by $\Delta\delta^*_\theta = \alpha\left\{\delta^*_\theta - \left(\theta_{j,f} - \theta_{k,g}\right)\right\}$ when the target angular displacement δ^*_θ is given.

Inter-joint position The equality task for relative joint position is derived using the Jacobian $\mathbf{J}_{j,f}$ and $\mathbf{J}_{k,g}$ as follows:

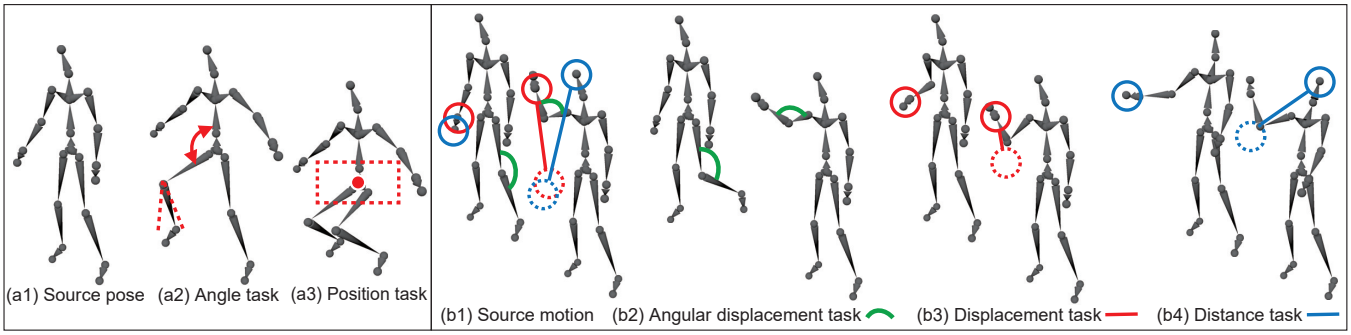
$$\left(\mathbf{J}_{j,f} - \mathbf{J}_{k,g}\right)\Delta\mathbf{m} = \Delta\boldsymbol{\delta}^*_{\mathbf{p}} \ .$$

When the target displacement $\boldsymbol{\delta}^*_{\mathbf{p}}$ is specified, the target variation is calculated by the following: $\Delta\boldsymbol{\delta}^*_{\mathbf{p}} = \alpha\left\{\boldsymbol{\delta}^*_{\mathbf{p}} - \left(\mathbf{p}_{j,f} - \mathbf{p}_{k,g}\right)\right\}$. Figure 2 (b1) and (b3) illustrates the condition of $j = k$ and $f \neq g$ by the red solid line, where the right wrist in the first frame and that in the last frame are nearer one another.

Inter-joint distance The relative distance task is designed to constrain the Euclidean distance between two joint positions and is derived from the first-order approximation of the distance change with respect to the motion variation $\frac{\partial}{\partial\mathbf{m}}\left\|\mathbf{p}_{j,f} - \mathbf{p}_{k,g}\right\|\Delta\mathbf{m}$, as follows:

$$\left\{\left(\mathbf{p}_{j,f} - \mathbf{p}_{k,g}\right)^T\left(\mathbf{J}_{j,f} - \mathbf{J}_{k,g}\right) \Big/ \left\|\mathbf{p}_{j,f} - \mathbf{p}_{k,g}\right\|\right\}\Delta\mathbf{m} = \alpha\left(\delta^*_{\text{dist}} - \left\|\mathbf{p}_{j,f} - \mathbf{p}_{k,g}\right\|\right) \ ,$$

where δ^*_{dist} and $\|\cdot\|$ denote the target distance and Euclidean norm, respectively. This task is illustrated by the solid blue line in Figure 2 (b1) and (b4). In this example, the Euclidean distance between the right hand in the first frame and the head in the last frame is



(a) Unary tasks (b) Binary relational tasks

Figure 2: Two types of primitive tasks. The unary tasks are used (a2) to constrain the joint rotation to a certain angle or (a3) the joint position to a certain location. The binary relational tasks constrain the (b2) angular displacement, (b3) positional displacement, and (b4) distance between temporally-separated and/or different two joints. The two poses of each binary relational task represent the first and last frame of the motion clip, respectively.

increased. Note that we can substitute the joint position $\mathbf{p}_{k,g}$ with the fixed position. For example, the inequality task can be used to tolerate the positional error between a joint and certain location.

5.3 Combinational Tasks

More complex tasks can be designed by combining the primitive tasks, such as the following two types.

Joint hull shape The shape of the convex hull among three or more joints can be constrained using a combination of inter-joint distance tasks. For example, when the character transfers a large box using both hands, the hands should remain parallel to the fixed distance. This combinational task consists of three distance tasks between the hand tips, between the wrists, and between the hand tip and the wrist of the other side. The usage example of this task is demonstrated in §6.2

Curve Shape The shape of the motion curve, including both the spatial trajectory of the joint position and the temporal profile of the joint angle, is constrained using the combinational displacement tasks. For example, a monotonical increase in rotational angle of the j-th joint over the time interval $[F_1, F_2]$ is imposed using a set of inequality tasks as follows:

$$\forall f \in [F_1, F2),\ -\left(\mathbf{S}_{j,f+1} - \mathbf{S}_{j,f}\right)\Delta\mathbf{m} \le \theta_{j,f+1} - \theta_{j,f}\ .$$

A local maximum or minimum of the animation curve can be completed using this approach. For example, the local maximum at the τ-th frame within the time period $[F_1, F_2]$ is imposed by the following

$$\begin{aligned}\forall f \in [F_1, \tau),\quad & -\left(\mathbf{S}_{j,f+1} - \mathbf{S}_{j,f}\right)\Delta\mathbf{m} \le \theta_{j,f+1} - \theta_{j,f},\\ \forall f \in [\tau, F_2),\quad & \left(\mathbf{S}_{j,f+1} - \mathbf{S}_{j,f}\right)\Delta\mathbf{m} \le -\left(\theta_{j,f+1} - \theta_{j,f}\right)\ .\end{aligned}$$

We can substitute the inequality tasks with the equality tasks for imposing the certain displacement magnitude. The usage example of the curve shape task is explained in §6.3

5.4 Style Preservation Layer

We impose a minimum motion variation and smooth velocity tasks to preserve the content and style of the source motion in the lowest layer, corresponding to E_{mot} in Equation 1. The minimum variation between the original motion and adapted motion is imposed using the tasks for the positional displacement of the root translation and the angular displacements on the root orientation and every joint rotation as follows:

$$\begin{aligned}\forall f,\quad & \mathbf{J}_{\text{root},f}\Delta\mathbf{m} = \beta_{\text{disp}}\left(\overline{\mathbf{p}}_{\text{root},f} - \mathbf{p}_{\text{root},f}\right)\ ,\\ \forall j, f,\quad & \mathbf{S}_{j,f}\Delta\mathbf{m} = \beta_{\text{disp}}\left(\overline{\theta}_{j,f} - \theta_{j,f}\right)\ ,\end{aligned}$$

where β_{disp} is the weighting coefficients and $\overline{\theta}_{j,f}$ and $\overline{\mathbf{p}}_{\text{root},f}$ denote the j-th joint rotation and the root translation of the source motion in the f-th frame, respectively. The smooth velocity task is imposed between neighboring timeframes using the positional displacement tasks and angular displacement tasks as follows.

$$\begin{aligned}\forall j, f,\quad & \left(\mathbf{S}_{j,f+1} - \mathbf{S}_{j,f}\right)\Delta\mathbf{m} = \beta_{\text{vel}}\left\{0 - (\theta_{j,f+1} - \theta_{j,f})\right\}\ ,\\ \forall f,\quad & \left(\mathbf{J}_{\text{root},f+1} - \mathbf{J}_{\text{root},f}\right)\Delta\mathbf{m} = \beta_{\text{vel}}\left\{\mathbf{0} - (\mathbf{p}_{\text{root},f+1} - \mathbf{p}_{\text{root},f})\right\}\ ,\end{aligned}$$

where β_{vel} is the weighting coefficients for the velocity change. The tasks in the style preservation layer are always in conflict with the other tasks because they constrain the rotations of all joints. Therefore, they must be given the lowest priority layer.

Note that several spacetime methods [Ho et al. 2010; Ho and Shum 2013] impose the minimum variation task as the objective function $\Delta\mathbf{m}^T\Delta\mathbf{m}$ of the optimization. However, our method cannot employ this simple approach because the minimization of the motion variation at each iteration does not guarantee the minimum difference after multiple iterations.

6 RESULTS

We implemented our prototype system using C++ language and an Operator Splitting Quadratic Program solver [Stellato et al. 2017] for solving the QP of each priority layer. The step size coefficients α for each task were set such that the magnitude of the target variation decreases to less than 0.5 and the weighting coefficients

for minimum variation β_{disp} and smooth velocity β_{vel} were set to 1.0. The computational time was measured on a workstation with dual Intel Xeon Gold 6154 CPUs of 3.0GHz and 128 GB RAM. All resulting animations are supplied as supplemental material.

6.1 Adaptation of Reaching Motion

A reaching motion was adapted to the environmental change using cascaded inequality tasks. The source motion reached the right hand to a certain location as shown in Figure 3 (a). The right hand showed a nearly-straight trajectory because there was no obstacle between the actor and the goal location in the real capture session. The source motion was adapted to reach the right hand for the same goal while avoiding a roof place over the goal. We designed five types of adaptation tasks as follows, where the priority order is noted as a Roman number:

I-a Range of joint motion (inequality)
I-b Poses at both end frames to match the source ones (equality)
II-a Foot positions (equality)
II-b Obstacle avoidance (less-than),
III Goal position of right hand (equality)

The adapted motion successfully avoided the collision by moving the right hand downward earlier than the source as shown in Figure 3 (b). The overall movement was visually similar to the source motion thanks to the style preservation of the lowest layer. Without the style preservation layer, jerky movement was caused as shown in the supplemental video because there was no other task to maintain temporal coherence. The computational time of the iterative optimization was 65 s.

Figure 3 (c) shows the motion produced by fixing the root translation and using the narrower range of motion of the torso joints to approximate the character having the stiffer torso. The character failed to reach the goal but maintained the height of the wrist under the roof according to the order of the task priority **II-b→III**. By swapping the priority order of collision avoidance and reaching the goal as **III→II-b**, the right hand could reach the goal while colliding with the roof as shown in Figure 3 (d). Figures 3 (e) and (f) show the same experiments but further limiting the range of joint motion of the torso. Both motions failed to reach the goal and the right wrist was at different positions according to the priority order. These results confirm that the cascaded optimization strictly fulfills the priority order of the adaptation tasks with stable numerical optimization.

To clarify this advantage of the cascaded optimization, we used a naive QP in which all the adaptation tasks were of the same priority in a single layer. Figure 3 (g) shows the resulting motion by assigning a uniform weight to all tasks. The range of joint motion tasks, corresponding to the stiffer torso joints, were obviously violated because the errors among all tasks were equalized. Next, we assigned prioritized weights such that a more important task had a greater weight as $10^{(5-\text{PriorityOrder})}$. For instance, the weight for the range of joint motion task was 10^4 and that for the style preservation was 1.0. However, the resulting motion collapsed, as shown in Figure 3 (h), because of the numerical instability. This experiment demonstrates the advantage of our framework which achieves stable computation that supports intuitive task design without tedious parameter tweaking.

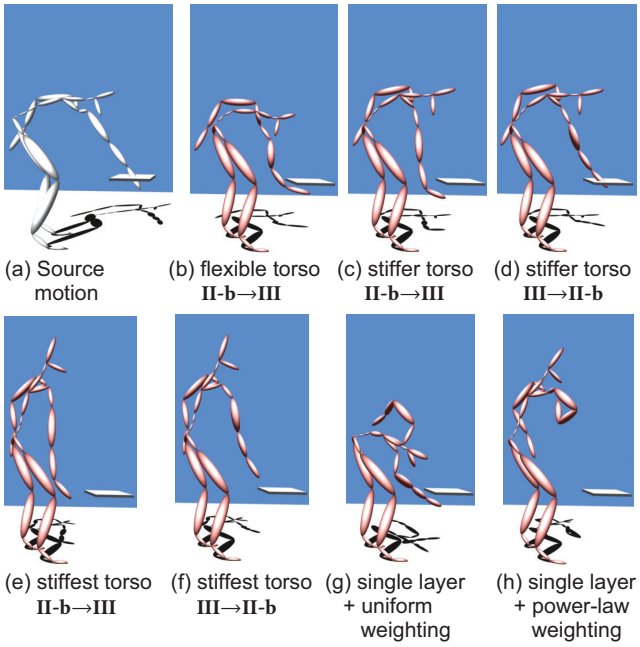


Figure 3: Adaptation of the reaching motion to avoid the roof. (a) Source motion. (b) Adapted motion using three-layer tasks. (c)-(f) Narrowing down the range of motion of the torso joints and swapping the layer II-b and III. (g) Non-cascaded optimization with uniform task weights. (h) Non-cascaded optimization with prioritized weighting according to task importance.

At the same time, this experiment also clarified the design difficulty of the obstacle avoidance task. This task was actually imposed as an upper bound constraint on wrist position during a time interval centered on the arrival time of the goal. The interval length was heuristically specified without explicitly considering the roof size. We could not formulate the roof avoidance as a simple inequality because it uses conditional branching *when the horizontal projection of the wrist enters that of the roof, the wrist must be under it.* To overcome this problem, we should develop a dynamic mechanism for imposing the adaptation task depending on the character configuration in our future work.

6.2 Motion Retarget with Kinematic Tasks

Our method is applicable to adapt a short motion clip to a new character with arbitrary kinematic tasks such as the trajectory of end-effectors and joint angles within a certain timeframe. The basic assumption here is that the source and target character have a similar skeletal topology in which each animating joint of the source skeleton has a one-by-one correspondence with the target skeleton. The retargeting process starts by simply copying the source motion to the target character neglecting the violation of physical or environmental constraints. The adaptation solver is then applied to the target character to satisfy the given retargeting tasks.

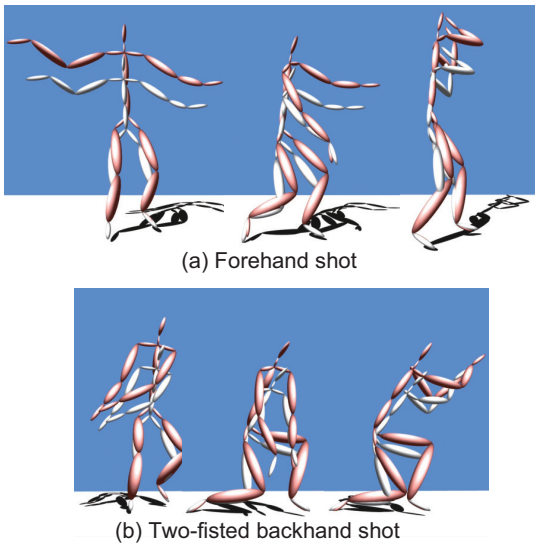


Figure 4: Retargeting of tennis strokes. (1) The forehand stroke was retargeted to the taller character using three tasks. (2) The two-fisted backhand stroke was retargeted to the taller character using four-layered tasks.

Figure 4 (a) shows the retargeting of a forehand stroke in tennis from a short actor to tall actor. The tall character tracked the foot trajectory of the source motion which was formulated using the joint position tasks. The right-hand trajectory around the shooting moment was also constrained to the source trajectory. In this example, we designed three-layered tasks as follows:

- **I** Range of joint motion (inequality)
- **II** Foot positions (equality)
- **III** Right hand trajectory around the shooting moment (equality)

The adapted motion successfully tracked the same trajectory of the feet and right hand. The computational time of 20 iterations was approximately 15 s.

Figure 4 (b) shows the motion retargeting of the two-fisted backhand stroke using the following four tasks.

- **I** Range of joint motion (inequality)
- **II** Foot positions (equality)
- **III** Right hand trajectory around the shooting moment (equality)
- **IV** Joint hull shape among the wrists, the left-hand tip, and right-hand tip (equality)

The first to third tasks are equivalent to the case of the forehand stroke. The joint hull shape task **IV** was added to maintain the spatial relationship between the two hands to grasp the racket using both hands. The priority of this task was relatively lower than the others because a slight movement of hands is observed in actual tennis play. As a result, the plausible movement of the backhand stroke was produced using this four-layered optimization which required 29 s.

6.3 Locomotion Adaptation to a Different Environment

The next experiment adapted a walking motion on a flat surface (Figure 5 (a)) to the stairs. We used the following adaptation tasks.

- **I** Range of joint motion (less-than)
- **II** Foot positions during ground contact (equality)
- **III** Vertical movement of feet during flight (higher-than / lower-than)

The second task was defined for each foot during ground contact. The third task is a curve shape task to inherit the outline of the vertical foot trajectory from the source motion. This task was defined for each foot during the flighting phase in which the upper bound constraint was imposed at the time when the foot of the source motion moved downward, and the lower bound constraint when the foot moved upward. The motion adaptation required 92 s with 80 iterations. The number of iterations was relatively higher than that of the other experiments because it required more iterations to address the large displacement in the character from the ground to the top of the stairs.

Figure 5 (b) demonstrates that the adapted motion preserved the walking motion content, particularly the vertical movement of feet. Each foot moves down immediately before contact and moves upward during the first half of the flighting phase; this behavior is easily lost in the conventional spacetime optimization framework. In fact, the feet penetrated through the steps without the inequality task **III** as shown in Figure 5 (c).

The drawback of this adaptation is in the purely kinematic approach. The adapted motion appears somewhat unnatural because the optimization does not consider the muscle activation, center of the mass movement, zero moment point, and other physical criteria. Although we could maintain the center of the mass position within the valid workspace using a weighted combination of joint position tasks, such an approach for static balance is not suited for dynamic biped locomotion. Moreover, the motion of climbing up stairs seems too fast because our method cannot modify the motion timing and duration. Our future work will include investigation of the temporal control and velocity editing in the optimization framework.

6.4 Multi-character Interaction

Our framework can be applied to motion adaptation of multiple characters by extending the optimization variables according to the number of characters. This approach provides flexible editing of the spatiotemporal relationship of multi-character interactions [Ho et al. 2010; Jin et al. 2018; Liu et al. 2006]. During this experiment, the source motions were the walking motion where the two characters passed each other, as shown in Figure 6 (a). The adaptation made the white character stretch his right hand to grasp the hand of the red character and the red character attempted to avoid the grasp. We designed four adaptation task types for this scenario.

- **I** Range of joint motion (less-than)
- **II** Foot positions (equality)
- **III** Minimum distance between the right hands of two characters (greater-than)

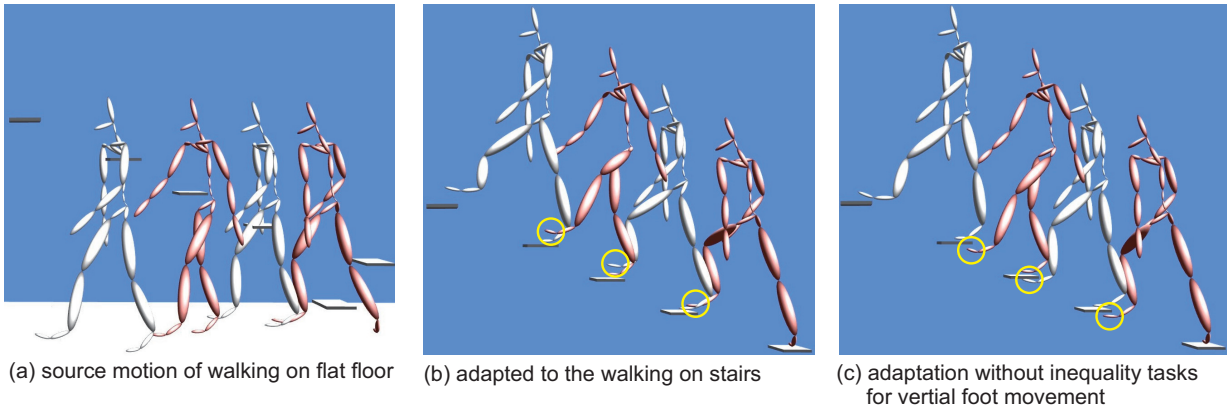


Figure 5: Environmental adaptation of walking motion on a flat floor to stairs. The white and red poses indicate the poses immediately before the right and the left foot contacts the ground, respectively. (a) Source motion. (b) Adapted motion using the inequality task III to constrain the vertical foot movement in which the flighting foot moved down immediately before contact as the yellow circles indicate. (c) Adapted motion without task III where the flighting foot moved up toward the target position and causing penetration through the steps.

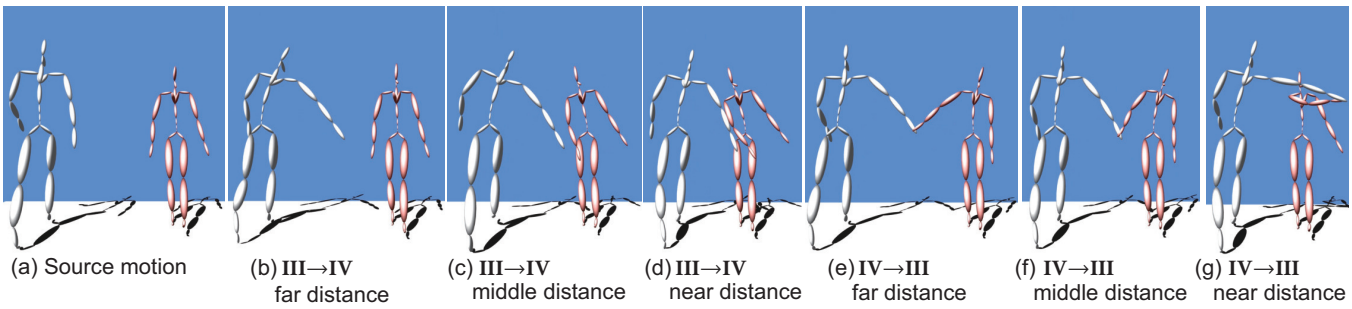


Figure 6: Two characters passing each other. The white character reached his right hand to grasp the right hand of the red character and the red character attempted to avoid the grasp. (a) Source walking motion. (b)-(d) Assigning the higher priority of the grasping task IV compared to that of the avoidance task III. (e)-(g) Swapping the priority order of these tasks.

IV No positional displacement between the right hands when passing each other (equality)

Figure 6 (b), (c), and (d) shows the adapted motion in which the distance between the characters changed. In Figure 6 (b), the red character did not take any avoidance reaction because the white character could not reach her hand because of the long distance. The avoidance behavior of the red character increased as the distance between the character approached that shown in Figure 6 (c) and (d). The average computational time was approximately 26 s.

Next, the priority order of the avoidance task **III** and the grasping task **IV** were swapped under the same three situations. As Figure 6 (b), (c), and (d) shows, the red character stretched out her hand because the minimum distance task was suppressed by the positional displacement task. These results were intuitively obtained by changing the priority order without tweaking any other continuous parameters.

7 DISCUSSION

We have proposed a motion adaptation framework using a cascaded structure of prioritized equality and inequality tasks. Our method extends the IK technique using hierarchical quadratic programs to the problem of motion adaptation. We defined two types of primitive spatiotemporal tasks and several combinational ones. The experimental results demonstrated the capability and flexibility of our method under several adaptation scenarios.

Our method formulates the motion adaptation problem as large and sparse linear systems with many constraints. In practice, we cannot process a long motion sequence with many tasks and priority layers because such a large system often results in a huge computational cost that is quadratic in the number of frames and the degrees of freedom of the character's skeleton. We could improve the computational performance using an advanced HQP solver [Escande et al. 2014]. The block coordinate descent technique [Liu et al. 2006] could also accelerate the optimization of multi-character interaction.

Another disadvantage is that our method can only kinematically optimize a motion clip of fixed length. The resulting motion often shows physically implausible result because our method involves no physical principle such as muscle energy minimization and joint torque limit. Moreover, our method does not modify the motion timing and duration, which limits the flexibility in creating variations from a single clip. One possible solution might be to integrate timewarp parametrization [Liu et al. 2006] into our framework. Generalization of hierarchically prioritized tasks in physically-based motion adaptation [de Lasa et al. 2010] is another challenging problem.

Our future work will include an investigation of an intuitive interface for designing spacetime tasks. For example, complex collision avoidance cannot be easily incorporated into our framework because the QP solver can only address task formulated as a linear relation. We should develop an adaptive mechanism of task generation for more complex adaptation. It is a more challenging problem to automatically optimize the priority order for each scenario.

ACKNOWLEDGMENTS

This work was supported by JSPS KAKENHI Grant Number 15K16110 and 15H02704.

REFERENCES

M. Abdul-Massih, Innfarm Yoo, and B. Benes. 2017. Motion Style Retargeting to Characters With Different Morphologies. *Computer Graphics Forum* 36, 6 (2017), 86–99.

Yeuhi Abe, Marco da Silva, and Jovan Popović. 2007. Multiobjective Control with Frictional Contacts. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2007*. 249–258.

Rami Ali Al-Asqhar, Taku Komura, and Myung Geol Choi. 2013. Relationship Descriptors for Interactive Motion Adaptation. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2013*. 45–53.

Andreas Aristidou and Joan Lasenby. 2011. FABRIK: A fast, Iterative Solver for the Inverse Kinematics Problem. *Graphical Models* 73, 5 (2011), 243–260.

Paolo Baerlocher and Ronan Boulic. 2004. An Inverse Kinematics Architecture Enforcing an Arbitrary Number of Strict Priority Levels. *The Visual Computer* 20, 6 (2004), 402–417.

Antonin Bernardin, Ludovic Hoyet, Antonio Mucherino, Douglas Gonçalves, and Franck Multon. 2017. Normalized Euclidean Distance Matrices for Human Motion Retargeting. In *Proc. of International Conference on Motion in Games*. 15:1–15:6.

Mazen Al Borno, Ludovic Righetti, Michael J. Black, Scott L. Delp, Eugene Fiume, and Javier Romero. 2018. Robust Physics-based Motion Retargeting with Realistic Body Shapes. *Computer Graphics Forum* 8 (2018), 81–92.

Ufuk Celikcan, Ilker O. Yaz, and Tolga Capin. 2015. Example-Based Retargeting of Human Motion to Arbitrary Mesh Models. *Computer Graphics Forum* 1 (2015), 216–227.

Byungkuk Choi, Roger Blanco i Ribera, J. P. Lewis, Yeongho Seol, Seokpyo Hong, Haegwang Eom, Sunjin Jung, and Junyong Noh. 2016. SketchiMo: Sketch-based Motion Editing for Articulated Characters. *ACM Transactions on Graphics* 35, 4 (2016), 146:1–146:12.

Kwang-Jin Choi and Hyeong-Seok Ko. 2000. Online Motion Retargetting. *Journal of Visualization and Computer Animation, vol. 11, pp. 223-235, 2000.* 11, 5 (2000), 223–235.

Marco da Silva, Yeuhi Abe, and Jovan Popović. 2008a. Interactive Simulation of Stylized Human Locomotion. *ACM Transactions on Graphics* 27, 3 (2008), 82:1–82:10.

Marco da Silva, Yeuhi Abe, and Jovan Popovi|'c. 2008b. Simulation of Human Motion Data Using Short-Horizon Model-Predictive Control. *Computer Graphics Forum* 27, 2 (2008), 371–380. Proc. of Eurographics 2008.

Martin de Lasa, Igor Mordatch, and Aaron Hertzmann. 2010. Feature-Based Locomotion Controllers. *ACM Transactions on Graphics* 29, 4 (2010), 131:1–131:10.

Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. 2010. Fast resolution of hierarchized inverse kinematics with inequality constraints. In *Proc. of IEEE International Conference on Robotics and Automation*. 3733–3738.

Adrien Escande, Nicolas Mansard, and Pierre-Brice Wieber. 2014. Hierarchical Quadratic Programming: Fast Online Humanoid-Robot Motion Generation. *The International Journal of Robotics Research* 33, 7 (2014), 1006–1028.

Michael Gleicher. 1998. Retargetting Motion to New Characters. In *Proc. of SIGGRAPH 98*. 33–42.

F. Sebastian Grassia. 1998. Practical Parameterization of Rotations Using the Exponential Map. *Graphics Tools* 3, 3 (1998), 29–48.

Chris Hecker, Bernd Raabe, Ryan W. Enslow, John DeWeese, Jordan Maynard, and Kees van Prooijen. 2008. Real-time Motion Retargeting to Highly Varied User-Created Morphologies. *ACM Transactions on Graphics* 27, 3 (2008), 27:1–27:12.

Edmond S.L. Ho, Taku Komura, and Chiew-Lan Tai. 2010. Spatial Relationship Preserving Character Motion Adaptation. *ACM Transactions on Graphics* 29, 4 (2010), 33:1–33:8.

Edmond S. L. Ho and Hubert P. H. Shum. 2013. Motion Adaptation for Humanoid Robots in Constrained Environments. In *IEEE International Conference on Robotics and Automation*. 1–6.

Jessica K. Hodgins and Nancy S. Pollard. 1997. Adapting Simulated Behaviors for New Characters. In *Proc. of SIGGRAPH 97*. 153–162.

Taeil Jin, Meekyoung Kim, and Sung-Hee Lee. 2018. Aura Mesh: Motion Retargeting to Preserve the Spatial Relationships between Skinned Characters. *Computer Graphics Forum* 37, 2 (2018), 311–320.

Oussama Kanoun, Florent Lamiraux, and Pierre-Brice Wieber. 2011. Kinematic Control of Redundant Manipulators: Generalizing the Task-Priority Framework to Inequality Task. *IEEE Transaction on Robotics* 27, 4 (2011), 785–792.

Manmyung Kim, Kyunglyul Hyun, Jongmin Kim, and Jehee Lee. 2009. Synchronized Multi-Character Motion Editing. *ACM Transactions on Graphics* 28, 3 (2009), 79:1–79:9.

C. Karen Liu, Aaron Hertzmann, and Zoran Popović. 2005. Learning Physics-Based Motion Style with Nonlinear Inverse Optimization. *ACM Transactions on Graphics* 3 (2005), 1071–1081.

C. Karen Liu, Aaron Hertzmann, and Zoran Popović. 2006. Composition of Complex Optimal Multi-Character Motions. In *Proc. of ACM SIGGRAPH/Eurographics Symposium on Computer Animation 2006*. 215–222.

Zhiguang Liu, Antonio Mucherino, Ludovic Hoyet, and Franck Multon. 2018. Surface based Motion Retargeting by Preserving Spatial Relationship. In *Proc. of Annual International Conference on Motion, Interaction, and Game*. 7:1–7:11.

Eray Molla, Henrique Galvan Debarba, and Ronan Boulic. 2017. Egocentric Mapping of Body Surface Constraints. *IEEE Transactions on Visualization and Computer Graphics* 7 (2017), 2089–2102.

Jean-Sébastien Monzani, Paolo Baerlocher, Ronan Boulic, and Daniel Thalmann. 2000. Using an Intermediate Skeleton and Inverse Kinematics for Motion Retargeting. *Computer Graphics Forum* 19, 3 (2000), 11–19.

Zoran Popović and Andrew Witkin. 1999. Physically Based Motion Transformation. In *Proc. of SIGGRAPH 99*. 11–20.

Hyun Joon Shin, Jehee Lee, Sung Yong Shin, and Michael Gleicher. 2001. Computer Puppetry: An Importance-based Approach. *ACM Transactions on Graphics* 20, 2 (2001), 67–94.

B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd. 2017. OSQP: An Operator Splitting Solver for Quadratic Programs. *ArXiv e-prints* (Nov. 2017). arXiv:math.OC/1711.08013

Steve Tonneau, Rami Ali Al-Ashqar, Julien Pettré, Taku Komura, and Nicolas Mansard. 2016. Character Contact Re–positioning Under Large Environment Deformation. *Computer Graphics Forum* 35, 2 (2016), 127–138.

Katsu Yamane, James J. Kuffner, and Jessica K. Hodgins. 2004. Synthesizing Animations of Human Manipulation Tasks. *ACM Transactions on Graphics* 23, 3 (2004), 532–539.

Katsu Yamane and Yoshihiko Nakamura. 2003. Natural Motion Animation through Constraining and Deconstraining at Will. *IEEE Transactions on Visualization and Computer Graphics* 9, 3 (2003), 352–360.

A IMPLEMENTATION OF MOTION VECTOR AND ITS VARIATION

We composed a motion vector $\mathbf{m}$ with a three-dimensional translation vector of the root joint and rotational quaternions of each joint rotation. The motion variation $\Delta\mathbf{m}$ consists of 3D displacement of the root position and a logarithm map of a joint rotational quaternion as used in [Choi et al. 2016]. For example, a joint rotation at the u-th iteration is represented as a quaternion $\boldsymbol{\theta}_u$, and the motion variation $\Delta\boldsymbol{\theta}_u$ is actually a logarithm map $\boldsymbol{\nu}_u$. The update of the motion vector is therefore computed by the product of the rotational quaternion and the exponential of the logarithm map as follows: $\boldsymbol{\theta}_{u+1} = \exp(\boldsymbol{\nu}_u)\,\boldsymbol{\theta}_u$ rather than the simple addition $\boldsymbol{\theta}_u + \Delta\boldsymbol{\theta}_u$. Each Jacobian matrix is also computed with respect to the logarithm map [Grassia 1998]. We experimentally confirmed that this approach

achieves singularity-free, and gimbal lock-free stable computation, rather than using Euler angles.

B HIERARCHICAL QUADRATIC PROGRAMS

We here explain the optimization algorithm for the hierarchical quadratic programs [Kanoun et al. 2011]. The optimization in the first layer starts with the equality tasks which are relaxed as the objective function and the inequality tasks which is also relaxed using slack variables. All the augmented constraints, i.e. $\tilde{\mathbf{A}}$, $\tilde{\Delta\mathbf{b}}$, $\tilde{\mathbf{C}}$, and $\tilde{\Delta\mathbf{d}}$, are initialized to empty before entering the first layer. The optimization in the first layer is thus formulated as follows:

$$
\begin{aligned}
&\min_{\Delta\mathbf{m}^{(1)},\boldsymbol{\sigma}^{(1)}} \frac{1}{2}\left(E_{\mathrm{eq}}^{(1)} + E_{\mathrm{ie}}^{(1)}\right),\\
&\quad E_{\mathrm{eq}}^{(1)} := \left(\mathbf{A}^{(1)}\Delta\mathbf{m}^{(1)} - \Delta\mathbf{b}^{(1)}\right)^{T} \mathbf{W}_{\mathrm{eq}}^{(1)} \left(\mathbf{A}^{(1)}\Delta\mathbf{m}^{(1)} - \Delta\mathbf{b}^{(1)}\right),\\
&\quad E_{\mathrm{ie}}^{(1)} := {\boldsymbol{\sigma}_u^{(1)}}^{T} \mathbf{W}_{\mathrm{ie}}^{(1)} \boldsymbol{\sigma}^{(1)},\\
&\text{subject to}\quad \mathbf{C}^{(1)}\Delta\mathbf{m}^{(1)} \le \Delta\mathbf{d}^{(1)} + \boldsymbol{\sigma}^{(1)},\\
&\qquad\qquad \boldsymbol{\sigma}^{(1)} \ge \mathbf{0}.
\end{aligned}
$$

Note that we can exceptionally impose some equality tasks as hard constraints if feasible.

The augmented constraints are then updated using the solution $\Delta\mathbf{m}^{(1)}$ to prevent further violation of infeasible tasks in the second priority layer. First, the current solution of the equality tasks is added to the augmented equality constraints as follows:

$$
\begin{aligned}
\tilde{\mathbf{A}} &\leftarrow \tilde{\mathbf{A}} \oplus \mathbf{A}^{(1)},\\
\tilde{\Delta\mathbf{b}} &\leftarrow \tilde{\Delta\mathbf{b}} \oplus \mathbf{A}^{(1)}\Delta\mathbf{m}^{(1)},
\end{aligned}
$$

where $\oplus$ denotes the row-wise matrix concatenation. Each element of inequality tasks is then evaluated whether $\mathbf{c}_h^{(1)}\Delta\mathbf{m}^{(1)} \le \Delta d_h^{(1)}$ is satisfied, where $\mathbf{c}_h^{(1)}$ and $\Delta d_h^{(1)}$ are the h-th row vector of $\mathbf{C}^{(1)}$ and the h-th element of $\Delta\mathbf{d}^{(1)}$, respectively. If the h-th subtask is satisfied, the same task is added to the augmented inequality constraints as follows:

$$
\begin{aligned}
\tilde{\mathbf{C}} &\leftarrow \tilde{\mathbf{C}} \oplus \mathbf{c}_h^{(1)},\\
\tilde{\Delta\mathbf{d}} &\leftarrow \tilde{\Delta\mathbf{d}} \oplus \Delta d_h^{(1)}.
\end{aligned}
$$

However, the infeasible subtask is added to the augmented equality task to prevent further violation in the lower layers as follows:

$$
\begin{aligned}
\tilde{\mathbf{A}} &\leftarrow \tilde{\mathbf{A}} \oplus \mathbf{c}_h^{(1)},\\
\tilde{\Delta\mathbf{b}} &\leftarrow \tilde{\Delta\mathbf{b}} \oplus \mathbf{c}_h^{(1)}\Delta\mathbf{m}^{(1)}.
\end{aligned}
$$

The update of the augmented constraints is repeated for all task elelements h.

Next, the second priority layer optimizes the motion variation with the updated augmented constraints and the tasks in the second layer as follows:

$$
\begin{aligned}
&\min_{\Delta\mathbf{m}^{(2)},\boldsymbol{\sigma}^{(2)}} \frac{1}{2}\left(E_{\mathrm{eq}}^{(2)} + E_{\mathrm{ie}}^{(2)}\right),\\
&\quad E_{\mathrm{eq}}^{(2)} := \left(\mathbf{A}^{(2)}\Delta\mathbf{m}^{(2)} - \Delta\mathbf{b}^{(2)}\right)^{T} \mathbf{W}_{\mathrm{eq}}^{(2)} \left(\mathbf{A}^{(2)}\Delta\mathbf{m}^{(2)} - \Delta\mathbf{b}^{(2)}\right),\\
&\quad E_{\mathrm{ie}}^{(2)} := {\boldsymbol{\sigma}^{(2)}}^{T} \mathbf{W}_{\mathrm{ie}}^{(2)} \boldsymbol{\sigma}^{(2)},\\
&\text{subject to}\quad \tilde{\mathbf{A}}\Delta\mathbf{m}^{(2)} = \tilde{\Delta\mathbf{b}},\\
&\qquad\qquad \tilde{\mathbf{C}}\Delta\mathbf{m}^{(2)} \le \tilde{\Delta\mathbf{d}},\\
&\qquad\qquad \mathbf{C}^{(2)}\Delta\mathbf{m}^{(2)} \le \Delta\mathbf{d}^{(2)} + \boldsymbol{\sigma}^{(2)},\\
&\qquad\qquad \boldsymbol{\sigma}^{(2)} \ge \mathbf{0}.
\end{aligned}
$$

The proceeding priority layers are recursively optimized using the same procedure. Finally, the optimal motion variation is obtained as the solution in the lowest priority layer $\Delta\mathbf{m}^{(L)}$.