Continuous-Time Collision Avoidance for Trajectory Optimization in Dynamic Environments

Wolfgang Merkt, Vladimir Ivan, and Sethu Vijayakumar

***Abstract*— Common formulations to consider collision avoidance in trajectory optimization often use either preprocessed environments or only check and penalize collisions at discrete time steps. However, when only checking at discrete states, this requires either large margins that prevent manipulation close to obstacles or dense time discretization increasing the dimensionality of the optimization problem in complex environments. Nonetheless, collisions may still occur in the interpolation/transition between two valid states or in environments with thin obstacles. In this work, we introduce a computationally inexpensive continuous-time collision avoidance term in presence of static and moving obstacles. Our penalty is based on conservative advancement and harmonic potential fields and can be used as either a cost or constraint in off-the-shelf nonlinear programming solvers. Due to the use of conservative advancement (collision checks) rather than distance computations, our method outperforms discrete collision avoidance based on signed distance constraints resulting in smooth motions with continuous-time safety while planning in discrete time. We evaluate our proposed continuous collision avoidance on scenarios including manipulation of moving targets, locomanipulation on mobile robots, manipulation trajectories for humanoids, and quadrotor path planning and compare penalty terms based on harmonic potential fields with ones derived from contact normals.**

I. Introduction

Planning collision-free motion in complex environments is an active area of research in robotics. Robots are expected to avoid undesired contact with themselves (self-collisions) and the environment (collision with obstacles) while performing desired interactions, e.g., grasping or making contact with support surfaces. While these tasks are well studied for sampling-based motion planning methods using a combination of conditional logic and binary validity checks for both discrete (states/vertices) and continuous (edges/transitions between two valid states) cases, incorporating them into optimization-based formulations is nontrivial. In particular, incorporating collision avoidance in trajectory optimization introduces nonlinearity (and often, if not pre-processed, non-convexity) or relies on reduced models (e.g., only considers the end-effector), preprocessed robot models or environments. Furthermore, these approaches often utilize distance computations between polyhedra or primitive shapes which can be expensive to compute and may introduce discontinuities in the proxy metric or its Jacobian[1] in most cases only considering collision avoidance at discrete knot points.

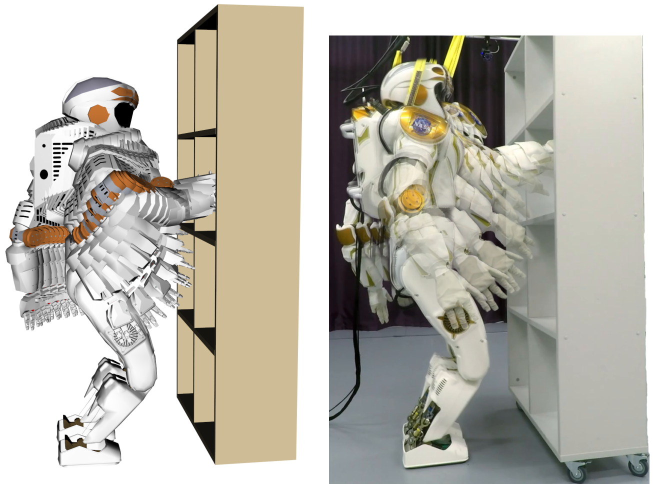


Fig. 1: Continuous time collision avoidance during whole-body manipulation execution on the 38-DoF NASA Valkyrie humanoid platform: Discrete metrics will consider a coarsely discretized trajectory to be valid while a transition between two states can be in collision with the environment. Continuous collision avoidance for discrete trajectories successfully avoids the obstacle and guarantees continuous-time collision safety.

A. Related work

Collision avoidance in motion optimization primarily focuses on defining penalty terms or nonlinear inequality constraints based on signed distance information between pairs of primitive shapes or polyhedral objects $\mathcal{A}$ and $\mathcal{B}$, with their transforms obtained from forward kinematics at the current joint configuration $\mathbf{q}$. In order to define gradients, local approximations based on motion along the normal spanned by the witness points on the respective objects are defined. However, in practice this approach suffers from two limitations: (a) numerical instability related to the implementations of the GJK [2] and EPA [3] algorithms in commonly available software libraries, and (b) fast motion of these witness points when surfaces are parallel and the resulting discontinuity in the gradients for non-strictly-convex shapes [4]. Additionally, the computation of the signed distance depends on the number and complexity (number of

This research is supported by the Engineering and Physical Sciences Research Council (EPSRC, grant reference EP/L016834/1), the EPSRC UK RAI Hub in Future AI and Robotics for Space (FAIR-SPACE, project ID: EP/R026092/1), and EU H2020 project Memory of Motion (MEMMO, project ID: 780684). The work has been performed at the University of Edinburgh under the Centre for Doctoral Training in Robotics and Autonomous Systems program.

All authors are with the Institute for Perception, Action, and Behaviour, School of Informatics, The University of Edinburgh (Informatics Forum, 10 Crichton Street, Edinburgh, EH8 9AB, United Kingdom). Email: wolfgang.merkt@ed.ac.uk.

[1] For illustration, cf. Figure 4 in [1].

vertices) of the collision shapes and, generally, is expensive to compute.

Many approaches to ensuring smooth gradients have thus focused on offline and/or online preprocessing the collision model and environment. Stasse et al. [5] used strict-convexity bounding volumes based on patches of spheres and tori (Sphere-Torus-Patches Bounding Volume, STP-BV) [4] to guarantee the continuity of the proximity distance gradient for real-time collision avoidance on a full-size humanoid robot. CHOMP [6] and STOMP [7] replace the robot collision model with overlapping sphere approximations and use Euclidean Distance Transforms for the environment. Similarly, for self-collisions only, Sugiura et al. [8] proposed self-collision-avoidance using an artificial force changing the desired posture target in the null-space of the main tasks using a sphere-swept lines bounding volume. In general, simplifications based on replacing polyhedra with enclosing primitive shapes, e.g., the use of cylinders and spheres for a redundant manipulator [9] or sphere swept volumes on a humanoid robot [8], make use of the availability of inexpensive to compute analytic, closed-form solutions—however, fitting approximations is often a time-consuming process and requires attention to convexity or otherwise local minima may arise [4]. RieMo [10], on the other hand, builds on Riemannian geometry allowing motion near thin or long obstacles to be planned which are common failure cases for pair-wise signed distance constraints. Alternatively, approaches to limit the computational cost associated with collision queries have been proposed such as adaptive collision checking densities with more checks closer to obstacles [11].

However, these approaches only enforce the collision constraint as an inequality bound on the signed distance at discrete collocation points to a safety margin ϵ. While there is a direct relationship between the selected safety margin, the maximum joint velocity, and the time discretization of the trajectory, no guarantee for a continuous-time collision-free trajectory can be given.

In sampling-based planning, continuous collision checks that check at distinct interpolations are used. In practice, some number of subsamples between two configurations $\mathbf{q}_t$ and $\mathbf{q}_{t+1}$ are evaluated – if one is in collision, the transition (edge) is considered invalid. Dynamic Roadmaps explicitly encode the swept volume of an edge and update the graph based on environment collision information. Available memory largely limited the number of edges that can be stored. Recently, a novel hierarchical variant has been proposed which explicitly resolves the required ϵ for discrete distance checking and stores a configuration-to-workspace-occupation mapping occupation information for configurations such that the edges are fully enclosed by two adjacent vertices [12].

However, for trajectory optimization, continuous-time collision avoidance remains an open challenge and has received less attention compared with discrete-time constraints. Instead, denser discretization of the time horizon is frequently applied, however, this results in larger optimization problems. Alternately, greater safety margins are proposed restricting close interactions such as reaching into deep boxes or through narrow gaps.

A notable exception—and similar to our approach—which considers continuous-time safety by constraining the minimum signed distance between swept volumes of convex shapes and non-convex shapes using sequential convex optimization is TrajOpt [1]. Here, convex-convex collision checking between approximately convex, static objects and swept-out (cast) volumes of convex collision bodies is used to formulate a maximum penetration penalty term as a convex hinge loss. Instead of using the obstacle shapes directly, Deits and Tedrake [13] apply greedy convex segmentation and mixed-integer optimization for dynamic quadrotor path planning ensuring that the planned motion lies within the convex subregions of obstacle-free space.

Recently, Hauser [14] proposed a trajectory optimization formulation based on semi-infinite programming which handles both non-convex obstacles and continuous-time collision avoidance by interactively adding constraints during the optimization. However, these methods depend on more complex optimization methods compared with off-the-shelf nonlinear programming or require the computation of signed distances, which are an order of magnitude more expensive than a simple, binary collision check.

Continuous-time collision detection, however, is a well-studied problem in computer graphics to address the *tunneling problem* where collisions occur between two simulation timesteps (e.g., a fast traveling bullet may otherwise pass through a wall). As a result, methods such as conservative advancement (CA) [15], [16] and continuous-collision detection (CCD) are based on bounding volume hierarchies for polyhedral models and are an efficient way to compute the time of contact/impact between two objects under motion while guaranteeing not to miss any collisions.

B. Contribution

In this work, we introduce a penalty term and its Jacobian for continuous collision avoidance in environments with static and moving obstacles. Our method is based on conservative advancement collision checks and harmonic potential fields and can be directly used as either a cost or constraint in nonlinear programming-based motion synthesis using off-the-shelf solvers without changes to the optimization scheme (e.g., outer loop changes to cost scheduling or interactive addition/removal of constraints between iterations).

We show the application of the metric in various high-dimensional scenarios initialized from feasible paths and compare with discrete-time-only penalties based on pairwise distances as well as constraints based on signed distances from continuous collision casts.

II. Problem Formulation

We consider path planning and motion planning as a constrained minimization of a canonical optimality criterion (e.g., minimum time, minimum torque, or a higher order smoothness term) subject to bound, equality, and inequality constraints:

$$\begin{array}{rrcl} \arg\min\limits_{x} & \ell(x) & & \\ \text{s.t.} & h_i(x) & = & 0 \\ & g_j(x) & \leq & 0 \\ & x_{lb} & \leq x \leq & x_{ub} \\ & \dot{x}_{lb} & \leq \dot{x} \leq & \dot{x}_{ub} \end{array} \qquad (1)$$

where a trajectory of length T with uniform time discretization Δt is represented as the sequence of state vectors $X = (x_1, x_2, ..., x_T)$.

A. Discrete-time collision avoidance

Traditionally, collision avoidance is integrated here as nonlinear inequality constraints of the closest signed distance to the collision shapes of the actuated links at the discrete time points t, subject to a safety margin ϵ: $h_i(x) = sd(x_i) - \epsilon \leq 0$.

Similarly, a smooth cost and gradient for unconstrained optimization can be formulated [17].

However, determining a suitable safety margin ϵ can prove tricky: while intuitive for the discrete-time case, in order for it to capture potential collisions in the continuous-time transition, an appropriately large ϵ has to be chosen such that the two collision shapes enlarged by their safety margins overlap (cf. Fig. 2). In particular, without an explicit joint velocity limit, solutions may become discontinuous—they will be valid at the discrete waypoints, but standard transition between two configurations (e.g., straight line interpolation) will be in collision (cf. Fig. 6).

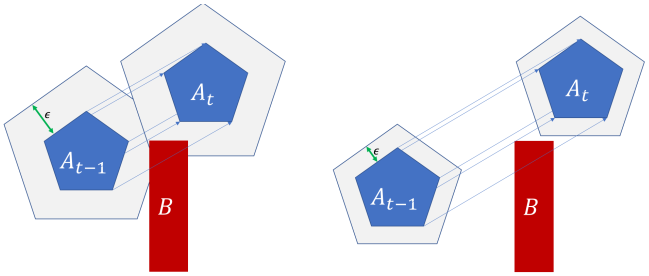


Fig. 2: Left: A wide safety margin ϵ along with a dense time discretization and limited maximum step capture the continuous-time transition between $t-1$ and t as the enlarged collision bodies overlap. Right: In contrast, a small ϵ or large permissible state transition stemming from the time discretization/joint velocity limit result in the interpolation from $t-1$ to t being in collision with $\mathcal{B}$ while the signed distance constraints at $\mathcal{A}_{t-1}$ and $\mathcal{A}_t$ are both satisfied.

In order to develop a metric for scaling ϵ for each actuated link i attached to a kinematic chain, an upper bound on the change between two states should be set (i.e., a joint velocity limit). Using this, an approximate scaling can be developed for each link $i \in 1..N$:

$$\epsilon_i = \epsilon_{i-1} + \frac{l_i \cos\left(\dot{x}_{max} \Delta t\right)}{2}, \text{ with } \epsilon_0 = 0 \qquad (2)$$

where l_i is the length of the ith link. As is evident, the ϵ has to be greater for links further removed from the root of the kinematic tree, which makes fine-grained manipulation or interaction in confined spaces intractable.

In contrast, in order to specify a maximum workspace distance, [12] developed a relationship between a workspace resolution and a corresponding required configuration space discretization to guarantee a collision-free edge (i.e., by ensuring the workspace occupation of two subsequent states/vertices overlap). It further provided a resolution completeness proof for general deterministically-sampled roadmaps. This approach can alternately be used to determine individual maximum joint velocities given a desired workspace resolution/collision avoidance ϵ. This, however, could easily result in the requirement for a fine discretization of the time horizon and in its current form does not extend to moving multiple joints as continuous variables at the same time as required in motion optimization.

III. Continuous Collision Avoidance

In practice, continuous collision checking for sampling-based algorithms can easily be implemented by subsampling states as interpolations between two waypoints or applying conservative advancement collision checks. The challenge arises when defining a differentiable cost or constraint metric for use in optimization-based algorithms.

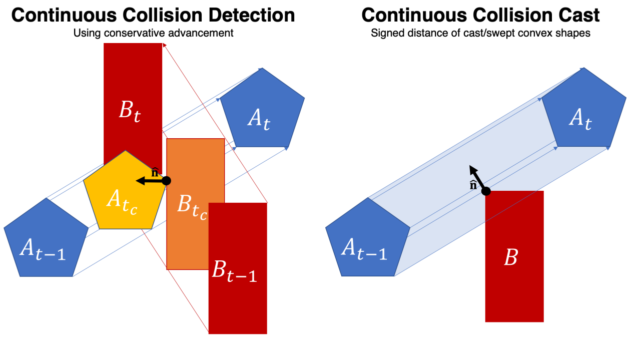

Continuous Collision Detection
Using conservative advancement
B_t
A_t
$\hat{n}$
A_{t_c}
B_{t_c}
A_{t-1}
B_{t-1}
Continuous Collision Cast
Signed distance of cast/swept convex shapes
A_t
$\hat{n}$
A_{t-1}
B

Fig. 3: The figure above highlights the two different continuous collision detection modes: conservative advancement (left) and convex collision shape casting (right, e.g., as in [1]). Note, the meaning and direction of the collision normal differs significantly.

In order to define a penalty term, we introduce the concept of a *continuous collision proxy* (CCP) which contains a binary flag c on whether the two objects are in collision along the interpolation, the time of contact t_c if they are in collision, their transforms $T_{\mathcal{A},t=c}, T_{\mathcal{B},t=c}$ at time of contact, the penetration depth between the objects d_p, the contact position P_c and the contact normal $\mathbf{n}_c$:

$$CCP = (c, t_c, T_{\mathcal{A},t=c}, T_{\mathcal{B},t=c}, d_p, P_c, \mathbf{n}_c) \qquad (3)$$

This information is calculated for every pair of robot to environment links by performing continuous collision detection (CCD) for two objects $\mathcal{A}$ and $\mathcal{B}$ given their initial and final world transforms and motion interpolation:

$$CCP = CCD(\mathcal{A}, \mathcal{B}, T_{\mathcal{A},t=0}, T_{\mathcal{B},t=0}, T_{\mathcal{A},t=T}, T_{\mathcal{B},t=T}) \quad (4)$$

Here, we use Conservative Advancement [15] to perform CCD. We note that this information can also be computed by considering the swept-out volume of the edge as a convex body computed from its support mappings as used in [1] and represents a geometrically more realistic interpretation of the direction and depth between the edge and a static obstacle (and additionally includes the signed distance for objects at distance). In this work, we rely on continuous collision detection between moving shapes and as thus, direct use of the computed normal information is not expected to be well-behaved. The difference is visualized in Fig. 3.

Using the closest continuous collision proxy for each actuated link, a penalty can be formulated if the link is in collision:

$$\phi(\mathbf{q}_{t-1}, \mathbf{q}_t) = \begin{cases} d_p & \text{if } d_p \geq 0 \\ 0 & \text{if } d < 0 \end{cases} \tag{5}$$

with a corresponding, approximate derivative:

$$\frac{\delta \phi}{\delta \mathbf{q_t}}(\mathbf{q}_{t-1}, \mathbf{q}_t) = \begin{cases} -\mathbf{n}_c \cdot J(P_c) & \text{if } d_p \geq 0 \\ \mathbf{0} & \text{if } d < 0 \end{cases} \tag{6}$$

where $J(P_c)$ is the geometric Jacobian of the contact point.

We note that this derivative is approximate and discontinuous at the boundary of collision (while the scalar metric is not), however, due to the use of continuous collision checks rather than distances to swept volumes (the former of which is computationally much faster) we obtain a very fast metric that in practice works well when initialized from a collision-free guess in complex environments, e.g., obtained from sampling-based planning or a memory of motion. We further formulate the derivative with respect to a single waypoint assuming the previous one to be fixed.

A. Harmonic potential field-based continuous collision avoidance in dynamic environments

One challenge with conservative advancement is that the computed contact information is the first contact between the two moving collision bodies (i.e., the two bodies are touching). The returned contact normal is often not representative of the normal vector of the maximum penetration of the cast/swept volume and in practice unstable, and extremely dependent on the implementation and its numerical stability.

In order to provide a more robust gradient for the contact penalty term, we utilize a harmonic potential field in place of the penetration depth in (5). Such fields exhibit local minima (saddle points) only due to the topology of the obstacle. This makes them ideal for gradient descent and optimization. Indeed, harmonic potential fields have been widely used for avoiding static obstacles [18] and for navigation problems [19]. A harmonic potential field around a 3D shape can be derived from equations for electric potential around a uniformly charged object [20]. The electric potential arising from a point charge q, at a distance r from the charge is defined as $V = \frac{1}{4\pi\epsilon_0}\frac{q}{r}$. However, we are interested in computing the potential over a triangulated surface of the obstacle's surface.[2] This will allow us to interact with generic shapes given the potentials of the triangles they are made up of. As this is an approximation of the true harmonic potential field of the original shape, local minima may arise. [21] analysed the resulting approximation error and [20] introduced a linear programming based method to readjust the local surface charge distribution.

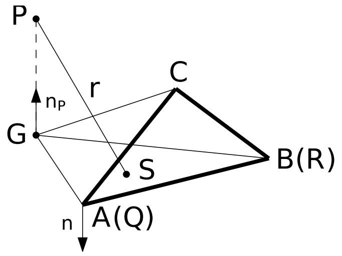


Fig. 4: Electrostatic potential due to a charged triangle. Figure reproduced from [20].

The integral of the potential V over the surface of an uniformly charged triangle ABC (cf. Fig. 4 for definitions of variables) has been derived in [21] as:

$$V = \frac{1}{4\pi\epsilon_0}\frac{q}{\triangle ABC} I \tag{7}$$

$$I = \pm I_{\triangle ABG} \pm I_{\triangle BCG} \pm I_{\triangle CAG} \tag{8}$$

$$\begin{aligned} I_{\triangle QRG} = & \frac{\sigma}{|\overrightarrow{RQ}|} \log \frac{\overrightarrow{RQ} \cdot \overrightarrow{RG} + |\overrightarrow{RQ}||\overrightarrow{RP}|}{-\overrightarrow{QR} \cdot \overrightarrow{QG} + |\overrightarrow{QR}||\overrightarrow{QP}|} \\ & + h \arctan \frac{\sigma \overrightarrow{RQ} \cdot \overrightarrow{RG}(h - |\overrightarrow{RP}|)}{\sigma^2 |\overrightarrow{RP}| + h(\overrightarrow{RQ} \cdot \overrightarrow{RG})^2} \\ & + h \arctan \frac{\sigma \overrightarrow{QR} \cdot \overrightarrow{QG}(h - |\overrightarrow{QP}|)}{\sigma^2 |\overrightarrow{QP}| + h(\overrightarrow{QR} \cdot \overrightarrow{QG})^2} \end{aligned} \tag{9}$$

$$\sigma = (\overrightarrow{QG} \times \overrightarrow{QR}) \cdot \frac{\overrightarrow{AC} \times \overrightarrow{BC}}{||\overrightarrow{AC} \times \overrightarrow{BC}||}, h = ||\overrightarrow{QP}||, \tag{10}$$

where ABC are the vertices of the triangle, P is the query point and G its projection onto the plane of the triangle ABC, ϵ_0 is the permittivity of vacuum, and q is the charge of the triangle. Since the physical meaning of equation (7) is irrelevant in the context of trajectory optimization, the charge q can be chosen to arbitrarily scale the potential to improve the numerical stability of the solver or as a relative weight against other cost terms. To calculate I, we substitute the permutations of the triangle vertices into equation (9). We then sum the potentials of all triangles to calculate the potential over the whole surface of the collision mesh. We obtain the derivative of the potential, also called the *electric field*, using the chain rule (see [20] for full formula).

For computational efficiency, we create and cache a harmonic potential field for each collision body as a set of triangles and transform all queries into the local frame of the collision body. We query the potential of the collision contact point on the surface of the active/moving robot shape within the harmonic potential field of the obstacle. For the moving obstacles, we transform the collision mesh of the

[2] To further improve the performance of this method, closed-form solutions for the potential and field of primitive shapes (boxes, cylinders, spheres, etc.) can be used.

moving obstacle half way between its start and end position. This is an approximation that allows us to avoid computing the mesh of the swept volume of the moving obstacle. We then use the geometric Jacobian of the relative point in the link of the robot to compute a derivative with respect to the control variables. The resulting term is then used as a constraint or a cost term in an optimization problem (1).

IV. Evaluation

We have implemented the proposed collision avoidance method in the planning prototype and benchmarking library EXOTica [22] using FCL [23] for continuous collision detection (Equation (4)) and the Bullet physics engine for continuous collision casts.[3] This comes with the advantage that all benchmarks use the same forward kinematics, collision checking, objective and Jacobian computation and thus performance differences are attributable to internal optimization solver computations and the number of problem updates they require to converge to a solution. All evaluations were carried out on a computer with an Intel Core i7-6700K CPU with $4\,\mathrm{GHz}$ base frequency and $32\,\mathrm{GB}$ $2133\,\mathrm{MHz}$ memory in a single thread. In this paper, we report our results using the commercial sparse, constrained nonlinear programming solvers SNOPT [24] and KNITRO [25] with default parameters. Supplementary material and videos is available at `http://www.wolfgangmerkt.com/continuous-collision-avoidance`. The results of our evaluation are in Table I and they test our approach on the following scenarios:

A. Quadrotor path planning

In this example, a sparsely discretized collision-free path for the base trajectory (6-DoF) of a quadrotor in presence of obstacles is optimized subject to maximum task-space velocities from a zero-motion initialization. We visualize a single, wide and thin obstacle dividing the space in Fig. 5a with benchmark results given in the first and second row of Table I. It is key to note that the discrete collision penalty converges quicker than the problems with continuous collision avoidance penalties, however, the trajectory cuts corners and passes through the thin obstacle. The harmonic potential field-based method converges quickly to a continuous-time collision-free path while the contact normal-based penalties result in divergence or termination from numerical instability.

B. Motion planning near thin obstacles

As thin obstacles pose particular problems for collision avoidance constraints at discrete waypoints, we demonstrate a sparsely discretized motion plan (10-20 waypoints) on a 7-DoF Kuka LWR3+ in Fig. 6 with benchmark results given in the rows 3 and 4 of Table I. Fig. 6a highlights the optimal motion from a start state on the left side of the obstacle to an end-effector constraint on the right side without considering collision avoidance. Fig. 6b depicts the optimal solution using a distance-based discrete collision constraint which is valid at each waypoint, but discontinuous and in collision during its transition as can be seen from the motion trail. Fig. 6c shows the optimal trajectory computed using the proposed continuous collision avoidance constraint—note, the computation times are significantly lower than for the discrete-time penalty.

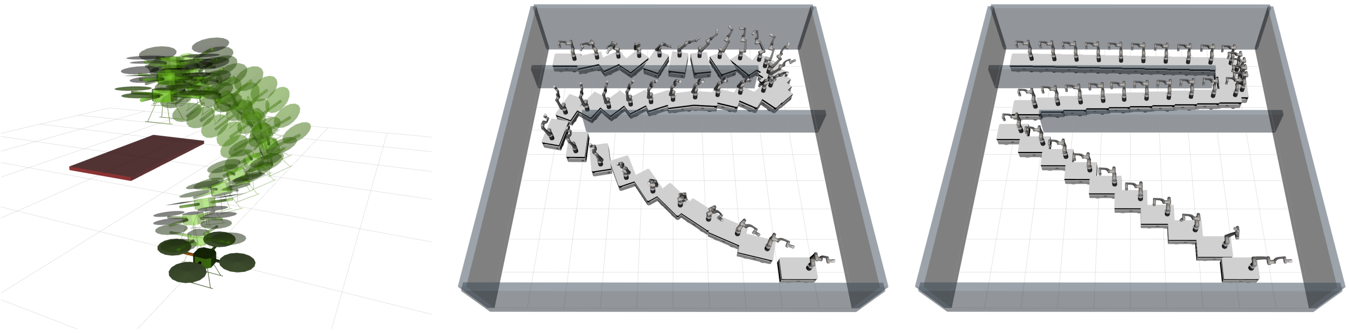
C. Moving environment obstacles

Collision avoidance in dynamic environments is a challenging problem in motion planning, and in particular for trajectory optimization. Assuming the trajectory of all objects in the scene is known a priori, time-configuration space sampling or optimization methods can be employed. As directly solving the full problem with a single optimization problem was considered infeasible with local optimization methods, [26] split the planning problem into a sequence of sampling- and optimization-based subproblems: reaching (collision-free, bidirectional sampling-based motion planning), grasping (trajectory optimization), and placing (collision-free, bidirectional sampling-based motion planning). Here, we demonstrate a scenario similar to those presented in [26] as a single optimization problem using harmonic potential field-based continuous collision avoidance and solve it from zero-motion initialization. The 7-DoF Franka Emika Panda robot has to reach into a box which moves on a trajectory at a velocity of $0.2\,\mathrm{m\,s^{-1}}$, follow a target object while the fingers are closing, and place it to the side. The proposed method solves the problem with a time horizon of $10\,\mathrm{s}$ at a $0.1\,\mathrm{s}$ discretization in $7.55\,\mathrm{s}$ (cf. Table I, rows 5 and 6, and Fig. 8). Using a higher relative function tolerance, a first feasible solution can be found in $1.77\,\mathrm{s}$. The penalties which are based on contact normals and using the information returned from continuous collision casts fail for this application as they were not designed to handle moving obstacles.

D. Locomanipulation

Locomanipulation considers navigation/locomotion and manipulation as a unified problem. To make the problem tractable, it is often split into pipeline-based approaches, e.g., where the base placement, navigation, and manipulator motion planning are treated as separate problems resulting in sub-optimal motions. Here, we consider a locomanipulation planning for a 9-DoF mobile manipulation consisting out of an omnidirectional base (3-DoF) with a 6-DoF manipulator. In a bug trap-like scenario, sampling-based planners for the entire motion deliver sub-optimal results (cf. Fig. 5b). We use the collision-free path as an initial solution (RRT-Connect with dense interpolation between two states to validate edges, $0.35\,\mathrm{s}$) and optimize using the proposed continuous collision avoidance metrics for a smooth whole-body motion ($0.40\,\mathrm{s}$, cf. Fig. 5c and Table I, rows 7 and 8).

[3] While we choose to mainly rely on FCL for this work, the flexible framework of EXOTica allows for the easy switching of collision solver plug-ins opening up the possibility for comparative evaluation of the proposed metrics using other libraries, e.g., NVIDIA PhysX, in future work. In this work, we use the ROS-Industrial middleware Tesseract as an interface to the Bullet physics engine as it implements the continuous collision casting as in [1].

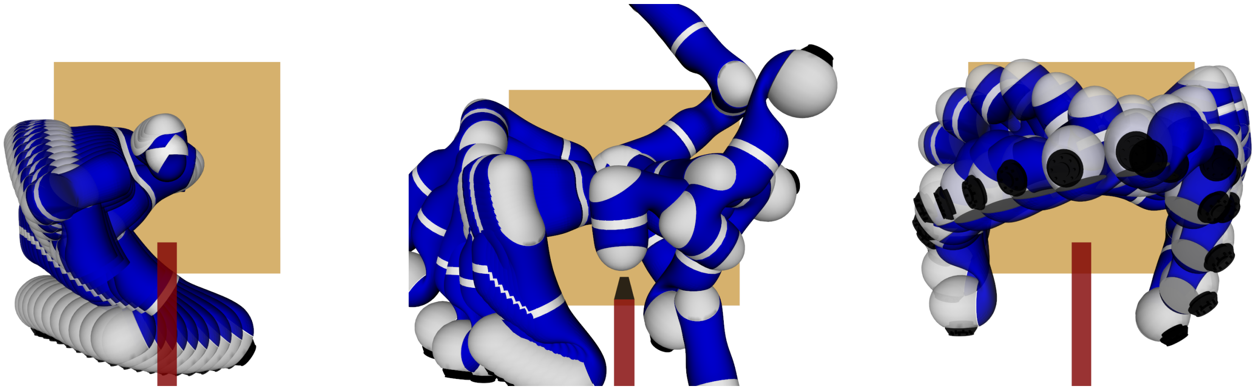


(a) Collision-free path for a quadrotor (zero-motion initialization, 8.19 ms).

(b) Collision-free initialization/feasible guess computed by RRT-Connect (0.35 s).

(c) Optimized, continuously collision-free locomanipulation trajectory (0.40 s).

Fig. 5: Experiment scenarios using a quadrotor (6-DoF) and mobile manipulation platform (9-DoF).



(a) No collision avoidance (0.05 s)

(b) Discrete-time penalty (2.37 s)

(c) Continuous-time penalty (0.68 s)

Fig. 6: Sparsely discretized motion on a 7-DoF manipulator near a thin obstacle: The discrete collision avoidance inequality constraint results in a discontinuous and infeasible path (although valid as per the discrete objective) while taking longer (due to computation of distances) compared with the proposed continuous collision tasks which results in a smooth, continuous-time collision-free trajectory.

E. Humanoid shelf manipulation

Planning smooth, collision-free manipulation motion on bipedal robots in vicinity of obstacles is challenging due to the requirement to maintain balance while avoiding obstacles and remain within actuation limits. We focus on manipulation on a shelf with multiple cabinets which introduce local minima and non-convexity. We initialize our trajectory optimization problem from a global sampling-based planner (RRT-Connect, 2.40 s) and validate the resulting trajectory on the 38-DoF NASA Valkyrie humanoid platform (cf. Figure 1), with results given in row 9 of Table I (0.90 s optimization time).

F. Step-up swing trajectory planning for legged robots

For legged robots, the swing trajectories of the end-effectors transitioning between different contact configurations, e.g., footholds, are commonly parametrized using low-dimensional polynomials, splines, or three-point interpolations. While this is efficient and often works in practice, it comes with no guarantee that the swing trajectory is collision-free, particularly when stepping up tall steps or in cluttered environments—which depending on the planned footsteps can lead to clipped steps and falls. Here, we consider the problem of planning a collision-free trajectory for the swing leg while satisfying the quasi-static balance constraint given swing and double support duration and show the obtained collision-free swing trajectory in Fig. 7.

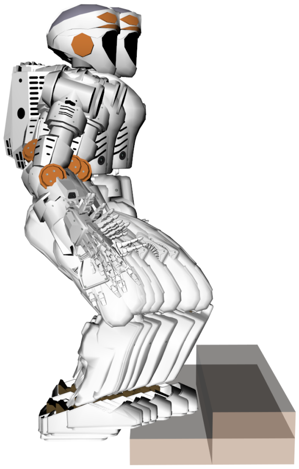


Fig. 7: Collision-free, quasi-statically balanced whole-body step-up/swing trajectory (38-DoF).

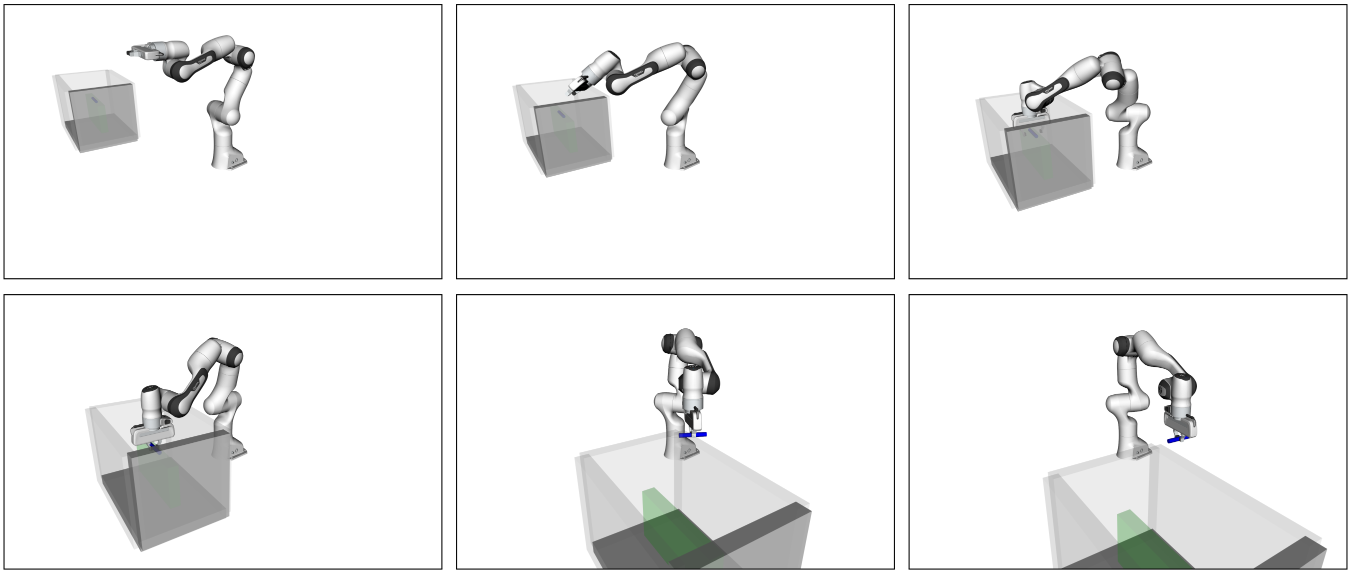


Fig. 8: Collision-free pick-and-place of a moving target with a 7-DoF Franka Emika Panda manipulator. The solution has been synthesized from a zero-motion initialization using the proposed harmonic potential field-based continuous collision avoidance constraint in $7.55\,\text{s}$.

	Experiment	DoF	Initialization	Solver	HPF (FCL)	Contact normal (FCL)	HPF (Tesseract/Bullet)	Contact normal (Tesseract/Bullet) [1]	Discrete (FCL)
1	UAV	6	Zero-motion	SNOPT	✓$\mathbf{0.008 \pm 0.001}$	✗0.053 ± 0.001	✗0.019 ± 0.001	✗0.021 ± 0.001	✗0.003 ± 0.001
2				KNITRO	✓0.075 ± 0.006	✗0.581 ± 0.007	✓$\mathbf{0.068 \pm 0.001}$	✗0.162 ± 0.002	✗0.008 ± 0.001
3	LWR, thin obstacle	7	Zero-motion	SNOPT	✓0.170 ± 0.001	✗0.769 ± 0.011	✗0.786 ± 0.014	✓$\mathbf{0.104 \pm 0.002}$	✗3.124 ± 0.060
4				KNITRO	✓1.803 ± 0.025	✓3.158 ± 0.024	✓2.293 ± 0.012	✓$\mathbf{1.457 \pm 0.029}$	✗5.679 ± 0.049
5	Panda, moving obstacle	7	Zero-motion	SNOPT	✓$\mathbf{7.557 \pm 0.089}$	✓14.178 ± 0.119	✗475.403 ± 4.172	✗105.864 ± 0.126	✗11.528 ± 0.016
6				KNITRO	✗126.665 ± 1.017	✗252.125 ± 3.806	✗400.606 ± 4.264	✗211.840 ± 0.235	✗252.471 ± 1.890
7	Locomanipulation with mobile manipulator	9	Collision-free (RRT-Connect)	SNOPT	✓$\mathbf{0.396 \pm 0.019}$	✓0.400 ± 0.029	✓0.431 ± 0.025	✓0.429 ± 0.008	✓7.604 ± 0.049
8				KNITRO	✓$\mathbf{1.855 \pm 0.089}$	✓7.753 ± 0.233	✓11.625 ± 0.579	✗16.245 ± 0.592	✗839.747 ± 23.034
9	Valkyrie: shelf manipulation	10	Collision-free (RRT-Connect)	SNOPT	✓$\mathbf{0.904 \pm 0.228}$	✗1.308 ± 0.858	✗1.835 ± 1.721	✗3.877 ± 0.693	✗1.624 ± 0.145

TABLE I: Computation times for a selection of motion planning problems indicating their degrees of freedom (DoF), initialization strategy (zero-motion or collision-free from a global, sampling-based algorithm), and non-linear programming solver. We compare two continuous-time metrics (contact normal and harmonic potential field, HPF) for two different collision solvers (FCL and Bullet), as well as a discrete distance-based nonlinear inequality constraint (using FCL). All computation times are given in seconds and averaged over 10 runs. The best-performing algorithm with a valid continuous-time collision-free solution is highlighted in bold. We indicate whether a trajectory is valid satisfying all constraints and is continuous-time collision-free (as validated by using very dense interpolation) using ✓ and ✗ otherwise.

V. Discussion

This paper considered the formulation of fast continuous collision avoidance penalty terms in the presence of moving obstacles which can be incorporated directly into formulations solved with off-the-shelf nonlinear programming solvers. To the best of our knowledge, this is the first work to address continuous-collision avoidance for trajectory optimization in dynamic environments.

In particular, we combine conservative advancement with harmonic potential fields around collision shapes to obtain a smooth, continuously differentiable proxy metric for continuous collision avoidance. We highlighted the versatility of the proposed method on a variety of high-dimensional trajectory optimization problems from feasible and zero-motion initialization and validated our motion plans using hardware experiments on the NASA Valkyrie humanoid.

As a limitation, the conservative advancement implementation we deployed in this work relies on linear or screw interpolation between initial and final transforms as a motion model. Due to this, the method cannot handle arbitrarily large transitions. While other motion models could be chosen, they also form an approximation to the actual nonlinear motion of the collision body rigidly attached to a kinematic chain. For the case of a continuous collision cast, [1] presents an analysis of the difference between a linear sweep/cast and the exact nonlinear shape and states that, in practice, the difference can often be neglected as it is contained within the safety threshold ϵ.

As we used local nonlinear optimization, our method still frequently requires collision-free (but not necessarily feasible) initialization in order to converge. It has shown to perform well in combination with sampling-based planning, e.g., as part of a hybrid planner where a collision-free guess can be provided quickly from sampling-based or roadmap-based methods and then refined to fit optimality criteria using trajectory optimization. Alternately, it can be used within a memory of motion framework for warm-start initialization as in [17]. Its faster computation time compared with constraints based on distance computations allows online deployment both for direct motion optimization as well as path simplification and motion smoothing as commonly employed in sampling-based motion planning—with the added advantage that in addition to smoothing, motion constraints can be satisfied.

An intriguing approach is to consider convex formulations of the problem. Schulman et al. [1] use a convex hinge penalty on the signed distance obtained from a continuous collision cast in combination with a custom sequential convex optimization solver and are able to synthesize motion from an infeasible initialization. Deits and Tedrake [13] apply greedy subdivision of the space into convex subregions and use mixed-integer optimization to plan globally optimal continuously-collision-free trajectories for the dynamic model of a quadrotor in complex environments. However, the scalability of the approach to articulated robots has not yet been investigated. Considering the properties of convex optimization with regards to convergence and optimality, formulating a convex continuous-collision avoidance constraint in presence of moving obstacles following these lines of work presents an interesting area of study.

All of the discussed approaches required computing signed distances between polyhedra or pre-processed proxies such as Euclidean Distance Transforms. Campana et al. [27] avoid the need to compute distance information altogether by iteratively adding (or modifying existing) linear constraints whenever a collision is detected. While this requires a custom optimization scheme due to the changing optimization problem (i.e., the number of constraints differs between iterations), it would be interesting to explore whether this scheme could be extended to continuous-time collision avoidance due to its low computational cost and numerical stability.

While the case switching in (5) due to the binary collision indicator c is discontinuous at the boundary in contrast to signed distance-based approaches (the Jacobian is set to zero when not in collision, which is strictly speaking incorrect), it worked well in practice particularly from collision-free initialization. However, further investigating the use of approximate harmonic potential fields for a fully continuous metric is an interesting avenue for future work.

We do not currently explicitly handle self-collisions as part of the continuous collision avoidance term and instead incorporated them through a combination of joint limits and discrete collision costs, which in future work could be learnt as a robot model-specific term.

Finally, we considered path planning and high-dimensional kinematic trajectory optimization in this paper with ongoing work focusing on extending the formulation to dynamic optimal control problems.

References

[1] J. Schulman, Y. Duan, *et al.*, "Motion planning with sequential convex optimization and convex collision checking," *The Int. J. Rob. Res. (IJRR)*, vol. 33, no. 9, pp. 1251–1270, 2014.

[2] E. G. Gilbert, D. W. Johnson, and S. S. Keerthi, "A fast procedure for computing the distance between complex objects in three-dimensional space," *IEEE J. Robot. Autom.*, vol. 4, no. 2, pp. 193–203, Apr 1988.

[3] G. Van Den Bergen, "Proximity queries and penetration depth computation on 3d game objects," in *Game Developers Conference*, Mar 2001, pp. 821–837.

[4] A. Escande, S. Miossec, *et al.*, "A strictly convex hull for computing proximity distances with continuous gradients," *IEEE Trans. Robot.*, vol. 30, no. 3, pp. 666–678, Jun 2014.

[5] O. Stasse, A. Escande, *et al.*, "Real-time (self)-collision avoidance task on a HRP-2 humanoid robot," in *IEEE ICRA*, May 2008, pp. 3200–3205.

[6] N. Ratliff, M. Zucker, *et al.*, "CHOMP: Gradient optimization techniques for efficient motion planning," in *IEEE ICRA*, May 2009, pp. 489–494.

[7] M. Kalakrishnan, S. Chitta, *et al.*, "STOMP: Stochastic trajectory optimization for motion planning," in *IEEE ICRA*, May 2011, pp. 4569–4574.

[8] H. Sugiura, M. Gienger, *et al.*, "Real-time self collision avoidance for humanoids by means of nullspace criteria and task intervals," in *IEEE-RAS Humanoids*, Dec 2006, pp. 575–580.

[9] R. V. Patel, F. Shadpey, *et al.*, "A collision-avoidance scheme for redundant manipulators: Theory and experiments," *Journal of Robotic Systems*, vol. 22, no. 12, pp. 737–757, 2005.

[10] N. Ratliff, M. Toussaint, and S. Schaal, "Understanding the geometry of workspace obstacles in motion optimization," in *IEEE ICRA*, May 2015, pp. 4202–4209.

[11] D. Pavlichenko and S. Behnke, "Efficient stochastic multicriteria arm trajectory optimization," in *IEEE/RSJ IROS*, Sep 2017, pp. 4018–4025.

[12] Y. Yang, W. Merkt, *et al.*, "HDRM: A Resolution Complete Dynamic Roadmap for Real-Time Motion Planning in Complex Scenes," *IEEE Robot. Autom. Lett.*, vol. 3, no. 1, pp. 551–558, Jan 2018.

[13] R. Deits and R. Tedrake, "Efficient mixed-integer planning for UAVs in cluttered environments," in *IEEE ICRA*, May 2015, pp. 42–49.

[14] K. Hauser, "Semi-Infinite Programming for Trajectory Optimization with Nonconvex Obstacles," in *WAFR*, 2018.

[15] B. V. Mirtich, "Impulse-based dynamic simulation of rigid body systems," Ph.D. dissertation, 1996.

[16] S. Redon, M. C. Lin, *et al.*, "Fast continuous collision detection for articulated models," *Journal of Computing and Information Science in Engineering*, vol. 5, no. 2, pp. 126–137, 2005.

[17] W. Merkt, V. Ivan, and S. Vijayakumar, "Leveraging precomputation with problem encoding for warm-starting trajectory optimization in complex environments," in *IEEE/RSJ IROS*, Oct 2018, pp. 5877–5884.

[18] J.-O. Kim and P. Khosla, "Real-time obstacle avoidance using harmonic potential functions," in *IEEE ICRA*, Apr 1991, pp. 790–796 vol.1.

[19] R. Daily and D. M. Bevly, "Harmonic potential field path planning for high speed vehicles," in *Am. Control Conf.*, Jun 2008, pp. 4609–4614.

[20] H. Wang, K. A. Sidorov, *et al.*, "Harmonic parameterization by electrostatics," *ACM Trans. Graph. (TOG)*, vol. 32, no. 5, pp. 155:1–155:12, Sep 2013.

[21] E. Goto, Y. Shi, and N. Yoshida, "Extrapolated surface charge method for capacity calculation of polygons and polyhedra," *J.Comp. Phys.*, vol. 100, no. 1, pp. 105 – 115, 1992.

[22] V. Ivan, Y. Yang, *et al.*, "EXOTica: An Extensible Optimization Toolset for Prototyping and Benchmarking Motion Planning and Control," in *Robot Operating System (ROS): The Complete Reference (Volume 3)*, A. Koubaa, Ed. Springer, 2019.

[23] J. Pan, S. Chitta, and D. Manocha, "FCL: A general purpose library for collision and proximity queries," in *IEEE ICRA*, May 2012, pp. 3859–3866.

[24] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Rev.*, 2005.

[25] R. H. Byrd, J. Nocedal, and R. A. Waltz, *Knitro: An Integrated Package for Nonlinear Optimization*. Boston, MA: Springer, 2006, pp. 35–59.

[26] Y. Yang, W. Merkt, *et al.*, "Planning in time-configuration space for efficient pick-and-place in non-static environments with temporal constraints," in *IEEE-RAS Humanoids*, Nov 2018, pp. 1–9.

[27] M. Campana, F. Lamiraux, and J.-P. Laumond, "A gradient-based path optimization method for motion planning," *Advanced Robotics*, vol. 30, no. 17-18, pp. 1126–1144, 2016.