ACM SIGGRAPH / Eurographics Symposium on Computer Animation 2024
M. Skouras and H. Wang
(Guest Editors)

COMPUTER GRAPHICS *forum*
Volume 43 (2024), Number 8

PartwiseMPC: Interactive Control of Contact-Guided Motions

N. Khoshsiyar[1] R. Gou[1] T. Zhou[1] S. Andrews[2,3] M. van de Panne[1]
[1]University of British Columbia, Canada
[2]École de Technologie Supérieure, Canada
[3]Roblox, USA

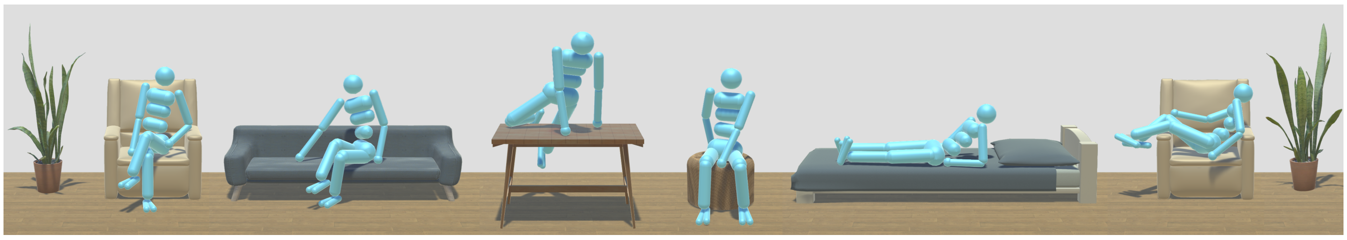


Figure 1: *Physics-based simulations of contact-rich tasks including: (a) crossing legs while seated; (b) sliding sideways on a sofa; (c) climbing on-and-off a table; (d) seated rotation on a stool; (e) climbing into bed; (f) turning sideways in a chair.*

Abstract
Physics-based character motions remain difficult to create and control. We make two contributions towards simpler specification and faster generation of physics-based control. First, we introduce a novel partwise model predictive control (MPC) method that exploits independent planning for body parts when this proves beneficial, while defaulting to whole-body motion planning when that proves to be more effective. Second, we introduce a new approach to motion specification, based on specifying an ordered set of contact keyframes. These each specify a small number of pairwise contacts between the body and the environment, and serve as loose specifications of motion strategies. Unlike regular keyframes or traditional trajectory optimization constraints, they are heavily under-constrained and have flexible timing. We demonstrate a range of challenging contact-rich motions that can be generated online at interactive rates using this framework. We further show the generalization capabilities of the method.

CCS Concepts
• ***Computing methodologies*** $\rightarrow$ *Physical simulation; Robotic planning;*

1. Introduction

Character motions coming from a physics-based simulation exhibit a palpable presence and embodiment in their surroundings that is often missing from kinematic motions. However, solving for the control needed to create purposeful physics-based motions is known to be a challenging problem. In this context, reinforcement learning methods have seen significant success in producing efficient and capable control policies for physics-based characters. This comes with related limitations: hours or days of compute for the offline learning; a common reliance on motion capture data to guide the solutions; limited generalization outside the space of situations it has been trained on; and often a lack of diversity when faced with the same situation twice.

Online control provides an alternative class of solutions, usually designed around a receding horizon planner. Commonly known as model predictive control (MPC), these optimize the control actions for a fixed horizon duration, H, into the future, often using repeated simulations forward from the current state. The first action(s) are then executed and then the planning process is repeated. These methods have the advantages of (a) requiring no offline computation and therefore supporting fast design iteration; (b) being able to generalize to new situations on demand; (c) success at exploring discontinuous solution spaces, therefore addressing issues of exploration; and (d) producing solution diversity.

In this paper we explore interactive-speed MPC-based solutions to rich-contact motions, such as those shown in Figure 1. We introduce a novel partwise MPC solver that improves on a standard sample-based MPC solver. At the cost of a few rollouts, this adds control samples which recombine existing samples in a way that can, at all times, exploit any body-part independence that may exist. This stands in contrast to the basic sampler which assumes that all body motions must be fully coordinated across all joints. We further introduce *contact keyframes* as a minimal sparse task description. This consists of a set of one-or-more contact pairs that should be achieved between a designated point on the body and a point in the environment. The contact keyframes serve as a minimally-constrained specification of a desired motion or motion strategy. They act as sequential waypoint constraints for the motion but do not contain timing information.

Our contributions are as follows:

- PartwiseMPC, a method that can readily exploit the varying degrees of full-body-coordination vs body-part-independence that exist during complex motions;
- contact keyframes, a sparse and flexible means to specify a motion or motion strategy that allows for significant generalization and motion diversity; and
- a suite of scenes and related skills that demonstrate interactive, online synthesis of contact-rich interactions between a physics-based humanoid and its environment.

2. Related Work

Our method has connections with previous work in animation, robotics, and control. We focus our review on the closest related work. For a longer-term history of physics-based character animation we refer the reader to suitable survey papers, e.g., [GP12].

2.1. Trajectory Optimization and Contact Planning

Methods based on trajectory optimization have long been proposed in computer animation and robotics to generate complex motions for physics-based characters or robots. In this class of approach, motions are modeled as a function of time and then optimized to minimize an objective function that encodes desired characteristics of the motion. The motion physics can be included by adding the equations of motion as trajectory constraints, as introduced for physics-based animation in [WK88]. Alternatively, forward dynamics simulations can be used to determine the motion resulting from given control actions, as used by *shooting* or *multiple-shooting* methods. In model predictive control (MPC), trajectory optimization is applied over some limited time horizon H. The first action(s) of the resulting plan are then taken, followed by replanning again with another iteration of trajectory optimization. Trajectory optimization methods have been developed to track motion sketches [LP02, LYvdP*10, LYG15, XK21], control rotational behavior [BMYZ13], or optimize task-specific objectives [AB-dLH13, BPF17]. Motion planning using optimal control methods and trajectory optimization are common in robotics, generally using similar methodologies to those proposed for physics-based animation. We refer the reader to a recent survey paper for a more in-depth summary of robotics methods [WPH*23].

Two phase approaches are common, recognizing that it can be useful to first find a coarse motion plan, including contact locations, followed by a full-body motion plan, e.g., [TDPP*18, KLVDP20, SFH23]. Relatedly, Mordatch et al. [MTP12] employ a trajectory optimization framework for generating motions defined by a pre-allocated set of contact phases, using a continuation scheme. Although contact planning is discontinuous due to the binary nature of contacts, continuous auxiliary variables are introduced that encode the activation of potential contacts within the allocated contact phases. With this in place, they are able to synthesize long duration complex motions. It is limited by a simplified physical model, the inability to generate interactions with sliding contacts, and not being interactive. Tassa et al. [TET12] use model predictive control (MPC) and the iterative LQG method to synthesize complex motions for physics-based humanoids. Hämäläinen et al. [HET*14, HRL15] develop particle belief propagation methods to demonstrate the suitability of sampling-based MPC methods for interactive control of challenging physics-based motions, including difficult balance-related tasks.

Our work shares key ideas with prior work, particularly the important role of contacts in defining a motion and the use of trajectory optimization. However, it differs in several key respects. Our contact keyframes are a sparse subset of the actual contacts that can-and-will occur during a motion. Furthermore while they are fixed in order, their timing is flexible. We are able to generate sliding motions. Our partwise MPC algorithm improves on sampling-based methods by exploiting body-part independence when it exists, yielding improved optimization. Our implementation builds on the predictive sampling method and the underlying MuJoCo physics engine as presented in MuJoCo MPC [HGT*22].

2.2. Deep Reinforcement Learning

Deep reinforcement learning (DRL) has seen significant success in recent years for generating skilled control for physics-based characters. The recent survey by Kwiatkowski et al. [KAK*22] gives a comprehensive overview of reinforcement learning based approaches for character animation. Control policies trained through DRL demonstrate the capability to synthesize diverse motion skills, including locomotion [BCHF19, HTS*17], agile motions [YYVDPY21], climbing [BWL*23], martial arts [WGH21], getting up [TWGvdP22], and imitation of animation clips [PALvdP18]. Recent work has addressed the prolonged learning times associated for training control policies for new tasks, e.g. through effective use of pre-trained control policies [XXA*23]. In contrast, our approach does not require the offline learning inherent to RL-based methods. Consequently, new motions for complex tasks can be synthesized at interactive frame rates.

Several recent works have leveraged the idea that a part-level view of the motion planning task can be beneficial. For instance, Bae et al. [BWL*23] builds motion controllers that encode local part-wise skills, which are then combined to produce whole body motions for novel circumstances. Other work by has examined how to compose upper- and lower-body motions by leveraging discriminator ensembles from pre-trained policies and allowing the agent to explore novel combinations of partial body motions [XSZK23].

2.3. Contact-aware Motion Synthesis

Several kinematic strategies for contact-aware pose synthesis or motion synthesis have been explored, including the estimation of good poses inferred from an object's shape [KCGF14], art-directable contact specification [LFL*23], and kinematic planning and interpolation for contact-rich motion sequences such as getting on a bicycle [KL21]. In general, it is challenging to optimize trajectories through contacts, with specific methods being introduced to address this challenge, e.g., [PCT14, NFWB17]. Kapadia et al. [KXN*16] proposed to pre-compute annotations for a virtual environment that encode affordances for movement through contact. Tonneau et al. [TPM14] generate skeletal configurations of virtual characters that satisfy tasks and workspace constraints. Al-Asqhar et al. [AAKC13] adapt the motions of 3D characters during close interactions with other characters and their environment by introducing novel spatial relationship descriptors based on the proximity between body parts and sample points from the environment. Their method was later extended to address motion editing under large environmental changes that preserves characteristics of the original contact configurations [TAAP*16].

3. Method

Motion objectives can be described in many ways. In this work, we aim for motion descriptions that are rich enough to broadly describe motion strategies and motion outcomes, while also being sufficiently abstract to allow significant freedom in its execution. We thus define motion objectives using a combination of general motion objectives and a sequence of *contact keyframes*, which serve as a sparse and flexible specification of a contact-rich motion. The total motion objective to be minimized is given by J_{total}, which we describe in detail later. We use a *partwise MPC* technique to optimize the motion objective, as we now describe in detail.

3.1. Model Predictive Control

The motion objectives are optimized using online receding-horizon control optimization. As is typical of MPC, a planning loop optimizes for the future control actions beginning from the current state, over a fixed time horizon, H. The early portions of the actions corresponding to the best plan will be executed, while the later portions will be discarded as they are replaced by improved plans coming from next iterations of the online planning loop.

Baseline MPC. We build on the core predictive sampling MPC method used for Mujoco MPC [HGT*22]. The core idea of sampling-based MPC methods is to explore and evaluate a fixed number of possible futures ("samples"), and then select the best one. Each sample consists of a simulation *rollout*: beginning at the known current state, a forward simulation is computed over a fixed future time horizon H, using a sample-specific time-sequence of control actions. The objective function is evaluated for each rollout sample and this is used to select the best rollout. The actions of the best rollout define the *nominal* action trajectory: the default action sequence that will be progressively executed over time, to be used until a future planning iteration produces a revised plan.

In our work, the actions space consists of cubic spline curves defining joint torques and are parameterized by the values at the knot locations of the spline curves. The knot locations are equally spaced in time across the finite horizon; we typically use 3 knots across a horizon $H = 0.35$ s. The action space has dimension $n_{\text{joints}} \times n_{\text{knots}}$. The predictive sampling planner works by sampling

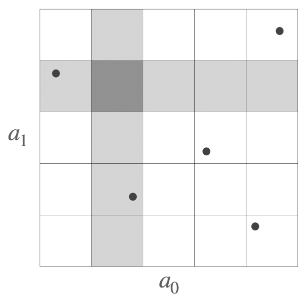


Figure 2: *2D optimal action search space.*

around the nominal action trajectory, which represents the best action plan resulting from the previous planning iteration. For a given sample, new cubic spline curves for the actions are produced by generating new knot points. For each rollout or sample, the value of each spline knot point is computed as the sum of (a) the current nominal action value observed at the time associated with the given spline knot, and (b) a randomized offset sampled from a zero mean Gaussian. A simulation rollout then determines the motion associated with the given action sample. The planner optimization uses a set of N rollouts, run in parallel threads, with the rollout corresponding to the best observed result, evaluated using J_{total}, then defining the new nominal trajectory.

The planner runs asynchronously, and in parallel with, the main simulation thread for the agent. In order to allow for more planning rollouts (and therefore possibly better plans), the agent simulation can be run with a *slowdown factor*, Γ. For example, running the agent simulation with $\Gamma = 0.25$ enables four times more compute for the planner per unit simulation time for the agent.

Partwise MPC. A shortcoming of the method described above is that it becomes inefficient when not all action components need to be synergistic, such as when arms and legs are free to move independently to accomplish individual objectives during some motion phases. As an extreme example, consider the case of a collection of K independent agents, each with their own objective function, being treated as a single fully-coordinated higher-dimensional action. If the probability of a randomly-sampled individual agent action performing well is p, then the probability of simultaneously sampling a good action across all K actions becomes p^K. As a result, there is a significant price to be paid when unnecessarily assuming fully coordinated actions.

To further understand the cost of ignoring partwise independence, we consider the toy 2D "dart board" problem shown in Figure 2. For action dimension a_0, the optimal solution falls in the gray column. Similarly, for action dimension a_1, the optimal solution falls in the gray row. Given a total objective $J = J_0(a_0) + J_1(a_1)$, the dark gray cell in position (2,2) then indicates the globally optimal solution. Intuitively, we have a low chance of sampling the dark gray cell using the 2D-action space, as illustrated by the 5 example samples. Alternatively, by recognizing that each dimension of the action has a mutually independent impact on the objective, we can readily identify the optimal combined action as (a_0^*, a_1^*), where * indicates the optimal action for each given dimension.

More formally, consider the D-dimensional variation of this problem, where $J_i \in [0,1]$ represents the objective cost for each dimension i. This defines a total objective $J \in [0,D]$ given by

$$J = \sum_{i=1}^{D} J_i$$

Given a success probability of $\mathbb{E}[J_i] = \alpha$ for any dimension i, then the probability of finding the global optimum for any sample is given by α^D. For N samples, the probability of failure for all N samples is then $P_{\text{fail}} = (1-\alpha^D)^N$. The global optimum success probability is then given by (assuming small α):

$$P_{success} = 1-(1-\alpha^D)^N \approx 1-(1-N\alpha^D) = N\alpha^D \qquad (1)$$

In the case of partwise planning, since each independent dimension is assessed separately, the probability of failure across all N samples, per dimension, is given by $(1-\alpha)^N$. The probability of finding the global optimal solution given D independent actions, again assuming α is small, becomes:

$$P'_{success} = (1-(1-\alpha)^N)^D \approx (1-(1-N\alpha))^D = N^D\alpha^D \qquad (2)$$

Taking the ratio $P'_{success}/P_{success} = N^{D-1}$ shows that for $D > 1$, the partwise method is more likely to find the optimal solution by a factor that is exponential in D, by avoiding the *curse of dimensionality* where possible . The above analysis provides a simple upper bound on the expected performance gain. The actual performance will then be significantly mitigated by many factors, most commonly a lack of the assumed independence.

Partwise character model. In our physics-based character control setting, we can partition the character into D *joint groups*, where the actions for each joint group can then be optimized independently of each other. At one extreme, we can treat each of the character joints as an independent group, resulting in n_{joints} groups. At the other extreme, we can assume full body coordination is required across all joints, resulting in a single joint group that encompasses the whole body, i.e., no partitioning, which is the assumption of the baseline MPC method. In practice, PartwiseMPC considers five unique partitionings of the body and limbs, shown in Figure 3. For a given partitioning, there is the assumption that the controls for the joint groups within that partitioning should be optimized independently.

By way of example, consider the second partitioning shown in Figure 3 which consists of three joint groups: left arm joints, right arm joints, and all the remaining joints. The best motions for these three groups are identified independently, typically each coming from a different rollout. We then synthesize, through composition, a final optimal action spline from these different rollouts; the best joint actions for each group are taken from their respective best rollouts. This then becomes the new partwise-optimized action.

Optimization by partwise partitioning. The use of partwise optimization requires no change to the MPC planner rollouts; instead, it only requires a reinterpretation of rollout results. The baseline objective function consists of a sum of objective terms related to contact keyframes as well as more generic terms, as we describe in detail shortly. For partwise optimization, each joint group optimizes a subset of the objective function terms, namely those that directly involve the joints in the given joint group. Contact keyframe

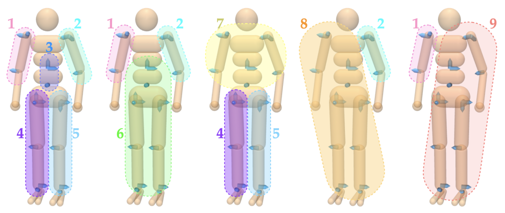


Figure 3: *Five partitionings of the joints for partwise rollout optimization. For a given partitioning, the best actions for each numbered group are identified independently, and then recombined.*

		C	RL	LL	RA	LA
Posture	upright torso	✓	-	-	-	-
	upright pelvis	✓	-	-	-	-
	upright left foot	-	-	✓	-	-
	upright right foot	-	✓	-	-	-
	head height	✓	-	-	-	-
	torso height	✓	-	-	-	-
Balance	align knee-feet	-	✓	✓	-	-
	align COM-feet	✓	✓	✓	✓	✓
Facing	torso direction	✓	-	-	-	-
Regularize	COM velocity	✓	✓	✓	✓	✓
	joint velocity	✓	✓	✓	✓	✓
	control	✓	✓	✓	✓	✓

Table 1: *Common objective terms and their correlated parts: Core (C), Right Leg (RL), Left Leg (LL), Right Arm (RA), Left Arm (LA). Note that the regularization terms and any term involving center of mass apply to all the joints.*

terms are included if the current contact specification involves a link that belongs to the given joint group. Common objective terms are included if they involve relevant joints, as documented in Table 1. For example, the left arm joint group of the second partitioning in Figure 3 sees the four left-arm (LA) common objective terms as designated in the right column of the table. Similarly, for the same partitioning, joint group 6, which spans the core body, left leg, and right leg, sees the objective terms designated by the checkmarks in the first three columns of the Table. During rollout evaluations, all objective terms are computed as a cumulative sum across timesteps as computed during the rollout.

Given the five partitioning choices, it still remains to be determined if the resulting partwise recombination of the actions for these cases actually does realize a better objective result, J_{total}. This is because in practice, the outcome may actually be worse given that each partition choice makes strong assumptions about motion independence that may be wrong. A solution is to immediately run a separate round of five rollouts that empirically evaluates the outcomes of the recombined actions that result from each partitioning choice. However, this comes at the cost of slowing down the planner with this extra round of evaluations. Instead, we evaluate the five partwise solutions in parallel with the rollouts belonging to the next iteration of the sample-based planner. The partwise solutions thus compete with the baseline sampling method. This comes at only a small cost, i.e., using $N-5$ baseline samples instead of the

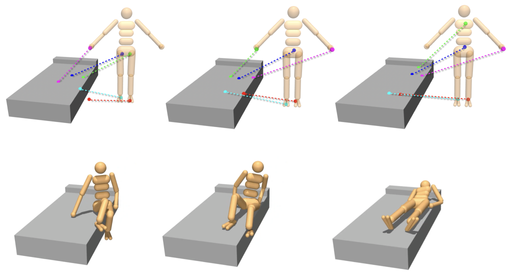


Figure 4: *Three contact keyframes used for lying-down in bed and then reused in reverse order for getting-up from bed, each consisting of five contact pairs.*

original N. Our experiments use 48-virtual-core machines, and thus we typically use $N = 45$ or $N = 90$. We provide detailed evaluations of the benefit of the partwise-optimization method. We further note that the recombined control actions may also now be slightly out-of-date given that the starting state has also advanced in the meantime. However, if this is truly problematic this will also be discovered in the resulting rollout evaluations.

3.2. Motion Specification

A sequence of contact keyframes is a collection of body-to-environment or body-to-body contacts that should be achieved within a given distance tolerance ϵ. More formally, a contact keyframe $\mathcal{K}_i$ is defined by a set of N_{cp} desired-contact pairs, $\{\mathbf{p}_1, \mathbf{q}_1\}, \ldots, \{\mathbf{p}_{N_{cp}}, \mathbf{q}_{N_{cp}}\}$. A contact pair consists of a point on the body $\mathbf{p}_j \in \mathbb{R}^3$ that should be in contact with a given point in the environment $\mathbf{q}_j \in \mathbb{R}^3$, although body-to-body contacts are also useful to specify. A motion task is then defined using a sequence of contact keyframes.

Contact points are defined in the local coordinates of the geometry they are defined with respect to. All contacts described by $\mathcal{K}_i$ should be achieved within their tolerance before proceeding to $\mathcal{K}_{i+1}$. In practice, we work with contact keyframes having 1–5 contact pairs, and use sequences of 5–9 contact keyframes.

Contact keyframes provide a convenient halfway point between providing only sparse end goals, leading to overly difficult exploration problems, and providing a fully detailed example trajectory, which is unlikely to generalize well and requires significantly more knowledge to specify. The motion timing is also left unconstrained, given that it is difficult in any case to know how much time may be required to achieve given motion phases. Importantly, contact keyframes readily support various types of motion generalization and emergent stylistic diversity, as we demonstrate later. They also allow for sequential composition and reuse in creating long motion sequences.

Figure 4 shows a sequence of three contact keyframes used to define a “getting up out of bed” motion. Contact keyframes are manually specified in our work, offering a compact loosely-constrained motion task specification, while still being sufficiently constrained to allow for efficient motion optimization. For symmetric motions, we can procedurally generate symmetric keyframes, e.g., for a rotation while seated on a stool. Alternatively, for some motions it is sufficient to use the same keyframes in a reverse order. These reuse strategies minimize the effort needed to generate new keyframes.

3.3. Trajectory Optimization using Contact Keyframes

At any point in time, the motion objective to be minimized is defined by

$$J_{\text{total}} = J_{\text{common}} + J_{\text{contact}}\,, \tag{3}$$

where J_{common} includes basic posture, balance, facing direction, as well as velocity and control effort regularization terms. Depending on the motion, a subset of these cost terms are activated and accordingly weighted. We elaborate more on these common terms shortly.

Contact objective. The contact objective encourages motions where the displacement vector $\mathbf{d}_j = \mathbf{p}_j - \mathbf{q}_j$ between a pair of contact points is zero. We concatenate all of the contact distance vectors for all pairs into a global contact displacement vector, such that

$$\mathbf{D} = [\mathbf{d}_1^T, \mathbf{d}_2^T, \ldots, \mathbf{d}_N^T]\,. \tag{4}$$

We then compute the objective term according to the smoothed distance function

$$J_{\text{contact}} = \sum_{k=1}^{3N} \sqrt{(D_k^2 + \alpha^2)} - \alpha\,, \tag{5}$$

where D_k denotes the kth element of the global contact displacement vector, and $\alpha = 0.1$ is a coefficient allowing control over the linearity of the loss function.

The success criteria for each contact keyframe is to sustain the contact pairs within a distance threshold ϵ for a duration T_C, and achieving this within a time limit $T_{\max}$. These parameters are generally held constant across all contact keyframes $\mathcal{K}_i$, but can also be adjusted per contact keyframe if needed.

Common objective. The term J_{common} provides additional control over the style of the motion. For example, the upright posture term helps avoid a crouched posture, in favor of a more upright style. The weights (and hence the associated styles) can be associated with designated keyframes, e.g., allowing the character to be as upright as possible while standing, and to assume other more curved or slumped postures when seated. Similarly, the regularization terms also generally help with stylizing the motion. For example, increasing the control regularization gives the character a look of lazy and less inclined to move, whereas removing the control regularization allows for more dynamic movement of the joints, at the price of possible jittery motion. A detailed breakdown of these common objective terms is given in the Appendix. Each objective applies to certain body parts during partwise optimization, the breakdown of which is shown in Table 1. We note that the body parts that are influenced by any given objective term are fixed apriori. Setting an objective term weight to zero effectively disables that term. By default, we keep the regularization terms unchanged when authoring new motions, only changing them if we observe them to be a hindrance to the optimization. Because many motions could base on these, we define default weights on the posture and

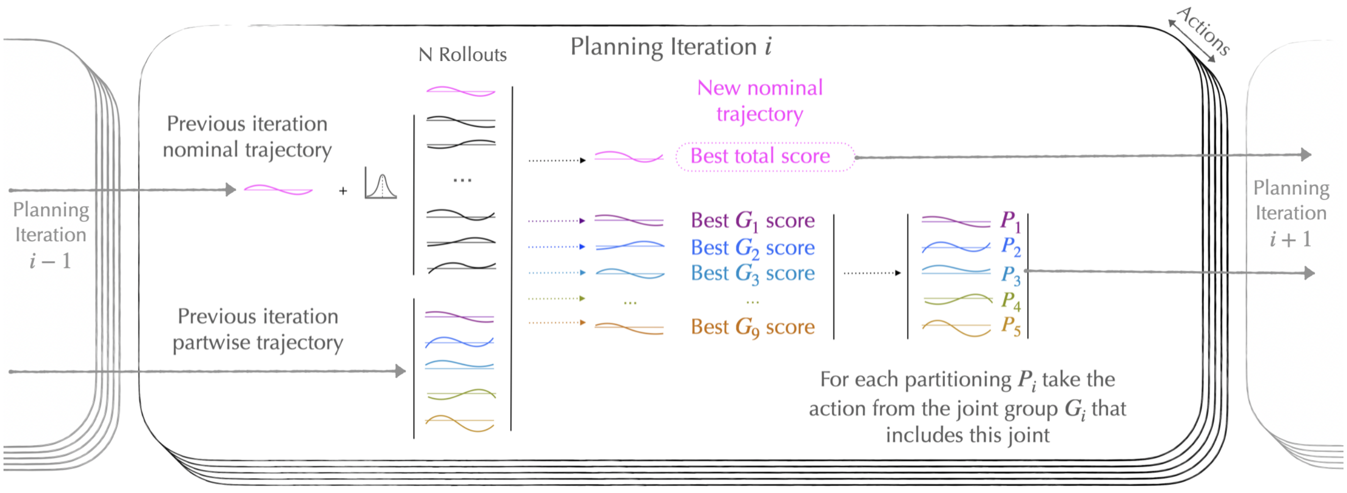


Figure 5: *Partwise sampling overview: in each planning iteration, we take the nominal trajectories from the previous iteration and add some noise to them during the rollouts. We keep the partwise nominal trajectories constant. Then we assess the best overall rollout, as well as 9 different assessments for 9 groupings of the joints. These are used to compose best action trajectories for the 5 partitions, namely partwise nominal trajectories. The global and partwise nominal trajectories are then resampled for the next planning iteration.*

balance terms for sitting and standing keyframes. We provide further details on the use of the common terms in Appendix A.

4. Results

We use the simulation environment provided by MuJoCo MPC [HGT*22]. Unless otherwise noted, simulations are run with a slowdown of $\Gamma = 0.25 \ldots 0.5$ (i.e., $25\% - 50\%$ of real-time). While smaller Γ will increase the quality of the generated motions, by allowing for more planning iterations per unit of agent simulation time, our given choice allows for experiments at interactive rates. Each contact keyframe contains between one to five contact pairs. Each basic strategy contains between three to nine contact keyframes, as listed in Table 2. Longer motions can be generated by concatenating existing keyframe sequences.

4.1. Skills

The results for this paper are best understood via the accompanying supplementary video. We demonstrate a diverse set of contact-rich motions with everyday objects, all computed online at interactive speeds. Many of these tasks are easy for humans to perform while being challenging to reproduce in simulation. This is due in part to the complexity of the multitude of possible interactions with the environment and the shortcomings of offline methods in such cases, e.g., needing to train in advance on all expected variations.

Skills diversity. We demonstrate that our method can generate successful control for a diverse set of skills. Three of these are shown in Figure 6. Table 2 provides an insight into the stability of the generated motions, their success rates and the diversity of resulting motions.

Emergent motion diversity. Due to the loosely constrained nature of our contact keyframe specification, we observe emergent diversity in the resulting motions. Given the same target contact keyframe, PartwiseMPC generates various valid, natural-looking solutions, as shown in Figure 7. The optimization has the flexibility to explore a variety of ways to solve the task due to the underspecification of the task. In the bench shuffle task each keyframe consists of only one contact pair between the character's pelvis and the bench and therefore several solution modes arise. Some episodes push with the hands on the bench or with the feet on the ground, and others discover that bringing the legs up and pushing on the bench is a viable solution. We note that with additional slowdown, e.g., $0.01 \leq \Gamma \leq 0.1$, and therefore more planning compute per unit of simulation time, the success rate and the stability of the results improve significantly. This comes at the cost of the motion being generated much slower than real time. As well, we observed that the planner often produced less diverse motions for small values of Γ; we suspect this is a result of preferring actions near the optimal solution.

Environment generalization. One of the main strengths of online trajectory optimization is its adaptability to new and unseen environments. We apply our authored contact keyframes to different variations of the same environment, and our method generalizes to these new settings with no further changes. In our experiments, we introduce rotate and translate the objects, as well as altering the friction coefficient. Further details on the environment variations are given in the Appendix.

Character generalization. Similarly, the existing motion strategies are transferable to characters with varying morphology. Figure 8 shows three different variations of a character that can successfully perform the given task, using the same contact keyframes authored for the default character proportions.

Skill concatenation. Contact keyframes can be resequenced to compose longer motions, allowing for repeated motions, motion

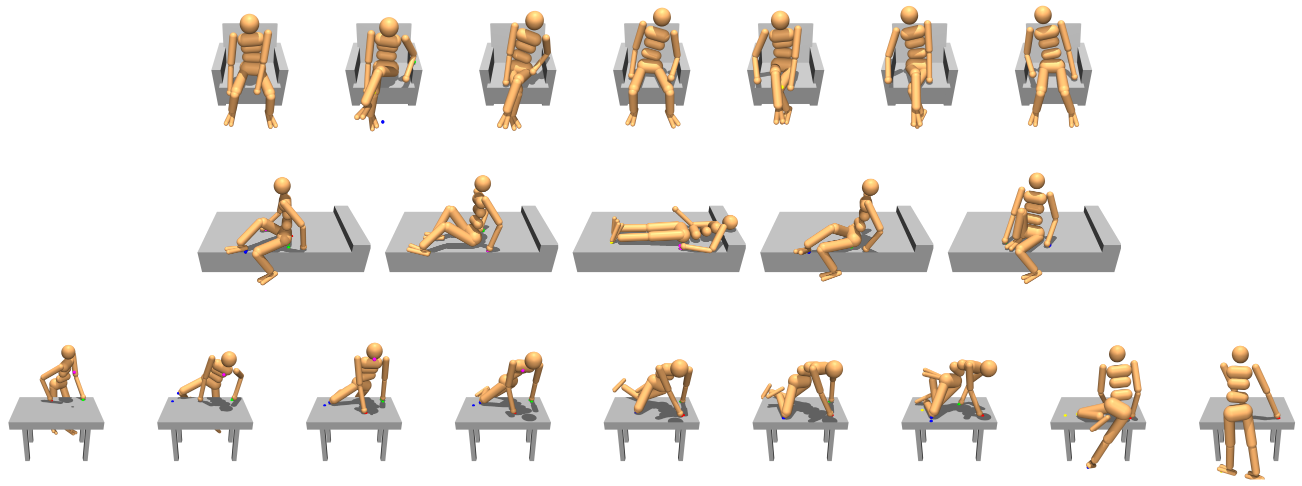


Figure 6: *Example skill strategies, from top to bottom for (a) Leg crossing; (b) Lying down on a bed and sitting up; (c) Climbing a table.*

Scene	Task	# KFs (N_{cp})	Success%	Mean Motion Duration (s)	Duration min-max (s)
Armchair	Stand to Sit to Stand	6 (4,4,4,4,1,1)	94	5.9 ± 0.5	5 - 9
	Cross Legs	7 (3,3,3,3,3,3,3)	96	20.0 ± 1.7	18 - 27
	Turn to relax	6 (4,4,4,4,4,4)	100	7.0 ± 0.2	6 - 8
Stool	Rotate	9 (3,3,3,3,3,3,3,3,3)	89	19 ± 5	15 - 26
Bench	Shuffle	5 (1,1,1,1,1)	89	22.9 ± 4.4	20 - 27
Bed	Lie Down and Get up	5 (5,5,5,5,3)	98	8.5 ± 1.2	7 - 15
Table	Climb	9 (3,4,4,5,5,5,5,5,5)	75	11.9 ± 10.6	9 - 26

Table 2: *Quantitative results for basic motion strategies. Performance assessed over 100 trials at $\Gamma = 0.25$, with success defined as completing all keyframes within the time limit. Total number of keyframes as well as number of contact pairs per keyframe are indicated for each motion strategy. Motion duration is reported for successful runs as mean ± one standard deviation and their minimum and maximum. Note that some motions include explicitly specified keyframe sustaining time. The high variance in the successful motion duration shows the diversity of the results produced.*

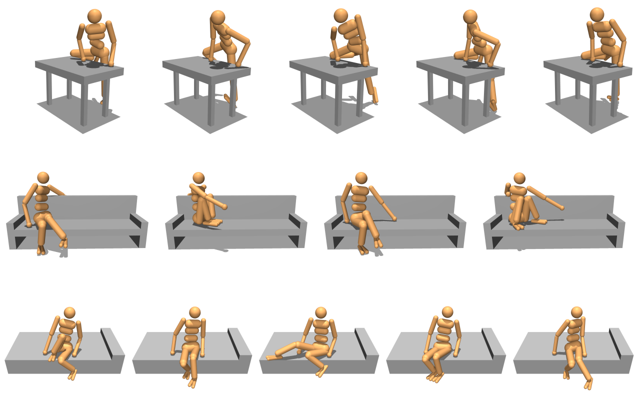


Figure 7: *Emergent diversity across different runs, given the same environment and aligned according to a given target contact keyframe. From top to bottom: (a) starting to climb the table; (b) moving sideways on the couch; (c) sitting up on the bed.*

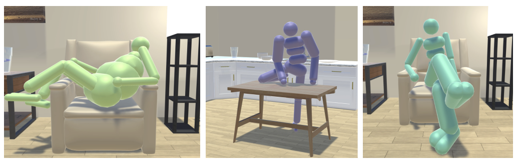


Figure 8: *Character morphology generalization, all tasks shown in the figure are successfully completed.*

reversal, or motion transitions. New keyframes can also be dynamically added via the user interface, as the simulation is running. Please refer to our supplemental video for examples of motion diversity, generalization experiments, and longer motion sequences.

4.2. Discussion

The motions produced by RL approaches may be of higher quality as they are typically smoother in nature, reflecting the smooth reference motion capture data. However, this comes at the price of requiring example motion data to track, lengthy offline computation, limited inference-time generalization to new situations, and lack of solution diversity. Smoother motions might be achieved for MPC methods in general via additional objective terms, but this may result in a more difficult optimization problem to solve.

Hyperparameters play an important role in the sample-based MPC methods discussed here, i.e., the baseline MuJoCo MPC and PartwiseMPC. The planning horizon, H, needs to be long enough to make measurable progress towards target contact keyframes while avoiding local minima in the objective, but should be short enough to allow for the randomized sampling in the action space to yield promising solutions within a limited compute budget. The number of sample rollouts, N, is another key parameter. Too few rollouts results in an impoverished ability to find near-optimal action sequences. Increasing N requires more compute budget per planning iteration, which by default results in less frequent (re)planning and therefore may not actually improve the solution quality. Alternatively, we can choose to compensate for the cost of more rollouts by employing a smaller slowdown factor $\Gamma \in (0,1]$, i.e., advancing the agent simulation, which runs in parallel to the planner itself, more slowly. This then allows for more planning computations per unit of agent simulation time. In general, if we wish to replan every k simulation time steps and we hold the available computing resources fixed, then the following holds true: $N \cdot H \cdot \Gamma = kT \cdot C$, where T is the simulation timestep and C is a constant. This does not take into account thread parallelism, which in practice is limited by the number of virtual cores.

4.3. Ablations

We perform several ablation studies to compare PartwiseMPC to the baseline predictive sampling method presented in [HGT*22]. Our planner is most effective when the motions of different body parts (*groups* in our terminology) do not require extensive coordination. Moreover, the benefits of our method become the most obvious at the limits of the sampling planner, e.g., low number of rollouts or for larger Γ (higher simulation speeds). Further results are provided below when discussing performance.

Quality vs. baseline. Motion trajectories generated using PartwiseMPC generally look smoother and less jittery in comparison with the baseline method. The partwise solutions help reduce redundant motions of joints that are not directly essential for a given motion phase. If a joint sees only regularization terms, PartwiseMPC solutions do better at optimizing these terms, resulting in smoother joint movements. Relatedly, for precise motions that do not require full body coordination between different limbs, PartwiseMPC generates noticeably improved results. This is best illustrated in the Reach-hand-targets task shown in the video.

Performance. We compare PartwiseMPC against the baseline MPC method in several ways. First, we compare the objective function values achieved at runtime during a given contact keyframe interval for three tasks, shown in Figure 9. PartwiseMPC results in better objective optimization at all points in time for all three motions. We further note that the baseline MPC completely fails at finishing the reach-target task with 15 rollouts, while PartwiseMPC consistently succeeds.

The next evaluation tracks the fraction of times that one of the 5 partwise solutions was chosen over 85 baseline sample solutions for N=90 rollouts. This is shown in Figure 10. The bench-shuffle exploits partwise the least (22%), while the hand-targets task exploits partwise the most frequently (66%). Note that these numbers may further vary across the different contact-keyframe intervals that comprise each task. An interesting perspective is to consider this evaluation as a practical method of quantifying the degree of partwise independence that exists in these motion task.

We further compare PartwiseMPC with baseline MPC on three metrics: (a) success-rate (failure occurs if it takes too long to reach a given keyframe); (b) total cost across the full task (lower is better); and (c) motion duration (shorter is better). Here we briefly summarize the conclusions of these evaluations, while Figure 12 in the Appendix illustrates the success-rate and total-cost metrics. For table-climb, lie-down-get-up, and stand-sit-stand, PartwiseMPC achieves a better success rate. A lower cost is also achieved for 14 of the 18 conditions tested. We further include the rotate-on-stool task as an exception, likely due to its highly underconstrained nature. The motion durations for PartwiseMPC tasks are also shorter than the baseline for 13 out of 18 conditions tested.

5. Conclusions

In this paper we have introduced PartwiseMPC, a method that can exploit the varying degrees of inter-limb coordination that are seen during human movements. In particular, it can do better at optimizing motions in situations when some partwise independence exists during a motion. As demonstrated experimentally, this occurs surprisingly often. We further introduce contact keyframes as a minimal and flexible motion abstraction for producing contact-rich human motions. The resulting system allows for *interactive* generation of novel physics-based motions, and generalizes well with respect to environment and morphology variations. It readily allows for physics-based characters to get in-and-out of bed, have basic body-aware manipulation skills with respect to chairs and benches, climb on-and-off tables, and more.

The current method still has a number of limitations. The contact keyframes that serve to loosely define the motion strategies are currently manually authored via a GUI that allows them to added and edited during a live simulation. It currently requires on the order of 10-15 minutes of interactive experimentation to design suitable keyframes for a new motion of moderate complexity. The specification of keyframes could be further automated in future work by extracting contact configurations from videos or from the output of contact-invariant optimization [MTP12].

The generated trajectories from our method could be used as target motions for RL-based policies, which would be trained offline but which, once trained, allow for truly fast-and-efficient real-time control. The lack of timing specification for the contact keyframes allows for more emergent behavior, but consequently offers less control over the speed of the resulting motions. We note that the method is currently less suitable for highly dynamic motions, which are better tackled using other methods such as direct imitation of an example motion. We furthermore do not focus on locomotion because higher-quality results are currently read-

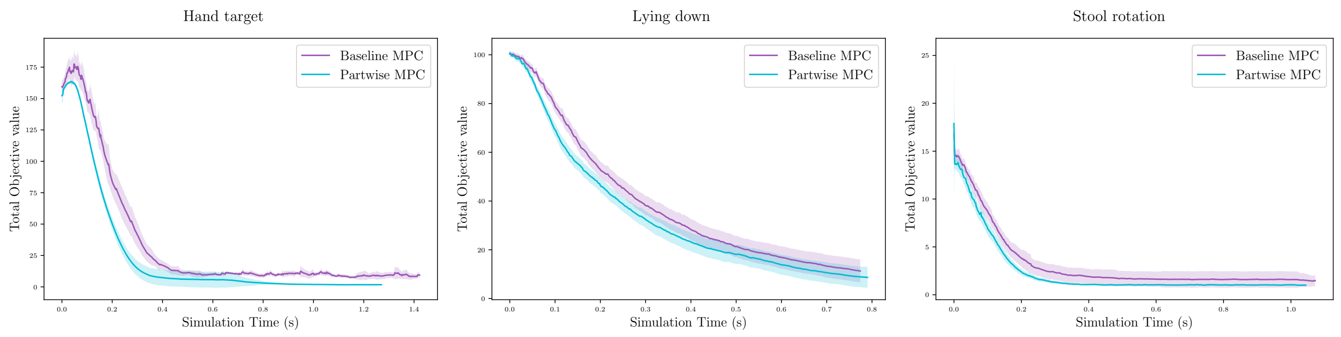


Figure 9: *Comparison of total objectives values achieved during the first keyframe, averaged over 50 trials, using N=45 rollouts, $\Gamma = 0.5$. From left to right:(a) hand target reach; (b) lying down on bed; (c) rotating on stool.*

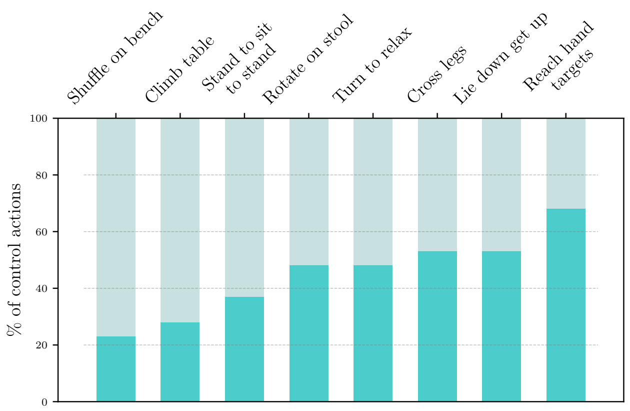


Figure 10: *Percentage of time the partwise solutions were chosen at any given planning time step, across 8 tasks. Results are averaged over 50 trials, using N=90 rollouts and $\Gamma = 0.5$.*

ily achieved with imitation-based RL approaches given the ready availability of motion capture data for such common motions.

Acknowledgements

This work was supported in part by NSERC RGPIN-2020-05929 and RGPIN-2018-05723. We thank Taylor Howell and the MuJoCo team at DeepMind, including Tom Erez, Saran Tunyasucunakool, and Yuval Tassa, for valuable discussions.

References

[AAKC13] AL-ASQHAR R. A., KOMURA T., CHOI M. G.: Relationship descriptors for interactive motion adaptation. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2013), SCA '13, Association for Computing Machinery, p. 45–53. URL: https://doi.org/10.1145/2485895.2485905, doi:10.1145/2485895.2485905. 3

[ABdLH13] AL BORNO M., DE LASA M., HERTZMANN A.: Trajectory optimization for full-body movements with complex contacts. *IEEE Transactions on Visualization and Computer Graphics 19*, 8 (2013), 1405–1414. doi:10.1109/TVCG.2012.325. 2

[BCHF19] BERGAMIN K., CLAVET S., HOLDEN D., FORBES J. R.: Drecon: data-driven responsive control of physics-based characters. *ACM Trans. Graph. 38*, 6 (nov 2019). URL: https://doi.org/10.1145/3355089.3356536, doi:10.1145/3355089.3356536. 2

[BMYZ13] BROWN D. F., MACCHIETTO A., YIN K., ZORDAN V.: Control of rotational dynamics for ground behaviors. In *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (New York, NY, USA, 2013), SCA '13, Association for Computing Machinery, p. 55–61. URL: https://doi.org/10.1145/2485895.2485906, doi:10.1145/2485895.2485906. 2

[BPF17] BORNO M. A., PANNE M. V. D., FIUME E.: Domain of attraction expansion for physics-based character control. *ACM Transactions on Graphics (TOG) 36*, 2 (2017), 1–11. 2

[BWL*23] BAE J., WON J., LIM D., MIN C.-H., KIM Y. M.: Pmp: Learning to physically interact with environments using part-wise motion priors. In *ACM SIGGRAPH 2023 Conference Proceedings* (New York, NY, USA, 2023), SIGGRAPH '23, Association for Computing Machinery. URL: https://doi.org/10.1145/3588432.3591487, doi:10.1145/3588432.3591487. 2

[GP12] GEIJTENBEEK T., PRONOST N.: Interactive character animation using simulated physics: A state-of-the-art review. *Computer graphics forum 31*, 8 (2012), 2492–2515. 2

[HET*14] HÄMÄLÄINEN P., ERIKSSON S., TANSKANEN E., KYRKI V., LEHTINEN J.: Online motion synthesis using sequential monte carlo. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 1–12. 2

[HGT*22] HOWELL T., GILEADI N., TUNYASUVUNAKOOL S., ZAKKA K., EREZ T., TASSA Y.: Predictive sampling: Real-time behaviour synthesis with mujoco, 2022. arXiv:2212.00541. 2, 3, 6, 8

[HRL15] HÄMÄLÄINEN P., RAJAMÄKI J., LIU C. K.: Online control of simulated humanoids using particle belief propagation. *ACM Trans. Graph. 34*, 4 (jul 2015). URL: https://doi.org/10.1145/2767002, doi:10.1145/2767002. 2

[HTS*17] HEESS N., TB D., SRIRAM S., LEMMON J., MEREL J., WAYNE G., TASSA Y., EREZ T., WANG Z., ESLAMI S., ET AL.: Emergence of locomotion behaviours in rich environments, 2017. 2

[KAK*22] KWIATKOWSKI A., ALVARADO E., KALOGEITON V., LIU C. K., PETTRÉ J., VAN DE PANNE M., CANI M.-P.: A survey on reinforcement learning methods in character animation. *Computer Graphics Forum 41*, 2 (2022), 613–639. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14504. 2

[KCGF14] KIM V. G., CHAUDHURI S., GUIBAS L., FUNKHOUSER T.: Shape2pose: Human-centric shape analysis. *ACM Transactions on Graphics (TOG) 33*, 4 (2014), 1–12. 2

[KL21] KIM Y., LEE S.-H.: Keyframe-based multi-contact motion synthesis. *The Visual Computer 37*, 7 (2021), 1949–1963. 2, 3

[KLVDP20] KWON T., LEE Y., VAN DE PANNE M.: Fast and flexible multilegged locomotion using learned centroidal dynamics. *ACM Transactions on Graphics (TOG) 39*, 4 (2020), 46–1. 2

[KXN*16] KAPADIA M., XIANGHAO X., NITTI M., KALLMANN M., COROS S., SUMNER R. W., GROSS M.: Precision: Precomputing environment semantics for contact-rich character animation. In *Proceedings of the 20th ACM SIGGRAPH Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2016), I3D '16, Association for Computing Machinery, p. 29–37. URL: https://doi.org/10.1145/2856400.2856404, doi:10.1145/2856400.2856404. 3

[LFL*23] LAKSHMIPATHY A. S., FENG N., LEE Y. X., MAHLER M., POLLARD N.: Contact edit: Artist tools for intuitive modeling of hand-object interactions. *ACM Transactions on Graphics (TOG) 42*, 4 (2023), 1–20. 2

[LP02] LIU C. K., POPOVIĆ Z.: Synthesis of complex dynamic character motion from simple animations. *ACM Transactions on Graphics (TOG) 21*, 3 (2002), 408–416. 2

[LYG15] LIU L., YIN K., GUO B.: Improving sampling-based motion control. *Computer Graphics Forum 34*, 2 (2015), 415–423. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12571. 2

[LYvdP*10] LIU L., YIN K., VAN DE PANNE M., SHAO T., XU W.: Sampling-based contact-rich motion control. In *ACM SIGGRAPH 2010 Papers* (New York, NY, USA, 2010), SIGGRAPH '10, Association for Computing Machinery. URL: https://doi.org/10.1145/1833349.1778865, doi:10.1145/1833349.1778865. 2

[MTP12] MORDATCH I., TODOROV E., POPOVIĆ Z.: Discovery of complex behaviors through contact-invariant optimization. *ACM Transactions on Graphics (Proc. SIGGRAPH) 31*, 4 (jul 2012). URL: https://doi.org/10.1145/2185520.2185539, doi:10.1145/2185520.2185539. 2, 8

[NFWB17] NEUNERT M., FARSHIDIAN F., WINKLER A. W., BUCHLI J.: Trajectory optimization through contacts and automatic gait discovery for quadrupeds. *IEEE Robotics and Automation Letters 2*, 3 (2017), 1502–1509. doi:10.1109/LRA.2017.2665685. 3

[PALvdP18] PENG X. B., ABBEEL P., LEVINE S., VAN DE PANNE M.: Deepmimic: Example-guided deep reinforcement learning of physics-based character skills. *ACM Trans. Graph. 37*, 4 (jul 2018). URL: https://doi.org/10.1145/3197517.3201311, doi:10.1145/3197517.3201311. 2

[PCT14] POSA M., CANTU C., TEDRAKE R.: A direct method for trajectory optimization of rigid bodies through contact. *The International Journal of Robotics Research 33*, 1 (2014), 69–81. doi:10.1177/0278364913506757. 3

[SFH23] SLEIMAN J.-P., FARSHIDIAN F., HUTTER M.: Versatile multicontact planning and control for legged loco-manipulation. *Science Robotics 8*, 81 (2023), eadg5014. 2

[TAAP*16] TONNEAU S., AL-ASHQAR R. A., PETTRÉ J., KOMURA T., MANSARD N.: Character contact re-positioning under large environment deformation. *Computer Graphics Forum 35*, 2 (2016), 127–138. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.12817. 3

[TDPP*18] TONNEAU S., DEL PRETE A., PETTRÉ J., PARK C., MANOCHA D., MANSARD N.: An efficient acyclic contact planner for multiped robots. *IEEE Transactions on Robotics 34*, 3 (2018), 586–601. 2

[TET12] TASSA Y., EREZ T., TODOROV E.: Synthesis and stabilization of complex behaviors through online trajectory optimization, 2012. doi:10.1109/IROS.2012.6386025. 2

[TPM14] TONNEAU S., PETTRÉ J., MULTON F.: Using task efficient contact configurations to animate creatures in arbitrary environments. *Computers & Graphics 45* (2014), 40–50. URL: https://www.sciencedirect.com/science/article/pii/S009784931400079X, doi:https://doi.org/10.1016/j.cag.2014.08.005. 3

[TWGvdP22] TAO T., WILSON M., GOU R., VAN DE PANNE M.: Learning to get up. In *ACM SIGGRAPH 2022 Conference Proceedings* (New York, NY, USA, 2022), SIGGRAPH '22, Association for Computing Machinery. URL: https://doi.org/10.1145/3528233.3530697, doi:10.1145/3528233.3530697. 2

[WGH21] WON J., GOPINATH D., HODGINS J.: Control strategies for physically simulated characters performing two-player competitive sports. *ACM Trans. Graph. 40*, 4 (jul 2021). URL: https://doi.org/10.1145/3450626.3459761, doi:10.1145/3450626.3459761. 2

[WK88] WITKIN A., KASS M.: Spacetime constraints. *ACM Siggraph Computer Graphics 22*, 4 (1988), 159–168. 2

[WPH*23] WENSING P. M., POSA M., HU Y., ESCANDE A., MANSARD N., DEL PRETE A.: Optimization-based control for dynamic legged robots. *IEEE Transactions on Robotics* (2023). 2

[XK21] XIE K., KRY P. G.: Inverse dynamics filtering for sampling-based motion control. *Computer Graphics Forum 40*, 6 (2021), 304–314. arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1111/cgf.14274. 2

[XSZK23] XU P., SHANG X., ZORDAN V., KARAMOUZAS I.: Composite motion learning with task control. *ACM Trans. Graph. 42*, 4 (jul 2023). URL: https://doi.org/10.1145/3592447, doi:10.1145/3592447. 2

[XXA*23] XU P., XIE K., ANDREWS S., KRY P. G., NEFF M., MCGUIRE M., KARAMOUZAS I., ZORDAN V.: Adaptnet: Policy adaptation for physics-based character control. *ACM Trans. Graph. 42*, 6 (dec 2023). URL: https://doi.org/10.1145/3618375, doi:10.1145/3618375. 2

[YYVDPY21] YIN Z., YANG Z., VAN DE PANNE M., YIN K.: Discovering diverse athletic jumping strategies. *ACM Trans. Graph. 40*, 4 (jul 2021). URL: https://doi.org/10.1145/3450626.3459817, doi:10.1145/3450626.3459817. 2

Appendix A: Common Objective Terms

The common objective terms are defined as follows:

Upright torso. This term describes how upright the upper body of the character is. The lower the weight, the more crouched the character would appear. The value is the difference between the z axis of the torso geometry and the global z axis.

$$cost_{uprighttorso} = 1 - (\hat{k}_{\text{torso}} \cdot \hat{k})$$

Upright pelvis. Similar to the above, the value of this cost term is the difference between the z axis of the pelvis geometry and the global z axis.

$$cost_{uprightpelvis} = 1 - (\hat{k}_{pelvis} \cdot \hat{k})$$

Upright left foot. This term encourages keeping the left foot parallel to the ground. It helps avoid motions of the ankle which disrupt balance of the character in certain motions.

$$cost_{uprightleftfoot} = 1 - (\hat{k}_{leftfoot} \cdot \hat{k})$$

Upright right foot. Similarly, this term keeps the right foot horizontal.

$$cost_{uprightrightfoot} = 1 - (\hat{k}_{rightfoot} \cdot \hat{k})$$

High head height. Encourages head to stay at a standing height.

$$cost_{headheight} = z_{head} - 1.4$$

High torso height. Keeps the torso close to a standing height.

$$cost_{headheight} = z_{torso} - 1.2$$

Knee-feet alignment. This cost term is most useful to determine the feet are placed directly below the knees, as a style choice, for example when sitting on a chair.

$$cost_{knee-feet} = avg(xypos_{knees}) - avg(xypos_{feet})$$

COM-feet alignment. Similar to the term above, this term ensures that the center of mass is positioned above the feet, which proves beneficial in positions requiring a standing balance.

$$cost_{COM-feet} = xypos_{COM} - avg(xypos_{feet})$$

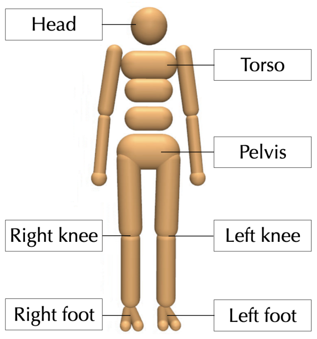


Figure 11: *Humanoid geometry labels.*

Torso facing direction. The user could choose any point on the floor, and this point on the horizontal plane will guide the facing direction of the humanoid's torso. This term helps guide the character to face a certain direction. This term was not used for the results presented in this paper.

COM velocity. Linear velocity of the center of mass is added directly as a residual term, to regulate the movements speed.

Joint velocity. The joint velocity is regulated to prefer slow motions. Typically the weight of this term is set to be low, to not overly affect the agility of the character.

Control. The joint torques are also regulated, to avoid erratic motions. Increasing the weight of this term causes the motion to look more relaxed. For motions that require a lot of micro-movements to keep balance, such as standing or table climbing, we keep the weight of this term very small (around 0.025). In contrast, for motions that look more relaxed and slow, we increase the weight of this term (0.8-1.0).

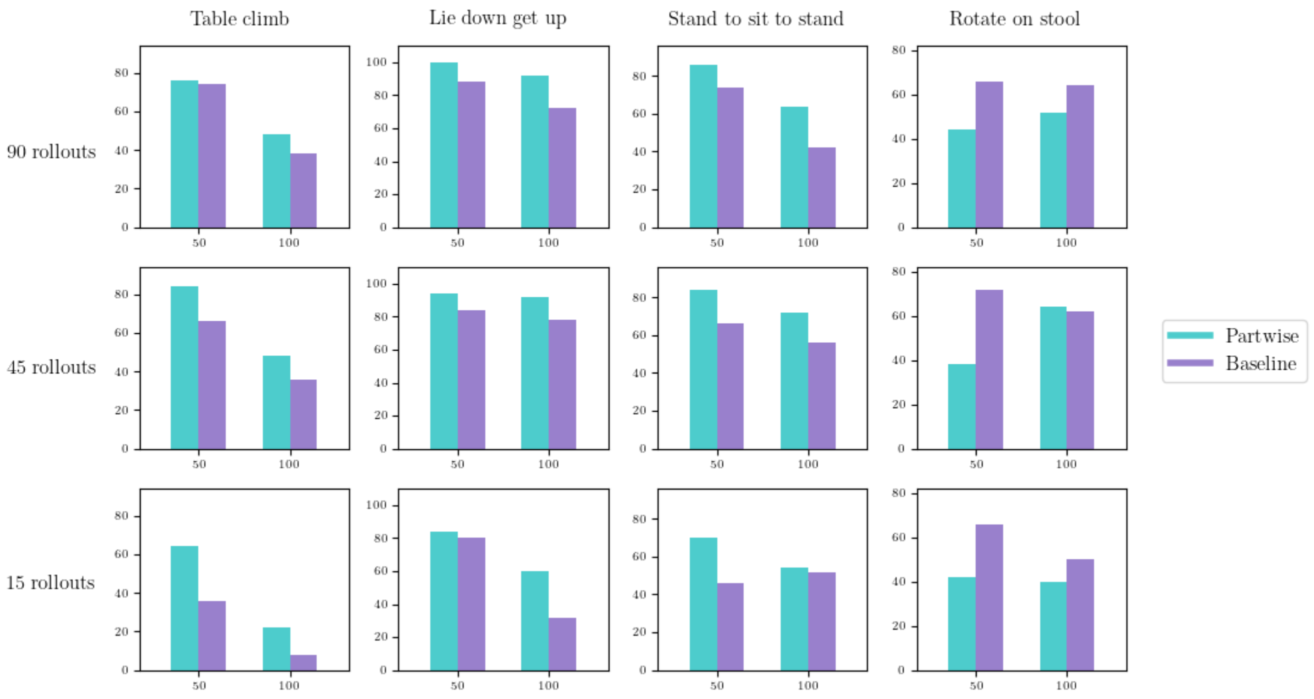
Appendix B: Generalization Parameters

Armchair	
default	height = 0.2 m, no tilt
tilted	entire chair tilted 15° backwards
short	height = 0.12m
very short	height = 0.04m.
Stool	
default	height = 0.4 m
short	height = 0.3 m
very short	height = 0.2 m
floor level	height = 0.001 m
Bench	
default	height = 0.25 m, no tilt, friction=0.7
tilted	left side raised, 15° angle with the floor
tall	height = 0.35 m
short	height = 0.05 m
smooth and tilted	10° angle, friction = 0.1
rough and tilted	10° angle, friction = 1
Table	
default	height = 0.55 m
short	height = 0.75 m
tall	height = 0.35 m
Character	
default	weight = 40.84 kg
big arms	weight = 52.65 kg
big legs	weight = 47.31 kg
big belly	weight = 64.95 kg

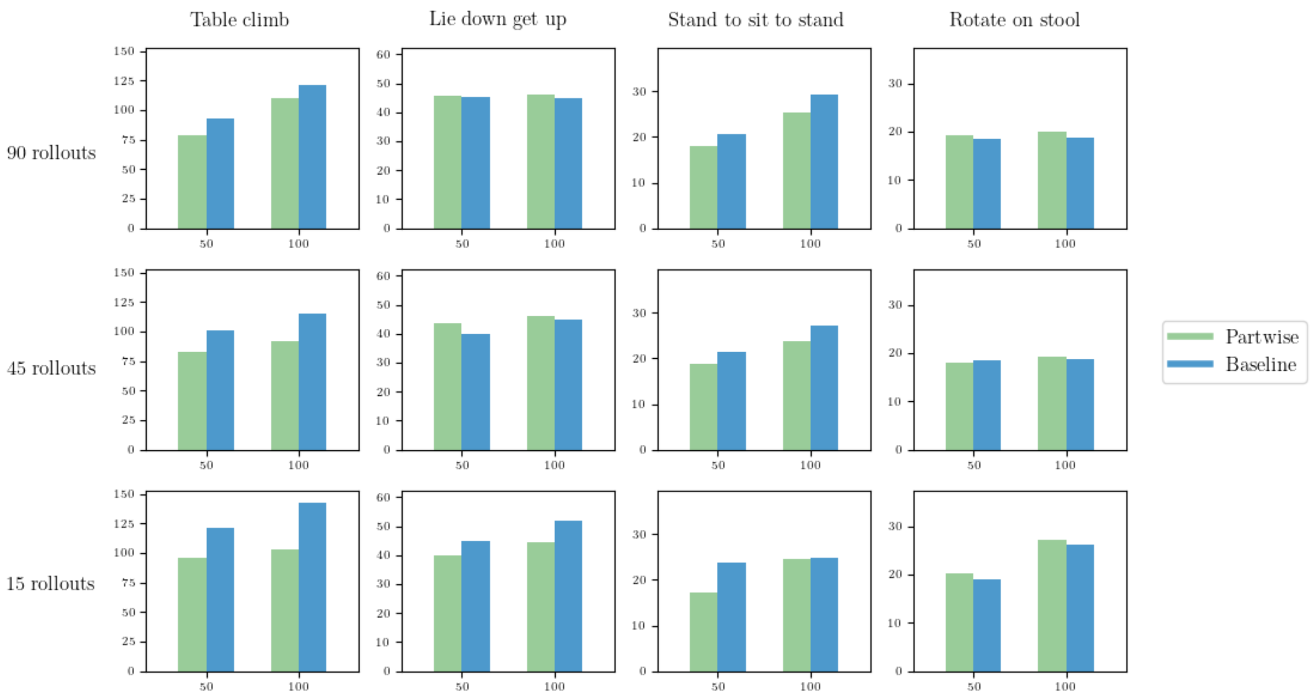
Table 3: *Generalization details.*

Appendix C: Baseline Comparisons

See next page.



(a) *A comparison of success rates (%). A motion succeeds only if all contact keyframes meet their success criterion within their designated time limit, T_{max}. (§3.3)*



(b) *A comparison of average cost per simulation timestep.*

Figure 12: *Each chart compares the performance of partwise and baseline, across four skills (columns), 15, 45, or 90 rollouts (rows) and for slowdown factors of 50% ($\Gamma = 0.5$) and 100% ($\Gamma = 1.0$), as noted along the x-axis.*