# Calibrated Per-Carrier Confidence and Certified Pruning for Gaussian-Splat Assets

Krishi Attri
Seoul National University
kattri@snu.ac.kr ORCID: 0009-0005-4695-6467

**Abstract.** A 3D Gaussian-splatting asset ships millions of primitives, each carrying a per-primitive confidence — a view count, an opacity — that downstream pruning, streaming, and level-of-detail decisions consume as if it meant something. It does not: these numbers are heuristics with no operational meaning and no guarantee, so a consumer that drops "low-confidence" primitives is guessing. We ask what it takes to give a splat asset a per-carrier confidence a consumer can actually trust, and report AURA, a preview-stage library (`v1.0.0`) that attaches one. The method has three parts, each pure-CPU post-processing over the exported carrier array. First, a held-out per-carrier *reliability* label from multi-view re-projection agreement: a carrier is reliable to the degree its stored colour matches what independent held-out views observe where it projects, computed under both an occlusion-blind colour label and a stricter occlusion-aware depth label. Second, isotonic (PAVA) calibration of an export-time feature — train-view colour agreement, computable without held-out ground truth — into a calibrated confidence. Third, a distribution-free *split-conformal pruning certificate*: prune every carrier below a threshold $\tau$ and the retained set's mean unreliability is at most $\epsilon$ with confidence $1 - \alpha$.

On four real scenes (Tanks-and-Temples *Truck*; Mip-NeRF-360 *Garden*, *Kitchen*, *Room*), the export-time feature predicts held-out reliability at correlation 0.91–0.98 (colour label) while the shipped view-count heuristic ($|r| \leq 0.13$) and opacity ($-0.18$ to $+0.16$) do not; isotonic calibration cuts expected calibration error from 0.46–0.59 to 0.0006–0.0016 (∼300–900×); and confidence-guided selection recovers 0.58–0.72 mean retained reliability, within 1–4% of an oracle ceiling and above opacity (0.37–0.53, at or below random) at every budget — keeping only 10% of carriers it retains 0.77–0.90 reliability versus opacity's 0.31–0.49. The properties survive the stricter occlusion-aware label; they further persist at native full resolution under a corrected, genuinely held-out split (which lowers one over-optimistic per-scene certificate from 1.00 to 0.77), transfer across scenes for selection (AUC within ±0.0004) with a small per-scene conformal set restoring the certificate, and survive a stricter render-loss label read from the actual alpha composite with honestly weaker margins. We extend the single pruning certificate into a certified level-of-detail ladder — a consumer streams carriers in descending calibrated confidence and may stop at any published level with a distribution-free, family-wise bound on the reliability mass it thereby discards (all 16 bounds hold across four scenes and four levels). We deliver the calibrated confidence through the `KHR_gaussian_splatting _AURA_CONFIDENCE` channel, the OpenUSD 26.03 `UsdVolParticleField3DGaussianSplat` schema, and a sidecar for Niantic's SPZ v4 container whose writer is cross-validated bit-exact against the reference implementation. We are deliberate about scope: AURA is a preview-stage system; the reliability labels are self-computed re-projection proxies, not human ground truth; the evaluation gates, while content-checked against the committed artifacts, are our own and run locally; the scenes are single captures; the typed-carrier backend result we report (+0.33 dB on *Truck*) is a reproduction of Deformable Beta Splatting, not a contribution of this paper. What we do claim is narrow and, we believe, new: a calibrated, certified, exportable per-carrier confidence is a capability a bare Gaussian splat does not have, and this paper measures it honestly on real scenes.

*July 2026 working version. Every quantitative claim below is machine-checked against a committed artifact by content-checked publication gates, and reproduces CPU-only, bit-for-bit, from a fresh clone (*`REPRODUCE.md`*).*

## 1 Introduction

3D Gaussian Splatting (3DGS) [1] has made captured radiance fields cheap to render and easy to ship: a scene is a bag of anisotropic primitives with position, covariance, opacity, and view-dependent colour, rendered in real time by mature toolchains [2]. As these assets move through pipelines — pruned to fit a memory budget, streamed at a level of detail, culled before a viewer sees them — the pipeline needs to know which primitives it can afford to drop. In practice it reaches for a per-primitive proxy: the opacity, or a count of how many training views saw the primitive, or a hand-tuned squash of one of these. None of these is a calibrated probability. A primitive with opacity 0.9 is not reliable 90% of the time; a primitive seen in many views is not thereby trustworthy; and dropping every primitive below a threshold carries no guarantee about what the retained set is worth. The confidence a splat asset ships today is decoration.

This paper asks what it takes to make that confidence real, and reports what we learned building the calibrated-

confidence layer of AURA, a preview-stage, open library that turns posed captures into a typed radiance asset. AURA does not try to out-render 3DGS or its Beta-kernel successor [3]; those remain the quality path. It adds the layer they do not provide: a per-carrier[1] confidence that is a *calibrated* reliability and comes with a *distribution-free pruning guarantee*, and that travels with the asset through standard interchange. Three questions organize the work.

**RQ1 (signal).** Is there an export-time signal — computable at asset-write time, with no held-out ground truth — that actually predicts a carrier's reliability, where the shipped heuristics do not? We define a held-out reliability label from multi-view re-projection agreement and show that a train-view colour-agreement feature tracks it at correlation 0.91–0.98 across four real scenes, while the view-count heuristic ($|r| \leq 0.13$) and opacity (−0.18 to +0.16) are uninformative (Sec. 5).

**RQ2 (calibration and certificate).** Can that signal be turned into a confidence a consumer can read as a probability, and into a pruning decision that comes with a guarantee? We fit an isotonic (PAVA) calibrator on a held-out split and attach a split-conformal certificate. Calibration cuts expected calibration error (ECE) by ∼300–900×; the certificate bounds the retained set's mean unreliability at confidence $1 - \alpha$ (Sec. 6).

**RQ3 (downstream value).** Does calibrated confidence actually make a better pruning signal than the alternatives a splat asset already carries? On a selection-quality metric — mean retained held-out reliability across pruning budgets — calibrated confidence lands within 1–4% of an oracle ceiling and beats opacity, the raw heuristic, and random at every budget, on every scene and under both reliability labels (Sec. 7).

We then place the confidence layer in the wider asset it belongs to (Sec. 10): the typed-carrier format, the Beta-kernel quality backend (whose +0.33 dB *Truck* result we report as a reproduction of Deformable Beta Splatting [3], not a claim of ours), and the two export paths that deliver the calibrated confidence. Throughout, we are explicit about what is validated and what is not (Sec. 11): AURA is preview-stage, the reliability labels are self-computed re-projection proxies rather than human annotations, the evaluation is local — machine-checked against committed artifacts (Sec. 9) but not yet externally reproduced — and the scenes are single captures. The contribution is not a leaderboard number; it is a calibrated, certified, exportable per-carrier confidence, and an honest measurement of it on real data.

**Contributions.**

1. **A held-out per-carrier reliability label** from multi-view re-projection agreement, in two variants — an occlusion-blind colour label and an occlusion-aware depth label with a coarse front-surface $z$-buffer — and the finding that an export-time train-agreement feature predicts it ($r = 0.91$–$0.98$) while the shipped view-count heuristic and opacity do not.
2. **Isotonic calibration of export-time confidence**, cutting ECE from 0.46–0.59 to 0.0006–0.0016 (∼300–900×) on four real scenes, with a monotone map that preserves the raw ordering.
3. **A distribution-free split-conformal pruning certificate** for splat assets: the most inclusive threshold whose retained set has mean unreliability $\leq \epsilon$ at confidence $1 - \alpha$, via a Hoeffding upper confidence bound on a bounded loss.
4. **A downstream selection-quality evaluation** showing calibrated confidence within 1–4% of the oracle ceiling and dominating opacity (at or below random) at every pruning budget; at a 10%-keep budget it retains 0.77–0.90 reliability against opacity's 0.31–0.49.
5. **An interchange path** that ships the calibrated confidence as a first-class channel on three standards: the `_AURA_CONFIDENCE` attribute of `KHR_gaussian_splatting` [4], the `primvars:aura:confidence` primvar on the OpenUSD 26.03 `UsdVolParticleField3DGaussianSplat` schema [5], and a `.spz.confidence` sidecar for Niantic's SPZ v4 container [6] (its writer cross-validated bit-exact against the reference implementation), with round-trip tests.
6. **A hardening study** on the same four scenes (Sec. 9): a single calibrator transfers across all 24 scene pairs for selection (AUC within ±0.0004) with gracefully degrading ECE and a per-scene conformal set that restores the certificate; the properties persist at native full resolution — correcting a train/test overlap that had made the core certificate mildly optimistic; and they survive a stricter render-loss label read from the actual alpha composite, with honestly weaker margins.
7. **A certified level-of-detail / streaming plan** on the same conformal machinery (Sec. 8): a consumer streams carriers by descending calibrated confidence and stops at any of $K$ published levels with a distribution-free, family-wise (Bonferroni $\alpha' = \alpha/R$ over the $R$ non-trivial levels; the full-keep level is deterministic) bound on the reliability mass discarded — all 16 bounds (four scenes × four levels) hold on disjoint evaluation halves.
8. **Systems maturation of the asset** (Sec. 10): a BVH-accelerated batched ray query at bit-for-parity with brute force (0 mismatches on the real 129k-carrier *Truck*), a maturity-typed carrier registry (*trained*/*demo*/*metadata*) enforced by a gate with an explicit provenance-annotated

[1] We use *carrier* for a typed primitive; for the assets in this paper a carrier is a Gaussian or a Beta splat. The confidence machinery is carrier-type agnostic.

fallback, and a codebook that compresses per-carrier semantic features (1.53 GB → 1.05 MB at $k$=64 on real DINOv2 *Truck* features).

9. **A machine-checked verification apparatus** (Sec. 9.4): 17 content-checked publication gates whose thresholds are derived from committed artifacts, a split guard that makes the leak class above mechanically impossible, continuous integration on every push, and a CPU-only, bit-for-bit reproduction from a fresh clone (`REPRODUCE.md`).

# 2 Related Work

**Gaussian splatting and its confidence heuristics.** 3DGS [1] and its open toolchains [2] represent a scene as primitives whose only native trust signals are opacity and, at densification time, gradient/view statistics. Deformable Beta Splatting (DBS) [3] replaces the Gaussian kernel with a bounded Beta kernel for higher fidelity at fewer primitives; it is the quality backend AURA uses for Beta carriers, and the source of the reproduction result in Sec. 10. Neither ships a calibrated per-primitive confidence.

**Pruning and per-splat confidence.** Splat pruning is an active area, but existing signals are either uncalibrated or not exported. PUP 3D-GS [7] prunes by a principled sensitivity/uncertainty score computed from a Hessian, improving compression, but the score is an internal training-time quantity, not a calibrated confidence attached to the shipped asset. Confident Splatting [8], the closest work, learns a per-splat confidence as a Beta distribution and prunes low-confidence splats; the confidence is trained end-to-end for compression, is Gaussian-only, and is not calibrated against a held-out reliability nor exported with a guarantee. Closest to our *label* is WarpRF [9], which quantifies radiance-field uncertainty training-free from multi-view warping/reprojection consistency — the same signal our reliability label uses (Sec. 3.1) — but uses it for active view selection rather than calibrating, certifying, or exporting a per-carrier confidence. More broadly, uncertainty quantification for radiance fields — a Bayesian–Laplace perturbation field over a trained NeRF (Bayes' Rays [10]) or a Fisher-information criterion for view selection (FisherRF [11]) — estimates *where* a reconstruction is unsure, but yields a spatial or view-level uncertainty for active capture and artifact removal, not a per-carrier confidence calibrated to a held-out reliability rate, certified for pruning, and shipped with the asset. Our contribution is orthogonal and complementary: a post-hoc, distribution-free calibration of *whatever* export-time signal a carrier has, plus a pruning certificate and an interchange channel. We do not claim a better compressor; we claim a trustworthy, exportable number.

**Calibration.** Isotonic regression via pool-adjacent-violators (PAVA) [12] is the classical monotone, non-parametric calibrator, used by Zadrozny and Elkan [13] to turn classifier scores into probabilities; expected calibration error [14] is the standard diagnostic. We apply isotonic calibration per carrier: the map is monotone, so it never inverts the raw ordering (a more-agreeing carrier never becomes less confident) while making the value a reliability.

**Conformal prediction and risk control.** Split-conformal prediction [15, 16] turns any score into a distribution-free guarantee from a held-out calibration set. Conformal risk control [17] and distribution-free risk-controlling prediction sets [18] extend this to bound the expectation of a bounded loss. Our pruning certificate is an instance: the loss is per-carrier unreliability, and we report the most inclusive threshold whose retained-set risk is provably bounded, using a Hoeffding upper confidence bound [19] on the $[0, 1]$ loss.

**Standards for splat interchange.** The `KHR_gaussian_splatting` glTF 2.0 extension [4] (release candidate, 2026) standardizes splat delivery with `POSITION`/`ROTATION`/`SCALE`/`OPACITY` attributes and a vendor-attribute mechanism, and OpenUSD 26.03 [5] adds a native `UsdVolParticleField3DGaussianSplat` schema that renderers such as Omniverse RTX path-trace natively [20]. AURA rides both to carry its calibrated confidence as a first-class channel; the export is a delivery path, not a benchmarked claim.

# 3 Method

AURA attaches a trustworthy per-carrier confidence to an already-trained splat asset in three post-processing stages, all pure-CPU over the exported carrier array (means, colours, opacity, and a raw confidence). Nothing here retrains the scene or touches the renderer. Figure 1 overviews the layer end to end; Fig. 4 and Fig. 7 are computed by exactly the pipeline described below.

## 3.1 A held-out per-carrier reliability label

We need a ground-truth-like answer to "is this carrier trustworthy?" that does not consult the carrier's own stored confidence. We derive one from *multi-view re-projection agreement*. Hold out every eighth captured frame (the `llffhold` convention). For each carrier $i$ with centre $\mu_i$ and stored colour $c_i$, project $\mu_i$ into every held-out camera; where it lands in-frame and in front, sample the ground-truth pixel colour. Let $\bar{c}_i$ be the robust (masked-median, over observed views only) observed colour. The reliability label is

$$y_i \;=\; \exp(-\kappa \,\|c_i - \bar{c}_i\|_2) \in [0, 1], \qquad (1)$$

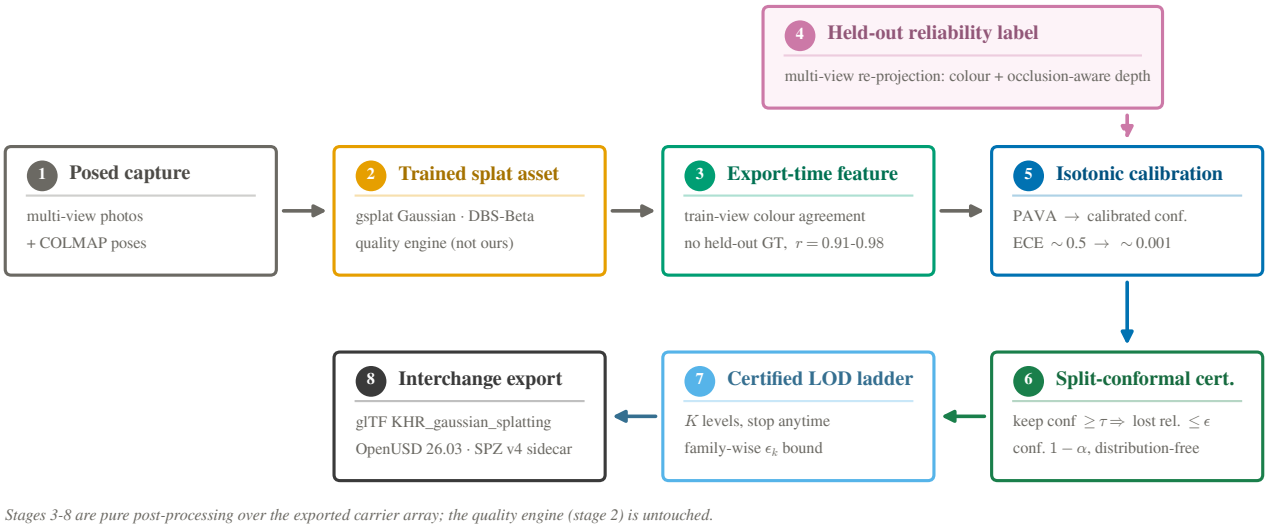


**Figure 1:** Method overview. A trained splat asset (stage 2, produced by an unmodified quality engine — gsplat or the DBS-Beta backend) gains a trustworthy confidence through three pure-CPU post-processing stages: an export-time train-agreement feature (3) is calibrated (5) against a held-out multi-view re-projection reliability label (4), then a split-conformal certificate (6) turns the calibrated confidence into a safe pruning threshold, extended into a certified level-of-detail ladder (7) and carried through standard interchange (8). Badges index the paper's sections; the diagram is a schematic redrawn programmatically (`figures/make_qual_figures.py`), not a rendered result.

with decay constant $\kappa = 4$ (we write $\kappa$, not $\beta$, to avoid collision with the Beta-kernel parameter of Sec. 10), and $y_i = 0$ for carriers never observed in the held-out set. A carrier that sits on a real, consistently observed surface agrees with what independent views see ($y_i \to 1$); a floater, a misplaced carrier, or one rarely seen disagrees ($y_i \to 0$). We label a carrier only when it is observed in at least three held-out views. The masked median (rather than a sentinel-filled median) is essential: it takes the centre over *only* the views in which the carrier is actually observed, so carriers seen in fewer than half the views are not poisoned.

**Two labels: colour and occlusion-aware.** The colour label above counts a carrier as observed in any held-out view it projects into, so an interior carrier occluded across those views samples the *occluding* surface's colour and scores a false low. We therefore also compute an occlusion-aware *depth* label. Before sampling, we build a coarse front-surface depth buffer: over an 8×8-pixel block grid, take the minimum camera-space $z$ of all sufficiently opaque in-frame carriers. A carrier counts as observed in a view only if it is the visible front surface there ($z_i \le z_{\text{front}}\,(1 + \text{tol})$, tol $= 0.03$), i.e. not hidden behind opaque content. Fewer views per carrier survive (the labelled fraction drops to 61–98% across scenes as floaters and occluded interiors are dropped), which is exactly why the masked median is required. The two labels bracket the reliability signal: the colour label is occlusion-blind, the depth label is a stricter, occlusion-resolved variant. Both are conservative — they under-credit rather than over-credit a carrier.

This reliability label is a self-computed re-projection proxy, not a human annotation or a photometric render loss; that is a real limitation, revisited in Sec. 11. But it is honest and cheap, it needs only the trained carriers and the held-out captures, and, crucially, it is disjoint from every signal we test against it.

## 3.2 An export-time feature

The reliability label of Eq. 1 consumes held-out ground truth, so it cannot itself be shipped. We need an export-time feature that predicts it from information available at asset-write time. We use *train-view colour agreement*: exactly the agreement statistic of Eq. 1 but computed over the *training* views the carrier was fit on, disjoint from the held-out views that define the label. Intuitively, a carrier whose stored colour already disagrees with the training pixels it should explain is an inconsistency or a floater; this is computable without any held-out data. We contrast it against the two signals a splat asset ships today: the view-count heuristic $1 - \exp(-n_{\text{views}}/12)$ (AURA's prior confidence, and a stand-in for the family of view-count squashes), and opacity (the engine's default pruning proxy).

### 3.3 Isotonic calibration

A raw feature is monotone but has no probabilistic meaning. We fit an isotonic calibrator $g$ : feature $\to [0, 1]$ by pool-adjacent-violators [12], the $L_2$-optimal non-decreasing fit, on a calibration split, against the reliability label. Because $g$ is monotone it preserves the feature's ordering while rescaling it so that carriers reported at confidence $\approx p$ are reliable $\approx p$ of the time. We measure calibration quality by expected calibration error over 15 equal-width bins [14],

$$\mathrm{ECE} = \sum_b \frac{|B_b|}{N} \left| \overline{\mathrm{conf}}_b - \overline{y}_b \right|, \tag{2}$$

the bin-weighted gap between mean confidence and mean reliability. The calibrated value replaces the carrier's raw confidence and is flagged `confidence_calibrated`, so downstream export ships the trustworthy number.

### 3.4 A split-conformal pruning certificate

Calibration makes the number meaningful; a certificate makes pruning *safe*. Given calibrated confidences and the held-out reliabilities on a calibration split, we want the most inclusive threshold $\tau$ such that keeping $\{i : \mathrm{conf}_i \geq \tau\}$ leaves a retained set whose mean *unreliability* $1 - y$ is at most a budget $\epsilon$, with distribution-free confidence $1 - \alpha$. For a retained set of size $m$ with empirical mean unreliability $\hat{R}$, a Hoeffding upper confidence bound [19] on the bounded $[0, 1]$ loss gives

$$R \;\leq\; \hat{R} + \sqrt{\tfrac{\log(1/\alpha)}{2m}} \;=\; \mathrm{UCB}, \tag{3}$$

finite-sample valid and distribution-free. Scanning thresholds from most inclusive upward, we return the first $\tau$ whose $\mathrm{UCB} \leq \epsilon$ (Angelopoulos et al. [17]; Bates et al. [18]). As in the RCPS "first-crossing" argument [18], this requires the retained risk $\hat{R}(\tau)$ to be monotone non-increasing in $\tau$ — a stricter threshold cannot make the kept set less reliable — so that the pointwise-valid bound at the first crossing controls risk without a multiplicity correction over the threshold grid; the risk curve is empirically monotone on every scene we report (Fig. 6), and the multi-level LOD ladder of Sec. 8 does apply an explicit family-wise split. The certificate reads: "drop everything below $\tau$, losing at most $\epsilon$ reliability mass, with confidence $1 - \alpha$" — an abstention guarantee attached to the asset itself. It never claims more than the held-out risk supports.

### 3.5 Selection-quality evaluation

Finally we measure whether calibrated confidence is a *good pruning signal*, independent of the certificate's threshold. For a candidate score $s$ (higher = keep first), sort carriers by $s$, and at each keep-fraction $f \in \{0.1, \ldots, 1.0\}$ record the mean held-out reliability of the kept set. This selection curve, and its budget-averaged area (AUC), say how much true reliability a consumer retains when it prunes by $s$. We compare calibrated confidence against opacity, the raw view-count heuristic, random, and an *oracle* that sorts by the true held-out reliability — the achievable ceiling. All quantities are read on an evaluation split disjoint from the calibration split (a seed-fixed 50/50 partition of the labelled carriers), so no carrier trains the calibrator that later scores it.

| Scene | dataset | carriers | train | held-out | lbl. (c/d) |
|---|---|---|---|---|---|
| Truck | T&T | 129,531 | 219 | 32 | 128k / 79k |
| Garden | Mip-360 | 120,000 | 161 | 24 | 113k / 75k |
| Kitchen | Mip-360 | 120,000 | 244 | 35 | 120k / 118k |
| Room | Mip-360 | 106,809 | 272 | 39 | 107k / 78k |

**Table 1:** The four real scenes. "lbl. (c/d)" is the number of carriers labelled under the colour / depth-aware label (a carrier is labelled if observed in $\geq 3$ held-out views; the depth label drops occluded interiors, so fewer survive).

## 4 Experimental Setup

**Scenes.** We evaluate on four real scenes (Fig. 2): *Truck* from Tanks-and-Temples [21] and *Garden*, *Kitchen*, *Room* from Mip-NeRF 360 [22] (one outdoor, two indoor). Each is trained no-densify (densification balloons large Mip-360 scenes past ~11M carriers and makes the asset write impractical), yielding 107k–130k carriers per scene, at 0.25× resolution. Table 1 lists the per-scene carrier counts and the train / held-out view split. Every eighth view is held out for labelling; the rest train.[2] Mip-360 evaluation is at image downsample; Sec. 9.2 re-runs all four scenes at native full resolution and the conclusions hold (Sec. 11).

**Protocol.** Per scene and per label we run `per_carrier_reliability.py` to produce the held-out reliability label, the train-agreement feature, the view-count heuristic, and opacity; then `calibrate_confidence.py` fits the isotonic calibrator and the certificate on a seed-0, 50/50 calibration/evaluation split of the labelled carriers and reports ECE, the certificate, and the selection curves. The calibration and certificate core is `src/aura/calibration.py` (10 CPU unit tests). All figures in this paper regenerate from the committed reliability arrays through this same core; the reproduction is bit-exact against the committed reports.

[2]The committed core carriers were in fact trained on *all* frames, so the every-eighth "held-out" views were seen during training — a protocol leak. Sec. 9.2 re-trains train-split-only (genuinely disjoint) and finds the correlation and ECE unchanged, the absolute certificate and selection AUC mildly optimistic; we report both and mark the corrected values.


**Figure 2:** The four evaluated scenes, shown as source capture photographs (one held-out frame each): *Truck* from Tanks-and-Temples and *Garden*, *Kitchen*, *Room* from Mip-NeRF 360 (one outdoor, two indoor). Per-scene carrier counts are from Table 1. Generated by `figures/make_qual_figures.py` from the committed capture manifests.

| corr. with held-out reliability | Truck | Garden | Kitchen | Room |
|---|---|---|---|---|
| **train-agreement (export feat.)** | **0.91** | **0.93** | **0.98** | **0.96** |
| view-count heuristic (shipped) | −0.05 | −0.13 | −0.01 | 0.05 |
| opacity (engine default) | −0.18 | 0.16 | 0.08 | 0.05 |
| *under the occlusion-aware depth label:* | | | | |
| **train-agreement (export feat.)** | **0.75** | **0.90** | **0.97** | **0.94** |
| opacity (engine default) | −0.15 | 0.20 | 0.12 | 0.08 |

**Table 2:** Pearson correlation of each per-carrier signal with the held-out reliability label. The export-time train-agreement feature predicts reliability; the shipped view-count heuristic and opacity do not. Rank (Spearman) correlation for the feature is comparable ($\rho$ = 0.91–0.98, colour label), so the relationship is genuinely monotone rather than an artifact of linear scale — consistent with the rank-preserving isotonic calibration and AUC-based selection used downstream. Under the stricter occlusion-aware label the feature still predicts ($r$ = 0.75–0.97; lowest on *Truck*, whose many floaters make the depth label sparsest).

# 5 RQ1: Which Signal Predicts Reliability?

The premise of the whole layer is that *some* export-time signal tracks held-out reliability while the shipped ones do not. Table 2 and Fig. 3 settle it. The train-view colour-agreement feature correlates with held-out reliability at $r$ = 0.91–0.98 (colour label) on all four scenes. The view-count heuristic is uninformative ($|r| \leq 0.13$): on densely captured scenes it saturates near 1 and cannot discriminate. Opacity is the honest surprise. It is *negatively* correlated on *Truck* (−0.18) and mildly positive on the Mip-360 scenes (+0.05 to +0.16) — it has no consistent sign, let alone a useful magnitude. The robust, scene-independent statement is the selection-AUC result of Sec. 7: opacity is a poor pruning signal everywhere.

The occlusion-aware label confirms rather than weakens the finding: the feature still predicts reliability at $r$ = 0.75–0.97 (Table 2, lower block). The one softening is *Truck* (0.91 → 0.75), whose many floaters make the depth-labelled set the sparsest (61% of carriers), leaving a noisier fit. The answer to RQ1 is unambiguous: a feature computable at export time predicts a carrier's held-out reliability, and the numbers the asset ships today do not.

| ECE (eval split) | Truck | Garden | Kitchen | Room |
|---|---|---|---|---|
| raw view-count heuristic | 0.586 | 0.551 | 0.557 | 0.464 |
| **calibrated** | **0.0014** | **0.0015** | **0.0006** | **0.0016** |
| raw (depth label) | 0.620 | 0.537 | 0.550 | 0.471 |
| **calibrated (depth label)** | **0.0037** | **0.0017** | **0.0009** | **0.0024** |

**Table 3:** Expected calibration error, raw shipped heuristic vs. isotonically calibrated confidence, on the held-out evaluation split. Calibration cuts ECE by ∼300–900×, under both reliability labels.

# 6 RQ2: Calibration and the Certificate

## 6.1 Isotonic calibration

A predictive feature is not yet a confidence. Table 3 reports ECE on the disjoint evaluation split. The raw view-count heuristic is grossly miscalibrated (ECE 0.46–0.59): its value has nothing to do with the reliability rate. The train-agreement feature is already far better uncalibrated (ECE 0.005–0.016; per-scene values recorded in the committed `calib_<scene>.json` as `ece_feature_uncalibrated`) because it is a colour-space quantity on the same scale, and isotonic calibration finishes the job, reaching ECE 0.0006–0.0016 — a ∼300–900× reduction over the heuristic. Figure 4 shows the reliability diagrams: the calibrated confidence lies on the diagonal, the raw heuristic wanders far from it. Figure 5 renders that calibrated confidence spatially on a held-out *Truck* view, where it varies across the scene while the view-count heuristic would saturate. Under the occlusion-aware label the calibrated ECE is 0.0009–0.0037, from raw 0.47–0.62 — the same verdict.

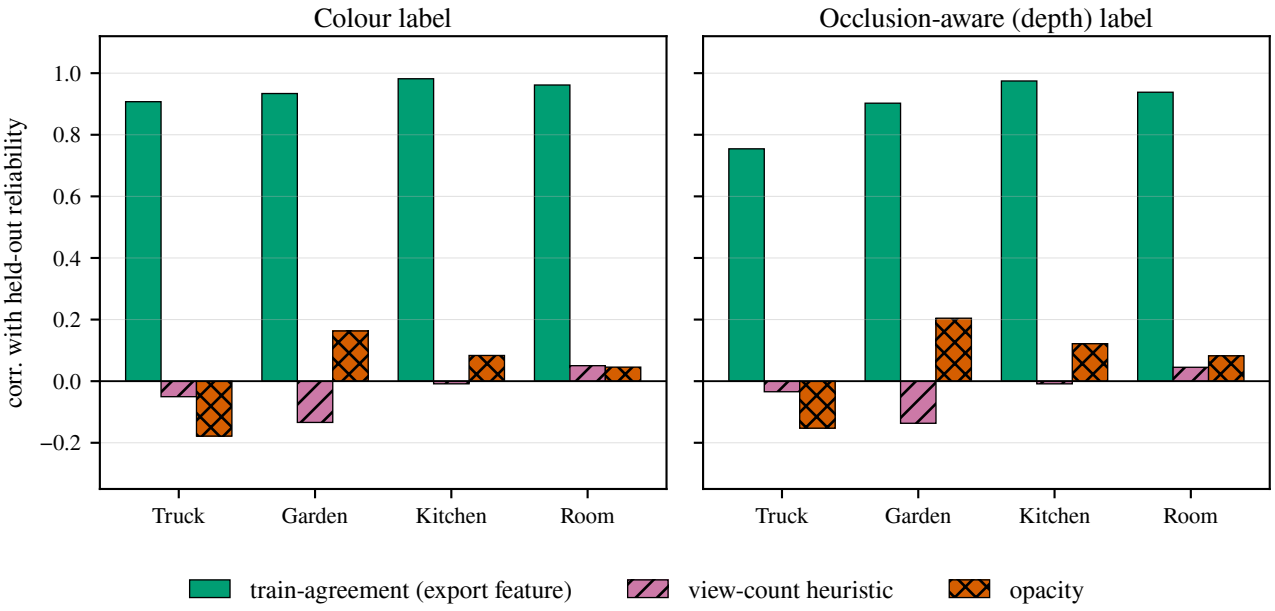


**Figure 3:** RQ1: correlation of each per-carrier signal with held-out reliability, per scene, under the colour label (left) and the occlusion-aware depth label (right). The export-time train-agreement feature (green) tracks reliability everywhere; the view-count heuristic (pink) and opacity (orange) do not, opacity flipping sign across scenes.

### 6.2 The pruning certificate

Table 4 reports the split-conformal certificate at budget $\epsilon = 0.6$, $\alpha = 0.1$ (Eq. 3). At this loose budget the trained scenes are reliable enough that keeping the full set is already certified (kept fraction 1.0 on all four colour-label scenes; the retained mean unreliability, 0.47–0.59, sits under $\epsilon$ with the Hoeffding margin). The one scene that is forced to prune is *Truck* under the occlusion-aware label, where the certificate certifies keeping 90.3% at $\tau = 0.13$. This is honest about what the certificate is: at a loose $\epsilon$ it confirms the asset is already good enough to keep whole; its purpose is realized when a consumer demands a *tighter* bound, which trades kept fraction for reliability. Figure 6 sweeps $\epsilon$ and shows exactly that trade — as $\epsilon$ tightens, the certified kept fraction falls smoothly while the retained-set risk it guarantees drops with it. The certificate is distribution-free: it never certifies a kept fraction the held-out risk does not support. *Correction:* the carriers behind Table 4 were trained on all frames, so the evaluation split overlapped training (the leak of Sec. 4); a leak-corrected re-run at full resolution (Sec. 9.2) leaves correlation and ECE essentially unchanged but lowers the *Truck* colour-label certificate from the 1.00 shown here to 0.77 (*Garden*, *Kitchen*, *Room* still certify 1.00). We treat 0.77 as the corrected *Truck* value rather than silently keep the optimistic one.

| | colour label | | depth label | |
|---|---|---|---|---|
| Scene | kept | risk | kept | risk |
| Truck | 1.00$^{\dagger}$ | 0.593 | **0.90** | 0.591 |
| Garden | 1.00 | 0.574 | 1.00 | 0.561 |
| Kitchen | 1.00 | 0.557 | 1.00 | 0.550 |
| Room | 1.00 | 0.471 | 1.00 | 0.477 |

**Table 4:** Split-conformal pruning certificate at $\epsilon = 0.6$, $\alpha = 0.1$. "kept" is the certified kept fraction; "risk" is the empirical mean unreliability of the kept set (bounded above by $\epsilon$ with confidence $1 - \alpha$ via the Hoeffding UCB). At this loose budget most scenes certify keeping the whole asset; *Truck* under the occlusion-aware label certifies 90.3%. Figure 6 sweeps $\epsilon$. $^{\dagger}$The *Truck* colour-label carriers were trained on all frames, so this entry is mildly optimistic; the leak-corrected full-resolution value is **0.77** (Sec. 9.2), which we treat as authoritative.

## 7 RQ3: Calibrated Confidence Is a Near-Oracle Pruning Signal

The payoff is the selection-quality result. Table 5 reports the budget-averaged retained reliability (AUC) for each pruning signal, and Fig. 7 plots the full curves. Three readings matter.

First, calibrated confidence is near-oracle. Its AUC (0.581/0.605/0.629/0.720) sits within 1–4% of the oracle ceiling (0.601/0.619/0.633/0.729) on all four scenes — it recovers almost all of the selection quality achievable if the true held-out reliability were known. Sec-

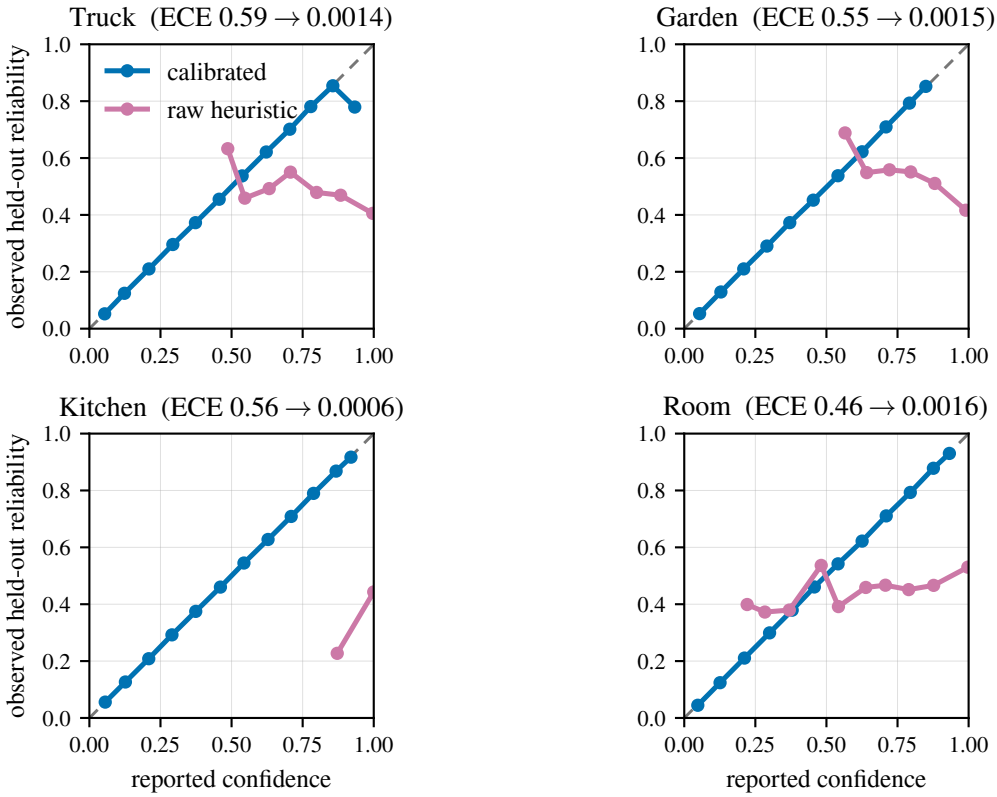


**Figure 4:** RQ2 (calibration): reliability diagrams on the evaluation split. Perfect calibration is the diagonal. The isotonically calibrated confidence (blue) lies on it; the raw view-count heuristic (pink) does not. Each title gives the raw→calibrated ECE.

ond, opacity is a poor pruning signal everywhere: its AUC (0.367/0.450/0.458/0.534) is at or below random (0.406/0.426/0.442/0.528) on every scene, and the raw heuristic is no better. This is the robust, scene-independent form of the RQ1 opacity observation: whatever its correlation sign, opacity does not rank carriers by reliability. Third, the gap is largest exactly where pruning is most aggressive. At a 10%-keep budget — drop 90% of the asset — calibrated confidence retains 0.77–0.90 mean reliability against opacity's 0.31–0.49 (Table 5, last two rows). A consumer that keeps a tenth of the carriers by calibrated confidence holds onto nearly all the reliable ones; by opacity it does no better than chance. Figure 8 shows this trade qualitatively at a 30%-keep budget on *Room*: opacity preserves the render but keeps unreliable carriers, calibrated confidence keeps the reliable ones.

**Robustness to the occlusion-aware label.** Under the stricter depth label the verdict holds: calibrated AUC is 0.523/0.616/0.639/0.713 against an oracle of 0.575/0.638/0.645/0.727 (within 1–9%, the 9% again on floater-heavy *Truck*) and opacity 0.342/0.459/0.475/0.531, at or below random on every scene. Resolving occlusion in the label does not change which signal ranks carriers by reliability.

| selection signal (AUC) | Truck | Garden | Kitchen | Room |
|---|---|---|---|---|
| **calibrated confidence** | **0.581** | **0.605** | **0.629** | **0.720** |
| oracle ceiling | 0.601 | 0.619 | 0.633 | 0.729 |
| raw heuristic | 0.426 | 0.404 | 0.439 | 0.552 |
| random | 0.406 | 0.426 | 0.442 | 0.528 |
| opacity (engine default) | 0.367 | 0.450 | 0.458 | 0.534 |
| calibrated @ 10%-keep | **0.770** | **0.797** | **0.836** | **0.896** |
| opacity @ 10%-keep | 0.306 | 0.450 | 0.470 | 0.489 |

**Table 5:** Selection quality: mean retained held-out reliability averaged over pruning budgets (AUC; higher is better), colour label. Calibrated confidence is within 1–4% of the oracle ceiling and beats opacity (at or below random) everywhere. The last rows give the 10%-keep operating point.

**Reading the numbers honestly.** The AUC is a mean of retained reliability, which is bounded by the scene's overall mean reliability (~ 0.41–0.53); this is why even the oracle sits

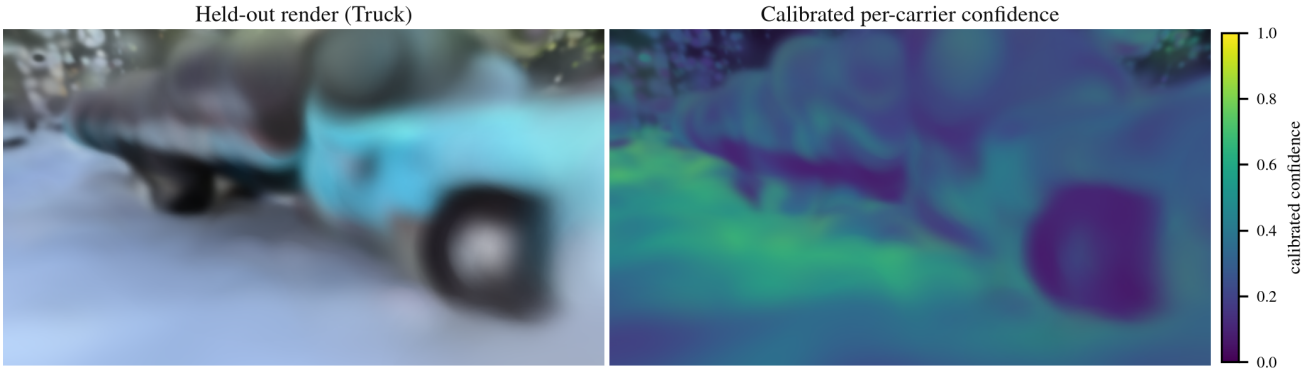


**Figure 5:** RQ2 (qualitative): the calibrated per-carrier confidence, made spatial, on a held-out *Truck* view. Left: the RGB render of the core carriers. Right: the same carriers coloured by their *calibrated* confidence — the isotonic map fit exactly as `calibrate_confidence.py` (PAVA on the train-agreement feature, seed-0 disjoint half of `reliability_truck.npz`) and then applied to every carrier. Higher confidence (yellow/green) concentrates on the consistently observed ground plane; a distinct low-confidence (purple) pocket marks poorly-observed content. This is the calibrated confidence, *not* the shipped view-count heuristic (which saturates near 1 almost everywhere and is uninformative, Sec. 5). Generated by `figures/make_qual_figures.py`.

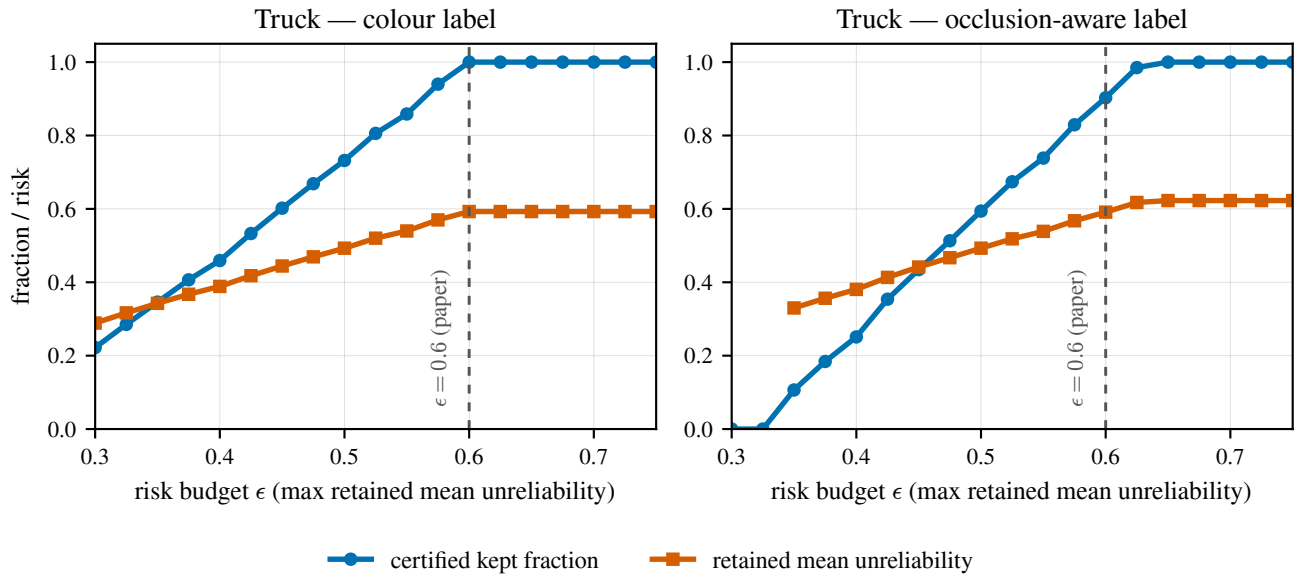


**Figure 6:** RQ2 (certificate): the operating curve on *Truck*, colour label (left) and occlusion-aware label (right). As the risk budget $\epsilon$ tightens, the distribution-free certificate keeps a smaller certified fraction (blue) whose guaranteed retained unreliability (orange) drops with it. The dashed line marks the $\epsilon = 0.6$ operating point of Table 4.

at 0.60–0.73 rather than near 1. The meaningful comparison is therefore calibrated-vs-oracle (the achievable gap, 1–4%) and calibrated-vs-random/opacity (the realized gain, e.g. *Truck* 0.581 vs 0.406/0.367). We report all baselines so the metric cannot be misread as an accuracy. The absolute AUC and 10%-keep figures here are the leaked split's mildly optimistic precursors; Sec. 9.2 re-measures them clean at full resolution, where the relative gaps — near-oracle, dominating opacity — are unchanged or slightly larger.

# 8 Certified Level-of-Detail Streaming

The certificate of Sec. 3.4 answers one question — may a consumer drop everything below a single threshold? — and

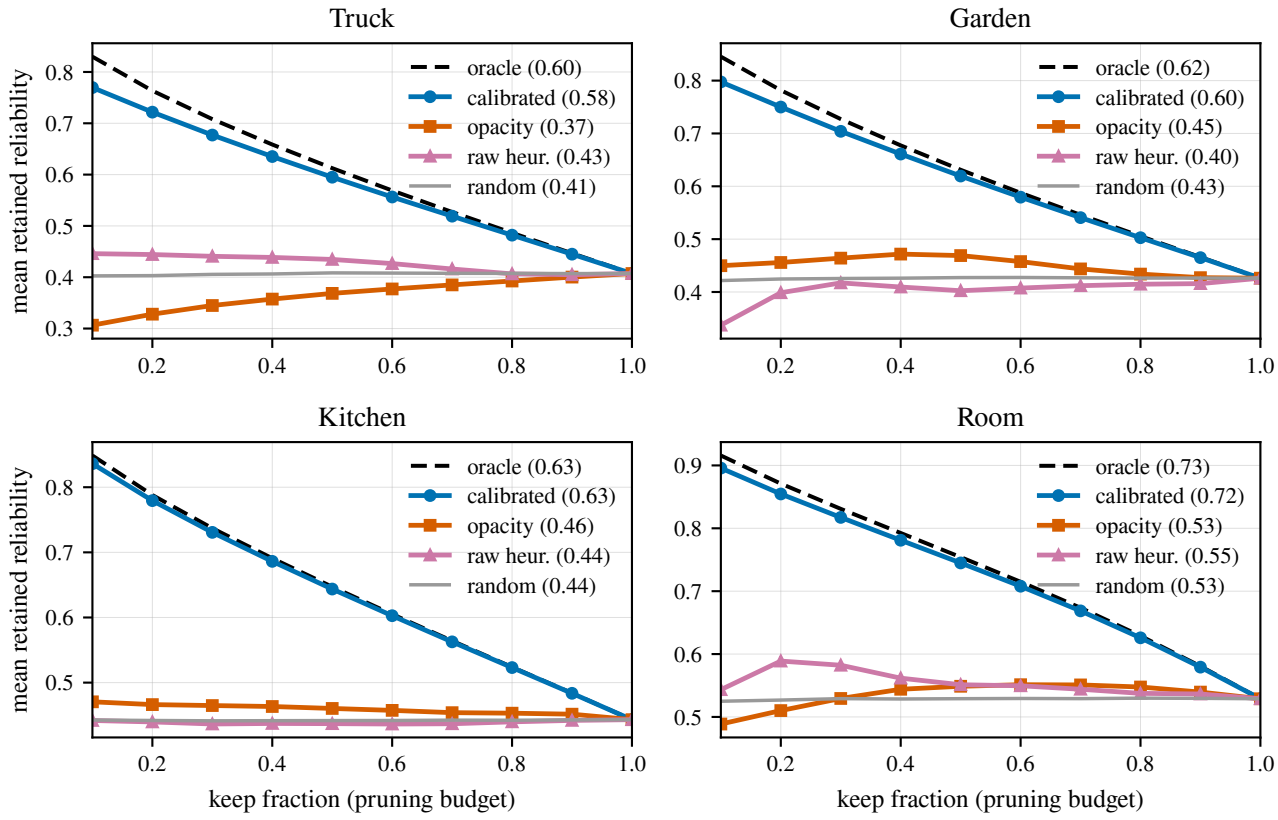


**Figure 7:** RQ3: mean retained reliability vs. keep fraction, per scene. Calibrated confidence (blue) tracks the oracle ceiling (dashed) at every budget; opacity (orange), the raw heuristic (pink), and random (grey) stay flat and far below. Budget-averaged AUCs are in the legend and Table 5. Curves regenerate bit-exactly from the committed reliability arrays.

at a loose budget the answer is usually "keep the whole asset" (Table 4). A streaming or level-of-detail consumer asks a sharper question: it receives carriers progressively and wants to *stop early*, at a chosen fidelity, with a stated bound on what stopping has cost it. We turn the one certificate into an ordered, multi-level plan that answers exactly this, on the same split-conformal machinery. This is the capability a bare 3DGS/DBS splat cannot offer — a level-of-detail ladder whose every rung carries a finite-sample guarantee rather than a heuristic.

**The plan.** Sort the carriers by *calibrated* confidence, descending (stable tiebreak by index); this is the streaming order. Publish $K$ keep-fractions $f_1 < \cdots < f_K$ (default $\{0.10, 0.25, 0.50, 1.00\}$); level $k$'s boundary threshold $\tau_k$ is the confidence of the round$(f_k n)$-th carrier, and its retained set is $\{i : \mathrm{conf}_i \geq \tau_k\}$, non-increasing in $f_k$. A consumer that stops at level $k$ keeps the top $f_k$ and discards the rest; what it forfeits is the reliability of the discarded carriers.

**The certified quantity.** Define the per-carrier discard loss over the conformal set,

$$\ell_i^k \;=\; \mathrm{reliability}_i \cdot \mathbf{1}[\,\mathrm{conf}_i < \tau_k\,] \;\in\; [0,1], \qquad (4)$$

whose mean $\frac{1}{n}\sum_i \ell_i^k$ is the per-capita reliability mass thrown away by pruning to level $k$: zero at $f_k = 1.00$, growing as the stream is cut shorter, and small exactly when the discarded carriers are unreliable — the calibrated-confidence promise. Because $\ell^k$ is a bounded $[0,1]$ loss, its mean is bounded by the *same* Hoeffding upper confidence bound the pruning certificate uses (Eq. 3), evaluated on the calibration half:

$$\epsilon_k \;=\; \widehat{\mathrm{mean}}_{\mathrm{cal}}(\ell^k) + \sqrt{\frac{\log(1/\alpha')}{2\,n_{\mathrm{cal}}}}. \qquad (5)$$

The two certificates share one audited bound (`conformal_mean_upper_bound`), factored out so the pruning certificate and the streaming ladder cannot drift apart.

**Family-wise validity.** A $K$-level plan makes $K$ simultaneous statements read from one conformal set, so a naive per-level $1-\alpha$ would be optimistic. But only the $R$ *non-trivial*


**Figure 8:** RQ3 (qualitative): pruning to 30% of carriers on a held-out *Room* view. Left, the full asset; centre, keeping the top 30% by *calibrated confidence*; right, keeping the top 30% by *opacity* (the engine default). Each panel is annotated with its render PSNR against the ground-truth image and, for the pruned panels, the mean held-out reliability of the kept set. This single view illustrates the aggregate trade-off of Fig. 7 and Table 9: opacity — being the alpha-composite blend weight — preserves the render (22.7 dB vs. confidence's 18.7 dB) but keeps unreliable carriers (retained reliability 0.53), while calibrated confidence keeps the reliable ones (0.82) at the cost of PSNR. This is the 30%-keep operating point of those curves; it is *not* a PSNR claim (Sec. 9.3: opacity is the render-preserving prune). Committed artifact `repo/assets/pruning_30pct.png`, generated by `experiments/make_pruning_sweep_gif.py` (calibrator fit exactly as `calibrate_confidence.py`, seed 0).

levels ($f_k < 1$) are *random* certificates that can miscover; the $f_K = 1.00$ level is a *deterministic* statement ($\epsilon_K = 0$: nothing is pruned, so the discarded mass is exactly zero with no sampling uncertainty), flagged as such, and it can never fail. By the union bound the family-wise error is at most the sum of the per-level errors, and the deterministic level contributes exactly 0, so it needs no error budget. We therefore split $\alpha$ over the $R$ random levels only, certifying each at $\alpha' = \alpha/R$; then $R$ random bounds at $\alpha'$ union-bound to $R\alpha' = \alpha$ and, together with the deterministic level, all $K$ bounds hold *simultaneously* with family-wise confidence $1 - \alpha$. This is strictly tighter than the naive $\alpha/K$ (correcting over the trivial level would waste budget on a statement that never fails): with the default $K = 4$ levels, $R = 3$, so $\alpha' = \alpha/3 \approx 0.0333$ rather than $\alpha/4 = 0.025$, giving smaller $\epsilon_k$ at the same honest guarantee. Bonferroni still widens each random interval relative to an uncorrected $1 - \alpha$ ($\log(R/\alpha) > \log(1/\alpha)$) — the honest direction.

**All published bounds hold.** Table 6 builds each scene's plan on its seed-0 calibration half — the identical split of Sec. 3.5, so the evaluation half never touches plan construction — at $\alpha = 0.1$ family-wise, $\alpha' = \alpha/3 \approx 0.0333$ over the $R = 3$ non-trivial levels (the fourth, $f = 1.00$, level is deterministic), and measures the empirical discarded mass on the disjoint evaluation half. All 16 bounds hold (12 non-trivial, 4 trivial), on every scene and level. The Hoeffding margins are small ($\sim 0.005$–$0.006$ at $n_{\text{cal}} \approx 5$–$6 \times 10^4$) yet always suffice, because the two halves are exchangeable draws of one scene, so the eval discarded mass sits just below the cal-half bound at every rung; both $\epsilon_k$ and the empirical loss fall monotonically to zero as $f_k \to 1$. The numbers read correctly only with the scale in mind: keeping a tenth of *Room*'s carriers discards $\epsilon \approx 0.44$ of the scene's total reliability mass even though each *kept* carrier is individually very reliable (Sec. 7: 0.77–0.90 retained reliability at the same 10%-keep). Both are true — there are many moderately reliable carriers, so a hard prune forfeits much of the aggregate mass — and the certificate bounds that forfeited mass honestly rather than hiding it.

**Remark: plateau-valued thresholds.** $\tau_k$ is stored at full precision, not rounded. The isotonic calibrator produces plateaus (many carriers share a knot value) and a level boundary frequently lands on one — on *Truck*, 1,240 evaluation carriers share the exact boundary confidence. Rounding $\tau_k$ up then moves the conf $< \tau_k$ cut across an entire plateau at once, silently reclassifying that whole block from kept to dropped; an intermediate implementation that rounded to six decimals produced three spurious "violations" by exactly this mechanism. The fix is to store $\tau$ at full precision and round only for display — a genuine correctness subtlety of plateau-valued calibrators, now regression-tested.

**Scope.** Three boundaries, all inherited from the certificate and stated with it. (i) $\epsilon_k$ bounds a *reliability-mass* proxy, not rendered PSNR: as Sec. 9.3 establishes, opacity — not reliability — is the render-preserving prune signal, so a certified stream trades on the *trustworthiness* of the retained carriers, not on preserving the alpha-composited image; the opacity caveat of Sec. 9.3 is not repaired here. (ii) The guarantee is *family-wise*, not per-level: $1-\alpha$ covers the whole $K$-level plan, and a single level's $\epsilon_k$ is not an independent $1 - \alpha$ bound. (iii) It is distribution-free but *exchangeability*-

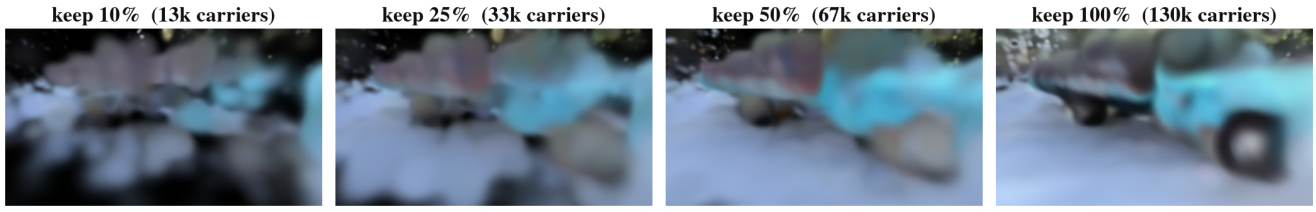


**Figure 9:** The certified level-of-detail ladder rendered, on a held-out *Truck* view. Each panel keeps the top carriers by *calibrated* confidence at one of the four published levels ($\{10, 25, 50, 100\}$%); as the stream is cut shorter, more (low-confidence) carriers are dropped and the render visibly thins. Under each panel are the family-wise-corrected certified bound $\epsilon_k$ and the empirical discarded-reliability mass on the disjoint evaluation half — the *Truck* row of Table 6, which the generator re-derives from `reliability_truck.npz` and asserts bit-equal to the committed `lod_certified.json` before rendering. All four bounds hold; $\epsilon_k$ bounds discarded *reliability mass*, not rendered PSNR (Sec. 9.3). Generated by `figures/make_qual_figures.py`.

| Scene | keep | $\tau$ | $\epsilon_k$ (cert.) | emp. loss | holds |
|---|---|---|---|---|---|
| Truck | 0.10 | 0.7203 | 0.3340 | 0.3281 | ✓ |
| | 0.25 | 0.5977 | 0.2360 | 0.2320 | ✓ |
| | 0.50 | 0.3939 | 0.1102 | 0.1051 | ✓ |
| | 1.00 | 0.0019 | 0.0000 | 0.0000 | ✓† |
| Garden | 0.10 | 0.7427 | 0.3472 | 0.3401 | ✓ |
| | 0.25 | 0.6180 | 0.2451 | 0.2377 | ✓ |
| | 0.50 | 0.4193 | 0.1224 | 0.1164 | ✓ |
| | 1.00 | 0.0021 | 0.0000 | 0.0000 | ✓† |
| Kitchen | 0.10 | 0.7761 | 0.3641 | 0.3598 | ✓ |
| | 0.25 | 0.6340 | 0.2544 | 0.2511 | ✓ |
| | 0.50 | 0.4425 | 0.1255 | 0.1216 | ✓ |
| | 1.00 | 0.0018 | 0.0000 | 0.0000 | ✓† |
| Room | 0.10 | 0.8561 | 0.4429 | 0.4390 | ✓ |
| | 0.25 | 0.7420 | 0.3176 | 0.3117 | ✓ |
| | 0.50 | 0.5597 | 0.1596 | 0.1514 | ✓ |
| | 1.00 | 0.0010 | 0.0000 | 0.0000 | ✓† |

**Table 6:** *(See Fig. 9 for the* Truck *row rendered.)* Certified level-of-detail plan: all 16 published bounds hold on the disjoint evaluation half. Each plan is built on the seed-0 calibration half ($n_{\text{cal}}$: *Truck* 63.9k, *Garden* 56.6k, *Kitchen* 60.0k, *Room* 53.4k; mean evaluation reliability 0.41/0.43/0.44/0.53). $\tau$ is the level-boundary calibrated confidence; $\epsilon_k$ is the family-wise-corrected certified bound ($\alpha = 0.1$, Bonferroni $\alpha' = \alpha/3 \approx 0.0333$ over the $R = 3$ non-trivial levels; the $f = 1.00$ level is deterministic); "emp. loss" is the empirical discarded reliability mass on the evaluation half; the bound holds when emp. loss $\leq \epsilon_k$. † trivial level ($f = 1.00$: nothing pruned, $\epsilon = 0$ by definition).

*dependent*: it assumes the calibration and streamed carriers are exchangeable (same scene, seed-0 split), and a distribution shift can violate it — cross-scene deployment therefore needs a small local conformal set per scene (Sec. 9.1), and the test suite includes a synthetic anti-correlated case where the bound honestly fails.

# 9 Hardening: Transfer, Full Resolution, and a Render Loss

The core result (Secs. 5–7) fits and evaluates a calibrator *per scene*, at the tractable 0.25× training resolution, against the re-projection *proxy* label. This section hardens it along the three axes that most directly threaten it: whether a calibrator *transfers* across scenes (Sec. 9.1), whether the story is a *reduced-resolution* artifact (Sec. 9.2), and whether it survives a label read from the *actual render* (Sec. 9.3). The full-resolution re-run also corrects the train/test overlap flagged in Sec. 4. Every number traces to a committed JSON (`cross_scene_transfer.json`, `cert_sweep.json`, `p2_summary.json`); Fig. 10 and Fig. 11 regenerate from them with a built-in reproduction check.

## 9.1 Cross-scene calibrator transfer

The core result fits one calibrator per scene and (Sec. 11) declines a transfer claim; we now measure it. For every ordered pair of scenes ($A \to B$) we fit the isotonic calibrator on all of source $A$'s labelled carriers and evaluate it on target $B$'s held-out set — the *identical* evaluation set the in-scene (diagonal) calibrator uses — under both labels, all 24 off-diagonal pairs.

**Selection transfers for free.** An isotonic map is monotone and selection AUC is rank-based, so the order in which $B$'s carriers are scored is the same whether the calibrator was fit on $A$ or on $B$; only the absolute probabilities differ. Empirically the transferred-vs-in-scene AUC delta is within ±0.0004 on

every pair (colour mean −0.0000, depth mean −0.0001): a transferred calibrator prunes exactly as well as a native one.

**Absolute calibration degrades gracefully.** The real transfer question is ECE — whether $A$'s probability scale is calibrated on $B$. Figure 10 shows the ECE transfer matrices. Off-diagonal ECE rises from the in-scene ∼ 0.0013 (colour) / 0.0022 (depth) to a mean of 0.0084 / 0.0256, worst case 0.0173 / 0.0579 (both *truck*→*room*) — still 1–2 orders of magnitude below the uncalibrated raw heuristic (∼ 0.54). *Truck* is the outlier source and target: its heavy floater population gives it the most idiosyncratic feature→reliability map, so every worst pair involves it, while neighbouring indoor Mip-360 scenes (*kitchen*↔*room*) transfer best. The occlusion-aware label transfers roughly 3× worse than colour.

**The certificate stays valid.** We compute each transferred certificate on $B$'s *own* calibration split; because the calibrator is only a monotone re-scoring, split-conformal validity depends on the local labelled conformal set, not on where the calibrator came from. All 24 transferred certificates are valid at $\epsilon = 0.6, \alpha = 0.1$ (colour kept 1.00 everywhere; depth kept 0.91–1.00). The honest deployment recipe follows: *ship one global calibrator* (selection is free and ECE stays 1–2 orders below uncalibrated) *and keep a small per-scene conformal set* to restore the distribution-free certificate. A scene-native calibrator is worth fitting only when absolute-probability ECE, not ranking, is the binding requirement and the scenes are dissimilar.

**Where the certificate becomes selective.** Sweeping $\epsilon$ over 0.30–0.65 on all four scenes and both labels (extending the single-*Truck* sweep of Fig. 6) locates the onset $\epsilon^\star$, the loosest budget at which the certificate first prunes. It tracks each scene's mean held-out reliability monotonically: onsets are *Truck* 0.59/0.62 (colour/depth; mean reliability 0.41/0.38), *Garden* 0.57/0.56 (0.43/0.44), *Kitchen* 0.56/0.55 (0.44/0.45), *Room* 0.47/0.48 (0.53/0.52) — the least reliable scene (*Truck*) becomes selective first (its depth label already at the headline $\epsilon = 0.6$, keeping 90%), the most reliable (*Room*) last. The certificate is neither vacuous above the onset nor over-eager below it.

## 9.2 Full resolution, and a certificate correction

The core carriers were fit at 0.25× (Sec. 4). We re-train all four scenes at native full resolution (*Truck* 979 × 546; Mip-360 up to *Garden* 5187 × 3361, 16× the pixels the core optimised against) and, in the same pass, hold every-eighth view out of training as well, so both labels are genuinely disjoint from the fit.

**The story holds at full resolution** (Table 7). The export feature still predicts held-out reliability at $r = 0.92$–$0.98$ (colour); calibration still crushes ECE to 0.0007–0.0019; calibrated selection lands within 0.8–2.9% of the oracle ceiling and beats opacity everywhere. Crucially, full resolution is at or slightly above the 0.25× quarter control on every scene (calibrated AUC, quarter in parentheses in Table 7) — the core conclusions are *not* a reduced-resolution artifact; if anything, full resolution helps a little.

| full-res (colour) | corr | ECE r→c | AUC (qtr) | orac. | opac. | kept |
|---|---|---|---|---|---|---|
| Truck | 0.93 | 0.66→0.0017 | 0.505 (.492) | 0.520 | 0.302 | **0.77** |
| Garden | 0.92 | 0.50→0.0010 | 0.642 (.602) | 0.659 | 0.482 | 1.00 |
| Kitchen | 0.98 | 0.54→0.0007 | 0.646 (.619) | 0.651 | 0.478 | 1.00 |
| Room | 0.96 | 0.45→0.0019 | 0.734 (.715) | 0.743 | 0.546 | 1.00 |

**Table 7:** Full native resolution, colour label, leak-corrected held-out split. "corr" = export feature vs. held-out reliability; "ECE r→c" = raw heuristic → calibrated; "AUC (qtr)" = calibrated selection AUC with the 0.25× quarter control in parentheses; "orac."/"opac." = oracle ceiling / opacity AUC; "kept" = certified kept fraction at $\epsilon = 0.6$. Full resolution matches or exceeds the quarter control everywhere. *Truck*'s kept 0.77 is the leak-corrected value for the core's optimistic 1.00 (Table 4).

**Certificate correction.** The leak fix lowers the absolute AUC and certificate slightly below the mildly-optimistic core values, and the quarter control localises the shift to leakage, not resolution: it moves the same way at 0.25× and 1.0×, with correlation and ECE essentially unchanged (e.g. *Room* corr 0.96, ECE 0.47 → 0.0013 at quarter matches the core). One number in Table 4 is corrected outright: the core certifies keeping all of *Truck* (colour) at $\epsilon = 0.6$, but that 1.00 was computed on the leaked split; on the clean full-resolution split *Truck* certifies keeping 0.77. *Garden*, *Kitchen*, and *Room* still certify 1.00 at full resolution.

## 9.3 A render-loss reliability label

The core label is a colour-agreement proxy that never renders. We replace it with a label read from the actual alpha-composited render on held-out views, via *exact* blend-weight attribution rather than a first-order approximation. The render is linear in carrier colour, $R_p = \sum_i w_{i,p} c_i$ with blend weight $w_{i,p} = \alpha_i T_i$ ($T_i$ the transmittance in front, so occluded carriers get $w \approx 0$ for free); three colour-backward passes per held-out view therefore recover, exactly, each carrier's blend-weighted RMS colour error, from which reliability is $\exp(-\kappa\,\mathrm{dist}_i)$ as in Eq. 1. (A first-order gate-gradient alternative was tried and rejected: in an over-complete splat scene individual carriers are redundant, so the gradient is dominated by noise and does not track true finite ablation — correlation ≈ 0.11 to group-ablation deltas. The render is genuinely *linear* in colour, so the exact attribution is available and is what we use.)

**The property survives, with honestly weaker margins** (Table 8, Fig. 11). Calibration still crushes ECE to ∼ 0.001. But the export feature predicts the render-grounded label more weakly — $r = 0.66$–$0.81$ versus the proxy's 0.92–0.98 — because the render label penalises occlusion and

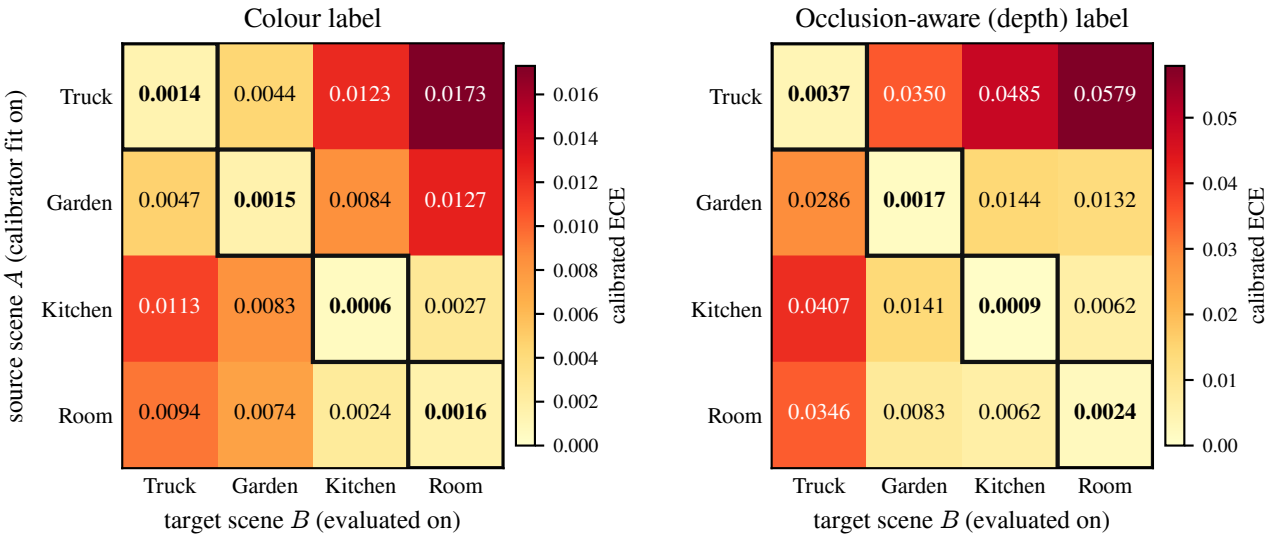


**Figure 10:** Sec. 9.1 (cross-scene transfer): calibrated-ECE transfer matrices, colour label (left) and occlusion-aware label (right). Row $A$ = the scene the calibrator is fit on, column $B$ = the held-out scene it is evaluated on; the boxed diagonal is the in-scene reference (reproducing `calib_<scene>.json`). Off-diagonal (transferred) ECE degrades gracefully — worst case *truck*→*room* — but every cell stays 1–2 orders of magnitude below the uncalibrated raw heuristic (∼ 0.54). Selection AUC (not shown) transfers within ±0.0004 because the isotonic map is rank-preserving.

| render-loss | $r$(ft) | $r$(c,r) | ECE r→c | AUC | orac. | opac. | kept |
|---|---|---|---|---|---|---|---|
| Truck | 0.66 | 0.64 | 0.73→0.0013 | 0.353 | 0.407 | 0.237 | 0.26 |
| Garden | 0.75 | 0.75 | 0.62→0.0009 | 0.444 | 0.481 | 0.343 | 0.79 |
| Kitchen | 0.81 | 0.82 | 0.65→0.0006 | 0.451 | 0.479 | 0.346 | 0.78 |
| Room | 0.78 | 0.78 | 0.54→0.0014 | 0.578 | 0.618 | 0.443 | 1.00 |

**Table 8:** Render-loss label on the full-resolution carriers. "$r$(ft)" = export feature vs. the render-loss label; "$r$(c,r)" = colour proxy vs. render-loss label; remaining columns as Table 7. The feature predicts the render label more weakly and the oracle gap widens (Fig. 11), but calibration still crushes ECE and the certificate stays valid, refusing carriers the proxy over-credited. *Garden*'s render-loss label was rasterized at 0.5× (see Sec. 11); its carriers are native 1.0×.

visibility-weighted colour error the occlusion-blind proxy misses (the two labels themselves correlate only $r$ = 0.64–0.82). Calibrated confidence stays above opacity and near oracle, but the achievable gap widens from ∼ 1–3% to ∼ 6–13%. The certificate stays valid but becomes more selective: at $\epsilon$ = 0.6 the kept fraction drops from ∼ 1.0 toward the floater-heavy scenes' true reliability (*Truck* 0.26, *Garden*/*Kitchen* ∼ 0.78, *Room* 1.00) — the render label honestly refuses to certify carriers the proxy over-credited.

**The load-bearing caveat, reconfirmed.** A directional finite-ablation prune under the render label reproduces the structural fact that opacity, not reliability, is the render-preserving prune: dropping the lowest-opacity carriers leaves held-out render L1 essentially unchanged (e.g. *Room* 0.0408 → 0.0408), while dropping the least-*reliable* carriers costs as much or more (*Room* 0.0408 → 0.0458, *Kitchen* 0.0664 → 0.0722). Opacity *is* the alpha blend weight, so the least-reliable carriers are often the most *visible* — load-bearing but slightly wrong-coloured. Reliability and render-PSNR-preservation optimise different things; the render-loss label makes this explicit rather than resolving it. Calibrated confidence answers "which carriers are trustworthy," not "which are free to drop for PSNR" — the latter is opacity's job.

## 9.4 A machine-checked verification apparatus

The results above are only as trustworthy as the link between each reported number and the artifact behind it, so we make that link mechanical. Every quantitative claim in this paper is content-checked against a committed artifact by one of 17 **publication gates** (`src/aura/publication.py`): a gate does not check that a file exists but that its contents clear a threshold derived from the committed result — the calibrated ECE is below its bound *and* out-selects opacity, each per-scene and transferred certificate is certified at the stated $\epsilon$, $\alpha$, all 16 certified-LOD bounds (Sec. 8) hold, the carrier-registry maturities match their evidence, and so on. Fifteen gates are fully content-checked on a CPU-only clone; the remaining two probe a large GPU-produced trained asset and report an explicit *unverified* status when it is absent — a state distinct

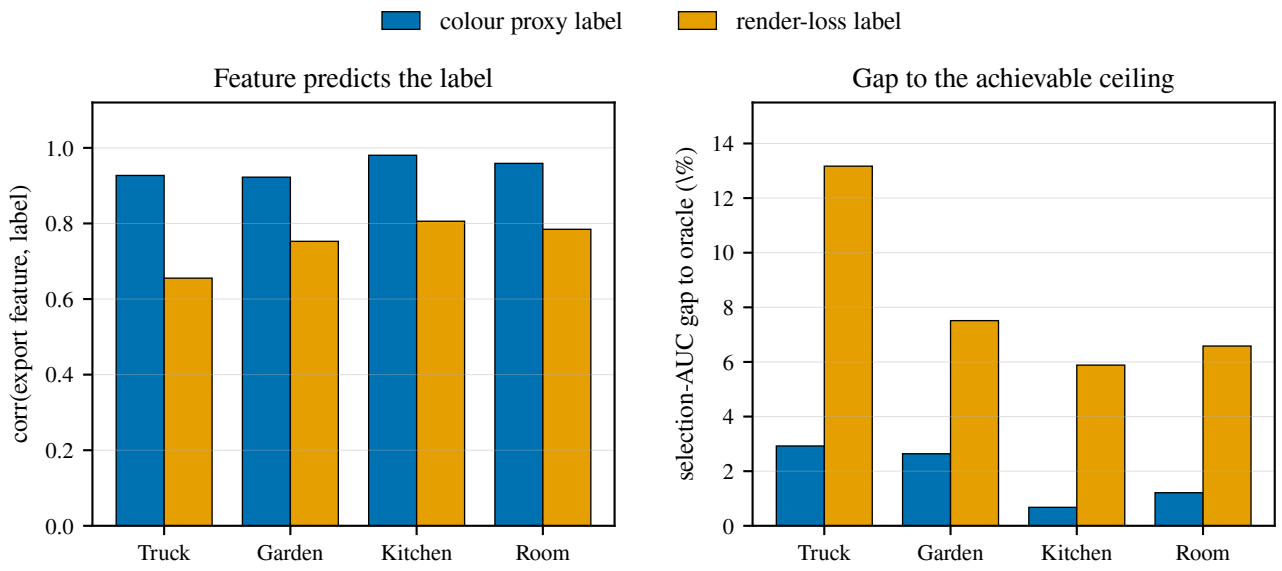


**Figure 11:** Sec. 9.3 (render-loss label): colour proxy (blue) vs. render-loss label (amber) on the same full-resolution carriers. Left: the export feature predicts the render-grounded label more weakly ($r$ = 0.66–0.81 vs. 0.92–0.98). Right: the calibrated selection-AUC gap to the per-label oracle ceiling widens from ~ 1–3% to ~ 6–13%. Calibration still crushes ECE to ~ 0.001 under both labels (Table 8).

from *failed*, so a gate never silently passes.

**The leak class, made impossible.** Sec. 9.2 corrected a train/test overlap that had made one certificate optimistic. Rather than fix that instance and move on, we made the whole *class* mechanically detectable. A split guard (`split_guard.py`) is the single source of truth for the llffhold partition and reconstructs each producer's recorded train/eval view counts: a run in which carriers were trained on all frames while the "held-out" set reused a subset of them no longer forms a disjoint partition and fails the calibration gate at the source. The same gate carries the specific fingerprint of the historical leak — because the clean-split correction lowered the *Truck* colour certificate from the leaked 1.00 to 0.77 while the other scenes stayed at 1.00, a *Truck* full-resolution colour certificate back at 1.00 would signal the leak's return, and the gate rejects it.

**Continuous integration and CPU reproduction.** The CPU-testable suite runs on every push and pull request (GitHub Actions, Python 3.11 and 3.12), so a regression in the calibration, certificate, or streaming math breaks the build. Reproducibility is equally concrete: `REPRODUCE.md` is a verified, GPU-free walkthrough that, from a fresh clone and a fresh virtual environment, recomputes the calibrated confidence, the pruning certificate, and the certified-LOD plan from the committed reliability arrays with pure NumPy at seed 0 — and the recomputed JSONs are byte-identical to the committed ones (an empty `git diff` is the built-in check). The methodology, in one line: the claims in this paper are not merely reproducible in principle but machine-checked against the artifacts that back them, on every commit and from a clean checkout.

# 10 System Context

The calibrated confidence is one layer of AURA, a typed-carrier radiance asset. We describe the surrounding system briefly and honestly; the results in this section are *context*, not contributions of this paper, and we flag each one's provenance.

**A typed-carrier asset with confidence as a first-class channel.** AURA packages a trained scene as typed carriers (Gaussian and Beta in this work) with per-carrier payloads — colour, opacity, and confidence — in a package plus a `carriers.npz` sidecar. Confidence is not a bolt-on: the calibrated value from Sec. 3.3 replaces the carrier's raw confidence in place (`attach_calibrated_confidence`) and is flagged calibrated, so every downstream consumer of the asset reads the trustworthy number by construction.

**Beta-kernel quality backend (a reproduction, not a claim).** For visual quality AURA trains Beta carriers on a DBS fork [3]. On *Truck*, at a matched ~1M-carrier budget, the Beta path reaches 26.35 dB PSNR against a fixed-Gaussian control's 26.02 dB (+0.33 dB; SSIM 0.896 vs 0.890, LPIPS 0.122 vs 0.128), and reaches the control's quality at about half the carriers (26.07 dB at 0.5M). Figure 12 shows the matched-

budget comparison on a held-out view. Across eight scenes the Beta path averages +0.80 dB over the control. On *Truck* we additionally ran a *true* gsplat-3DGS control — an MCMC run [23] at the same matched ∼1M-carrier budget, 30k steps, the same every-eighth-view split — to test whether the frozen-$\beta$ control merely flattered the comparison. It did not: the Beta path (26.39 dB) beats the true gsplat-3DGS run (25.94 dB, final at 30k) by +0.45 dB, and the frozen-$\beta$ control (25.96 dB) lands within 0.03 dB of the true 3DGS run. The frozen control was therefore *not* artificially weak, and the typed win is not an artifact of a hobbled baseline. (These 25.94/25.96/26.39 figures come from the multiscene-evaluation run, whose matched-budget *Truck* numbers differ marginally from the 26.02/26.35 ablation-configuration figures above; both are at a matched ∼1M budget on *Truck*.) This true gsplat-3DGS control exists for *Truck* only (1 of 8 scenes); the +0.80 dB eight-scene mean remains measured against the frozen-$\beta$ control. We report these numbers only to situate the asset, and we are explicit about their limits: this *reproduces* DBS's published claim and is not a contribution of ours; the eight-scene control is a frozen-$\beta$ DBS ablation rather than a real gsplat-3DGS run (a true 3DGS control was added for *Truck* only, above); and the Mip-360 evaluation is at image downsample. The honest negatives that came with the reproduction — adaptive per-carrier $\beta$ does not beat a good global $\beta$ (26.35 vs 26.42 at uniform $\beta$=2), and the +dB win decomposes largely to the spherical-Beta colour model rather than to adaptivity — are recorded in the project's documentation and are not claimed here as findings.

**Interchange: carrying the calibrated confidence.** The point of calibrating a per-carrier confidence is to deliver it. AURA exports the calibrated value as a first-class channel on three interchange standards. Through `KHR_gaussian_splatting` [4] — still a Release Candidate as of July 2026 — it ships as the `_AURA_CONFIDENCE` core-spec custom attribute of the glTF splat primitive (an end-to-end GLB export of ∼1M Beta carriers is ∼52 MB). Through OpenUSD 26.03 [5] it ships as the `primvars:aura:confidence` primvar (vertex interpolation) on the native `UsdVolParticleField3DGaussianSplat` schema (via `usd-core`), which Omniverse RTX path-traces natively [20]. And because Niantic's compact `.spz` container [6] (version 4) has no per-splat confidence attribute and no per-point extension indexing, AURA carries the calibrated value in an aligned `.spz.confidence` sidecar joined on the point order; the SPZ writer itself is cross-validated bit-exact against the reference C++ implementation (commit `bb0efad`) — files AURA writes decode through the reference `loadSpz` and files the reference `saveSpz` writes decode here, and on identical bytes the two decoders agree on positions, scales, and spherical harmonics. All three are *delivery-path* validations — structural round-trips and reference cross-validation, not third-party-viewer render benchmarks — and we claim them only as such.

**On-asset ray query (a systems note).** An AURA asset answers ray queries against its carriers — the substrate for secondary-ray readiness probes and pick/selection. The query accepts a carrier when a ray passes within $k \cdot \max(\text{scale})$ of its centre, an isotropic hit test whose acceptance sphere the axis-aligned leaf box provably supersets; a median-split (object-median) BVH over those boxes therefore only *narrows* the candidate set and applies the exact brute-force hit arithmetic to what survives, so the accelerated query returns bit-for-parity identical results — 0 mismatches across the synthetic edge cases and 300 rays on the real 129,531-carrier *Truck* asset. A batched API amortises traversal across a frame's rays. This is a CPU-only algorithmic result (an $O(N)$ per-ray scan replaced by sub-linear node visits, with parity as the acceptance criterion); a wall-clock comparison against the CUDA ray tracers now landing in gsplat's main branch (3DGRT/3DGUT) [2] is explicitly deferred, and AURA's secondary rays remain readiness probes, not a physically-based tracer.

## 10.1 Carrier registry and semantic codebook

AURA advertises seven carrier types, and they are not equally real; over-claiming which ones are is the failure mode a typed-asset format invites. The registry now carries an explicit *maturity* on every type — *trained* (trains and renders on real scenes with committed evidence: only Gaussian and Beta today), *demo* (a footprint validated in 2D or PRISM-extension use only: Gabor, neural), or *metadata* (a typed contract with no trained render family: surface, volume, semantic) — and a publication gate fails if any type advertises a maturity its committed artifacts do not back, or if a *trained* claim lacks its `calib_<scene>.json` evidence. In the same spirit, an unimplemented carrier footprint is no longer routed silently to the Gaussian branch: the fallback is now explicit and provenance-annotated (it emits a warning and records `fallback:gaussian`), so a substitution is visible rather than hidden.

For the *semantic* type we include the storage machinery a shipped open-vocabulary asset needs, following the LangSplatV2 codebook idea [24]: heavy per-carrier features live once in a shared $K$-entry codebook while each carrier stores a small integer index, which also affords an $O(K \cdot d + N)$ open-vocabulary query (score the $K$ entries, fan out to carriers by index) in place of a dense $O(N \cdot d)$ scan. On a real DINOv2 [25] distillation of *Truck* (998,950 carriers × 384-d features), a $k$=64 codebook compresses the 1.53 GB feature tensor to 1.05 MB (uint8 indices, a ∼1400× reduction) at reconstruction relative error 0.319. We are explicit


**Figure 12:** System context (a reproduction of Deformable Beta Splatting [3], not a contribution): the Beta-kernel backend on a held-out *Truck* view at a matched ∼1M-carrier budget — ground truth, fixed Gaussian (26.02 dB), and adaptive Beta (26.35 dB), with a shared detail crop; the control shown is a frozen-$\beta$ DBS ablation. A separate matched-budget run adds a *true* gsplat-3DGS MCMC control for *Truck* (25.94 dB, final@30k), which adaptive Beta (26.39 dB) still beats by +0.45 dB while the frozen-$\beta$ control (25.96 dB) lands within 0.03 dB of it (Sec. 10, *Truck* only, 1 of 8 scenes). It is shown to place the confidence layer in the asset it annotates, not as a quality claim of this paper.

that this is the *storage* layer only: distilling features onto the trained carriers and shipping them inside the `.aura` asset is GPU-gated future work, so *semantic* remains a metadata carrier and the open-vocabulary capability is demonstrated by the distillation experiment, not yet backed by committed per-carrier features.

# 11 Limitations

We state the boundaries of this work plainly; several are load-bearing for how the results should be read.

**Preview-stage system, local gates.** AURA is a research preview (`v1.0.0`). Its publication gates are content-checked — each parses the committed artifact behind a claim and enforces numeric thresholds (Sec. 9) — but they are *ours* and run locally: no external party has yet reproduced the results (`REPRODUCE.md` exists precisely to make that cheap). Every number in Secs. 5–7 regenerates bit-exactly from committed arrays through a tested CPU core, but the evaluation harness is our own.

**The reliability label is a self-computed proxy.** The core reliability of Eq. 1 is multi-view colour agreement — a floater/consistency proxy, conservative (it under-credits occluded or rarely seen carriers) and cheap, but a proxy. We narrow this along two axes: an occlusion-aware label (a coarse 8-pixel-block $z$-buffer, which approximates rather than resolves sub-pixel transparency) and, in Sec. 9.3, a label read from the *actual* alpha-composited render via exact blend-weight attribution — and the property survives both, with honestly weaker margins under the render label. What remains is that even the render label scores a carrier against the render of *its own* fit; a label grounded in human judgement or an *independent* reconstruction is future work.

**Single-capture scenes; no independent-reconstruction test.** All four scenes are single captures, evaluated on held-out views of the same capture. We test transfer across *scenes* (Sec. 9.1) but not across *independent* reconstructions of the same scene or capture conditions; those independent re-captures remain open future work.

**Cross-scene transfer: measured, not assumed.** Sec. 9.1 now measures calibrator transfer rather than declining to: selection transfers for free and ECE degrades gracefully across all 24 scene pairs, but a foreign calibrator's absolute ECE can be $\sim 10\times$ the in-scene value, so we recommend a small per-scene conformal set rather than claiming a single calibrator is scene-agnostic.

**The certificate is loose at the reported budget.** At $\epsilon = 0.6$ the certificate mostly certifies keeping the whole asset (Table 4); its selective regime, now mapped on all four scenes and both labels (Sec. 9.1), begins at $\epsilon^\star \approx 0.47$–$0.62$ and tracks scene reliability. The Hoeffding bound is conservative; we report the operating curve rather than a single flattering point.

**The bound treats carriers as exchangeable draws; spatial correlation is not modelled.** The Hoeffding UCB (Eq. 3) bounds the retained-set mean unreliability under the assumption that the calibration carriers are exchangeable $[0, 1]$ observations. Neighbouring carriers on the same surface patch are spatially correlated (shared occlusion, texture,

colour error), so the *effective* sample size can be below the nominal $m$; under strong spatial dependence the reported margins ($\sim 0.005$–$0.006$ at $m \approx 5$–$6\times10^4$) are optimistic and the true coverage could dip below $1-\alpha$. Our held-out checks (the LOD ladder validates every published $\epsilon_k$ on a disjoint evaluation half, Sec. 8) find no violation, but we do not test coverage under a spatial block-bootstrap; a cluster-aware resampling of carriers is the honest way to certify the margin and is future work.

**Point estimates at one split and one seed.** Every reported correlation, ECE, AUC, and certificate is a single point estimate at seed 0 and a single 50/50 calibration/evaluation split; we do not report resampled confidence intervals over splits or seeds. The margins between the calibrated signal and the baselines are large and consistent across four scenes and two labels, but the individual numbers should be read as point estimates, not as bracketed with uncertainty. Combined with the spatial-correlation caveat above, a full seed/split resampling study would tighten the honest error bars on the ECE and selection numbers.

**Four scenes; resolution.** Four scenes is a small sample. The core evaluation is at 0.25× (a no-densify, tractable-asset constraint), but Sec. 9.2 re-runs all four at native full resolution and the conclusions hold or strengthen, so the downsample is not load-bearing. One concession: *Garden*'s 17.4 MP render-loss label was rasterized at 0.5× (its full-resolution raster OOMs under concurrent GPU load; its colour and occlusion labels ran at native 1.0×, as did its carriers).

**No rasterizer-performance claim.** AURA includes a research rasterizer (PRISM) as an extension substrate; it is not a fast renderer and we make no rendering-speed claim for it. The quality backend is the DBS fork, and the confidence layer is renderer-agnostic post-processing.

**Relighting is a preview, pre-registered as such.** AURA includes a relighting path, but it is a preview, not inverse rendering: its albedo is the baked SH-DC colour (which still contains shading), its normals are the unsigned covariance short axis, and no per-carrier material is optimised from data. We report no relighting numbers here, and we have *pre-registered* — before the GPU attempt — the rule that will decide the capability's framing: a single run on all four TensoIR-Synthetic scenes plus Stanford-ORB, promoted to "real inverse-rendered relighting" only if it clears, jointly, mean relight PSNR $\geq$ 27 dB, albedo PSNR $\geq$ 27 dB, and signed-normal MAE $\leq 8°$ (with a Stanford-ORB margin fixed from the live leaderboard at eval time), and otherwise formally descoped to a "confidence-weighted relighting preview." The bar is anchored to the object-centric Gaussian-relighting methods that actually report on these benchmarks (TensoIR, GS-IR, IRGS, SVG-IR, Relightable-3DGS, and path-traced inverse rendering; relight PSNR $\approx$ 28–32 dB, albedo PSNR $\approx$ 29–32 dB, normal MAE $\approx 4°$); three methods named as baselines in our own earlier planning (SSD-GS, R3GW, GS$^3$) do not in fact report TensoIR-Synthetic or Stanford-ORB numbers, so they are qualitative cross-references, not the numeric bar.

**The Beta backend result is a reproduction.** As stated in Sec. 10, the +0.33/+0.80 dB numbers reproduce DBS against a frozen-$\beta$ ablation control and are not a contribution of this paper. A true gsplat-3DGS MCMC control was run for *Truck* only (1 of 8 scenes), where the Beta win holds (+0.45 dB) and the frozen control proves within 0.03 dB of true 3DGS; the eight-scene mean remains a frozen-control number, and a further planned quality-backend variant was left unbuilt.

# 12 Conclusion

A Gaussian-splat asset ships a per-primitive confidence that means nothing and guarantees nothing, and yet pipelines prune, stream, and cull by it. We showed, on four real scenes, that this is fixable with cheap post-processing: an export-time train-agreement feature predicts a carrier's held-out reliability ($r = 0.91$–$0.98$) where the shipped heuristics do not; isotonic calibration turns it into a confidence a consumer can read as a probability (ECE cut ~300–900×); a split-conformal certificate makes pruning below a threshold safe with a distribution-free bound; and the calibrated confidence is a near-oracle pruning signal (within 1–4% of the ceiling, dominating opacity everywhere), delivered through standard glTF and OpenUSD channels. The properties hold under an occlusion-aware label; they further persist at native full resolution — correcting a train/test overlap that had made one certificate mildly optimistic (*Truck* 1.00 $\rightarrow$ 0.77) — transfer across scenes for selection with a per-scene conformal set restoring the guarantee, and survive a stricter render-loss label read from the actual alpha composite, with honestly weaker margins. We have been deliberate that this is a preview-stage system measured against self-computed proxy labels on single captures, and that the typed-carrier quality backend it annotates is a reproduction of prior work. We also turn the single certificate into a certified level-of-detail ladder — a consumer streams by calibrated confidence and stops at any level with a family-wise bound on the reliability mass it has discarded (all 16 published bounds hold) — and deliver the confidence through glTF, OpenUSD, and an SPZ v4 sidecar. What is new is narrow and, we think, worth stating cleanly: a calibrated, certified, exportable per-carrier confidence is a capability a bare splat does not have, and it can be built and honestly measured today. The library, the reliability arrays, the calibration core, and the figure-regeneration scripts back every number reported here; and the link from claim to artifact is mechanical — every quantitative result is content-checked by publication gates on each commit and reproduces CPU-only, bit-for-bit, from a fresh clone.

| Scene | signal | 10% | 20% | 40% | 60% | 80% | 100% |
|---|---|---|---|---|---|---|---|
| Truck | calib. | 0.770 | 0.722 | 0.635 | 0.556 | 0.482 | 0.407 |
| | opac. | 0.306 | 0.328 | 0.357 | 0.377 | 0.393 | 0.407 |
| Garden | calib. | 0.797 | 0.750 | 0.661 | 0.579 | 0.503 | 0.426 |
| | opac. | 0.450 | 0.456 | 0.472 | 0.458 | 0.434 | 0.426 |
| Kitchen | calib. | 0.836 | 0.779 | 0.686 | 0.603 | 0.523 | 0.443 |
| | opac. | 0.470 | 0.466 | 0.463 | 0.457 | 0.453 | 0.443 |
| Room | calib. | 0.896 | 0.855 | 0.781 | 0.708 | 0.626 | 0.529 |
| | opac. | 0.489 | 0.510 | 0.544 | 0.551 | 0.548 | 0.529 |

**Table 9:** Retained mean reliability at selected keep-fractions (colour label), calibrated confidence vs. opacity. At 100% keep both equal the scene base rate; as the budget tightens, calibrated confidence rises steeply (keeps the reliable carriers) while opacity stays near the base rate.

# A Per-Scene Selection Curves

Table 9 lists the calibrated-confidence and opacity retained-reliability curves at every budget for all four scenes (colour label), the numbers behind Fig. 7 and the AUCs of Table 5. The monotone decline of the calibrated curve toward the base rate as the budget grows, tracking the oracle, is the selection signal; opacity's near-flat curve is its absence.

# B Reproduction

Every number regenerates from the committed reliability arrays (`outputs/reliability_{scene}{,_depth}.npz`) and the calibration core (`src/aura/calibration.py`). `REPRODUCE.md` is a verified, GPU-free walkthrough that recomputes the calibrated confidence, the pruning certificate, and the certified-LOD plan (Sec. 8) from a fresh clone and virtual environment, at seed 0, and checks that the recomputed JSONs are byte-identical to the committed ones. Per scene and label:

- **Reliability label + feature** (GPU, accuracy job): `per_carrier_reliability.py --aura <pkg> --manifest <m> --label {color,depth_aware} --out reliability_<scene>.npz`.
- **Calibration + certificate + selection** (CPU): `calibrate_confidence.py --reliability reliability_<scene>.npz --scene <scene> --report calib_<scene>.json`, at defaults $\alpha$=0.1, $\epsilon$=0.6, seed 0.
- **Hardening** (Sec. 9): `cross_scene_transfer.py`, `cert_sweep.py` (both pure-CPU over the committed `.npz`, seed 0) → `cross_scene_transfer.json`, `cert_sweep.json`; `run_p2.sh` + `collect_p2.py` (full-res train + three labels) → `p2_summary.json`.
- **Certified LOD** (Sec. 8, CPU): `lod_certified_eval.py` → `lod_certified.json` (all 16 bounds hold on the disjoint eval half); `aura lod-plan` emits a scene's Bonferroni-valid streaming plan.
- **Figures**: `figures/make_figures.py` (core) and `figures/make_p1p2_figures.py` (Sec. 9) read the committed arrays/JSONs and reproduce the committed reports before plotting (a built-in reproduction check). The method overview (Fig. 1), the qualitative confidence/pruning/LOD figures (Figs. 5, 8, 9) and the scene grid (Fig. 2) are (re)generated by `figures/make_qual_figures.py`, which re-derives the *Truck* LOD plan and asserts it matches `lod_certified.json` before rendering.

# References


[1] B. Kerbl, G. Kopanas, T. Leimkühler, and G. Drettakis, "3D Gaussian splatting for real-time radiance field rendering," *ACM Transactions on Graphics*, vol. 42, no. 4, pp. 139:1–139:14, 2023.

[2] V. Ye, R. Li, J. Kerr, M. Turkulainen, B. Yi, Z. Pan, O. Seiskari, J. Ye, J. Hu, M. Tancik, and A. Kanazawa, "gsplat: An open-source library for Gaussian splatting," *Journal of Machine Learning Research*, vol. 26, no. 34, pp. 1–17, 2025.

[3] R. Liu, D. Sun, M. Chen, Y. Wang, and A. Feng, "Deformable beta splatting," in *ACM SIGGRAPH 2025 Conference Papers*, 2025. arXiv:2501.18630.

[4] Khronos Group, "KHR_gaussian_splatting: glTF 2.0 extension for 3D Gaussian splatting." https://github.com/KhronosGroup/glTF/tree/main/extensions/2.0/Khronos/KHR_gaussian_splatting, 2026. Release candidate, accessed July 2026.

[5] Alliance for OpenUSD, "OpenUSD v26.03: UsdVol ParticleField 3D Gaussian splat schema." https://aousd.org/blog/openusd-v26-03/, 2026. Accessed July 2026.

[6] Niantic Labs, "SPZ: A compressed Gaussian splat file format." https://github.com/nianticlabs/spz, 2024. Version 4 (NGSP) container; reference C++ implementation at commit `bb0efad`, accessed July 2026.

[7] A. Hanson, A. Tu, V. Singla, M. Jayawardhana, M. Zwicker, and T. Goldstein, "PUP 3D-GS: Principled uncertainty pruning for 3D Gaussian splatting," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2025. arXiv:2406.10219.

[8] A. N. Razlighi, E. B. Golezani, and S. Kasaei, "Confident splatting: Confidence-based compression of 3D Gaussian splatting via learnable beta distributions," *arXiv preprint arXiv:2506.22973*, 2025.

[9] S. Safadoust, F. Tosi, F. Güney, and M. Poggi, "WarpRF: Multi-view consistency for training-free uncertainty

quantification and applications in radiance fields," *arXiv preprint arXiv:2506.22433*, 2025.

[10] L. Goli, C. Reading, S. Sellán, A. Jacobson, and A. Tagliasacchi, "Bayes' Rays: Uncertainty quantification for neural radiance fields," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2024. arXiv:2309.03185.

[11] W. Jiang, B. Lei, and K. Daniilidis, "FisherRF: Active view selection and uncertainty quantification for radiance fields using fisher information," in *European Conference on Computer Vision (ECCV)*, 2024. arXiv:2311.17874.

[12] R. E. Barlow, D. J. Bartholomew, J. M. Bremner, and H. D. Brunk, *Statistical Inference Under Order Restrictions: The Theory and Application of Isotonic Regression*. John Wiley & Sons, 1972.

[13] B. Zadrozny and C. Elkan, "Transforming classifier scores into accurate multiclass probability estimates," in *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD)*, 2002.

[14] C. Guo, G. Pleiss, Y. Sun, and K. Q. Weinberger, "On calibration of modern neural networks," in *International Conference on Machine Learning (ICML)*, 2017.

[15] V. Vovk, A. Gammerman, and G. Shafer, *Algorithmic Learning in a Random World*. Springer, 2005.

[16] A. N. Angelopoulos and S. Bates, "Conformal prediction: A gentle introduction," *Foundations and Trends in Machine Learning*, vol. 16, no. 4, pp. 494–591, 2023.

[17] A. N. Angelopoulos, S. Bates, A. Fisch, L. Lei, and T. Schuster, "Conformal risk control," in *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2208.02814.

[18] S. Bates, A. Angelopoulos, L. Lei, J. Malik, and M. I. Jordan, "Distribution-free, risk-controlling prediction sets," *Journal of the ACM*, vol. 68, no. 6, pp. 43:1–43:34, 2021.

[19] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 13–30, 1963.

[20] NVIDIA, "Gaussian splats (particle fields) — omniverse materials and rendering." https://docs.omniverse.nvidia.com/materials-and-rendering/latest/particle-fields.html, 2026. Accessed July 2026.

[21] A. Knapitsch, J. Park, Q.-Y. Zhou, and V. Koltun, "Tanks and temples: Benchmarking large-scale scene reconstruction," *ACM Transactions on Graphics*, vol. 36, no. 4, pp. 78:1–78:13, 2017.

[22] J. T. Barron, B. Mildenhall, D. Verbin, P. P. Srinivasan, and P. Hedman, "Mip-NeRF 360: Unbounded anti-aliased neural radiance fields," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022.

[23] S. Kheradmand, D. Rebain, G. Sharma, W. Sun, Y.-C. Tseng, H. Isack, A. Kar, A. Tagliasacchi, and K. M. Yi, "3D Gaussian splatting as Markov chain Monte Carlo," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2024. arXiv:2404.09591.

[24] W. Li, Y. Zhao, M. Qin, Y. Liu, Y. Cai, C. Gan, and H. Pfister, "LangSplatV2: High-dimensional 3D language Gaussian splatting with 450+ FPS," *arXiv preprint arXiv:2507.07136*, 2025.

[25] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jégou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "DINOv2: Learning robust visual features without supervision," *Transactions on Machine Learning Research (TMLR)*, 2024. arXiv:2304.07193.