

Real-Time Collision Handling for Signed Distance Fields via Caching and Importance Sampling

C. Giles¹  and S. Andrews² 

¹Independent researcher, USA

²École de technologie supérieure, Canada

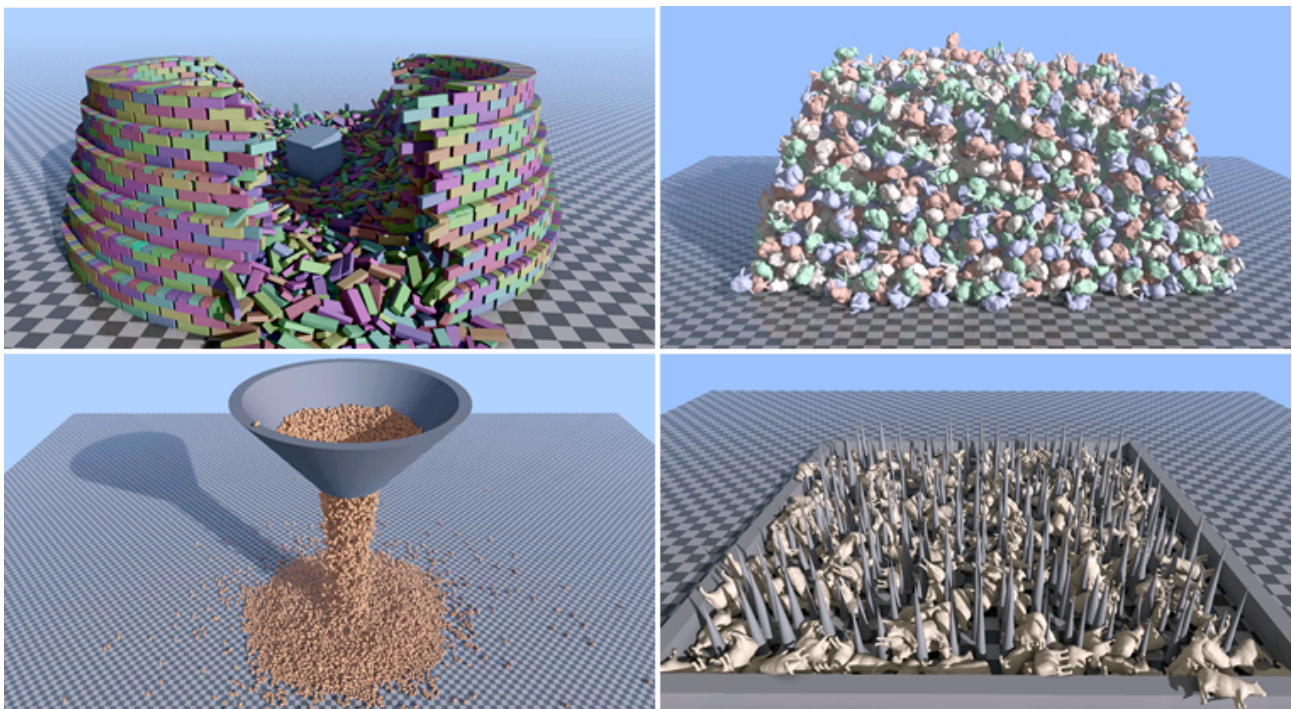


Figure 1: Large-scale environments simulated using our SDF-SDF collision framework: Colosseum (top-left), Bunnies (top-right), Funnel with cylinders (bottom-left), Cows on spikes (bottom-right). Real-time frame rates are maintained across all scenes, which feature complex interlocking geometries and thousands of concurrent contacts. Dynamic results for these scenarios are available in the supplementary video.

Abstract

This paper introduces a robust collision detection framework for SDF-encoded geometry, which has seen widespread adoption as a geometric representation in computer graphics. Our approach is designed to satisfy the competing requirements of numerical stability and computational efficiency of interactive and real-time applications. Our collision management strategy pairs temporal coherence caching with curvature-guided importance sampling. This scheme reliably forms sparse, stable contact manifolds, even for complex non-convex geometries. Quantitative evaluations against existing baselines demonstrate significant improvements in both runtime performance and simulation stability. While compatible with a range of numerical backends, our framework demonstrates how specific solver choices, such as derivative-free optimizers, yield additional gains in speed and convergence stability. Furthermore, the algorithm is highly parallelizable and well-suited for GPU architectures, scaling effectively to complex environments containing thousands of rigid bodies while maintaining real-time frame rates. Our proposed framework is validated through comparisons on complex simulation scenes and ablation studies examining the influence of various optimizers and sampling strategies.

CCS Concepts

• Computing methodologies → Physical simulation; Collision detection;

1. Introduction

Collision detection is a cornerstone of physics-based animation, ensuring that motion respects physical constraints and remains free of non-physical interpenetration. While traditional pipelines rely on explicit geometric representations, such as polygonal meshes or discrete primitives, these approaches are often burdened by increased computational overhead, high storage requirements for complex shapes, and often fail to elegantly handle topological edge cases [Erl18].

Signed Distance Fields (SDFs), have emerged as a robust alternative [MEM*20, PGMAA25, LM24, LZW*24, NE26]. By representing geometry as a continuous scalar field, SDFs provide immediate access to point-wise penetration depths and collision gradients. However, interactive applications require both collision fidelity and real-time performance, thus demanding efficient methods that can resolve collisions directly using SDF representations. While evaluating simple analytical primitives or discrete voxel grids is computationally light, complex representations, such as neural SDFs requiring costly forward passes [PFS*19] or procedural fields relying on dense trigonometric functions, make spatial query minimization essential for high-throughput simulation. Over-generating contact points also degrades downstream contact solver performance by introducing redundancies, increasing linear system dimensions, and inducing ill-conditioned system matrices [Cat06, AP97]. Furthermore, existing SDF collision algorithms are susceptible to local minima in computing penetrations [MEM*20, NE26], which can impede optimizer convergence and prevent the formation of stable multi-point contact manifolds, particularly in scenarios involving complex non-convex geometries [GBF03, TKH*05].

With these challenges in mind, we propose an optimization framework to efficiently identify concurrent contact regions between two general SDFs. Our approach generates a stable contact manifold by distributing a sparse set of contact points across the intersection volume, rather than relying on a single deepest point. To encourage spatial coverage, our objective incorporates a repulsion term between collision seed points that explicitly reduces clustering at local minima and ensures the contact manifold remains well-distributed. To address the inherent non-convexity of the SDF intersection landscape, we employ a stochastic sampling strategy that is aware of high-curvature regions, based on the Laplacian of the SDF, where collision is likely to occur. This ensures that the optimizer is robust to local minima and capable of discovering disjoint contact regions, which is essential for capturing the full geometric complexity of the interaction. By prioritizing geometrically salient features, our approach mitigates the risk of local minima and ensures a more robust convergence toward a globally consistent contact manifold.

To summarize, we contribute a discrete collision detection (DCD) framework for SDFs that integrates temporal contact caching, curvature-guided importance sampling, and spatially dispersed seed placement (via repulsion) to reliably form sparse, physically stable contact manifolds across complex geometries in real-time simulation. The capabilities of our approach are demonstrated across challenging scenarios that include complex, interlocking structures that use both analytical and voxel-based SDFs.

Our method addresses a niche in the performance-accuracy trade-off space by maintaining physical stability even when utilizing a sparse set of collision points, making it well-suited for real-time applications. An open-source implementation of our framework is made available here: <https://github.com/savant117/sdf-dcd-demo>

2. Related work

The fundamental task of collision detection involves determining the intersection state and penetration depth between geometric primitives to resolve physical interpenetration constraints. The principles of collision detection and response for physics-based animation have been extensively documented in courses [AEF22], surveys [TKH*05] and texts [Eri04].

In this section, we cover prior work on collision detection research utilizing SDF representations, as well as the challenges they introduce. We emphasize methods developed for interactive and real-time applications, where performance is a primary concern.

2.1. SDF representations

Collision detection involving distance fields has long been a focus of research in computer graphics. Early work by McNeely et al. [MPT99, MPT05] introduced efficient voxmap-pointshell algorithms for resolving collisions between point clouds and voxelized distance fields in haptic and rendering contexts. These approaches were later enhanced by Xu et al. [XB17], who introduced Continuous Collision Detection (CCD) variants to improve temporal robustness.

To mitigate the storage overhead inherent in sampled distance fields, various compression and discretization schemes have been proposed, such as adaptively sampled distance fields [FPRJ00, BMF05]. Koschier et al. [KDBB17] further improved accuracy by fitting local polynomial approximations to grid cells. Work in rendering has likewise leveraged polynomial approximations for adaptive ray-marching steps that reduces surface overshooting and undershooting [BV25]. More recently, neural architectures have emerged as compact, continuous alternatives for complex shapes [PFS*19, SJ22]. Our methodology remains agnostic to the specific SDF representation, treating it as a query interface. We demonstrate consistent performance across analytical and voxel-based representations, although other representations should pose no problem.

2.2. SDF-SDF collision detection

Recent research has advanced SDF-SDF collision detection by framing the interaction as a continuous optimization task. Lopez-Adeva et al. [LM24] proposed a framework for computing interactions between convex bodies as a convex optimization problem using the Ellipsoid method. Their algorithm derives exact minimum distances and penetration depths directly from SDF representations, achieving performance and robustness comparable to well-established support-function methods, such as the Gilbert–Johnson–Keerthi (GJK) algorithm [GJK88]. However, their formulation is strictly limited to convex geometry. To handle

more general shapes, Liu et al. [LZW*24] introduced a hierarchical real-time pipeline that utilizes interval arithmetic for analytic functions and gradient-based optimization for general fields. Their method mitigates local minima by decomposing geometries into multiple parts via oriented bounding boxes. In contrast, our method focuses on computing a persistent contact manifold for arbitrary geometries using curvature-aware initialization and bypasses the computational overhead of hierarchical bounding volume hierarchies and shape decomposition. We compare to their approach in this work. Concurrent work by Nielsen and Erleben [NE26] formulate SDF-based CCD as an alternating spatial-temporal optimization, reaching sub-millisecond time-of-impact queries for complex geometries. Our work focuses on the DCD regime, which is complementary to CCD algorithms that focus on determining the exact time of impact.

2.3. Continuous optimization for mesh-SDF hybrids

Interactions between SDFs and triangle meshes have seen significant recent progress through the lens of nonlinear programming. Macklin et al. [MEM*20] proposed an optimization-driven approach to identify the deepest penetration between a triangle and an SDF by treating the triangle as a compact convex feasible set. The approach was recently extended by Pelletier-Gu  nette et al. [PG-MAA25] into the CCD regime, utilizing both spatial and temporal gradient information to ensure robust detection for thin, high-velocity geometry. Although these mesh-based approaches leverage the simplicity of the triangle’s geometry to avoid costly projections, SDF-SDF collisions present a more complex landscape.

2.4. Warm-starting in simulation

Many algorithms in physics simulation leverage temporal coherence, or “warm-starting”, to initialize time-critical routines across successive frames. Previous work on the GJK algorithm demonstrated that caching the simplex from previous steps drastically reduces iteration counts for distance queries of convex objects [GJK88, dB99], while Lin and Canny [LC91] exploited feature-level coherence to track closest points between polyhedra. Warm-starting contact impulses has also become standard in impulse-based constraint solvers to stabilize stacking and accelerate convergence [CB21, AEF22]. Inspired by these approaches, we apply temporal contact caching to SDF-SDF collision queries to accelerate seed optimization, pairing it with a priority queue selection scheme to maintain a sparse contact manifold across simulation steps.

2.5. Implicit operators and field quality

Implicit surface modeling pipelines often construct complex shapes by composing primitives using Constructive Solid Geometry [Ric73, PASS95], non-linear warping [Bar84], and soft blending [GBC*13]. While these operators provide expressive control, they fundamentally alter the underlying scalar field. An exact SDF satisfies the Eikonal constraint $\|\nabla\phi\| = 1$. However, boolean operations can create non-differentiable derivative ridges, spatial warping distorts gradient magnitudes ($\|\nabla\phi\| \neq 1$), and blending alters

Algorithm 1 Pseudocode summarizing our SDF-SDF discrete collision detection method between a single pair of bodies A and B.

Require: bodies A, B

```

1:  $\mathcal{V} \leftarrow$  bounding volume overlap of A and B
2:  $\mathcal{M} \leftarrow \emptyset$ 
3:
4: // Warm-start samples in cache
5: for  $\mathbf{p} \in \text{CACHE}(\mathbf{A}, \mathbf{B})$  do
6:    $\mathbf{p} \leftarrow$  integrate  $\mathbf{p}$  by velocities of A and B // Eq. 6
7:    $\mathbf{x}^* \leftarrow \text{MINIMIZE}(\mathbf{p}, \phi_A, \phi_B)$  // Eq. 3
8:    $d, \mathbf{n} \leftarrow \text{CONTACT}(\mathbf{x}^*, \phi_A, \phi_B)$  // Eq. 5
9:    $\mathcal{M} \leftarrow \mathcal{M} \cup \{\mathbf{x}^*, d, \mathbf{n}\}$ 
10: end for
11:
12: // Stochastic samples in  $\mathcal{V}$ 
13: for  $i \in 1 \dots n_{\text{rand}} + \max(0, n_{\text{cache}} - |\text{CACHE}(\mathbf{A}, \mathbf{B})|)$  do
14:    $\mathbf{p} \leftarrow \text{STOCHASTICSAMPLE}(\mathcal{V})$  // Section 3.5.2
15:    $\mathbf{x}^* \leftarrow \text{MINIMIZE}(\mathbf{p}, \phi_A, \phi_B)$  // Eq. 3
16:    $d, \mathbf{n} \leftarrow \text{CONTACT}(\mathbf{x}^*, \phi_A, \phi_B)$  // Eq. 5
17:    $\mathcal{M} \leftarrow \mathcal{M} \cup \{\mathbf{x}^*, d, \mathbf{n}\}$ 
18: end for
19:
20: // Remove duplicates based on spatial proximity
21: for  $\{\mathbf{x}_i, d_i, \mathbf{n}_i\} \in \mathcal{M}$  do
22:   for  $\{\mathbf{x}_j, d_j, \mathbf{n}_j\} \in \mathcal{M}$  do
23:     if  $i \neq j \wedge \|\mathbf{x}_i - \mathbf{x}_j\| < \tau$  then
24:        $\mathcal{M} \leftarrow \mathcal{M} \setminus \{\mathbf{x}_j, d_j, \mathbf{n}_j\}$ 
25:     end if
26:   end for
27: end for
28:  $\mathcal{L} \leftarrow \{\mathbf{x}_i, d_i, \mathbf{n}_i\} \in \mathcal{M}$  sorted by  $g(\mathbf{x}_i)$  in ascending order
29:  $\text{CACHE}(\mathbf{A}, \mathbf{B}) \leftarrow \mathcal{L}_{1 \dots n_{\text{cache}}}$  // Store best  $n_{\text{cache}}$  samples
30: Return  $\mathcal{M}$ 

```

local field steepness, transforming exact SDFs into distorted, quasi-exact scalar fields [BV25].

With regards to collision detection pipelines, these field distortions introduce numerical challenges. First-order optimizers, such as gradient descent, rely on well-behaved field gradients to resolve penetration depth and compute contact normals. On procedurally modeled or warped SDFs, gradient-based queries stall, “chatter” along interior ridges, or diverge in non-metric regions. Because real-world physics solvers require stable, reliable contact manifolds across arbitrary edited geometries, handling inexact distance fields is critical. We evaluate our framework on distorted and inexact SDFs, demonstrating robust performance and convergence stability even under non-linear spatial perturbations.

3. Methodology

Our framework solves the narrow-phase intersection problem between SDFs by algorithmically coupling a robust optimization routine with a targeted sampling strategy. Potential collision points are initialized via a hybrid strategy of temporal warm-starting and stochastic importance sampling guided by curvature-based heuristics. This strategy exploits the spatio-temporal coherence of col-

lision points across frames, while also allowing the optimizer to better navigate non-convex regions and avoid local minima. The methodology is developed with efficiency and real-time performance in mind, producing stable and persistent contact manifolds for simulations of thousands of interacting bodies at real-time frame rates. The pseudocode of our algorithm is provided in Algorithm 1, and further details regarding the principal steps are provided below.

3.1. Broad-phase and domain pruning

Prior to the narrow-phase optimization of discrete contact points, a broad-phase test isolates candidate intersection regions. For each pair of bodies, we compute the intersection of their axis-aligned bounding boxes (AABBs), $\mathcal{V} = \mathcal{B}_A \cap \mathcal{B}_B$. Because an SDF is strictly positive outside its object boundary, intersection is restricted to $\mathcal{V}$ and regions outside this overlap are ignored. This intersection volume acts as the search domain where candidate collision points are stochastically sampled and optimized toward points of maximum penetration depth.

3.2. Narrow-phase optimization problem

The objective of the narrow-phase is to identify the point of deepest intersection between two colliding bodies, A and B, each represented by signed distance field $\phi \in \mathbb{R}$ and evaluated at point $\mathbf{x} \in \mathbb{R}^3$. The intersection of two such implicit volumes can be characterized by the pointwise maximum of their respective distance functions

$$\max(\phi_A(\mathbf{x}), \phi_B(\mathbf{x})), \quad (1)$$

where $\phi < 0$ denotes the interior of a body and $\phi > 0$ outside. The “hardmax” formulation in Eq. 1 produces a sharp transition in gradient and values as the point moves between bodies (e.g., where two distance fields meet), which can introduce numerical instabilities or stalled convergence in iterative solvers. To give a function landscape that is more amenable to optimization, we use a “softmax” approximation:

$$\frac{\phi_A(\mathbf{x}) + \phi_B(\mathbf{x}) + \sqrt{(\phi_A(\mathbf{x}) - \phi_B(\mathbf{x}))^2 + \epsilon}}{2}. \quad (2)$$

This formulation is an upper bound on Eq. 1, providing a conservative smooth surrogate that guarantees candidate points lie within the true intersection volume while eliminating gradient discontinuities along the equal-distance boundary where $\phi_A(\mathbf{x}) = \phi_B(\mathbf{x})$. In other words, the softmax behaves much like the hardmax version when point $\mathbf{x}$ is solidly inside either body A or B, and giving a smooth C^1 continuous transition when $\phi_A(\mathbf{x}) \approx \phi_B(\mathbf{x})$ [BBCW10].

A further aim of our framework is to generate stable contact manifolds for SDF collisions given a limited number of sample points. The contact manifold $\mathcal{M}$ is the collection of points $\mathbf{x}$, normals $\mathbf{n}$, and penetration depths d that describe the interface between two colliding objects. A manifold with collision points that are located close together can produce jitter and pivoting behavior, due to sudden changes in the deepest penetrating point. This can lead not only to jarring visual behavior, but also numerical instabilities.

Adding a repulsion term to the objective function for finding

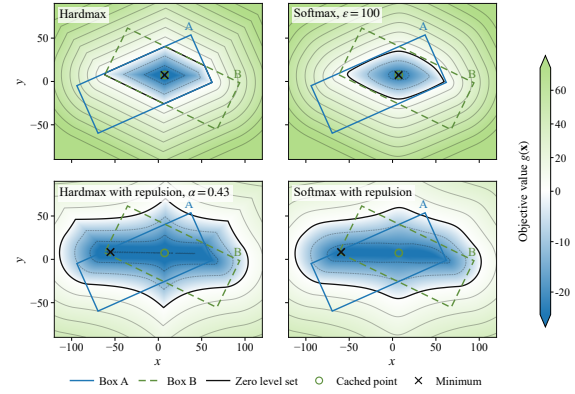


Figure 2: Objective function landscapes for overlapping 2D boxes A (solid blue) and B (dashed green): hardmax, softmax approximation, hardmax and repulsion, and softmax and repulsion.

the deepest intersection point encourages exploration of overlapping areas where current collision points do not exist. Therefore, the function in Eq. 2 is augmented with a penalty term that promotes points that are situated far from other collision points on the manifold, such that

$$g(\mathbf{x}) = \frac{\phi_A(\mathbf{x}) + \phi_B(\mathbf{x}) + \sqrt{(\phi_A(\mathbf{x}) - \phi_B(\mathbf{x}))^2 + \epsilon}}{2} - \alpha \sum_{\{\mathbf{x}_i, d_i, \mathbf{n}_i\} \in \mathcal{M}} \|\mathbf{x} - \mathbf{x}_i\|, \quad (3)$$

where the parameter α controls the influence of the repulsion. The deepest intersection point $\mathbf{x}^* \in \mathbb{R}^3$ can then be found by minimizing the intersection objective in Eq. 3:

$$\mathbf{x}^* = \arg \min_{\mathbf{x}} g(\mathbf{x}). \quad (4)$$

In our implementation, we set $\epsilon = 0.1$, which provides a smoothing region roughly ten times the convergence tolerance. The repulsion parameter is determined experimentally across a collection of challenging simulations, with $\alpha = 0.01$ used in our experiments. Furthermore, by using a square-root-based softmax approximation rather than logarithmic or exponential variants, we achieve more efficient computational throughput on modern GPU hardware.

Fig. 2 visualizes how the terms in Eq. 1–3 reshape the objective landscape for a 2D example. The softmax rounds the hardmax ridge where the two distance fields are equal while preserving the overlap minimum, and the repulsion term moves the selected minimum away from an existing cached contact so that subsequent samples cover a wider contact manifold.

3.3. Narrow-phase optimization method

We evaluate several optimization paradigms to minimize the smoothed softmax objective defined in Eq. 2. While gradient-based solvers, such as Stochastic Gradient Descent (SGD) [RM51], offer an intuitive approach given the analytical availability of SDF gradients, they are often susceptible to stagnation within the local minima of highly non-linear intersection landscapes. Recent work has introduced the Ellipsoid method as an efficient technique for resolving interactions [LM24]. However, its convergence is predicated on the assumption of overlapping convex regions.

For general SDF-SDF collisions involving highly non-convex geometries, we find that simplex-based methods, namely the derivative-free Nelder-Mead algorithm [NM65], provide better robustness and run-time performance. By maintaining a simplex that adapts to the local topology of the intersection volume, the algorithm effectively navigates non-convex basins of attraction that might otherwise impede gradient-descent convergence. Although, the Ellipsoid method remains competitive for convex shapes. Our framework is agnostic to the underlying optimizer.

A convergence test of $\|\mathbf{x}^{(k+1)} - \mathbf{x}^{(k)}\| < \tau$, when the change in $\mathbf{x}$ between two sequential iterates is less than τ , determines the termination criteria of the optimizer. As detailed in the Results section, we conduct a comparison of these optimizers across various simulation benchmarks. Our findings indicate that while the Ellipsoid method serves as a performant algorithm for certain near-convex configurations, the Nelder-Mead solver remains the most consistent choice for resolving the contact manifolds characteristic of arbitrary implicit geometry.

3.4. Contact data synthesis

Once the narrow-phase optimizer converges to a candidate collision point $\mathbf{x}^*$, we extract the local collision geometry required for a physics solver. We evaluate the signed distance values ϕ_A, ϕ_B and their respective gradients $\nabla\phi_A, \nabla\phi_B$ at $\mathbf{x}^*$. For samples where $\phi_A(\mathbf{x}^*) < 0$ and $\phi_B(\mathbf{x}^*) < 0$, the point $\mathbf{x}^*$ is classified as an active contact within the intersecting region.

The penetration depth d and contact normal $\mathbf{n}$ are obtained from the more deeply interpenetrating field, ensuring that the contact normal is derived from the most locally pertinent geometric features:

$$(d, \mathbf{n}) = \begin{cases} (\phi_A(\mathbf{x}^*), \nabla\phi_A(\mathbf{x}^*)) & \phi_A(\mathbf{x}^*) < \phi_B(\mathbf{x}^*) \\ (\phi_B(\mathbf{x}^*), -\nabla\phi_B(\mathbf{x}^*)) & \text{otherwise.} \end{cases} \quad (5)$$

In the second case, the gradient is negated so that the normal $\mathbf{n}$ always points from body A toward body B.

3.5. Stochastic seeding and manifold initialization

Computing the intersection of arbitrary, non-convex SDFs presents a highly non-linear objective landscape with numerous local minima. Consequently, resolving the single deepest point of interpenetration is not sufficient for complex interactions and fails to stabilize collisions between non-convex bodies [GBF03]. Our methodology must instead identify a diverse set of points to produce a stable contact manifold. In a large-scale simulation involving thousands of interacting bodies, which characterizes our target application, the computational cost of dense surface sampling is prohibitive. Thus our sampling strategy for identifying collision points aims to provide sufficient spatial coverage to correctly constrain rotational and translational motion, while maintaining a small sampling budget for real-time throughput.

To achieve this, we employ a hybrid initialization scheme that balances temporal persistence with stochastic global exploration. We provide the optimizer with initial estimates derived from two

primary sources: (i) cached candidates that leverage temporal coherence to maintain stable support across frames, and (ii) stochastic importance sampling, which is not tethered to historical collision data and ensures robustness to disjoint collision events.

By seeding the optimizer with this mixture of persistent and exploratory candidates, we ensure robust convergence toward globally consistent contact manifolds. These strategies are described below.

3.5.1. Warm-starting collisions

We exploit the temporal coherence inherent to many physics simulations to produce stable contact manifolds across successive time steps. Rather than re-initializing the narrow-phase search from a cold start, we maintain a cache of up to n_{cache} candidate points for each interacting pair of bodies. This “warm-starting” strategy significantly reduces the number of iterations required for convergence by seeding our optimizer in the immediate neighborhood of the previously resolved minimum

As shown in the adjacent figure, the position of a cached point $\mathbf{p} \in \mathbb{R}^3$ from the previous frame are advected using a first-order integration of the velocities of the interacting bodies over time step Δt , such that

$$\mathbf{p} \leftarrow \mathbf{p} + \Delta t \mathbf{v}.$$

where $\mathbf{v} \in \mathbb{R}^3$ is computed as the average velocity of A and B:

$$\mathbf{v} = \frac{\text{VEL}(\mathbf{A}, \mathbf{p}) + \text{VEL}(\mathbf{B}, \mathbf{p})}{2}. \quad (6)$$

The function VEL returns the world-space velocity of a point on a body given its current linear and angular velocity. In cases where one body is static, only the velocity of the moving body is used.

We found that this warm-starting approach not only accelerates convergence, but also prevents “chatter” in the contact manifold that typically arises from purely stochastic initialization.

At the end of each time step, n_{cache} samples are selected from each collision manifold. First, the existing collision points are sorted in ascending order, since lower values are better, such that

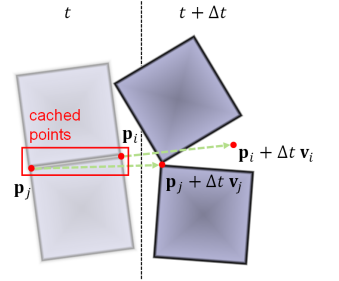
$$\mathcal{L} = \text{sort}(\{\mathbf{x}_i \mid \{\mathbf{x}_i, d_i, \mathbf{n}_i\} \in \mathcal{M}\} \text{ by } g(\mathbf{x}_i)).$$

The cache of the manifold A,B is then updated to store $\mathcal{L}_{1 \dots n_{\text{cache}}}$.

When the temporal cache contains fewer than n_{cache} points, we supplement the manifold by drawing $n_{\text{cache}} - |\text{CACHE}(\mathbf{A}, \mathbf{B})|$ additional candidates using our stochastic sampling strategy. This initialization mechanism is critical for seeding a robust contact manifold during the time step when two bodies initially enter into collision, specifically, the first frame where their bounding volumes overlap.

3.5.2. Stochastic sampling of collision points

Standard optimization techniques tend to gravitate toward the nearest local minimum. However, for complex scenes consisting of



non-convex geometry, the intersection points may change significantly from one time step to the next. By scattering random seeds across the overlapping bounding volume, we increase the probability that at least one of these seeds starts in a basin of attraction toward global minima across multiple disjoint regions. Therefore, in addition to n_{cache} warm-started collision points, we randomly sample n_{rand} points from the overlapping region. Providing these additional collision points helps to produce contact manifolds that adapt to changes in the contact configuration between two bodies, especially during cases where the relative velocity is significant.

We observed in early experiments that in challenging contact scenarios, collision points are often located in regions of high-curvature. Seed points are therefore precomputed in regions of high curvature and selected as potential collision points at run-time. Specifically, high curvature seed points within the overlapping bound volumes are randomly selected and used to initialize the optimizer, a process referred to as STOCHASTICSAMPLE in Alg. 1.

Computing high-curvature seed points. The mean curvature is effectively the divergence of the gradient vector field of the SDF, which near the surface is given by the Laplacian

$$\Delta\phi = \frac{\partial^2\phi}{\partial x^2} + \frac{\partial^2\phi}{\partial y^2} + \frac{\partial^2\phi}{\partial z^2},$$

which we approximate at $\mathbf{x}$ using finite differences:

$$\Delta\phi(\mathbf{x}) \approx \frac{(\nabla\phi(\mathbf{x} + \delta\mathbf{e}_i) - \nabla\phi(\mathbf{x} - \delta\mathbf{e}_i)) \cdot \mathbf{e}_i}{2\delta}. \quad (7)$$

Note that $\mathbf{e}_i \in \mathbb{R}^3$ is the i th standard orthonormal basis vector in Cartesian coordinates. The curvature is computed for a dense sampling of points on the surface of each SDF, and points where $|\Delta\phi| > \kappa$ are saved as good initialization points. If the distance field is generated from a 3D mesh, this can be done directly using a Poisson disk sampling on the mesh. Otherwise, we randomly sample points in the axis-aligned bounding box of the SDF and project them to the surface using the gradient. At run-time, high-curvature points inside the bounding volumes of both bodies are then randomly selected as potential collision points. Fig. 3 shows the seed points generated for a collection of geometries used in our experiments. Our SDF models are normalized to fit inside a $1 \times 1 \times 1$ bounding box and a fixed $\kappa = 4$ and $\delta = 0.001$ is used across all models. These parameters were chosen empirically though testing in various scenes.

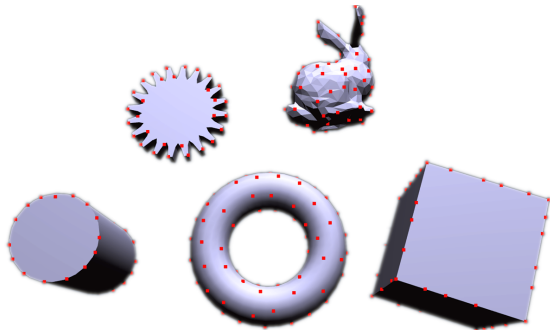


Figure 3: High-curvature points ($\kappa = 4$) for a selection of geometries used in our experiments.

3.6. SDF representations

Our framework is evaluated using a variety of implicit representations. Specifically, we utilize two common paradigms:

Analytical primitives and composites. For basic geometries, such as boxes, cones, and cylinders, use exact closed-form functions to evaluate pointwise distances and gradients. These functions can be combined to create more complex shapes, such as the spiky plates shown in several of our experiments.

Voxel-based discretizations. For complex, arbitrary geometries, we represent the SDF as a sampled scalar field on a regular grid, a common approach for large-scale simulation [AEF22, KDBB17]. These volumes are computed from polygon mesh geometry in a preprocessing step. Unless otherwise specified, we use a grid resolution of 100^3 and trilinear interpolation for distance queries.

3.7. Other details

Following each time step, and to encourage manifold diversity, the collision points within each manifold $\mathcal{M}$ are pruned to mitigate spatial redundancy. Despite the inclusion of a repulsion term in the objective function, multiple seeds may still converge toward the same local minimum or closely adjacent basins of attraction. To maintain a diverse and sparse contact support, collision points found to be within a specified proximity threshold are consolidated, and redundant samples are discarded from the temporal cache.

For the Ellipsoid optimizer, we additionally warm-start the ellipse using the same approach as Lopez-Adeva et al. [LM24]. For the Nelder-Mead optimizer, the initial simplex is constructed as a tetrahedron centered on $\mathbf{p}$ that is 10 percent the size of the intersected bounding box $\mathcal{V}$.

4. Results

In this section, we assess the performance and physical stability of our collision framework across a series of benchmark scenarios. All experiments were conducted on a workstation equipped with an Intel Core i9 13900H processor, 32 GB of RAM, and an NVIDIA GeForce RTX 4090 GPU. For the large-scale simulation benchmarks (Sec. 4.1), we utilize a GPU-accelerated version that resolves contact manifolds for all interacting pairs in parallel. All other experiments (Sec. 4.2–4.5) are performed using a single-threaded CPU implementation. Our CPU simulator implements a velocity-level, constraint-based formulation [AEF22]. The solver constructs the Delassus operator using object masses and constraint Jacobians, incorporating Baumgarte stabilization to resolve interpenetration drift and uses a sequential impulses solver. In contrast, our GPU implementation adopts the Augmented Vertex Block Descent (AVBD) formulation proposed by Giles et al. [GDY25], combining primal constraint minimization using block coordinate descent with dual augmented Lagrangian updates to stably resolve stiff contacts. Source code for our C++ collision and simulation framework is available here: <https://github.com/savant117/sdf-dcd-demo>

Unless otherwise specified, we employ a consistent parameter set for all experiments: cached samples $n_{\text{cache}} = 4$, stochastic samples $n_{\text{rand}} = 1$, repulsion weight $\alpha = 0.01$, optimizer termination



Figure 4: Bunny Iron Maiden: A group of bunnies is sandwiched between plates with spikes. All geometries are encoded by analytical (spikes and plates) and voxel-based signed distance functions (bunnies). This challenging test creates a landscape of narrow basins of attraction and disjoint contact regions.

Table 1: Run-time performance of various scenes using our algorithm with Nelder-Mead (NM), Stochastic Gradient Descent (SGD), and Ellipsoid (E) optimizers. All simulations use time step $\Delta t = 16$ ms and tolerance $\tau = 0.01$.

Scene	# bodies	t_{coll} (NM)	t_{coll} (E)	t_{coll} (SGD)
Spikes (cows)	583	0.7 ms	1.0 ms	1.3 ms
Funnel (cylinders)	16000	2.2 ms	1.8 ms	7.1 ms
Funnel (horses)	16000	20.0 ms	20.2 ms	42.3 ms
Colosseum	8819	0.5 ms	0.4 ms	1.9 ms
Bunnies	4314	1.8 ms	2.4 ms	5.1 ms

Table 2: Performance of the Bunnies benchmark under varying cache size (n_{cache}) and stochastic samples (n_{rand}).

n_{cache}	n_{rand}	t_{coll}
4	1	1.8 ms
4	2	2.4 ms
4	4	4.2 ms
6	1	2.7 ms
6	2	3.0 ms
6	4	4.4 ms
8	1	3.9 ms
8	2	4.2 ms
8	4	5.0 ms
16	1	7.7 ms
16	2	8.1 ms
16	4	8.9 ms

tolerance $\tau = 0.01$, softmax smoothing $\epsilon = 0.1$, and simulation time step $\Delta t = 16$ ms. These values were chosen to balance the competing requirements of contact precision and computational throughput. Specifically, τ is calibrated based on the scale of our SDFs, and ϵ is selected to ensure C^1 continuity without excessive smoothing of sharp features. In our testing, we found that the framework exhibited robust convergence across a range of parameter variations.

4.1. Run-time performance

The performance of our collision framework is evaluated on a number of large-scale scenes, involving hundreds or thousands of bodies. These scenes are simulated using a GPU implementation of

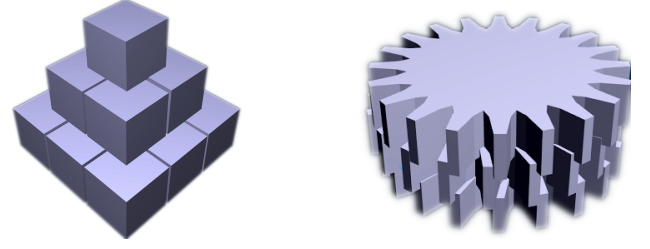


Figure 5: Box-stacking (left) and Gear-stacking (right). The boxes are represented by analytical shapes, whereas the gears are encoded as voxel-based SDFs with 100^3 resolution.

Alg. 1 as a kernel executed for each body pair. Screenshots from these scenes are shown in Fig. 1.

Performance statistics from this experiment are summarized in Tab. 1. The total time for collision detection per simulation frame, t_{coll} is shown for the Nelder-Mead, Ellipsoid method, and SGD methods. We observe that the N-M optimizer converges well for scenes involving arbitrary geometry, such as the cows, horses, and bunny geometries. Whereas the Ellipsoid method also demonstrates good performance, converging with comparable wall-clock time, and even being the most performant method for simulations involving convex geometries such as cylinders and the boxes in the Colosseum example. SGD gives the worst performance of the three optimizers, with collision detection times more than twice the other optimizers in most cases.

The funnel example with 16k horse meshes is one of the most geometrically demanding scenes, yet our framework achieves near real-time performance while maintaining high-fidelity physical motion (see supplementary video). In contrast, while the cylinder scene contains a comparable rigid-body count, it exhibits a significantly lower collision detection time (t_{coll}). This performance disparity stems from two primary factors. First, the cylindrical geometry is convex, allowing the optimizers to converge rapidly. Second, its SDF is evaluated via a closed-form analytical function, whereas the horse model requires a more computationally expensive voxel grid lookup. This contrast highlights the benefits of our collision framework, which constructs a stable contact manifold using a sparse collection of point queries regardless of the underlying field representation.

As shown in Tab. 2, our framework maintains real-time performance even under expanded sampling budgets. Collision detection runtime scales linearly with the total sample count, demonstrating predictable computational overhead.

4.2. Comparison with other collision detection algorithms

We further benchmark our framework against several baseline algorithms. These include our implementation of hierarchical SDF pipeline proposed by Liu et al. [LZW*24] (octree depth=3), the Gilbert-Johnson-Keerthi (GJK) algorithm combined with the expanding polytope algorithm (EPA), and the classic separating axis theorem [Eri04]. The Box-stacking and Gear-stacking benchmarks, shown in Fig. 5, serve as the test cases for these comparisons. A comprehensive breakdown of the performance statistics across

Table 3: Statistics for the Box-stacking scene in Fig. 5 with the average collision detection time t_{coll} per time step, number of contacts, solver iterations, and number of SDF queries per time step.

Method	t_{coll}	# contacts	iterations	# SDF queries
SAT	0.09 ms	116	-	-
GJK+EPA	0.18 ms	116	1000	-
[LZW*24]	12.75 ms	372	12000	12000
NM	1.15 ms	116	1000	2500
NM (no softmax)	2.23 ms	116	2840	7100
NM (no cache)	3.03 ms	116	3800	7700
Ellipsoid	0.46 ms	116	550	550
Ellipsoid (no softmax)	3.67 ms	116	8100	8100
Ellipsoid (no cache)	1.75 ms	116	3600	3600
SGD	2.07 ms	116	6600	6000
SGD (no softmax)	3.19 ms	116	8700	8700
SGD (no cache)	2.87 ms	116	8500	8500

Table 4: Statistics for the Gear-stacking scene in Fig. 5 with the average collision detection time t_{coll} per time step, number of contacts, solver iterations, and number of SDF queries per time step.

Method	t_{coll}	# contacts	iterations	# SDF queries
SAT	1.36 ms	182	-	-
GJK+EPA	2.39 ms	182	26600	-
[LZW*24]	3.99 ms	62	1200	1200
NM	0.1 ms	12	180	420
NM (no softmax)	0.14 ms	12	460	980
NM (no cache)	0.19 ms	12	420	860
Ellipsoid	0.1 ms	12	160	160
Ellipsoid (no softmax)	0.29 ms	12	1100	1100
Ellipsoid (no cache)	0.19 ms	12	510	510
SGD	0.16 ms	12	700	700
SGD (no softmax)	0.21 ms	12	1200	1200
SGD (no cache)	FAIL	FAIL	FAIL	FAIL

these algorithms, including various ablation configurations of our framework, are summarized in Tab. 3 and 4.

As expected, for the convex Box-stacking scene, the analytic SAT and GJK+EPA implementations achieve strong performance because they avoid SDF queries by exploiting explicit geometric primitives, which in this case are simple and convex. In contrast, the hierarchical approach of Liu et al. [LZW*24] generates a higher density of collision points, largely due to its sampling strategy that evaluates all overlapping leaf nodes within octree sub-domains. Since the local overlapping volumes behave as bounded convex domains, the Ellipsoid method yields superior convergence rates compared to Nelder-Mead.

The advantages of our pipeline are more pronounced in the Gear-stacking scene. Here, baselines such as SAT and GJK+EPA suffer a performance degradation, exposing their vulnerability when subjected to geometries that demand dense, fragmented convex decompositions. Conversely, our framework maintains efficiency by restricting optimization to a fixed sample footprint per body pair by maintaining a stable contact manifold through temporal caching and a stochastic sampling strategy. Specifically, in the Box-stacking scenario, we achieve an average wall-clock time of

$t_{\text{coll}} = 0.6$ ms, while the Gear-stacking simulation, despite higher geometric complexity, achieves $t_{\text{coll}} = 0.1$ ms when using the Ellipsoid and Nelder-Mead optimizer.

We also ablate the benefits of using the smoothed softmax objective and temporal caching for each optimizer. Both components contribute substantially to the framework’s efficiency, with temporal caching yielding the most pronounced performance gains across different solver configurations. Notably, the Stochastic Gradient Descent solver exhibited poor convergence and robustness, failing to converge in gear-stacking scenario without a warm-started cache.

4.3. Penetration depth

We evaluate our framework’s capacity to mitigate geometric interpenetration by dropping various non-convex shapes—specifically the Cow and Bunny models—onto a rigid ground plane and tracking the maximum penetration depth at each frame. We average the results over 100 independent simulation rollouts, initializing the object with a randomized 3D orientation for each trial.

Because our pipeline pairs a discrete collision detection framework with a constraint-based contact model, which is standard practice in interactive physics engines [CB21, AEF22], minor penetrations naturally occur in order to generate restorative forces. Therefore, to establish an accuracy ceiling, we compare our solver against an offline geometric oracle that performs an exhaustive, brute-force evaluation using 1,000 precomputed surface points.

Fig. 6 plots these temporal penetration profiles across multiple hyperparameter configurations, sweeping the temporal cache size ($n_{\text{cache}} \in \{4, 8, 16\}$) and the stochastic sample budget ($n_{\text{sto}} \in \{1, 4\}$). The results indicate that scaling the cache to $n_{\text{cache}} = 16$ yields tracking accuracy comparable to the dense reference baseline, while a modest allocation of $n_{\text{cache}} = 4$ proves sufficient to ensure simulation stability without severe penetration artifacts. Curvature aware importance sampling further reduces transient tracking errors and frame-to-frame variance of penetration depth.

We again observe that sample caching exerts the most profound impact on error reduction, highlighting the value of leveraging spatio-temporal coherence in interactive collision pipelines.

4.4. Energy dynamics

A goal of our framework is to efficiently compute stable contact manifolds that provide rotational and translational support. In the absence of a well-distributed support, forces often fluctuate and induce a non-physical “chatter” that is characterized by velocity oscillations and poor numerical convergence toward a resting state.

To evaluate the benefits of the curvature based stochastic sampling, we monitor the total kinetic energy produced by bodies at each simulation frame. This allows us to assess the framework’s ability to settle and reach a stable state. We compare these results against a reference baseline using a dense stochastic sampling of 1000 seed points per body, which serves as the “ground truth” for the energy dissipation profile. We utilize some of the more geometrically challenging scenarios—the Pile of Chairs and Bunny Iron Maiden shown in Fig. 9 and Fig. 4 respectively. Simulation results are averaged over 5 rollouts.

Table 5: Average per time step statistics for the operator-distorted Falling Boxes scene (see Fig. 8), including collision time t_{coll} per time step, number of contacts, solver iterations, and number of SDF queries averaged over all simulation frames.

Operator / Solver	t_{coll}	# contacts	iterations	# SDF queries
Undistorted				
NM	0.33 ms	45.7	1,066	2,302
Ellipsoid	0.66 ms	36.8	1,949	1,949
SGD	0.71 ms	52.2	5,269	5,269
BEND ($k = 0.15$)				
NM	0.77 ms	46.0	2,389	5,010
Ellipsoid	1.80 ms	39.8	5,149	5,149
SGD	1.25 ms	49.8	8,166	8,166
BEND ($k = 0.55$)				
NM	0.75 ms	39.8	2,346	4,896
Ellipsoid	1.78 ms	37.2	5,223	5,223
SGD	1.22 ms	40.5	8,054	8,054
BEND ($k = 0.95$)				
NM	0.90 ms	36.4	2,842	5,870
Ellipsoid	1.78 ms	36.8	5,224	5,224
SGD	1.24 ms	43.7	7,557	7,557
DISPLACE ($k = 0.05$)				
NM	0.69 ms	42.0	1,707	3,600
Ellipsoid	1.28 ms	34.5	3,317	3,317
SGD	1.27 ms	42.3	6,506	6,506
DISPLACE ($k = 0.10$)				
NM	0.73 ms	36.1	1,861	3,939
Ellipsoid	1.87 ms	33.4	5,077	5,077
SGD	1.26 ms	41.6	6,734	6,734
DISPLACE ($k = 0.20$)				
NM	0.91 ms	36.1	2,330	4,982
Ellipsoid	2.25 ms	32.6	6,144	6,144
SGD	1.51 ms	41.5	7,992	7,992
TWIST ($k = 0.50$)				
NM	0.99 ms	45.8	3,161	6,561
Ellipsoid	2.47 ms	44.4	7,092	7,092
SGD	1.38 ms	47.8	8,635	8,635
TWIST ($k = 2.50$)				
NM	1.14 ms	42.1	3,354	7,081
Ellipsoid	2.41 ms	38.0	7,073	7,073
SGD	1.60 ms	52.8	9,636	9,636
TWIST ($k = 5.00$)				
NM	1.01 ms	43.5	2,743	5,871
Ellipsoid	2.26 ms	39.2	6,747	6,747
SGD	1.43 ms	50.6	8,526	8,526

As shown in Fig. 7, simulations conducted with curvature-based importance sampling settle more rapidly, attaining a low-energy stable steady state that is similar to the reference simulation. Without the just uniform stochastic sampling, there remains an elevated and persistent kinetic energy in the system. Importance sampling allows the simulation to settle rapidly, better approximating the asymptotic behavior of the reference simulation. The supplementary video also shows animations for these simulations.

4.5. Evaluation on Operator-Distorted Fields

We assess robustness to inexact SDFs by testing fields distorted by classic procedural warping operators: spatial bending (BEND), additive sinusoidal displacement (DISPLACE), and axis-aligned twisting (TWIST). Our implementation is inspired by Quilez [Qui]:

$$\text{BEND}(\mathbf{x}, k) = \phi(\mathbf{q}), \quad \mathbf{q} = \begin{pmatrix} \cos(k\mathbf{x}_x) & -\sin(k\mathbf{x}_x) & 0 \\ \sin(k\mathbf{x}_x) & \cos(k\mathbf{x}_x) & 0 \\ 0 & 0 & 1 \end{pmatrix} \mathbf{x},$$

$$\text{TWIST}(\mathbf{x}, k) = \phi(\mathbf{q}), \quad \mathbf{q} = \begin{pmatrix} \cos(k\mathbf{x}_y) & 0 & -\sin(k\mathbf{x}_y) \\ \sin(k\mathbf{x}_y) & 0 & \cos(k\mathbf{x}_y) \\ 0 & 1 & 0 \end{pmatrix} \mathbf{x},$$

$$\text{DISPLACE}(\mathbf{x}, k) = \phi(\mathbf{x}) + \sin(k\mathbf{x}_x) \sin(k\mathbf{x}_y) \sin(k\mathbf{x}_z).$$

As illustrated by the Falling Boxes example in Fig. 8, these distortions introduce varying degrees of spatial perturbation, altering both the surface geometry and the metric structure of the underlying SDF. Example animations produced by for each distortion operator can also be found in the supplemental video. We assess our framework’s ability to maintain convergence stability and performance despite operator-induced gradient changes and metric ill-conditioning for the NM, Ellipsoid, and SGD optimizers across varying distortion parameters k . Statistics from these experiments are presented in Table 5.

As expected, spatial warping increases overall optimization overhead, yet the derivative-free Nelder–Mead proves most resilient to field distortions compared to gradient-based methods, maintaining real-time performance ($t_{\text{coll}} \leq 1.00$ ms) and the lowest absolute iteration count across all distortions. Conversely, Ellipsoid suffers iteration “bloat”, adding $> 5k$ iterations under twisting (TWIST with $k = 5.0$). Interestingly, SGD requires the highest absolute iteration count, but exhibits the lowest growth relative to the undistorted baseline. This is likely due to the fixed step size that attenuates sensitivity to non-linear warping.

Among the evaluated operators, spatial twisting imposes the highest computational penalty, producing a $2.0 \times$ to $3.6 \times$ run-time increase over the undistorted baseline. Bending and displacement induce more moderate slowdowns. Notably, contact counts remain stable across all solvers and distortion parameters (≈ 35 – 52 contacts per time step), giving some assurance that our framework preserves manifold coverage regardless of scalar field perturbations.

The increased wall-clock collision time t_{coll} for Ellipsoid and SGD is likely driven by compounding higher iteration counts with heavier per-iteration gradient evaluations, caused by trigonometric operations within the distortion functions. This further underscores the necessity of our framework’s sparse collision manifold and highlights the advantages of derivative-free optimizers like Nelder–Mead. Avoiding gradient evaluations additionally reduces overhead and bypasses metric distortions caused when procedural warping breaks the Eikonal property e.g., $\|\nabla\phi\| \neq 1$.

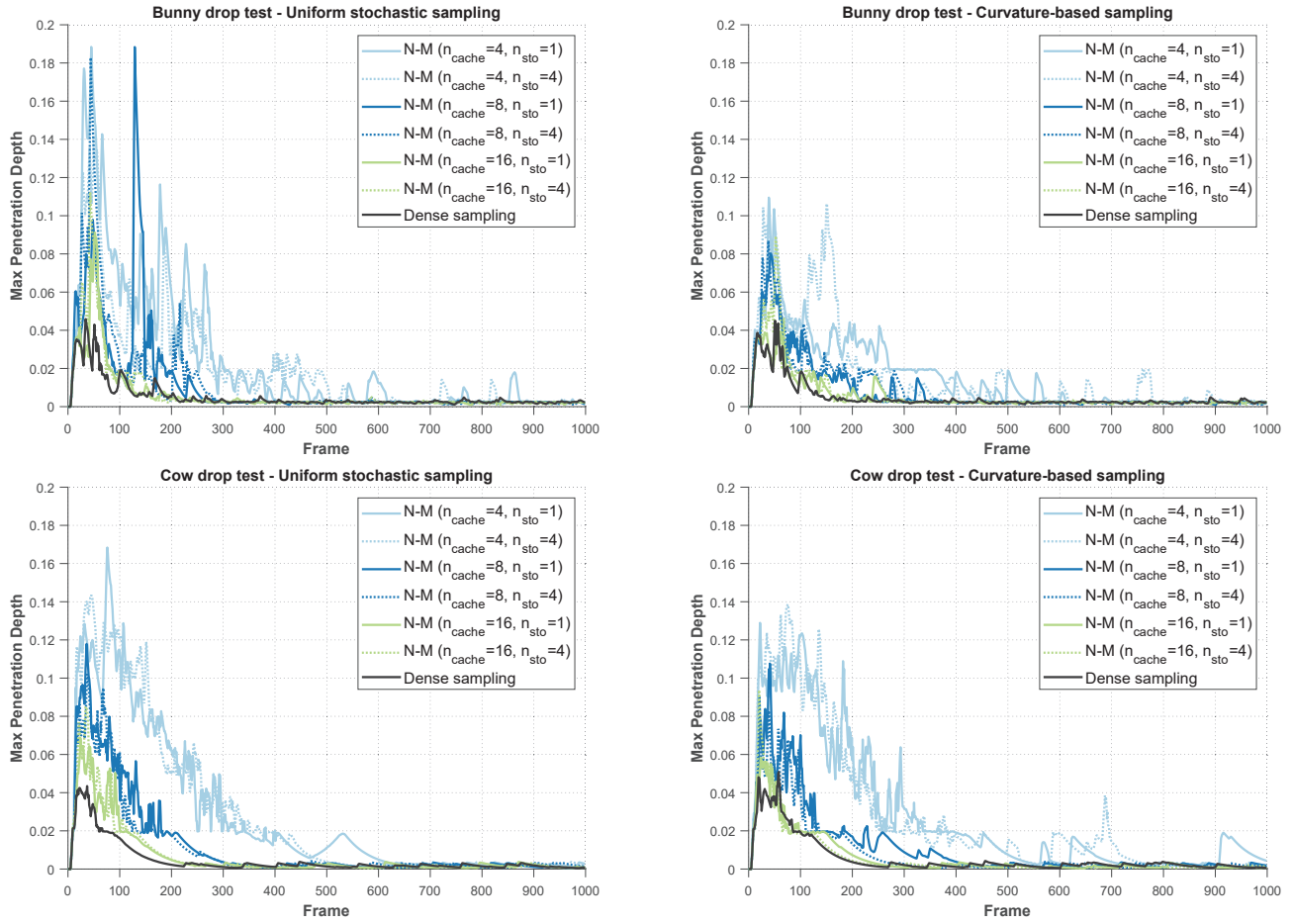


Figure 6: Maximum penetration depth over time for the Bunny (top) and Cow (bottom) under varying parameters. Each curve tracks the maximum penetration across 100 randomized initial orientations dropped onto a ground plane. Increasing the temporal cache capacity (n_{cache}) drives our method to converge toward the offline oracle baseline (dense grid-sampled SDF with 1000 surface points). Curvature-based importance sampling (right) significantly suppresses interpenetration and compared to a uniform stochastic sampler (left).

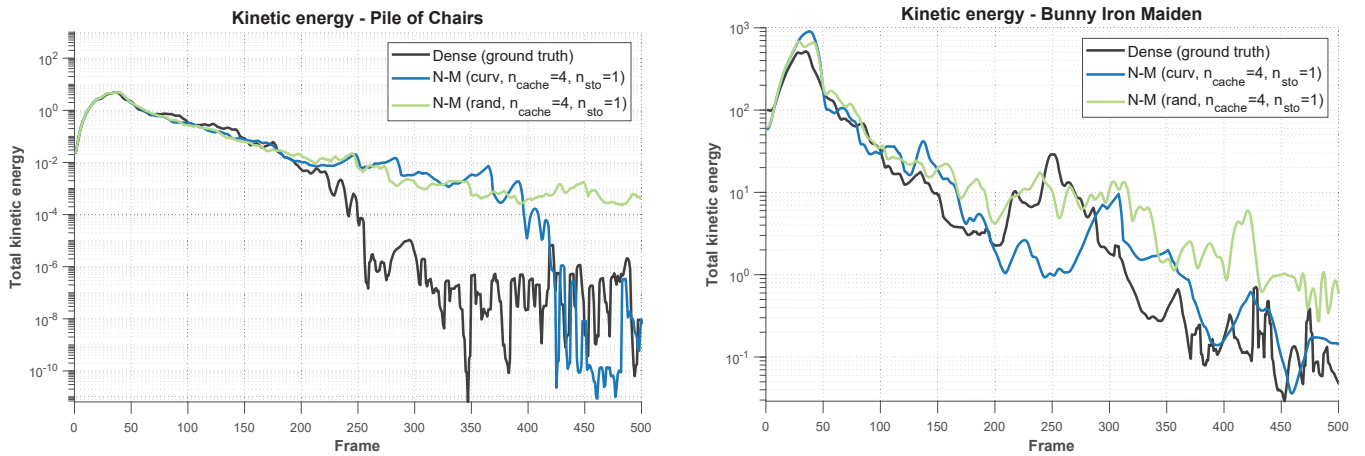


Figure 7: Total kinetic energy in the Pile of Chairs (left) and Bunny Iron Maiden (right) simulations, using curvature based importance sampling (“curv”) and uniform stochastic sampling (“rand”). A comparison to a reference simulation with dense sampling provides an energy dissipation baseline.

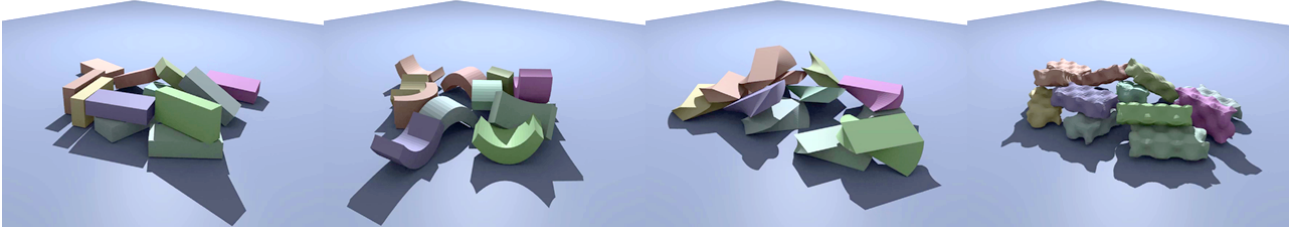


Figure 8: Distortions of a pile of Falling Boxes. Left-to-right: Undistorted, BEND ($k = 0.95$), TWIST ($k = 0.20$), DISPLACE ($k = 5.0$).

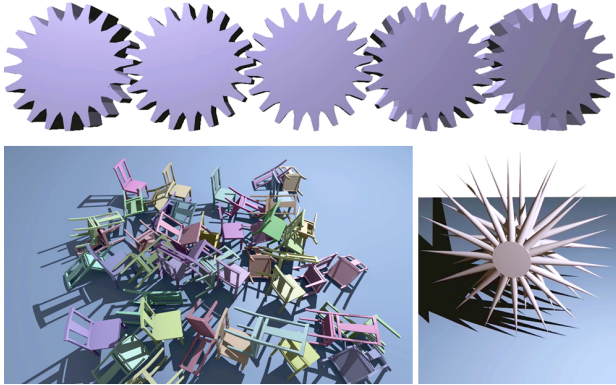


Figure 9: Other scenes simulated using our method— Spinning Gears (top), Pile of Chairs (bottom left), Spiky Wheel (bottom right)

4.6. Other examples

Fig. 9 shows several other scenes simulated using SDF objects, which all demonstrate smooth, stable motion at real-time frame rates (see supplementary video).

4.7. Discussion

Experiments demonstrate that Nelder-Mead optimization is well-suited for resolving non-convex interactions, often providing superior convergence and wall-clock performance compared to other methods (Tab. 1 and Tab. 5). To maintain real-time performance, we intentionally avoid dense sampling of the SDF overlapping regions, instead leveraging our framework as a manifold reduction technique. This efficiency is driven by the sorting mechanism in Alg. 1, which selects the top-performing samples at each time step. When combined with stochastic sampling, this yields an evolutionary refinement where the most persistent and geometrically significant contacts are retained, while new potential collision points are also explored. Maintaining a lean manifold also reduces constraint solver overhead.

Manifold stability is further improved by our hybrid caching and importance sampling strategy, which is designed to suppress chatter and oscillations. Combining these components into a unified framework results in a highly efficient, real-time capable methodology that consistently gives stable simulation behavior, even under complex contact scenarios.

5. Conclusion

This paper presents a robust collision framework for general SDF-SDF interactions, targeting real-time environments where main-

taining stable contact manifolds is paramount under limited computational budgets. The primary novelty of our work lies in the algorithmic coupling of temporal contact caching, curvature-based importance sampling, and collision point repulsion. Performance may further be fine-tuned by selection of the optimizer. Specifically, the Nelder-Mead method demonstrates efficiency in resolving interpenetrations between complex, non-convex geometries, while an Ellipsoid-based technique is a competitive alternative for strictly convex domains (Tab. 3). Empirical results confirm that the integration of temporal warm-starting with curvature-prioritized stochastic sampling provides a highly effective initialization strategy, ensuring rapid convergence and spatial diversity of contact points. Also, by prioritizing sparse stochastic samples near high-curvature features, the sampling strategy acts as a preconditioner for a local optimizer and generally leads to stable contact manifolds.

Surprisingly, the derivative-free optimizer often succeeds where gradient-based methods struggle. While first-order gradients are readily available in many SDF representations, they can become discontinuous or ill-conditioned, and relying on a simplex search rather than local derivatives improves convergence. Additionally, as highlighted in Sec. 4.5, distance and gradient evaluation query overhead can become a performance bottleneck.

Future work. An interesting avenue for future research is the development of an adaptive collision detection framework that dynamically changes the sampling strategy and optimization routine based on the local geometric properties of the participating distance fields. For instance, by leveraging analysis of the SDF's spectral or differential characteristics, e.g. based on the principal curvatures of both fields, the framework could automatically switch between robust simplex methods for complex, non-convex regions and a faster Ellipsoid or gradient-based solver for smoother, convex primitives. Furthermore, investigating learned importance sampling techniques, where a lightweight neural network predicts potential contact manifolds based on past temporal snapshots, could further reduce the stochastic search space. Such an adaptive methodology would optimize the balance between computational throughput and physical fidelity, allowing for even higher-density scenes with minimal loss in simulation stability.

Acknowledgements

We gratefully acknowledge the financial support of the Natural Sciences and Engineering Research Council of Canada (NSERC) grant RGPIN-2025-05982. We also thank Paul Kry and Kenny Erleben for providing valuable feedback and discussions.

References

- [AEF22] ANDREWS S., ERLEBEN K., FERGUSON Z.: Contact and friction simulation for computer graphics. In *ACM SIGGRAPH 2022 Courses* (New York, NY, USA, 2022), SIGGRAPH '22, Association for Computing Machinery. doi:10.1145/3532720.3535640. 2, 3, 6, 8
- [AP97] ANITESCU M., POTRA F. A.: Formulating dynamic multi-rigid-body contact problems with friction as solvable linear complementarity problems. *Nonlinear Dynamics* 14, 3 (1997), 231–247. doi:10.1023/A:1008292328909. 2
- [Bar84] BARR A. H.: Global and local deformations of solid primitives. *SIGGRAPH Comput. Graph.* 18, 3 (Jan. 1984), 21–30. doi:10.1145/964965.808573. 3
- [BBCW10] BERNHARDT A., BARTHE L., CANI M.-P., WYVILL B.: Implicit blending revisited. *Computer Graphics Forum* 29, 2 (2010), 367–375. doi:10.1111/j.1467-8659.2009.01606.x. 4
- [BMF05] BRIDSON R., MARINO S., FEDKIW R.: Simulation of clothing with folds and wrinkles. In *ACM SIGGRAPH 2005 Courses* (New York, NY, USA, 2005), SIGGRAPH '05, Association for Computing Machinery, p. 3–es. doi:10.1145/1198555.1198573. 2
- [BV25] BÁN R., VALASEK G.: Generalized lipschitz tracing of implicit surfaces. *Computer Graphics Forum* 44, 1 (2025), e15202. doi:10.1111/cgfm.15202. 2, 3
- [Cat06] CATTO E.: Iterative dynamics with temporal coherence. In *Game Developers Conference (GDC)* (2006). 2
- [CB21] COUMANS E., BAI Y.: Pybullet, a python module for physics simulation for robotics, games and machine learning. <http://pybullet.org>, 2016–2021. 3, 8
- [dB99] DEN BERGEN G. V.: A fast and robust gjk implementation for collision detection of convex objects. *Journal of Graphics Tools* 4, 2 (1999), 7–25. doi:10.1080/10867651.1999.10487502. 3
- [Eri04] ERICSON C.: *Real-time collision detection*. CRC Press, USA, 2004. doi:10.1201/b14581. 2, 7
- [Erl18] ERLEBEN K.: Methodology for assessing mesh-based contact point methods. *ACM Trans. Graph.* 37, 3 (July 2018). doi:10.1145/3096239. 2
- [FPRJ00] FRISKEN S. F., PERRY R. N., ROCKWOOD A. P., JONES T. R.: Adaptively sampled distance fields: a general representation of shape for computer graphics. In *Proc. of the 27th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 2000), SIGGRAPH '00, ACM Press/Addison-Wesley Publishing Co., p. 249–254. doi:10.1145/344779.344899. 2
- [GBC*13] GOURMEL O., BARTHE L., CANI M.-P., WYVILL B., BERNHARDT A., PAULIN M., GRASBERGER H.: A gradient-based implicit blend. *ACM Trans. Graph.* 32, 2 (Apr. 2013). doi:10.1145/2451236.2451238. 3
- [GBF03] GUENDELMAN E., BRIDSON R., FEDKIW R.: Nonconvex rigid bodies with stacking. *ACM Trans. Graph.* 22, 3 (2003), 871–878. doi:10.1145/882262.882358. 2, 5
- [GDY25] GILES C., DIAZ E., YUKSEL C.: Augmented vertex block descent. *ACM Trans. Graph.* 44, 4 (July 2025). doi:10.1145/3731195. 6
- [GJK88] GILBERT E., JOHNSON D., KEERTHI S.: A fast procedure for computing the distance between complex objects in three-dimensional space. *IEEE Journal on Robotics and Automation* 4, 2 (1988), 193–203. doi:10.1109/56.2083. 2, 3
- [KDBB17] KOSCHIER D., DEUL C., BRAND M., BENDER J.: An hp-adaptive discretization algorithm for signed distance field generation. *IEEE Trans. on Visualization and Computer Graphics* 23, 10 (2017), 2208–2221. doi:10.1109/TVCG.2017.2730202. 2, 6
- [LC91] LIN M., CANNY J.: A fast algorithm for incremental distance calculation. In *Proc. of the IEEE International Conference on Robotics and Automation* (1991), pp. 1008–1014 vol.2. doi:10.1109/ROBOT.1991.131723. 3
- [LM24] LÓPEZ-ADEVA FERNÁNDEZ-LAYOS P., MERCHANTE L. F.: Convex body collision detection using the signed distance function. *Computer-Aided Design* 170 (2024), 103685. doi:10.1016/j.cad.2024.103685. 2, 4, 6
- [LZW*24] LIU P., ZHANG Y., WANG H., YIP M. K., LIU E. S., JIN X.: Real-time collision detection between general sdfs. *Computer Aided Geometric Design* 111 (2024), 102305. doi:10.1016/j.cagd.2024.102305. 2, 3, 7, 8
- [MEM*20] MACKLIN M., ERLEBEN K., MÜLLER M., CHENTANEZ N., JESCHKE S., CORSE Z.: Local Optimization for Robust Signed Distance Field Collision. *Proc. of the ACM on Computer Graphics and Interactive Techniques* 3, 1 (04 2020), 1–17. doi:10.1145/3384538. 2, 3
- [MPT99] MCNEELY W. A., PUTERBAUGH K. D., TROY J. J.: Six degree-of-freedom haptic rendering using voxel sampling. In *Proc. of the 26th Annual Conference on Computer Graphics and Interactive Techniques* (USA, 1999), SIGGRAPH '99, ACM Press/Addison-Wesley Publishing Co., p. 401–408. doi:10.1145/311535.311600. 2
- [MPT05] MCNEELY W. A., PUTERBAUGH K. D., TROY J. J.: Advances in voxel-based 6-dof haptic rendering. In *ACM SIGGRAPH 2005 Courses* (New York, NY, USA, 2005), SIGGRAPH '05, Association for Computing Machinery, p. 50–es. doi:10.1145/1198555.1198606. 2
- [NE26] NIELSEN R. G., ERLEBEN K.: Alternating spatial-temporal optimization for continuous collision detection of signed distance fields. *Computer Graphics Forum* 45, 8 (2026). doi:10.1111/cgfm.70570. 2, 3
- [NM65] NELDER J. A., MEAD R.: A simplex method for function minimization. *The computer journal* 7, 4 (1965), 308–313. doi:10.1093/comjnl/7.4.308. 5
- [PASS95] PASKO A., ADZHIEV V., SOURIN A., SAVCHENKO V.: Function representation in geometric modeling: concepts, implementation and applications. *The Visual Computer* 11, 8 (1995), 429–446. doi:10.1007/BF02464333. 3
- [PFS*19] PARK J. J., FLORENCE P., STRAUB J., NEWCOMBE R., LOVEGROVE S.: Deepsdf: Learning continuous signed distance functions for shape representation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (Los Alamitos, CA, USA, June 2019), IEEE Computer Society, pp. 165–174. doi:10.1109/CVPR.2019.00025. 2
- [PGMAA25] PELLETIER-GUÉNETTE J., MERCIER-AUBIN A., ANDREWS S.: Real-time triangle-sdf continuous collision detection. *Proc. ACM Comput. Graph. Interact. Tech.* 8, 4 (Aug. 2025). doi:10.1145/3747862. 2, 3
- [Qui] QUILEZ I.: Distance functions. Accessed: 2026-08-17. URL: <http://iquilezles.org/articles/distfunctions/>. 9
- [Ric73] RICCI A.: A constructive geometry for computer graphics. *The Computer Journal* 16, 2 (1973), 157–160. doi:10.1093/comjnl/16.2.157. 3
- [RM51] ROBBINS H., MONRO S.: A stochastic approximation method. *The annals of mathematical statistics* (1951), 400–407. doi:10.1214/aoms/1177729586. 4
- [SJ22] SHARP N., JACOBSON A.: Spelunking the deep: guaranteed queries on general neural implicit surfaces via range analysis. *ACM Trans. Graph.* 41, 4 (July 2022). doi:10.1145/3528223.3530155. 2
- [TKH*05] TESCHNER M., KIMMERLE S., HEIDELBERGER B., ZACHMANN G., RAGHUPATHI L., FUHRMANN A., CANI M.-P., FAURE F., MAGNENAT-THALMANN N., STRASSER W., VOLINO P.: Collision detection for deformable objects. *Computer Graphics Forum* 24, 1 (2005), 61–81. doi:10.1111/j.1467-8659.2005.00829.x. 2
- [XB17] XU H., BARBIĆ J.: 6-dof haptic rendering using continuous collision detection between points and signed distance fields. *IEEE Trans. on Haptics* 10, 2 (2017), 151–161. doi:10.1109/TOH.2016.2613872. 2