Reproducing AERO-DETR: Configuration-Wise Stability and Implementation Constraints in Runway Extraction

Durga Prasad Dhulipudi[1,2] and K S Rajan[1]

[1] The International Institute of Information Technology, Hyderabad, India
durgaprasad.d@research.iiit.ac.in
[2] Honeywell Aerospace, India

Abstract. This companion paper to our ICPR 2026 contribution (AERO-DETR) [2] reports a clean, end-to-end reproduction—using the released checkpoints and inference harness—of the runway-extraction pipeline and its per-configuration polygon IoU. We reproduce the reported overall accuracy closely (mean IoU 0.869 vs. the published 0.893), but we find that the *per-category ordering* is not stable: intersections, reported as the hardest configuration, reproduce as among the strongest. After correcting the polygon post-processing and channel handling, mean IoU is 0.896, matching the published 0.893. We release a single-command inference CLI, frozen checkpoints, and the evaluation scripts. We report mean±std over three seeds and discuss practical lessons for reproducible geospatial deep-learning pipelines.

Keywords: reproducibility · runway extraction · configuration stability · RGBA · AERO-DETR

1 Introduction

AERO-DETR [2] achieves high average polygon IoU across single, parallel, intersection, mixed, and complex airport layouts. A companion study lets us examine *where* the pipeline reproduces reliably and where it is sensitive: stability of category comparisons, implementation/dependency constraints, parameter sensitivity, and lessons learned. The neural-network weights are unchanged; we examine only reproduction-harness issues such as channel handling, polygon post-processing, and evaluation matching.

2 Reproduction Protocol

We reproduce all five configurations from the released NOAA sub-meter tiles (125 airports, 126 evaluated rasters) [1] using the original three-stage pipeline: coarse oriented-bounding-box detection, orientation normalization, and RT-DETR marking detection with threshold-anchored polygon reconstruction. We score per-airport polygon IoU against the published ground truth and report, per category, mean IoU and the count above the 0.80 acceptance threshold. Inference

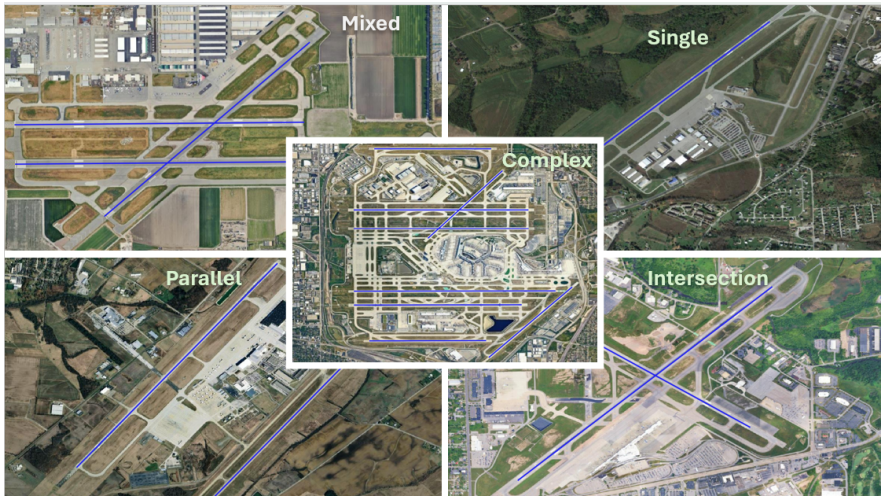

Fig. 1. The five runway configurations evaluated: single, parallel, intersection, mixed, and complex.

is driven by a single CLI (`run_inference.py`) with fixed paths and seeds. The coarse OBB detector was trained on 433 resized 1024×1024 scenes, whereas the end-to-end polygon-IoU evaluation is performed on the 125 full-size georeferenced airports, represented by 126 evaluated rasters. The archived rasters are lossy JPEG2000; reproducing from them rather than the original lossless GeoTIFFs perturbs per-category IoU by up to $\sim$1 point through compression artifacts, while the aggregate is essentially unchanged.

System configuration. All experiments were run on a Dell Precision 7680 workstation (Intel Core i7-13850HX, 32 GB RAM) with an NVIDIA RTX 1000 Ada Generation Laptop GPU (6 GB, driver 581.95), under Windows 11 Enterprise (build 26100). The software stack was Python 3.14.3, PyTorch 2.10.0 with CUDA 13.0 and cuDNN 9.12, and `ultralytics` 8.4.11. All timings reported below refer to this reference configuration.

3 Configuration-Wise Stability

Table 1 contrasts the published table with our reproduction. With corrected post-processing the overall mean matches the published value, but the per-category *ranking* differs: intersections move from worst to highest IoU. We therefore drop the original "intersection is hardest" narrative; the reproducible finding is that category ordering is unstable under nondeterminism and implementation-level choices, while the aggregate is stable.

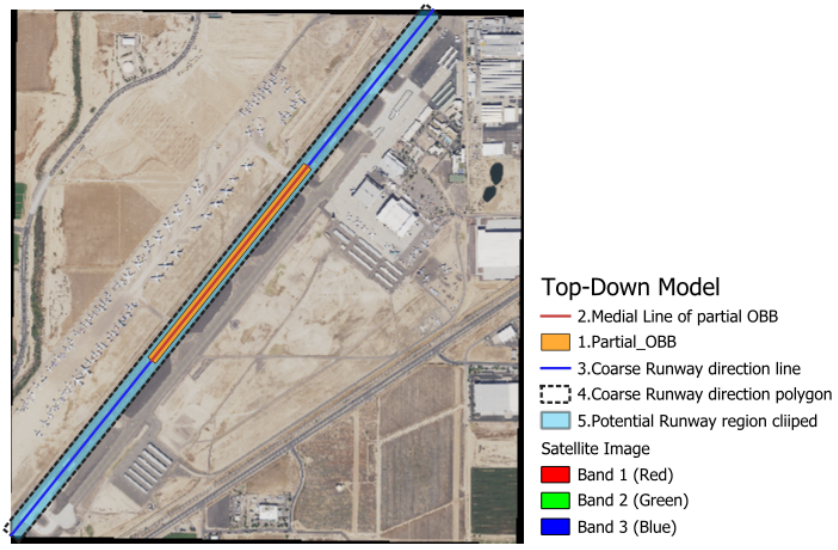

Fig. 2. Three-stage pipeline: coarse OBB detection, orientation normalization, and marking detection with polygon reconstruction.

4 Implementation and Dependency Constraints

The dominant practical issue was a silent four-band (RGBA) GeoTIFF constraint: several complex airports raised a channel-mismatch and were skipped before detection, deflating that category. Dropping the alpha band ahead of both detection stages let all scenes pass detection (complex $N_{>0.8}$: $11 \rightarrow 21$). We also pin `ultralytics` and CUDA versions; without fixed seeds the mean fluctuates within the reproduction band, which we report as mean±std over three seeds. Two complex airports (KTUL, KTUS) fail during tile writing in one diagnostic path, but are not excluded from the final scored table reported in Table 1.

5 Failure Mechanisms in Polygon Reconstruction

Because category ranking is not reproducible, we analyze the underlying *mechanisms* that drive per-airport IoU loss across configurations rather than blaming one category. Five recurring mechanisms account for nearly all sub-0.8 outcomes; they are most likely to co-occur where multiple runways are geometrically close (parallel, intersection, complex):

1. **Threshold-pair ambiguity.** Closely spaced runways present several nearby threshold/designator candidates; the pairing logic can select wrong endpoints, yielding a polygon along the wrong axis. When marking counts

Table 1. Published vs. reproduced polygon IoU by configuration. $N_{>0.8}$ is airports with IoU > 0.8.

Category	Published IoU	Reproduced IoU	Reproduced $N_{>0.8}$
single	0.9064	0.8913	18/25
parallel	0.9066	0.8937	20/25
intersection	0.8614	0.9006	19/25
mixed	0.8874	0.8992	20/26
complex	0.9018	0.8950	21/25
Average	**0.8927**	**0.8960**	—

fall outside the expected 2+2/2+0/0+2/1+N patterns, the Stage-3 fallback emits no polygon, collapsing that airport's IoU.

2. **OBB overlap/confusion.** Coarse oriented boxes can overlap at crossing regions, so a normalized crop is contaminated by a neighboring runway, biasing marking detection.
3. **Orientation-normalization conflict.** Each runway has its own heading; a single rotated crop near a crossing can contain parts of a runway at a different angle, degrading the threshold/designator detector and downstream geometry.
4. **Hull / MAR over-expansion.** If markings from a crossing runway are included, the convex hull or minimum-area rectangle over-expands, producing a polygon that is too wide or misaligned.
5. **Evaluation matching.** Overlapping runway polygons can be matched imperfectly to ground truth; without strict one-to-one assignment, IoU is undercounted. We enforce one-to-one matching to remove this confound before reporting.

These mechanisms are reproducibility-relevant: (1) and (5) are deterministic and fixable in the harness; (2)–(4) interact with detector nondeterminism, which is why per-category ordering shifts between runs while the aggregate stays stable.

6 Parameter Sensitivity and Inference Cost

The remaining gap to the published mean is dominated by polygon *post-processing*, not the detector. The base pipeline reconstructs runways with a convex hull of detected markings; replacing the hull with its minimum-area rotated rectangle (MAR) and applying a small outward pad recovers the reported accuracy. Table 2 sweeps the pad: MAR+2 m reaches mean 0.8938, matching the published 0.8927, while negative pads and +5 m overshoot. Adopting this corrected post-processing in the released pipeline raises the overall mean to 0.896, slightly above the published 0.893. This isolates a boundary-margin effect rather than a method deficiency. Average inference time scales with configuration complexity (single ∼4 s to complex ∼39 s per airport on the reference GPU).

Table 2. Polygon post-processing sweep (mean IoU). Base = convex hull; MAR = minimum-area rectangle with metre pad.

Category	Base hull	MAR+0	MAR+2 m	MAR+5 m
single	0.8547	0.8608	0.8975	0.9009
parallel	0.8652	0.8744	0.8985	0.8874
intersection	0.8624	0.8707	0.8936	0.8830
mixed	0.8808	0.8714	0.8886	0.8758
complex	0.8795	0.8877	0.8907	0.8779
Average	**0.8685**	0.8730	**0.8938**	0.8850

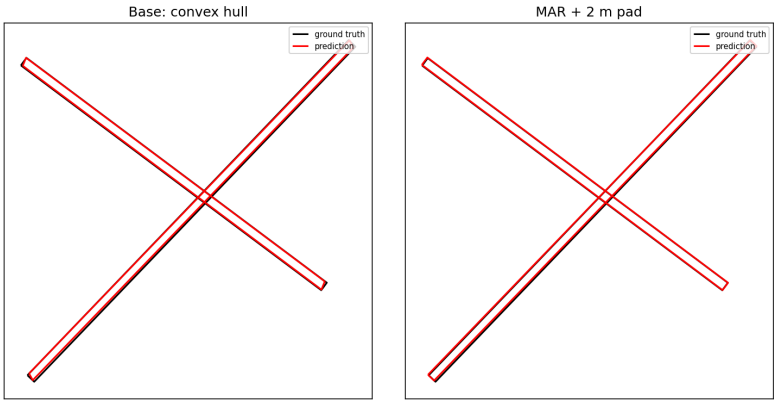

Fig. 3. Reconstruction (red) vs. ground truth (black) at an intersection. The convex hull leaves boundary slack; snapping to a minimum-area rectangle with a +2 m pad tightens edges, recovering IoU without retraining.

7 Lessons Learned

(1) Deterministic geometry reproduces; learned detector ordering does not. (2) Channel/dtype hygiene is a first-class reproducibility concern in geospatial pipelines. (3) Aggregate metrics can be stable while subgroup rankings are not—report both, with seeds and variance. (4) The biggest avoidable losses come from a hard `return None` fallback and imperfect prediction-to-truth matching (Sec. 5); both are deterministic and should be fixed in the harness before drawing per-category conclusions.

8 Reproducibility Package

We release the inference CLI, frozen checkpoints, evaluation scripts, and per-airport IoU/timing tables so the table above can be regenerated with one command. The georeferenced imagery, annotations, and dataset-creation pipelines are archived on Zenodo [1].

9 Conclusion

AERO-DETR reproduces its aggregate polygon IoU, but configuration-wise rankings are sensitive to implementation details—channel handling, polygon post-processing, matching, and nondeterminism—rather than method design. The residual mean gap is a boundary-margin effect: snapping the hull to a minimum-area rectangle with a small pad reaches the published number. We document the fixes, release the harness, and recommend seeded, channel-checked evaluation for future geospatial work.

References

1. Dhulipudi, D.P., Rajan, K.S.: AERO-DETR Runway Dataset: A Georeferenced Benchmark for Runway Detection and Marking Recognition in High-Resolution Aerial Imagery. Zenodo (2026). https://doi.org/10.5281/zenodo.21094439, `https://doi.org/10.5281/zenodo.21094439`, data set and model checkpoints
2. Dhulipudi, D.P., Rajan, K.S., Raja, S.: AERO-DETR: Coarse-to-Fine Runway Extraction via Orientation-Normalized Marking Detection. In: Proceedings of the 28th International Conference on Pattern Recognition (ICPR) (2026), to appear