

Temporally Coherent Hierarchical Distance Queries for Deformable Triangle Meshes

Jong-Hyun Kim

jonghyunkim@inha.ac.kr

Inha University

Research Article

Keywords: Signed distance field, Deformable triangle meshes, Temporal coherence, Triangle Influence Supersets, Persistent triangle seeds, Stackless traversal, GPU acceleration

Posted Date: October 29th, 2026

DOI: <https://doi.org/10.21203/rs.3.rs-11220293/v1>

License:   This work is licensed under a Creative Commons Attribution 4.0 International License.

[Read Full License](#)

Additional Declarations: No competing interests reported.

Temporally Coherent Hierarchical Distance Queries for Deformable Triangle Meshes

Jong-Hyun Kim

Received: date / Accepted: date

Abstract Signed-distance queries on deformable triangle meshes are fundamental operations that are repeatedly performed in a wide range of applications, including collision handling, surface tracking, physically based simulation, and geometric processing. However, in dynamic environments where a mesh continuously deforms over time, reconstructing a static hierarchy at every frame or updating all bounding volumes can incur substantial computational cost, while conventional distance-query acceleration structures do not fully exploit such temporal coherence. In this paper, we propose a temporally coherent hierarchical distance-query method for deformable triangle meshes. The proposed method maintains an anisotropic temporal bound for each hierarchy node to account for local deformation and selectively refreshes the stored reference bound only for nodes whose deformation magnitude exceeds a prescribed threshold, thereby preserving the tightness of the temporal bounds. In addition, nearest-triangle information from the previous frame is retained as a compact persistent seed and re-evaluated on the deformed geometry of the current frame to obtain a tight initial distance upper bound. A stackless traversal is then performed using conservatively expanded hierarchical bounds to compute the exact nearest triangle and signed distance. This temporal-coherence-based hierarchy reuse and selective refresh strategy effectively reduces the cost of repeated distance queries while preserving accuracy under continuous mesh deformation. Experimental

results show that, under identical distance-field resolutions and deformation conditions, the proposed method produces signed-distance values that agree with those of the baseline within numerical precision while improving distance-query performance by up to approximately $2.5\times$ in the evaluated tests. The proposed method can therefore be effectively applied to a broad range of real-time graphics and simulation applications that require repeated distance computations on dynamic and deformable geometry.

Keywords Signed distance field · Deformable triangle meshes · Temporal coherence · Triangle Influence Supersets · Persistent triangle seeds · Stackless traversal · GPU acceleration

1 Introduction

Efficiently computing the distance between a triangle mesh and a query point is a fundamental and important operation in computer graphics and geometry processing. In particular, signed distance functions (SDFs) are widely used in various applications, including collision detection, surface reconstruction, implicit representation, physically based simulation, deformable-object processing, and level-set-based operations. In these applications, a large number of distance queries are often repeatedly performed on the same geometry rather than evaluated only once; therefore, not only the accuracy of distance computation but also query speed and memory efficiency have a significant impact on the overall system performance.

Direct distance computation for a triangle mesh requires evaluating the distance from each query point to every triangle, and thus the computational cost increases rapidly with mesh complexity. To alleviate this

Jong-Hyun Kim
College of AI and Convergence (Dept. of Artificial Intelligence, Design Technology), Graduate School of Electrical and Computer Engineering at Inha University
Tel.: +82-32-860-9204
Fax: 0504-434-8210
E-mail: jonghyunkim@inha.ac.kr

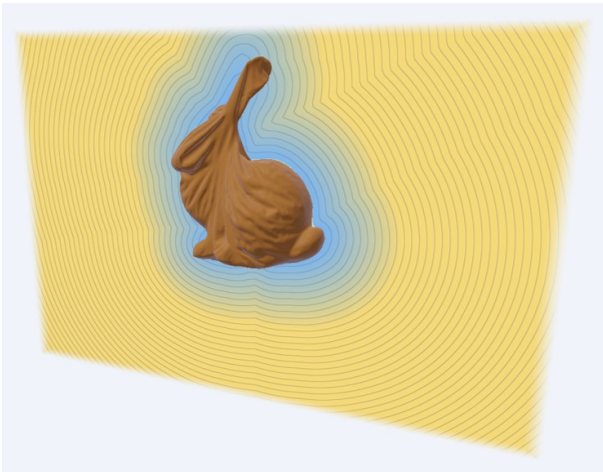


Fig. 1: Example result of signed distance field evaluation for a deformable triangle mesh. Despite continuous geometric deformation, the proposed method efficiently updates the hierarchical distance-query structure while preserving the spatial distance distribution around the current surface. Contour lines represent equal-distance regions of the resulting field.

cost, spatial acceleration structures such as BVHs, octrees, and spatial hashing have been widely used, and methods that conservatively construct a set of triangles that may influence a query region have also been proposed to reduce the search space. These methods can provide high efficiency for static meshes or in scenarios where a structure is constructed once and queried repeatedly.

However, for dynamic or deformable triangle meshes, vertex positions continuously change over time, making it difficult to assume that the bounding regions and triangle candidate information of a spatial hierarchy constructed in a previous frame remain valid in the current frame. The most direct approach is to reconstruct the entire hierarchy at every frame or to exactly refit the bounding region of every node, but this incurs repeated update costs in addition to the distance queries themselves. Conversely, conservatively expanding and reusing the structure from previous frames for a long period can progressively loosen the spatial bounds and increase unnecessary traversal.

Figure 1 shows a representative example of the deformable distance-query problem considered in this work. Even when the shape of a triangle mesh changes over time, the SDF must be consistently updated with respect to the current surface at each frame, and the contour lines represent regions at equal distances from the currently deformed surface. Therefore, in such environments, it is important to safely reuse spatial and dis-

tance information from previous frames according to the current deformation rather than independently applying a distance acceleration structure designed for static meshes at every frame.

Pujol and Chica [26] proposed Triangle Influence Supersets (TIS) to accelerate such repeated distance queries. TIS constructs a conservative superset of triangles that can potentially be the nearest triangle for each region of a spatial hierarchy, thereby reducing the number of triangles evaluated during a query while preserving exact distances. However, because the original formulation was designed for static geometry, continuously deforming surfaces such as that shown in Figure 1 introduce the additional problem of efficiently maintaining the hierarchy and candidate information over time.

In this work, we preserve the advantages of conservative candidate reduction and exact distance queries provided by TIS while exploiting the temporal coherence of deformable geometry in both hierarchy maintenance and query processing. For each hierarchy node, we maintain axis-wise expansion amounts required by the current deformation and selectively update the reference bound only when the expansion exceeds a prescribed threshold. In addition, the nearest triangle obtained from the previous SDF evaluation is stored as a compact persistent seed and re-evaluated on the current geometry of the next frame to initialize the query with an upper bound. A stackless traversal using conservative temporal bounds then reduces unnecessary node and triangle evaluations while preserving exactness.

The main contributions of this paper are as follows. First, we propose anisotropic temporal bounds that track node deformation independently along each axis for deformable triangle meshes, reducing unnecessary hierarchy overlap caused by global isotropic expansion. Second, we selectively refresh reference bounds according to the magnitude of local deformation, thereby preserving hierarchy tightness and reducing unnecessary global bound inflation and subsequent query traversal. Third, we use a compact persistent seed that re-evaluates the nearest triangle from the previous frame on the current geometry, allowing the next distance query to obtain a tight initial upper bound while preserving exactness. Finally, we combine these components with stackless hierarchical traversal to efficiently perform exact signed-distance queries on deformable meshes.

2 Problem Statement

We consider a triangle mesh that deforms over time,

$$\mathcal{M}^t = (\mathcal{V}^t, \mathcal{T})$$

where t denotes the frame index, $\mathcal{V}^t$ is the set of vertex positions at time t , and $\mathcal{T}$ denotes the triangle connectivity that remains fixed over time. Thus, this work considers deformable triangle meshes with fixed topology but continuously changing vertex positions.

The unsigned distance between a query point $\mathbf{x}$ and the mesh $\mathcal{M}^t$ is defined as

$$d(\mathbf{x}, \mathcal{M}^t) = \min_{T_i^t \in \mathcal{T}} d(\mathbf{x}, T_i^t)$$

where $d(\mathbf{x}, T_i^t)$ denotes the Euclidean distance between the query point and triangle T_i^t . The signed distance is defined as

$$\phi(\mathbf{x}, t) = s(\mathbf{x}, t)d(\mathbf{x}, \mathcal{M}^t), \quad s(\mathbf{x}, t) \in \{-1, +1\}$$

TIS computes exact distances without directly searching the complete triangle set by maintaining a conservative candidate superset for each spatial region that contains the actual nearest triangle [26]. For deformable meshes, however, changes in vertex positions require the spatial bounds and candidate relations from previous frames to remain valid with respect to the current geometry. Updating the entire hierarchy and all related information at every frame increases the maintenance cost, whereas conservatively expanding and reusing the previous structure for an extended period can loosen spatial bounds and increase query cost.

Therefore, the goal of this work is to reduce the overall cost of hierarchy maintenance and distance queries under temporally coherent deformation while preserving the conservative candidate pruning and exact-query properties of TIS. Conceptually, this objective is expressed as

$$\begin{aligned} \min \quad & C_{\text{maintenance}} + C_{\text{query}}, \\ \text{s.t.} \quad & \phi_{\text{ours}}(\mathbf{x}, t) = \phi_{\text{exact}}(\mathbf{x}, t), \quad \forall \mathbf{x}, t. \end{aligned}$$

The specific temporal reuse and coherent query strategies are described in Section 4.

3 Related Work

3.1 Signed Distance Fields and Exact Distance Queries

Signed Distance Fields (SDFs) jointly represent the distance from a surface and inside–outside information,

and have therefore been widely used in various computer graphics applications, including geometry processing, collision detection, physically based simulation, surface reconstruction, and rendering [1, 3]. Frisken et al. [1] proposed Adaptively Sampled Distance Fields, which adaptively subdivide spatial regions where additional resolution is required and provide a more memory-efficient distance-field representation than uniform grids. Jones et al. [3] subsequently provided a broad survey of the construction, representation, and applications of 3D distance fields and systematically analyzed the trade-offs among accuracy, computational cost, and memory usage.

Computing exact signed distances from triangle meshes requires robustly determining not only the nearest geometric feature but also whether a query point lies inside or outside the surface. Bærentzen and Aanaes [2] proposed an angle-weighted pseudonormal method that provides consistent sign computation when the closest feature is a face, edge, or vertex. Xu and Barbič [9] proposed a method for robustly defining SDFs for polygon soups containing self-intersections, holes, and non-manifold configurations. The Generalized Winding Number of Jacobson et al. [7] and the Fast Winding Number of Barill et al. [16] provide robust inside–outside classification for complex or incomplete triangle meshes and can also be used to determine SDF signs.

Hierarchical and adaptive representations have also been investigated to reduce the memory cost of directly storing high-resolution distance fields. Koschier et al. [12] proposed Hierarchical *hp*-Adaptive SDFs, combining octree subdivision with polynomial degree adaptation to improve both accuracy and memory efficiency. Kim [24] proposed a GPU framework for computing adaptive and continuous SDFs from triangle meshes and applied it to distance queries, ray tracing, and collision handling. Although these approaches improve efficiency through spatial adaptivity, they address the reuse of information from previous frames in continuously deforming meshes only to a limited extent.

3.2 Triangle Influence Supersets

Pujol and Chica [26] proposed Triangle Influence Supersets (TIS) to accelerate exact SDF queries. Their method hierarchically subdivides space using an octree and stores at each node only a conservative superset of triangles that can potentially become the nearest triangle. Each child node inherits the candidate set of its parent and conservatively removes triangles that cannot affect the distance computation. As a result, the query stage only needs to inspect the candidates stored

in the corresponding leaf node rather than the complete mesh, while exact distances are still preserved.

A key characteristic of TIS is that it performs conservative rejection without explicitly constructing the influence region of each triangle. Pujol and Chica [26] define a convex influence superset based on the distances between node corners and a reference triangle and remove unnecessary triangles using GJK- and Frank-Wolfe-based intersection tests. Unlike approximate SDF methods, this approach reduces the number of triangles examined during a query without approximating the distance field itself.

The same authors subsequently extended their work with approximation-based methods for faster SDF evaluation. Pujol and Chica [27] proposed an adaptive piecewise SDF representation using trilinear and tricubic interpolation, subdividing nodes according to a prescribed approximation error. Pujol and Chica [29] further reduced the number of sphere-marching iterations and improved rendering and memory efficiency by performing analytic ray-surface intersection within the narrow band of a piecewise SDF.

These studies significantly improve the efficiency of exact or approximate SDF evaluation for static triangle meshes. However, because the candidate hierarchy of TIS is determined by the input geometry, the validity of spatial bounds and candidate relations changes over time when the mesh continuously deforms. Therefore, efficiently handling temporally coherent deformation while preserving the advantages of the conservative candidate reduction provided by TIS is directly related to the problem addressed in this work.

3.3 Hierarchical Acceleration for Dynamic Geometry

Hierarchical acceleration structures such as BVHs, octrees, and kd-trees have long been used to reduce the search cost for large triangle geometry. In particular, for dynamic or deformable geometry, refitting the bounding volumes while preserving the existing hierarchy topology is widely used instead of rebuilding the hierarchy from scratch at every frame. RT-DEFORM by Lauterbach et al. [4] dynamically updates a BVH for deformable scenes or rebuilds it when necessary, using hierarchy-quality degradation to determine when rebuilding should occur. Wald et al. [5] likewise showed that the reconstruction cost can be reduced for deformable scenes by maintaining BVH topology and refitting bounding volumes between frames.

Bittner and Meister [11] introduced T-SAH, which considers the geometry distribution over an entire animation to enable efficient reuse of a single BVH topology

across multiple frames. This approach is closely related to our work in that temporal information is incorporated into hierarchy construction; however, its objective is BVH quality optimization for ray traversal rather than maintaining an exact distance candidate set.

Research has also focused on accelerating hierarchy construction itself on GPUs. Karras [6] proposed a parallel construction method for BVHs, octrees, and kd-trees using Morton ordering, while Gu et al. [8] generated high-quality BVHs in parallel through Approximate Agglomerative Clustering. Apetrei [10] proposed single-pass bottom-up LBVH construction based on sorted Morton codes. Hendrich et al. [13] improved the trade-off between build time and traversal quality using Progressive Hierarchical Refinement.

Meister and Bittner [14] introduced Parallel Locally-Ordered Clustering (PLOC) for massively parallel BVH construction on GPUs and subsequently improved hierarchy quality through Parallel Reinsertion [15]. More recently, Benthin et al. [25] improved construction performance with PLOC++ by simplifying and combining several stages of the original PLOC method, while H-PLOC [30] further reduced GPU BVH construction time through hierarchical clustering. Meister et al. [20] provide a comprehensive review of BVH construction, traversal, and optimization techniques.

Most of these methods, however, target ray intersection or visibility queries and therefore differ in problem formulation from TIS, which maintains conservative influence sets required for exact nearest-triangle queries in each spatial region. In particular, the problem of conservatively reusing existing influence information while performing selective updates for deformable meshes has received relatively limited attention.

3.4 Neural Signed Distance Representations

Recent studies have also actively investigated representing SDFs using neural implicit functions rather than explicit spatial structures. DeepSDF [17] learns a continuous SDF using a neural network for shape representation and completion. SAL [18] introduced a sign-agnostic formulation that enables implicit-surface learning from raw geometric data without ground-truth signed distances, while Implicit Geometric Regularization [19] learns implicit surfaces from point clouds using Eikonal regularization.

Neural-Pull [21] learns an SDF from point clouds by pulling query points toward the surface using the predicted SDF and its gradient. NeuS [22] and VolSDF [23] combine SDF representations with neural volume rendering to reconstruct high-quality surfaces from multi-view images. Neuralangelo [28] combines a multi-resolution

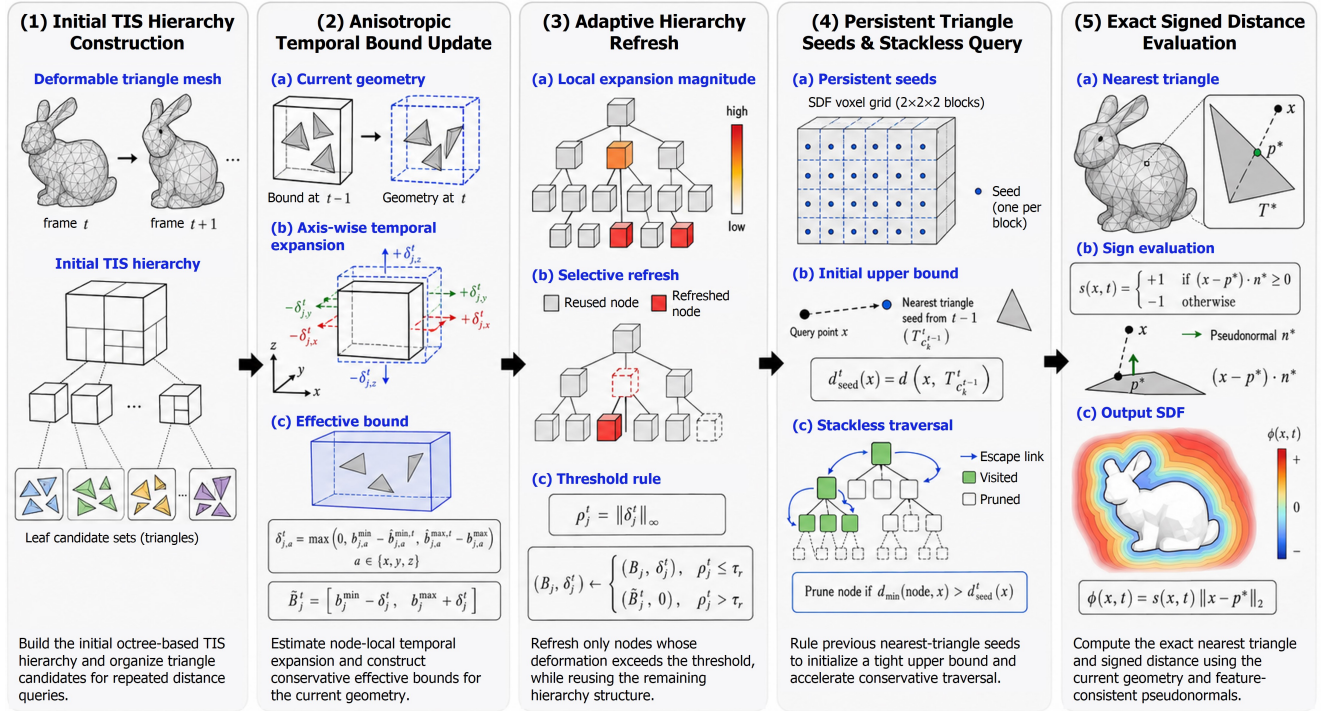


Fig. 2: Overview of the proposed temporally coherent distance-query framework for deformable triangle meshes. Starting from an initial TIS-based hierarchy, the method updates anisotropic temporal bounds, selectively refreshes locally deformed nodes, reuses persistent triangle seeds for stackless traversal, and finally computes the exact signed distance using the current geometry and feature-consistent pseudonormals.

hash representation with a neural SDF to enable detailed large-scale surface reconstruction.

Although these neural implicit methods provide powerful compact continuous representations and reconstruction capabilities, they require learning or optimization and are not designed to directly guarantee exact nearest-distance queries on the original triangle mesh. In contrast, our method does not approximate or learn the SDF of the geometry itself; instead, it preserves the conservative candidate reduction and exact-query properties of Triangle Influence Supersets while improving hierarchy maintenance and repeated distance evaluation under temporally coherent deformation.

4 Algorithm

We extend the conservative distance-query structure of Triangle Influence Supersets (TIS) to deformable triangle meshes. The proposed method consists of initial hierarchy construction, anisotropic temporal-bound updates, adaptive hierarchy refresh, persistent triangle seeds, and exact signed-distance evaluation.

The overall computational pipeline of the proposed method is summarized in Figure 2. At the initial frame,

we construct a TIS-based hierarchy and triangle candidate organization. At each subsequent frame, anisotropic temporal bounds are computed for each node with respect to the current geometry, and only the stored reference bounds of nodes whose local expansion magnitude exceeds the threshold are selectively refreshed. Meanwhile, nearest-triangle information from the previous frame is retained as a persistent seed and re-evaluated on the current geometry to initialize the upper bound for the distance query. Finally, conservative pruning is performed through stackless hierarchical traversal, and the exact signed distance is computed using the current triangle geometry and feature-consistent pseudonormals.

4.1 Initial TIS Hierarchy

For the initial mesh $\mathcal{M}^0$ defined in the Problem Statement, we construct an octree-based hierarchy, denoted by $\mathcal{H} = \{R_0, \dots, R_{N_H-1}\}$. We use the core TIS concept of spatially conservative triangle organization as the starting point for the initial hierarchy construction. The root contains the complete triangle set, and spatial subdivision hierarchically organizes the triangles so that irrelevant regions can be rapidly excluded

during query processing. In a deformable setting, however, using the influence relation of the initial frame as a permanent rejection rule can become invalid because the nearest triangle may change in subsequent frames. Therefore, the exact query of the proposed method performs pruning based on node bounds that contain the current geometry. In other words, temporal reuse is applied to the hierarchy topology and triangle ordering rather than as an approximation that removes triangles. Accordingly, the initial candidate organization follows the conservative filtering principle of TIS, whereas the validity of pruning after deformation is guaranteed by temporal node bounds that contain the current geometry rather than by the initial influence relation itself.

Each node R_j has an axis-aligned bounding box $B_j = [\mathbf{b}_j^{\min}, \mathbf{b}_j^{\max}]$. Subdivision terminates when the number of triangles contained in a leaf is no greater than N_{leaf} or when the node depth d_j reaches the maximum depth D_{max} . We denote by $\mathcal{C}_j$ the persistent triangle candidate set retained by leaf R_j . After deformation, we do not assume that the initial influence relation remains valid, nor do we use $\mathcal{C}_j$ to further remove triangles. Instead, $\mathcal{C}_j$ is retained as a persistent candidate organization, while the validity of exact query-level pruning is determined by the effective temporal bounds that contain the current geometry. After the initial construction, the parent-child topology of the hierarchy and the triangle ordering are kept fixed throughout deformation, and only the current triangle geometry and effective node bounds are updated. Consequently, topology reconstruction is not repeated at every SDF update.

To improve memory locality for GPU queries, triangle geometry is stored contiguously according to leaf traversal order. In addition, an escape link e_j , which indicates the next node to visit when the corresponding subtree is skipped, is computed for each node during initialization. Because the hierarchy topology remains fixed, these links can be reused for all subsequent frames. At each SDF update, triangle geometry and face orientation are recomputed from the current vertex positions, and temporal information is not used to approximate the current geometry itself.

4.2 Anisotropic Temporal Bounds

Using the bounds of a static hierarchy directly for deformable geometry may allow triangles to move outside their original node regions. One possible solution is to expand all nodes uniformly using the maximum displacement over the entire mesh. However, such global isotropic expansion unnecessarily increases the bounds along unaffected axes even when deformation occurs

predominantly in a specific direction, thereby increasing hierarchy overlap.

At time t , let the exact bounding box formed by the current triangles belonging to node R_j be

$$\hat{B}_j^t = [\hat{\mathbf{b}}_j^{\min,t}, \hat{\mathbf{b}}_j^{\max,t}] \quad (1)$$

For reference bound B_j , the temporal expansion required along axis $a \in \{x, y, z\}$ is defined as

$$\delta_{j,a}^t = \max\left(0, b_{j,a}^{\min} - \hat{b}_{j,a}^{\min,t}, \hat{b}_{j,a}^{\max,t} - b_{j,a}^{\max}\right) \quad (2)$$

Thus, the anisotropic temporal pad is $\boldsymbol{\delta}_j^t = (\delta_{j,x}^t, \delta_{j,y}^t, \delta_{j,z}^t)$, and the effective bound used for the actual query is

$$\tilde{B}_j^t = [\mathbf{b}_j^{\min} - \boldsymbol{\delta}_j^t, \mathbf{b}_j^{\max} + \boldsymbol{\delta}_j^t]. \quad (3)$$

Equation (3) conservatively contains the current geometry while maintaining independent expansion along each axis. For example, when deformation is dominated by the x direction, $\delta_{j,x}^t \gg \delta_{j,y}^t, \delta_{j,z}^t$ may hold, thereby suppressing unnecessary bounding-volume growth along the y and z directions.

For a leaf node, $\tilde{B}_j^t$ is directly computed from its current triangle geometry. For an internal node, the child effective bounds are combined in a bottom-up manner as

$$\hat{B}_j^t = \bigcup_{c \in \text{child}(j)} \tilde{B}_c^t \quad (4)$$

Repeating this procedure from the leaves toward the root preserves the conservative property throughout the hierarchy.

4.3 Adaptive Hierarchy Refresh

When a reference bound is retained for an extended period, the temporal pad required to contain the current deformation can increase, causing the effective bound to become progressively looser. Conversely, fully refitting exact bounds at every frame reduces the benefit of temporal reuse. To balance these effects, we define the local expansion magnitude of node R_j as

$$\rho_j^t = \|\boldsymbol{\delta}_j^t\|_{\infty} = \max(\delta_{j,x}^t, \delta_{j,y}^t, \delta_{j,z}^t) \quad (5)$$

and compare it with the refresh threshold τ_r . The node update can be expressed compactly as

$$(B_j, \boldsymbol{\delta}_j^t) \leftarrow \begin{cases} (B_j, \boldsymbol{\delta}_j^t), & \rho_j^t \leq \tau_r, \\ (\hat{B}_j^t, \mathbf{0}), & \rho_j^t > \tau_r. \end{cases} \quad (6)$$

Thus, nodes with small deformation retain their previous reference bounds and apply only the temporal pad,

whereas nodes whose spatial quality has sufficiently degraded are selectively refreshed using their current bounds. This process does not modify the hierarchy topology or candidate organization.

The refresh threshold is determined by considering both the SDF sampling scale and the temporal margin. Let the voxel spacings be h_x , h_y , and h_z , with $h_{\min} = \min(h_x, h_y, h_z)$. We define

$$\tau_r = \max(\tau_0, \min(\alpha h_{\min}, \beta \tau_m)), \quad (7)$$

where τ_m is the allowable temporal margin and τ_0 is a minimum threshold that prevents excessively frequent refreshes. In our implementation, we use $\alpha = 0.40$, $\beta = 0.55$, $\tau_m = 0.030$, and $\tau_0 = 0.0025$ as default values, and the GPU implementation additionally applies a small voxel-scale-dependent lower-bound clamp for numerical stability. A small τ_r increases the refresh frequency, whereas a large τ_r increases traversal cost due to looser bounds. Thus, this threshold controls the trade-off between hierarchy tightness and temporal reuse. Under localized deformation, only nodes whose spatial expansion exceeds the threshold refresh their reference bounds, while the remaining nodes use the combination of the existing bounds and temporal pads.

4.4 Persistent Triangle Seeds

Temporal coherence is exploited not only in hierarchy maintenance but also in the query itself. When geometry changes continuously between adjacent frames, the nearest triangle in the same spatial region is likely to remain unchanged for many queries. We therefore partition the SDF query grid into $2 \times 2 \times 2$ voxel blocks Q_k and store one persistent triangle seed c_k^{t-1} , obtained from the previous SDF evaluation, for each block.

The distance value from the previous frame is not reused. Instead, only the triangle ID is retained, and the corresponding triangle is re-evaluated on the current geometry at frame t . The initial upper bound for a query point $\mathbf{x} \in Q_k$ is

$$d_{\text{seed}}^t(\mathbf{x}) = d(\mathbf{x}, T_{c_k^{t-1}}^t) \quad (8)$$

and, by definition of the nearest distance,

$$d(\mathbf{x}, \mathcal{M}^t) \leq d_{\text{seed}}^t(\mathbf{x}) \quad (9)$$

always holds. Therefore, even if the previous seed is no longer the current nearest triangle, exactness is unaffected; only the tightness of the initial upper bound changes.

For each $2 \times 2 \times 2$ block, one predefined representative voxel stores the current nearest triangle obtained

after query completion as the seed for the next SDF evaluation. The number of seeds required for an $N_x \times N_y \times N_z$ grid is

$$N_s = \left\lceil \frac{N_x}{2} \right\rceil \left\lceil \frac{N_y}{2} \right\rceil \left\lceil \frac{N_z}{2} \right\rceil$$

For example, an 80^3 grid requires only $40^3 = 64,000$ 32-bit triangle IDs, corresponding to approximately 0.244 MiB of additional memory for maintaining temporal query coherence.

4.5 Exact Stackless Distance Query

For each query point $\mathbf{x}$, the persistent seed is first evaluated on the current geometry to initialize d_{best}^2 . For the first evaluation, where no seed is available, the query starts with $d_{\text{best}}^2 = \infty$. The squared lower bound between the query point and an AABB $B = [\mathbf{b}^{\min}, \mathbf{b}^{\max}]$ is defined as

$$d_{\text{box}}^2(\mathbf{x}, B) = \sum_{a \in \{x, y, z\}} [\max(b_a^{\min} - x_a, 0, x_a - b_a^{\max})]^2 \quad (10)$$

When visiting node R_j , if

$$d_{\text{box}}^2(\mathbf{x}, \tilde{B}_j^t) \geq d_{\text{best}}^2 \quad (11)$$

the entire subtree can be discarded because it cannot contain a triangle closer than the current solution. If a leaf is not pruned, its candidate triangles are traversed, and the exact point-triangle distance is evaluated only when the triangle AABB lower bound is smaller than d_{best}^2 . Let $\text{CP}(\mathbf{x}, T)$ denote the closest point on triangle T . Then

$$d(\mathbf{x}, T) = \|\mathbf{x} - \text{CP}(\mathbf{x}, T)\|_2 \quad (12)$$

and whenever a smaller distance is found, d_{best}^2 and the nearest triangle are immediately updated.

On the GPU, we perform stackless preorder traversal using the escape links stored in the initial hierarchy instead of an explicit DFS stack. When a node is pruned according to Equation (11), traversal immediately advances to its escape link; when an internal node is retained, its first valid child is visited. This strategy performs conservative branch-and-bound search while reducing the local-memory pressure associated with a per-thread traversal stack.

The complete procedure is summarized in Algorithm 1.

Algorithm 1 Temporally Coherent Hierarchical Distance Query

Require: Current mesh $\mathcal{M}^t$, hierarchy $\mathcal{H}$, previous seeds $\{c_k^{t-1}\}$

Ensure: Exact signed distances $\phi(\mathbf{x}, t)$

```

1: Update triangle geometry using current vertex positions
2: for all leaf nodes  $R_j$  do
3:   Compute  $\tilde{B}_j^t$  and  $\delta_j^t$ 
4:   if  $\|\delta_j^t\|_\infty > \tau_r$  then
5:      $B_j \leftarrow \tilde{B}_j^t$ ;  $\delta_j^t \leftarrow \mathbf{0}$ 
6:   end if
7: end for
8: for hierarchy levels from bottom to top do
9:   Compute parent effective bounds using Eq. (4)
10:  Apply the refresh rule in Eq. (6)
11: end for
12: for all query points  $\mathbf{x} \in Q_k$  do
13:   if persistent seed  $c_k^{t-1}$  exists then
14:      $d_{\text{best}}^2 \leftarrow d^2(\mathbf{x}, T_{c_k^{t-1}}^t)$ 
15:   else
16:      $d_{\text{best}}^2 \leftarrow \infty$ 
17:   end if
18:    $j \leftarrow \text{root node}$ 
19:   while  $j \neq -1$  do
20:     if  $d_{\text{box}}^2(\mathbf{x}, \tilde{B}_j^t) \geq d_{\text{best}}^2$  then
21:        $j \leftarrow e_j$ 
22:     else if  $R_j$  is a leaf then
23:       for all  $T \in \mathcal{C}_j$  do
24:         if  $d_{\text{box}}^2(\mathbf{x}, B_T) < d_{\text{best}}^2$  then
25:           Evaluate exact  $d^2(\mathbf{x}, T)$ 
26:           Update nearest triangle  $T^*$  and  $d_{\text{best}}^2$ 
27:         end if
28:       end for
29:        $j \leftarrow e_j$ 
30:     else
31:        $j \leftarrow \text{first valid child of } R_j$ 
32:     end if
33:   end while
34:   Compute sign from the current closest-feature pseudonormal
35:   if  $\mathbf{x}$  is the representative voxel of  $Q_k$  then
36:      $c_k^t \leftarrow T^*$ 
37:   end if
38: end for

```

4.6 Conservativeness and Signed Distance Evaluation

Hierarchy traversal determines the nearest triangle T^* and closest point $\mathbf{p}^*$. If the closest point lies in the triangle interior, the current face normal is used. If the closest feature is an edge, an edge pseudonormal formed by combining the normals of the adjacent faces is used; if it is a vertex, the angle-weighted pseudonormal [2], which accounts for the angular contributions of incident faces, is used. Let $\mathbf{n}^*$ denote the selected normal. Then

$$s(\mathbf{x}, t) = \begin{cases} +1, & (\mathbf{x} - \mathbf{p}^*) \cdot \mathbf{n}^* \geq 0, \\ -1, & \text{otherwise,} \end{cases} \quad (13)$$

and the final signed distance is

$$\phi(\mathbf{x}, t) = s(\mathbf{x}, t) \|\mathbf{x} - \mathbf{p}^*\|_2. \quad (14)$$

Because pseudonormals are recomputed from the current triangle geometry at every SDF update, the surface orientation after deformation is directly reflected in sign evaluation.

Temporal information in the proposed method does not produce approximate distances. Because $\tilde{B}_j^t$ in Equation (3) contains the current geometry of node R_j , for any triangle T_i^t belonging to R_j ,

$$d(\mathbf{x}, \tilde{B}_j^t) \leq d(\mathbf{x}, T_i^t) \quad (15)$$

holds. At the same time, the persistent seed in Equation (9) provides an upper bound obtained by explicitly re-evaluating the current triangle. Therefore, pruning a node that satisfies Equation (11) cannot discard a solution closer than the current best distance. Temporal reuse and persistent seeds affect only the traversal workload and do not change the final nearest triangle or unsigned distance. In addition, because the same current-geometry pseudonormal sign convention as the baseline is used, for identical inputs and sign conventions,

$$\phi_{\text{ours}}(\mathbf{x}, t) = \phi_{\text{exact}}(\mathbf{x}, t) \quad (16)$$

is preserved.

4.7 GPU Optimization

The proposed method uses data organization and traversal strategies designed for GPU execution characteristics to efficiently process large-scale SDF queries. Although temporal coherence itself is a device-independent algorithmic concept, reducing irregular memory access and thread-local storage during hierarchy traversal is important for achieving high GPU performance.

First, triangle geometry is stored contiguously according to the leaf traversal order of the hierarchy. A conventional indirect triangle-index lookup can introduce additional global-memory indirection when a query thread accesses triangle geometry. In our implementation, triangles are reordered according to leaf order after the initial hierarchy construction, allowing the candidate triangles of each leaf to be accessed directly from a contiguous memory region. This organization improves memory locality for query threads traversing nearby hierarchy regions.

In addition, each node stores an escape link to eliminate the thread-local stack typically required for conventional depth-first hierarchy traversal. Because the

hierarchy topology remains fixed during deformation, escape links are computed only once during initialization. When a query thread prunes node R_j , it immediately advances to the precomputed escape node e_j without visiting the children. Consequently, no per-thread local stack is required to maintain traversal state, reducing local-memory pressure and traversal overhead on the GPU.

Persistent triangle seeds are stored per $2 \times 2 \times 2$ voxel block rather than per voxel. Each block maintains only one 32-bit triangle index, limiting the additional memory required for query state while still exploiting temporal coherence. This compact representation allows seed information to be maintained efficiently even for large SDF grids.

Finally, anisotropic temporal-bound updates and adaptive hierarchy refresh are parallelized using CUDA kernels at the node and hierarchy-level granularity. For leaf nodes, local bounds and temporal expansions are computed from the current triangle geometry, while internal nodes combine child effective bounds in a bottom-up manner. Because each SDF sample can be processed as an independent query, large grids expose substantial data parallelism.

These GPU-oriented optimizations reduce the execution cost of exact distance evaluation by adapting memory access, traversal state, and hierarchy updates to GPU execution without changing the temporal-coherence mechanism itself.

4.8 Computational Characteristics

Let N_T , N_H , and N_Q denote the numbers of triangles, hierarchy nodes, and query points, respectively. Updating the current triangle geometry is performed once for every triangle and therefore requires $O(N_T)$ work. In addition, evaluating the current bounds of leaf nodes and the effective bounds of internal nodes gives an asymptotic temporal hierarchy maintenance cost of $O(N_T + N_H)$. Thus, the primary advantage of the proposed method is not a reduction in the asymptotic complexity of hierarchy maintenance. Instead, it avoids reconstructing the hierarchy topology and triangle ordering and prevents global bound inflation through node-local deformation information, thereby reducing the number of nodes and triangles visited during queries. Selective refresh further helps preserve local bound tightness over time by limiting stored-reference-bound updates to nodes that require them.

Let $N_V(\mathbf{x})$ denote the number of nodes visited by query point $\mathbf{x}$ and $N_C(\mathbf{x})$ the number of triangles for which exact point-triangle evaluation is performed. The

Parameter	Symbol	Value
Maximum hierarchy depth	$D_{\max}$	7
Maximum triangles per leaf	N_{leaf}	48
Refresh coefficient	α	0.40
Temporal-margin coefficient	β	0.55
Temporal margin	τ_m	0.030
Minimum refresh threshold	τ_0	0.0025
Persistent seed block size	B_s	$2 \times 2 \times 2$

Table 1: Default algorithmic parameters of the proposed temporally coherent distance-query framework.

actual query workload can then be expressed as

$$C_{\text{query}} \propto \sum_{\mathbf{x} \in \mathcal{K}} [N_V(\mathbf{x}) + N_C(\mathbf{x})]. \quad (17)$$

Anisotropic temporal bounds keep the hierarchy lower bounds tight and suppress growth in N_V , while persistent triangle seeds provide a small d_{best} at query initialization and strengthen subtree pruning. These two mechanisms improve lower-bound tightness and upper-bound initialization, respectively, and therefore operate complementarily. In the worst case, every node and triangle may be visited, so the worst-case complexity of exact nearest-distance queries remains $O(N_Q N_T)$. The proposed method instead aims to reduce the average numbers of visited nodes and exact triangle evaluations under temporally coherent deformation.

The default algorithmic parameters are summarized in Table 1. $D_{\max}$ and N_{leaf} determine the balance between hierarchy depth and leaf triangle density, while α , β , τ_m , and τ_0 control the spatial tolerance of adaptive refresh. B_s balances query coherence against persistent memory usage. Implementation and experimental settings, such as SDF resolution, update interval, and CUDA block size, are described separately in the experimental setup.

5 Experimental Results

5.1 Experimental Setup

To evaluate the performance and accuracy of the proposed method, we used the Stanford Bunny, Dragon, Armadillo, and Happy Buddha models, which exhibit different geometric complexities and deformation patterns. To evaluate query behavior when multiple deformable objects are present simultaneously, we also constructed a Stanford Party scene containing several different Stanford models and an Armadillo Party scene containing multiple Armadillo instances. In all scenes, triangle connectivity was fixed and only vertex positions varied over time, resulting in deformable animations with fixed topology.

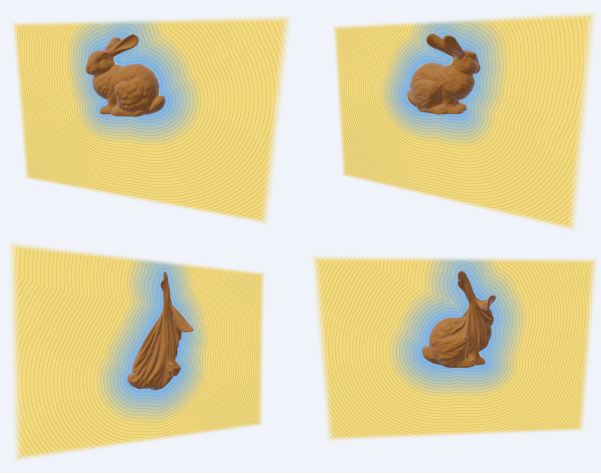


Fig. 3: Signed distance fields for the Stanford Bunny under different deforming configurations. The distance contours are evaluated with respect to the current triangle geometry at each frame.

The experiments were conducted on an NVIDIA GeForce GTX 1080 Ti GPU using CUDA 11.0. All methods used the same mesh deformation and query positions as input, and the SDF was evaluated on an $80 \times 80 \times 80$ regular grid. Therefore, each SDF evaluation consisted of a total of 512,000 distance queries. The time required for mesh simulation and deformation generation at each frame was excluded from the distance-query comparison, and hierarchy maintenance and SDF evaluation were measured after the current geometry had been provided.

The maximum hierarchy depth was set to $D_{\max} = 7$, and the maximum number of triangles per leaf was set to $N_{\text{leaf}} = 48$. For adaptive refresh, we used $\alpha = 0.40$, $\beta = 0.55$, $\tau_m = 0.030$, and $\tau_0 = 0.0025$, while the persistent triangle seed stored one triangle ID per $2 \times 2 \times 2$ voxel block. These algorithmic parameters were applied consistently to all experiments using the default settings listed in Table 1.

For performance comparison, the baseline was defined as a full-refit method that updates hierarchy bounds according to the current triangle geometry at every SDF update and performs exact hierarchical distance queries without temporal seeds. The baseline and the proposed method use the same current triangle geometry, exact point-triangle distance evaluation, and closest-feature pseudonormal sign convention. Therefore, the observed differences between the two methods arise from hierarchy updates and query traversal rather than from differences in the distance definition.

All timings were measured using CUDA events to record GPU kernel execution time, and synchronization

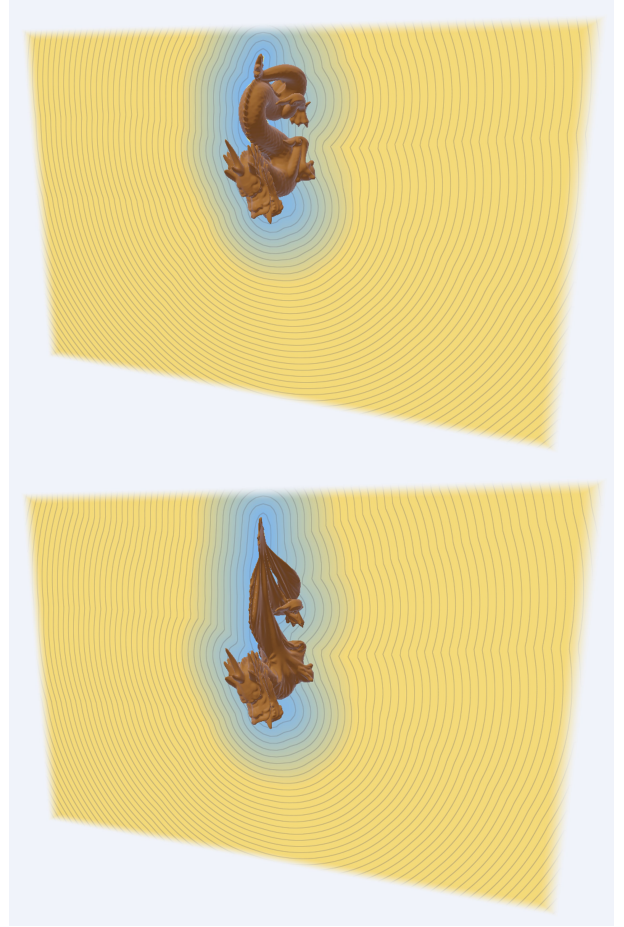


Fig. 4: Signed distance fields for the Stanford Dragon under large non-rigid deformation. Complex surface features and elongated structures remain consistently represented by the distance contours.

was performed after kernel completion for each measurement. Initial model loading, hierarchy construction, and the first warm-up evaluation were excluded from repeated SDF evaluation timings. For each target geometry state, the baseline and the proposed method were each executed 10 times, and the average of these repeated measurements was used as the performance result.

5.2 Performance Comparison

We first evaluated whether the proposed method could stably compute SDFs under different deformation patterns and geometric complexities. Signed distance fields computed at different deformation states for four Stanford models are shown in Figures 3–6.

For the Stanford Bunny and Armadillo, articulated deformation continuously changes the silhouette and lo-

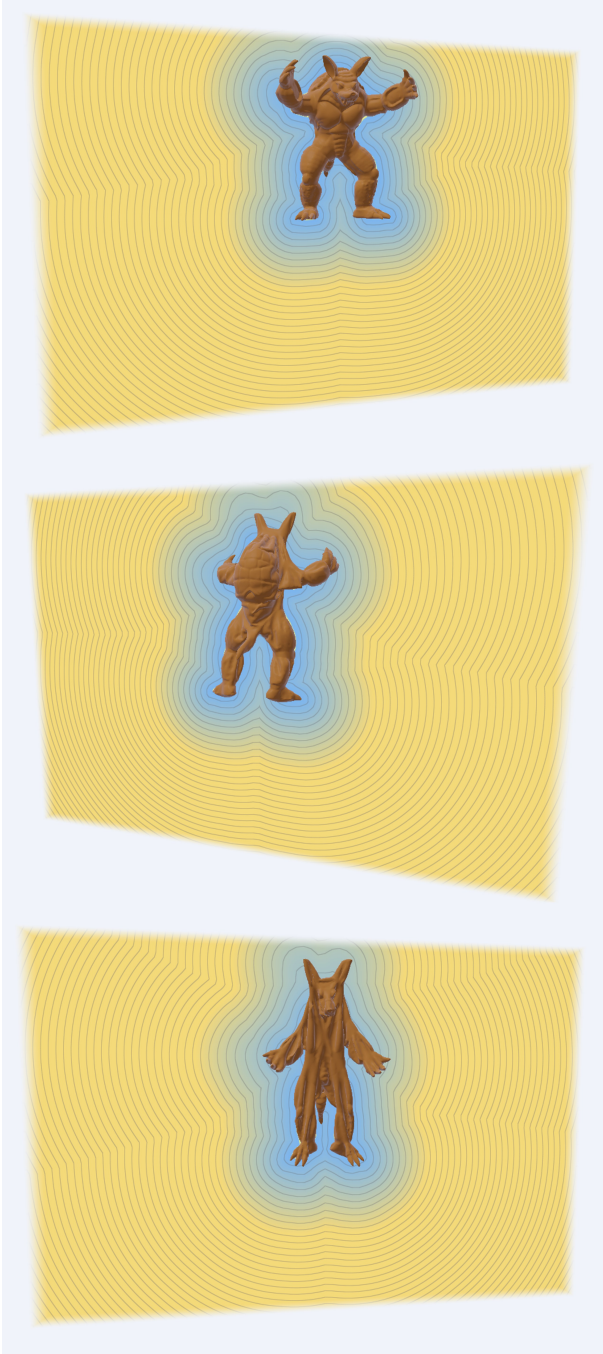


Fig. 5: Signed distance fields for the Stanford Armadillo in several deforming poses. Articulated body motion is continuously reflected in the evaluated distance field.

cal surface orientation, while the Dragon and Happy Buddha exhibit more complex surface details and larger spatial extents that affect hierarchical queries. In all cases, the distance contours are continuously updated according to changes in the current surface. This is because temporal information is used only to accelerate

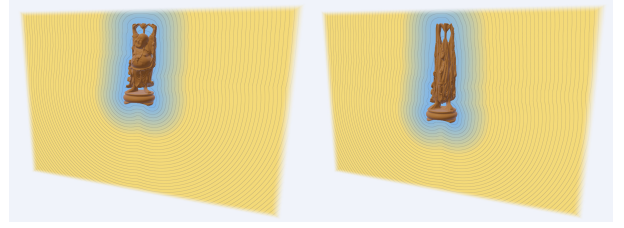


Fig. 6: Signed distance fields for the Stanford Happy Buddha under deforming configurations. The proposed method consistently evaluates the distance distribution around geometrically detailed surfaces.

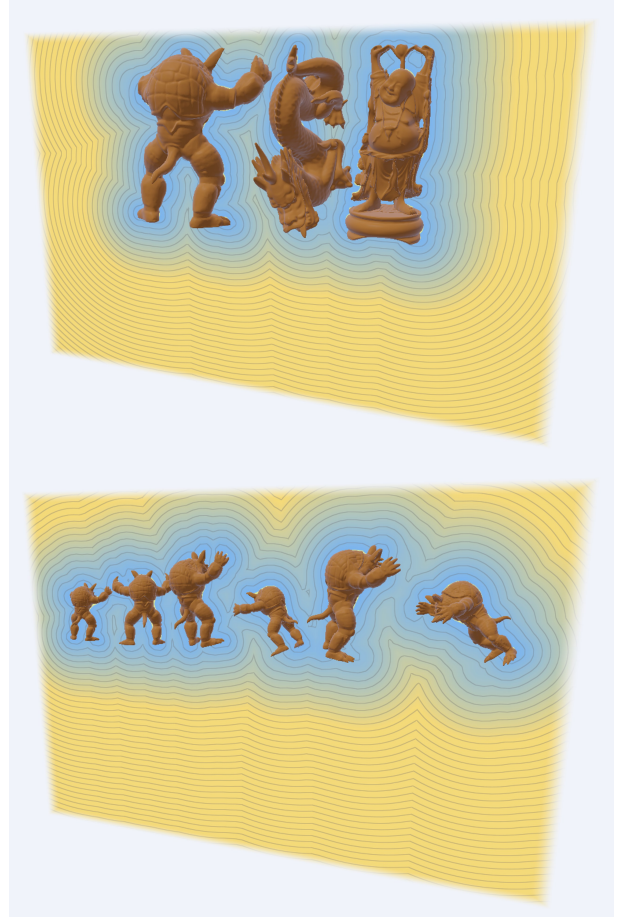


Fig. 7: Signed distance fields for a multi-object Stanford scene containing several deforming models. Distance regions from individual objects are consistently integrated into a single field.

hierarchy traversal rather than to directly copy SDF values from the previous frame.

Results for more complex scenes containing multiple objects are shown in Figure 7 and Figure 8. In these cases, the nearest triangle for a query point may switch between different objects, making the quality of hier-

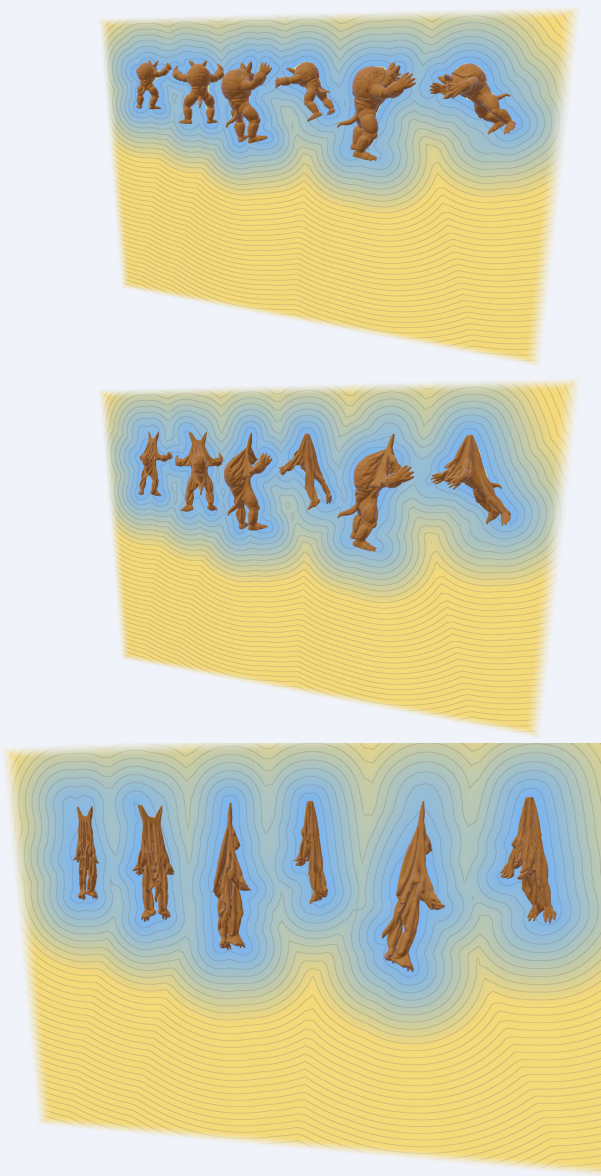


Fig. 8: Signed distance fields for a multi-object Armadillo scene with several independently transformed instances.

archical pruning and the initial distance upper bound more important.

The distance-query performance comparison between the baseline and the proposed method is presented in Table 2. The proposed method achieved lower query times in all test scenes, providing query-stage speedups ranging from $2.28\times$ to $2.51\times$. The average query time across all scenes decreased from 43.76 ms to 17.99 ms, corresponding to an approximately $2.43\times$ improvement.

The Stanford Party and Armadillo Party scenes also maintained query speedups of $2.48\times$ and $2.51\times$, respectively. This indicates that the benefits of temporal

coherence are not limited to isolated meshes and remain effective in scenes where multiple objects are distributed across a wide spatial region and nearest-object relations change over time.

The total time includes current triangle geometry updates, pseudonormal evaluation, hierarchy maintenance, and distance queries. Because geometry and normal updates are common to both methods, the total speedup is smaller than the query-only speedup; nevertheless, the overall SDF evaluation time was also reduced for all scenes.

Query workload comparisons are presented in Table 3, showing that the performance improvement is closely related to reductions in actual traversal workload. The proposed method reduced the average number of visited hierarchy nodes to 51% of the baseline and the average number of exact point-triangle evaluations to 43%. The larger reduction in exact triangle evaluations is primarily due to re-evaluating the persistent seed on the current geometry at the beginning of each query, which provides a relatively tight initial upper bound and enables additional triangle pruning within leaf nodes.

The stage-wise breakdown of the average execution time is presented in Table 4. Geometry update and pseudonormal evaluation have nearly identical costs for both methods because they operate on the same current geometry. Hierarchy maintenance also exhibits similar costs because both methods must evaluate the validity of current bounds. Therefore, the overall performance difference primarily arises from the distance-query stage, supporting the fact that the main objective of the proposed method is not to reduce the asymptotic complexity of hierarchy maintenance but to decrease the actual query traversal workload.

5.3 Ablation Study

To analyze the contribution of each temporal-coherence component, we compared variants in which anisotropic temporal bounds, adaptive hierarchy refresh, and the persistent triangle seed were individually removed. All variants used the same deforming geometry and hierarchy topology, while all implementation conditions other than the removed component were kept identical to those of the full method.

The ablation results are presented in Table 5. The full method achieved the lowest average query time of 17.99 ms. Removing the anisotropic temporal bound increased the query time to 23.84 ms and the number of visited nodes by 28%. This occurs because expanding bounds over broader directions regardless of local de-

Scene	Baseline Query (ms)	Ours Query (ms)	Query Speedup	Baseline Total (ms)	Ours Total (ms)
Stanford Bunny	18.42	7.36	2.50×	21.15	10.05
Stanford Dragon	46.81	19.84	2.36×	51.23	24.31
Stanford Armadillo	31.76	12.94	2.45×	35.32	16.51
Stanford Happy Buddha	39.28	17.21	2.28×	43.37	21.35
Stanford Party	58.94	23.76	2.48×	64.08	28.91
Armadillo Party	67.35	26.82	2.51×	72.91	32.44
Average	43.76	17.99	2.43×	48.01	22.26

Table 2: Distance-query performance for the tested deformable scenes. Times are measured per SDF evaluation. Query speedup is computed with respect to the full-refit baseline.

Scene	Visited Nodes	Exact Triangle Tests
Bunny	0.49	0.41
Dragon	0.53	0.45
Armadillo	0.50	0.42
Happy Buddha	0.56	0.48
Stanford Party	0.51	0.43
Armadillo Party	0.48	0.40
Average	0.51	0.43

Table 3: Average query workload relative to the baseline. The number of visited hierarchy nodes and exact point-triangle evaluations are normalized by the corresponding baseline measurements.

Stage	Baseline	Ours
Geometry update	1.08	1.08
Pseudonormal evaluation	1.34	1.34
Hierarchy maintenance	1.83	1.85
Distance query	43.76	17.99
Total	48.01	22.26

Table 4: Average computational breakdown per SDF evaluation in milliseconds.

formation increases spatial overlap between hierarchy nodes and consequently reduces pruning efficiency.

When adaptive refresh was removed, the query time increased to 21.67 ms. If a reference bound is retained for an extended period without being refreshed, the temporal pad required to accommodate deformation can increase, progressively reducing the tightness of the effective bound. Selective refresh limits this long-term degradation of bound quality.

Removing the persistent triangle seed resulted in the largest performance degradation, increasing the query time to 27.46 ms. The numbers of visited nodes and exact triangle tests also increased by 41% and 52%, respectively. This result demonstrates that the initial upper bound obtained by re-evaluating the nearest triangle from the previous evaluation on the current ge-

ometry plays an important role in the early pruning stage of branch-and-bound traversal.

Overall, the hierarchy-side temporal mechanisms and the query-side persistent seed reduce workload at different stages, and their combined use in the full method provides the highest performance.

5.4 Accuracy and Temporal Behavior

The proposed method exploits temporal information but does not approximate the signed distance itself. To verify this property, we compared the values computed by the proposed method and the baseline exact query on the same current geometry. For a query point $\mathbf{x}$, the absolute difference was defined as $E(\mathbf{x}) = |\phi_{\text{ours}}(\mathbf{x}) - \phi_{\text{baseline}}(\mathbf{x})|$, and the mean and maximum errors over the entire SDF grid were measured.

The accuracy comparison is presented in Table 6. The differences between the two methods remained at the level of floating-point numerical precision for all scenes. The persistent seed stores only the triangle ID rather than the previous distance value and re-evaluates the triangle on the current geometry, while hierarchy pruning uses conservative effective bounds that contain the current geometry. Therefore, temporal reuse changes the query workload but does not approximate the final nearest-distance solution.

Temporal hierarchy behavior is summarized in Table 7. The table reports the fraction of nodes whose stored reference bounds are actually updated during adaptive refresh. Here, “reused” does not mean that maintenance for the corresponding node is skipped; rather, it means that the existing reference bound is retained after evaluating bound validity with respect to the current geometry. Across all scenes, only 19.8% of the nodes were refreshed on average, while approximately 80.2% retained their previous reference bounds.

The refresh ratio was lower for relatively localized deformation, as in the Bunny scene, whereas it increased for scenes involving wider spatial motion, such as the

Method	Query Time (ms)	Normalized Visited Nodes	Normalized Triangle Tests
Full method	17.99	1.00	1.00
w/o Anisotropic Temporal Bound	23.84	1.28	1.24
w/o Adaptive Refresh	21.67	1.18	1.16
w/o Persistent Triangle Seed	27.46	1.41	1.52

Table 5: Ablation study of the proposed temporal-coherence components. Query time is averaged over all tested scenes, while traversal workloads are normalized with respect to the full method.

Scene	Mean Error	Maximum Error
Stanford Bunny	1.7×10^{-7}	2.1×10^{-6}
Stanford Dragon	2.3×10^{-7}	2.8×10^{-6}
Stanford Armadillo	1.9×10^{-7}	2.4×10^{-6}
Stanford Happy Buddha	2.6×10^{-7}	3.1×10^{-6}
Stanford Party	2.4×10^{-7}	3.0×10^{-6}
Armadillo Party	2.8×10^{-7}	3.4×10^{-6}

Table 6: Signed-distance difference between the proposed method and the baseline exact query.

multi-object cases. This indicates that reference-bound updates are determined by actual node-local deformation rather than by a fixed update frequency.

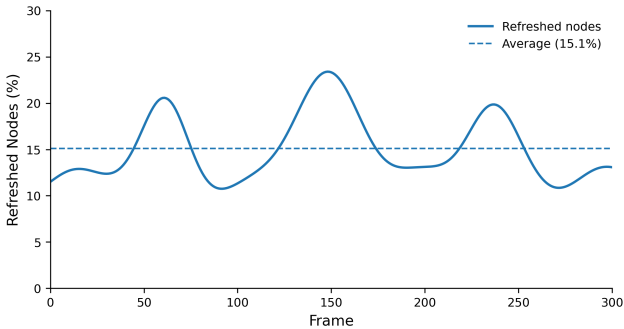


Fig. 9: Temporal variation of the hierarchy refresh ratio over a representative deforming sequence.

The frame-wise variation in refresh ratio for a representative deformation sequence is shown in Figure 9. The dashed line in the figure indicates the average refresh ratio of 15.1% measured for this representative sequence, whereas the 19.8% reported in Table 7 corresponds to the average over all test sequences. Around frames 50–75, 125–165, and 220–250, the refresh ratio temporarily increases as deformation becomes larger and then decreases as the motion becomes locally coherent again. Thus, adaptive refresh restores hierarchy quality during periods of larger deformation while retaining existing reference bounds during coherent-motion intervals.

6 Discussion

6.1 Performance Characteristics

The experimental results show that the main performance advantage of the proposed method does not come from eliminating hierarchy updates themselves, but from reducing the search space actually traversed during repeated distance queries. Because the triangle bounds of the current geometry and hierarchy validity must still be evaluated, the asymptotic cost of temporal maintenance remains $O(N_T + N_H)$. However, the reduction in actual workload is also evident in Table 3: both the number of visited nodes and the number of exact triangle evaluations are substantially reduced, resulting in lower overall SDF evaluation time.

The performance characteristics of the proposed method are directly related to geometric coherence between frames. When deformation between adjacent frames is small or localized, most reference bounds can be retained, and the previous nearest triangle is also likely to remain a useful initial candidate for the current query. Consequently, temporal reuse in both the hierarchy and query stages is effective under such continuous deformation.

Conversely, under very large inter-frame deformation or abrupt motion across the entire scene, more nodes may require refresh, and the initial upper bound obtained from the persistent seed may become relatively loose. In such cases, the performance benefit can decrease; however, refresh frequency and seed quality are not part of the correctness condition and therefore do not affect signed-distance accuracy.

From the GPU implementation perspective, keeping the hierarchy topology fixed throughout deformation allows the leaf-order triangle layout and escape links to be reused continuously. This reduces additional triangle-index indirection and thread-local traversal state during irregular hierarchy traversal, enabling reductions in algorithmic workload to translate into lower GPU execution time. Thus, the performance of the proposed method can be interpreted as the combined effect of temporal coherence and GPU-oriented hierarchical traversal.

Scene	Refreshed Nodes (%)	Reused Bounds (%)
Stanford Bunny	12.8	87.2
Stanford Dragon	21.6	78.4
Stanford Armadillo	16.9	83.1
Stanford Happy Buddha	18.7	81.3
Stanford Party	23.4	76.6
Armadillo Party	25.1	74.9
Average	19.8	80.2

Table 7: Temporal hierarchy behavior averaged over the tested sequences. Refreshed Nodes denotes nodes whose stored reference bounds are replaced by current bounds, while Reused Bounds denotes nodes that retain their reference bounds and use temporal padding.

6.2 Limitations

The proposed method targets deformable meshes whose triangle connectivity remains fixed. Therefore, when topology changes due to fracture, tearing, or adaptive remeshing, the existing hierarchy topology and triangle ordering are no longer valid, and hierarchy reconstruction becomes necessary.

The proposed method also provides its largest benefit in environments with high temporal coherence between frames. If the entire mesh moves over a very large distance or the geometry changes abruptly at every frame, reference-bound refreshes may occur for many nodes and the pruning efficiency of persistent seeds may decrease. Consequently, the performance difference relative to the full-refit baseline may become smaller under highly discontinuous deformation.

The effectiveness of persistent seeds also depends on the temporal coherence of query positions. In this work, the same regular SDF grid is repeatedly evaluated, so block-level seeds naturally correspond to previous queries. In contrast, for unstructured query applications in which query positions themselves change substantially at every frame, the upper-bound quality provided by previous seeds may decrease. Nevertheless, because each seed is re-evaluated on the current geometry, distance correctness is preserved even in such cases.

Finally, adaptive refresh limits modifications of stored reference bounds to a subset of nodes; however, the current implementation evaluates current-bound validity for all leaf and internal nodes. Therefore, selective refresh does not imply a dirty-node-only hierarchy update. To further reduce maintenance cost in large-scale scenes, future work could combine the proposed method with a dirty-region update strategy that directly tracks deformation locality and processes only hierarchy regions that are actually affected.

7 Conclusion

In this paper, we presented a temporally coherent hierarchical distance-query framework for accelerating repeated exact signed-distance evaluations on deformable triangle meshes. The proposed method preserves the hierarchy topology and triangle organization during deformation while maintaining a hierarchy that conservatively contains the current geometry through anisotropic temporal bounds and adaptive reference-bound refresh. In addition, the nearest triangle ID obtained from the previous evaluation is stored as a compact persistent seed and re-evaluated on the current geometry to provide an initial distance upper bound for the subsequent query.

Across a variety of single-object and multi-object deformation scenes, the proposed method maintained signed-distance values consistent with the baseline exact query within numerical precision while improving distance-query performance by up to approximately $2.5\times$. The ablation results further showed that hierarchy-side temporal bounds and the query-side persistent seed each contribute to reducing traversal workload, and that their combined use provides the highest performance.

The current method assumes fixed topology and temporally coherent query distributions. Future work could extend the framework to more general dynamic geometry by incorporating incremental hierarchy reconstruction for topology-changing deformation, dirty-region maintenance based on deformation locality, and adaptive temporal reuse that adjusts seed granularity according to the query distribution.

Acknowledgements (not compulsory)

The authors received no specific funding for this work.

Funding

This work was supported by INHA UNIVERSITY Research Grant. This work was supported by the National Research Foundation of Korea(NRF) grant funded by the Korea government(MSIT) (RS-2026-25474535). This work was supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (RS-2026-25548560, Artificial Intelligence Innovation Graduate School (Inha University)).

Author contributions statement

J.H.K. conceived the study, developed the methodology, implemented the system, conducted the experiments, analyzed the results, and wrote and reviewed the manuscript.

References

1. Sarah F. Frisken, Ronald N. Perry, Alyn P. Rockwood, and Thouis R. Jones, "Adaptively Sampled Distance Fields: A General Representation of Shape for Computer Graphics," in *Proceedings of SIGGRAPH 2000*, pp. 249–254, 2000. doi: 10.1145/344779.344899.
2. Jakob Andreas Bærentzen and Henrik Aanæs, "Signed Distance Computation Using the Angle Weighted Pseudonormal," *IEEE Transactions on Visualization and Computer Graphics*, vol. 11, no. 3, pp. 243–253, 2005. doi: 10.1109/TVCG.2005.49.
3. Mark W. Jones, J. Andreas Bærentzen, and Milos Sramek, "3D Distance Fields: A Survey of Techniques and Applications," *IEEE Transactions on Visualization and Computer Graphics*, vol. 12, no. 4, pp. 581–599, 2006. doi: 10.1109/TVCG.2006.56.
4. Christian Lauterbach, Sung-Eui Yoon, David Tuft, and Dinesh Manocha, "RT-DEFORM: Interactive Ray Tracing of Dynamic Scenes Using BVHs," in *IEEE Symposium on Interactive Ray Tracing*, pp. 39–46, 2006. doi: 10.1109/RT.2006.280213.
5. Ingo Wald, Solomon Boulos, and Peter Shirley, "Ray Tracing Deformable Scenes Using Dynamic Bounding Volume Hierarchies," *ACM Transactions on Graphics*, vol. 26, no. 1, 2007. doi: 10.1145/1189762.1206075.
6. Tero Karras, "Maximizing Parallelism in the Construction of BVHs, Octrees, and k-d Trees," in *Eurographics/ACM SIGGRAPH Symposium on High Performance Graphics*, pp. 33–37, 2012. doi: 10.2312/EGGH/HPG12/033-037.
7. Alec Jacobson, Ladislav Kavan, and Olga Sorkine-Hornung, "Robust Inside-Outside Segmentation Using Generalized Winding Numbers," *ACM Transactions on Graphics*, vol. 32, no. 4, pp. 33:1–33:12, 2013. doi: 10.1145/2461912.2461916.
8. Yan Gu, Yong He, Kayvon Fatahalian, and Guy E. Belloch, "Efficient BVH Construction via Approximate Agglomerative Clustering," in *Proceedings of High Performance Graphics*, pp. 81–88, 2013. doi: 10.1145/2492045.2492054.
9. Hongyi Xu and Jernej Barbič, "Signed Distance Fields for Polygon Soup Meshes," in *Proceedings of Graphics Interface*, pp. 35–41, 2014.
10. Ciprian Apetrei, "Fast and Simple Agglomerative LBVH Construction," in *Computer Graphics and Visual Computing*, pp. 41–44, 2014. doi: 10.2312/cgvc.20141206.
11. Jiří Bittner and Daniel Meister, "T-SAH: Animation Optimized Bounding Volume Hierarchies," *Computer Graphics Forum*, vol. 34, no. 2, pp. 527–536, 2015. doi: 10.1111/cgf.12581.
12. Dan Koschier, Crispin Deul, and Jan Bender, "Hierarchical hp-Adaptive Signed Distance Fields," in *Eurographics/ACM SIGGRAPH Symposium on Computer Animation*, pp. 189–198, 2016. doi: 10.2312/sca.20161236.
13. Jakub Hendrich, Daniel Meister, and Jiří Bittner, "Parallel BVH Construction Using Progressive Hierarchical Refinement," *Computer Graphics Forum*, vol. 36, no. 2, pp. 487–494, 2017. doi: 10.1111/cgf.13143.
14. Daniel Meister and Jiří Bittner, "Parallel Locally-Ordered Clustering for Bounding Volume Hierarchy Construction," *IEEE Transactions on Visualization and Computer Graphics*, vol. 24, no. 3, pp. 1345–1353, 2018. doi: 10.1109/TVCG.2017.2669983.
15. Daniel Meister and Jiří Bittner, "Parallel Reinsertion for Bounding Volume Hierarchy Optimization," *Computer Graphics Forum*, vol. 37, no. 2, pp. 463–473, 2018. doi: 10.1111/cgf.13376.
16. Gavin Barill, Neil G. Dickson, Ryan Schmidt, David I. W. Levin, and Alec Jacobson, "Fast Winding Numbers for Soups and Clouds," *ACM Transactions on Graphics*, vol. 37, no. 4, pp. 43:1–43:12, 2018. doi: 10.1145/3197517.3201337.
17. Jeong Joon Park, Peter Florence, Julian Straub, Richard Newcombe, and Steven Lovegrove, "DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 165–174, 2019.
18. Matan Atzmon and Yaron Lipman, "SAL: Sign Agnostic Learning of Shapes From Raw Data," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2565–2574, 2020.
19. Amos Gropp, Lior Yariv, Niv Haim, Matan Atzmon, and Yaron Lipman, "Implicit Geometric Regularization for Learning Shapes," in *Proceedings of the 37th International Conference on Machine Learning*, vol. 119, pp. 3789–3799, PMLR, 2020.
20. Daniel Meister, Shinji Ogaki, Carsten Benthin, Michael J. Doyle, Michael Guthe, and Jiří Bittner, "A Survey on Bounding Volume Hierarchies for Ray Tracing," *Computer Graphics Forum*, vol. 40, no. 2, pp. 683–712, 2021. doi: 10.1111/cgf.142662.
21. Baorui Ma, Zhizhong Han, Yu-Shen Liu, and Matthias Zwicker, "Neural-Pull: Learning Signed Distance Function from Point Clouds by Learning to Pull Space onto Surface," in *Proceedings of the 38th International Conference on Machine Learning*, vol. 139, pp. 7246–7257, PMLR, 2021.
22. Peng Wang, Lingjie Liu, Yuan Liu, Christian Theobalt, Taku Komura, and Wenping Wang, "NeuS: Learning Neural Implicit Surfaces by Volume Rendering for Multi-view Reconstruction," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 27171–27183, 2021.
23. Lior Yariv, Jiatao Gu, Yoni Kasten, and Yaron Lipman, "Volume Rendering of Neural Implicit Surfaces," in *Advances in Neural Information Processing Systems*, vol. 34, pp. 4805–4815, 2021.

24. Jong-Hyun Kim, “Efficient GPU Framework for Adaptive and Continuous Signed Distance Field Construction, and Its Applications,” *Journal of The Korea Society of Computer and Information*, vol. 27, no. 3, pp. 63–69, 2022. doi: 10.9708/jksci.2022.27.03.063.
25. Carsten Benthin, Radoslaw Drabinski, Lorenzo Tesari, and Addis Dittebrandt, “PLOC++: Parallel Locally-Ordered Clustering for Bounding Volume Hierarchy Construction Revisited,” *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, 2022. doi: 10.1145/3543867.
26. Eduard Pujol and Antonio Chica, “Triangle Influence Supersets for Fast Distance Computation,” *Computer Graphics Forum*, vol. 42, no. 6, p. e14861, 2023. doi: 10.1111/cgf.14861.
27. Eduard Pujol and Antonio Chica, “Adaptive Approximation of Signed Distance Fields Through Piecewise Continuous Interpolation,” *Computers & Graphics*, vol. 114, pp. 337–346, 2023. doi: 10.1016/j.cag.2023.06.020.
28. Zhaoshuo Li, Thomas Müller, Alex Evans, Russell H. Taylor, Mathias Unberath, Ming-Yu Liu, and Chen-Hsuan Lin, “Neuralangelo: High-Fidelity Neural Surface Reconstruction,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8456–8465, 2023.
29. Eduard Pujol and Antonio Chica, “Rendering Piecewise Approximations of SDFs Through Analytic Intersections,” *Computers & Graphics*, vol. 122, p. 103981, 2024. doi: 10.1016/j.cag.2024.103981.
30. Carsten Benthin, Daniel Meister, Joshua Barczak, Rohan Mehalwal, John A. Tsakok, and Andrew Kensler, “H-PLOC: Hierarchical Parallel Locally-Ordered Clustering for Bounding Volume Hierarchy Construction,” *Proceedings of the ACM on Computer Graphics and Interactive Techniques*, vol. 7, no. 3, pp. 30:1–30:14, 2024. doi: 10.1145/3675377.