Eurographics Workshop on Visual Computing for Biology and Medicine (VCBM) (2026)
Robert Krueger, Eric Mörth, and Katarína Furmanová (Editors)

Conference Paper

Coupled Visualization-Quantification Pipeline for Time-Resolved Cement Injection Imaging

I. Sharma[1], J.-S. L. Völter[2], D. Gehweiler[3], O. Röhrle[2,4] and D. Schmalstieg[1]

[1]Visualisierungsinstitut, University of Stuttgart, Germany
[2]Institute for Modelling and Simulation of Biomechanical Systems, University of Stuttgart, Germany
[3]AO Research Institute Davos, Switzerland
[4]Stuttgart Center for Simulation Science, University of Stuttgart, Germany

Figure 1: *Time-series visualization of cement injection during vertebroplasty, with cement volume label and quantification parameter knobs.*

Abstract

In image-guided medical procedures, quantitative measurements may inform irreversible decisions. One such procedure is percutaneous vertebroplasty, in which experts must precisely monitor both the location and volume of the injected cement deposit. Domain scientists routinely use CT imaging to study cement flow, aiming for safer clinical interventions. However, current monitoring solutions can exhibit severe drift. Visual renderings of the volume and volumetric measurements may disagree substantially in both space (due to competing ways of processing the volume) and time (due to computational latency). We present a prototype software application that improves the current status quo by introducing an execution model of shared classification between the volumetric renderer and the volumetric measurement, so the measured and displayed regions coincide. We demonstrate through an empirical evaluation on a cadaveric vertebra and on aluminum foam phantoms that our method delivers results within tight bounds established by a Verification, Validation and Uncertainty Quantification (VVUQ) protocol.

Keywords: Software and its engineering → Requirements analysis

1. Introduction

The dynamic characterization of fluid progression within porous media is a fundamental challenge across experimental domains from geophysics to biomedical engineering. Our work arose from a quantitative bottleneck encountered by domain scientists studying cement injection into bone (percutaneous vertebroplasty) via time-resolved, four-dimensional (4D) X-ray computed tomography. To assess procedural efficacy and manage clinical risk, one must con-

tinuously track, segment, and quantify the physical volume of injected cement as it perfuses the target site.

The software platform used routinely by our collaborating vertebroplasty experts is Amira (Thermo Fisher Scientific), a proprietary research visualization platform. When planned research required access to the source and extensibility, we turned to ParaView (Kitware), which is increasingly adopted as an open-source visualization front-end for 3D biomedical research [LWP*25, PIG*16]. Our empirical assessment revealed that both approaches support interactive exploration and temporal quantification individually, but exhibit systematic limitations once the two are required simultaneously, as they are in percutaneous vertebroplasty. Apart from user interface challenges, the pairing came apart in two ways: Spatially, a reported volume could shift substantially while the rendered image remained fixed. Temporally, interaction stuttered and playback hung intermittently. The interconnected nature of these problems motivated us to adopt an approach that relies on a coherent spatiotemporal representation.

Our central thesis is that in general-purpose visualization data-flow models, rendering and numerical analysis are decoupled as a consequence of the modular pipelines within otherwise monolithic multi-purpose systems. In procedures that demand a tighter bound on all workflow classification of data, such as during a synchronized, interactive 4D quantification task, such systems might exhibit structural misalignment with the requirements of the task. Their explicit synchronization mechanisms may work to their disadvantage, not because correct correspondence is unreachable in such systems, but because maintaining it depends on human vigilance rather than on the architecture itself. We address such requirements with the scene-graph library of OSPRay Studio [SDH*21]. Its frontend is light enough for targeted development, and its directed acyclic graph (DAG) of nodes keeps all data for a sequence unit in one place rather than distributed across pipeline objects. The hierarchy then supplies the structural properties we require: Quantification and rendering derive from the same classification; time integration is native through subtree switching rather than applied frame-by-frame via external parameters, and graph traversal maintains synchronization of classification parameters across functions and objects by construction, without explicit orchestration.

This work contributes a design rationale and a prototypical application arising from our study of the workflow challenges and interface limitations encountered when applying certain broader tools to a domain-specific task such as percutaneous vertebroplasty. It derives the foundational requirements for addressing the needs of this procedure and others similar to it, and presents the implementation details of the system along with a Verification, Validation and Uncertainty Quantification (VVUQ) analysis of its volume quantification error bounds. The result is intended to complement existing image-processing and visualization workflows rather than replace the established platforms, which remain the stronger option for general visualization and post hoc analysis of volumetric image data. Our platform trades that breadth for a degree of data integration the node structure affords, keeping synchronized 4D visualization and continuous quantification of cement injection within a single interface and enabling reliable interactive assessment of cement propagation during the injection phase itself. This functionality facilitates the objective evaluation of injection strategies, supports surgeon training through immediate visual and quantitative feedback, and provides experimentally derived data for the calibration and validation of computational models of cement flow. The main contributions of this paper can therefore be listed as follows:

- We evaluate two established visualization platforms, one commercial and one open-source, to characterize their practical limitations in simultaneous 4D fluid volumetry and interactive time-series exploration.
- We trace these limitations to a set of core architectural and functional requirements, and provide a system implementation that addresses them.
- We provide volume quantification of cement in percutaneous vertebroplasty with validated error bounds as an integrated component of this system.

2. Related Work

2.1. Time-Varying Volume Visualization

Coherent rendering of time-resolved volumes has a long history, Ma [Ma03] and Bai et al. [BTL20] provide surveys on this topic. Shen et al. [SCM99] introduced the time-space partitioning tree, and Tikhonova et al. [TCM10] proposed coherent compression for exploratory visualization, both exploiting temporal coherence to avoid holding the full sequence in memory. These methods target datasets too large to keep resident in memory and devote their efforts to compression and partial loading. Orthogonal to the form in which a sequence is stored, each volume must be classified for display with a transfer function. That brings us to the design of transfer functions, which is a deep subject in its own right [KKH02]. Its extension to time-varying data, where a function tuned for one volume does not transfer cleanly to the next, is a recognized open problem [JKM01, LKG*16, NSHAS24]. Stable cross-frame classification is available in a few deployed tools: ParaView [Aya15a], can lock a colormap's range across an entire timeline. While such consistency serves visual comparison across timesteps, it does not guarantee that the rendered region and the measured one coincide.

2.2. Quantitative Volumetry and Boundary Artifacts

Extracting a volume from a scalar field requires committing to a boundary, and CT-based cement quantification rests on a few established methods: Semi-automated thresholding for

fill and leakage [MRF*03], density-weighted sub-voxel estimation that softens partial-volume mis-classification introduced by a hard threshold at material interfaces [LVW*05], and cohort-scale correlation of the resulting numbers with outcomes [NBvED12]. All of these approaches are intended for retrospective visualization: a single post-procedure scan, one threshold, no requirement that the classification be coherent over time, and no coupling to a live display. The work of Trivedi et al. [TGW*23] is closer to the dynamic case. They acquired an injection-phase cine and paired it with a porous media simulation, reporting per-scenario measurements. Another work correlates plunger spread [LFNK08] to narrow the intraoperative information gap, but without providing a 3D solution. In neither case is the reported quantity tied to the display through which the injection is observed.

2.3. Visualization Pipeline Software Architectures

Visualization systems have historically been constructed in multiple ways, for example, OSPRay Studio [SDH*21] represents the entire scene as a directed acyclic graph: Nodes hold retained scene state, and operations are implemented as visitors traversing this graph. All operations consult a single live representation of the scene, and a state change invokes a commit chain along the affected sub-region only. ParaView [Aya15a] and VisIt [CBW*12] follow the demand-driven data-flow model [HM90, SML06]: Data streams from sources through filters to mappers and renderers, and each timestep change re-executes this pipeline. Under this model, a derived quantity, computed, for instance, by a *threshold* → *integrate* filter, occupies a branch parallel to the volume-rendering branch that classifies the displayed data; the measured and displayed regions share the data but not the classification, and keeping them in agreement requires explicit effort and considerable familiarity with the internals of VTK. 3D Slicer [FBKC*12] and Amira [SWH05a], though not pipeline systems, show a similar decoupling: Slicer's SequenceBrowser keeps per-timestep data copies consistent by propagating display parameters through proxy nodes, a synchronization that a shared graph would not require, and Amira reads quantities from static, per-timestep segmentation artifacts, detached from the live volume classification. Section 3 quantifies the cost of this decoupling in ParaView and Amira.

3. Limitations Derivation and Design Requirements

3.1. Experimental Setup

We initially conducted the vertebroplasty analysis using two widely adopted applications, Amira 6.4.0 [SWH05a] and ParaView 6.1.1 [SML06]; the specific limitations we encountered in both, motivated the design of our system. Amira is a mature commercial platform for 3D/4D visualization and quantitative analysis, applied to visualize dynamic CT of bone structures and biomaterials [The17, SWH05b] regularly; ParaView is Kitware's flagship VTK application and should be representative of the intended use of VTK in interactive visualization of large volumetric and time-resolved biomedical data [AGL05, Aya15b, ZFH*22].

In both applications, measurement tends to sit apart from the rendering path, for example, a standalone closed-source spreadsheet export, or per-path classifications computed over timesteps that are loaded independently. We use *decoupled* strictly in this sense (two independently writable classification states for rendering and quantification), whereas *coupled* denotes a single state read by both. The distinction concerns state rather than where computation runs, and their agreement rests on explicit syncing of the former. The limitations noted below are therefore rooted in this dual-classification model for our use case, over independent breadth of either tool. We provide reasoning for the derivation of our design choices through this focused evaluation on the two applications that offer a reasonable coverage of current practices.

All numbers provided in this section were obtained on a mobile workstation (Intel Core i9 14th-generation CPU, NVIDIA RTX 4090 Laptop GPU, 16 GB VRAM). The test cine comprises 52 timesteps of a controlled 2 ml injection of bone cement into a cadaveric lumbar vertebra (VERTEBRA). Framerate and execution performance is evaluated from memory resident timestep data and warm cache for every frame. Hence, the reported latencies reflect pipeline and traversal cost alone, with rendering fixed at 720p.

Amira. The volume measurement logic of Amira is proprietary. Rendering and measurement occupy separate branches of its project graph and are independently configurable. Slice views, surface rendering, and direct volume rendering (DVR) may be combined freely, with DVR conveying the bolus and surrounding trabecular structure with greater morphological fidelity. However, measurement involves segmentation-based surface extraction of the bolus whose geometry is visibly smoothed, and the final value is mutually exclusive of the choice of bolus rendering. The segmented label field is converted to a triangulated mesh, and the enclosed volume of that mesh is then quantified and reported alongside its surface area. In the final frame of the injection sequence, the cement volume measurement was reported at 3.04 ml, an overestimate of 52% over injected volume. The derived values populate an independent spreadsheet view that can be reached in a couple of clicks. This style appears to be suitable for retrospective analysis due to the lack of a playback measurement label. Rendering remains interactive at approximately 30 fps using their native closed source renderer.

ParaView. ParaView supports quantification through filter pipelines: A thresholded region can be integrated (*Threshold* → *Integrate Variables*) and swept across the series (*Plot Data Over Time*). However, the operative counting thresh-

old τ (the value T_{quant} supplied to the quantification filter) is configured independently of the threshold T_{vis} used for volume rendering, so a divergence between the value shown and measured can go unnoticed. Table 1 illustrates this in our 2 ml experiment: Sweeping $\tau \in \{600, 700, 800\}$ HU, while the rendering remains fixed at $T_{\text{vis}} = 700$ HU, changes the reported volume from 4.80 to 2.65 ml, with no change in the displayed region. Even in the coincident setting ($\tau = T_{\text{vis}} = 700$ HU), the two methods report different volumes (3.55 vs. 2.49 ml), because the default *Threshold* filter admits or rejects whole voxels, and the inclusion criterion can be chosen to make this boundary more erosive or dilative. Partial occupancy requires a separate filter that interpolates a cut at the threshold value, but, since its boundary fractions are anchored to τ itself, displacing the threshold then displaces every boundary fraction, the reported volume continues to track τ. Before experimenting with custom partial-volume weighting scripts, we gauged the temporal behavior, and, in our experiments, that proved to be a considerable limitation. On every scrub tick, the pipeline re-executes ($\sim$45 ms cold, $\sim$ 14-22 ms per re-execution, $\sim$8.5 ms with a warm cache) and the frame must then be rendered. The rendering throughput is $\sim$20 -30 fps with native GPU rendering; but the combination of playback with simultaneous data exploration like zoom or rotate produced flicker and stalls, making it impractical in our tests.

Table 1: *Volumetric quantification sensitivity over a* $\pm$100 *HU threshold window (*VERTEBRA *dataset, against a* 2 *ml control injection).*

Threshold τ (HU)	**ParaView** (Swept T_{quant}, fixed T_{vis})	**Our Method** (Coupled $T_{\text{quant}} \equiv T_{\text{vis}}$)
600 ($\Delta\tau = -100$)	4.80 ml	2.54 ml
700 (Nominal onset T_{vis})	3.55 ml	2.49 ml
800 ($\Delta\tau = +100$)	2.65 ml	2.44 ml
Spread over $\pm$100	2.15 ml ($\approx 60\%$)	0.10 ml ($\approx 4\%$)
Error vs. 2 ml control	+78%	+25%

3.2. Design Requirements

From our experiments, we distilled five requirements that define the basis of our system:

1. **Display and Measurement Correspondence.** A single classification state must govern both quantification and rendering. By strategic topological construction, a common classification layer on top of the timestep data can be the single point of entry for different task logic.
2. **Temporal Quantification State.** Quantification must reside as a mutable state rather than an artifact of a calculation and is re-evaluated on threshold changes through a global mechanism at every timestep as the time-varying data advances.
3. **Interactive Scrubbing with Co-Located Readout.** Timestep transitions must lower execution overhead per tick, having a low data-ready latency and stutter-free real-time scrubbing. The per-timestep measurement must be surfaced in the same interface that drives playback, rather than on a second path such as a separate pipeline branch, a spreadsheet view, or an offline scripted pass.
4. **Inspectability, Morphological Fidelity, and Rendering Throughput.** The classification threshold, the partial-volume model, and the per-specimen calibration constants must be explicit, user-visible parameters of an open-source implementation; at the same time, the pipeline must preserve morphological fidelity and sustain interactive frame rates ($\approx 16-18$ fps [ION23]).
5. **Partial-Volume Boundary Treatment.** Boundary voxels must contribute fractionally via native partial-volume (α-weighted) accumulation, yielding threshold-stable measurements.

4. System Design

Our framework is built on OSPRay Studio, whose application state is a single scene graph—a DAG whose root is a *Frame* node with *Camera*, *FrameBuffer*, *Renderer*, and *World* as its children. Volumes, transforms, transfer functions, and metadata scalar parameters can be added as nodes under the *World*; *visitors* implement task specific requirement and traverse the graph. Shared pointer semantics manage node copies and a node placed above the timestep subtrees is inherited by every timestep. State written onto a node persists with the graph, and an edit propagates to all dependent subtrees through the standard commit mechanism. Our design adds the following components on top of OSPRay Studio to realize our system: First, a **4D workflow derived topology** (Fig. 2), where timesteps are loaded as self-contained subtrees with a shared transfer function under the world node. Second, two importers for **Amira and TIFF** timeseries files, that parse each timestep into a host-resident 16-bit voxel buffer owned by the volume node shared with the renderer's volume backend. Third, an *additive* **quantification visitor**, *QuantifyCement* computes the per-frame cement signal under the classification the renderer applies while leaving the rendering path untouched. Fourth, a **cement volume label and quantification parameter knobs** integrated with native ImGui animation widget and manager, that provide current measurements.

4.1. Display and Measurement Correspondence

Two thresholds are implicit in any quantifying display: An opacity onset T_{vis} fixes the rendered region, and a counting threshold T_{quant}, which fixes the integrated one. The counting threshold in our system $\tau = T_{\text{quant}}$, by default, is not a free parameter of the quantification but is derived from the

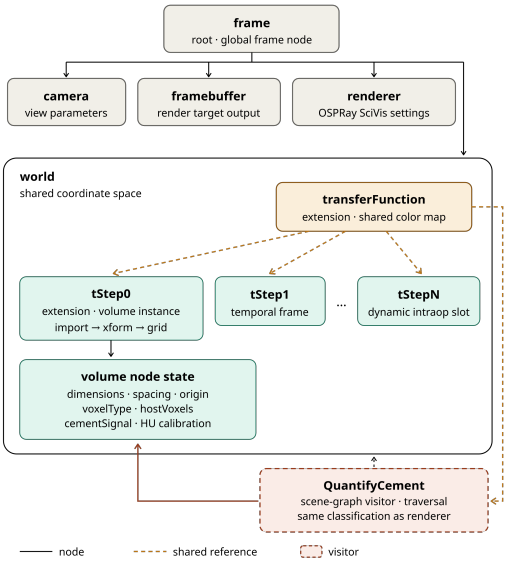


Figure 2: *DAG topology for our system including the quantification visitor as a flow diagram*

transfer function in scope for the volume. Since it classifies through the sampled opacity array $\{\omega(s_k)\}$, we define the onset at that resolution as $T_{\text{vis}} = \min\{s_k : \omega(s_k) > 0\}$, where the sample positions $s_k = v_{\min} + (v_{\max} - v_{\min})\,\frac{k}{N-1}$ are uniform over the value range. This is the coupled setting with τ as the derived threshold that agrees with the rendered classification, enforcing $T_{\text{quant}}(t) = T_{\text{vis}}(t)$. Quantification at the pre-injection timestep 0 measures the static anatomy above τ; the dense cement injected later is the per-timestep delta.

Traversals corresponding to rendering and quantification resolve the shared transfer function by similar traversal logic: The *RenderScene* visitor maintains a LIFO stack of transfer-function nodes, pushed on descent and popped on ascent, so each node scopes over its own subtree; *QuantifyCement* mirrors this resolution without modifying the rendering path. However, the transfer function sits above the timestep subtrees in our DAG topology rather than within each volume node. This enables every volume to inherit from the same classification, and the visitor derives the counting threshold τ. Editing the opacity ramp therefore moves the rendered and the measured regions together. Such a guarantee matters in the injection-phase regime this system targets, where attention is on the scene rather than the parameter panel and a silently diverged threshold might propagate into a different clinical judgment.

4.2. Temporal Quantification State

A 4D study yields a sequence of measurements, one per timestep, and the interesting design question is where the values live post calculation rather than the calculation itself. If they are emitted at import or dumped by a script, measurements are frozen artifacts. If the classification changes, they silently describe a scene that no longer exists. Our solution to avoid such inconsistencies is to make the scene graph itself the measurement store. Each frame's quantification value is materialized on the timestep sub-tree of DAG it describes and tagged with the classification it was computed under, so validity is a local, checkable property, and consistency across the whole sequence is restored by re-traversing the graph rather than by re-running a pipeline. The lifetime of a measurement is thereby tied to the lifetime of the scene, not to the moment of its computation.

Concretely, the visitor calculates the absolute cement signal and the threshold it was computed under and writes each timestep's result back onto its volume node as two scalar children. It holds no values of its own and adds both quantities as scene-graph-only, so they persist without leaking into the render backend. A traversal at loading time populates this state for every timestep, so the measurement exists for the whole sequence before playback begins. When the shared transfer function is edited, the current threshold is derived at the edit callback, where the sampled opacities are guaranteed to be current, and a re-traversal carries it to every timestep. At each volume, the visitor compares the incoming τ against the cached one, recounts frames with stale state, and skips fresh ones without re-entering the voxel loop.

4.3. Interactive Scrubbing with Co-Located Readout

Each timestep is parsed once at load into its host-resident voxel buffer, shared with OSPRay renderer [WJA*17] and no voxel is copied at playback time. We extend the animation manager, originally oriented toward glTF-style keyframed animation, to select timestep subtrees instead. On each tick, the active subtree is hot-swapped below the world node, detaching the previous and attaching the current one, and the scene is refreshed through the commit mechanism, while camera, renderer, and framebuffer state are unchanged. A transition is therefore a vector lookup followed by a reattach, which makes forward and reverse playback, and random-access scrubbing the same operation—an index into the resident timestep map, with data-ready latency independent of data size. Reading the quantification values resident within volume nodes, adds nothing to this cost. The measurement is memoized per frame and keyed on its threshold (Sec. 4.2), so an ordinary swap reads a single cached float in O(1), and only a transfer-function edit, an interaction-rate event off the hot path, triggers a recount, with color-only edits leaving the opacity onset and hence τ untouched. No measurement code executes inside the render loop, so frame rate during rotation, panning, and scrubbing is set by the renderer alone. The cached value is routed to the cement volume label on the animation widget's panel, so navigation and quantitative readout occupy one surface.

4.4. Inspectability and Morphological Fidelity

All parameters entering the volume estimate are exposed knobs, and the provenance of the delimiting threshold is always explicit. The per-specimen calibration constants $\mathrm{HU}_{\mathrm{bg}}$ and $\mathrm{HU}_{\mathrm{cement}}$, which determine the fractional occupancy α_j in Eq. (1), are passed to the quantification visitor as explicit arguments, whose value is provided by editable parameters in the UI. They calibrate material fraction, but do not delimit the bolus. The threshold τ supplies that delimitation, and can be set in two ways: It is derived from the onset of opacity of the shared transfer function (the default coupled setting) or fixed explicitly through an override parameter. τ configured independently of the opacity map incurs the cost of $T_{\mathrm{quant}}(t) \neq T_{\mathrm{vis}}(t)$, but allows our system to be open to different look-dev options for the data. Morphological fidelity follows from rendering the classified volume directly with OSPRay's SciVis renderer, without an intermediate surface extraction. Fine structures on the cement progression front is therefore not smoothed or decimated by mesh generation.

4.5. Boundary-Voxel Treatment

At the voxel scale, a binary inclusion test at the interface between cement and tissue can increase the sensitivity of a reported volume to its threshold, because every interface voxel flips between contributing maximum or nothing. Our counting loop separates interior from surface voxels with a 6-neighborhood test: A voxel above τ, whose six face-neighbors all exceed τ, counts as fully occupied. A voxel with at least one sub-threshold neighbor contributes a fractional occupancy

$$\alpha_j \;=\; \mathrm{clamp}\!\left(\frac{\mathrm{HU}_j - \mathrm{HU}_{\mathrm{bg}}}{\mathrm{HU}_{\mathrm{cement}} - \mathrm{HU}_{\mathrm{bg}}}, 0, 1\right), \tag{1}$$

where $\mathrm{HU}_{\mathrm{bg}}$ and $\mathrm{HU}_{\mathrm{cement}}$ are the per-specimen calibration constants passed to the visitor explicitly. The measured volume is $(\text{interior count} + \sum_j \alpha_j)$ scaled by the physical voxel volume implied by the grid spacing.

A surface voxel's contribution as τ moves is now considered at smaller steps of α_j rather than one. This is the mechanism behind the $\approx 4\%$ spread in Table 1. The total measurement changes are much less prone to discontinuities, as the voxels well inside the cement have α_j close to one and the correction acts on the voxels only above τ, which sets a hard limit on what is measured. The family of volume models built on Eq. (1) and this split between interior and surface voxels, together with the specimen-internal derivation of the calibration constants, follow in Sec. 5.

5. Quantification Method and Validation

Our quantification method applies a partial-volume model with fractional voxel occupancies, and is implemented as a graph traversal visitor. For each timestep i, voxels exceeding the threshold τ are classified via a 6-connected neighborhood into *interior* voxels (n_{int}: all six face-neighbors exceed τ) and *surface* voxels (n_{surf}: at least one neighbor falls below τ).

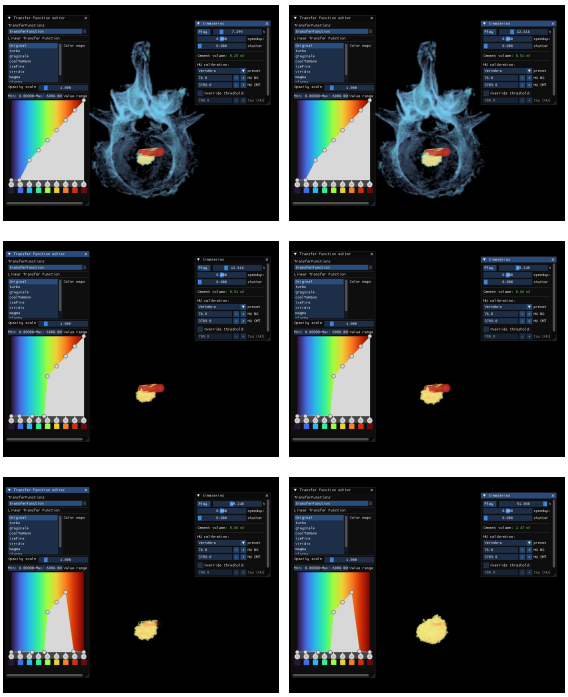


Figure 3: *Propagation of transfer-function updates across timesteps. Row 2 shows adjustment to isolate the cement lobe and syringe; row 3 further refines the transfer function to display only the cement.*

$$V_{\mathrm{calib}}(t_i) = \left[n_{\mathrm{int}}(t_i) + \sum_{j \in S} \alpha_j(t_i)\right] v_{\mathrm{vox}}, \tag{2}$$

where S is the set of surface voxels, v_{vox} is the nominal voxel volume, and $\alpha_j \in [0,1]$ is the per-voxel partial-volume cement fraction given by Eq 1. We have implemented two other models for comparison:

$$V_{\mathrm{raw}}(t_i) = n_{\mathrm{above}}(t_i) \cdot v_{\mathrm{vox}}, \tag{3}$$

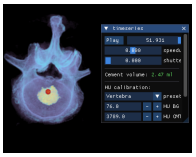
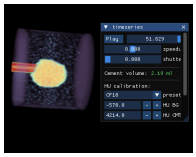
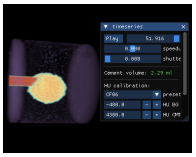
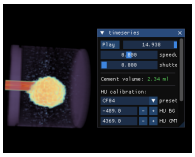
$$V_{\mathrm{unif}}(t_i) = \left[n_{\mathrm{int}}(t_i) + \tfrac{1}{2} n_{\mathrm{surf}}(t_i)\right] v_{\mathrm{vox}}, \tag{4}$$

While V_{raw} and V_{unif} require no parameters beyond τ, the partial-volume corrected volume (V_{calib}) utilizes two measured, specimen-specific variables extracted directly from each specimen's own data stream: $\mathrm{HU}_{\mathrm{bg}}$ (from the pre-injection frame 0) and $\mathrm{HU}_{\mathrm{cement}}$ (from the plateau frame).

5.1. Datasets and Experimental Framework

The volume models described earlier are evaluated on the VERTEBRA dataset with matched pre- and post-injection micro-CT (Figure 4) and three aluminum-foam phantoms, all imaged on a GE Revolution EVO in cine mode during

Table 2: *Unified Specimen Dashboard: From Physical and Acquisition Properties to Volumetric Validation*

	Biological Cohort	**Porous-Media Vertebral Phantoms [TGW*23]**		
Property / Volumetric Metric	VERTEBRA	CF10	CF06	CF04
Specimen end timestep renders with PVC-Calibrated model				
Voxel Volume (v_{vox}, mm^3)	0.149 (Coarse)	0.149 (Coarse)	0.025 (Fine)	0.025 (Fine)
Injection Flow Rate (ml $\cdot$ s^{-1})	0.1	0.1	0.1	0.4 (4$\times$ flow variant)
Number of Timesteps (n_t)	52	52	52	15
Background Intensity (HU$_{\text{bg}}$)	+76	−570	−480	−489
Cement Intensity (HU$_{\text{cement}}$)	3,789	4,214	4,380	4,369
Reference Volume (ml)	2.298 (μCT)	2.00 (nom.)	2.00 (nom.)	2.00 (nom.)
Raw Volume (V_{raw}, ml)	2.70 [+17.5%]	2.35 [+17.5%]	2.45 [+22.5%]	2.49 [+24.5%]
PVC-Uniform (V_{unif}, ml)	2.54 [+10.5%]	2.20 [+10.0%]	2.32 [+16.0%]	2.36 [+18.0%]
PVC-Calibrated (V_{calib}, ml)	**2.48** [+**7.9**%]	**2.19** [+**9.5**%]	**2.29** [+**14.5**%]	**2.34** [+**17.0**%]
Simulation Endpoint (ml)	-	-	1.9065 [−4.7%]	1.9065 [−4.7%]

Note: All errors are signed deviations from each specimen's own reference volume (positive = overestimate). VERTEBRA is referenced to an independent micro-CT measurement ($V_{\mu\text{CT}} = 2.298$ ml) rather than the nominal 2.00 ml injection: in biological tissue, cement extravasation means the injected volume is not a valid proxy for the volume retained in the vertebral body. The rigid foam phantoms retain essentially all injected cement, so the nominal 2.00 ml is used directly. Simulation endpoints are from matched continuum Cannella-model runs.

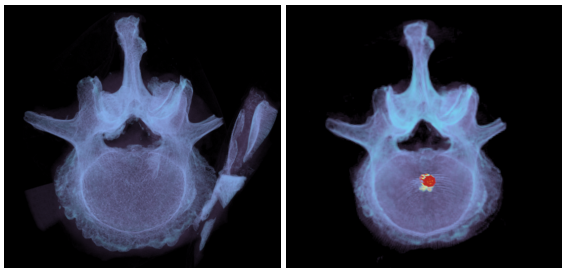


Figure 4: *Micro-CT vs clinical CT comparison. The 60 µm micro-CT resolves fine trabecular bone structures that are blurred by partial volume effects at anisotropic* $0.488 \times 0.488 \times 0.625$ *mm clinical voxel resolution*

injection of 2.00 ml PMMA cement. Table 2 reports the results alongside their captured voxel resolution, injection flow rate and number of timesteps at that rate, and the independently calibrated HU thresholds. The vertebral and phantom specimens differ in three primary ways: first, the rigid aluminum foam matrix exhibits higher attenuation (HU) than trabecular bone, hence necessitating HU calibration of our partial-volume model; second, the interstitial space is filled with air in the phantoms versus marrow, blood, and fluids in the bone; and third, cement simply displaces low-attenuating air in the phantoms, whereas in bone it displaces and interacts with viscous marrow at the injection front. Consequently, bone segmentation exhibits lower contrast, which motivates the use of high-resolution micro-CT to establish the baseline for comparison.

5.2. Quantification Validation

To evaluate the cement quantification method ($\Delta(t) \equiv 0$) and bound its error under realistic operational stress, we adopt a *five-vector VVUQ protocol*: a domain-specific instance of the verification, validation, and uncertainty-quantification paradigm used to establish computational-model credibility [ASM18, OR10]. Each vector pairs an empirical measurement with an *independent* physical or theoretical bound, and the estimator is accepted when the worst-case normalized margin is minimized and remains non-negative (please see the supplementary material on VVUQ). The vectors span orthogonal failure modes of a volumetric estimator: (1) accuracy against a gold-standard reference, (2) activeness of the calibration components, (3) robustness across operating regimes, (4) consistency with independent imaging and simulation modalities, and (5) stability under parameter perturbation. The result and details of acceptance criterion of these vectors are consolidated in Table 3.

6. Evaluation and Conclusion

Three domain experts assessed the delivered system along three dimensions:

Usability. Rated easy to use: a single executable with direct mouse navigation, and the volume read-out updates in the same view as the rendering.

Learning curve. Short: interaction follows conventions the experts already knew, so no instruction was needed to operate the widgets.

Integration. Minimal: their existing Amira sequences are used without preprocessing, and provided volume estimate was closer to baseline than their existing solution.

Table 3: *Five-vector VVUQ protocol of the quantification estimator. pp denotes percentage points.*

Validation vector	Measured result	Acceptance criterion
Geometric error bounds	+7.9% deviation from the micro-CT reference $V_{\mu CT}$ (measured 2.48 ml)	Error should remain below the ∼18% partial-volume uncertainty floor of a single-voxel shell at 488 μm resolution
Algorithmic ablation	3.3 pp error reduction when moving from the uniform baseline (PVC-uniform) to full physical calibration (PVC-calibrated)	Calibration should reduce volumetric error monotonically across the testing matrix, confirming the physical constraints are numerically active
Operational scale invariance	2.3 pp estimation variance under a 4× injection-velocity divergence (CF06 vs. CF04)	The geometric estimate should remain stable across flow regimes, so a single calibration holds without mid-procedure retuning
Multi-modal bracketing	Estimator falls between clinical CT (+15.2% to +17.5%) and porous-media transport simulation (−4.7%)	Output should lie within the two-sided envelope set by two independent physical modalities
Stochastic sensitivity limits	±0.028 ml at 95% confidence (2.48 ml, [2.44, 2.55] ml); metric shift below 2.4 pp	Monte Carlo variation of the calibration parameters should shift the result by less than the scanner-limited tolerance

In the experts' view, our solution has relevant use in vertebroplasty research and surgeon training rather than live intervention, which today is mostly guided by biplanar fluoroscopy [KKA*96, NKO*22].

In summary, we presented a scene-graph based system for exploring and quantifying time-resolved cement flow, built for research in vertebroplasty and similar procedures. We evaluated two well-adopted tools that provide interactive display and bounded measurement based on varying degree of configuration and based the delta of our requirements to the available features in these broader tools. A central decision involves one classification, one measurement state, and one resident copy of the sequence live in a single graph, so correspondence between image and number, re-evaluation on edits, and instant scrubbing follow from the topology rather than from added machinery kept in step by convention.

Outlook. A potential next step for this system is an estimator which uses machine learning, but its dependence on an internal representation decoupled from the rendered scene will reintroduce the very split this design removes, since training datasets will have limited scope of classification, and the resulting errors will be hard to bound outside its training distribution. Nevertheless, learning can augment the measurement rather than replacing it in future, as in denoising or extravasation alerting.

Acknowledgments

The donor of the human cadaveric specimen gave informed consent within the donation of anatomical gift statement during lifetime. This work was supported by the Alexander von Humboldt Foundation sponsored by the German Ministry of Research, Technology and Space.

References

[AGL05] AHRENS J., GEVECI B., LAW C.: Paraview: An end-user tool for large data visualization. In *The Visualization Handbook*, Hansen C. D., Johnson C. R., (Eds.). Elsevier, Amsterdam, 2005, ch. 36, pp. 717–731. doi:10.1016/B978-012387582-2/50038-1. 3

[ASM18] ASME: *V&V 40-2018: Assessing Credibility of Computational Modeling through Verification and Validation: Application to Medical Devices*. American Society of Mechanical Engineers, New York, NY, 2018. Standard No. ASME V&V 40-2018. 7

[Aya15a] AYACHIT U.: *The ParaView Guide: A Parallel Visualization Application*. Kitware, Inc., 2015. 2, 3

[Aya15b] AYACHIT U.: *The ParaView Guide: A Parallel Visualization Application*. Kitware, Inc., Clifton Park, NY, USA, 2015. URL: https://www.paraview.org/paraview-guide/. 3

[BTL20] BAI Z., TAO Y., LIN H.: Time-varying volume visualization: A survey. *Journal of Visualization 23*, 5 (2020), 745–761. doi:10.1007/s12650-020-00654-x. 2

[CBW*12] CHILDS H., BRUGGER E., WHITLOCK B., MEREDITH J., AHERN S., PUGMIRE D., ET AL.: VisIt: An end-user tool for visualizing and analyzing very large data. In *High Performance Visualization - Enabling Extreme-Scale Scientific Insight*. Chapman and Hall/CRC, 2012, pp. 357–372. 3

[FBKC*12] FEDOROV A., BEICHEL R., KALPATHY-CRAMER J., FINET J., FILLION-ROBIN J.-C., PUJOL S., ET AL.: 3D slicer as an image computing platform for the quantitative imaging network. *Magnetic Resonance Imaging 30*, 9 (2012), 1323–1341. doi:10.1016/j.mri.2012.05.001. 3

[HM90] HABER R. B., MCNABB D. A.: Visualization idioms:

A conceptual model for scientific visualization systems. In *Visualization in Scientific Computing*, Nielson G. M., Shriver B., Rosenblum L. J., (Eds.). IEEE Computer Society Press, Washington, DC, 1990, pp. 74–93. 3

[ION23] IONOS DIGITAL GUIDE: Frames per second (fps) in tv, cinema, and gaming, January 2023. URL: https://www.ionos.com/digitalguide/server/know-how/fps/. 4

[JKM01] JANKUN-KELLY T. J., MA K.-L.: A study of transfer function generation for time-varying volume data. In *Volume Graphics 2001: Proceedings of the Joint IEEE TCVG and Eurographics Workshop* (Vienna, 2001), Mueller K., Kaufman A. E., (Eds.), Springer, pp. 51–68. doi:10.1007/978-3-7091-6756-4_4. 2

[KKA*96] KATADA K., KATO R., ANNO H., OGURA Y., KOGA S., IDA Y., SATO M., NONOMURA K.: Guidance with real-time CT fluoroscopy: Early clinical experience. *Radiology 200*, 3 (1996), 851–856. doi:10.1148/radiology.200.3.8756943. 8

[KKH02] KNISS J., KINDLMANN G., HANSEN C.: Multidimensional transfer functions for interactive volume rendering. *IEEE Transactions on Visualization and Computer Graphics 8*, 3 (2002), 270–285. doi:10.1109/TVCG.2002.1021579. 2

[LFNK08] LOEFFEL M., FERGUSON S. J., NOLTE L.-P., KOWAL J. H.: Vertebroplasty: Experimental characterization of polymethylmethacrylate bone cement spreading as a function of viscosity, bone porosity, and flow rate. *Spine 33*, 12 (2008), 1352–1359. doi:10.1097/BRS.0b013e3181732aa9. 3

[LKG*16] LJUNG P., KRÜGER J., GRÖLLER E., HADWIGER M., HANSEN C. D., YNNERMAN A.: State of the art in transfer functions for direct volume rendering. *Computer Graphics Forum 35*, 3 (2016), 669–691. doi:10.1111/cgf.12934. 2

[LVW*05] LIBICHER M., VETTER M., WOLF I., NOELDGE G., KASPERK C., GRAFE I., ET AL.: CT volumetry of intravertebral cement after kyphoplasty: Comparison of polymethylmethacrylate and calcium phosphate in a 12-month follow-up. *European Radiology 15*, 8 (2005), 1544–1549. doi:10.1007/s00330-005-2709-x. 3

[LWP*25] LI Y., WEIGAND J. D., PUELZ C., OLUFSEN M. S., TAYLOR-LAPOLE A.: A one dimensional (1D) computational fluid dynamics study of fontan-associated liver disease (FALD), 2025. URL: https://doi.org/10.48550/arXiv.2501.12396, arXiv:2501.12396, doi:10.48550/arXiv.2501.12396. 2

[Ma03] MA K.-L.: Visualizing time-varying volume data. *Computing in Science & Engineering 5*, 2 (2003), 34–42. doi:10.1109/MCISE.2003.1182960. 2

[MRF*03] MOUSAVI P., ROTH S., FINKELSTEIN J., CHEUNG G., WHYNE C.: Volumetric quantification of cement leakage following percutaneous vertebroplasty in metastatic and osteoporotic vertebrae. *Journal of Neurosurgery: Spine 99*, 1 (2003), 56–59. doi:10.3171/spi.2003.99.1.0056. 3

[NBvED12] NIEUWENHUIJSE M. J., BOLLEN L., VAN ERKEL A. R., DIJKSTRA P. D. S.: Optimal intravertebral cement volume in percutaneous vertebroplasty for painful osteoporotic vertebral compression fractures. *Spine 37*, 20 (2012), 1747–1755. doi:10.1097/BRS.0b013e318254871c. 3

[NKO*22] NAKATANI M., KARIYA S., ONO Y., MARUYAMA T., UENO Y., KOMEMUSHI A., TANIGAWA N.: Radiation exposure and protection in computed tomography fluoroscopy. *Interventional Radiology 7*, 2 (2022), 49–53. doi:10.22575/interventionalradiology.2022-0010. 8

[NSHAS24] NASIM SARAVI A., HORACSEK J., ALIM U., SILVA J. D.: Transferring transfer functions (TTF): A guided approach to transfer function optimization in volume visualization. *Computers & Graphics 124* (2024), 104067. doi:10.1016/j.cag.2024.104067. 2

[OR10] OBERKAMPF W. L., ROY C. J.: *Verification and Validation in Scientific Computing*. Cambridge University Press, Cambridge, UK, 2010. doi:10.1017/CBO9780511760396. 7

[PIG*16] PERDIKARIS P., INSLEY J. A., GRINBERG L., YU Y., PAPKA M. E., KARNIADAKIS G. E.: Visualizing multiphysics, fluid-structure interaction phenomena in intracranial aneurysms. *Parallel Computing 55* (2016), 9–16. doi:10.1016/j.parco.2015.10.016. 2

[SCM99] SHEN H.-W., CHIANG L.-J., MA K.-L.: A fast volume rendering algorithm for time-varying fields using a time-space partitioning (TSP) tree. In *Proceedings Visualization '99* (1999), IEEE, pp. 371–377. doi:10.1109/VISUAL.1999.809910. 2

[SDH*21] SHARMA I., DEMARLE D., HOTA A., CHERNIAK B., GÜNTHER J.: OSPRay studio: Enabling multi-workflow visualizations with OSPRay. In *VisGap - The Gap between Visualization Research and Visualization Software* (2021), Gillmann C., Krone M., Reina G., Wischgoll T., (Eds.), The Eurographics Association. doi:10.2312/visgap.20211086. 2, 3

[SML06] SCHROEDER W., MARTIN K., LORENSEN B.: *The Visualization Toolkit: An Object-Oriented Approach to 3D Graphics*, 4th ed. Kitware, Inc., 2006. URL: https://vtk.org/vtk-textbook/. 3

[SWH05a] STALLING D., WESTERHOFF M., HEGE H.-C.: Amira: A highly interactive system for visual data analysis. In *The Visualization Handbook*, Hansen C. D., Johnson C. R., (Eds.). Elsevier, Burlington, 2005, pp. 749–767. doi:10.1016/B978-012387582-2/50040-X. 3

[SWH05b] STALLING D., WESTERHOFF M., HEGE H.-C.: Amira: A highly interactive system for visual data analysis. In *The Visualization Handbook*, Hansen C. D., Johnson C. R., (Eds.). Elsevier, Amsterdam, 2005, ch. 38, pp. 749–767. doi:10.1016/B978-012387582-2/50040-X. 3

[TCM10] TIKHONOVA A., CORREA C. D., MA K.-L.: An exploratory technique for coherent visualization of time-varying volume data. *Computer Graphics Forum 29*, 3 (2010), 783–792. doi:10.1111/j.1467-8659.2009.01690.x. 2

[TGW*23] TRIVEDI Z., GEHWEILER D., WYCHOWANIEC J. K., RICKEN T., GUEORGUIEV B., WAGNER A., ET AL.: A continuum mechanical porous media model for vertebroplasty: Numerical simulations and experimental validation. *Biomechanics and Modeling in Mechanobiology 22*, 4 (2023), 1253–1266. doi:10.1007/s10237-023-01715-4. 3, 7

[The17] THERMO FISHER SCIENTIFIC: *Amira Software User's Guide*. Thermo Fisher Scientific, Waltham, MA, USA, 2017. URL: https://www.thermofisher.com/us/en/home/electron-microscopy/products/software-em-3d-vis/amira-software.html. 3

[WJA*17] WALD I., JOHNSON G. P., AMSTUTZ J., BROWNLEE C., KNOLL A., JEFFERS J., ET AL.: OSPRay - a CPU ray tracing framework for scientific visualization. *IEEE Transactions on Visualization and Computer Graphics 23*, 1 (2017), 931–940. doi:10.1109/TVCG.2016.2599041. 5

[ZFH*22] ZHOU L., FAN M., HANSEN C., JOHNSON C. R., WEISKOPF D.: A review of three-dimensional medical image visualization. *Health Data Science 2022* (2022), 9840519. PMID: 38487486; PMCID: PMC10880180. doi:10.34133/2022/9840519. 3