Sensors and Materials, Vol. 38, No. 7 (2026) 3907–3920
MYU Tokyo

S & M 4541

Lightweight Real-time Slicing Method of 3D Geological Body for Mine Visualization

Xiao Xingxing, Wang Canhui,* and Jing Yanwu

Surveying and Mapping Geographic Information Center,
Sichuan Institute of Geological Survey, Chengdu 610072, China

(Received April 7, 2026; accepted June 12, 2026)

Keywords: three-dimensional geological model, real-time slicing, GPU rendering, mine visualization, attribute index texture

High-precision geological models in smart mines typically feature massive data volumes and highly complex topological structures. Consequently, traditional geometric slicing methods struggle to meet millisecond-level interactive demands, while voxel-based approaches suffer from precision loss when delineating orebody boundaries. To address these issues, we propose a lightweight, real-time slicing algorithm tailored for mine visualization. By leveraging the parallel computing capabilities of GPUs, we introduce a pixel-masking slicing strategy that transforms complex geometric intersections into pixel-level retention judgments, effectively decoupling the computational load from the model's geometric complexity. Furthermore, an attribute-indexed texture mechanism is developed. Through dual-layer "geometry–texture–attribute" mapping, critical geological data—such as lithology and ore grade—are baked directly into the UV space. This overcomes the traditional limitations of render-based slicing, enabling the real-time querying of stratum attributes. Experimental results demonstrate that for complex geological models containing up to 10 million facets, the single-slice response time remains strictly between 1.2 and 1.6 ms, with dynamic interactive frame rates sustained at approximately 60 frames per second. Notably, this performance exhibits no fluctuation despite exponential increases in mining area and geological body scale. The generated cross sections display crisp, anti-aliased boundaries and seamlessly support real-time interactive queries for ore grade, lithology, and reserves. Demonstrating both high efficiency and high fidelity, this approach resolves the bottlenecks of real-time spatial analysis in complex geological environments, providing robust visualization support for building smart mine digital twins and predicting deep-seated mineral deposits.

1. Introduction

As global shallow mineral resources become increasingly depleted, China has comprehensively implemented the strategic initiative of "deep earth exploration". Consequently, mining operations are progressively extending into deep, complex geological environments,

*Corresponding author: e-mail: 1108130023033@stu.bucea.edu.cn
https://doi.org/10.18494/SAM6369

ISSN 0914-4935

imposing higher demands on the precise representation and real-time analysis of geological information.[1] As the core technology for constructing "digital mines" and "smart mines", 3D geological modeling has been widely applied in key areas such as orebody delineation, reserve estimation, mining design, and geological hazard early warning.[2] High-precision 3D geological models not only intuitively reconstruct the spatial distribution characteristics of underground structures but also provide reliable quantitative bases for engineering decision-making.[3] In geological engineering practice, real-time slicing is the most direct and frequently utilized interactive method for exploring the internal structure of geological bodies, revealing stratigraphic contact relationships, and analyzing mineralization enrichment patterns.[4] However, with the continual improvement in geological survey precision, the geometric scale of engineering-level 3D models has grown exponentially. Achieving efficient slicing that simultaneously ensures "real-time interactive efficiency", "geometric precision", and "deep attribute querying" in a massive data environment has become a critical bottleneck restricting the digital development of mines.

Existing 3D geological slicing technologies can generally be categorized into three types. The first category comprises precise slicing methods based on CPU geometric intersection, including triangulated irregular network (TIN) intersection and generalized topological operations.[5] These methods reconstruct the cross-sectional geometry by calculating the topological intersection between the slicing plane and the model mesh, theoretically guaranteeing result accuracy.[6] However, their computational complexity exhibits a linear or even superlinear relationship with the number of model facets. When the model scale exceeds millions of facets, a single slicing operation often takes several seconds, failing to meet the demands for real-time dynamic analysis at engineering sites.[7] The second category involves discretization-based slicing methods utilizing voxels or octrees.[8] These approaches rapidly extract cross-sectional voxels via spatial indexing. Although computational efficiency improves significantly, this process is fundamentally a dimensional reduction approximation of continuous geometry. It leads to a pronounced "staircase aliasing effect" on section boundaries, with resolution inherently constrained by voxel size, making it inadequate for the stringent boundary precision requirements of fault intersection analysis and precise reserve calculation.[9]

With the leap in general-purpose graphics processing unit (GPU) computing capabilities, real-time slicing schemes that integrate the section generation process with the GPU rendering pipeline have increasingly become mainstream. Deng *et al.* proposed a key technology for the online cross-sectional analysis of 3D geological models within a virtual earth environment, validating the feasibility of GPU-accelerated geological slicing in web-based platforms.[10] Li *et al.* further advanced this direction by introducing a per-pixel linked list (PPLL) architecture that leverages fragment shaders for pixel-level ray casting, enabling the real-time generation of multiple simultaneous geological sections without repeated model rendering.[11] These methods leverage the GPU's parallel architecture to dynamically cull geometric primitives within the rendering pipeline, achieving millisecond-level interactive responses. Nevertheless, existing GPU slicing techniques predominantly focus on 'geometric clipping' at the visual presentation level. Essentially functioning as pixel-level display control, they lack an associative mechanism for geological semantic information. In smart mine construction, 3D subsurface geological models are increasingly required to support not only visual rendering but also real-time deep

attribute querying for underground digital twin applications;[12] however, conventional GPU fragment-discarding operations result in the loss of critical geological attributes embedded within geometric primitives—such as lithological codes, ore grades, and mechanical parameters—creating an information gap where users can only 'visualize the cross-sectional morphology but cannot query the underlying attribute data'. The rapidly growing scale and complexity of engineering-grade geological models further exacerbate this limitation, as the real-time visualization of massive hexahedral or tetrahedral meshes remains a fundamental performance bottleneck.[13]

This critical flaw severely restricts the application depth of GPU slicing technology in quantitative mine analysis. Furthermore, driven by the digital transformation of mines and the increasing adoption of digital twin technologies for underground engineering, geological visualization application scenarios are migrating from high-performance workstations to web browsers and mobile terminals. As highlighted in recent comprehensive reviews of 3D geological modeling, integrating multisource geological data and achieving simultaneous high-efficiency visualization and attribute querying remain persistent challenges across diverse geological scenarios.[14] Existing engineering-oriented visualization systems—such as shear-warp volume rendering-based approaches for mine 3D visualization[15] and multisensor fusion-based digital twin platforms for coal mine tunnels[16]—have advanced the state of the art but are predominantly designed for high-performance hardware deployments and offer limited support for lightweight cross-platform access. Furthermore, recent advances in implicit neural-network-based geological modeling methods have substantially expanded the representable scale and topological complexity of subsurface models,[17] imposing even more demanding requirements on real-time rendering systems. Traditional heavy desktop software, hindered by complex deployment, high hardware thresholds, and difficulties in data sharing, struggles to adapt to these evolving demands for lightweight, collaborative visualization workflows.

In response to the above-mentioned issues, the proposed method shifts the slicing computation from the traditional CPU-based per-facet geometric intersection to the pixel level within the GPU rendering pipeline. As illustrated in Fig. 1, the CPU dynamically defines the

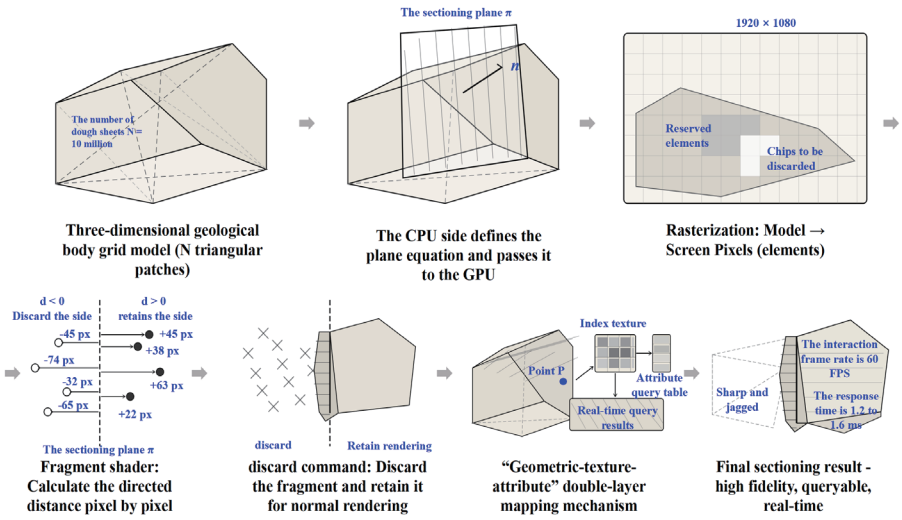

Fig. 1. (Color online) Schematic diagram of lightweight real-time sectioning method.

slicing plane equation on the basis of user interactions and transmits these parameters to the GPU. Following the rasterization of the geological model, the fragment shader evaluates the spatial relationship between each generated fragment and the clipping plane in parallel. Fragments positioned in the culling half-space are explicitly discarded, whereas those on the retained side proceed through the standard rendering pipeline, thereby producing a highly precise cross-sectional profile. Furthermore, by prebaking stratum identifiers into a UV index texture, the system enables interactive interrogation: when a user clicks any location on the section, a reverse query of the attribute table is executed via texture sampling. This allows for the real-time retrieval of localized geological properties, such as lithology and ore grade, fundamentally resolving the "visible but unqueryable" bottleneck typical of conventional GPU-based slicing approaches.

2. Lightweight Real-time Slicing Method

Figure 2 illustrates the workflow of the proposed lightweight, real-time slicing method for 3D geological bodies in engineering visualization, with the specific procedure detailed as follows. (1) 3D geological model data are first acquired and converted into the requisite format, followed by the construction of a mapping lookup table that links stratum IDs to their corresponding geological attributes. (2) The slicing plane equation and its normal vector are calculated to formulate the clipping plane parameters, which are subsequently transmitted to the GPU shader.

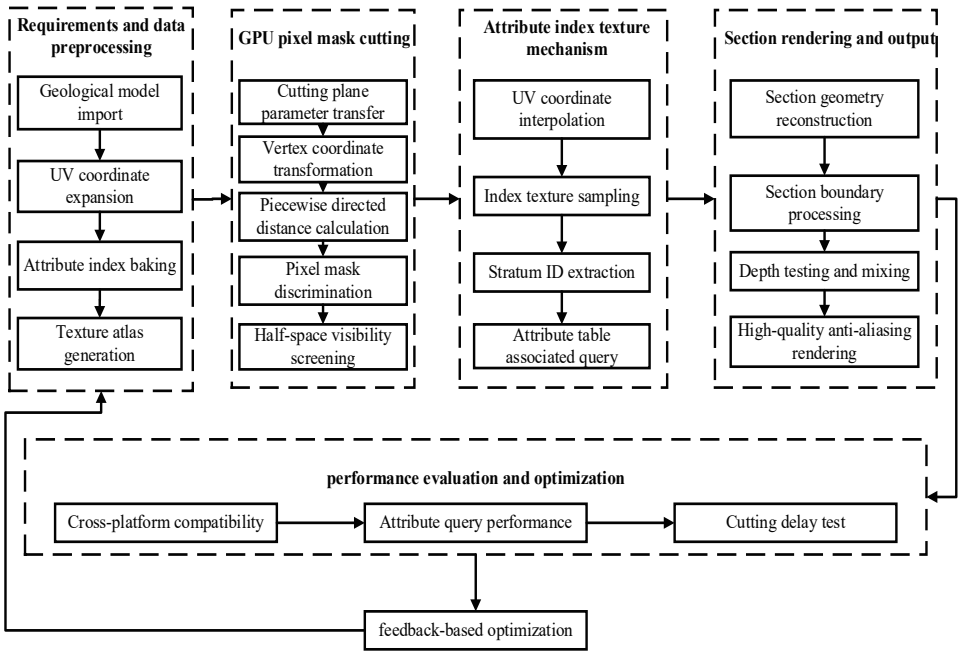

Fig. 2. Schematic diagram of lightweight real-time sectioning method. Note: 'Cross-platform compatibility' refers to an inherent property of the WebGL 2.0 implementation framework rather than an independently proposed algorithmic component; its inclusion reflects the practical deployment objective of the system.

(3) Utilizing a GPU-based pixel masking algorithm, the signed distance from each fragment to the slicing plane is computed within the fragment shader; fragments located on the culling side are directly discarded on the basis of the distance sign, thereby achieving a millisecond-level slicing response. (4) An attribute-indexed texture mechanism is then employed to bake the stratum identifiers into the UV texture space; during the slicing operation, the stratum ID is retrieved via texture sampling, enabling immediate queries of the lookup table to extract comprehensive stratum attributes. (5) Color mapping is applied on the basis of the stratum IDs, and the sliced result is output to the frame buffer, supporting both real-time on-screen display and high-definition off-screen rendering. (6) Finally, rigorous evaluations encompassing slicing latency, attribute query efficiency, and cross-platform compatibility are conducted to verify that the algorithm meets the real-time and accuracy requirements of engineering visualization, ultimately yielding a cross-sectional model that supports interactive querying.

2.1 Real-time sectioning algorithm based on GPU pixel mask

Traditional geological slicing methods typically execute geometric intersection operations on 3D mesh models at the CPU end, a process that inherently suffers from low computational efficiency. To overcome this limitation, we abandon the conventional computational logic of performing CPU-based geometric intersections on geological meshes. Instead, we propose shifting the slicing computation from the geometric domain to the rendering pipeline. By leveraging the GPU fragment shader to implement screen-space pixel-level filtering, the slicing operation is fundamentally transformed into a half-space visibility determination problem. In 3D Euclidean space $\mathbb{R}^3$, an arbitrary slicing plane π can be uniquely defined by its unit normal vector $n = (n_x, n_y, n_z)$ and a scalar plane parameter d. The plane equation is formally expressed as

$$\pi: \boldsymbol{n}\cdot\boldsymbol{x} + d = 0, \tag{1}$$

where the normal vector n points toward the retained region within the visible half-space, $P(x, y, z)$ represents the coordinates of an arbitrary spatial point, and d denotes the signed distance from the origin to the plane. For any given point on the surface of the geological body $\boldsymbol{p} \in \mathbb{R}^3$ (defined within the world coordinate system), its signed distance $\delta(\boldsymbol{p})$ to the slicing plane π is formulated as

$$\delta(p) = n \cdot p + d. \tag{2}$$

The algebraic sign of this signed distance $\delta(\boldsymbol{p})$ strictly dictates the spatial positional relationship of the point P relative to the slicing plane π. Accordingly, the pixel state determination function, denoted as $State(\boldsymbol{p})$, is constructed as

$$State(\boldsymbol{p}) = \begin{cases} Retain(\text{Reserve}), & \text{if } \delta(\boldsymbol{p}) \geq 0 \\ Discard(\text{Eliminate}), & \text{if } \delta(\boldsymbol{p}) < 0 \end{cases} \tag{3}$$

Specifically, as defined in Eq. (3), when $\delta(\boldsymbol{p}) \geq 0$, the point p resides either exactly on the plane or within the half-space indicated by the normal vector, designating it as part of the visible region that must be retained by the rendering pipeline. Conversely, when $\delta(\boldsymbol{p}) < 0$, the point is situated on the reverse side of the plane; in this scenario, the rendering pipeline immediately aborts all subsequent processes for the associated fragment.

This algorithm is deeply based on the modern programmable graphics rendering pipeline. By intervening in the processing flow of geometric data at the shader stage, it achieves the nondestructive real-time sectioning of geological bodies. The implementation logic is shown in Fig. 3 and mainly consists of three core stages: CPU parameter passing, vertex data transformation and interpolation, and chip-level mask discrimination.

The construction and transmission of plane parameters at the CPU end primarily address the real-time calculation of the slicing plane's general equation [Eq. (1)], $Ax + By + Cz + d = 0$, immediately after the user defines the slicing position and orientation via the interactive interface. The algorithm extracts the plane's unit normal vector $\vec{n} = (A, B, C)$ and the origin distance parameter d, encapsulating them into a 4D vector $U_{plane} = (A, B, C, D)$. This vector is subsequently transmitted to the GPU video random access memory (VRAM) via a uniform variable mechanism. It serves as global read-only data shared across all shader cores, effectively circumventing the computational overhead associated with recalculating geometric intersection lines on a per-frame basis. During the world coordinate transformation and interpolation phase within the vertex shader, the system retrieves the local vertex coordinates, P_{local}, of the geological model. By multiplying these local coordinates by the model matrix M, they are transformed into absolute coordinates within the world coordinate system, denoted as

$$P_{world} = M \cdot P_{local}. \tag{4}$$

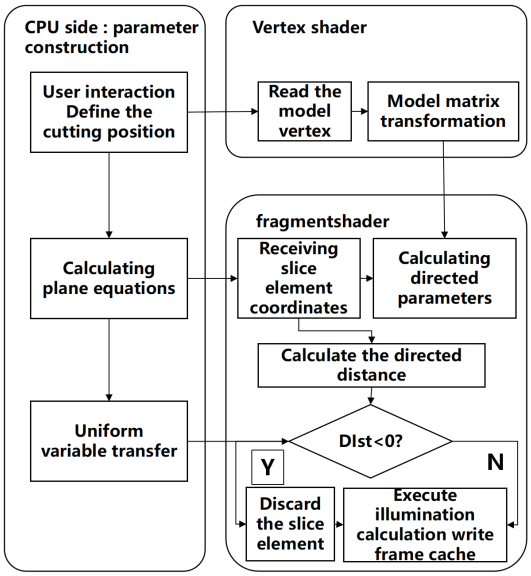

Fig. 3. Real-time slicing algorithm logic based on GPU pixel masks.

The calculated world coordinate, P_{world} [Eq. (4)], is declared as a varying variable and passed to the rasterizer. During primitive generation, the rasterizer automatically executes barycentric linear interpolation across the world coordinates of the triangle vertices, thereby assigning a highly precise world-space position coordinate, P_{frag}, to each subsequently generated fragment. In the ensuing pixel masking and culling phase executed within the fragment shader, the shader evaluates every pixel on the screen by receiving its interpolated coordinate, P_{frag}. By incorporating the transmitted plane parameters, U_{plane}, the shader calculates the signed distance, *Dist*, from the given fragment to the slicing plane:

$$Dist = \vec{n} \cdot P_{frag} + D. \tag{5}$$

The spatial positional relationship of the fragment relative to the clipping plane is determined by the algebraic sign of *Dist* [Eq. (5)]. If $Dist < 0$ (indicating that the fragment lies within the negative half-space of the slicing plane), the graphics API's inherent fragment discard instruction is invoked to forcibly terminate any further processing of that fragment. Discarded fragments are subsequently excluded from both the depth and color buffers, thereby generating a visually sharp and geometrically precise cross-sectional profile without any need to reconstruct the mesh topology.

By fundamentally shifting the dimensionality of the slicing computation from the "geometric primitive level" down to the "screen pixel level", the algorithm significantly simplifies the complexity model of the slicing operation. Leveraging the GPU's parallel architecture, the pixel masking strategy offloads the computational burden to the post-rasterization fragment stage. Thus, the computational volume processed by the GPU depends exclusively on the number of pixels covered within the current viewport, achieving complete decoupling from the model's intrinsic geometric facet count, N. For a given screen resolution of $W \times H$, the algorithm's time complexity is strictly bounded by $O(W \times H)$. Consequently, provided the rendering resolution remains constant, the computational overhead of the slicing operation on massive geological models is effectively reduced to a constant level, $O(1)$, relative to the model scale. This architectural advantage guarantees that the system consistently sustains a millisecond-level interactive response speed, even when navigating immense volumes of geological data.

2.2 Attribute-index texture mechanism

To resolve the issue of geometric forms being severed from geological semantics in traditional graphics slicing methods, and to achieve lossless attribute transmission and real-time retrieval during the slicing process, a dual-layer "geometry–texture–attribute" mapping mechanism is employed. By decoupling geometric rendering from attribute storage, this approach establishes a precise indexing link from screen pixels to deep-seated geological attributes while maintaining visualization efficiency. Although traditional rendering pipeline-based slicing technologies can efficiently generate visual cross sections, they fundamentally manipulate color values within the frame buffer. Consequently, geological attributes attached to the geometry—such as lithology, permeability, and mechanical strength—are inevitably lost during the rasterization process. To

address this critical flaw, a dual-layer data architecture comprising an attribute-indexed texture and an attribute lookup table is utilized to physically separate graphic rendering data from engineering attribute data.

The attribute-indexed texture is configured as a 2D data container within the GPU VRAM and functions as the middleware between the geometric and attribute spaces. Serving as a spatial lookup table, its texel channels store the unique stratum identifier corresponding to that specific location. Moreover, the attribute lookup table operates Q_{Table} as a structured semantic database, residing within a structured buffer in either the main memory or VRAM. This architecture establishes a definitive mapping relationship from the Layer ID to the complete set of geological attributes. Mathematically, the attribute retrieval $A(P)$ process for an arbitrary point P on the surface of the geological body can be formalized as a composite function of the geometric index mapping f_{geo} and the attribute association mapping f_{attr}:

$$A(P) = f_{attr} \circ f_{geo}(P) = f_{attr}\left(f_{geo}(P)\right). \tag{6}$$

In the above formula, f_{geo} denotes the geometric index mapping function, which maps a surface point P to its corresponding UV coordinates (u_P, v_P) via the precomputed LSCM parameterization, and subsequently retrieves the stratigraphic code k by sampling the index texture T_{ID} at those coordinates using nearest-neighbor filtering.

$$k = f_{geo}(P) = Sample(T_{ID}, uv_P) \tag{7}$$

f_{attr} is an attribute-associated mapping function. According to the obtained code k, it retrieves the corresponding attribute set Ω_k in the query table, including lithology description, geological age, elastic modulus, and other engineering parameters.

$$\Omega_k = f_{attr}(k) = Lookup(Q_{Table}, k) \tag{8}$$

The technical implementation logic primarily encompasses two core phases: offline preprocessing and online rendering interaction. The offline preprocessing of the attribute index texture involves three tightly coupled steps. First, each stratigraphic layer mesh is decomposed into geometrically coherent charts using an 85° dihedral angle threshold, and least squares conformal mapping (LSCM) is applied within each chart to minimize angular distortion; all charts are then packed into the unit UV square $[0,1]^2$ via a shelf-packing algorithm with a mandatory 2-texel interchart margin to prevent cross-chart bleeding. Second, the index texture T_{ID} is configured as a single-channel 32-bit unsigned integer texture at 4096 × 4096 resolution, ensuring the lossless storage of stratigraphic IDs without quantization error. Third, during index value baking, a conservative rasterization strategy is adopted so that every texel partially overlapping a triangle boundary is assigned the corresponding stratigraphic integer k, eliminating sub-texel gaps at chart edges. A seam-padding postprocess further replicates texel

values across UV-discontinuous seam boundaries, and nearest-neighbor filtering is strictly enforced at runtime to prevent interpolation-induced misclassification, collectively guaranteeing 100% attribute mapping accuracy across all tested model scales. This restriction prevents default linear interpolation from generating invalid intermediate floating-point values at stratigraphic boundaries, thereby guaranteeing absolute accuracy in attribute index retrieval. Furthermore, at the interaction and data architecture level, a comprehensive "geometry-attribute" decoupled storage scheme is adopted. Only the lightweight ID index texture resides within the GPU VRAM, where ray casting is employed to acquire UV coordinates and execute real-time reverse queries for the Layer ID via Eqs. (7) and (8). Concurrently, massive volumes of borehole logs and engineering text data are retained in the main memory or a cloud database, fetched strictly on demand upon user interaction. This architectural design not only achieves microsecond-level query responses but also markedly reduces VRAM overhead within the WebGL environment, providing vital support for the lightweight deployment of complex geological data across both web and mobile platforms.

3. System Implementation and Experimental Results

3.1 Experimental environment and data

All experiments were conducted on a hardware platform equipped with an Intel Core i7-12700K CPU (3.60 GHz), 32 GB of DDR4 RAM, and an NVIDIA GeForce RTX 3070 GPU. The software environment was built upon the WebGL 2.0 standard and executed within the Google Chrome browser (version 118.0). The 3D geological modeling data were sourced from a silica sandstone mining area in Sichuan Province, China. Located in the tectonic transition zone between the front and back ranges of the Longmen Mountains, the study area lies within the middle to low-middle mountainous region on the northern margin of the Sichuan Basin. The topography exhibits high elevations in the northwest and lower elevations in the southeast, characterized by intense topographic dissection. With a maximum relative elevation difference of 560.8 m and predominant topographic slopes ranging from 30 to 60°, the region presents highly complex geological structures and geomorphic features. These typical characteristics of a high and steep mountainous mine impose rigorous demands on the precision and terrain representation of the 3D modeling. Consequently, high-density geological borehole data from the mining area underwent systematic cleaning and standardized preprocessing. By eliminating erroneous values and topological anomalies from the original records, and standardizing all spatial data to the China Geodetic Coordinate System CGCS2000, a robust geological database was established, which ultimately facilitated the construction of the high-precision 3D orebody model.

3.2 Comparative analysis of sectioning efficiency

Figure 4 further visualizes these comparative advantages, highlighting the algorithmic superiority of the proposed GPU pixel masking method. As clearly depicted in the response time

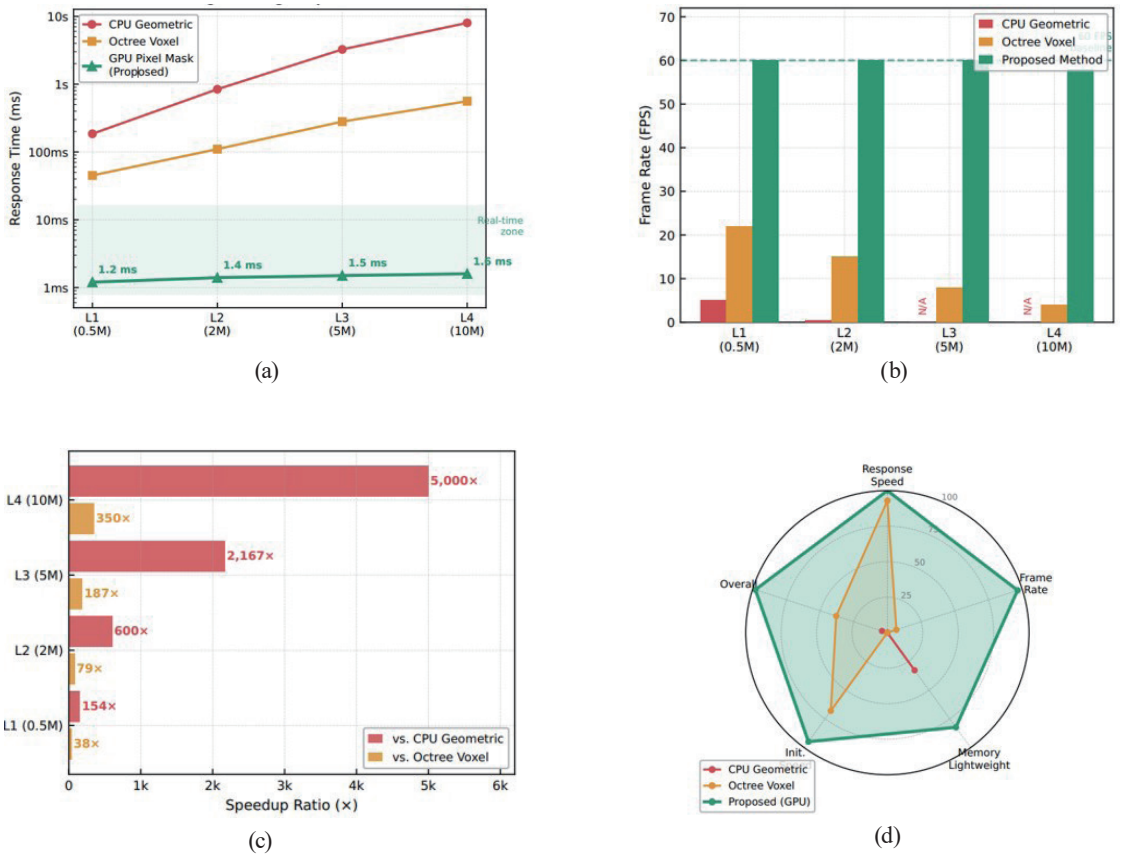

Fig. 4. (Color online) Performance characteristics of real-time slicing algorithms for 3D geological models. (a) Single slicing response time. (b) Dynamic interaction frame rate. (c) Speedup ratio of proposed GPU method. (d) Comprehensive performance radar (L4: 10M triangles).

trends in Fig. 4(a), the proposed method reduces the slicing computation to a constant-time complexity, $O(1)$, which depends solely on screen resolution rather than geometry. This transformation completely decouples the computational load from the model's geometric volume, consistently maintaining an ultralow latency of 1.2–1.6 ms (within the ideal real-time zone) and a full interactive frame rate of approximately 60 FPS across all tested scales, as shown in Fig. 4(b). Furthermore, the speedup ratio analysis results in Fig. 4(c) illustrate that the proposed method achieves up to 5000-fold and 350-fold accelerations compared with the CPU geometric and octree voxel methods, respectively, at the maximum model scale. Ultimately, the comprehensive performance radar chart in Fig. 4(d) confirms that the proposed method dominates across all metrics, with significant advantages in response speed, interactive fluidity, and memory lightweight characteristics. This proves that the method is highly capable of supporting the real-time analysis of massive mining geological data on both web and mobile platforms.

As detailed in Table 1, the quantitative performance metrics across multilevel model scales (L1–L4) reveal the severe computational limitations of conventional approaches. The traditional CPU geometric intersection method is fundamentally constrained by its linear complexity, $O(N)$. Consequently, when processing with the 10-million-facet model (L4), its single-sectioning

Table 1
Performance characteristics of different slicing algorithms under multilevel model scales.

Model scale (number of patches)	Algorithm type	Init. time (ms)	Single slicing response time (ms)	Dynamic frame rate (FPS)	VRAM/ memory (MB)	Attribute query response time (μs)	Attribute mapping accuracy (%)
L1 (500K)	CPU geometry intersection	1240	185	5	180	−3200	100
	Octree voxel method	450	45	22	320	−1800	91.3
	Proposed method	120	1.2	60+	85	**<12**	**100**
L2 (2M)	CPU geometry intersection	4860	840	< 1	720	−14500	100
	Octree voxel method	1820	110	15	1150	−4200	89.7
	Proposed method	350	1.4	60+	210	**<15**	**100**
L3 (5M)	CPU geometry intersection	14500	3250	Stutter	1850	N/A	100
	Octree voxel method	4200	280	8	2600	−9600	87.2
	Proposed method	820	1.5	60	480	**<18**	**100**
L4 (10M)	CPU geometry intersection	>30000	>8000	Unable	>3500	N/A	100
	Octree voxel method	9600	560	4	5200	−21300	86.5
	Proposed method	1,550	1.6	58	920	**<23**	**100**

response time exceeds 8000 ms, causing the system to completely lose its real-time interactive capability. Conversely, the octree voxel method encounters a severe resource bottleneck: as the model scale and precision requirements increase, memory consumption surges markedly, peaking at 5200 MB for the L4 model. This memory overhead is accompanied by a precipitous drop in dynamic interactive frame rate to a mere 4 FPS. In stark contrast, benefiting from the dual-layer "geometry–texture–-attribute" mapping mechanism, the proposed method's video memory consumption is restricted to 920 MB at the highest scale—approximately one-fifth of that required by the voxel method—demonstrating exceptional resource efficiency.

3.3 Verification of sectioning effect and attribute query

To evaluate the comprehensive performance of the proposed lightweight real-time 3D geological body slicing method in practical engineering scenarios, real-time interactive tests were conducted on a WebGL platform using the high-precision 3D geological model of the silica sandstone mine in Sichuan Province. Figure 5 illustrates the real-time visual output of the slicing function alongside its corresponding geological profile analysis window. The algorithm delivers exceptionally high visual fidelity during the arbitrary-angle slicing of complex layered geological

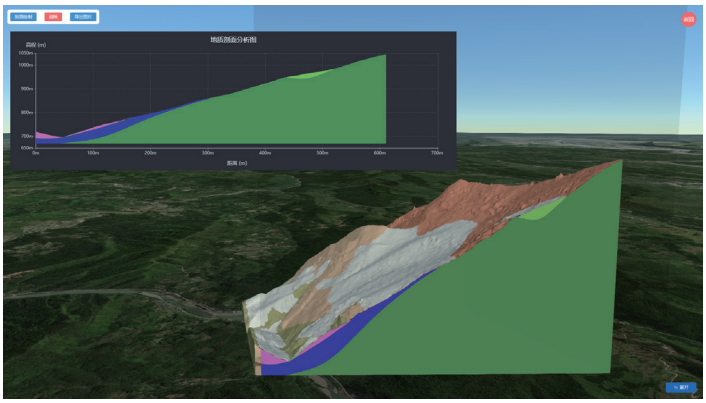

Fig. 5. (Color online) Real-time slicing effect and geological attribute analysis verification based on the proposed method. The inset (upper left) demonstrates point-wise attribute querying: clicking any pixel on the cross section triggers a reverse UV lookup in the index texture, retrieving and displaying the lithology, stratigraphic age, and ore grade at that specific location in real time.

bodies that include fault structures. By leveraging the GPU fragment-level pixel mask decision method, the resulting cross-sectional edges remain sharp and smooth, effectively mitigating the staircase aliasing effect commonly associated with traditional voxelization approaches. This ensures an accurate reconstruction of the intersection geometry between the strata and the slicing plane. Furthermore, critical internal features within the profile, including the stratigraphic sequence, thickness variations, and fault slicing relationships, are depicted with marked clarity and continuity. This successful representation validates the algorithm's robustness when processing complex topological structures.

Figure 5 illustrates the real-time slicing and linked analysis effects on complex geological bodies achieved using the proposed method. The inset in the upper-left corner of the interface displays a 2D geological profile analysis map, generated dynamically by the dual-layer "geometry–texture–attribute" mapping mechanism. As the user arbitrarily drags or rotates the clipping plane within the 3D scene, this 2D profile map synchronizes and updates at the millisecond level. Even within authentic mining environments featuring complex surface terrains, this mechanism accurately and distinctly renders the stratigraphic undulations and contact relationships along the slicing path. This robust performance thoroughly validates the advanced nature of the attribute-indexed texture mechanism. By entirely circumventing the computationally exhaustive traversal of massive 3D mesh data, the system relies solely on highly efficient GPU texture sampling to instantaneously reconstruct high-precision cross-sectional geometry. Additionally, the stratigraphic colors in the 2D analysis map maintain strict consistency with the 3D model, accurately reflecting the spatial locations of tectonic structures such as faults. A deep graphical-attribute linkage is simultaneously established, precisely correlating any pixel on the profile to the backend attribute table via UV coordinates. This facilitates the real-time reverse querying and extraction of detailed engineering data at that specific point, including lithology, stratigraphic age, and ore grade.

To further quantify the attribute querying performance of the proposed method, systematic tests were conducted across all four model scales (L1–L4). As reported in Table 1, the attribute

query response time of the proposed method remains below 23 μs even at the maximum scale of 10 million facets (L4), compared with approximately 21300 μs for the octree voxel method, representing a reduction of nearly three orders of magnitude. Regarding attribute mapping accuracy, the proposed method achieves 100% correctness across all tested scales, as each texel in the attribute-indexed texture stores a discrete, integer-encoded stratigraphic ID retrieved via strict nearest-neighbor filtering, which completely eliminates the interpolation-induced misclassification errors inherent in linear filtering. In contrast, the octree voxel method exhibits a progressive decline in accuracy from 91.3% at L1 to 85.6% at L4, attributable to the geometric approximation errors introduced by voxelization at stratigraphic boundaries. These results provide rigorous quantitative validation that the dual-layer "geometry–texture–attribute" mapping mechanism achieves both microsecond-level query responsiveness and lossless attribute fidelity, confirming that the proposed method resolves the "visible but unqueryable" bottleneck of conventional GPU-based slicing approaches. Ultimately, this approach resolves the performance bottleneck of "unsliceable" massive geological models and overcomes the functional limitation of traditional GPU slicing, which is typically "visible but unqueryable". This breakthrough achieves a paradigm shift from mere "visual observation" to interactive "quantitative analysis", demonstrating the method's technological superiority in geological engineering visualization.

4. Conclusions

To address the inherent dilemma of balancing the real-time rendering of massive geological data with precise deep-seated attribute querying in smart mine construction, in this paper, we proposed a lightweight, real-time slicing method for 3D geological bodies tailored to engineering visualization. By integrating GPU graphics pipeline programming with texture mapping techniques, we constructed a highly efficient slicing architecture based on pixel masking and attribute-indexed textures. By transforming complex 3D geometric intersections into 2D pixel-level determinations, the algorithm completely decoupled the computational load from the model scale. Experimental results demonstrated that for complex models comprising up to 10 million facets, the single-slice response time remains stable between 1.2 and 1.6 ms, sustaining an interactive frame rate of approximately 60 FPS, significantly outperforming traditional geometric intersection methods. Furthermore, we developed a dual-layer "geometry-texture-attribute" mapping mechanism utilizing lightweight attribute-indexed textures to replace conventional vertex attribute storage. Requiring only one-fifth of the VRAM consumed by voxel-based methods, this mechanism overcame the loss of geological semantics inherent in traditional rendering-based slicing, thereby enabling millisecond-level attribute reverse querying at any cross-sectional location. Crucially, the proposed method eliminated the need for complex voxelization preprocessing, effectively avoiding the "staircase aliasing effect" and geometric distortion at orebody boundaries. Its WebGL-based implementation facilitated seamless cross-platform deployment, providing highly efficient and high-fidelity visualization support for the development of smart mine digital twin systems. Nevertheless, certain limitations of the proposed method should be acknowledged. The pixel masking strategy produced visually

precise cross sections but did not reconstruct explicit closed contour geometry, limiting its direct applicability to downstream workflows such as reserve volume computation and finite element mesh generation. Additionally, UV parameterization was computed as a static offline process, meaning that dynamic updates to the geological model required the re-baking of the index texture, which has not yet been automated. The current experimental validation was also confined to a single silica sandstone mining site, and generalization to more complex geological settings—such as metallic ore deposits or karst terrains—warrants further investigation. In future work, we plan to explore lightweight contour extraction integrated into the rendering pipeline, incremental texture re-baking for dynamic model updates, and systematic cross-domain validation across diverse geological environments.

Acknowledgments

This work was supported by the Key Technology and Application Research on Multi-source Stereo Spatiotemporal Information Integrated Management (No. KJ2025-02) and the Research and Demonstration of Full-process Information Technology Support System and Key Technologies for County-level Land Change Survey (No. SCIGS-CZDXM-2025009).

References

1 Q. Di, G. Xue, D. Lei, Q. Zeng, C. Fu, and Z. An: Sci. China Earth Sci. **64** (2021) 1524. https://doi.org/10.1007/s11430-020-9818-2
2 Q. Wu and H. Xu: Sci. China Earth Sci. **57** (2014) 491. https://doi.org/10.1007/s11430-013-4671-9
3 F. Wellmann and G. Caumon: Adv. Geophys. **59** (2018) 1. https://doi.org/10.1016/bs.agph.2018.09.001
4 M. W. Jessell, L. Ailleres, E. A. de Kemp, M. Lindsay, F. Wellmann, M. Hillier, G. Laurent, T. Carmichael, and R. Martin: Soc. Econ. Geol. Spec. Publ. **18** (2014) 261. https://doi.org/10.5382/SP.18.13
5 L. X. Wu: Comput. Geosci. **30** (2004) 405. https://doi.org/10.1016/j.cageo.2003.06.005
6 G. Caumon, P. Collon-Drouaillet, C. Le Carlier de Veslud, S. Viseur, and J. Sausse: Math. Geosci. **41** (2009) 927. https://doi.org/10.1007/s11004-009-9244-2
7 T. Hinks, H. Carr, L. Truong, and D. F. Laefer: Comput. Geosci. **56** (2013) 25. https://doi.org/10.1016/j.cageo.2013.02.008
8 W. E. Lorensen and H. E. Cline: ACM SIGGRAPH Comput. Graph. **21** (1987) 163. https://doi.org/10.1145/37401.37422
9 S. W. Houlding: 3D Geoscience Modeling: Computer Techniques for Geological Characterization (Springer-Verlag, Berlin, 1994). https://doi.org/10.1007/978-3-642-79012-6
10 Y. Deng, J. Zhang, Y. Sun, Y. Tian, Q. Chen, and B. Qiu: Earth Sci. Inform. **17** (2024) 441. https://doi.org/10.1007/s12145-023-01180-8
11 S. Li, Z. Qin, Y. Ding, and M. Wang: Int. J. Digit. Earth **18** (2025) 2501770. https://doi.org/10.1080/17538947.2025.2501770
12 C. Shi and Y. Wang: Tunnelling Undergr. Space Technol. **126** (2022) 104493. https://doi.org/10.1016/j.tust.2022.104493
13 R. Sicat, M. Ibrahim, A. Ageeli, F. Mannuss, P. Rautek, and M. Hadwiger: IEEE Trans. Vis. Comput. Graph. **29** (2023) 1491. https://doi.org/10.1109/TVCG.2021.3120372
14 H. Zhu, J. Jin, and S. Zhao: Sensors **25** (2025) 6153. https://doi.org/10.3390/s25196153
15 M. Hillier, F. Wellmann, E. A. de Kemp, B. Brodaric, E. Schetselaar, and K. Bédard: Geosci. Model Dev. **16** (2023) 6987. https://doi.org/10.5194/gmd-16-6987-2023
16 X. Cao, Z. Liu, C. Hu, X. Song, J. A. Quaye, and N. Lu: Minerals **14** (2024) 686. https://doi.org/10.3390/min14070686
17 N. Qu, Y. Yan, T. Cheng, T. Li, and Y. Wang: Mob. Inf. Syst. **2022** (2022) 8472472. https://doi.org/10.1155/2022/8472472