Fensalir: A Fractal Domain Reservoir Pipeline for Resolution-Decoupled Field Rendering

GameCult / Fensalir

Living Draft, May 2026

Abstract

We present Fensalir, a real-time rendering architecture that treats geometry, material, transport, and sensor observations as typed spatial evidence rather than as independent pixel-producing passes. Fensalir combines ideas from resampled importance sampling, temporal super-resolution, anisotropic splat rendering, hierarchical virtualized geometry, implicit-field rendering, and sensor fusion into one pipeline centered on a *fractal domain reservoir*. Unlike screen-space temporal antialiasing pipelines, the reservoir is not defined by output texels. It stores selected samples of semantic field and radiance claims in their native domains: cube-sphere tiles, object frames, surface charts, volume domains, tube/curve supports, sensor rigs, and dynamic hierarchy nodes. Pixel rows are only the current GPU execution projection of this deeper sample state.

The core claim is that real-time field rendering should be resolution-decoupled at the level of sampling authority. Frame time and resolution may change the number of probes, candidate samples, hierarchy depth, reuse radius, and reconstruction aggressiveness, but they must not change which subsystem owns identity, support, visibility, contribution mass, or temporal validity. This paper describes Fensalir's pipeline, data model, reservoir update equations, domain-shift validation rules, reconstruction contract, resource-budget model, and implementation strategy. It is written as a living research document: the architecture is confident, the measurements are intentionally explicit about what has and has not yet been proven, and every named mechanism is tied to the research archive that shaped it.

1 Introduction

Real-time rendering pipelines usually inherit a hidden premise: the frame is a grid of pixels, and temporal systems exist to make that grid look less undersampled. Temporal antialiasing accumulates subpixel jitter over frames. Temporal super-resolution uses richer rejection and reconstruction machinery to make fewer samples look like more. Reservoir-based path sampling reuses selected samples across time and space. These are powerful ideas, but the pixel grid remains a gravitational center.

Fensalir starts elsewhere. It is the native runtime and renderer for Aquarium-style clients: authored field grammars, live sensor observations, direct SDF and tube fields, mesh summaries, density/extinction volumes, and planetary-scale flow organs all need to become one coherent visible field. A pixel is not a natural owner for that evidence. A spectrum tube has a rolling buffer coordinate and curve support. A terrain ridge has a cube-face tile, grammar path, brush envelope, and conservative summary. A sensor feature has a calibration frame, timestamp, modality confidence, and fusion window. A flame has density, extinction, emission, and transport uncertainty. The common object is not a pixel. The common object is a bounded claim in a domain.

The thesis of this paper is therefore:

> FENSALIR renders by lowering semantic field and radiance claims into a hierarchical, budgeted, domain-native reservoir. Screen-space rows are a projection of selected evidence, not the foundation of sampling authority.

This reorients familiar techniques. ReSTIR showed that one representative sample can carry the weight mass of many candidates and be reused across neighboring pixels and frames [**?**, **?**]. GRIS made the domain and shift-mapping structure explicit [**?**]. Area ReSTIR demonstrated that reservoir samples over a pixel/lens/filter area must store their actual sample coordinates and support [**?**, **?**]. TSR-style reconstruction showed that temporal history must be inspected, rejected, and reconstructed at the layer that owns visible output [**?**, **?**]. FENSALIR takes the next step: the sample domain is not merely pixel area or lens area, but the fractal spatial domain tree of the scene itself.

2 Contributions

This living draft makes five architectural contributions.

1. **Fractal domain reservoir.** We define a reservoir sample as a selected domain claim carrying identity, selected coordinates, support, target value, source proposal policy, represented candidate count, contribution weight, conservative bounds, and legal domain shifts.
2. **Resolution-decoupled sampling authority.** We separate the native sampling domain from the output grid. Resolution and frame-time targets adjust probe budgets and reconstruction policy, but not semantic ownership.
3. **Unified field evidence pipeline.** We place authored IFS/CSG claims, direct SDF proxies, mesh summaries, tube fields, density/extinction volumes, sensor features, and planetary flow fields behind one bounded Form/Appearance/Transport evidence contract.
4. **TSR/Area-ReSTIR synthesis.** We combine domain-coordinate-aware reservoir reuse with temporal rejection and reconstruction signals, so the denoiser consumes reservoir confidence, support, variance, age, rejection reason, and selected sample coordinates rather than opaque pixel history.
5. **Budgeted hierarchy and residency model.** We describe a dynamic contribution tree that rebalances node probes with bounded stochastic updates and maps selected evidence to CPU, RAM, SSD, and GPU responsibilities.

3 Prior Work and Research Spine

Reservoir resampling. Resampled importance sampling and weighted reservoir sampling underlie ReSTIR's direct-light reuse [**?**]. ReSTIR GI applies the same spatiotemporal sample reuse shape to indirect paths [**?**]. GRIS generalizes the theory to correlated samples, domain changes, unknown PDFs, and shift mappings [**?**]. RTXDI's integration documentation provides practical pass decomposition for initial sampling, temporal reuse, spatial reuse, and final shading [**?**, **?**]. FENSALIR borrows the reservoir spine but moves it earlier: primary field candidates themselves are sampled and reused, not only lighting candidates attached to a known surface.

Area-domain reuse. Area ReSTIR extends reservoir reuse to the area of the pixel filter and lens aperture, storing selected subpixel and lens coordinates and validating reuse by support overlap, shift mappings, visibility, PDFs, and Jacobians [?, ?]. This is the decisive lesson for Fensalir: a tube, SDF splat, density envelope, or fractal tile sample must carry selected coordinates and support in its own domain. A reservoir that only knows "which texel wrote me" cannot prove legal reuse.

Temporal reconstruction. Modern TAA and TSR systems demonstrate that temporal history needs explicit reprojection, disocclusion classification, clamping/rejection, accumulated sample accounting, and debug visualization [?, ?]. Fensalir treats these as reservoir resolve responsibilities. A downstream presentation pass may filter the resolved field, but it must not become a second owner of producer identity or temporal persistence.

Implicit fields and splats. Hart's sphere tracing establishes the safety contract for implicit surfaces: distance steps require conservative bounds [?]. EWA surface and volume splatting show how anisotropic kernels and screen-space filtering avoid aliasing and excessive blur [?, ?]. 3D Gaussian Splatting and Octree-GS demonstrate the practical value of anisotropic primitives, visibility sorting, and hierarchy-aware splat LOD [?, ?]. Fensalir adopts compact-support anisotropic envelopes and projected-support scoring, while rejecting infinite Gaussian tails as a default hot-path cost.

Hierarchical geometry and residency. Geometry clipmaps keep terrain detail in nested windows around the viewer [?, ?]. Nanite demonstrates fine-grained hierarchical cuts driven by projected error and resident summaries [?]. Instant-NGP shows that explicit multiresolution data structures can make stochastic spatial learning fast and GPU-friendly [?]. Fensalir combines these into an occupancy and contribution graph: a hierarchy of conservative summaries, online contribution estimates, residency state, and selected cuts.

Domain addressing. Planetary fields use cube-sphere or quadrilateralized spherical cube domains to avoid latitude-longitude singularities [?, ?, ?]. The Fensalir archive currently favors tangent cube-sphere projection as a measured first default, with QSC/COBE-like mappings retained for higher area-regularity paths [?]. This is not merely a planet trick. It is the canonical example of a domain-first renderer: the surface chart owns the address before the renderer owns packets.

Sensor and flow fields. Curless and Levoy's volumetric fusion and KinectFusion show that streamed sensor observations can accumulate into implicit fields [?, ?]. Fensalir's planetary flow organ extends the same evidence vocabulary to stochastic advection fields: cube-sphere domains, dataset provenance, transition kernels, uncertainty, and runtime packets [?]. Its archive cites PlasticAdrift, GEBCO, ETOPO, ERA5, GLORYS, OSCAR, HYCOM, GraphCast, SFNO/FourCastNet, MeshGraphNets, GenCast, and NeuralGCM as external anchors for data and model choices [?, ?].

Presentation and implementation substrate. The renderer remains scene-linear through exposure, bloom, and display transform, following ACES/filmic/HDR production guidance [?, ?, ?, ?, ?, ?, ?, ?]. D3D12 implementation follows explicit descriptor, barrier, memory, fence, and performance guidance [?, ?, ?, ?, ?, ?, ?, ?, ?]. These are not the novel parts of the pipeline,

but they constrain the machine: no research architecture survives contact with a GPU if resource ownership is fog.

4 System Overview

Fensalir consumes producer intent and emits a resolved HDR field texture plus diagnostics. Producers may be authored DSL grammars, direct SDF shaders, generated TubeField resources, mesh packages, volume textures, surface pages, Mimir sensor observations, or planetary simulation outputs. They do not draw final truth directly. They publish bounded claims.

```
DSL / producer intent / resident resource / live sensor input
-> domain binding
-> semantic Form / Appearance / Transport claims
-> ownership tree and conservative summaries
-> stochastic contribution cache and residency scheduler
-> bounded proposals with target and sourcePdf
-> fractal domain reservoir
-> temporal domain validation and shift
-> spatial/domain validation and shift
-> TSR-style rejection, reconstruction, and denoiser features
-> backend packets and render passes
-> resolved HDR field texture, debug telemetry, evidence ledger
```

The important ownership rule is simple: a backend is a lowering, not a truth owner. A mesh may be a first-class contributor if it provides stable identity, motion or previous-frame mapping, conservative bounds, material/field encoding, and guide data. A mesh that only paints pixels is a fallback draw. The same rule applies to SDF proxies, TubeField paths, volumes, surface pages, and sensor fields.

5 Data Model

5.1 Domains

A domain says where coordinates live and how they map to a parent space:

Field	Meaning
`DomainKey`	stable identity
`ParentKey`	lineage for shift/reuse
`Kind`	cube tile, surface chart, object, volume, tube, sensor rig
`Frame`	local basis and transform
`Projection`	mapping to parent/world coordinates
`Bounds`	conservative support in parent/domain space
`Periodicity`	torus, wrap, seam, or none
`Owner`	engine/client/producible authority

Domains are the first validation surface. If two samples do not have a legal lineage or shift mapping, they do not reuse each other.

5.2 Claims

Claims are semantic statements, not render packets:

```
Claim {
  ClaimKey, DomainKey, NodeKey,
  Layer: Form | Appearance | Transport,
  Encoding: Height | SignedDistance | Density | Extinction |
            Material | Phase | Emission | Radiance | Feature | Confidence,
  LocalFrame, Envelope, Payload, Tags, CostTier, Seed
}
```

The Form/Appearance/Transport split replaces the earlier implementation shorthand of separate SDF, PBR, and Radiosity packet names. Form states what exists where; Appearance states local material and emission behavior; Transport states how energy moves through or from the field. Opaque SDF surfaces, transparent density volumes, sensor confidence regions, mesh summaries, and spline fields can therefore share machinery without blending incompatible meanings.

5.3 Reservoir Samples

The fractal domain reservoir stores a selected sample:

```
ReservoirSample {
  SelectedClaim,
  DomainKey,
  SelectedDomainCoordinate,
  SupportFootprint,
  SelectedTarget,
  WeightSum,
  CandidateCount,
  ContributionWeight,
  SourcePdfOrPolicy,
  ProposalKind,
  ShiftMappingMetadata,
  ConservativeBounds,
  Confidence,
  SampleAge,
  RejectionOrValidationMask
}
```

The selected coordinate and support are mandatory for area-domain contributors. For a TubeField, this includes selected subpixel and curve/buffer-domain coordinates. For a cube-sphere terrain sample, it includes face/tile/local UV and brush support. For a sensor feature, it includes sensor-rig/world mapping and calibration confidence. For a volume, it includes local volume coordinate and extinction/density support.

6 Reservoir Update

Fensalir uses the usual RIS reservoir shape, but the sample type is a field claim rather than a direct-light sample. For a candidate c with target $\hat{p}(c)$, source proposal density $p(c)$, and represented candidate count m_c, the unnormalized candidate weight is

$$w_c = \frac{\hat{p}(c)}{p(c)}. \tag{1}$$

The reservoir maintains

$$W \leftarrow W + w_c, \qquad M \leftarrow M + m_c, \tag{2}$$

and selects the candidate with probability

$$P(c \text{ selected}) = \frac{w_c}{W}. \tag{3}$$

The contribution weight exported for final evaluation follows

$$C = \frac{W}{M \cdot \hat{p}(s)}, \tag{4}$$

where s is the selected sample. Invalid zero, negative, infinite, or NaN targets/PDFs are rejected before proposal emission. Deterministic structural proposals use an explicit source policy, commonly $p = 1$, rather than pretending to be free samples.

Merging another reservoir R_o into the current reservoir R uses the standard reservoir merge probability:

$$P(R_o \text{ selected}) = \frac{W_o}{W + W_o}. \tag{5}$$

The merge is legal only after the candidate reservoir's selected sample has been shifted or re-evaluated in the target domain and its target/source support story remains valid.

7 Domain Shifts and Validation

Reuse is guilty until proven compatible. A temporal or spatial pass may not merge a reservoir because two pixels are nearby, two travels are similar, or two field ids match. Those are hints. Legal reuse requires a domain shift.

7.1 Temporal Reuse

Temporal reuse validates:

- domain lineage and selected sample support;
- camera/object motion and previous-domain reprojection;
- disocclusion and occlusion;
- travel/depth and normal/gradient compatibility;
- field id, material class, layer, and encoding;
- motion, previous UV/domain coordinate, and guide confidence;
- sample age, variance, and producer-domain validity.

The Area ReSTIR lesson is explicit: if the previous selected sample does not lie in the target support, the reuse pass either rejects it or applies a legal shift/reconnection with proper evaluation. The previous reservoir cannot become visible fallback by itself.

7.2 Spatial and Domain Reuse

Spatial reuse is a domain-neighbor problem, not only a screen-neighbor problem. Valid neighborhoods include screen tiles, cube-face seams, quadtree siblings, periodic torus wraps, curve supports, volume cells, object-local frames, and calibrated sensor regions. For each neighbor, FENSALIR evaluates a shift:

$$T_{a\to b} : \mathcal{S}_a \to \mathcal{S}_b. \tag{6}$$

The shifted sample must be re-evaluated in the target domain b. If the mapping changes measure, the shift metadata must carry a Jacobian or an equivalent MIS correction. If the mapping is deterministic structural replay, the proposal policy must say so explicitly.

8 Hierarchy: The Dynamic Contribution Tree

The reservoir does not float in isolation. It sits inside an occupancy and contribution graph:

```
Node {
  NodeKey, DomainKey, Bounds,
  SummaryPayload, ChildPayload,
  MeanContribution, Variance, Confidence,
  SampleAge, ResidencyState,
  LastVisibleFrame, LastUpdateFrame
}
```

Each node has a conservative summary. Missing children render through parent summaries. The selected hierarchy cut is driven by projected error, material delta, current reservoir confidence, uncertainty, and estimated CPU/GPU/RAM/SSD cost.

The projected contribution score begins as a conservative heuristic:

$$r_{\text{px}} = \text{projectRadius}(B_n, camera), \tag{7}$$

$$e_{\text{px}} = \text{projectError}(E_n, B_n, camera), \tag{8}$$

$$S_n = \frac{\max(e_{\text{px}} \cdot V_n,\ \Delta m_n \cdot A_n) \cdot I_n}{\max(C_n, \epsilon)}, \tag{9}$$

where B_n is node bound, E_n is maximum Form error, V_n is visible coverage estimate, Δm_n is material delta, A_n is projected area, I_n is importance bias, and C_n is estimated update/render cost.

8.1 Bounded Stochastic Probing

The tree cannot update every node every frame. FENSALIR uses an online estimator with stochastic exploration:

$$P_{\text{update}}(n) \propto \frac{P_{\text{visible}}(n) \cdot \text{stale}(n) \cdot \text{uncertainty}(n) \cdot \max(S_n, S_{\text{parentBias}})}{\max(C_n, \epsilon)}. \tag{10}$$

On observation o,

$$\delta = o - \mu_n, \tag{11}$$
$$\mu_n \leftarrow \mu_n + \alpha\delta, \tag{12}$$
$$\sigma_n^2 \leftarrow (1-\beta)\sigma_n^2 + \beta\delta^2. \tag{13}$$

On non-update, confidence decays and uncertainty grows. This is deliberately the boring first machine: EMA, variance, and bandit-style exploration. A learned predictor may later rank update and residency priority, but it may not replace conservative summaries, SDF bounds, calibration bounds, or visibility safety.

9 Backend Lowerings

Fensalir can lower the same evidence contract into multiple execution paths:

- 2D height/material/signed-distance/confidence pages;
- 3D compact SDF splats for opaque Form;
- density/extinction/emission splats for transparent Form and Transport;
- 2D fields projected onto 3D surfaces;
- direct SDF proxy draws;
- generated TubeField curve/tube meshes and analytic tube SDF proposals;
- resource-backed mesh packages with stable ids and guide outputs;
- sensor confidence volumes and calibrated feature fields;
- debug overlays and receipts.

The first implementation already has pieces of this: shaped 2D brush envelopes, semantic claim/tree rows, conservative summaries, fractal contribution state, projected SDF splat compilation, GPU splat/reservoir receipt harness, direct TubeField resource lowerings, surface, volume, and mesh resource declarations, and a shared four-row field reservoir execution ABI. The live ABI currently uses:

Row	Meaning
0	current-frame initial RIS
1	temporal reuse reservoir
2	spatial/domain reuse reservoir
3	final resolved reservoir consumed by bloom/presentation

Each row is now a nine-lane `FieldReservoirSample`: color/travel, metadata, control, guide, motion, domain sample, domain support, proposal, and stats. The domain sample lane stores selected pixel/sample UV plus packed producer coordinates; the domain support lane stores support footprint, domain kind, and shift kind. TubeField proposals write selected subpixel UV, logical source column, curve coordinate, tube support, and explicit motion when available. Field id still

carries physical rolling-column identity. SDF proposals reconstruct object-local hit coordinates and mark object-motion replay. Temporal and spatial reuse now reject missing domain state and include support overlap in the validation weight. The first TubeField replay path now re-evaluates the shifted selected logical column/curve coordinate through a bounded producer-keyed replay manifest and raw source descriptor table. SDF object-hit replay reconstructs the previous selected world hit from stored UV/travel, carries it through object center motion, and validates current ray/travel/support before temporal reuse can merge it; exact client SDF distance replay remains a producer-owned extension point.

10 Reconstruction and Denoising

Fensalir does not place a conventional TAA pass after rendering and hope it discovers identity. Reservoir resolve consumes:

- selected domain sample coordinates;
- selected support/filter footprint;
- contribution weight, weight sum, selected target, and candidate count;
- reservoir confidence, sample age, variance, and update probability;
- field id, layer, encoding, normal/gradient, travel, and motion;
- rejection reason, support miss, disocclusion, occlusion, and shift kind.

The denoiser is therefore part of the reservoir pipeline, not downstream cosmetics. If a region is rejected because support is missing, it should be reconstructed differently from a region rejected because of occlusion, field-id mismatch, low confidence, or high variance. The scheduler can also respond: persistent reconstruction stress increases future probe probability in the relevant domain nodes.

This closes the loop:

```
reservoir confidence / rejection / variance / denoiser stress
-> contribution cache update
-> future probe and residency priority
-> better selected claims
-> less reconstruction debt
```

11 Resource Contract

Lower frame budgets reduce probes, node depth, candidate count, richer shift paths, and denoiser feature quality. They do not change domain ownership. High budgets buy more evidence. Low budgets render parent summaries and older validated reservoirs. Neither mode lets a pixel row become truth.

12 Implementation Status

Fensalir is not a paper-only machine. The archive records working slices:

Resource	Owned responsibility
CPU	grammar expansion, domain math, coarse scoring, stochastic scheduling, selected cuts, page requests, eviction, evidence logs
RAM	summaries, estimator state, warm metadata, selected cuts, recent probes
SSD	serialized payload pages, summaries, probe history, optional training datasets; never frame-blocking
GPU	selected packet evaluation, SDF/tube/mesh/volume passes, compute scoring where profitable, page-table sampling, reservoir rows, debug views

Table 1: The resource contract. The architecture is budget-adaptive because each resource has a narrow job.

- cube-sphere projection harness with tangent projection measured against normalize projection [**?**];
- shaped compact-support brush envelopes and height-field lowering [**?**];
- semantic contracts for domains, claims, ownership trees, summaries, and selected cuts [**?**];
- contribution cache and occupancy graph scaffolding;
- Form encodings for signed distance, density, and extinction;
- projected 2D-to-3D SDF splat compiler;
- GPU receipt harness with two million splats and two million reservoirs per layer resident, updating 50,000 entries per layer per frame with two candidates per update, measured at 4.471 ms/frame over 120 frames on a GTX 1070-class machine [**?**];
- D3D12 TubeField resource and generated-mesh lowerings;
- shared field reservoir ABI with current/temporal/spatial/final row semantics, selected domain/sample coordinates, support footprints, proposal lanes, stats lanes, and row-3 debug modes for rejection, proposal, domain, support, and shift state;
- HDR presentation path with exposure, bloom pyramid, ACES-fit transform, and debug modes [**?**].

The known architectural gap has moved forward: the current row ABI now carries domain/sample-coordinate authority, TubeField replay uses a bounded producer-keyed manifest/source table to re-evaluate selected logical column/curve coordinates in the current frame, and SDF object-hit replay carries the previous selected hit through object motion before validating current ray/travel/support. The next proof cuts are exact producer SDF replay, measured temporal leakage/ghosting, and measured rejection precision. The reservoir work itself is no longer allowed to grow to native presentation resolution by accident: scene evidence targets, reservoir candidate/history rows, reservoir resolve, bloom sources, and match-window graph targets are tied to a clamped internal work grid, while the final present buffer owns full resolution reconstruction and display. Those captures now have an explicit comparison switch: native mode enables the

domain validity, support-overlap, and producer-replay gates, while texel-baseline mode uses the same row ABI but disables those native-domain gates during temporal reuse. The first paired Fensalir capture set `reservoir-compare-20260529-231904-*` produced a 10.8208% final-frame pixel delta. The repeatable script `scripts/measure-reservoir-mode-comparison.ps1` reports that delta; it is a live-switch check, not a quality claim. The sequence harness can now prefer a temporal-owner disocclusion mask over the broad rejection debug view. That mask is derived from the reservoir rejection codes for previous history outside the frame, closer previous history, and support loss; it localizes where an eventual high-budget reference score should look first. It can also capture a same-time native-domain reference at a larger reservoir work-grid scale, producing reference-error rows without introducing a native-resolution intermediate reservoir. Those rows feed an offline budget-pressure probe that ranks fixed tiles by masked reference error, giving the future scheduler a concrete pressure signal without letting a debug script own runtime visibility. The renderer now has a fixed spatial-reuse sampling budget in the history update owner: current and temporal rows remain per-work-grid-pixel, while 3x3 spatial neighbor reuse runs only on the active budget phase for the frame. The active phase is visible through the Reservoir Spatial Budget debug view, so budget enforcement can be checked against captured pixels rather than inferred from state alone. The system has cut the false "nearest-travel-minus-coverage" priority owner and the side-cache temptation.

13 Evaluation Plan

The pipeline should be evaluated at four levels.

Reservoir correctness. Unit tests should verify RIS add/merge selection, contribution weight, candidate count preservation, invalid target/PDF rejection, deterministic RNG, and CPU/GPU parity for packing and update rules.

Domain reuse. Fixtures should exercise cube-face seams, quadtree neighbors, tube support, SDF proxy movement, mesh summaries, density/extinction regions, sensor calibration shifts, and previous-frame motion. Negative checks matter: an old texel path must no longer be able to override the selected domain owner.

Visual behavior. Captures should compare final color and debug views for ghosting, occluded-tube leakage, background speckle, disocclusion rejection, spatial reuse, contribution weight, support overlap, and denoiser stress.

Budget scaling. Benchmarks should sweep resolution and target frame budgets while recording probe count, selected cut size, GPU time, page requests, stale confidence, variance, reconstruction debt, and visible error. The expected result is not constant quality. The expected result is graceful degradation without authority moving back to screen texels.

14 Discussion

Fensalir is intentionally vertical. Its DSL, hierarchy, cache, reservoir, renderer, temporal reconstruction, denoiser, resource residency, and debug ledger are designed as one machine. That is a risk: vertical integration can become a private cathedral of assumptions. The countermeasure is explicit

ownership. Candidate generators own proposal distributions. Target evaluators own importance. Reservoirs own resampling math. Reuse passes own validation and shift mappings. Conservative summaries own safety. Residency owns resource pressure. Resolve owns reconstruction. Debug surfaces must observe the same layer where the user sees failure.

This is also why Fensalir does not simply "use ReSTIR" or "use TSR". Those systems solve crucial subproblems, but the renderer's native objects are broader: domain claims, selected hierarchy nodes, density fields, sensor features, flow kernels, and packet lowerings. The proper synthesis is not another post-process. It is a domain reservoir with reconstruction-aware feedback.

15 Conclusion

Fensalir proposes a real-time pipeline in which pixels are consumers of domain evidence, not owners of sampled truth. A fractal domain reservoir stores selected field and radiance claims, their support, their contribution mass, and their legal reuse story. A dynamic contribution tree decides which domains deserve more probes under budget. TSR-style reconstruction consumes the actual reservoir diagnostics and feeds sampling pressure back into the scheduler.

The result is a renderer that can scale toward arbitrary resolution and frame targets by adjusting evidence budgets rather than changing the architecture. The machine is still being built, but the invariant is now sharp enough to cut against implementation drift: if a sample cannot name its domain, selected coordinate, support, target, proposal policy, represented mass, and legal shift, it is not yet a Fensalir reservoir sample.

References

[1] B. Bitterli, C. Wyman, M. Pharr, P. Shirley, A. Lefohn, and W. Jarosz. Spatiotemporal reservoir resampling for real-time ray tracing with dynamic direct lighting. *ACM Transactions on Graphics*, 2020. https://research.nvidia.com/labs/rtr/publication/bitterli2020spatiotemporal/

[2] Y. Ouyang, S. Liu, M. Kettunen, M. Pharr, and J. Pantaleoni. ReSTIR GI: Path resampling for real-time path tracing. *Computer Graphics Forum*, 2021. https://research.nvidia.com/publication/2021-06_restir-gi-path-resampling-real-time-path-tracing

[3] D. Lin, M. Kettunen, B. Bitterli, J. Pantaleoni, C. Yuksel, and C. Wyman. Generalized resampled importance sampling: Foundations of ReSTIR. *ACM Transactions on Graphics*, 2022. https://graphics.cs.utah.edu/research/projects/gris/sig22_GRIS.pdf

[4] S. Zhang, D. Lin, M. Kettunen, C. Yuksel, and C. Wyman. Area ReSTIR: Resampling for real-time defocus and antialiasing. *ACM Transactions on Graphics*, 2024. https://graphics.cs.utah.edu/research/projects/area-restir/AreaReSTIR-SIGGRAPH2024.pdf

[5] S. Zhang. Area-ReSTIR source code repository. https://github.com/guiqi134/Area-ReSTIR

[6] NVIDIA. RTXDI ReSTIR GI documentation. https://github.com/NVIDIA-RTX/RTXDI/blob/main/Doc/RestirGI.md

[7] NVIDIA. RTXDI ReSTIR PT documentation. https://github.com/NVIDIA-RTX/RTXDI/blob/main/Doc/RestirPT.md

[8] Epic Games. Temporal Super Resolution in Unreal Engine. https://dev.epicgames.com/documentation/unreal-engine/temporal-super-resolution-in-unreal-engine

[9] Epic Games. Temporal Super Resolution Frequently Asked Questions. https://dev.epicgames.com/documentation/unreal-engine/temporal-super-resolution-frequently-asked-questions-for-unreal-engine

[10] J. C. Hart. Sphere tracing: A geometric method for the antialiased ray tracing of implicit surfaces. *The Visual Computer*, 1996. https://experts.illinois.edu/en/publications/sphere-tracing-a-geometric-method-for-the-antialiased-ray-tracing/

[11] M. Zwicker, H. Pfister, J. van Baar, and M. Gross. EWA splatting. *IEEE Visualization*, 2001. https://dash.harvard.edu/bitstream/handle/1/4138240/Zwicker_EWA.pdf

[12] L. Ren, H. Pfister, and M. Zwicker. Object space EWA surface splatting: A hardware accelerated approach to high quality point rendering. *Computer Graphics Forum*, 2002. CMU publication page

[13] B. Kerbl, G. Kopanas, T. Leimkuehler, and G. Drettakis. 3D Gaussian Splatting for Real-Time Radiance Field Rendering. *ACM Transactions on Graphics*, 2023. https://arxiv.org/abs/2308.04079

[14] K. Ren et al. Octree-GS: Towards consistent real-time rendering with LOD-structured 3D Gaussians. 2024. https://arxiv.org/abs/2403.17898

[15] F. Losasso and H. Hoppe. Geometry clipmaps: Terrain rendering using nested regular grids. *ACM Transactions on Graphics*, 2004. https://hhoppe.com/geomclipmap.pdf

[16] A. Asirvatham and H. Hoppe. Terrain rendering using GPU-based geometry clipmaps. *GPU Gems 2*, 2005. https://developer.nvidia.com/gpugems/gpugems2/part-i-geometric-complexity/chapter-2-terrain-rendering-using-gpu-based-geometry

[17] B. Karis, R. Stubbe, and G. Wihlidal et al. A deep dive into Nanite virtualized geometry. *Advances in Real-Time Rendering*, SIGGRAPH 2021. https://advances.realtimerendering.com/s2021/Karis_Nanite_SIGGRAPH_Advances_2021_final.pdf

[18] T. Mueller, A. Evans, C. Schied, and A. Keller. Instant neural graphics primitives with a multiresolution hash encoding. *ACM Transactions on Graphics*, 2022. https://research.nvidia.com/publication/2022-07_instant-neural-graphics-primitives-multiresolution-hash-encoding

[19] PROJ contributors. Quadrilateralized Spherical Cube. https://proj.org/en/stable/operations/projections/qsc.html

[20] F. K. Chan and E. M. O'Neill. A quadrilateralized spherical cube earth data base. NASA Technical Report, 1975. https://ntrs.nasa.gov/citations/19810002572

[21] P. Clarberg. Cube-to-sphere projections for procedural texturing and beyond. *Journal of Computer Graphics Techniques*, 2018. https://jcgt.org/published/0007/02/01/paper.pdf

[22] B. Curless and M. Levoy. A volumetric method for building complex models from range images. *SIGGRAPH*, 1996. https://graphics.stanford.edu/papers/volrange/volrange.pdf

[23] R. A. Newcombe, S. Izadi, O. Hilliges, D. Molyneaux, D. Kim, A. J. Davison, P. Kohli, J. Shotton, S. Hodges, and A. Fitzgibbon. KinectFusion: Real-time dense surface mapping and tracking. *ISMAR*, 2011. https://www.microsoft.com/en-us/research/publication/kinectfusion-real-time-dense-surface-mapping-tracking/

[24] Academy Color Encoding System. ACES system overview. https://docs.acescentral.com/background/overview/

[25] Epic Games. How Epic Games is handling auto exposure in 4.25. https://www.unrealengine.com/tech-blog/how-epic-games-is-handling-auto-exposure-in-4-25?lang=en-US

[26] Epic Games. Bloom in Unreal Engine. https://dev.epicgames.com/documentation/en-us/unreal-engine/bloom-in-unreal-engine?application_version=5.6

[27] J. Jimenez. Next generation post processing in Call of Duty: Advanced Warfare. https://www.iryoku.com/next-generation-post-processing-in-call-of-duty-advanced-warfare/

[28] J. Hable. Filmic tonemapping operators. https://filmicworlds.com/blog/filmic-tonemapping-operators/

[29] J. Hable. Filmic tonemapping with piecewise power curves. https://filmicworlds.com/blog/filmic-tonemapping-with-piecewise-power-curves/

[30] J. Hable. Uncharted 2 HDR lighting. SIGGRAPH 2010 course material. https://advances.realtimerendering.com/s2010/Hable-Uncharted2%28SIGGRAPH%202010%20Advanced%20RealTime%20Rendering%20Course%29.pdf

[31] S. Lagarde and C. de Rousiers. Moving Frostbite to physically based rendering 3.0. SIGGRAPH 2014 course material. https://cgvr.cs.uni-bremen.de/teaching/cg_literatur/Moving%20Frostbite%20to%20Physically%20Based%20Rendering%203.0%2C%202014%2C%20Sebastien%20Lagarde.pdf

[32] PlasticAdrift. PlasticAdrift project. https://plasticadrift.org/

[33] R. McAdam and E. van Sebille. Surface connectivity and transition matrices for ocean plastic transport. https://dbc.library.uu.nl/handle/1874/362629

[34] GEBCO. GEBCO 2025 Grid. https://www.gebco.net/data-products-gridded-bathymetry-data/gebco2025-grid

[35] NOAA NCEI. ETOPO global relief model. https://www.ncei.noaa.gov/products/etopo-global-relief-model

[36] Copernicus Climate Change Service. ERA5 climate reanalysis. https://climate.copernicus.eu/climate-reanalysis

[37] Copernicus Marine Service. GLORYS global ocean physics reanalysis. https://data.marine.copernicus.eu/product-detail/GLOBAL_MULTIYEAR_PHY_001_030

[38] Earth and Space Research. OSCAR surface currents. https://www.esr.org/data-products/oscar/oscar-surface-currents/

[39] HYCOM Consortium. HYCOM data server. https://www.hycom.org/

[40] R. Lam et al. Learning skillful medium-range global weather forecasting. GraphCast. https://deepmind.google/research/publications/22598/

[41] T. Bonev et al. Spherical Fourier neural operators: Learning stable dynamics on the sphere. https://research.nvidia.com/publication/2023-06_spherical-fourier-neural-operators-learning-stable-dynamics-sphere

[42] A. Pathak et al. FourCastNet: A global data-driven high-resolution weather model using adaptive Fourier neural operators. https://authors.library.caltech.edu/records/k959a-53q45

[43] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. Battaglia. Learning mesh-based simulation with graph networks. https://arxiv.org/abs/2010.03409

[44] I. Price et al. Probabilistic weather forecasting with machine learning. GenCast. https://www.nature.com/articles/s41586-024-08252-9

[45] S. Kochkov et al. Neural general circulation models for weather and climate. https://www.nature.com/articles/s41586-024-07744-y

[46] Microsoft. Resource binding flow of control in Direct3D 12. https://learn.microsoft.com/en-us/windows/win32/direct3d12/resource-binding-flow-of-control

[47] Microsoft. Fence-based resource management. https://learn.microsoft.com/en-us/windows/win32/direct3d12/fence-based-resource-management

[48] Microsoft. Memory management in Direct3D 12. https://learn.microsoft.com/en-us/windows/win32/direct3d12/memory-management

[49] Microsoft. Direct3D 12 interfaces. https://learn.microsoft.com/en-us/windows/win32/direct3d12/direct3d-12-interfaces

[50] Microsoft DirectX Developer Blog. A look inside D3D12 resource state barriers. https://devblogs.microsoft.com/directx/a-look-inside-d3d12-resource-state-barriers/

[51] NVIDIA. Advanced API performance: barriers. https://developer.nvidia.com/blog/advanced-api-performance-barriers/

[52] AMD GPUOpen. RDNA performance guide. https://gpuopen.com/learn/rdna-performance-guide/

[53] AMD GPUOpen. D3D12 Memory Allocator. https://gpuopen.com/d3d12-memory-allocator/

[54] AMD GPUOpen. Using D3D12 heap type GPU upload. https://gpuopen.com/learn/using-d3d12-heap-type-gpu-upload/

[55] Fensalir archive. Perfect Machine architecture. file:///E:/Projects/Fensalir/docs/perfect-machine-architecture.md

[56] Fensalir archive. Fractal domain reservoir. file:///E:/Projects/Fensalir/docs/fractal-domain-reservoir.md

[57] Fensalir archive. Temporal spatial evidence reservoir. file:///E:/Projects/Fensalir/docs/temporal-spatial-evidence-reservoir.md

[58] Fensalir archive. Fractal brush architecture plan. file:///E:/Projects/Fensalir/research/rendering/fractal-brush-architecture-plan.md

[59] Fensalir archive. Cube-sphere fractal brush research. file:///E:/Projects/Fensalir/research/rendering/cube-sphere-fractal-brushes.md

[60] Fensalir archive. Fractal brush implementation roadmap. file:///E:/Projects/Fensalir/research/rendering/fractal-brush-implementation-roadmap.md

[61] Fensalir archive. Fractal projection report. file:///E:/Projects/Fensalir/research/rendering/fractal-projection-report.md

[62] Fensalir archive. HDR/PBR presentation research. file:///E:/Projects/Fensalir/research/rendering/hdr-pbr-presentation.md

[63] Fensalir archive. Planetary flow organ literature. file:///E:/Projects/Fensalir/research/planetary-flow-organ-literature.md

[64] Fensalir archive. HLSL renderer documentation. file:///E:/Projects/Fensalir/docs/hlsl-renderer.md

[65] Fensalir archive. TSR-inspired TAA specification. file:///E:/Projects/Fensalir/docs/tsr-inspired-taa-spec.md

[66] Fensalir archive. Rendering teardown/rebuild protocol. file:///E:/Projects/Fensalir/docs/rendering-teardown-rebuild-protocol.md

[67] VibeGeometry archive. Procedural doctrine and CSG research. file:///E:/Projects/VibeGeometry/docs/research/procedural-doctrine.md, file:///E:/Projects/VibeGeometry/docs/research/dream-machine-grammar.md, file:///E:/Projects/VibeGeometry/docs/research/realtime-csg-doctrine.md

[68] GameCult archive. The Perfect Machine Has One Eye. file:///E:/Projects/gamecult-site/GameCult/Blog/perfect-machine-fractal-reservoir-architecture.md