Environment Warped Gait Trajectory Optimization for Complex Terrains

Zherong Pan[1], Tan Chen[2], Xianzhong Fang[3], Timothy Bretl[4], Xifeng Gao[1], Kui Wu[1]

***Abstract*—Contact-aware gait trajectory optimization is a challenging non-convex programming problem, especially for complex terrain shapes, where prominent numerical algorithms can fail to find a solution or fall into local minima. To alleviate this issue, we propose an environment warping technique that changes the coordinates for decision variables. Given a terrain of some general shape, our method first generates a locally injective, as-conformal-as-possible function that maps the ambient space around the terrain to a warped space. We then formulate the trajectory optimization in the warped space by remapping all the decision variables. Our method frees the numerical optimizer from tuning the trajectories to fit changing terrain shapes, leading to better numerical conditioning and fewer local minima. Numerical results show that our method outperforms direct trajectory optimization in terms of both success rates and quality of solutions.**

***Index Terms*—Contact-aware Locomotion, Trajectory Optimization, Environment Modeling**

I. Introduction

GAIT and trajectory generation is a core component in the robot design, planning, and control loop for legged robots. Recent advances focus on contact-aware trajectory optimization, which mostly relies on gradient-based methods [1] to find feasible and locally optimal motion plans. Starting from a still pose far from the goal, the state-of-the-art methods [2, 3, 4, 5] in contact-aware trajectory optimization can generate complex motions to reach the goal pose by simultaneously searching for robot gaits, contact points, and forces in a joint, high-dimensional search space over a long horizon. Despite their ability to scale to high dimensions, local optimizers [1] can fail to find a solution or fall into sub-optimal minima.

The issue of optimization failure is much more substantial in a contact-aware setting [2, 3, 4, 5] than in non-contact-aware cases [6, 7, 8] due to the additional complementary constraints. Even worse, robot needs to traverse on uneven terrains, as illustrated in Figure 1. The existence of irregular terrain shapes can significantly complicate the landscapes of objective and constraint functions, further decreasing the chances of finding a solution. Additional techniques, such as multi-start optimization [9] and stochastic optimization [7], have been explored to improve the quality of trajectories, but they are costly to be applied in long-horizon contact-aware settings.

We propose environment warping, an approach to improve the robustness of contact-aware trajectory optimization by a change of coordinates. As illustrated in Figure 1, our method warps the ambient space of a curved terrain $\mathcal{A}$ to a warped space $\mathcal{F}$, mapping the terrain to a flat surface. We then formulate a novel contact-aware trajectory optimization based on [10] with all the decision variables pulled back from $\mathcal{A}$ to $\mathcal{F}$. Although a similar idea has been explored in prior work [11], our method is the first to be applied to contact-aware planning problems. When we compare the trajectory optimization baseline [10] formulated in the original space $\mathcal{A}$ and our warped-space $\mathcal{F}$, experiments show that our warped-space formulation achieves a 13 – 100% higher success rate. Our technical contributions include: a new representation of the smooth map, warping ambient space of curved terrains (Section IV-A); a finite-element-based optimization technique to generate locally injective terrain warping function (Section IV-B); and a warped-space formulation for contact-aware trajectory optimization (Section IV-C).

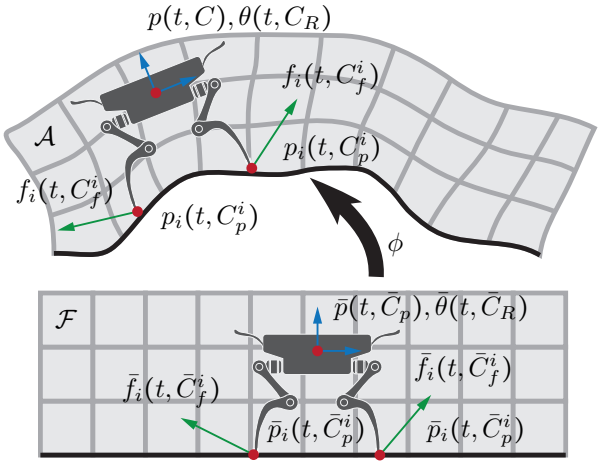


Fig. 1: We warp a flat terrain, as well as a narrow band neighborhood (gray) of the ambient space $\mathcal{F}$, to a curved terrain (black). All the decision variables, including robot torso and contact positions (red), torso orientations (blue), and contact forces (green), can be pulled back from $\mathcal{A}$ to $\mathcal{F}$ via our locally injective function ϕ, leading to our contact-aware trajectory optimization formulated in $\mathcal{F}$.

II. Related Work

Trajectory optimization can be categorized into contact-aware and non-contact-aware settings. A non-contact-aware optimizer searches for sequences of robot gaits [8] or control signals [6, 7]. It is noteworthy that non-contact-aware optimizers can also plan motions through contacts via model predictive control combined with smooth approximate contact models [12] or stochastic optimizations [7, 13]. However, these methods suffer from gradient explosion, preventing them from finding complex motions over a long horizon.

On the other hand, contact-aware trajectory optimization adopts the augmented joint search space of robot gaits, contact points, and forces. These methods use explicit constraints to

Manuscript received June 25, 2022; Revised October 10, 2022; Accepted October 12, 2022. This paper was recommended for publication by Editor Abderrahmane Kheddar upon evaluation of the Associate Editor and Reviewers' comments.

[1]Zherong Pan, Xifeng Gao, and Kui Wu are with the LightSpeed Studios, Tencent-America. {zrpan, xifgao, kwwu}@global.tencent.com

[2]Tan Chen is with the Department of Electrical and Computer Engineering at Michigan Technological University. tanchen@mtu.edu

[3]Xianzhong Fang is with Ningbo University. fangxianzhong@nbu.edu.cn

[4]Timothy Bretl is with the Coordinated Science Laboratory at the University of Illinois Urbana-Champaign. tbretl@illinois.edu

enforce the regularity of contacts, i.e., the requirement that external forces can only appear when the robot touches the environments [2, 3, 4, 5]. With the augmented formulation, constraints become differentiable, locally supported, and do not suffer from ill-conditioned gradients, enabling optimizers to discover motions with many contact state changes. An inherent downside of such formulation is a high-dimensional search space, where only locally optimal solutions can be found within a tractable computational budget. Modern interior point optimizers can take hours to find long trajectories of full body motions [2, 4] even on the GPU [5]. Even worse, optimizers can oftentimes fail to find feasible solutions or converge to sub-optimal motions, which correspond to robots taking unnecessarily small steps or performing unstable maneuvers.

Along with this work, there are multiple efforts in improving the robustness of local trajectory optimizations. Several prior works [9, 14] propose to warm-start optimizers from well-conditioned initial guesses. However, none of these methods can be applied in the contact-aware setting. Recently, sampling-based motion planner and trajectory optimizer has been combined in planning contact-aware motions [15, 16], but their trajectory optimizers assumes fixed contacts. Another approach is phase-based optimization [10, 17], which restricts the solutions to a well-conditioned subspace. These methods restrict robot end-effectors in prescribed contact mode sequences, freeing optimizers from handling the difficult complementary constraints. The work most closely related to ours is [11], which warps the workspace and formulates trajectory optimization in the parametric space without considering contacts. With marginal overhead, Mainprice *et al.* [11] showed that optimizers in a warped space have better conditions and faster convergence. In comparison to [11], we propose a series of new techniques allowing the warping techniques to be applied to the contact-aware setting.

Environment warping belongs to a broader category of researches on surface and volume parameterization [18, 19, 20]. These methods were originally designed to discretize and numerically solve partial differential equations or generate detailed appearances in virtual environments. Recently, they were rediscovered in the robotic community [11, 21] and used to parameterize the workspace of motion planning problems. These methods rely on the parameterization of the infinite ambient space via the solution of exterior Laplace equation using the boundary element method [21]. This technique is not only time-consuming to compute, but its solution is represented as a boundary integral that is also singular on the domain boundary itself. These properties make prior methods [11, 21] inappropriate for contact-aware trajectories. We tackle these problems by discretizing the ambient space using the finite element method with high-order continuous shape functions. A large body of researches such as [22] have contributed to finite-element-based surface parameterization. As a key advantage, finite element methods can deal with a larger variety of objective functions than the boundary element methods. For our application, we will show that angle-preserving conformal warping [18] is better than a smooth warping produced by solving the Laplace equation [21], which can only be realized via the finite element method. Additionally, some objectives can ensure locally or even globally injective mapping functions [22, 23]. In a parallel effort to tackle complex environments, animators use motion editing [24, 25, 26] to warp a reference motion to adapt to modified environments. But these methods are not suitable for robotic applications, since their generated trajectories do not satisfy the equations of motion.

Riemannian motion policies (RMP) [27, 28] are a series of research studying the relationship among smooth manifolds. Key to these methods is the pushforward and pullback operators, that bring tangent bundles, functions, and metrics through smooth maps. Making use of the linearity of tangent spaces, multiple Riemannian policies are inherently composable. This leads to the follow-up researches on composable and learning-based RMP [29], which is orthogonal to our research. We heavily use the pushforward and pullback operators in formulating our contact-aware trajectory optimizations. Since we need to handle multiple types of decision variables, we carefully control the order of continuity of our smooth map ϕ to ensure all the variables are sufficiently smooth to be properly handled by gradient-based optimizers. In order to represent robot orientations in $\mathrm{SO}(3)$, we further introduce an operator projecting a point of the ambient space to its closest neighbor on $\mathrm{SO}(3)$.

III. Contact-aware Trajectory Optimization

Throughout our paper, we use uppercase letters for matrices, functions, and constants, and lowercase letters are for scalars and vectors. Our method is built off of the phase-based gait trajectory optimization [10] and we briefly review their formulation in this section. This method features the Centroidal Dynamics Model (CDM) and phase-based contact parameterization.

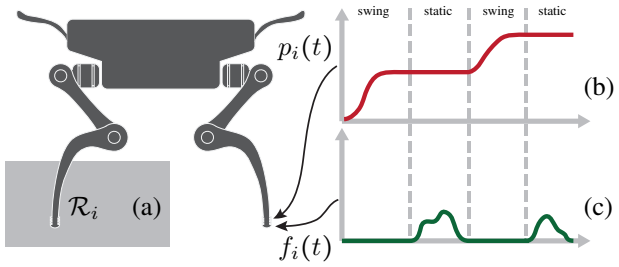


Fig. 2: We illustrate some key notions in phased-based gait trajectory optimization. (a): A simplified box-shaped reachable set R_i (gray) is identified for each robot end-effector. Each end-effector has a position sequence (b) and a force sequence (c). The sequences are divided into alternating static and swing phases and the switching times (dashed line) are decision variables.

A. Robot Trajectory Representation

A robot is modeled as a torso with orientation trajectory $R(t)$ and translation trajectory $p(t)$. The robot has a number of E end-effectors located at $p_i(t)$ exerting contact forces $f_i(t)$, where $i = 1, \cdots, E$ and $t \in [0, T]$ is the time variable. The torso position trajectory is parameterized using high-order splines with control points c_p, denoted as $p(t) \triangleq p(t, c_p)$. The torso orientation is represented using three Euler angles θ, which in turn is parameterized using high-order splines with

control points c_R, denoted as $R(t) \triangleq R(\theta(t, c_R))$. Key to the phase-based formulation is an alternating parameterization of $p_i(t)$ and $f_i(t)$. Each robot contact trajectory is divided into an alternating sequence of static and swing phases. During the static phase, $p_i(t)$ lies statically on the ground and $f_i(t)$ can be non-zero, while during the swing phase, $p_i(t)$ can freely move above the ground while $f_i(t)$ must be zero. As a result, the position-force complementarity is implicitly encoded without requiring additional constraints. Such a parameterization can be realized by two alternating sequences of high-order splines as illustrated in Figure 2b. Winkler *et al.* [10] further improves the flexibility by treating the time spans of each static and swing phase as additional decision variables, under the constraints that all the time spans are positive and sum up to T. We denote the control points as well as the time spans for the ith contact point and force sequence as c_p^i and c_f^i, respectively. As a result, we have $p_i(t) \triangleq p_i(t, c_p^i)$ and $f_i(t) \triangleq f_i(t, c_f^i)$. We denote $c \triangleq \left(c_R, c_p, c_p^i, c_f^i\right)$ as a concatenation of all the decision variables.

B. Contact-aware Trajectory Optimization

Our trajectory optimization baseline takes the following form:

$$\underset{c}{\text{argmin}}\ E(c) \quad \text{s.t.} \tag{1}$$

$$\begin{cases} \forall t \in \mathcal{T}_d: \quad m\ddot{p}(t) = \sum_{i=1}^{E} f_i(t) + mg \\ I\dot{\omega}(t) + \omega(t) \times I\omega(t) = R(t)^T \sum_{i=1}^{E} (p_i(t) - p(t)) \times f_i(t) \end{cases} \tag{2}$$

$$\begin{cases} \forall t \in \mathcal{T}^i_{\text{static}}(c_p^i): \quad z^T p_i(t, c_p^i) = h(p_i(t, c_p^i)) \\ \forall t \in \mathcal{T}^i_{\text{swing}}(c_p^i): \quad z^T p_i(t, c_p^i) \geq h(p_i(t, c_p^i)) \end{cases} \tag{3}$$

$$\forall t \in \mathcal{T}_d: \quad R(t)^T (p_i(t) - p(t)) \in \mathcal{R}_i \tag{4}$$

$$\forall t \in \mathcal{T}_d: \quad \left\| \left[I - n_i(t) n_i(t)^T \right] f_i(t) \right\| \leq \nu n_i(t)^T f_i(t), \tag{5}$$

which handles three types of integrity constraints ensuring physical, geometric, and force correctness. The physical constraints Equation (2) ensure the robot torso complies with the Newton-Euler's equation, where I is the inertia tensor and $\omega(t)$ is the angular velocity, both measured at the local frame of reference. m is the robot torso weight and g is the gravitational coefficient. Equation (2) is tested at fixed time intervals denoted as a discrete set $\mathcal{T}_d$. Our environment is modeled as a heightfield $h(x, y) : \mathbb{R}^2 \to \mathbb{R}$, mapping a horizontal point to its vertical ground height. The geometric constraints Equation (3) require that robot end-effectors must make contacts with the heightfield during the static phase. While during the swing phase, the end-effectors cannot penetrate the terrain. Equation (3) are tested at a set of discrete time instances $\mathcal{T}^i_{\text{static}}$ and $\mathcal{T}^i_{\text{swing}}$. Note that, since timing of phase-changes could be optimized in [10], $\mathcal{T}^i_{\text{static}}$ and $\mathcal{T}^i_{\text{swing}}$ are functions of c_p^i. Further, each robot end-effector p_i must be within a reachable set $\mathcal{R}_i$ relative to the torso (Figure 2), which is formulated in Equation (4) and tested at a fixed set of time instances $\mathcal{T}_d$, same as those for Equation (2). Finally, the force constraints Equation (5) require that contact forces lie in the frictional cone during each static phase, where ν is the frictional coefficient and we again use the same test set $\mathcal{T}_d$. $n(t)$ is the normal direction of terrain calculated as:

$$n(t) \triangleq \left(-\nabla_x h, -\nabla_y h, 1\right) / \left\| \left(-\nabla_x h, -\nabla_y h, 1\right) \right\|.$$

Additionally, $E(c)$ could be additional cost functions formulating the control task. Equation (1) is solved by gradient-based optimizers, which require all the objective and constraint functions to be at least differentiable. As a result, the heightfield must be sufficiently smooth, which can be achieved by fitting the heightfield via spline patches as introduced in the following section.

IV. Environment Warped Trajectory Generation

In differential geometry, the pullback operator maps a function from one manifold to the other via a smooth map. Our method could be interpreted as a pullback operator ϕ for the optimization formulation in Equation (1), from the ambient space $\mathcal{A}$ to a warped space $\mathcal{F}$. We first introduce how the smooth map ϕ is defined in Section IV-A, IV-B, and then introduce the pullback operator in Section IV-C.

A. Finite Element Environment Warping Function

Prior work [11] proposes to use an $\mathcal{A}$-space regular grid to resample the solution of the exterior Laplace equation, which warps the entire ambient space. Instead, we use a $\mathcal{F}$-space regular grid to discretize only a narrow band neighborhood above the terrain $h(x, y)$ as illustrated in Figure 1. We refer readers to [30, 31] for basics of high-order finite element methods. We use tri-cubic B-spline basis as our shape function, so our grid consists of $N_x \times N_y \times N_z$ tri-cubic, B-spline cells and $(N_x + 3) \times (N_y + 3) \times (N_z + 3)$ control points. Each control point is associated with a 3D mapped point denoted as $\psi_{ijk}, i = 1, \cdots, N_x + 3, j = 1, \cdots, N_y + 3, k = 1, \cdots, N_z + 3$ and a shape function $\mathcal{B}_{ijk}(x) : \mathbb{R}^3 \to \mathbb{R}$. For any $x \in [0, N_x] \times [0, N_y] \times [0, N_z]$, we can then define our smooth map $\phi(x)$ as:

$$\phi(x) = \sum_{i=1}^{N_x+1} \sum_{j=1}^{N_y+1} \sum_{k=1}^{N_z+1} \psi_{ijk} \mathcal{B}_{ijk}(x).$$

We further define the associated Jacobian matrix as $J(x) \triangleq \nabla_x \phi$, which is used to define the pushforward operator. Our smooth map endows several properties making it suitable for defining the pushforward and pullback operators in contact-aware trajectory optimizations. First, ϕ can be efficiently computed since the shape function $\mathcal{B}(x)$ is only locally supported with non-zero values only on a $4 \times 4 \times 4$ neighboring grid. Second, our shape function and thus $\phi(x)$ is C^2-continuous over the entire $\mathbb{R}^3$. We will show that C^2-continuity is just enough for all the constraints to have well-defined Jacobian matrices. Third, although we only discretize a narrow band neighborhood, $\phi(x)$ is well-defined and differentiable outside our narrow band. This property is important when used with infeasible optimization algorithms, which allow solutions to be temporarily outside the feasible domain. This is the case with most off-the-shelf optimization algorithms [1].

B. Warping Function Optimization

The finite element method parameterizes the mapping function using the control points ψ_{ijk}, which are our decision vari-

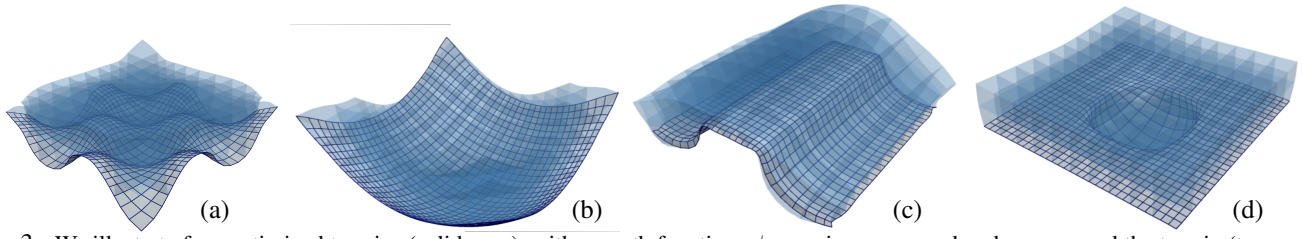


Fig. 3: We illustrate four optimized terrains (solid gray), with smooth functions ϕ mapping a narrow band area around the terrain (transparent blue). We set the band thickness to be 5 times that of the robot height at its rest pose. The bottom of ϕ is aligned with the terrain $h(x, y)$ and illustrated using solid color.

ables. And we define ψ_{ijk} as the minimizer of the following integrated energy:

$$E_{\text{map}} \triangleq \int_{[0,N_x]\times[0,N_y]\times[0,N_z]} \rho(x, \psi_{ijk}) dv,$$

where ρ is the energy density function. In practice, evaluating the above integral for a general, analytic function ρ is intractable and we use Gauss quadrature for an approximation. Unlike the boundary element method, which can only handle ρ with known fundamental solutions, the finite element method allows much more flexibility in choosing ρ. For example, if we choose ρ as the Dirichlet energy [32], then the Laplace equation is recovered. Instead, since we need to represent robot orientations, it is best for ϕ to preserve angles, i.e., the pushforward operator is closed under SO(3). Smooth maps with this property are denoted as conformal maps. Although perfectly conformal maps are not realizable in 3D, we could use the following energy to make ϕ as conformal as possible, which was originally proposed in [33]:

$$\rho(x, \psi_{ijk}) \triangleq \|J - UV^T|\Sigma|^{1/3}\|^2 - \mu \sum_{i=1}^{3} \log(\Sigma_i),$$

where we denote the SVD decomposition of the Jacobian as: $J = U\Sigma V^T$ and the ith singular value as Σ_i. The second term above ensures the Jacobian to have positive singular values and the smooth map ϕ is locally injective [34], where μ is a small positive weight of log-barrier functions. We further need the bottom of our mapped space to be aligned with the given terrain $h(x, y)$. This is achieved using a terrain matching objective function:

$$E_{\text{match}} \triangleq \int_{[0,N_x]\times[0,N_y]\times\{0\}} \|\phi_2(x) - h(\phi_0(x), \phi_1(x))\|^2 ds,$$

where $\phi_i(x)$ is the ith element of $\phi(x)$. The above integral is also approximated using Gauss quadrature. Putting things together, we define:

$$\psi^*_{ijk} = \underset{\psi_{ijk}}{\text{argmin}}\ E_{\text{map}} + E_{\text{match}}. \tag{6}$$

We propose two techniques to solve for the optimal ψ^*_{ijk}. If $\mu = 0$ then the optimization can be solved in an alternating local-global manner similar to [35], which is summarized in Algorithm 1. The Algorithm 1 is guaranteed to converge because the energy is monotonically decreasing over iterations. This method ignores the gradient of $UV^T|\Sigma|^{1/3}$ and $h(\phi_0(x), \phi_1(x))$ with respect to ψ_{ijk}, leading a quadratic objective function with a fixed lefthand side. As a result, the Hessian matrix only needs to be factorized once and the iterative cost becomes extremely low. Some examples of optimized maps are illustrated in Figure 3. If $\mu > 0$, then a full blown Gauss-Newton is used with explicit line search to ensure $J(x)$ lies in the feasible domain with positive singular values Σ_i.

Algorithm 1: Local-Global Optimization of ψ_{ijk}

for iteration $k = 1, 2, 3, \cdots$ **do**
  for each Gauss quadrature sample point x **do**
    $J^*(x) \leftarrow UV^T|\Sigma|^{1/3}$ ▷ Local
  for each Gauss quadrature sample point x **do**
    $h^*(x) \leftarrow h(\phi_0(x), \phi_1(x))$ ▷ Local
  $\psi^*_{ijk} \leftarrow \underset{\psi_{ijk}}{\text{argmin}} \begin{cases} \int \|J(x) - J^*(x)\|^2 dv + \\ \int \|\phi_2(x) - h^*(x)\|^2 ds \end{cases}$ ▷ Global

C. Gait Trajectory Optimization in Warped Space

In this section, we use our map ϕ to define pullback operators for each constraint in Section III. We use $\bar{\bullet}$ to denote variables in the warped space $\mathcal{F}$. The warped-space trajectory optimization is formulated as follows:

$$\underset{\bar{c}}{\text{argmin}}\ E(\bar{c}) \quad \bar{c} \triangleq \left(\bar{c}_R, \bar{c}_p, \bar{c}^i_p, \bar{c}^i_f\right) \quad \text{s.t.} \tag{7}$$

$$\begin{cases} \forall t \in \mathcal{T}^i_{\text{static}}(\bar{c}^i_p): & z^T \bar{p}_i(t, \bar{c}^i_p) = h(\bar{p}_i(t, \bar{c}^i_p)) \\ \forall t \in \mathcal{T}^i_{\text{swing}}(\bar{c}^i_p): & z^T \bar{p}_i(t, \bar{c}^i_p) \geq h(\bar{p}_i(t, \bar{c}^i_p)) \end{cases} \tag{8}$$

$$\forall t \in \mathcal{T}_d: \quad \left\|\left[I - zz^T\right] \bar{f}_i(t, \bar{c}^i_f)\right\| \leq \nu z^T \bar{f}_i(t, \bar{c}^i_f) \tag{9}$$

$$\begin{cases} \forall t \in \mathcal{T}_d: & R(\bar{\theta}(t, \bar{c}_R))^T \mathcal{P}\left[J(\bar{p}(t, \bar{c}_p))\right]^T \\ & \left(\phi(\bar{p}_i(t, \bar{c}^i_p)) - \phi(\bar{p}(t, \bar{c}_p))\right) \in \mathcal{R}_i \end{cases} \tag{10}$$

$$\begin{cases} & \sum_i f_i = \frac{M}{\Delta t^2}(p(t) - 2p_- + p_{--}) \\ & \sum_i (p_i(t) - p(t)) \times f_i(t) \\ = & \frac{\rho}{\Delta t^2} \int_{x\in\Omega} (2R_- x - R_{--} x) \times R(t) x dv \end{cases}, \tag{11}$$

and we explain each constraint below.

a) Warped constraint Equation (3)*:* Corresponding to the end-effector position sequence $p_i(t, c^i_p)$ in $\mathcal{A}$, we introduce warped-space force sequence $\bar{p}_i(t, \bar{c}^i_p)$ in $\mathcal{F}$. Since the terrain is mapped to the $z = 0$ plane and the spline $\bar{p}_i$ is linear in its control points $\bar{c}^i_p$, position correctness constraints take the linear form of Equation (8).

b) Warped constraint Equation (5)*:* We define force sequence $\bar{f}_i(t, \bar{c}^i_f)$ in tangent space $\mathcal{T}_{\mathcal{F}}(\bar{p}_i(t, \bar{c}^i_p))$ corresponding

to $f_i(t, c_f^i)$ in $\mathcal{T}_{\mathcal{A}}(\phi(\bar{p}_i(t, \bar{c}_p^i)))$. The force sequence maps to $\mathcal{T}_{\mathcal{A}}$ via the pushforward operator:

$$f_i(t) \triangleq J(\bar{p}_i(t, \bar{c}_p^i))\bar{f}_i(t, \bar{c}_f^i).$$

Similarly, the normal direction is mapped to $\mathcal{T}_{\mathcal{A}}$ as:

$$n_i(t) \triangleq J(\bar{p}_i(t, \bar{c}_p^i))z / \left\| J(\bar{p}_i(t, \bar{c}_p^i))z \right\|.$$

Plugging these operators into Equation (5) and we derive the pullback of force correctness constraint. However, we argue for a more succinct form of constraint when ϕ is made as conformal as possible. In this case, the Jacobian $J(\bar{p}_i(t, \bar{c}_p^i))$ is nearly a scaled rotation matrix: $J \approx UV^T|\Sigma|^{1/3}$. If we replace J with such approximation in Equation (5), we derive the constraint Equation (9), which is convex because the normal direction z is a constant and the spline $\bar{f}_i$ is linear in $\bar{c}_f^i$. Such a constraint can be handled more efficiently by modern optimizers than the non-convex Equation (5). Although Equation (9) is not an exact pullback operator since our smooth map is not perfectly conformal, we found empirically the difference is rather small as shown in Figure 4.

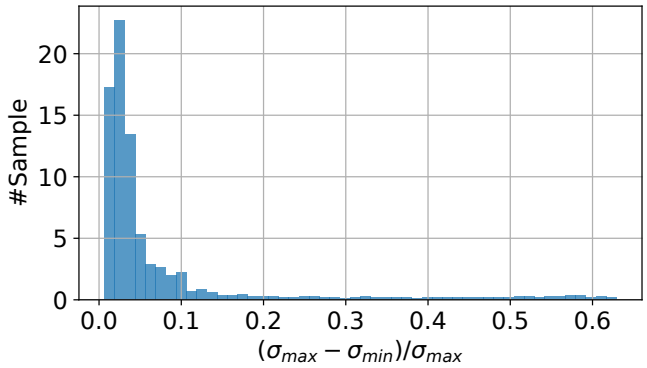


Fig. 4: We plot the relative conformal error over a set of regularly sampled points on the terrain in Figure 3a, estimated by the relative difference between the maximal (σ_{max}) and minimal singular value (σ_{min}) of J. Most sample points has an error of less than 5% and the average error is 7.8%.

c) Warped constraint Equation (4)*:* Both physics correctness and reachability constraints involve the robot torso position $p(t)$ and orientation $R(t)$. For the torso position, we introduce the sequence $\bar{p}(t, \bar{c}_p)$ in $\mathcal{F}$. In order to pushforward the orientation R, we consider an arbitrary vector $v \in \mathcal{T}_{\mathcal{F}}$ that is brought to $Jv \in \mathcal{T}_{\mathcal{A}}$. An orientation R can be considered as a matrix of three orthogonal vectors, which is brought to JR under the pushforward operator. However, the matrix $JR \notin \mathrm{SO}(3)$ in general. Therefore, we introduce an additional projection operator $\mathcal{P}$. Given any $J \in \mathbb{R}^{3\times3}$, $\mathcal{P}(J)$ maps J to the nearest element of $\mathrm{SO}(3)$ in the following optimal sense:

$$\mathcal{P}(J) \triangleq \underset{R\in\mathrm{SO}(3)}{\operatorname{argmin}} \|R - J\|_F^2. \tag{12}$$

The above optimization has the following properties leading to the efficient computation of projection and its derivatives:

Corollary IV.1. *Suppose the SVD decomposition of matrix J is $J = U\Sigma V^T$. The projection operator has closed form solution $\mathcal{P} = UV^T|UV^T|$. For any $R \in SO(3)$, $\mathcal{P}(JR) = \mathcal{P}(J)R$. Finally, we have a closed form Jacobian of $\mathcal{P}$ as follows:*

$$\frac{\partial \mathcal{P}(J)}{\partial \alpha} \triangleq \omega \times \mathcal{P}(J)$$
$$\omega \triangleq U(tr(\Sigma)I - \Sigma)^{-1}V^T \left[\mathcal{P}(J)^T \frac{\partial J}{\partial \alpha} - \frac{\partial J}{\partial \alpha}^T \mathcal{P}(J) \right]_{\times^{-1}},$$

where α is some arbitrary variable of J and $[\bullet]_{\times^{-1}}$ is the inverse cross product operator such that $[\bullet]_{\times^{-1}} \times = \bullet$.

Proof. The first result is due to [36]. The second result is immediate because the SVD decomposition of JR is $JR = U\Sigma V^T R$. The third result is due to [37]. □

With the projection operator, we define the robot torso rotation sequence in $\mathcal{F}$ as:

$$R(t) \triangleq \mathcal{P}\left[J(\bar{p}(t, \bar{c}_p))\right] R(\bar{\theta}(t, \bar{c}_R)). \tag{13}$$

In other words, we first use Euler angles to define the rotation in $\mathcal{F}$, pushing it forward to some element of $\mathcal{T}_{\mathcal{A}}(\phi(\bar{p}(t, \bar{c}_p)))$ and then project it back to $\mathrm{SO}(3)$. Note that our smooth map ϕ is C^2-continuous and the rotation $R(t)$ and force $f_i(t)$ already depends on the Jacobian J, so $R(t)$ and $f_i(t)$ is only C^1-continuous. Plugging Equation (13) into the reachability constraint Equation (4), we derive its pullback form in Equation (10), which has well-defined Jacobian. However, the pullback of physics correctness constraints is more involved.

D. Position-Based Centroid Dynamics Model

In a similar fashion as Equation (10), we could pullback the Newton-Euler Equation (2). However, such pullback constraint would require $R(t)$ to be C^2-continuous when computing the constraint Jacobian, which could not be achieved via ϕ discretized using cubic basis functions. To obtain sufficient smoothness, one option would be using quartic basis functions, but this could significantly increase computational overhead of the smooth map optimization procedure, as well as all constraint evaluations. Instead, we follow [5] and propose a Position-based Centroid Dynamics Model (PCDM), which only requires C^1-continuity.

Since the physics constraint is tested at a regular interval $t \in \mathcal{T}_d$. We can denote the sampling interval as Δt and use the following shorthand notations:

$$\bullet_- = \bullet(t - \Delta t) \quad \bullet_{--} = \bullet(t - 2\Delta t), \tag{14}$$

where $\bullet$ is an arbitrary variable. PCDM assume that the physically correct configuration at time instance t is the minimizer of the following combined inertial and potential energy:

$$R, p = \underset{R\in\mathrm{SO}(3),p}{\operatorname{argmin}} - \sum_i f_i^T p_i(t) + \frac{\rho}{2\Delta t^2} \int_{x\in\Omega} \|(Rx + p) - 2(R_- x + p_-) + (R_{--}x + p_{--})\|^2 dv,$$

where Ω is the volume occupied by the robot torso in its frame of reference. By setting the gradient to be zero in the tangent bundle of $\mathrm{SE}(3)$, we derive the PCDM governing Equation (11), which only requires $R(t), p(t)$ to be C^1-continuous. The integral in Equation (11) can be evaluated

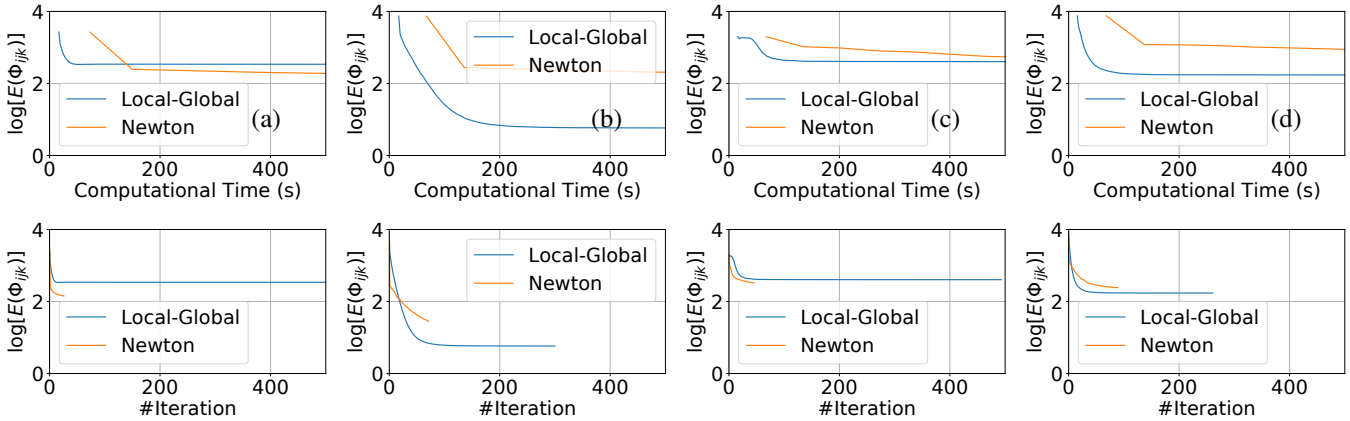


Fig. 5: The convergence history of the two methods, local-global method (Algorithm 1) and Newton's method, for the four results in Figure 3, where we use the same cost function with $\mu = 0$. The Newton's method can sometimes converge to better solutions but its iterative cost is much higher than that of Algorithm 1, requiring matrix factorization during each iteration.

in the same way as the inertia tensor. In summary, the cubic basis function for the smooth map ϕ is just enough for all the constraints in Equation (7) to have a well-defined Jacobian. Some optimizers require the Hessian of Lagrangian function, which is computed via low-rank approximation.

V. Evaluations

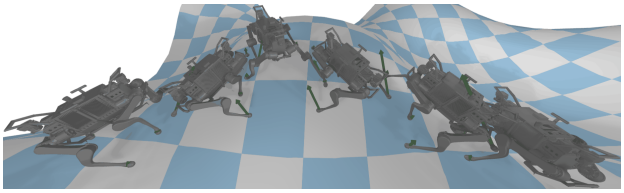
We refer readers to our video for all the animated results. We implement the baseline optimization (Equation (1)) and our reformulated optimization (Equation (7)). For fairness of comparison, we also use PCDM as physics constraints for the baseline formulation. All experiments are performed in a simulated environment using the ANYmal robot model. We extract the largest rectangular reachable set to be used as $\mathcal{R}_i$ and use Knitro [38] as the optimizer backend. We set the maximal number of iterations to be 200 and the optimizer terminates early if the maximal constraint violation is smaller than 10^{-4} or the relative solution change is smaller than 10^{-4} over 5 consecutive iterations. All experiments are performed on a desktop machine with a 6-core Intel i7-10750H CPU and 8Gb memory. We use multi-thread to accelerate the computation of constraints and their Jacobian. We set the parameters to be $T = 10$s, $\nu = 0.75$, and $\Delta t = 0.05$s. All the trajectories, including position, orientation, and force, are discretized using cubic B-splines. On average over all examples, a trajectory of 10 seconds can be computed in around 1 minute.

We first profile the performance of smooth map ϕ optimization (Equation (6)) and compare the two solutions: local-global (Algorithm 1) and Newton's approach. As illustrated in Figure 5, the local-global approach converges much faster than the Newton's approach. Although the local-global approach can generate non-injective maps, we have not observed this artifact in our four benchmarks of Figure 3. However, we propose to use the local-global approach first and only revert to the more costly Newton's method if injectivity is violated. For most examples, the smooth map ϕ can be optimized within 200s. In order to ensure that the robot never leaves the narrow band ambient space, we set the band thickness to be 5 times that of the robot height at its rest pose, as illustrated in Figure 3.

Our first benchmark (Figure 6) uses the rippled terrain (Figure 3a). We initialize the robot in the center of this terrain $(0, 0)$ and have the robot move to the horizontal point $(r \sin\theta/2, r\cos\theta/2)$ where r is half the size of the terrain. The target position is added as a hard constraint on the X-Y plane. We sample θ from $0°$ to $360°$ at an interval of $15°$. For each θ, we run optimization for a maximum of 200 iterations and plot the per-iteration constraint violation in Figure 8. We denote a trajectory as successful if the maximal constraint violation is less than 10^{-4} within 200 iterations. Under this criterion, our method achieves a success rate of 92%, which is significantly higher than 48% using the baseline formulation. We further compare the computational overhead of the two methods in Figure 9. Except for $1-2$ walking directions, our method incurs a lower computational overhead by using fewer iterations to converge. Our second benchmark (Figure 7) uses the conic terrain (Figure 3b). We keep all other settings the same except that robots are horizontally initialized at $(-r\sin\theta/2, -r\cos\theta/2)$, so the robot needs to walk over a longer distance. In this more challenging case, the baseline formulation achieves a success rate of 0% as compared with 100% using our method.

Our third benchmark (Figure 10a) uses the hilly terrain in Figure 3c. The slope of the hill in the middle is rather steep. Therefore, we allow the robot to climb the hill by modeling its two front legs with suckers, allowing it to apply adhesive forces by removing two of the frictional cone constraints. We uniformly sample 15 target positions uphill. The success rate of our method and the baseline are 100% and 0%, respectively. Indeed, it is difficult for low-order polynomial curves to fit the drastically changing terrains, while our method allows a straight line in the warped space to align with the environment in the ambient space. We plot the average per-iteration constraint violation of one trajectory in Figure 11. Similarly, our fourth benchmark (Figure 10b) uses the terrain in Figure 3d, which has a deep pit in the middle, with a much sharper cliff than that of the hill in Figure 3c. We uniformly sample 15 target positions outside the pit and our method

(a): Terrain in Figure 3a



(b): Terrain in Figure 3a

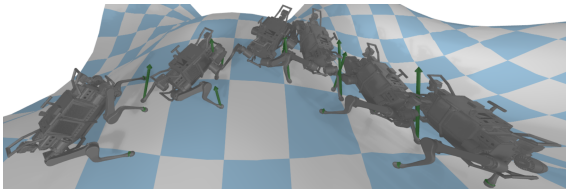


Fig. 6: An exemplary trajectory of the robot walking on a rippled terrain (Figure 3a), with contact forces in green, using our method (a) and the baseline (b).

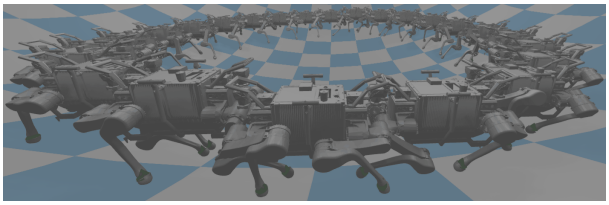


Fig. 7: We computed robot walking along all directions to evaluate the success rate using terrains in Figure 3b.

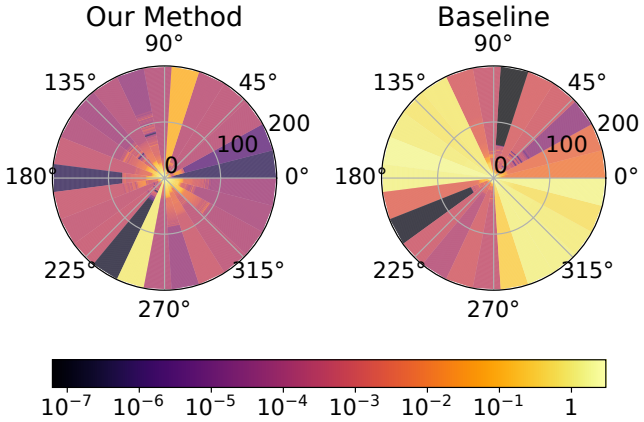


Fig. 8: For the terrain in Figure 3a, we generate trajectories for the robot to walk in all directions ($\theta \in [0, 360]$ with a sampling interval of 15°). For each optimization, we run a maximum of 200 iterations and plot the constraint violation of each iteration (radius). We consider a trajectory successful if the maximal constraint violation is less than 10^{-4} within 200 iterations. Our method achieves a success rate of 92% as compared with 48% using the baseline.

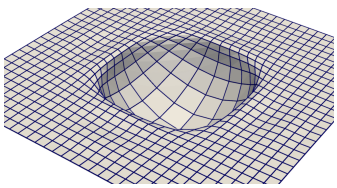
achieves a success rate of 13%, as compared with 0% using the baseline. We summarize the success rate of both methods in Table I.

Terrain	Figure 3a	Figure 3b	Figure 3c	Figure 3d
Our Method	92%	100%	100%	13%
Baseline	48%	0%	0%	0%

TABLE I: Success rate of both methods on our 4 benchmarks.

VI. Conclusion & Limitations

We present an environment warping technique for contact-aware trajectory optimization of legged robots on complex terrain shapes. Our method *pulls*



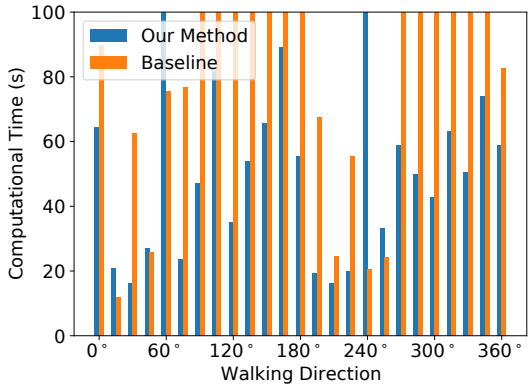


Fig. 9: Computational time for generating a trajectory of 10 seconds with different walking directions. Most problem instances can be solved within 60s. Some instances require much higher computational time, which is clamped in the plot.

(a): Terrain in Figure 3c

(b): Terrain in Figure 3d

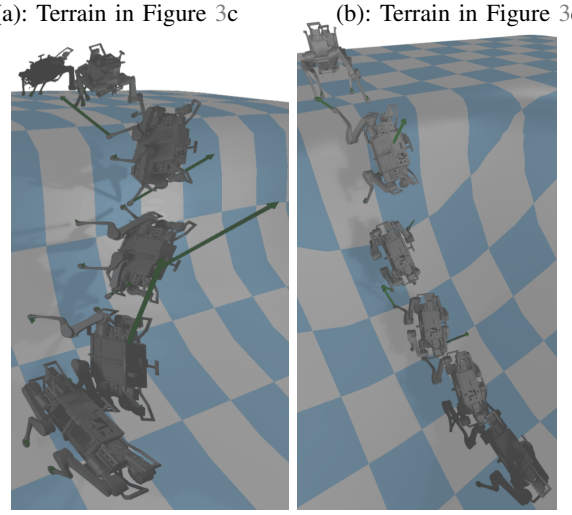


Fig. 10: (a): Robot climbing up the hill (Figure 3c); (b): robot climbing out of the pit (Figure 3d), where the two front legs are capable of applying adhesive forces.

back the objective and constraint functions from the ambient space back to a warped space while pushing forward force and rotational variables from the warped space to the ambient space. As our major limitation, we represent terrains using a smooth surface, which is presumably the reason for our method to achieve only a success rate of 13% on the terrain shown in Figure 3d, which has a discontinuous geometric shape highlighted in the inset. In the future, we plan to use a piecewise continuous warping function, allowing finitely many geometric gaps. As a result, a mixed discrete-continuous optimizer such as [39] could be used to search for gap-aware

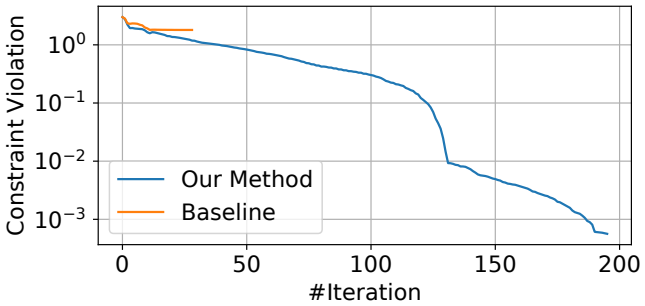


Fig. 11: The iteration-wise constraint violation of two methods for a hill-climbing trajectory. The baseline algorithm terminates early at an infeasible solution.

trajectories. Further, our method cannot support full-body trajectory generation, because our warping function leads to undesirable deformations to a rigid robot link. A workaround for this issue is to use our CDM-based formulation, but derive conservation bounds on the accelerations to guarantee physics realizability.

REFERENCES

[1] S. Leyffer and A. Mahajan, "Nonlinear constrained optimization: Methods and software," *Argonee National Laboratory, Argonne, Illinois*, vol. 60439, 2010.

[2] I. Mordatch, E. Todorov, and Z. Popović, "Discovery of complex behaviors through contact-invariant optimization," *ACM Transactions on Graphics (TOG)*, vol. 31, no. 4, pp. 1–8, 2012.

[3] M. Posa, C. Cantu, and R. Tedrake, "A direct method for trajectory optimization of rigid bodies through contact," *The International Journal of Robotics Research*, vol. 33, no. 1, pp. 69–81, 2014.

[4] J. Schulman *et al.*, "Motion planning with sequential convex optimization and convex collision checking," *The International Journal of Robotics Research*, vol. 33, no. 9, pp. 1251–1270, 2014.

[5] Z. Pan, B. Ren, and D. Manocha, "Gpu-based contact-aware trajectory optimization using a smooth force model," in *Proceedings of the 18th annual ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2019, pp. 1–12.

[6] N. Ratliff, M. Zucker, J. A. Bagnell, and S. Srinivasa, "Chomp: Gradient optimization techniques for efficient motion planning," in *2009 IEEE International Conference on Robotics and Automation*, IEEE, 2009, pp. 489–494.

[7] M. Kalakrishnan, S. Chitta, E. Theodorou, P. Pastor, and S. Schaal, "Stomp: Stochastic trajectory optimization for motion planning," in *2011 IEEE international conference on robotics and automation*, IEEE, 2011, pp. 4569–4574.

[8] C. Park, J. Pan, and D. Manocha, "Itomp: Incremental trajectory optimization for real-time replanning in dynamic environments," in *Twenty-Second International Conference on Automated Planning and Scheduling*, 2012.

[9] T. S. Lembono, A. Paolillo, E. Pignat, and S. Calinon, "Memory of motion for warm-starting trajectory optimization," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 2594–2601, 2020.

[10] A. W. Winkler, C. D. Bellicoso, M. Hutter, and J. Buchli, "Gait and trajectory optimization for legged systems through phase-based end-effector parameterization," *IEEE Robotics and Automation Letters*, vol. 3, no. 3, pp. 1560–1567, 2018.

[11] J. Mainprice, N. Ratliff, and S. Schaal, "Warping the workspace geometry with electric potentials for motion optimization of manipulation tasks," in *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2016, pp. 3156–3163.

[12] Y. Tassa, T. Erez, and E. Todorov, "Synthesis and stabilization of complex behaviors through online trajectory optimization," in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2012, pp. 4906–4913.

[13] E. Theodorou, J. Buchli, and S. Schaal, "A generalized path integral control approach to reinforcement learning," *The Journal of Machine Learning Research*, vol. 11, pp. 3137–3181, 2010.

[14] J. Pan, Z. Chen, and P. Abbeel, "Predicting initialization effectiveness for trajectory optimization," in *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, pp. 5183–5190.

[15] E. Jelavic, F. Farshidian, and M. Hutter, "Combined sampling and optimization based planning for legged-wheeled robots," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 8366–8372.

[16] Y. Ding, M. Zhang, C. Li, H.-W. Park, and K. Hauser, "Hybrid sampling/optimization-based planning for agile jumping robots on challenging terrains," in *2021 IEEE International Conference on Robotics and Automation (ICRA)*, 2021, pp. 2839–2845.

[17] K. Wampler and Z. Popović, "Optimal gait and form for animal locomotion," *ACM Transactions on Graphics (TOG)*, vol. 28, no. 3, pp. 1–8, 2009.

[18] X. Gu and S.-T. Yau, "Global conformal surface parameterization," in *Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, 2003, pp. 127–137.

[19] M. S. Floater and K. Hormann, "Surface parameterization: A tutorial and survey," *Advances in multiresolution for geometric modelling*, pp. 157–186, 2005.

[20] A. Chern, U. Pinkall, and P. Schröder, "Close-to-conformal deformations of volumes," *ACM Transactions on Graphics (TOG)*, vol. 34, no. 4, pp. 1–13, 2015.

[21] H. Wang, K. A. Sidorov, P. Sandilands, and T. Komura, "Harmonic parameterization by electrostatics," *ACM Transactions on Graphics (TOG)*, vol. 32, no. 5, pp. 1–12, 2013.

[22] Z. Jiang, Z. Zhang, Y. Hu, T. Schneider, D. Zorin, and D. Panozzo, "Bijective and coarse high-order tetrahedral meshes," *ACM Transactions on Graphics (TOG)*, vol. 40, no. 4, pp. 1–16, 2021.

[23] C. Schüller, L. Kavan, D. Panozzo, and O. Sorkine-Hornung, "Locally injective mappings," in *Computer Graphics Forum*, Wiley Online Library, vol. 32, 2013, pp. 125–135.

[24] A. Witkin and Z. Popovic, "Motion warping," in *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 1995, pp. 105–108.

[25] R. A. Al-Asqhar, T. Komura, and M. G. Choi, "Relationship descriptors for interactive motion adaptation," in *Proceedings of the 12th ACM SIGGRAPH/Eurographics Symposium on Computer Animation*, 2013, pp. 45–53.

[26] S. Tonneau, R. A. Al-Ashqar, J. Pettré, T. Komura, and N. Mansard, "Character contact re-positioning under large environment deformation," in *Computer Graphics Forum*, Wiley Online Library, vol. 35, 2016, pp. 127–138.

[27] N. Ratliff, M. Toussaint, and S. Schaal, "Understanding the geometry of workspace obstacles in motion optimization," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 4202–4209.

[28] C.-A. Cheng *et al.*, "Rmpflow: A computational graph for automatic motion policy generation," in *International Workshop on the Algorithmic Foundations of Robotics*, Springer, 2018, pp. 441–457.

[29] M. Mukadam, C.-A. Cheng, D. Fox, B. Boots, and N. Ratliff, "Riemannian motion policy fusion through learnable lyapunov function reshaping," in *Conference on robot learning*, PMLR, 2020, pp. 204–219.

[30] P. Solin, K. Segeth, and I. Dolezel, *Higher-order finite element methods*. Chapman and Hall/CRC, 2003.

[31] K. Höllig, *Finite element methods with B-splines*. SIAM, 2003.

[32] K. Crane, F. De Goes, M. Desbrun, and P. Schröder, "Digital geometry processing with discrete exterior calculus," in *ACM SIGGRAPH 2013 Courses*, 2013, pp. 1–126.

[33] Y. Yoshiyasu, W.-C. Ma, E. Yoshida, and F. Kanehiro, "As-conformal-as-possible surface registration," in *Proceedings of the Symposium on Geometry Processing*, ser. SGP '14, Goslar, DEU: Eurographics Association, 2014, 257–267.

[34] C. Schüller, L. Kavan, D. Panozzo, and O. Sorkine-Hornung, "Locally injective mappings," *Computer Graphics Forum (proceedings of Symposium on Geometry Processing)*, vol. 32, no. 5, 2013.

[35] O. Sorkine and M. Alexa, "As-rigid-as-possible surface modeling," in *Symposium on Geometry processing*, vol. 4, 2007, pp. 109–116.

[36] O. Sorkine-Hornung and M. Rabinovich, "Least-squares rigid motion using svd," *Computing*, vol. 1, no. 1, pp. 1–5, 2017.

[37] T. Papadopoulo and M. I. Lourakis, "Estimating the jacobian of the singular value decomposition: Theory and applications," in *European Conference on Computer Vision*, Springer, 2000, pp. 554–570.

[38] R. A. Waltz and J. Nocedal, "Knitro 2.0 user's manual," *Ziena Optimization, Inc.[en ligne] disponible sur http://www. ziena. com (September, 2010)*, vol. 7, pp. 33–34, 2004.

[39] M. Toussaint, "Logic-geometric programming: An optimization-based approach to combined task and motion planning," in *Twenty-Fourth International Joint Conference on Artificial Intelligence*, 2015.