Projective Motion Correction with Contact Optimization

Sukwon Lee, *KAIST* and Sung-Hee Lee, *KAIST*

Abstract—When motion capture data is applied to virtual characters, the applied motion often exhibits geometric and physical errors, which necessitates a cumbersome refinement process. We present a novel framework to efficiently obtain a corrected motion as well as its supporting contact information from multi-contact motion capture data. To this end, first, we present a projective dynamics-based method for optimizing character motions. By carefully defining objective functions and constraints using differential representation of motions, we develop a highly efficient motion optimizer that can create geometrically and dynamically adjusted motions given reference motion data and contact information. Second, we develop a contact optimizer that finds a set of contacts that allows the motion optimizer to generate a motion that best follows the reference motion under dynamic and geometric constraints. This is achieved by iteratively improving the hypothesis on the best set of contacts by getting feedback from the motion optimizer. We demonstrate that our method significantly improves the naturalness of a wide range of motion capture data, from walking to rolling.

Index Terms—Character Animation, Motion Capture, Motion Retargeting, Multi-Contact.

1 Introduction

Despite the widespread usage of motion capture data, obtaining a natural motion by applying motion capture data to a virtual character still remains a challenging task. Sensor noise and processing error during the capturing session can compromise the naturalness of motion data. In addition, due to size differences between a captured human subject and a target character, the character, when moving according to motion capture data, exhibits artifacts, such as penetration, sliding, and levitation. Therefore, a motion refinement process, which often requires expensive manual tuning, is almost always necessary. The refinement becomes even more challenging if a motion engages multiple contacts with the environment. This paper deals with correcting character motion so that it looks geometrically and dynamically natural while creating appropriate, possibly multiple, contacts with the environment.

Contact detection between a body part moving according to motion capture data and the environment is a fundamental problem for many data-driven animation processes. Nevertheless, effective solutions to this problem are rather rare. A simple approach is to judge that the contact has taken place if the height and velocity of a body part fall below certain thresholds [1]; this works fine for motion data with a low noise level. Due to its simplicity, this method has been widely used as a preprocess in computer graphics research [2], [3]. However, because the judgment is made only with respect to basic kinematic information, for challenging motions, such as high dynamic or multi-contact motions, the accuracy drops severely and uniform thresholds over the motion sequence do not perform properly.

A physics-based approach can provide powerful tools for physically correcting motion capture data, and researchers have developed a number of methods in this area, as will be discussed in Sec. 2. Most methods, however, assume that the contact information is given as an input. A notably different method is [4], in which a novel, sampling-based approach discovers both physically correct motion and contacts for complex multi-contact motions. This generalization naturally requires heavier computation. Our goal is not to develop a physics-simulated character but to modify a motion to make it look dynamically plausible, which allows us to take an efficient inverse dynamics-based approach.

This paper introduces a novel method to efficiently obtain both natural looking motion and its contact from captured data of multi-contact motion. Our method is characterized by a novel framework that discovers geometrically and physically realistic motion and its supporting contacts in an iterative manner. Two major components are a motion optimizer and a contact optimizer. The contact optimizer finds the optimal *contact state sequence*, which specifies which body parts should contact the environment at a certain time frame. Given the contact state sequence and the reference motion capture data, the motion optimizer computes a corrected motion that best follows the reference motion. The correction is performed with respect to the geometric feasibilities, such as avoiding environment penetration and foot sliding, and the dynamic feasibility with respect to the centroidal dynamics [5]. The quality of the resulting motion is evaluated by the motion optimizer, and is fed back to the contact optimizer to discover a better contact state sequence, which in turn is given to the motion optimizer.

Each optimizer is constructed based on novel contributions. For the motion optimizer to efficiently produce a plausible motion satisfying a given contact state sequence, we use an efficient projective dynamics technique [6], which was originally developed for creating soft body deformation. In order to adapt this technique to the motion editing problem, we carefully design constraints with regard to the geometric and physical correctness, in a form that is

• *Sukwon Lee and Sung-Hee Lee are with the Graduate School of Culture Technology, Korea Advanced Institute of Science and Technology, Yuseong-gu, Daejeon 34141, Republic of Korea.*
E-mail: {sukwonlee, sunghee.lee}@kaist.ac.kr

applicable to a motion, which is represented as a mesh. In particular, for fast convergence, geometric constraints such as pose preservation are expressed in differential coordinates.

The large number of possible contact combinations of body parts in a motion makes the computation of the optimal contact state sequence very challenging. As a practical solution to this problem, we develop a sampling-based, iterative approach that hypothesizes an optimal contact state sequence and verifies it by measuring its quality in terms of the error evaluated by the motion optimizer. From this evaluation, the hypothesis is updated to better estimate the contact state sequence. Specifically, we solve the contact optimization as a shortest path problem of a graph, and develop an algorithm that iteratively learns the nodal costs of the graph, such that the optimal path in the graph (i.e., the contact state sequence) due to the nodal costs leads to the creation of an optimal motion with minimum cost.

In summary, the major contributions of our method are a novel, efficient motion optimizer based on projective dynamics and a novel contact optimizer that discovers an optimal contact state sequence that enables the motion optimizer to generate a plausible motion that best follows the given reference motion.

Our method can be applied to a wide range of motions, such as walking and running, that have sparse contacts with the environment, multi-contact motions, such as standing up from a lying pose, and complex motions, such as cartwheeling. In addition, computation modules are developed to support parallel computing, and thus motion correction is achieved within a reasonable time duration as reported in Sec. 4.

The remainder of this paper is organized as follows. After reviewing related studies in Sec. 2, we present our motion correction method in Sec. 3. Section 4 reports our experiments and analyzes results. Section 5 concludes the paper after discussing limitations of our method and future research directions.

2 Related Work

Our work is related to research on motion editing, dynamic motion filtering and multi-contact motion generation. These research topics have been investigated intensively for decades. In the following, we review previous studies that are closely related to our work.

2.1 Geometric Motion Editing

Motion editing with geometric constraints is a common strategy for modeling interaction with the environment [7] and post-processing movement [1]. A motion graph-based approach has been shown to create natural motions that realize varied contact targets [8]. Geometric constraints can define not only the spatial relationship between the environment and a character, but also the relationship between a characters' joints [9] and among multiple characters [10], [11]. In many methods, a discrete Laplacian deformation [12] is used to preserve details of the source motion while the overall movement is modified according to the given constraints. In this paper, we represent a motion as a mesh that represents the spatial and temporal relationship of joints, and thus the technique of minimizing Laplacian energy plays a central role in preserving the style of the reference motion.

2.2 Dynamic Motion Filtering

Researchers have developed various methods, e.g., [13], [14], to modify motion cpature data in order to obtain a physically correct motion.

A popular approach for this is to develop controllers that drive a physics-based character to track motion capture data. A proportional-derivative (PD) controller is often used for this purpose, usually in combination with a balancing mechanism for locomotive movements. To enhance the robustness against perturbations, control parameters need be tuned, either manually [15] or by learning [16]. Since a PD controller is effective only locally, the global-state feedback policy has been learned to perform wider ranges of motions. [17]. The robustness of a PD controller can also be enhanced by modulating the reference trajectory [18]. Besides the PD control, optimal control schemes have also been successfully used to track motion capture data [19]. Another effective approach is to involve dynamics equations of a character in an optimization to solve for the joint torques and accelerations [14], [20]. Momentum has been used for correcting physical plausibility of motion capture data [21]. Involving inverse dynamics analysis and motion priors in motion capture makes it possible to use sparse sensor set yet achieve high accuracy in motion capture for a wide range of dynamic movements [22].

If underlying physical and control parameters are discovered, these can be used to create similar motions in a different environment, create different motions, and change character sizes. Liu et al. [23] developed an inverse space-time optimization to estimate detailed physical and control-related parameters of human subjects. Fang and Pollard [24] developed a set of objective functions and constraints that lead to linear time analytic derivatives, which allow for faster optimization. Other studies [25], [26] have extracted essential movement characteristics with respect to reduced physical models. Dynamic motion filtering has been studied in the robotics field as well in order to retarget human motion to robots [27], [28], [29].

Most previous studies on dynamic motion filtering have focused on creating physically correct motion and have assumed that the desired contact states, i.e., identification of body parts that should be in contact with the environment, are already known from the reference motion data. This is a reasonable assumption when dealing with motions that have small number of contacts, such as locomotion, but may not hold for multi-contact motions, such as stand-up. By contrast, our method discovers both the physically plausible motion and the contact states from a reference motion.

2.3 Multi-Contact Motion Generation

Synthesizing physically plausible motion that involves multiple contacts with the environment is a challenging problem. As the number of contactable body parts increases, the number of possible contact states from which a motion

generator may need to select an appropriate contact state increases exponentially [30], [31].

Liu et al. [4] developed a controller learning scheme that is able to create multi-contact motions, such as rolling, by combining the strategic sampling of control parameters and forward dynamics to evaluate the samples. Their following work [17] composed controllers for which the parameters are learned from a random walk through the motion capture data. With these learned controllers, they constructed a control graph that enables the motion phases to transition smoothly.

Mordatch et al. [32] synthesized motions involving multiple contacts with the environment or with other characters under the framework of space time optimization, without using reference motion data. To guide faster convergence to the optimum, they introduced an auxiliary variable to deal with the discontinuity incurred from contact transitions. The auxiliary variable balances the physical validity and the convergence to the solution. Their method inspired our contact optimizer in that the auxiliary variable serves as a contact indicator or a contact probability. Similarly, our contact state sequence indicates the contact of candidate body parts. The differences are that our contact state sequence is composed of binary variables whereas the auxiliary variable is continuous, and our method is to find a proper contact set underlying the motion capture data instead of generating a task-based motion.

Posa et al. [33] developed a direct trajectory planning method that resolves contact constraint forces while simultaneously optimizing a trajectory that satisfies the complementarity constraints due to contact.

Due to the high dimensions of the solution space, multi-contact motion planning is generally solved via a sampling-based approach, in which a set of candidate contact points are sampled on the environment and the reachability of the contact points are examined [34]. In order to efficiently find valid contact points and associated natural motions, motion capture data have been utilized for multi-contact motion planning, in the form of simple sets of motion clips [35], transition graphs of contact configurations [36], and contact graphs with feasibility predictors [37]. Rather than focusing on finding optimal contact positions from the environment, our work tackles the problem of determining the optimal contact state for significantly increased number of candidate body parts. Regarding contact positions, we simply select the closest point from a contacting body part as the target contact position under the assumption that the motion and the environment can be roughly aligned by the user.

3 METHODS

Figure 1 describes our framework. Our system takes a captured motion as input and outputs motion optimized with respect to the given geometric and physical properties of a character and its contact information. A motion is represented with 3D mesh vertices v, which identify positions of every joint during the motion. The motion optimizer produces a physically plausible motion satisfying a given contact state sequence $\mathbf{C}$. The contact state sequence represents the contact state at each time frame t as the list of body parts in contact with the environment. The error $\mathbf{e}$

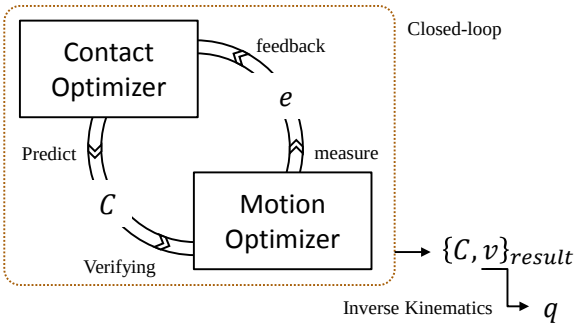


Fig. 1: Overview of our framework.

determined by the motion optimizer represents the quality of the produced motion; this information is fed to the contact optimizer, which can then generate a better contact state sequence. The contact optimizer generates the contact state sequence $\mathbf{C}$ by solving the minimum cost path problem of a graph that represents all possible contact state sequences. In order to generate an optimal contact state sequence that will lead to the optimal motion with minimum error $\mathbf{e}$, the nodal costs of the graph are learned iteratively in a closed loop by hypothesizing a contact state sequence and verifying it with the motion optimizer. Since our motion optimizer works on joint positions, we finally solve the inverse kinematics to obtain the character animation.

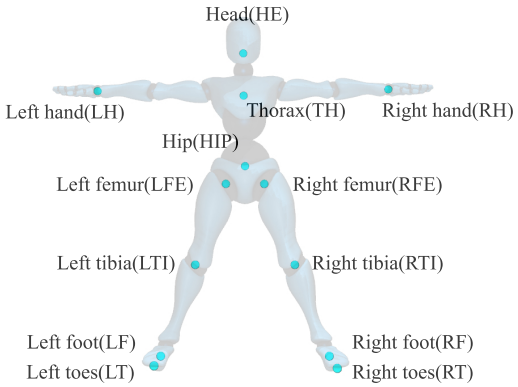


Fig. 2: Contactable joints of our character.

Character Model and Motion Representation

The character model used for our experiments has 31 joints and 62 degrees of freedom, same as the one in the CMU motion database. It is more reasonable to use the representative subset of the joints for finding contact sequence, we select 13 contactable joints as shown in Figure 2. The model has the mass of 70 kg in total, which is distributed over the body according to average human body ratio. Since our objective function terms are defined in the world space, such as the position of the contact with the nearest part of the environment or the acceleration of the center of mass, we use joint positions instead of joint angles for motion editing. Let us represent a character's pose with $|v_t|$ vertices ($v_t \in \mathbb{R}^{J\times 3}$ where J is the number of joints). A motion with T frames consists of $|v_t| \times T$ vertices and the motion vertices are denoted as the variable v ($\in \mathbb{R}^{JT\times 3}$) .

We construct a mesh-like structure by connecting vertices with two types of edges as shown in Fig. 3; One is the bone edge that connects a child joint v^c to its parent v^p at the same frame t, and the other is the joint trajectory edge that connects joint positions between adjacent frames.After converting joint positions to a mesh, the solver can perform motion editing with appropriately defined constraints.

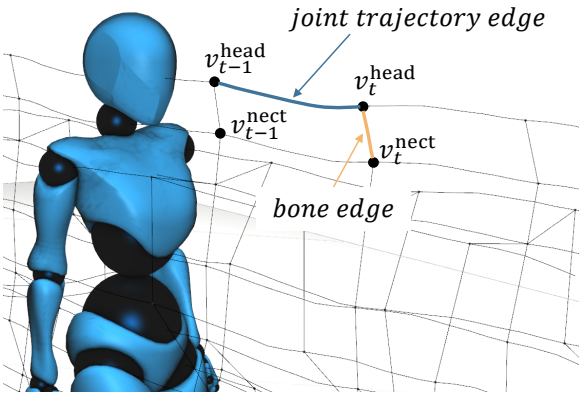


Fig. 3: Mesh representation of motion.

3.1 Motion Optimizer

We briefly review the local/global alternating solver of [6] here. By introducing auxiliary variables p_c on each constraint c, the local step projects p_c onto the constraint manifold such that the potential is minimized. A general form of the constraint potential W is written as

$$W(p_c) = w_c \|A_c v - p_c\|^2 + \eta(p_c), \tag{1}$$

where p_c is an auxiliary variable for constraint c, and $\eta(p_c)$ is an indicator function that returns a value of zero if p_c is the set of the desired configuration and returns $+\infty$ otherwise. The matrix A_c represents the relationship of joint positions and its specific structure is defined by the constraint. In the local step, since the global variable v is treated as constant, p_c is projected to the constraints packed in $\eta(p_c)$ while minimizing the potential function (1). In addition, every constraint of potential function is independent of each other, which allows parallelization of a local step. After projecting every auxiliary variable, a global step merges them into a global variable v in the least-squares manner.

$$\arg\min_v \sum_c w_c \|A_c v - p_c\|^2 \tag{2}$$

This equation is the same as

$$\arg\min_v \|Av - p\|^2, \tag{3}$$

where the matrix A is a stack of all $w_c A_c$ by row, and p is also a stack of all $w_c p_c$. The global step updates the global variable v (the joint vertices) by minimizing (3) while keeping the auxiliary variable p fixed. Solving the global step includes an expensive operation of computing the inverse of A, but this can be pre-computed because A is constant while the contact state sequence remains unchanged. When we need to change a contact configuration, a multiple rank update method can be applied, which only requires a few milliseconds, if the number of updated contacts is relatively smaller than the size of A. If not, the matrix A is re-computed as in the initializing step. The alternating solver has strong advantages of simplicity and robustness because it decouples a potential function into the Euclidean distance and constraint parts; this encapsulates the non-linearity of the constraint with each auxiliary variable.

3.2 Potentials and Projection Operators

In this section, we explain the potential functions designed for the motion refinement problem and their projection operators. As explained above, an input motion has a mesh-like structure as shown in Figure 3, and thus the potential functions work on the joint vertices. To construct a potential function, the matrix A_c needs to be configured with respect to the input, that is the contact state sequence $\mathbf{C}$. For example, the ground constraint, which causes a joint to plant itself on the ground, is configured according to each contact state specified in $\mathbf{C}$. Most of our potential functions do not modify the input $\mathbf{C}$ and thus, the matrix A remains as a constant and it helps the optimization converge in a fast speed with pre-computed inverse matrix A^{-1}. One exception is the input-independent penetration potential (Sec. 3.2.6), which directly modifies A when v violates the penetration constraint. Even in this case, A^{-1} can be updated efficiently by the multiple rank update method because the modified number of rows is relatively small.

3.2.1 Bone Length

Joint angle representation naturally satisfies the bone length constraints, but the world space operation on the joint angle coordinates requires repetitive computation of the Jacobian matrix, which is an expensive operation. Operating on joint positions v is more intuitive for dealing with the world space constraints, but we need to enforce the bone length preservation explicitly. In our motion optimizer, the bone-length constraint is treated as a soft constraint that allows a small violation, which helps the solver converge by compromising conflicts with other constraints. The bone-length constraint specifies the distance between vertices in bone edges,

$$W_{bl}(p_{bl}, l) = w_{bl} \|A_{bl} v - P_{bl}(v, l)\|^2 + \eta(p_{bl}(v)), \tag{4}$$

$$P_{bl}(v, l) = l \frac{v_c - v_p}{\|v_c - v_p\|}, \tag{5}$$

where l is the original length of the bone corresponding to v^c and v^p, and $p_{bl}(v)$ checks the violation of the bone length. The matrix A_{bl} is constructed such that $A_{bl}\, v$ outputs $v^c - v^p$ for all joint and time. The projection operator for the bone-length P_{bl} projects the vector $v^c - v^p$ on the closest constraint manifold, leading to a vector with the same direction but length scaled to l. Note that the potential function forms differential coordinates. Since the differential coordinates are invariant to translation, the convergence rate does not degrade due to the translational movement.

3.2.2 Joint Trajectory

The joint trajectory potential encourages the motion to follow the reference motion style with respect to the joint trajectory edges along the time. To preserve the style of the motion despite the changes in the contacted positions, we define this potential using the Laplacian energy minimization [12]. The potential function is

$$W_j(\delta_j^{ref}) = w_j||A_j v - \delta_j^{ref}||^2, \tag{6}$$

where δ_j^{ref} represents the differential coordinates of the joint trajectory edges of the reference motion, and A_j is a discrete Laplacian matrix that results in the differential coordinates of the multiplied vertices v. Hence, $A_j\, v_{ref}$ results in δ_j^{ref} where v_{ref} are the joint vertices of the reference motion. More specifically, A_j is a stacked matrix of the first order discrete Laplacian matrices corresponding each joint and time; i.e.

$$A_j(t) = \begin{pmatrix} A_j(1,t) \\ \vdots \\ A_j(B,t) \end{pmatrix} \tag{7}$$

$$A_j(b,t)\; v = \frac{1}{2}(v_{t-1}^b + v_{t+1}^b) - v_t^b. \tag{8}$$

Since W_j plays the role of suggesting the desired motion style, the violation indicator function, $\eta(p_j)$, is removed from the potential (Eq. 6), and thus it has a high chance of being violated. In addition, to avoid unnecessary conflict with the other constraints, W_j is not applied to the contacted joints, which are included in the contact state sequence $\mathbf{C}$. Instead, these joints are handled by non-slip and ground contact constraint, as explained next.

3.2.3 Non-Slip and Ground Contact

This potential is applied only to the joints belonging to the contact state sequence. Let us assume $(\mathsf{c}, t) \in \mathbf{C}$ where c is a contact configuration, i.e., $\mathsf{c} = \{b\}$ for the contacted body parts b. Then, the set of body parts $\{b\}$ should be planted on the ground and not move from time $t-1$ to t. To this end, we define a non-slip potential and a ground contact potential to prevent a contacted body part from moving or floating above the ground.

$$W_{sg}(\mathsf{c}, t) = w_{sg}\|A_{sg}(\mathsf{c}, t)\; v - P_{sg}(\mathsf{c}, v)\|^2 + \eta(p_{sg}(\mathsf{c})), \tag{9}$$

where $p_{sg}(\mathsf{c})$ checks whether the contacted body parts are floating or not. For the non-slip potential, $A_{sg}(\mathsf{c}, t)$ gives the vector of edge from previous joint v_{t-1}^b to current v_t^b, and the projection operator P_{sg} causes the magnitude of the vector to become zero. For the ground contact potential, P_{sg} projects v_t^b onto the ground by setting its height to zero. If contact is to be made against some complex environment, the closest point is used for the projection operator. The ground potential only works on the set of contacted body parts $\{b\}$, and does not prevent other body parts of the character from penetrating into the environment. The penetration will be dealt with by the penetration potential introduced in Sec. 3.2.6.

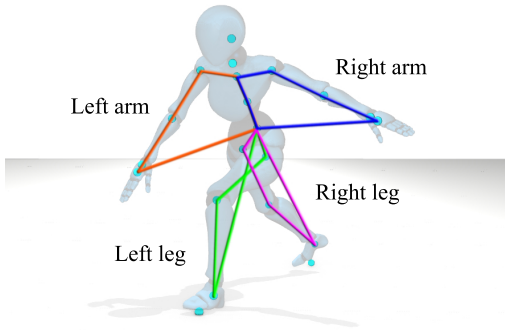


Fig. 4: Each limb mesh is visualized in different colors. Only vertices and edges corresponding to one time frame are shown.

3.2.4 Reference Pose

Similar to the joint trajectory potential, the reference pose potential, with a small weight value, acts as a guide term to help preserve the spatial relationship of bone edges in each time frame. Many previous studies have defined the pose difference in terms of joint angles but, in our case, the reference pose cost needs to be evaluated with respect to the joint positions similar to the cases of the interactive mesh [9] or the Laplacian deformation energy [12]. After a number of trials to find a suitable pose cost that efficiently preserves the pose style while not significantly affecting the other potentials, we determined the reference pose potential as follows. We first define the limb meshes by separating sub-meshes corresponding to each limb of the character and connecting edges from terminal vertices to the root (Fig. 4), and then apply a discrete Laplacian to each limb mesh.

$$W_{pose} = w_{pose}\left\|A_{pose}v - A_{pose}v_{ref}\right\|^2. \tag{10}$$

In this equation, A_{pose} is a discrete Laplacian matrix that results in the differential coordinates of limb meshes when multiplied by the joint positions. Similar to A_j of the joint trajectory potential, A_{pose} is the stacked matrix of the Laplacian matrix corresponding to each limb. The projection operator of this potential is $A_{pose}v_{ref}$, which is constant.

The structure of our body mesh in Fig. 4 is different from denser body meshes, which directly connect end-effectors, used in previous work such as [9]. Denser mesh preserves the overall pose better, but we found that it may cause unintended consequences. For example, when a foot identified to be the contact state is moved towards to the ground, it may change the contact state of other end-effectors that are already in the desired state. However, there can be cases that our sparse body mesh may lose the semantics of input motion, e.g., two holding hands can be separated as the optimizer ignores the distance between two hands. In general, the structure of an ideal body mesh depends on the content of the motion.

3.2.5 Dynamic Balance

In order to check the state of dynamic balance, the projection operator examines whether the acceleration of the center of mass (CoM) is admissible given the set of contact points and friction cones. The global dynamics with respect to the rates

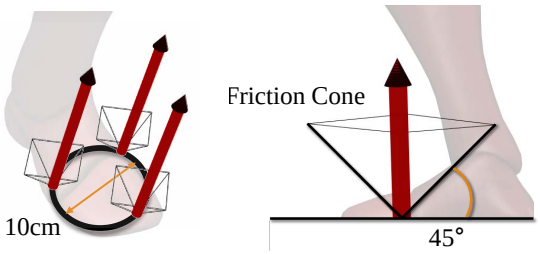
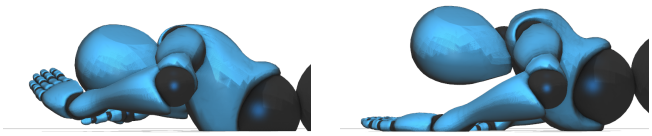


Fig. 5: Each joint has 3 (i.e., minimal number for balance maintenance) contact points, which are distributed around the joint with an appropriate radius (5cm in our examples). An approximated friction cone at a contact point of the right toes (RT) joint is shown in wireframe.



(a) Reference motion (b) Refined motion

Fig. 6: Penetration constraint pushes the body up to the surface; inverse kinematics aligns the normal of the hands to that of the ground.

of change of linear momentum l and angular momentum k is

$$\dot{l} = \sum_i f_i + mg \tag{11}$$

$$\dot{k} = \sum_i r_i \times f_i \tag{12}$$

In these equations, f_i is the ground reaction force (GRF) acting on each contact point i, m is the mass of the character, g is the gravitational acceleration, and r_i is the vector from CoM to the contact point i. Therefore, we can verify the admissibility of the dynamic balance by checking whether any set of f_i can satisfy (11) and (12). If not, the projection operator corrects the value of $\dot{l}$ so that the external forces, i.e., GRFs and gravity, can support the acceleration of CoM. To obtain a corrected $\dot{l}$, we compute GRFs that make the admissible momentum rate change as close as possible to the current one, as the solution of the projection operator. This equation is written as

$$\begin{aligned} \underset{\beta\in\mathbb{R}^{4N},\tau\in\mathbb{R}^{3K}}{\operatorname{argmin}} \quad & w_l\|\dot{l}(v,t) - \sum_i^N V\beta_i - mg\|^2 + \\ & w_k\|\dot{k}(v,t) - \sum_i^N r_i \times V\beta_i - \sum_i^{|\mathsf{c}|}\tau_i\|^2 + \\ & w_\beta\|\beta\|^2 + w_\tau\|\tau\|^2 \\ s.t. \quad & V_y\beta_i \le f_{max}, \\ & 0 \le \beta, \\ & \|\tau\| \le \tau_{bound} \end{aligned} \tag{13}$$

where N is the number of contacts, $|\mathsf{c}|$ is the number of contacted joints, and $\tau = [\tau_1 \cdots \tau_{|\mathsf{c}|}]^T$ represents small torque allowed to act on the contacted joints. We found that (12) can be satisfied imperfectly because the contact configuration might not be able to generate the necessary torque, and that introducing a small τ helps the estimation of the GRF be more stable. The maximum magnitude of τ is bounded by setting $\tau_{bound} = 10Nm$. The GRF f is represented using four basis vectors V that approximate the friction cone, i.e., $f_i = V\beta_i$, where $\beta \ge 0$ because the GRF must be unilateral in nature (Fig. 5). The basis vectors V are rotated to have the same normal with the environment. The parameter f_{max} constrains the maximum value of normal component of GRF, and its effect will be discussed in Sec. 4.

The linear and angular momentum rate change functions $\dot{l}(v,t)$ and $\dot{k}(v,t)$ over current joint positions v are computed as follows.

$$\dot{l}(v,t) = \frac{m\,\ddot{\chi}_t}{\Delta t^2} \tag{14}$$

$$\dot{k}(v,t) = \sum_b^B m_b(r_t^b \times \ddot{\chi}_t^b + \dot{r}_t^b \times \dot{\chi}_t^b) \tag{15}$$

where χ is the position of the whole body CoM, and m^b and χ^b represent the mass and the CoM of body part b. Because the CoM of a body part cannot be determined from the joint vertex v exactly, we assume the body CoM lying on the middle point of the child and parent vertex. The whole body CoM χ is then computed as the sum of each body CoM χ^b multiplied by its mass m^b, and $r^b = \chi^b - \chi$. The duration of time step Δt is set to $1/30$ seconds in our examples. In Eq. (15), $\dot{k}(v,t)$ computes the rate change of the angular momentum by approximating each part's inertia as a point mass at its CoM. For computing acceleration and velocity on the joint vertex, we discretized these continuous quantity with the previous frame v_{t-1} and the next v_{t+1} such as

$$\begin{aligned} \dot{v}^b &= v_{t+1}^b - v_t^b \\ \ddot{v}^b &= v_{t+1}^b - 2v_t^b + v_{t-1}^b. \end{aligned} \tag{16}$$

With the optimized β^*, the admissible momentum rate change can be easily computed as

$$\dot{l}_{adm}(\mathsf{c},v,t) = \sum_i^N V\beta_i^* + mg. \tag{17}$$

Then the dynamic balance potential is computed as

$$W_l = w_l\|A_l v - \dot{l}_{adm}(\mathsf{c},v,t)\|^2 + \eta(p_{adm}(\dot{l})), \tag{18}$$

where A_l is constructed to output $\dot{l}(v,t)$ in Eq. (14) when multiplied by v, and $p_{adm}(\dot{l})$ verifies whether $\dot{l}$ is admissible.

3.2.6 Penetration

Penetration avoidance potential is added to prevent a character body from penetrating into the environment; it is stated as follows.

$$W_{pene} = w_{pene}\|Sv - P_{pene}(v)\|^2 + \eta(p_{out}(v)), \tag{19}$$

where $p_{out}(v)$ checks the penetration. This equation is similar to the ground constraint Eq. (9) except that this applies

to all joints. The matrix S is a selection matrix that indicates which body part is in collision with the environment. The operator P_{pene} projects a penetrating joint vertex v_b onto the closest boundary of the environment. For a skinned character, we allow a joint vertex to have a bounding sphere, and the penetration detection and projection are performed with respect to the sphere (Fig. 6).

With these potentials designed for the motion editing problem, the motion optimizer produces a motion that satisfies a given contact state sequence while following the motion style of recorded data. Every potential has their projected positions and are merged into a global variable while the importance of each constraint is controlled by the weights. Table 1 shows the weights used in our experiments.

TABLE 1: Weights for each potential.

Potential	Weight
Bone-length	1.0
Joint trajectory	1.0
Non-slip & ground contact	1.0
Reference pose	0.1
Dynamic balance	$\Delta t^2/m$
Penetration	0.5

3.3 Contact Optimizer

The contact state sequence $\mathbf{C}$ specifies which body parts should contact the environment, and thus directly controls the non-slip and ground contact potentials of the motion optimizer. Given $\mathbf{C}$, the motion optimizer then finds a motion that satisfies dynamic balance and follows reference motion as much as possible. Hence, the quality of the produced motion greatly depends on the quality of $\mathbf{C}$. We measure the quality of a motion in terms of some selected potentials of the motion optimizer, i.e., joint trajectory, reference pose, and dynamic balance. The first and second potentials measure the deviation from the reference motion, and the third one evaluates how well $\mathbf{C}$ support the motion. With these potentials, we aim to achieve an optimal $\mathbf{C}$ that will produce the lowest sum of the potentials.

$$\mathbf{e} = \sum_{i}^{P} \|A_i v - p_i\|^2, \tag{20}$$

where P includes only the terms for joint trajectory, reference pose, and dynamic balance. To search for an optimal $\mathbf{C}$, we represent the space of $\mathbf{C}$ with the directed acyclic graph (DAG) $\mathbf{G}$, in which each node represents a contact configuration c at a certain time frame t as shown in Fig. 7. In graph $\mathbf{G}$, dubbed the *contact state graph* in this paper, we list all possible nodes, and thus, the total number of nodes amounts to $2^B \times T$ where B is the number of body parts that can contact the environment. The edges completely connect all nodes at $t-1$ to all node at t. Then the contact optimization can be formulated as the optimal path finding problem in $\mathbf{G}$.

If the cost of a path can be decomposed into the path-independent costs of nodes and edges in the path, the optimal path can be obtained very efficiently because then

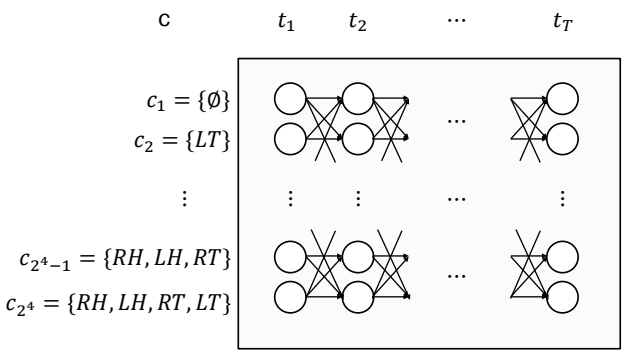


Fig. 7: A contact state graph $\mathbf{G}$ represents all possible contact states and transitions. The candidates of contact body is $\{RH, LH, RT, LT\}$, and thus the total number of possible c is 2^4.

we only need to compute the costs once for every node and edge. In our problem, however, the nodal cost depends on the path because the resulting pose at t may depend on a particular contact configurations at the previous time frames.

To cope with this challenge, we developed a novel contact optimizer that finds the optimal path in a reasonable time. Computational efficiency and reasonable optimality are achieved by following components. First, by carefully designing the path-invariant, approximate cost of the nodes in $\mathbf{G}$, we make it possible to use efficient shortest path algorithms for DAG. To this end, the cost of each node is computed purely based on the reference pose corresponding to the node, independent of any path. Specifically, the elements of cost, i.e., pose style, slipping, and dynamics, are computed from a pose created by modifying the reference pose. The cost computed this way is of course different from the true cost output by the motion optimizer. Our goal, rather than accurately predicting the error of the motion optimizer for the optimal path, is to determine the nodal costs such that they are good enough to find the optimal contact state sequence.

Second, the nodal costs are adjusted to produce the optimal path that will minimize the error of the motion optimizer. A nodal cost is determined by the weighted average of the sub-costs, as will be discussed shortly, where the weights control the relative importance of the sub-costs, and we approach the problem of finding the optimal nodal costs as a problem of finding the right weights for the sub-costs. This strategy is based on our experience that appropriate weights that produce natural motions vary depending on the motion. By iterating hypothesizing and correcting the statistics of the weights from the simulations, we obtain the optimal weights.

3.4 Elements of Graph Costs

This section explains the elements of the nodal cost and the edge cost in contact state graph $\mathbf{G}$. Under the assumption that the optimized motion retains the style of the reference motion, to compute the nodal cost, we create a pose by minimally modifying the reference pose to satisfy the contact

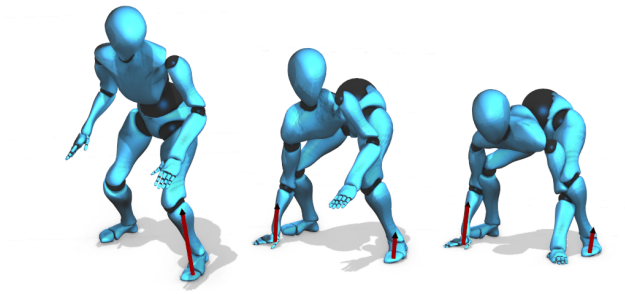
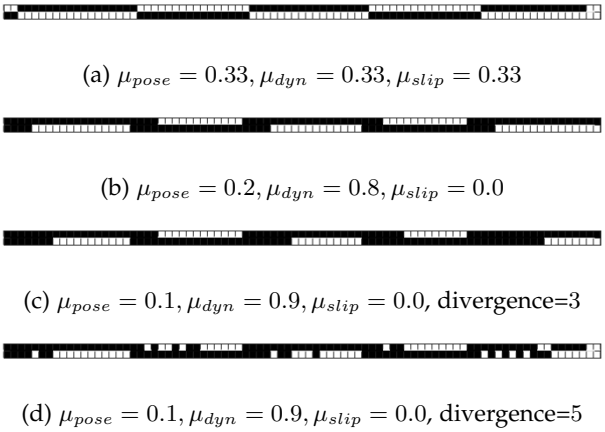


Fig. 8: Poses that correspond to instances of c, $\{RT, LT\}$ (left), $\{RH, RT, LT\}$ (middle), and $\{RH, LH, RT, LT\}$ (right).



(a) $\mu_{pose} = 0.33, \mu_{dyn} = 0.33, \mu_{slip} = 0.33$

(b) $\mu_{pose} = 0.2, \mu_{dyn} = 0.8, \mu_{slip} = 0.0$

(c) $\mu_{pose} = 0.1, \mu_{dyn} = 0.9, \mu_{slip} = 0.0$, divergence=3

(d) $\mu_{pose} = 0.1, \mu_{dyn} = 0.9, \mu_{slip} = 0.0$, divergence=5

Fig. 9: Contact state sequences of a walking motion with different weights. In this example, the weights are kept constant over time. The two rows denote $\{RT$(first row), LT(second row)$\}$ and a black cell at t mean the body part is in contact with the environment at that time.

state specified by the node. The nodal cost has the form of the weighted average of sub-costs:

$$\mu_{pose}(t)L_{pose}(\mathsf{c},t) + \mu_{dyn}(t)L_{dyn}(\mathsf{c},t) + \mu_{slip}(t)L_{slip}(\mathsf{c},t)$$

where

$$\mu_{pose}(t) + \mu_{dyn}(t) + \mu_{slip}(t) = 1. \quad (21)$$

The cost of a node $(\mathsf{c},t) \in \mathbf{G}$ represents the approximated error due to the contact state, and the weight value μ controls the importance of the sub-cost for reducing the motion error $\mathbf{e}$. The individual sub-cost functions are explained next.

3.4.1 Pose Style

This cost measures the deformation energy with the limb mesh discussed in Sec. 3.2.4. To evaluate the energy of a given contact set (c,t), we make the joints as marked contacted by c plant the ground while minimizing the deformation error. The pose style cost is then defined as

$$L_{pose}(\mathsf{c},t) = \|A_{pose}(t)\, v^*(\mathsf{c},t) - \delta_{ref}(t)\|^2 \quad (22)$$

where $A_{pose}(t)$ are from Sec. 3.2.4, which is a submatrix corresponding to time frame t. The vector $v^*(\mathsf{c},t)$ approximates the joint positions at time t given contacts c as

$$v^*(\mathsf{c},t) = \underset{v}{\operatorname{argmin}} \left|\left| \begin{pmatrix} A_{pose}(t) \\ A_{sg}(\mathsf{c},t) \\ \epsilon I \end{pmatrix} v - \begin{pmatrix} \delta_{ref}(t) \\ P_{sg}(\mathsf{c}, v_{ref}) \\ \epsilon v_{ref} \end{pmatrix} \right|\right|^2 \quad (23)$$

This function predicts joint positions when the contact configuration c is given, and it enables L_{pose} to approximate the deformation cost of the limb mesh. In Eq. 23, the top row encourages preserving the differential coordinate of the reference pose, δ_{ref}, and the second row makes the joints in the contact state c to plant to the positions P_{sg}. The bottom row guides the joint position to the reference where ϵ makes this term much weaker than other terms. Figure 8 shows three different poses obtained from the same reference pose with different contact states specified. Note that Eq. 23 has no dependence on the state of v, and thus all nodal costs can be computed without the motion optimization.

3.4.2 Dynamics

Evaluating the dynamic plausibility from the obtained pose $f_v(\mathsf{c},t)$ above is not path invariant because the rate of change of momentum depends on the nodes at previous and posterior time steps. Instead, we evaluate whether a given contact state c can generate the linear momentum rate change calculated from the reference motion. Addressing the dynamics cost this way makes the dynamic admissibility independent of the neighboring contact states. The cost is then defined as

$$L_{dyn}(\mathsf{c},t) = \|\dot{l}_{adm}(\mathsf{c}, v_{ref}, t) - \dot{l}(v_{ref}, t)\|^2, \quad (24)$$

where $\dot{l}_{adm}(\mathsf{c}, v_{ref}, t)$ returns an admissible $\dot{l}$ given contacts c, and $\dot{l}(v_{ref}, t)$ is $\dot{l}$ of the original motion at time t. This cost is similar to that of the balance constraint in Sec. 3.2.5 except that CoM location is computed from $f_v(\mathsf{c},t)$ of the pose style.

3.4.3 Slipping

This term penalizes a contact state c that includes joints with a large traveling distance.

$$L_{slip}(\mathsf{c},t) = \sum_{b}^{\mathsf{c}} \|v_{ref}(b,t-1) - v_{ref}(b,t)\|^2 \quad (25)$$

This term is related to the joint trajectory potential of the motion optimizer in that the joint trajectory of a fast moving joint, if included in c, must deform heavily to shrink to a single contact point by the motion optimizer.

Until now we have discussed sub-costs for the nodal cost. Note that our sub-cost terms do not use heuristics, such as a threshold on the height or speed of a joint, to determine the contact of the joint. Such heuristics are sensitive to noise and often fail to find correct contacts for complex motions.

Normalization

Each element of nodal cost has a different scale; for example, the dynamic cost distributes over 10^6 but the pose cost distributes under 10. This may cause a result with severely biased weights, which could lead to failure of the optimizer. Thus, we normalize them within the range of [0,1] so that the weight learning will not be affected by the scale. Because a nodal cost is independent of temporally adjacent nodes,

normalization is applied to the nodes in the same time t by dividing each sub-cost with the maximum value at the time.

3.4.4 Contact State Change

By penalizing the contact state changes across time, we can reduce unnecessary changes of contact state and thereby obtain temporally more consistent motion. This is achieved by applying the edge cost w_{jump} to the edges connecting two different contact states. Then, the total edge cost is computed as the number of contact state transitions:

$$L_{jump}(\mathbf{C}) = \sum^{U} w_{jump}, \tag{26}$$

where U is the number of transitions. The weight for this cost is controlled by the concept of diversity explained in Sec. 3.5.1.

3.5 Weight Optimization

Having computed the values of the sub-costs of each node, the task of learning the nodal cost is reduced to finding the appropriate weights $\mu(t) \in \mathbb{R}^3$ over time. Figure 9 shows how $\mathbf{C}$ is influenced by different weights and divergence value (Sec. 3.5.1). For example, Fig. 9b has higher weight for the dynamic cost than Fig. 9a, and thus it has a stronger tendency to keep contacts with the ground. Similarly, Fig. 9c, which has higher weight for the dynamic cost, shows longer contact time than Fig. 9b.

We assume that the optimal weights are continuous functions over time, and model the weights as simple piece-wise linear functions, with control weights are defined at key frames distributed in equal interval over time. Therefore, the dimension of the weight space is $2 \times K$ where K denotes the number of key frames. To find the optimal control weights, we adopt the sampling approach of [38], which is modified to enforce the sum to unity constraint (Eq. 21). Our modified algorithm is explained next.

First, initial samples $s \in \mathbb{R}^{2K}$ are picked uniformly over the whole variable space. In our experiment, the number of initial samples is 100, which is 10% of total samples. Each sample corresponds to a weight function. By applying sampled weights to the nodal costs of the graph $\mathbf{G}$, the optimal path $\mathbf{C}$ can be searched and an error $\mathbf{e}$ of $\mathbf{C}$ is evaluated by the motion optimizer.

Second, the optimizer selects the most probable sample, with its probability determined by its error $\mathbf{e}$ and volume. The volume of a sample is measured as the volume of the Voronoi cell containing the sample. If a sample has larger volume than others, the density around that sample is lower and thus more exploration is needed. In addition, a sample with a lower error indicates that good samples are in close proximity. In order to pick a sample that has sparser density and lower error, we set the probability of a sample proportional to its volume divided by the error. The reason why we use Voronoi cells instead of the kd-tree as in [38] is that the space of our samples is not a hypercube due to the constraint of Eq. 21. Computation of the approximate Voronoi cell volume can be performed fast enough.

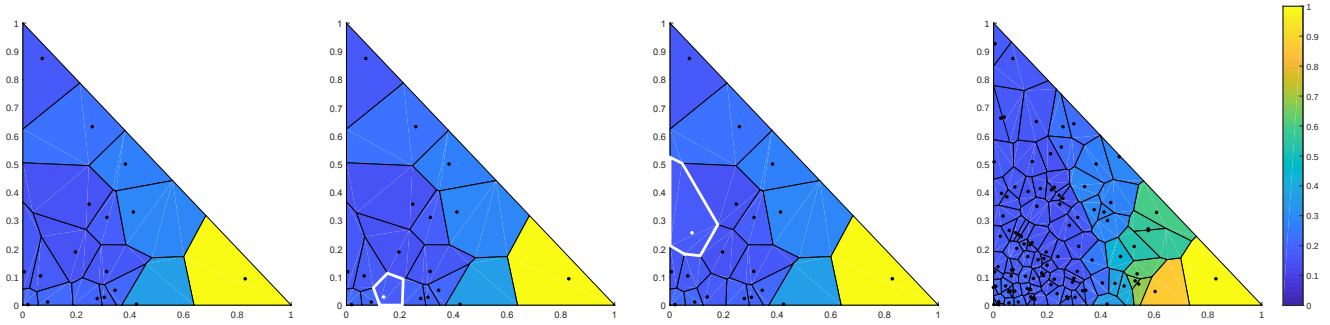
Next, a new sample is located in proximity to the selected sample. Specifically, the location is sampled from the multivariate Gaussian, with its mean set to the selected sample. Its covariance matrix is set to a diagonal matrix, with each diagonal element being the edge length of the bounding box of the Voronoi cell. Setting the variance to the length of the bounding box makes Gaussian function large enough to cover the cell, and it helps prevent falling in local minima.

Lastly, with the new sample, the volume and the error is updated for the next iteration. The optimization terminates when the iteration number reaches the predefined number. Figure 10 shows a weight sampling process at one key frame (K=1). Each axis represents μ_{pose} and μ_{dyn} respectively, and the diagonal boundary is the line where $\mu_{slip} = 0$ because of the sum to unity constraint. The volume of the cell is identical to the volume of the convex hull that resides in the sample space.

Algorithm 1 Optimizing weight with adaptive sampling

```
1:  L_{dyn,pose,sli} : Matrices of nodal sub-costs.
2:  M : Number of initial samples.
3:
4:  function EvaluateSample(s)
5:      G ← 0 ∈ R^{2^B × T}
6:      for t = 1 ··· T do
7:          i ← An index of segments of time t.
8:          μ_{dyn,pose,slip} ← LinearInterpolate(s_i, s_{i+1}, t)
9:          G(t) ← Σ_j^{dyn,pose,slip} μ_j L_j(t)
10:     end for
11:     C ← Search(G)
12:     (e, v*) ← MotionOptimizer(C, v_ref)
13:     return e
14: end function
15:
16: function Optimizing weight
17:     S ← {∅}                                  ▷ The set of samples.
18:     // Initial sampling.
19:     for i = 1...M do
20:         s ← Uniformly draw a sample.
21:         S ← S ∪ s
22:         e_i ← EvaluateSample(s)
23:     end for
24:     V ← ComputeVolume(S)
25:     W = V/e
26:     // Adaptive sampling.
27:     while n(S) < maxSample do
28:         s ← Draw a sample with the probability ∝ W
29:         s' ∼ N(s, LengthOfBoundingBox(s))
30:         S ← S ∪ s'
31:         e_i ← EvaluateSample(s')
32:         V ← ComputeVolume(S)
33:         W = V/e
34:     end while
35:     return the best sample in S.
36: end function
```

Algorithm 1 shows the contact optimization process. The pre-computed sub-nodal costs are stored in the matrices $\mathbf{L}$, where each row lists all candidate contact states. Line 7 retrieves the index of piece-wise linear function that includes time t, and the weights $\mu_{dyn,pose,slip}$ at t are linearly interpolated between the weights $s_i (\in \mathbb{R}^3)$ and s_{i+1}.



(a) 17th iteration (b) 18th iteration (c) 19th iteration (d) 99th iteration

Fig. 10: Voronoi diagram of the sampling procedure for a single key frame ($K = 1$). Only the first two axes for μ_{pose} and μ_{dyn} are shown. Each cell contains a sample, and our adaptive sampler picks a sample according to the volume of cell and the error. In (b) and (c), the white cell indicates a new sample in that iteration. Error values of the motion optimizer are shown in color.

3.5.1 Controlling Diversity

We now explain the method for adjusting the edge weight w_{jump}, which controls the degree of suppression of the contact state transitions. Unlike the nodal weights, the edge weight functions as a constraint and is not included in the weight optimization. If it were included in the weight optimization with the other nodal weights, the edge weight would be optimized to zero so that the minimum error path could be searched without restricting contact state transitions.

To control the edge weight, we introduce a diversity index to measure the complexity of the contact states. The diversity index [39] was originally developed to represent the diversity of a species, and here we use it to measure the diversity of contact states in a contact state sequence. This index has an advantage of providing an intuitive parameter that allows users to control the complexity of the contact states. The diversity of the contact states is defined as

$$D(\mathbf{C}) = \exp(-\sum^{R} p(\mathsf{c}) \ln p(\mathsf{c})), \tag{27}$$

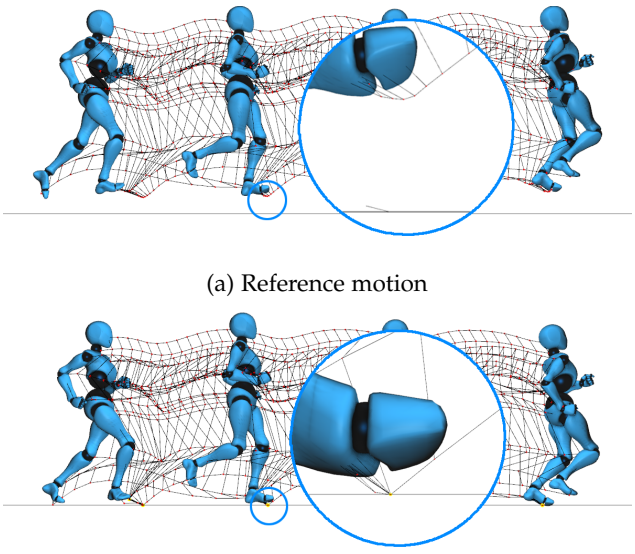
where $p(\mathsf{c})$ denotes the frequency of a contact state c in a given contact state sequence and R denotes the total number of contact state types in the sequence.

Before running the motion optimizer to evaluate the error of the contact state sequence, we measure the diversity of the sequence. If the diversity is higher than a user-specified value (Fig. 9d), which means the contact state sequence contains too many transitions, then w_{jump} is increased to lower the complexity (Fig. 9c). We found that the diversity index is more intuitive to control the contact complexity of a motion than directly controlling the weight value w_{jump}.

3.5.2 Inverse Kinematics

Using the final weights of the graph, we find the optimal contact state sequence, from which we obtain optimized trajectories of joints that are dynamically plausible while being similar to the reference joint trajectories. Using these results, we perform inverse kinematics to compute the joint angles of characters to follow the joint trajectories while minimizing joint accelerations and respecting joint limits. In addition, since only the joint positions of the end-effectors are specified, we need to determine the orientation of the end-effectors, which is done such that the end-effectors have the similar orientation as those of the reference motion while they do not penetrate into the environment. When an end-effector is in contact state, we align its desired normal direction with that of the environment.

4 Experiments and Results



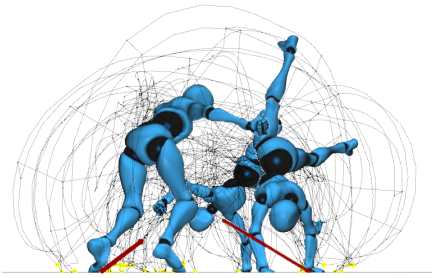
(a) Reference motion

(b) Refined motion

Fig. 11: Results of correcting running motion.

We tested our framework on various scenarios and validated the naturalness and the physical correctness of the generated motions. Tested motion clips were selected from the CMU motion database to have diverse characteristics, such as low and high dynamics, short and long contact duration, and complexity of environment. Selected motions were generally considered to be difficult to analyze or edit. With these motions, our framework produces dynamically filtered motions and corresponding contact information. As

(a) Reference motion



(b) Refined motion

Fig. 12: Results of correcting cart-wheel motion.

pre-process, we apply a Gaussian filter to the joint angle trajectories and downsample the motion data from 120Hz to 30Hz (60Hz for the cartwheel motion). Sometimes, the motion data contain axially twisted joints, and such joints cannot be corrected by our framework as our framework deals only with joint positions. Such artifacts are removed in the inverse kinematics stage, with post-processing to fix unnatural looking motions as detailed in Sec. 3.5.2.

There are several parameters used in the contact optimizer; most of them are insensitive to motions and can be applied uniformly. Weight functions are divided into 5 segments and each of them has $0.5 \sim 1$ sec long. In addition, the number of the sample is set to 100, which is enough to converge for most of examples. On the other hand, the diversity value and the maximum GRF (f_{max} in (13)) at each contact point are adjusted depending on the input motion; these values and the list of contact candidates are shown in Table 2.

During experiments, we found that lower f_{max} shows a tendency that the contact sequence maintains longer contact duration to accelerate the CoM. Conversely, higher f_{max} results in more bouncing motion with shorter contact duration because shorter time is necessary to accelerate the CoM if larger impulse from the environment is allowed. However, if f_{max} is too low to reproduce the acceleration of the reference motion then the dynamic cost (Eq. 24) contains inaccurate values to evaluate $\mathbf{C}$. Reasonable results are obtained if f_{max} is set higher than 200N.

Walking and Running

For walking and running motions on flat ground, both feet and toes are set as contact candidates. The original motion clip (Fig. 11a) has the toes airborne and sliding when they should be planted on the ground, and our method detects appropriate contacts and refines the whole body motion as shown in Fig. 11b.

TABLE 2: Parameters for the contact optimizer.

Experiments	Diversity	f_{max}	Contact Candidates
Walking	15	200	LT, RT, LF, RF
Running	15	500	LT, RT, LF, RF
Standing Up	30	1000	LT, RT, LF, RF, LTI, RTI, LH, RH
Cartwheeling	20	1000	LT, RT, LF, RF, LH, RH
Dancing	20	1000	LT, RT, LF, RF, LH, RH
Rolling	20	200	LT, RT, LH, RH, TH, HE, HIP
Sitting on Chair	20	300	LT, RT, LFE, RFE, LH, RH

Figure 13a shows a result that a level walking motion is applied to a hill environment. All the parameters remain the same as those in the level walking example. Despite the relatively steep slope, the resulting motion is quite reasonable.

Standing Up

Figure 13b shows a result that corrected a standing up motion. The transition from lying to standing includes frequent changes of contact, which is more complex than other example motions. Moreover, it has eight contact candidates and thus this motion requires the highest diversity value among the experiments.

Cartwheel and Dance

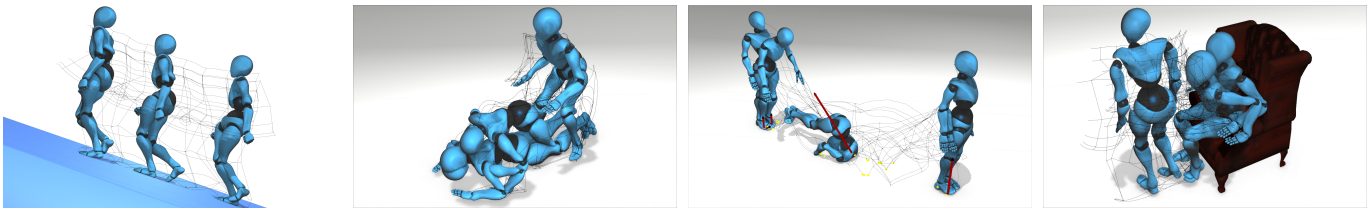
Our next experiment is correcting cartwheel motion as shown in Fig. 12. Because these motions exhibit vigorous momentum rate changes, it is challenging to analyze their dynamics. Moreover, captured poses contain high degrees of artifact due to the difficulty of capturing quickly changing motions. Therefore, we down-sampled the motion at 60Hz, which is twice as dense as that used for other motions, and set a higher value for the diversity.

Rolling

Rolling motion (Fig. 13c) shows that the rate of change of momentum is relatively more stable than those of other acrobatic motions, but a large portion of occluded body causes many artifacts in the motion data. The tested motion clip includes ankle motions unnaturally rotating back and forth and physically impossible standing up motion. These severe artifacts make both the motion and contact optimizations difficult, and also cause the inverse kinematics to produce unnatural motions even if the right contact state sequence is detected. The supplementary video shows that our method improves the naturalness of the input motion significantly but could not fix the artifacts to the satisfactory level.

Sitting on Chair

Figure 13d shows a result that corrects sitting motion on a chair. This experiment shows that our framework can deal with a more complex environment. As the environment is modeled as a mesh object, minor changes are necessary for the constraints of the motion optimizer and the cost of the contact optimizer. First, for the ground contact constraint, a joint vertex is projected onto the environment instead of onto the ground. To this end, we build a convex hull of each part of a chair so that the projection operator can find the closest point on the environment quickly using the kd-tree. Testing penetration into the environment is also performed



(a) Hill climbing (b) Standing up (c) Rolling (d) Sitting on chair

Fig. 13: Results of correcting motions.

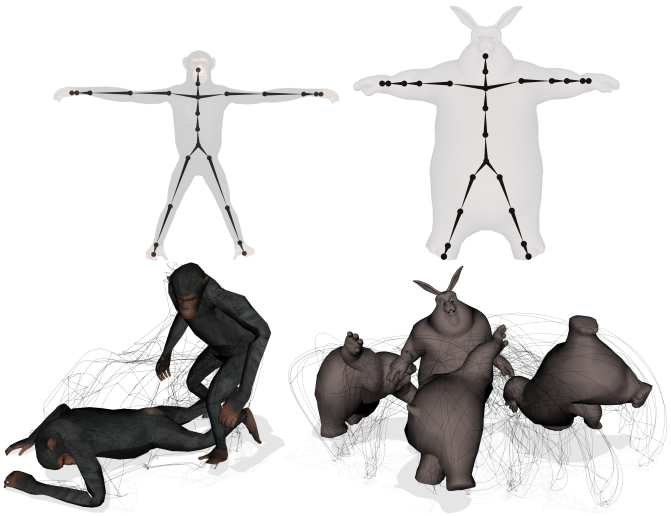


Fig. 14: The joint hierarchy of the additional characters used for motion retargeting. They are characterized by long arms (Chimpanzee) and fat body type (Bunny). Bottom: Chimpanzee for standing up example and bunny for cartwheel example.

with respect to the convex hulls, and if a vertex is found to penetrate, then the operator of the penetration constraint pushes the vertex to the closest point on the surface. Second, $f_v(\mathsf{c}, t)$ in (23) is modified to create a suitable pose for the given environment. Eventually, the ground projection operator $P_{sg}(\mathsf{c}, v_{ref})$ is replaced with a modified projection operator that finds the closest point on the environment. Note that contacts between the arms and the chair are also created by our method because the dynamic error of the motion optimizer decreases by the contacts.

Experiments on Other Characters

To verify the robustness of our method, we applied two new characters to the examples with the same contact state sequences. The characters, Chimpanzee and Bunny, have the identical joint hierarchy with the reference character (Fig. 14) but have different bone length ratios. The chimpanzee character has much longer arms than human, which may induce penetration of hands (Sec. 4). The bunny character has very thick flesh that causes self-penetration. To prevent this penetration, we attached virtual spheres on spine and wrist joints and applied penetration potential (Sec. 3.2.6) to these spheres during the motion optimization. Retargeting to new character was achieved without difficulty; motion optimizer takes the contact state sequence, new character and the reference motion as inputs, and then generates the retargeted motion that preserves the original style without making any serious visual artifacts. Figure 14 (bottom) shows the snapshot of the retargeted motions.

TABLE 3: The total computation time (sec.) for each example. The numbers in the parentheses are the average computation time per frame.

Motion	Motion optimizer (0.1)	Contact optimizer (0.2)	Number of Frames
Walking	13.2	17.7	85
Running	8.5	13.2	43
Standing Up	4.98	92.68	162
Cartwheel	11.90	73.51	255
Dancing	11.9	94.8	254
Rolling	8.50	153.29	210
Sitting on Chair	11.58	112.98	238

Implementation Details and Performance

Our framework is optimized for parallel processing in that each projection operator works independently; evaluating the samples on the contact optimizer also allows parallel computation. We experimented with a computer with a XEON E5-2630v3 CPU, which has 8 cores and 16 threads, and used OpenMP and the Intel Math Kernel Library (MKL). The computation time for each example is summarized in Table 3.

Although original motion clips have different bone lengths, we used the same character across all motions and our framework robustly worked for all motions. Since the matrix A, which represents a relationship of constraint between joint positions, is a sparse matrix, we use the sparse Cholesky solver, CHOLMOD package [40], to solve (3), and L-BFGS-B [41] to solve inverse kinematics under the joint limits.

Figure 15 compares the convergence rate of our motion optimizer and joint angle-based optimization method. The latter minimizes the same cost function as the motion optimizer except the bone-length constraint that is naturally satisfied. The LBFGS method was used with analytic Jacobians. The figure shows that our alternating solver reduces error to a satisfactory level much faster than the joint angle-based optimization.

Figure 16 shows the error of each potential. All potentials decrease stably along time. Note that the error of the bone-length constraint, which strongly affects visual realism, quickly converges towards zero after 10 iterations.

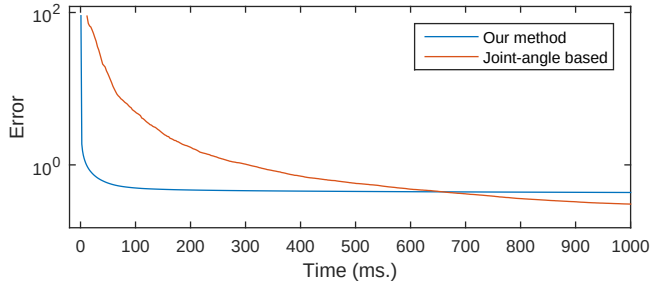


Fig. 15: Convergence rates of our motion optimizer (blue) and a joint angle-based optimizer (orange) for the walking example.

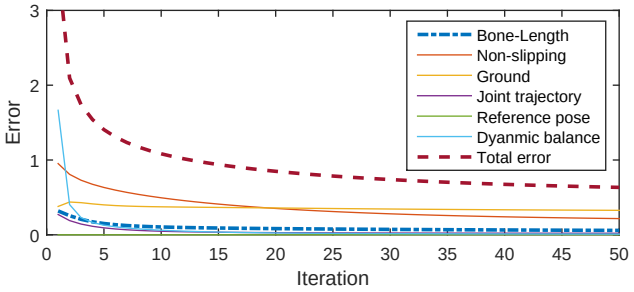


Fig. 16: Convergence rate of each potential for the walking example.

5 Discussion and Future Work

We introduced a motion correction framework that generates physically plausible motion as well as its supporting contact states. The computation time of our method is relatively short compared with other methods, which is due to several features of our method.

First, the contact optimizer discovers the optimal contact state sequence by solving an optimal path finding algorithm of a graph, which can be performed very efficiently. This was achieved by approximating the error of a pose corresponding to a contact state by the error of a pose obtained by modifying the reference pose, under the assumption that the optimized motion will have similar motion style with the reference. The actual cost of the graph is learned in an iterative manner, which shows a fast convergence within a few minutes or less than a minute. Second, by adopting the projective dynamics to a character motion editing problem, we could obtain the optimized motion within a few seconds. This fast performance of the motion optimizer makes our contact optimization framework performed in a feasible time duration. Finally, all of the methods described above can be processed in parallel, so multicore processors can speed up the processing.

Our method has several limitations that can be interesting topics for future work. First, the dynamic balance constraint considers angular momentum only indirectly. The angular momentum is taken into account when calculating GRFs (Eq. (13)), but the admissibility of angular momentum is not enforced explicitly. This scheme is effective when the reference motion is not severely damaged with respect to dynamic plausibility. In such a case, like the rolling motion example, however, our method shows a limited quality in terms of dynamic accuracy. More direct treatment of angular momentum may help improve these challenging cases.

Second, we only assumed the unilateral contact with the environment, and thus motions such as bar hanging cannot be created. In addition, our non-slip constraint in the motion optimizer prevents dynamic foot-ground contacts, such as sliding and compliant contact. One possible solution to enabling sliding contact would be that the motion optimizer penalizes only the normal component of contacting body parts. To this end, the contact optimizer also needs to be improved so that it can identify a contact to be a sliding or static contact. Including such contact will increase the range of producible motions.

Next, we have assumed that the environment of the reference motion is similar to the target environment, and thus relatively small modification of motion suffices to restore the naturalness. In the future, we will improve our method so that the motion refinement can hold larger change of environment, such as creating walking motion on irregular stepping-stones from a normal walking motion.

Lastly, our method deals with only the geometric and physical feasibilities of input motion but does not consider other factors related with the naturalness of human motion. Thus, resulting contact states and points may look implausible despite their geometric and physical correctness. The diversity term improved the naturalness by suppressing frequent change of contact but it was not enough. An important subject for future work is to identify high level factors for the naturalness of human motion and apply them to correcting and generating human motions.

Acknowledgments

This work was supported by ICT R&D program (2017-0-00162) funded by MSIT/IITP and Basic Science Research Program (NRF-2017R1A2B2006160) funded by MSIT, Korea.

References

[1] L. Kovar, J. Schreiner, and M. Gleicher, "Footskate cleanup for motion capture editing," in *Proceedings of the 2002 ACM SIGGRAPH/Eurographics symposium on Computer animation*. ACM, 2002, Conference Proceedings, pp. 97–104.

[2] D. Holden, T. Komura, and J. Saito, "Phase-functioned neural networks for character control," in *ACM Transactions on Graphics (TOG)*. ACM, 2017, Conference Proceedings.

[3] T. Kwon and J. K. Hodgins, "Momentum-mapped inverted pendulum models for controlling dynamic human motions," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 1, p. 10, 2017.

[4] L. Liu, K. Yin, M. van de Panne, T. Shao, and W. Xu, "Sampling-based contact-rich motion control," in *ACM Transactions on Graphics (TOG)*, vol. 29. ACM, 2010, Conference Proceedings, p. 128.

[5] D. E. Orin, A. Goswami, and S.-H. Lee, "Centroidal dynamics of a humanoid robot," *Autonomous Robots*, vol. 35, no. 2-3, pp. 161–176, 2013.

[6] S. Bouaziz, S. Martin, T. Liu, L. Kavan, and M. Pauly, "Projective dynamics: Fusing constraint projections for fast simulation," *ACM Trans. Graph.*, vol. 33, no. 4, pp. 154:1–154:11, Jul. 2014. [Online]. Available: http://doi.acm.org/10.1145/2601097.2601116

[7] M. Liu, A. Micaelli, P. Evrard, and A. Escande, "Task-driven posture optimization for virtual characters," in *Proceedings of the 11th ACM SIGGRAPH/Eurographics conference on Computer Animation*. Eurographics Association, 2012, Conference Proceedings, pp. 155–164.

[8] A. Safonova and J. K. Hodgins, "Construction and optimal search of interpolated motion graphs," *ACM Trans. Graph.*, vol. 26, no. 3, Jul. 2007. [Online]. Available: http://doi.acm.org/10.1145/1276377.1276510

[9] E. S. Ho, T. Komura, and C.-L. Tai, "Spatial relationship preserving character motion adaptation," *ACM Transactions on Graphics (TOG)*, vol. 29, no. 4, p. 33, 2010.

[10] T. Kwon, K. H. Lee, J. Lee, and S. Takahashi, "Group motion editing," in *ACM Transactions on Graphics (TOG)*, vol. 27. ACM, 2008, Conference Proceedings, p. 80.

[11] M. Kim, K. Hyun, J. Kim, and J. Lee, "Synchronized multi-character motion editing," in *ACM transactions on graphics (TOG)*, vol. 28. ACM, 2009, Conference Proceedings, p. 79.

[12] Y. Lipman, O. Sorkine, D. Cohen-Or, D. Levin, C. Rossi, and H.-P. Seidel, "Differential coordinates for interactive mesh editing," in *Shape Modeling Applications, 2004. Proceedings*. IEEE, 2004, Conference Proceedings, pp. 181–190.

[13] S. Tak, O. Song, and H. Ko, "Motion balance filtering," in *Computer Graphics Forum*, vol. 19. Wiley Online Library, 2000, Conference Proceedings, pp. 437–446.

[14] K. Yamane and Y. Nakamura, "Dynamics filter-concept and implementation of online motion generator for human figures," *IEEE transactions on robotics and automation*, vol. 19, no. 3, pp. 421–432, 2003.

[15] K. Yin, K. Loken, and M. van de Panne, "Simbicon: Simple biped locomotion control," in *ACM SIGGRAPH 2007 Papers*, ser. SIGGRAPH '07. New York, NY, USA: ACM, 2007. [Online]. Available: http://doi.acm.org/10.1145/1275808.1276509

[16] T. Geijtenbeek, N. Pronost, and A. F. van der Stappen, "Simple data-driven control for simulated bipeds," in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*. Eurographics Association, 2012, Conference Proceedings, pp. 211–219.

[17] L. Liu, M. V. D. Panne, and K. Yin, "Guided learning of control graphs for physics-based characters," *ACM Transactions on Graphics (TOG)*, vol. 35, no. 3, p. 29, 2016.

[18] Y. Lee, S. Kim, and J. Lee, "Data-driven biped control," *ACM Trans. Graph.*, vol. 29, no. 4, pp. 129:1–129:8, Jul. 2010. [Online]. Available: http://doi.acm.org/10.1145/1778765.1781155

[19] U. Muico, Y. Lee, J. Popović, and Z. Popović, "Contact-aware nonlinear control of dynamic characters," *ACM Trans. Graph.*, vol. 28, no. 3, pp. 81:1–81:9, Jul. 2009. [Online]. Available: http://doi.acm.org/10.1145/1531326.1531387

[20] M. da Silva, Y. Abe, and J. Popović, "Interactive simulation of stylized human locomotion," pp. 82:1–82:10, 2008. [Online]. Available: http://doi.acm.org/10.1145/1399504.1360681

[21] Y. Abe, C. K. Liu, and Z. Popović, "Momentum-based parameterization of dynamic character motion," in *Proceedings of the 2004 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, ser. SCA '04. Aire-la-Ville, Switzerland, Switzerland: Eurographics Association, 2004, pp. 173–182. [Online]. Available: http://dx.doi.org/10.1145/1028523.1028546

[22] S. Andrews, I. Huerta, T. Komura, L. Sigal, and K. Mitchell, "Real-time physics-based motion capture with sparse sensors," in *Proceedings of the 13th European Conference on Visual Media Production (CVMP 2016)*. ACM, 2016, p. 5.

[23] C. K. Liu, A. Hertzmann, and Z. Popović, "Learning physics-based motion style with nonlinear inverse optimization," *ACM Trans. Graph.*, vol. 24, no. 3, pp. 1071–1081, Jul. 2005. [Online]. Available: http://doi.acm.org/10.1145/1073204.1073314

[24] A. C. Fang and N. S. Pollard, "Efficient synthesis of physically valid human motion," in *ACM Transactions on Graphics (TOG)*, vol. 22. ACM, 2003, Conference Proceedings, pp. 417–426.

[25] Z. Popovi and A. Witkin, "Physically based motion transformation," in *Proceedings of the 26th annual conference on Computer graphics and interactive techniques*. ACM Press/Addison-Wesley Publishing Co., 1999, Conference Proceedings, pp. 11–20.

[26] T. Kwon and J. Hodgins, "Control systems for human running using an inverted pendulum model and a reference motion capture sequence," in *Proceedings of the 2010 ACM SIGGRAPH/Eurographics Symposium on Computer Animation*. Eurographics Association, 2010, Conference Proceedings, pp. 129–138.

[27] N. Naksuk, C. G. Lee, and S. Rietdyk, "Whole-body human-to-humanoid motion transfer," in *Humanoid Robots, 2005 5th IEEE-RAS International Conference on*. IEEE, 2005, Conference Proceedings, pp. 104–109.

[28] S. Nakaoka, A. Nakazawa, K. Yokoi, H. Hirukawa, and K. Ikeuchi, "Generating whole body motions for a biped humanoid robot from captured human dances," in *Robotics and Automation, 2003. Proceedings. ICRA'03. IEEE International Conference on*, vol. 3. IEEE, 2003, Conference Proceedings, pp. 3905–3910.

[29] N. S. Pollard, J. K. Hodgins, M. J. Riley, and C. G. Atkeson, "Adapting human motion for the control of a humanoid robot," in *Robotics and Automation, 2002. Proceedings. ICRA'02. IEEE International Conference on*, vol. 2. IEEE, 2002, Conference Proceedings, pp. 1390–1397.

[30] C. K. Liu, "Dextrous manipulation from a single grasping pose," *ACM Transactions on Graphics*, vol. 28, no. 3, 2009.

[31] C. Kang and S.-H. Lee, "Environment-adaptive contact poses for virtual characters," in *Computer Graphics Forum*, vol. 33, no. 7. Wiley Online Library, 2014, pp. 1–10.

[32] I. Mordatch, E. Todorov, and Z. Popovi, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, p. 43, 2012.

[33] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *Int. J. Rob. Res.*, vol. 33, no. 1, pp. 69–81, Jan. 2014. [Online]. Available: http://dx.doi.org/10.1177/0278364913506757

[34] A. Escande, A. Kheddar, and S. Miossec, "Planning contact points for humanoid robots," *Robotics and Autonomous Systems*, vol. 61, no. 5, pp. 428–442, 2013.

[35] K. Hauser, T. Bretl, J.-C. Latombe, K. Harada, and B. Wilcox, "Motion planning for legged robots on varied terrain," *The International Journal of Robotics Research*, vol. 27, no. 11-12, pp. 1325–1349, 2008.

[36] C. Mandery, J. Borras, M. Jochner, and T. Asfour, "Analyzing whole-body pose transitions in multi-contact motions," in *Humanoid Robots (Humanoids), 2015 IEEE-RAS 15th International Conference on*. IEEE, 2015, Conference Proceedings, pp. 1020–1027.

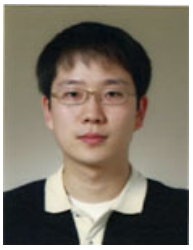
[37] C. Kang and S.-H. Lee, "Multi-contact locomotion using a contact graph with feasibility predictors," *ACM Transactions on Graphics (TOG)*, vol. 36, no. 2, p. 22, 2017.

[38] P. Hämäläinen, S. Eriksson, E. Tanskanen, V. Kyrki, and J. Lehtinen, "Online motion synthesis using sequential monte carlo," *ACM Transactions on Graphics (TOG)*, vol. 33, no. 4, p. 51, 2014.

[39] L. Jost, "Entropy and diversity," *Oikos*, vol. 113, no. 2, pp. 363–375, 2006. [Online]. Available: http://dx.doi.org/10.1111/j.2006.0030-1299.14714.x

[40] Y. Chen, T. A. Davis, W. W. Hager, and S. Rajamanickam, "Algorithm 887: Cholmod, supernodal sparse cholesky factorization and update/downdate," *ACM Trans. Math. Softw.*, vol. 35, no. 3, pp. 1–14, 2008.

[41] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, "Algorithm 778: L-bfgs-b: Fortran subroutines for large-scale bound-constrained optimization," *ACM Trans. Math. Softw.*, vol. 23, no. 4, pp. 550–560, Dec. 1997. [Online]. Available: http://doi.acm.org/10.1145/279232.279236



Sukwon Lee is currently a Ph.D. candidate with the Graduate School of Culture Technology at KAIST and he received the M.S. degree in Computer Science from Gwangju Institute of Science and Technology, Korea, in 2013 and B.S. degree in Computer Engineering from Korea Aerospace University, Korea, in 2011. He's research interests include physics-based character animation and machine learning techniques for analyzing human motion data.

Sung-Hee Lee is an Associate Professor with the Graduate School of Culture Technology at KAIST. His research interests include autonomous human animation, avatar motion generation, and human modeling. He received the Ph.D. degree in Computer Science from University of California, Los Angeles, USA, in 2008, and the B.S. and the M.S. degree in Mechanical Engineering from Seoul National University, Korea, in 1996 and 2000, respectively.