VISORCH: BENCHMARKING AGENT-DESIGNED VISION PIPELINES

Anonymous authors
Paper under double-blind review

ABSTRACT

Composing off-the-shelf vision models into task-specific pipelines avoids training from scratch, but manually designing them is costly. While typical vision agents invoke an LLM or VLM at every inference step, we study a deployment-efficient alternative: an agent designs an executable pipeline once during adaptation, which is then frozen and evaluated on unseen inputs with no such model calls at test time. We introduce VisOrch, a benchmark evaluating whether agents can automatically synthesize reusable vision pipelines from few examples. To ensure scores reflect genuine orchestration rather than a single dominant tool, VisOrch admits a task only when its strongest off-the-shelf component leaves substantial headroom. This yields nine tasks across four task families, totalling 540 adaptation, 480 validation, and 2,415 blind test examples; each task provides a fixed budget of 60 labelled images used to tune the tool baselines. On eight of the nine tasks, an agent-designed pipeline outperforms the strongest single tool on both metrics, whereas directly prompted frontier VLMs fall below it on every task they attempt. However, no single designer is universally reliable: iterative program search is ahead on both metrics at $p < 0.05$ on four tasks, a general coding agent leads on three, both improve on DECIMER, and neither improves on both metrics for PlantSeg. Finally, building VisOrch exposed key evaluation pitfalls, including shortcut exploitation and degenerate metrics, which the benchmark protocol now explicitly guards against.

1 INTRODUCTION

The rapid development of foundation models and specialised vision models has made many visual capabilities available as reusable off-the-shelf tools, including detection, segmentation, grounding, retrieval, and recognition (Radford et al., 2021; Oquab et al., 2024; Liu et al., 2024; Ravi et al., 2025). This creates an alternative to training a dedicated model for every new task: a task-specific solution may instead be built by composing existing vision tools. However, designing such pipelines remains highly task dependent, requiring decisions about tool selection, composition, parameterisation, and post-processing, and often substantial human engineering.

Meanwhile, LLM- and VLM-based agents have shown growing capability in program synthesis, tool use, software engineering, and machine-learning workflows (Mialon et al., 2024; Jimenez et al., 2024; Chan et al., 2025). This raises a natural question: *can an agent use a small set of task examples to automatically design a reusable vision pipeline?*

Our setting differs from many existing vision agents (Gupta & Kembhavi, 2023; Surís et al., 2023; Yang et al., 2023). In those systems, an LLM or VLM typically remains in the inference loop, reasoning about each test input and dynamically invoking tools, and paying the model's cost and latency on every input. We instead study *agent-designed vision pipelines*: an LLM/VLM agent is used only during an adaptation stage to discover an executable orchestration program. Once discovered, the program is fixed and applied to unseen inputs using only the underlying vision tools, with zero LLM or VLM calls at test time. The agent therefore serves as a pipeline designer rather than as part of the deployed prediction system, and the design cost is paid once, however many inputs the program later serves.

Answering this question requires a benchmark that evaluates the capability, and such evaluation requires more than measuring final task accuracy. If a strong off-the-shelf tool already solves a

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

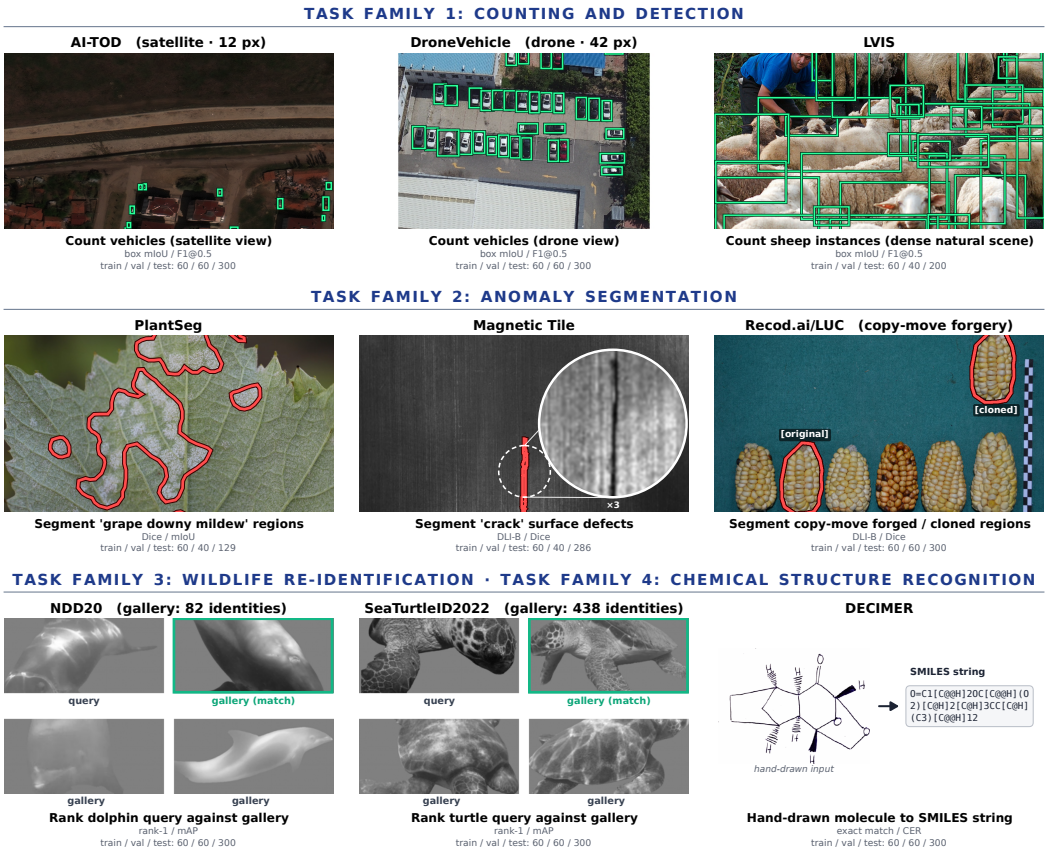


Figure 1: All nine tasks, grouped by the four task families of Table 2. Each panel shows one test example with its ground truth (boxes and the re-identification match in green, segmentation outlines in red, the target string for DECIMER), above a short statement of what the task asks, its ranking metrics and its train / validation / test sizes. The statement is abridged for the figure; Table E.1 gives the request each task carries verbatim. AI-TOD and DroneVehicle form a controlled pair: the request, the tool roster, the metrics and the split sizes are identical, and only the imagery differs, satellite against drone, with the median object size over the test split given beside each name (12 px against 42 px). The circular inset on Magnetic Tile magnifies the outlined crack three times, with contrast stretched. On Recod.ai/LUC, a region of the image has been copied and pasted elsewhere in the same image; the model must segment both duplicated regions (indicated here as [original] and [cloned] for illustration). In the re-identification panels, the query is shown with three of the gallery crops it is ranked against, and the one outlined in green is the same individual.

task well, an automatically generated program may achieve a high score without demonstrating orchestration. We therefore argue that a benchmark purposely designed for evaluating agent-designed vision pipelines should contain only tasks on which the strongest individual available tool does not already come close to solving the task, measured under the same adaptation budget the benchmark grants an agent. Moreover, this headroom depends jointly on the dataset, task formulation, and evaluation metric, so we treat a $(\text{dataset}, \text{task}, \text{metric})$ triplet as the relevant evaluation unit.

We introduce **VisOrch**, a benchmark built around this principle. Each candidate task is admitted using a pre-specified criterion based only on the performance of its constituent tools, independent of the agent being evaluated. The criterion thresholds a normalised performance measure (NPM) that places every metric on a common scale between a trivial baseline and its ceiling, and excludes a task whose strongest zero-shot single tool reaches an NPM of 90 or above (Sec. 3). Because the rule reads tool baselines and never the evaluated system, it can be fixed before any result exists, and a task on which that system later fails to improve stays in. The shortlist behind the admitted

set is reported in full: from a larger pool of candidate datasets we curated 32 candidates, ten of them were probed under the criterion, and nine entered (Appendix B). VisOrch contains nine tasks in four task families: counting and detection, anomaly segmentation, wildlife re-identification, and chemical structure recognition (Figure 1). For each task, an agent receives 60 labelled adaptation examples and a set of vision tools, constructs an executable program, selects it using validation data, and submits one fixed program for evaluation on a blind test split. The deployed program contains no LLM or VLM calls.

We evaluate direct LLM/VLM prediction, one-shot program synthesis, a general coding agent, and iterative program search. On eight of the nine tasks, an agent-designed program exceeds the strongest single tool on both metrics, whereas frontier VLMs prompted directly fall below it on every task they attempt, on both metrics (box mIoU 2.3 against 18.3 on AI-TOD). No single designer is reliable, however. Iterative program search is ahead at $p < 0.05$ on both metrics on four tasks; a general coding agent leads on three others, more than doubling the tool's rank-1 accuracy on re-identification, yet on two of the three segmentation tasks it simply adopts the strongest tool at its default setting and stops; both improve on DECIMER; and on PlantSeg neither improves on both metrics (Sec. 6, Appendix G). Making orchestration dependable across domains, and extending the benchmark beyond its initial nine tasks, remain open problems that VisOrch is built to measure; five screened candidates are already queued for the next phase (Appendix B).

Our contributions are as follows:

- **Offline vision pipeline synthesis.** We formalise a deployment-efficient setting where an agent acts exclusively as an adaptation-time pipeline architect: it writes an executable composition of vision tools that is frozen and evaluated on unseen inputs with zero LLM or VLM calls at test time, paying the design cost once rather than per input.
- **VisOrch: a saturation-screened benchmark.** To ensure high performance reflects tool orchestration rather than a single dominant model, VisOrch admits tasks only if the strongest zero-shot tool scores below an NPM of 90. Guided by this pre-registered criterion, which reads tool baselines only and so prevents post-hoc task selection, we curate nine tasks across four task families, spanning 3,435 total examples with a fixed budget of 60 labelled adaptation images per task and blind test splits.
- **Empirical limits of autonomous orchestration.** Benchmarking direct visual prompting, one-shot synthesis, general coding agents, and iterative program search reveals that no current approach reliably solves multi-tool composition: iterative search significantly outperforms the strongest single tool on four tasks but fails on three others where coding agents succeed; both approaches exceed it on DECIMER; and on two of the three segmentation tasks the coding agent simply collapses to the strongest tool's default setting (Sec. 6).
- **Evaluation pitfalls and protocol safeguards.** We identify five evaluation pitfalls in agent-designed vision pipelines—including shortcut exploitation and degenerate metrics that distort model rankings—and translate them into standardised protocol safeguards (Sec. 7).

2 Related Work

Vision agents and automatic pipeline generation. VisProg (Gupta & Kembhavi, 2023) and ViperGPT (Surís et al., 2023) compose pretrained vision modules into executable programs, scored on final-answer accuracy over VQA and grounding data where one strong model often suffices. MM-ReAct (Yang et al., 2023) and HuggingGPT (Shen et al., 2023) instead let ChatGPT call vision experts or Hugging Face models through prompts, and recent benchmarks evaluate multimodal agents that call tools (Zhu et al., 2026; Su et al., 2026; Liu et al., 2026; Ashraf et al., 2025). All of these keep an LLM or VLM in the inference loop; we study pipelines designed once, offline, and deployed with no LLM or VLM call.

Agent benchmarks and execution environments. Benchmarks with automatically checked outcomes exist for general assistants, software engineering, computer use and ML engineering (Mialon et al., 2024; Jimenez et al., 2024; Xie et al., 2024; Chan et al., 2025), and ToolBench (Qin et al., 2024) scales tool use to more than 16,000 real-world APIs; VisOrch brings the same verification to visual perception, running each delivered program on a blind split. These benchmarks take their

tasks as given; we ask which task belongs in one. Guo et al. (2026) share the suspicion that a tool-call trace need not mean the tool supplied the answer; they test it by ablating the agent, while we test it before any agent runs.

Benchmark saturation and shortcut mitigation. Akhtar et al. (2026) find nearly half of sixty language benchmarks saturated. Two earlier designs inform our response: SuperGLUE (Wang et al., 2019) kept a task only when a strong baseline left a margin to human performance, and BIG-bench (Srivastava et al., 2023) normalises heterogeneous metrics against a low baseline. We apply that normalisation to admission itself: a task enters only if its strongest single tool stays below an NPM of 90, and because the screen never reads the system under test, it cannot be tuned to that system's results. A dataset can also be solvable for reasons the task did not intend (Geirhos et al., 2020): on NDD20 the mean colour of a frame identifies the dolphin eight times as often as the random baseline (Sec. 7), so we treat shortcut controls as part of admission rather than as post-hoc analysis.

3 Task Selection

Saturation screen and task admission. A benchmark task is informative only when off-the-shelf single tools do not already saturate performance. Following BIG-bench (Srivastava et al., 2023), we place heterogeneous metrics on a common scale,

$$\mathrm{NPM}(s) = 100 \cdot \frac{s - s_{\text{trivial}}}{s_{\text{ceiling}} - s_{\text{trivial}}}, \tag{1}$$

where s_{trivial} is the best trivial input-independent baseline, s_{ceiling} the metric optimum, a form that applies unchanged to error metrics, whose ceiling lies below the trivial baseline. Because difficulty shifts across metrics (DECIMER's strongest tool reaches 66.7% exact match but an NPM of 87.2 under character error rate), we screen each $(\text{dataset}, \text{task}, \text{metric})$ triplet. A triplet is admitted only if its strongest zero-shot single tool stays below an NPM of 90, which leaves meaningful headroom for multi-tool orchestration. A single tool is one off-the-shelf model used at a single configuration; a scalar threshold, or one parameter of a standard preprocessing operation, may be selected on the adaptation split, since the evaluated pipelines are granted the same data. Multi-model ensembles and multi-stage pipelines are not single tools, and neither are models trained on data overlapping the validation or test split, which excludes Kaggle leaderboard models for Recod.ai/LUC and MegaDescriptor (Čermák et al., 2024) for NDD20; Appendix A defines the trivial baselines.

Nine tasks across four families pass this screen (Table 1), yielding eighteen triplets whose single-tool NPM ranges from 3.7 to 87.2. Because the screen reads only zero-shot component tools and never the agent under test, no task can be dropped because an evaluated system fails on it; Appendix B gives the full candidate list and Appendix G the sensitivity to the threshold.

4 Building VisOrch

Benchmark protocol. Each VisOrch task specifies a natural-language task description, a set of available vision tools, an input and output format, two evaluation metrics, and adaptation, validation and blind-test splits, used as described in Sec. 5.

Where multiple samples share an underlying source unit—such as a scene, individual, species, semantic category, or drawing batch—we construct the splits at that level to prevent information leakage across adaptation, validation, and test data. The complete task specifications, including split units, tool sets, and data sources, are provided in Appendix E.

Benchmark tasks. VisOrch contains nine tasks grouped into four task families (Figure 1).

- **Counting and detection. AI-TOD** (Wang et al., 2021) (60/60/300) evaluates small-object detection and counting in aerial imagery from xView (Lam et al., 2018); **DroneVehicle** (Sun et al., 2022) (60/60/300) evaluates vehicle detection and counting in RGB drone imagery; and **LVIS** (Gupta et al., 2019) (60/40/200) evaluates category-conditioned instance detection and counting. All three tasks are evaluated using box mIoU and F1@0.5. For LVIS, the adaptation, validation, and test splits contain disjoint categories, so test categories are unseen during pipeline design.

Table 1: Admission screen on the nine admitted tasks (test splits). For each (dataset, task, metric) triplet we run the zero-shot single tools of the task's roster and report the strongest. Trivial floor: the best of constant^c (training median or most frequent answer), whole-image^w, random^r and empty-output^e predictors. Ceiling: the metric optimum. A triplet is excluded at NPM ≥ 90. * Threshold or pre-processing parameter tuned on the training split. Tools are defined in Sec. 5; G-DINO is GroundingDINO, GSAM2 is Grounded-SAM 2, and NN is a nearest neighbour on masked greyscale crops.

Dataset (test n)	Metric	Strongest tool	Tool	Trivial	NPM
AI-TOD (300)	mIoU^{box} ↑	G-DINO (0.10)*	18.25	0.00^w	18.3
	F1@0.5↑	G-DINO (0.10)*	25.12	0.00^e	25.1
DroneVehicle (300)	mIoU^{box} ↑	G-DINO (0.05)*	26.82	0.02^w	26.8
	F1@0.5↑	G-DINO (0.05)*	40.93	0.00^e	40.9
LVIS (200)	mIoU^{box} ↑	SAHI	30.22	0.08^w	30.2
	F1@0.5↑	SAHI	39.27	0.04^w	39.2
PlantSeg (129)	Dice↑	AnomalyCLIP (0.20)*	48.79	18.49^w	37.2
	mIoU↑	AnomalyCLIP (0.40)*	14.19	2.79^w	11.7
Magnetic Tile (286)	DLI-B↑	SAM 2 automatic	28.98	8.82^w	22.1
	Dice↑	GSAM2 (0.40)*	38.04	13.53^w	28.4
Recod.ai/LUC (300)	DLI-B↑	TruFor	33.22	3.73^w	30.6
	Dice↑	TruFor	30.88	9.49^w	23.6
NDD20 (300)	rank-1↑	CLIP NN	7.33	1.33^r	6.1
	mAP↑	CLIP NN	15.79	6.51^r	9.9
SeaTurtleID2022 (300)	rank-1↑	DINOv2 NN	4.33	0.67^r	3.7
	mAP↑	DINOv2 NN	7.34	2.64^c	4.8
DECIMER (300)	exact match↑	DECIMER-HD	66.67	0.00^c	66.7
	CER↓	DECIMER-HD	11.94	93.42^c	87.2

- **Anomaly segmentation. PlantSeg** (Wei et al., 2026) (60/40/129) evaluates plant-disease lesion segmentation, with ten test species absent from the adaptation set; **Magnetic Tile** (Huang et al., 2020) (60/40/286) evaluates surface defect segmentation; and **Recod.ai/LUC** (Recod.ai and Loyola University Chicago, 2025) (60/60/300) evaluates copy-move forgery localisation in scientific figures. PlantSeg is evaluated using per-image Dice and mIoU, while Magnetic Tile and Recod.ai/LUC are evaluated using per-image Dice and DLI-B, a pooled foreground-IoU metric. The definition and behaviour of DLI-B are discussed in Sec. 7.
- **Wildlife re-identification. NDD20** (Trotter et al., 2020) (60/60/300 queries, 82 dolphins) and **SeaTurtleID2022** (Adam et al., 2024) (60/60/300 queries, 438 turtles) require identifying an individual against a gallery. Both tasks are evaluated using rank-1 accuracy and mAP. To eliminate background shortcuts, systems receive masked greyscale crops for both queries and gallery images; Sec. 7 explains this design choice.
- **Chemical structure recognition. DECIMER** (Brinkhaus et al., 2022) (60/60/300) requires transcribing a hand-drawn chemical structure into a SMILES string. Predictions are evaluated using exact match and character error rate after RDKit (Landrum et al., 2026) canonicalisation. An unparsable prediction receives zero exact-match accuracy and a character error rate of one.

Available vision tools. Rosters include GroundingDINO, SAM 2 and Grounded-SAM 2, CLIP and DINOv2 (Oquab et al., 2024), TruFor, and MolScribe with the DECIMER hand-drawn model; Appendix E lists each task's tools, split construction and split sizes.

Data access and licences. Because licences differ across sources, we release split indices, dataset builders and evaluation code rather than images; each split is rebuilt from its source under that source's terms (Appendix E).

5 Protocol and Baselines

Adaptation and model selection. For each task, a system receives 60 labelled adaptation examples, a validation split for model selection, and no access to the blind test split. The system must

ultimately produce a single executable program. Candidate programs may be constructed, evaluated, and refined using the adaptation split, but the final program is selected using validation data and is then run exactly once on the blind test split.

Each task is evaluated using two complementary ranking metrics. When a method produces multiple candidate programs, we rank candidates on the validation split using the sum of their ranks under the two metrics, with shared ranks for ties. Only the selected program is evaluated on the test split. Equations A.1 and A.2 give the selection rule, and Algorithm 1 in Appendix A summarizes the full task-selection and evaluation protocol.

Baselines. We benchmark four increasingly adaptive paradigms alongside the single-tool baseline, each corresponding to a column of Table 2.

- **Off-the-shelf tool (Tool).** For each metric, the strongest eligible individual vision tool from the task's roster, with one scalar threshold or one standard preprocessing parameter selected on the adaptation split; these are the single-tool baselines of the task-selection criterion in Sec. 3.
- **Standalone VLM.** A VLM is prompted directly to solve the visual task without external vision tools. We evaluate `claude-sonnet-4-6` and GPT-5.4. The paradigm does not apply to re-identification, where the output is a ranking over the full gallery.
- **One-shot program synthesis (One-shot).** An LLM writes a single executable program without execution feedback or candidate selection: CodeAct (Wang et al., 2024) for counting, chemical recognition and re-identification, and VisProg (Gupta & Kembhavi, 2023) for segmentation, on the same LLM as the other paradigms.
- **General coding agent (Claude Code).** Claude Code, driven by `claude-sonnet-4-6`, receives the same 60 labelled adaptation examples and follows the same adapt–validate–test protocol; within one session it may inspect training results, revise code and iterate, and it delivers one fixed program.
- **Iterative program search (VTOS).** VTOS (Ge et al., 2026) searches over executable tool programs. Starting from a seed program, it runs $n = 20$ iterations; in each, `claude-sonnet-4-6` proposes $k = 3$ executable Python programs over the task's tools, which are scored on the adaptation split. Because LLM sampling has no reproducible seed, we run $R = 2$ independent searches per task ($R = 1$ on NDD20) and pool their candidates before validation selection; the blind test split is never used in search or selection. Seeds, prompts and task-specific settings are described in Appendix F and released with the code; the single-run ranges and the remaining protocol deviations, namely the AI-TOD code-version change and the PlantSeg rerun after fixing evaluation-harness defects, are in Appendix G.

These paradigms differ in how the program is built, not in deployment: in every case it is fixed before test evaluation and contains no LLM or VLM calls.

Statistical testing. To determine whether a synthesized program improves on its task's strongest single tool, we test the program delivered by iterative program search against that tool (Table 1) on the same test examples: McNemar's test for per-example correctness measures (rank-1 accuracy and exact match), and a paired bootstrap with 10,000 resamples for the remaining metrics. We regard a difference as significant at $p < 0.05$ and additionally report Benjamini–Hochberg adjusted values across the eighteen metric-level comparisons in Table 2; all differences significant at $p < 0.05$ remain significant at $q < 0.05$. The stars and daggers of Table 2 mark this comparison only; the other columns are reported without significance tests.

6 RESULTS

Table 2 compares five approaches on the blind test splits: the strongest off-the-shelf vision tool, direct VLM prediction, one-shot program synthesis, a general-purpose coding agent, and iterative program search.

Gains over single-tool baselines. On eight of the nine tasks, an agent-designed program exceeds the strongest individual tool on both metrics, and iterative program search does so significantly on four of them: LVIS, DroneVehicle, Magnetic Tile and Recod.ai/LUC. On LVIS, the selected program

Table 2: Main results on the blind test splits ($\times 100$; n is the size of that split). Each task is ranked by two complementary metrics so that no single metric decides it; arrows give the direction of improvement. **Bold**: best in row. Underline: second best.

Dataset (n)	Metric	Off-the-shelf	Standalone VLM		Agent-designed program		
		Tool	Sonnet 4.6	GPT-5.4	One-shot	Claude Code	VTOS
Counting and detection							
AI-TOD (300)	mIoU^{box} ↑	18.3	2.3	2.9	7.3	**23.3**	18.3
	F1@0.5↑	25.1	0.9	1.1	10.2	**34.6**	24.8
DroneVehicle (300)	mIoU^{box} ↑	26.8	13.7	11.7	24.6	26.3	**37.3***
	F1@0.5↑	40.9	5.6	4.7	32.1	34.6	**51.2***
LVIS (200)	mIoU^{box} ↑	30.2	4.7	10.4	30.0	35.6	**39.3***
	F1@0.5↑	39.3	1.3	6.4	40.1	48.3	**48.6***
Anomaly segmentation							
PlantSeg (129)	Dice↑	**48.8**	28.6	36.0	36.8	48.2	45.1†
	mIoU↑	14.2	8.6	12.7	16.0	13.0	**17.6***
Magnetic Tile (286)	DLI-B↑	29.0	23.8	17.7	30.6	26.3	**41.5***
	Dice↑	38.0	17.3	19.0	35.9	**56.2**	45.8*
Recod.ai/LUC (300)	DLI-B↑	33.2	10.5	22.7	27.6	33.6	**56.4***
	Dice↑	30.9	13.9	22.4	30.5	30.7	**40.3***
Wildlife re-identification							
NDD20 (300)	rank-1↑	7.3	n/a‡	n/a‡	8.3	**16.3**	7.3
	mAP↑	15.8	n/a‡	n/a‡	15.1	**25.3**	14.8
SeaTurtleID2022 (300)	rank-1↑	4.3	n/a‡	n/a‡	3.0	**10.3**	3.3
	mAP↑	7.3	n/a‡	n/a‡	6.5	**16.7**	6.8
Chemical structure recognition							
DECIMER (300)	exact match↑	66.7	32.7	15.0	69.7	**72.7**	70.7
	CER↓	11.9	25.6	43.8	10.1	**8.5**	**8.5***

Columns. *Tool*: the strongest off-the-shelf model of the task's roster, one threshold tuned on the 60 adaptation images. *One-shot*: CodeAct (Wang et al., 2024) on counting, chemical OCR and re-identification, VisProg (Gupta & Kembhavi, 2023) on segmentation. *Claude Code*: one Claude Code session. *VTOS*: iterative program search (Ge et al., 2026). All four run `claude-sonnet-4-6`; the columns are defined in Section 5. ‡ not applicable: a re-identification answer ranks the whole gallery, which is the retrieval the *Tool* column already measures. */† VTOS better/worse than the tool at $p < 0.05$ (paired bootstrap; McNemar for rank-1 and exact match).

raises box mIoU from 30.2 to 39.3 and F1 from 39.3 to 48.6; on DroneVehicle, mIoU from 26.8 to 37.3 and F1 from 40.9 to 51.2. The largest gain is on Recod.ai/LUC, where combining SIFT-based matching with the TruFor heat map raises DLI-B from 33.2 to 56.4 and Dice from 30.9 to 40.3. Composing tools is what produces these gains: prompted directly, without tools, a VLM falls below the strongest tool on every task it can attempt, and on LVIS the stronger of the two standalone VLMs reaches 10.4 mIoU against 30.2 for the tool and 39.3 for program search.

Dedicated search versus a general coding agent. No single way of designing the program wins everywhere. Where search is not significantly better than the tool, the general-purpose coding agent often is: on AI-TOD, iterative program search is statistically indistinguishable from the strongest individual detector, whereas the coding agent reaches 23.3 mIoU and 34.6 F1 against 18.3 and 25.1 for the tool; on re-identification it raises rank-1 accuracy from 7.3 to 16.3 on NDD20 and from 4.3 to 10.3 on SeaTurtleID2022; and on DECIMER it obtains the highest exact match in Table 2 (72.7%) while matching the best CER of 8.5. The same agent, however, often does not compose tools at all: on two of the three segmentation tasks, although handed the task's full tool roster, it adopts the strongest tool at its default setting and stops (Appendix G). Across the six counting and segmentation tasks, it outperforms iterative program search on both metrics on one task, yields a mixed result on two, and is behind on the remaining three, so strong pipeline design is not specific to a purpose-built search algorithm, and which approach leads depends on the task.

Metric trade-offs, strong specialists and one-shot synthesis. Two tasks show where gains are hard to obtain. On PlantSeg, AnomalyCLIP alone achieves 48.8 Dice and 14.2 mIoU, whereas the selected search program obtains 45.1 Dice and 17.6 mIoU; the increase in mIoU is significant ($p = 0.012$), as is the decrease in Dice ($p = 0.020$), so the discovered pipeline improves instance-level region separation at the cost of overall pixel overlap. On DECIMER, the strongest individual model already reaches 66.7% exact match and a character error rate (CER) of 11.9; the searched program reconciles the outputs of two chemical-recognition models (Appendix F), reducing CER to 8.5 ($p < 0.001$) and eliminating unparsable outputs, while its exact match of 70.7% is not significantly higher ($p = 0.058$). The one-shot writers, CodeAct and VisProg, generate a single program without execution feedback, revision or validation-based search; they are competitive with the strongest tool on some tasks but fall well short of the paradigms that adapt on the labelled examples, reaching 24.6 mIoU on DroneVehicle against 37.3 for iterative program search, and 27.6 DLI-B on Recod.ai/LUC against 56.4.

Headroom on segmentation. Prompting SAM 2 with the ground-truth boxes gives 83.0 Dice on PlantSeg, 83.9 on Magnetic Tile and 89.2 on Recod.ai/LUC, against 48.8, 56.2 and 40.3 for the best method on each in Table 2. The generic mask tools, SAM 2 and Grounded-SAM 2, sit at high pixel recall and low precision on all three tasks, Grounded-SAM 2 on Recod.ai/LUC being the extreme case at 89.3 recall and 6.7 precision, whereas the box-prompted oracle is high on both (Table G.7). The remaining gap is in finding the regions, not in delineating them.

7 Evaluation Pitfalls and Protocol Safeguards

Building VisOrch revealed five evaluation issues that can confound the measurement of agent-designed vision pipelines. The first four directly affected model ranking or interpretation and therefore motivated explicit safeguards in the final protocol.

(a) Scene shortcuts. On NDD20, background information alone is sufficient to identify individuals far above chance: even after masking the dolphin itself, the remaining scene matches the strongest tool. On the 120 training and validation queries, the mean colour of a full frame, three numbers in all, retrieves the right individual for 10.0% of queries against a random baseline of 1.3%, and the background with every dolphin blanked out reaches 12.5% (Table C.1). We therefore evaluate re-identification using masked greyscale crops rather than full frames, on which every control falls to chance while the strongest tool keeps 8.3%.

(b) Degenerate metrics. Some metrics can reward trivial or undesirable behaviour. On counting MAE, a constant predictor based on the adaptation-set median outperforms every evaluated tool, while DLI-B can favour solutions that ignore small defects: on Magnetic Tile, raising the Grounded-SAM 2 threshold from 0.30 to 0.55 lifts test DLI-B from 11.8 to 19.9 while Dice and mIoU nearly halve. We therefore exclude the former from ranking and pair pooled metrics with per-image metrics where necessary.

(c) Selection noise. Validation splits of 40–60 examples can be too small to reliably distinguish closely performing programs. In several cases, the ordering of candidate programs on validation does not match their ordering on the blind test split. On Magnetic Tile, the second search leads the first by 10.3 validation DLI-B points and trails it by 4.5 on test; on DECIMER, the delivered program leads on validation exact match (85.0 against 81.7) yet trails the first search's own pick on test (70.7 against 72.7). We therefore report single-run variation and treat model selection uncertainty as a limitation of the benchmark rather than hiding it through repeated test selection.

(d) Evaluation-harness failures. A candidate program should be penalised only for failures attributable to the program itself. We identified three cases in which sandbox or execution-harness limitations incorrectly reduced candidate scores. These failures are now separated from genuine program failures in the evaluation protocol.

(e) Non-transferable adaptation. Improvements measured on the adaptation split do not necessarily generalise. On LVIS, a coding agent found that the detector answers reworded class names ("grape berry" returns 24 instances where "grapes" returns 10) and hard-coded such aliases for training categories; none of them appears in the 200 test records, so the gain measured during adaptation is worth nothing on the disjoint test categories. This motivates constructing splits around the appro-

priate independence unit, rather than allowing related categories or source objects to cross between adaptation and test data.

Appendix J gives further details on these cases; Table C.1 and Appendices C, D, and F provide the corresponding supporting analyses.

8 Limitations and Conclusion

We introduced VisOrch, a benchmark for whether an AI agent can turn a small labelled adaptation set into a reusable vision pipeline that runs with no LLM or VLM, on tasks chosen so that their strongest individual tool leaves measurable headroom.

Our experiments show that this headroom can be exploited, but not uniformly. Iterative program search substantially improves over the strongest individual tool on several tasks, while a general-purpose coding agent is stronger on others. On PlantSeg, neither improves on both metrics over the best available component. This variation is an intended property of the benchmark: successful pipeline design should be demonstrated rather than assumed.

The current benchmark has several limitations. First, it covers nine tasks in four task families and therefore represents only a subset of the visual problems for which agent-designed pipelines may be useful. Second, all program-synthesis and agent baselines use the same underlying LLM, `claude-sonnet-4-6`, so the results do not establish how strongly the conclusions depend on the choice of language model. Third, the validation splits contain only 40–60 examples, which can make selection among closely performing programs unstable (Sec. 7). Finally, iterative search is stochastic: we use $R = 2$ runs for most tasks and report run-level variation rather than treating it as measurement noise. We expect the design principle, admitting a task only where individual tools leave measurable headroom, to generalise beyond the current task set.

AI use statement

We used a large language model assistant (Claude, Anthropic) in this work, in the four ways declared on the submission form and in no others.

To aid or polish writing. It edited prose in this paper for clarity and concision.

For retrieval and discovery. It searched for related work, identified relevant literature, and checked reference records.

For research ideation or execution. It proposed and refined hypotheses about why individual tasks fail, gave feedback on experimental design, implemented methods (the tasklet builders, the evaluation code and the baseline harnesses), cleaned and reformatted the source datasets into tasklets, and helped interpret results.

To draft sections of the paper. It produced first drafts of sections, which the authors then revised.

We have reviewed all AI-assisted work and take responsibility for the final content of this paper. Code was reviewed and run by the authors; every reported number was traced back to the result file it comes from; every reference was verified against an independent bibliographic record.

Ethics statement

This work introduces no new image collection. Every task is built from a published dataset, used under its own licence and within the terms that licence sets: AI-TOD requires separate access to xView, NDD20 is non-commercial share-alike, SeaTurtleID2022 forbids redistribution and commercial use, and Recod.ai/LUC is share-alike under its competition rules. We therefore release split indices, builders and evaluation code rather than images. The only images the paper itself reproduces are the nine single frames of Figure 1, shown for illustration and attributed to their sources. Section 4 names the source and the licence of each task.

No task involves human subjects and none of the source data is personal. Two tasks use photographs of wild animals (dolphins, sea turtles) collected by the original authors under their own permits; we use them only for re-identification benchmarking, which is the purpose for which they were

released, and we do not publish locations. One task uses figures from biomedical papers that a published forensic dataset marks as manipulated: we treat those labels as the dataset's, make no claim about any specific publication or author, and report only aggregate localisation metrics. A method that localises manipulated regions can in principle be used to make a manipulation harder to detect; the tools we compose are already public, and we judge the benefit to integrity screening to outweigh that risk.

Reproducibility statement

The screen, the splits and the protocol are specified in Sections 3–5: Equation 1 defines the admission metric, Section 4 gives each task's splits, independence unit, near-duplicate rule and tool roster, and Section 5 gives the training budget, the selection rule, the best-of-R protocol and the significance tests, all of which were fixed before the runs. Appendix E lists the task cards, Appendix F the seed programs, prompts and search budgets, and Appendix G the per-task results behind every table.

We will release the tasklet builders, the split indices, the evaluation code, the seed programs and prompts, the per-image predictions of every delivered program and every baseline, and the token logs of the runs for which logging was already enabled (Appendix G). Two sources of variance cannot be removed and are reported instead of hidden: LLM sampling has no fixable seed, so we run $R = 2$ independent searches per task and give the range of the single-run test scores next to each result; and validation splits of 40–60 images are small enough to reorder close programs, as Section 7 shows. The deviations from the protocol are stated where they occur: the two AI-TOD searches on either side of an audit fix, the single NDD20 search, and the PlantSeg re-run after the search defects of Section 7 were fixed.

References

Lukáš Adam, Vojtěch Čermák, Kostas Papafitsoros, and Lukas Picek. SeaTurtleID2022: A long-span dataset for reliable sea turtle re-identification. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, 2024. doi: 10.1109/wacv57701.2024.00699. arXiv:2211.10307.

Mubashara Akhtar, Anka Reuel, Prajna Soni, et al. When AI benchmarks plateau: A systematic study of benchmark saturation. In *International Conference on Machine Learning (ICML)*, 2026. arXiv:2602.16763.

Tajamul Ashraf et al. Agent-X: Evaluating deep multimodal reasoning in vision-centric agentic tasks. *arXiv preprint arXiv:2505.24876*, 2025.

Henning Otto Brinkhaus, Achim Zielesny, Christoph Steinbeck, and Kohulan Rajan. DECIMER—hand-drawn molecule images dataset. *Journal of Cheminformatics*, 14(36), 2022. doi: 10.1186/s13321-022-00620-9.

Vojtěch Čermák, Lukas Picek, Lukáš Adam, and Kostas Papafitsoros. WildlifeDatasets: An open-source toolkit for animal re-identification. In *IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 5953–5963, 2024. doi: 10.1109/wacv57701.2024.00585. arXiv:2311.09118.

Jun Shern Chan, Neil Chowdhury, Oliver Jaffe, James Aung, Dane Sherburn, Evan Mays, Giulio Starace, Kevin Liu, Leon Maksin, Tejal Patwardhan, Lilian Weng, and Aleksander Mądry. MLE-bench: Evaluating machine learning agents on machine learning engineering. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.07095.

Jinchao Ge, Lingqiao Liu, Shuwen Zhao, and Lei Wang. VTOS: Learning to orchestrate vision tools by co-searching solutions and observers. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2026. arXiv:2606.20728.

Robert Geirhos, Jörn-Henrik Jacobsen, Claudio Michaelis, Richard Zemel, Wieland Brendel, Matthias Bethge, and Felix A. Wichmann. Shortcut learning in deep neural networks. *Nature Machine Intelligence*, 2:665–673, 2020. doi: 10.1038/s42256-020-00257-z.

Jiawei Guo et al. Do multimodal agents really benefit from tool use? A systematic study of capability gains. *arXiv preprint arXiv:2606.02357*, 2026.

Agrim Gupta, Piotr Dollár, and Ross Girshick. LVIS: A dataset for large vocabulary instance segmentation. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5356–5364, 2019. doi: 10.1109/cvpr.2019.00550. arXiv:1908.03195.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 14953–14962, 2023. doi: 10.1109/cvpr52729.2023.01436. arXiv:2211.11559.

Yibin Huang, Congying Qiu, and Kui Yuan. Surface defect saliency of magnetic tile. *The Visual Computer*, 36(1):85–96, 2020. doi: 10.1007/s00371-018-1588-5.

Carlos E. Jimenez, John Yang, Alexander Wettig, Shunyu Yao, Kexin Pei, Ofir Press, and Karthik Narasimhan. SWE-bench: Can language models resolve real-world GitHub issues? In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2310.06770.

Darius Lam, Richard Kuzma, Kevin McGee, Samuel Dooley, Michael Laielli, Matthew Klaric, Yaroslav Bulatov, and Brendan McCord. xView: Objects in context in overhead imagery. *arXiv preprint arXiv:1802.07856*, 2018.

Greg Landrum et al. RDKit: Open-source cheminformatics software. Zenodo, 2026.

Zinuo Li, Xuhang Chen, Chi-Man Pun, and Xiaodong Cun. High-resolution document shadow removal via a large-scale real-world dataset and a frequency-aware shadow erasing net. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 12449–12458, 2023. doi: 10.1109/iccv51070.2023.01144. arXiv:2308.14221.

Shilong Liu, Zhaoyang Zeng, Tianhe Ren, Feng Li, Hao Zhang, Jie Yang, Qing Jiang, Chunyuan Li, Jianwei Yang, Hang Su, Jun Zhu, and Lei Zhang. Grounding DINO: Marrying DINO with grounded pre-training for open-set object detection. In *European Conference on Computer Vision (ECCV)*, pp. 38–55, 2024. doi: 10.1007/978-3-031-72970-6_3. arXiv:2303.05499.

Yunqi Liu et al. EgoBench: An interactive egocentric multimodal benchmark for tool-using agents. *arXiv preprint arXiv:2605.27820*, 2026.

Grégoire Mialon, Clémentine Fourrier, Craig Swift, Thomas Wolf, Yann LeCun, and Thomas Scialom. GAIA: A benchmark for general AI assistants. In *International Conference on Learning Representations (ICLR)*, 2024. arXiv:2311.12983.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mahmoud Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Hervé Jégou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. arXiv:2304.07193.

Yujia Qin, Shihao Liang, Yining Ye, Kunlun Zhu, Lan Yan, Yaxi Lu, Yankai Lin, Xin Cong, Xiangru Tang, Bill Qian, Sihan Zhao, Lauren Hong, Runchu Tian, Ruobing Xie, Jie Zhou, Mark Gerstein, Dahai Li, Zhiyuan Liu, and Maosong Sun. ToolLLM: Facilitating large language models to master 16000+ real-world APIs. In *International Conference on Learning Representations*, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning (ICML)*, pp. 8748–8763, 2021. arXiv:2103.00020.

Nikhila Ravi, Valentin Gabeur, Yuan-Ting Hu, Ronghang Hu, Chaitanya Ryali, Tengyu Ma, Haitham Khedr, Roman Rädle, Chloe Rolland, Laura Gustafson, Eric Mintun, Junting Pan, Kalyan Vasudev Alwala, Nicolas Carion, Chao-Yuan Wu, Ross Girshick, Piotr Dollár, and Christoph Feichtenhofer. SAM 2: Segment anything in images and videos. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2408.00714.

Recod.ai and Loyola University Chicago. Recod.ai/LUC scientific image forgery detection. Kaggle competition, 2025. https://www.kaggle.com/competitions/recodai-luc-scientific-image-forgery-detection.

Yongliang Shen, Kaitao Song, Xu Tan, Dongsheng Li, Weiming Lu, and Yueting Zhuang. HuggingGPT: Solving AI tasks with ChatGPT and its friends in Hugging Face. In *Advances in Neural Information Processing Systems*, volume 36, 2023. doi: 10.52202/075280-1657.

Ray Smith. An overview of the Tesseract OCR engine. In *Ninth International Conference on Document Analysis and Recognition (ICDAR 2007)*, volume 2, 2007. doi: 10.1109/ICDAR.2007.4376991.

Aarohi Srivastava et al. Beyond the imitation game: Quantifying and extrapolating the capabilities of language models. *Transactions on Machine Learning Research*, 2023. arXiv:2206.04615.

Zhaochen Su et al. AgentVista: Evaluating multimodal agents in ultra-challenging realistic visual scenarios. *arXiv preprint arXiv:2602.23166*, 2026.

Yiming Sun, Bing Cao, Pengfei Zhu, and Qinghua Hu. Drone-based RGB-infrared cross-modality vehicle detection via uncertainty-aware learning. *IEEE Transactions on Circuits and Systems for Video Technology*, 32(10):6700–6713, 2022. doi: 10.1109/TCSVT.2022.3168279.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via python execution for reasoning. In *IEEE/CVF International Conference on Computer Vision (ICCV)*, 2023. doi: 10.1109/iccv51070.2023.01092. arXiv:2303.08128.

Cameron Trotter, Georgia Atkinson, Matt Sharpe, Kirsten Richardson, A. Stephen McGough, Nick Wright, Ben Burville, and Per Berggren. NDD20: A large-scale few-shot dolphin dataset for coarse and fine-grained categorisation. *arXiv preprint arXiv:2005.13359*, 2020.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2019. arXiv:1905.00537.

Jinwang Wang, Wen Yang, Haowen Guo, Ruixiang Zhang, and Gui-Song Xia. Tiny object detection in aerial images. In *International Conference on Pattern Recognition (ICPR)*, pp. 3791–3798, 2021. doi: 10.1109/ICPR48806.2021.9413340.

Xingyao Wang, Yangyi Chen, Lifan Yuan, Yizhe Zhang, Yunzhu Li, Hao Peng, and Heng Ji. Executable code actions elicit better LLM agents. In *International Conference on Machine Learning (ICML)*, pp. 50208–50232, 2024. arXiv:2402.01030.

Tianqi Wei, Zhi Chen, Xin Yu, Scott Chapman, Paul Melloy, and Zi Huang. A large-scale in-the-wild dataset for plant disease segmentation. *Scientific Data*, 13:205, 2026. doi: 10.1038/s41597-025-06513-4. arXiv:2409.04038.

Tianbao Xie et al. OSWorld: Benchmarking multimodal agents for open-ended tasks in real computer environments. In *Advances in Neural Information Processing Systems (NeurIPS), Datasets and Benchmarks Track*, 2024. arXiv:2404.07972.

Zhengyuan Yang, Linjie Li, Jianfeng Wang, Kevin Lin, Ehsan Azarnasab, Faisal Ahmed, Zicheng Liu, Ce Liu, Michael Zeng, and Lijuan Wang. MM-ReAct: Prompting ChatGPT for multimodal reasoning and action. *arXiv preprint arXiv:2303.11381*, 2023.

Xuanyu Zhu et al. VTC-bench: Evaluating agentic multimodal models via compositional visual tool chaining. *arXiv preprint arXiv:2603.15030*, 2026.

A THE SCREEN END TO END

Trivial baselines. For each task, s_{trivial} is the best-performing input-independent or minimally informative baseline applicable to it: a constant predictor (the training-set median for counting or the most frequent training answer where applicable), predicting the entire image as one box or mask, random ranking for re-identification, and an empty prediction. The screen is applied per triplet rather than per dataset because the same dataset can change difficulty with its formulation or metric: on PlantSeg the strongest tool reaches 37.2% of the NPM range under Dice but only 11.7% under per-instance mIoU (Table 1). DECIMER, whose specialist model reaches 66.7% exact match, is the strongest retained baseline; it stays because its headroom passes the criterion, which lets the benchmark test whether a pipeline can improve on a strong specialist.

The selection rule of Section 5, written out. A candidate p is ordered on a split D by the Borda rank sum of the task's two ranking metrics m_1, m_2, each oriented so that higher is better (error metrics are negated), with shared ranks for ties and remaining ties broken by m_1, then by the earlier candidate:

$$\rho(p, D) = \mathrm{rk}_{m_1}(p, D) + \mathrm{rk}_{m_2}(p, D), \qquad \mathrm{rk}_m(p, D) = 1 + \big|\{q : m(q, D) > m(p, D)\}\big|, \quad \text{(A.1)}$$

Writing C_r for the candidates of search r and S for the five best of each by training score together with the seed, the delivered program is

$$p^\star = \arg\min_{p \in S} \ \rho(p, D_{\text{val}}), \qquad S \subseteq \textstyle\bigcup_{r=1}^{R} C_r, \quad \text{(A.2)}$$

and only $p^\star$ is ever run on D_{test}.

Algorithm 1 Admission and delivery for one (dataset, task, metric) triplet

Require: tool roster $\mathcal{T}$, splits D_{train} (60 images), D_{val}, blind D_{test}, ceiling c, trivial predictor t, searches R

$s \leftarrow \max_{\tau \in \mathcal{T}} \ \mathrm{score}(\tau, D_{\text{test}})$ ▷ zero-shot; one threshold may be tuned on D_{train}
$\mathrm{NPM} \leftarrow 100 \cdot (s - t)/(c - t)$
if $\mathrm{NPM} \geq 90$ **then**
  return excluded ▷ decided before any system is run
end if
for $r = 1 \ldots R$ **do**
  $C_r \leftarrow \textsc{Search}(D_{\text{train}}, \mathcal{T})$ ▷ n iterations $\times$ k proposals, scored on D_{train}
end for
$S \leftarrow$ the five best of each C_r by training score, plus the seed
$p^\star \leftarrow \arg\min_{p \in S} \ \rho(p, D_{\text{val}})$ ▷ Borda over the two ranking metrics
return $\mathrm{score}(p^\star, D_{\text{test}})$ ▷ the delivered program is run on D_{test} once; D_{test} never enters selection

B THE SCREENED CANDIDATES

Before any probe is run, a curated shortlist is screened against published numbers. This pre-screen decides nothing on its own: it orders the queue, and a candidate leaves the queue only through the measured probe. The table is recorded here so that the path from the candidate pool to the nine admitted tasks is visible, including the candidates that are still waiting. Its *Area* column is the literature a candidate was drawn from, not one of the four task families of Section 4: those name what a task asks, and the counting family draws from both the detection and the counting areas.

Table B.1 holds the 32 curated candidates, drawn from a larger pool of candidate datasets, and every one of them carries a stated reason for where it went.

Ten sources have been measured by us: nine entered, and SD7K was measured and rejected at a CER NPM of 93.1. Thirteen were excluded without a measurement of ours, and the reason divides them. For seven the published record already shows no headroom to find: six sit on leaderboards their own sources call saturated or near-tied, four of them with a stated margin (GWHD 0.5 pp between the top two, RealRain-1k 0.3 dB, KolektorSDD2 0.2 pp, Severstal 0.01 across the top fifty), and DRIVE's best published model exceeds its own second annotator. The other six fail on grounds a

Table B.1: The 32 curated candidates and where each went; all nine admitted tasks appear in the first group. "Lit. NPM" is not a measurement of ours: it uses the best published specialist model, trained on the full training data; it is reference information only, except where that model was trained on a budget comparable to ours. These figures were read from the candidates' papers and public leaderboards during the pre-screen and we did not reproduce them; no admission decision rests on them. Admission is decided by the measured zero-shot probe (Table 1). The wider pool from which the 32 were drawn has not been screened.

Area	Candidate	Lit. NPM	Reason
Admitted after the measured probe			
Detection	AI-TOD (Wang et al., 2021)	34.6	probe in Table 1
Anomaly	Magnetic Tile (Huang et al., 2020)	75–83	mIoU with background as a class gives a blank prediction 45.6
OCR	DECIMER hand-drawn (Brinkhaus et al., 2022)	73.3	printed-structure recogniser: 92.6% printed, 7.65% hand-drawn
Forgery	Recod.ai/LUC (Recod.ai and Loyola University Chicago, 2025)	—	no published number; probe measured by us
Detection	DroneVehicle (Sun et al., 2022)	63.2	probed: NPM 26.8 / 40.9 (Table 1)
Re-ID	SeaTurtleID2022 (Adam et al., 2024)	86.8–95.0	probed: zero-shot rank-1 4.3%, NPM 3.7 (Table 1)
Counting	LVIS (Gupta et al., 2019)	—	probe in Table 1
Segmentation	PlantSeg (Wei et al., 2026)	—	probe in Table 1
Re-ID	NDD20 (Trotter et al., 2020)	—	probe in Table 1
Published headroom, queued for measurement (Phase 2)			
Counting	Maize Tassels (MTC)	88.4	floor derived from reported R^2 and RMSE; 361 images
Counting	DroneCrowd (by clip)	78–85	adjacent frames near-identical; split by clip
Segmentation	Rice Panicle UAV	—	no leaderboard; CC BY 4.0
Segmentation	MoNuSeg (AJI / PQ)	70–73	Dice bunched at 0.82–0.84; AJI 0.656
OCR	CROHME	79.7	ExpRate 79.7
Pending a framing or licence decision			
Counting	TRANCOS	88–92	near the line; GAME(0) ignores location
Counting	CVPPP leaf counting	54–82	licence limited to challenge use
Segmentation	CubiCasa-5K	—	room labelling reported as near-saturated, vectorisation far from it; we did not probe it
Estimation	GrapeCS-ML	—	no published benchmark
Not admitted (SD7K measured and rejected; the rest on licence, scope or the record)			
Segmentation	DRIVE vessels	107.8	beyond the second annotator
Enhancement	UIEB	84–88	redistribution forbidden
Enhancement	SD7K (Li et al., 2023)	47–80	probed: background division + Tesseract, CER NPM 93.1 (Appendix G)
Forgery	DocTamper	91.3	password-gated access
Estimation	NewFor	—	LiDAR point clouds, 14 plots
Re-ID	SealID	92.7	57 identities
Re-ID	Cattle Muzzle	99.1–99.5	closed-set classification
Re-ID	LeopardID2022	90.9	redundant with the ID2022 family
Deprioritised: the published record leaves no headroom to measure			
Counting	CARPK / PUCPR+	91–96	top-3 near-tied; localisation not saturated
Detection	PCB Defect	98.9	YOLOv11 mAP50 98.9
Detection	GWHD 2021	70.0	top-2 within 0.5 pp
Enhancement	RealRain-1k	90.5–93.1	top-2 within 0.3 dB
Anomaly	KolektorSDD2	99.1	top-2 within 0.2 pp
Anomaly	Severstal	39.3	top-50 within 0.01

measurement would not change: two forbid redistribution or gate access, three fall outside the input or label form the benchmark evaluates, and one is redundant with a dataset already admitted. The remaining nine are deferred rather than settled: five show published headroom and are queued for Phase 2, and four await a framing or licence decision.

LVIS, PlantSeg, NDD20 and Recod.ai/LUC carry no figure in the pre-screen column: no published-record figure was recorded for them, and their admission rests on the measured probe of Table 1 like every other admitted task.

C NDD20 Shortcut Controls

The controls behind Section 7(a): what each control retrieves on the 120 training and validation queries, on full frames and on the masked greyscale crops the benchmark actually serves.

Table C.1: Scene shortcut on NDD20: rank-1 hits out of 120 training and validation queries (random baseline 1.3%, as in Table 1). Each p is a one-sided binomial test against the 1.22% chance level of a uniform guess over the 82 identities. On full frames, the frame's mean colour or its background alone comes close to a re-identification tool. On masked greyscale crops every control falls to chance, while the strongest tool stays above it. The admission criterion was fixed before the masked controls were run: every control ≤ 5 hits and the strongest tool ≥ 6.

Input to the matcher	Hits	rank-1 (%)	p
Full RGB frames (as released)			
Strongest tool: CLIP, whole frame	17	14.2	2×10^{-13}
Strongest tool: DINOv2, dolphin box	16	13.3	2×10^{-12}
Mean frame colour (3 numbers)	12	10.0	3×10^{-8}
Background only (dolphin boxes blanked)	15	12.5	3×10^{-11}
same, stricter grouping (cosine 0.90)	13	10.8	3×10^{-9}
Masked greyscale crops (released format)			
Strongest tool: DINOv2	10	8.3	3×10^{-6}
Mean grey level inside the mask	4	3.3	0.06
Contrast and sharpness	4	3.3	0.06
Silhouette only	3	2.5	0.18
Original crop size and aspect ratio	3	2.5	0.18
Shape scalars	1	0.8	0.77

D Selection Noise

Section 7(c) gives two cases, on Magnetic Tile and DECIMER, in which a lead on the 40–60 validation images does not carry to the blind test; the third is on Recod.ai/LUC. On Recod.ai/LUC the Borda ranks are recomputed over both shortlists, which promotes a candidate its own run had placed second; that candidate then scores 12.8 DLI-B points above its run's own pick on test. Test never takes part in selection, so all three stay in the reported results.

E Tasklet Cards

Each card gives the source and licence, the split sizes and the independence unit, the metric pair, and the tool roster the search may call. Besides the tools, generated programs may use standard algorithmic operations such as filtering, non-maximum suppression, geometric processing, feature matching and post-processing. We use up to 300 blind test examples where the source permits; some datasets contain fewer eligible examples once the independence, near-duplicate and source-specific split constraints are applied, and there we keep the largest valid test split rather than relax those constraints. Split sizes are train / val / test. Table E.1 gives the request each task actually carries, verbatim from its `benchmark_*.json`; it is the only text a method receives about what to do, and the four families of Section 4 are read off it. The anomaly family is prompted, not unsupervised: each request names the condition to find—a disease, a defect type, copy-move tampering—so these

tasks are not the normal-samples-only setting the anomaly detection literature usually means by the term.

Table E.1: **What each task asks.** The request verbatim from the task's `benchmark_test.json`, which is the whole of what a method is told. *Variants* is the number of distinct requests on the test split: one where every image carries the same sentence, more where the category, disease or defect type changes per image, in which case one of them is shown. The four families group the requests by what they ask for—a count, a region, an identity, or a string—and not by the literature each source dataset came from (Table B.1).

Family	Task	Variants	Request
Counting and detection	AI-TOD	1	*How many vehicles are visible in this aerial image?*
	DroneVehicle	1	*How many vehicles are visible in this aerial image?*
	LVIS	10	*How many sheep are visible in this image?*
Anomaly segmentation	PlantSeg	21	*Localize the regions of 'citrus canker' on this Citrus leaf or plant.*
	Magnetic Tile	5	*Localize the 'blowhole' defect regions on this magnetic tile surface.*
	Recod.ai/LUC	1	*Localize the copy-move forged regions in this scientific figure.*
Wildlife re-identification	NDD20	1	*Which of the gallery's known dolphins is the one in the marked region? Rank the gallery identities, best first.*
	SeaTurtleID2022	1	*Which of the gallery's known turtles is the one in the marked region?*
Chemical structure recognition	DECIMER	1	*Read this hand-drawn chemical structure and output its SMILES string.*

Tool rosters. A roster is the set of names bound in the program sandbox before a search starts; the engine can bind further names during a run, and every name a program calls must be bound this way. The single-tool baselines of a family use the same roster; the LLM-agent baselines use a documented subset of it (Sec. 5). *Counting* (AI-TOD, DroneVehicle, LVIS): the manifest shown to the LLM declares `grounding_dino_detect(image_path, text_query, box_threshold)` and `sam_refine_bboxes(image_path, bboxes)`, plus any tool the search registers during a run; the contract variable is `bboxes`. *Segmentation* (PlantSeg): `detect`, `segment`, `propose_masks`, `clip_select`, `merge_polygons`, `subtract_polygons`, `filter_polygons_by_area`, `clean_mask`, `plant_healthy_prompt`, `plant_disease_prompt`, `anomaly_segment`; contract variables `final_bboxes` and `final_polygons`. Magnetic Tile drops the two plant-prompt tools (its tasks carry no species) and `anomaly_segment`, which entered the roster after the search that delivered its program; Recod.ai/LUC drops the plant prompts and `anomaly_segment` and adds `trufor`. *Re-identification* (NDD20, SeaTurtleID2022): `embed_dinov2`, `embed_clip`, `gallery_embeddings`, `gallery_crop`, `cosine_matrix`, `rank_ids`, `autocontrast`, `equalize`, `hflip`, `resize`; contract variable `final_ranked_ids`. *Chemical structure recognition* (DECIMER): `decimer`, `molscribe`, `molscribe_many`, `canonicalize`, `is_valid`, `mol_props`, and the image operations `to_gray`, `binarize`, `denoise`, `thicken`, `thin`, `crop_to_content`, `pad`, `resize`, `rotate`, `invert`, `autocontrast`; contract variables `final_smiles` and `final_confidence`. Every sandbox also binds `np`, `cv2`, `torch` and the image. Neither sandbox confines a program to its roster: the counting sandbox places no restriction on builtins at all, and the segmentation one passes an explicit set of thirty-nine that still includes `__import__`. A roster fixes what is bound *without* an import, not what a program may reach; of the 53 candidates of the delivered AI-TOD search, 52 import numpy and one imports `tempfile`, and none imports a model library although nothing stopped them. The coding-agent baseline, which loads GroundingDINO from `transformers` itself, is therefore not working in a larger space than the search was given; it used a freedom the search had and did not take. A program that leaves a contract variable unassigned is recorded as a failed program, not as an empty prediction (Sec. 7).

The harness defects, in numbers. Section 7(d) describes three defects of the search harness; these are their measured extents. The per-candidate wall-clock budget of 420 s discarded 14 of the 92 PlantSeg candidates of the two pre-fix searches (12 of 53 in one, 2 of 39 in the other), several of them after 48 of the 60 training images had already been scored, and the candidates it struck were the heavier detect-then-segment programs. The missing sandbox builtins (`id`, `divmod`) crashed one Magnetic Tile candidate on all 60 training images and one PlantSeg candidate on 55 of 60. Two further PlantSeg candidates assigned `final_polys` instead of `final_polygons` and scored 0 on every image with no error recorded. None of the three fired on the Magnetic Tile or Recod.ai/LUC searches that delivered their programs, which is why only PlantSeg was re-run.

A fourth instance of the same failure mode appeared on the other side of the protocol, in the coding-agent runs, and is worth recording because it is a hazard for anyone scoring a program an agent delivers rather than one a harness generated. The delivery contract asks a program to return empty lists rather than raise when it finds nothing, which is the right contract for a detector that sees no object. Two of the three delivered segmentation programs generalised it into a bare `except Exception` around their whole body; the third did not, and when the same import failed for it, it raised and the run stopped. The two failure modes are one bug and differ only in whether it is visible. When they were first run outside the directory they were written in, an import of a helper module in that directory failed, the blanket handler returned empty, and the evaluation recorded a page of legitimate-looking zeros: the expected row count, exit status 0, an aggregate with numbers in it, and no error on any record. This happened on a ten-image dry run over the validation split, which the protocol requires before any program is scored on test, so the one permitted read of the test split was not spent on it. What exposed it was neither the exit code nor the aggregate but the pixel counts, which read $\texttt{tp_px} = 0$ *and* $\texttt{fp_px} = 0$ — a program that predicts badly has false positives, and only a program that predicts nothing at all has neither. The evaluator now asserts that at least one record carries a prediction before it will report a score, and the run was repeated with the directory on the import path.

Cards. Splits are train / val / test. "Unit" is the independence unit by which the splits are drawn. "Floor" is the trivial predictor of Table 1 on the two ranking metrics, and "tool" the strongest zero-shot single tool on the same metrics.

- **AI-TOD** (Wang et al., 2021). 60 / 60 / 300. Source: xView (Lam et al., 2018) crops of 800×800, which the licence requires each user to obtain from the xView release. Unit: source image, at most two crops each, test source images disjoint from the other splits. Density tiers 31–50 and 53–100 in equal shares (`samples/ai_tod/build_ai_tod_tasklet.py`). Metrics: box mIoU, F1@0.5. Floor 0.0 / 0.0 (whole-image box, empty output). Tool: GroundingDINO at the train-tuned 0.10, 18.3 / 25.1.
- **LVIS** (Gupta et al., 2019). 60 / 40 / 200. Unit: category, with 15 / 5 / 10 disjoint categories, so test categories are unseen. The 52–100 tier caps the test size. Metrics as for AI-TOD. Floor 0.1 / 0.0 (whole-image box). Tool: SAHI, 30.2 / 39.2.
- **DroneVehicle** (Sun et al., 2022). 60 / 60 / 300. RGB drone frames of 840×712 including the dataset's 100-pixel white border; oriented boxes reduced to axis-aligned ones. Unit: the official train / validation / test frames, minus any frame within 8 pHash bits of one in an earlier split and any frame whose vehicles are annotated only as points. No licence of its own; the VisDrone project that publishes it states CC BY-NC-SA 3.0 for its benchmark data. Obtained from the Hugging Face mirror `McCheng/DroneVehicle`, whose `train.zip` carries 16 CRC-corrupt RGB images. Metrics as for AI-TOD. Floor 0.0 / 0.0. Tool: GroundingDINO at 0.05, 26.8 / 40.9.
- **PlantSeg** (Wei et al., 2026). 60 / 40 / 129. Unit: plant species; the test species are absent from training. A CLIP zero-shot check removed the images it scored above 0.55 as carrying a collage or an overlaid caption, 100 in all, which fixes the test size; the check is not exhaustive, and a few small overlays in a language the prompts did not name survive it. Licence: CC BY-NC 4.0 (current Zenodo release, v7, 2025). Metrics: per-image Dice, per-instance mIoU. Floor 18.5 / 2.8 (whole-image mask). Tools: AnomalyCLIP at 0.20 for Dice (48.8), at the train-tuned 0.40 for mIoU (14.2).
- **Magnetic Tile** (Huang et al., 2020). 60 / 40 / 286. All 386 usable defect images, stratified by defect class (blowhole, break, crack, fray, uneven). Licence: none named; the release welcomes research use provided its paper is cited. Metrics: DLI-B, the dataset's pooled-pixel IoU, and per-

image Dice. Floor 8.8 / 13.5. Tools: SAM 2 automatic masks for DLI-B (29.0), Grounded-SAM 2 at 0.40 for Dice (38.0).

- **Recod.ai/LUC** (Recod.ai and Loyola University Chicago, 2025). 60 / 60 / 300. Forged images only, stratified by forged area ($<$1%, 1–5%, $\geq$5%); pixel-level de-duplication found no duplicates. The ground truth marks both the source region and its pasted copy. Licence: CC BY-SA 4.0 under the competition rules; obtained from a public Kaggle re-upload. Metrics: DLI-B, Dice. Floor 3.7 / 9.5. Tool: TruFor, 33.2 / 30.9.
- **NDD20** (Trotter et al., 2020). 60 / 60 / 300 queries over 82 identities with three gallery images each. Unit: near-duplicate group (pHash $\leq$ 4, or same identity with DINOv2 cosine $\geq$ 0.95). Licence: non-commercial share-alike. Input: masked greyscale crops for queries and gallery alike (Sec. 7). Metrics: rank-1, mAP. Floor 1.3 / 6.5 (random). Tool: CLIP nearest neighbour, 7.3 / 15.8.
- **SeaTurtleID2022** (Adam et al., 2024). 60 / 60 / 300 queries over 438 identities with up to three gallery photographs each. Unit: the dataset's time-aware closed set: gallery from the official training days, queries from the validation and test days, at most one photograph per turtle per day in a partition. Licence forbids redistribution and commercial use. Input: masked greyscale whole-turtle crops. Metrics: rank-1, mAP. Floor 0.7 / 2.6. Tool: DINOv2 nearest neighbour, 4.3 / 7.3. Obtained from the Kaggle release `wildlifedatasets/seaturtleid2022` (v4).
- **DECIMER** (Brinkhaus et al., 2022). 60 / 60 / 300. Transcribe a hand-drawn structure to SMILES. Unit: drawing batch (22 / 6 / 11). Metrics: exact match and character error rate, both after RDKit canonicalisation; an unparsable prediction scores 0 and 1. Floor 0.0 / 93.4 (the most frequent training SMILES). Tool: the authors' hand-drawn model, 66.7 / 11.9.
- **SD7K** (Li et al., 2023), *excluded*. 138 test pages, shadow removal followed by OCR, scored by CER and WER. Tesseract after one background-division step whose blur is tuned on the training split reaches NPM 93.1 on CER, above the exclusion threshold, so the dataset is dropped (Sec. 3).

F Iterative Program Search (VTOS) Details

Configuration. Every search uses `claude-sonnet-4-6` through the Anthropic API, $n = 20$ iterations, $k = 3$ proposals per iteration, and the analyzer variant of the engine. One run departs from that and is truncated back to it: AI-TOD's first search was resumed and reached iteration 25, so its candidates from iterations 21 to 25 are discarded before selection and every task is then selected from the same budget. Discarding them changes nothing — the delivered program comes from iteration 13 and none of the fifteen ranks near the top — but leaving them in would have given one task a larger pool than the rest. The seed program is the family's standard pipeline at its default setting: a detector with tiling and box refinement for counting, Grounded-SAM 2 for Magnetic Tile, AnomalyCLIP at threshold 0.3 for PlantSeg, the TruFor arm for Recod.ai/LUC, a nearest-neighbour embedding for re-identification, and the domain model with a fallback reader for chemical OCR. Wherever the training split also carries a sweep of the same tool we compared the two. On three tasks the seed is the tool's default arm rather than its best: 34.37 Dice against 41.86 on Magnetic Tile, 34.33 DLI-B against 43.94 on Recod.ai/LUC, and 28.55 box mIoU against 29.86 on DroneVehicle. On PlantSeg the seed sits 0.2 Dice below the best swept arm on training (43.00 against 43.15) and trails it on mIoU (10.05 against 11.69). On the two re-identification tasks it is the strongest arm the roster allows: 6.67 rank-1 on NDD20 and 5.00 on SeaTurtleID2022, each matching the best masked-crop arm, the stronger full-frame arms being the scene shortcut that Section 7 discards. AI-TOD, LVIS and DECIMER carry no such sweep, so the comparison is open there. The searched programs still finish above every measured arm of their own roster on that split (44.92, 49.37 and 51.37 training Dice on PlantSeg, Magnetic Tile and Recod.ai/LUC), so no delivered result depends on the seed being the strongest start; seeding higher is the obvious improvement. Candidates are scored on the 60 training images, the five best plus the seed are re-scored on validation, and the validation ranking selects the delivered program.

What the delivered programs compose. Table F.1 lists, for each task, the distinct roster tools the delivered program calls and how it combines them. We count a program as composing when it calls two or more distinct tools and at least one call depends on another tool's output. Six of the nine qualify. On AI-TOD, DroneVehicle and LVIS the delivered program calls a single tool, the detector,

several times at different thresholds, tile grids and text queries, and merges the results with non-maximum suppression: those gains come from search over one tool's settings, not from composition. AI-TOD is the one task where this changed with the delivery: the program the pooled ranking replaced did call SAM to refine its boxes, and the one delivered does not reach past the detector, which is the task where the search ends level with that detector (Sec. 6). LVIS and DroneVehicle are also the two largest counting gains in Table 2, so we state this rather than leave it to be inferred.

Table F.1: The delivered program of each task: the distinct tools of its roster that it calls, how many calls it makes when a tool is invoked repeatedly, and how the results are combined. Tools outside the declared roster reach the program through the sandbox's `cv2` binding and are marked †.

Task	Distinct roster tools	Also used	Combination
AI-TOD	detect (×4)	—	single tool, merged
LVIS	detect (×4)	—	single tool, merged
DroneVehicle	detect (×5)	—	single tool, merged
PlantSeg	9 tools	—	conditional
Magnetic Tile	7 tools	—	conditional
Recod.ai/LUC	`trufor`, segment, merge, filter	SIFT, BFMatcher†	fusion
NDD20	CLIP embed, gallery, cosine, rank	—	re-ranking
SeaTurtleID2022	6 tools	ORB, RANSAC†	fusion
DECIMER	`decimer`, MolScribe, canonicalise, props	—	arbitration

How much of Recod.ai/LUC is SIFT alone? Recod.ai/LUC is the one delivered program that leans on a tool reached outside its roster, so we ran that stage on its own: SIFT keypoint self-matching with a ratio test and density clustering, with the TruFor gate, the cluster-count estimate and the mask refinement removed. On the same 300 test images it reaches 26.2 DLI-B and 23.1 Dice. That is below the domain tool it is paired with (33.2 and 30.9) and far below the delivered program (56.4 and 40.3), so neither component carries the result on its own.

Analyzer feedback. On the counting tasks the analyzer may call a small visual toolbox of three CV tools (density hotspots, image quality, overlapping clusters) and three VLM tools, with a cap of five VLM calls per iteration. It was enabled on the six searches of the three counting tasks, which logged 828 VLM calls: AI-TOD 172 and 137, LVIS 124 and 124, DroneVehicle 134 and 137. The cap binds: those calls arrive in 222 iterations, 3.7 per iteration against the cap of five. The segmentation, re-identification and OCR runners do not have this toolbox. Its log records call counts but not tokens, so those calls are excluded from the costs below.

Provenance of the delivered programs. Each task delivers one program, selected on validation from the pooled candidates of its $R = 2$ searches. AI-TOD delivers `sol_013_00` of its first search (2026-09-11), selected over the 117 candidates the two searches produced within the $n = 20$ budget; its second search (2026-09-12) reaches 13.80 mIoU on test. Scoring all 117 candidates rather than the two shortlists is what surfaces it: a search validates only the candidates that clear its training-score gate, and this one never reached validation, although its validation mIoU is the highest of the 117. Re-scoring reproduces each shortlisted candidate's original validation figures to the digits printed in its run log, so the pooled ranking and the runs' own rankings are on one scale. The pooled pick is not that run's own validation best, so its blind test was run after selection on the full 300 test images with per-image outputs kept (`scripts/test_pooled_pick.py`), and every image's ground-truth count is asserted to match the baseline's. LVIS delivers the first of two searches (39.30 against 40.23 mIoU on test); DroneVehicle `sol_018_00` of its first, likewise blind-tested after selection; PlantSeg `sol_007_01` of its second search, from the $R = 2$ pair run once AnomalyCLIP entered the roster and seeded the search. The first of that pair delivers the seed itself, and the pooled rank-sum prefers the second, which is also that run's own val-best, so its blind test stands without a further evaluation; Magnetic Tile delivers `sol_012_00` of a second search run on the fixed harness with the task's own unchanged prompts; its first search (2026-09-13) is pooled with it because none of the harness defects of Section 7(d) fired in that run, so its candidate scores are what the fixed harness would also have produced. Recod.ai/LUC delivers `sol_013_00` of its second search: pooling the two shortlists recomputes the Borda ranks and promotes it over that run's own pick, so it is the one candidate whose blind test had to be run after selection rather

than read from a run's own test log (`scripts/test_one.py`, the same evaluator and the same refusal to record a run with any fault). SeaTurtleID2022 delivers `sol_019_00` of its first search; DECIMER `sol_012_00` of its second. NDD20 has one search ($R = 1$).

What three delivered programs do. On DECIMER the program accepts a reading when the authors' model and MolScribe agree, retries on a binarised, thickened copy when they do not, and otherwise keeps the best of five readings by confidence and chemical plausibility. On PlantSeg it keeps the AnomalyCLIP regions at a threshold of 0.32, merges them, and subtracts CLIP-confirmed healthy tissue when the flagged area passes a quarter of the image. On AI-TOD it runs four passes of adaptive tiling, then a filter that drops boxes more than five times the median predicted area.

G Extended Results

Counting MAE. The AI-TOD and LVIS splits keep only images in a narrow density band, which makes mean absolute error degenerate: predicting the training median gives MAE 16.38 on AI-TOD and 15.78 on LVIS, against 27.72 and 31.79 for the strongest tool and 24.20 and 21.2 for the delivered programs. Lower is better, so the constant predictor wins on both. We report the metric here for completeness and rank counting by box mIoU and F1 in the main text (Sec. 7).

Table G.1: **Counting and detection.** The individual tool arms. Table 2 carries one *Tool* column per task and reports only the strongest of these arms in it. The arms are the roster tool at its published default, the same tool with one threshold tuned on the 60 adaptation images, and the tiling-and-refinement wrapper around it. The first row is not a method: it is the s_{trivial} of Eq. (1), the lower anchor every NPM in this paper is normalised against. Box mIoU and F1 give it nothing, which is why a raw score on these two metrics is already close to its normalised value; Table G.2 shows a metric where that does not hold. The methods of Table 2 are not repeated here and the ranking is there. Tables G.2, G.3 and G.4 carry the other three families.

Counting and detection	AI-TOD		DroneVehicle		LVIS	
Method	mIoU	F1	mIoU	F1	mIoU	F1
Trivial predictor	0.0	0.0	0.0	0.0	0.1	0.0
Single tool, default thr.	0.1	0.2	1.3	2.4	21.1	27.8
Single tool, train-tuned thr.	18.3	25.1	26.8	40.9	29.4	39.2
Tool-augmented	13.3	21.8	16.6	25.4	30.2	39.3

Table G.2: **Anomaly segmentation.** The individual tool arms; rows as in Table G.1, with the task's domain model added. The trivial predictor reaches 18.5 Dice on PlantSeg and 13.5 on Magnetic Tile before any image is looked at, so Dice has a floor that box mIoU and F1 do not; this is the calibration Eq. (1) removes.

Anomaly segmentation	PlantSeg		Recod.ai/LUC		Magnetic Tile	
Method	Dice	mIoU	DLI-B	Dice	DLI-B	Dice
Trivial predictor	18.5	2.8	3.7	9.5	8.8	13.5
Single tool, default thr.	36.8	11.3	3.1	9.8	11.8	37.4
Single tool, train-tuned thr.	36.1	11.8	3.8	10.0	15.2	38.0
Tool-augmented	19.5	3.5	14.1	21.4	29.0	27.8
Domain tool (AnomalyCLIP / TruFor / AnomalyCLIP)	48.8	14.2	33.2	30.9	12.0	18.7

Table 2 lists one column for the strongest single tool of each task, because the individual arms of a roster mostly repeat one another. Tables G.1–G.4 give them all, one per family. Two contrasts are worth reading off it. Tuning one threshold on the adaptation split moves a tool further than any of the orchestration methods move it: on AI-TOD the detector at its published default reaches 0.1 box mIoU and at the tuned threshold 18.3, and on DroneVehicle 1.3 against 26.8. This is what the admission rule is measured on, and it is why the rule tunes the tool rather than taking it at its default. In the other direction, the tiling-and-refinement wrapper that helps on LVIS (30.2 against

Table G.3: **Wildlife re-identification.** The individual tool arms; rows as in Table G.1. No domain model appears: a model whose training data overlap the evaluation split is not eligible as a single-tool baseline, which is why MegaDescriptor is absent on NDD20 (Sec. 3).

Wildlife re-identification	NDD20		SeaTurtleID2022	
Method	rank-1	mAP	rank-1	mAP
Trivial predictor	1.3	6.5	0.7	2.6
Single tool (CLIP / DINOv2)	7.3	15.8	4.3	7.3

Table G.4: **Chemical structure recognition.** The individual tool arms; rows as in Table G.1. CER is an error rate, so the trivial predictor's 93.4 is its floor read from the other direction.

Chemical structure recognition	DECIMER	
Method	exact match	CER
Trivial predictor	0.0	93.4
Single tool (MolScribe)	6.3	72.5
Domain tool (DECIMER-HD)	66.7	11.9

29.4 mIoU) hurts on DroneVehicle (16.6 against 26.8) and on PlantSeg (3.5 against 11.8 mIoU), so "add a standard wrapper" is not a baseline that can be assumed to help.

Table G.5: Full counting metrics on the three counting test splits ($\times 100$ for the rates; MAE and Bias are in objects). Bias is the mean of predicted minus true count, so a negative value is systematic under-counting. Same runs as Table 2, whose mIoU and F1 columns these reproduce. The trivial predictors are omitted: the trivial floor of Table 1 is the best of several predictors chosen per metric — the constant wins MAE, a whole-image box wins mIoU and an empty output wins F1 — so no single trivial predictor has a coherent row here.

AI-TOD ($n = 300$)	MAE↓	Bias	F1@0.5↑	mIoU^{box} ↑	P@0.5↑	R@0.5↑
Single tool, default thr.	56.2	-56.2	0.2	0.1	3.5	0.1
Single tool, train-tuned thr.	27.7	+10.4	25.1	18.3	22.8	31.0
Tool-augmented	77.7	+74.2	21.8	13.3	15.9	39.3
VLM only (Sonnet 4.6)	35.3	-33.6	0.9	2.3	1.5	0.7
VLM only (GPT-5.4)	25.1	-23.8	1.1	2.9	1.7	0.9
CodeAct	31.5	-29.2	10.2	7.3	13.7	9.0
VTOS ($R = 2$)	24.2	-19.3	24.8	18.3	29.5	23.1
DroneVehicle ($n = 300$)	MAE↓	Bias	F1@0.5↑	mIoU^{box} ↑	P@0.5↑	R@0.5↑
Single tool, default thr.	50.5	-50.5	2.4	1.3	15.3	1.5
Single tool, train-tuned thr.	45.6	+14.8	40.9	26.8	37.0	56.9
Tool-augmented	81.0	+69.7	25.4	16.6	18.4	51.4
VLM only (Sonnet 4.6)	16.7	-11.0	5.6	13.7	6.3	5.3
VLM only (GPT-5.4)	13.8	-12.7	4.7	11.7	5.4	4.4
CodeAct	26.0	-17.3	32.1	24.6	35.3	32.8
VTOS ($R = 2$)	22.2	-15.4	51.2	37.3	59.2	49.7
LVIS ($n = 200$)	MAE↓	Bias	F1@0.5↑	mIoU^{box} ↑	P@0.5↑	R@0.5↑
Single tool, default thr.	43.2	-41.6	27.8	21.1	50.9	24.6
Single tool, train-tuned thr.	36.9	-23.9	39.2	29.4	54.4	38.9
Tool-augmented	31.8	-0.8	39.3	30.2	39.6	44.5
VLM only (Sonnet 4.6)	39.6	-38.6	1.3	4.7	2.3	1.1
VLM only (GPT-5.4)	30.6	-28.9	6.4	10.4	10.6	5.2
CodeAct	36.4	-27.8	40.1	30.0	58.2	37.9
VTOS ($R = 2$)	21.2	-1.0	48.6	39.3	49.8	50.8

Table G.5 gives every counting method on all six metrics, and shows the same degeneracy from the other side. On DroneVehicle the no-tool VLMs have the lowest MAE of any method, 13.8 for GPT-5.4 and 16.7 for Sonnet 4.6 against 22.2 for the delivered program, while finding almost none of the vehicles: their F1@0.5 is 4.7 and 5.6 against 51.2. They score well on MAE because they under-count by about twelve objects on images that hold few enough for that error to stay small, not because their counts are right.

A counting method's predicted count is the number of boxes it predicts, not a number it states separately. The no-tool VLMs are asked for boxes and a count in the same reply, and the count they give is discarded. This is deliberate. A model that answers "about forty pigeons" while drawing five boxes has not located forty pigeons; it has guessed a number. Scoring the stated number would let a method win on MAE while its boxes are worthless, which is the same degeneracy Section 7 rejects the metric for. Tying the count to the boxes keeps MAE consistent with the two ranking metrics, at the cost of reading the no-tool rows as a statement about localisation and not about arithmetic.

Bias separates the failure modes that the ranking metrics blur together. Both no-tool VLMs and CodeAct under-count on all three tasks, by 11 to 39 objects. The tools fail in the opposite direction whenever the wrapper fires: SAHI slices each image and re-detects, which adds 74.2 objects per image on AI-TOD and 69.7 on DroneVehicle. The delivered program is near zero on LVIS alone (-1.0), where the tool-augmented arm matches it (-0.8) while being 74.2 out on AI-TOD. On the other two it under-counts with the rest, at -19.3 on AI-TOD and -15.4 on DroneVehicle: selecting on the two reported metrics (Section 7) buys localisation, not arithmetic.

Single-run test scores. Each task runs $R = 2$ independent searches and delivers the one selected on validation; the other search's blind-test score is given here so that the spread is visible. AI-TOD 16.99 and 13.80 mIoU; LVIS 39.30 and 40.23; DroneVehicle 37.7 and 33.4; PlantSeg 48.37 and 45.12 Dice; Magnetic Tile 41.47 and 45.95 DLI-B; Recod.ai/LUC 27.96 and 43.60 DLI-B; SeaTurtleID2022 3.33 and 2.67 rank-1; DECIMER 70.7 and 72.7 exact match. NDD20 has one search. Of these figures, the delivered program is LVIS 39.30, PlantSeg 45.12 Dice, Magnetic Tile 41.47 DLI-B, SeaTurtleID2022 3.33 rank-1 and DECIMER 70.7 exact match; on PlantSeg the lower-Dice search was selected because its programs lead on mIoU, which the Borda rank-sum rewards. On three tasks the delivery is neither run's own pick, because the pooled ranking reaches candidates a single run had set aside: Recod.ai/LUC (56.40 DLI-B on test), AI-TOD (18.28 mIoU) and DroneVehicle (37.32 mIoU). On AI-TOD the pooled program is above both runs' own picks on test, 18.28 against 16.99 and 13.80.

Search cost. Token logging was added partway through the runs, so the search cost is reported where it was measured and marked where it was not. On the two tasks whose whole $R = 2$ pair is logged, a pair costs 1.41M prompt and 0.34M completion tokens over 368 calls on LVIS, and 1.60M / 0.32M over 391 calls on DroneVehicle. Only the second AI-TOD search is logged (0.84M / 0.17M over 199 calls). The PlantSeg and Magnetic Tile logs belong to searches that were later superseded, and the Recod.ai/LUC, NDD20, SeaTurtleID2022 and DECIMER searches predate the logger. For comparison, an agent that writes one program per image and runs no search costs 0.13M prompt tokens on LVIS, 0.19M on DroneVehicle and 0.20M on AI-TOD (CodeAct), and 0.06M–0.17M on the segmentation tasks (VisProg): a search pair is 10.8 times the prompt tokens of the agent on LVIS and 8.2 times on DroneVehicle, the two tasks where both halves of the pair and the agent run are logged. All calls are `claude-sonnet-4-6`.

Sensitivity of the admission threshold. The rule excludes a triplet at NPM ≥ 90. Table G.6 reports what the admitted set would be at other thresholds. The rule's verdict is unchanged for any threshold between 87.2, the highest admitted value (DECIMER's character error), and 93.1, the lowest excluded one (SD7K's): the operating point of 90 sits inside that band with nothing else near it. Outside it the set moves in both directions. At 95 the rule excludes nothing and SD7K would enter the benchmark. At 85 DECIMER's character-error triplet goes as well, and at 80 SD7K's word-error triplet goes with it, which is the triplet the source-level argument given with the dropped task below removes anyway. No threshold reproduces the admitted eighteen on its own: at 85 the count is eighteen but the set is not, since it has dropped DECIMER's character error and kept SD7K's word error.

Table G.6: Admitted triplets as a function of the exclusion threshold. Values are the NPM of the strongest zero-shot single tool (Table 1); SD7K is the one probed source the rule rejects. The column counts what the threshold alone admits, which is why the operating point of 90 reads nineteen against the benchmark's eighteen: SD7K's word-error triplet clears every threshold above 82.4 and is removed by the source-level argument given with the dropped task below, not by the rule.

Threshold	Triplets excluded	Admitted
95	none	20
90	SD7K CER (93.1)	19
85	SD7K CER (93.1), DECIMER CER (87.2)	18
80	SD7K CER (93.1), DECIMER CER (87.2), SD7K WER (82.4)	17

The dropped task. SD7K is the one probed source the screen rejected. Tesseract (Smith, 2007) after dividing the image by a blurred copy of itself reaches CER 6.87 against an empty-output floor of 100, an NPM of 93.1, and WER 17.61 against the same floor, an NPM of 82.4. The CER triplet crosses the admission threshold and the WER triplet does not; we drop the source rather than keep the WER triplet alone, because the saturation belongs to the pipeline and not to the metric that reports it.

Pixel precision and recall. Table G.7 splits the segmentation scores of Table 2 into their two halves on the same test images. Against the generic mask tools of Section 6, the delivered programs trade recall for precision on all three tasks, and the two AnomalyCLIP arms show the same trade inside one tool.

Table G.7: Pixel precision and recall ($\times 100$) on the segmentation test splits, for the delivered program, the single tools of Table 1, and the ground-truth-box oracle.

Method	precision	recall
PlantSeg (129 images)		
SAM 2 automatic masks	13.5	65.0
Grounded-SAM 2, default	34.1	74.9
Grounded-SAM 2, train-tuned	30.6	80.4
AnomalyCLIP at 0.20	47.1	67.2
AnomalyCLIP at 0.40	58.7	49.5
VTOS (delivered)	56.6	53.6
Ground-truth boxes into SAM 2	83.5	85.4
Magnetic Tile (286 images)		
SAM 2 automatic masks	24.1	75.5
Grounded-SAM 2, default	33.4	84.0
Grounded-SAM 2, train-tuned	34.4	68.1
VTOS (delivered)	47.2	52.1
Ground-truth boxes into SAM 2	87.3	83.2
Recod.ai/LUC (300 images)		
SAM 2 automatic masks	17.1	59.3
Grounded-SAM 2, default	6.7	89.3
TruFor	45.5	31.5
VisProg	46.6	29.9
VTOS (delivered)	53.6	39.0
Ground-truth boxes into SAM 2	94.6	85.9

One generated program, without search. Given the same LLM and tools but no search, CodeAct and VisProg each write one program per image without seeing it. CodeAct falls far below the train-tuned detector on AI-TOD (mIoU 7.3 against 18.3, $p < 0.001$) and DroneVehicle ($p = 0.005$), and VisProg is not significantly different from the strongest tool on Magnetic Tile and falls below TruFor on Recod.ai/LUC although it can call it. The other three tasks read the same way: 69.7 exact match on DECIMER, and nothing at all on re-identification, where it ends below the search

on SeaTurtleID2022. These writers see neither the training images nor any execution feedback, so the comparison does not separate the value of search from that of adapting on labelled data.

A coding agent as the searcher. The general-purpose coding agent of Table 2 is Claude Code, Anthropic's coding agent, driving the same `claude-sonnet-4-6` that the search and the one-shot writers use; what differs is the harness around the model, not the model. VTOS is a program search built for this setting. The obvious question is whether a purpose-built search is needed at all, or whether pointing a general coding agent at the data would do. We ran that comparison on LVIS with every other variable held fixed: the same LLM (`claude-sonnet-4-6`), the same 60 labelled training images, the same protocol (adapt on training data, deliver one program, read the blind test split once), and the same detector checkpoint — the agent was given no tool list and independently chose `IDEA-Research/grounding-dino-tiny`, which is the detector VTOS's toolbox calls. Only the searcher differs. That last point holds for LVIS alone: every later prompt lists the checkpoints already in the local cache, and those lists name the same detector. The AI-TOD and DroneVehicle agents were therefore told the checkpoint was available, and the three segmentation agents were given the task's declared tool roster as callable functions, so that the comparison is not decided by which system had a tool ready to hand. The prompts of all eleven agent sessions are released verbatim with the code, together with the sandbox builder and the launch and evaluation scripts; the differences between them, including this one, are listed there rather than left for a reader to infer.

The agent developed in a directory holding the 60 training images and their labels and nothing else: no validation split, no test images, no access to this repository, and the `meta_info` field stripped from every record because it carries the identifier of the source photograph. It delivered one program, which we then scored through the same evaluator as every other row in Table 2.

On LVIS it reaches box mIoU 35.6 and F1@0.5 48.3, against 39.3 and 48.6 for VTOS: three and a half points behind on mIoU and level on F1, and ahead of every single tool and of CodeAct on both. The same task run on a newer model (`claude-sonnet-5`) gives 36.3 and 48.9, so the result is not an artefact of the model version. The session that produced it ran one hour and forty-two minutes by its transcript timestamps; the transcript records no token cost, so none is reported.

Run again on AI-TOD, where the images are aerial and the targets are a few pixels across, it reaches 23.3 and 34.6 against VTOS's 18.3 and 24.8, ahead on both ranking metrics, and its count is almost unbiased (-0.2 objects per image against the tuned detector's +10.4). On DroneVehicle it reaches 26.3 and 34.6 against 37.3 and 51.2, behind on both and by a wider margin than it led on AI-TOD. Across the three counting tasks the purpose-built search is therefore ahead on two and behind on one, and the size of the gap moves with the task rather than with the method.

The three segmentation tasks were run the same way with one deliberate change: each agent was given its task's declared tool roster as callable functions, so that the comparison is not decided by VTOS having a domain model wired up and the agent not. What the agents then did with that roster is the result. On PlantSeg the delivered program is seventeen lines long and calls `anomaly_segment` at its default threshold and nothing else; it scores Dice 48.2 and mIoU 13.0, against 45.1 and 17.6 for the search. On Recod.ai/LUC it calls `trufor` at the threshold of that model's own paper and adds an area filter, and lands at DLI-B 33.6 and Dice 30.7, within half a point of the TruFor row of Table 2 and far below the search's 56.4 and 40.3. On both tasks, given the roster, a forty-minute session converges on the strongest single tool of that roster and stops there. Magnetic Tile is the exception that makes the point: its roster carries no anomaly model, because the search that delivered its program ran before one entered the roster. The agent went outside the roster, imported the anomaly model's backend directly, and wrote a 185-line program that branches on the defect class. It wins Dice by 10.4 and loses DLI-B by 15.2, sitting at the opposite end of the trade Section 7(b) records on this task: it returns one confident region per image, which scores well per image and poorly once pixels are pooled, where raising a detector's threshold does the reverse. That cell is therefore not a controlled comparison, and we mark it as such rather than drop it.

The remaining three tasks were run the same way. On DECIMER the agent was given the task's roster and delivered a forty-nine-line program that runs the domain model on four preprocessing variants and keeps the RDKit-parseable reading with the highest confidence, falling back to the second recogniser; it reaches exact match 72.7 against the search's 70.7 and ties it on CER at one decimal. On the two re-identification tasks it was given the gallery and the roster, and fitted a PCA

whitening on the gallery embeddings before ranking against per-identity prototypes: rank-1 16.3 against 7.3 on NDD20 and 10.3 against 4.3 on SeaTurtleID2022, more than double the strongest single tool on both, and the largest margin any system in Table 2 holds over that tool.

Over all nine tasks the agent is ahead of the search on both ranking metrics on three (AI-TOD, NDD20, SeaTurtleID2022), behind on both on three (LVIS, DroneVehicle, Recod.ai/LUC), ahead on one metric and behind on the other on two (PlantSeg, Magnetic Tile), and ahead on one while level on the other on one (DECIMER). We read the pattern as a statement about the roster rather than about the two searchers: where a strong single tool is already bound and named, an agent finds it quickly and adds little, and the tasks where composition pays are the ones where no single bound tool is close. The two re-identification tasks are the clearest case of the second kind. Their rosters bind generic embeddings that were never trained to separate individuals, the search left them where it found them, and what moved them was a decorrelation of those embeddings that neither the roster nor the seed contained. We read this as the benchmark working rather than as a defect: the admission rule keeps a task whether or not any evaluated method wins it, and a task on which a forty-minute agent session beats a program search is a task with headroom left.

Adaptation that does not transfer. The counting splits hold disjoint categories, so the ten LVIS test categories are unseen during adaptation. Part of what the agent bought with its training budget was a discovery about prompt wording: on one image the detector returns 24 instances for "grape berry" against 10 for "grapes" (ground truth 50), and on another 18 for "ear of corn" against 3 for "edible corn" (ground truth 49). It hard-coded both aliases.

Both are training categories. The test split contains apple, banana, book, carrot, doughnut, pastry, person, sheep, tomato and toy, and we checked every one of the 200 test records: no override key appears, so the table is dead code at test time. The optimisation was real, was measurable on the adaptation split, and is worth exactly nothing on the blind one. Had the splits been drawn by image rather than by category, it would have been reported as a gain. This is the case the disjoint-category rule of Section 4 exists to catch, and it was produced by a general-purpose coding agent on the first task we tried it on.

Target size. Splitting each segmentation test split into tertiles by the ground-truth area of the target gives Table G.8. Small targets are the hardest case on PlantSeg and Recod.ai/LUC, while on Magnetic Tile the middle tertile is the weakest; on the latter two the delivered program improves steeply with target size. Magnetic Tile gains 25.6 Dice points from its smallest to its largest tertile, where the smallest is a defect covering 0.03–0.26% of the image. PlantSeg is the exception: it peaks on the middle tertile and falls back on the largest, where a diseased region covering more than 13% of the leaf is confused with the leaf itself. The tertiles are per task, so the boundaries differ; the median target covers 8.8% of the image on PlantSeg, 2.5% on Recod.ai/LUC and 0.4% on Magnetic Tile.

Table G.8: Mean Dice ($\times 100$) of the delivered program by ground-truth area tertile. Tertiles hold equal numbers of images and are cut on the mask area recorded for each test image in the per-image outputs. Boundaries, as a percentage of image area: PlantSeg 0.53–5.56 / 5.65–13.00 / 13.66–29.46; Magnetic Tile 0.03–0.26 / 0.27–4.46 / 4.69–48.50; Recod.ai/LUC 0.09–1.03 / 1.03–5.97 / 5.99–46.85.

Task	small	medium	large
PlantSeg (129 images)	41.0	**53.3**	41.1
Magnetic Tile (286)	37.9	35.8	**63.5**
Recod.ai/LUC (300)	32.3	38.1	**50.4**

Object density. Each counting split is built half from scenes the tasklet calls *normal* and half from *extreme* ones, so the delivered programs can be read against scene density directly (Table G.9). All three lose ground as the scene fills, and by different amounts: DroneVehicle drops 12.8 box mIoU and 14.0 F1 between the two halves, AI-TOD 4.8 and 4.7, LVIS 3.0 and 2.1. The counting gains of Table 2 are therefore carried mostly by the less crowded half of each split.

Table G.9: Delivered program on the counting test splits, split by the density tier of the scene ($\times 100$). Object-count ranges are given per tier.

	normal		extreme	
Task	mIoU	F1	mIoU	F1
AI-TOD (31–50 / 53–100 objects)	20.7	27.1	15.9	22.4
LVIS (23–50 / 52–100)	40.8	49.7	37.8	47.6
DroneVehicle (31–50 / 51–100)	43.7	58.2	30.9	44.2

H Related Work: Further Notes

Section 2 covers the related work; this appendix adds what it leaves out. Benchmarks are known to lose their power to separate systems as they age: Akhtar et al. (2026) define saturation over sixty language benchmarks and find that expert curation, not private test data, predicts resilience.

I What Each Column of Table 2 Is

Section 5 defines the five columns; two points are added here. *Tool*: Table 1 names the tool in each cell and Tables G.1–G.4 give every arm. *Standalone VLM*: marked not applicable (‡) on the two re-identification tasks because the answer there is a ranking of the whole gallery, so a VLM would have to be shown all 246 (NDD20) or 1292 (SeaTurtleID2022) gallery crops for every one of the 300 queries; and a model that ranked a gallery from its crops would be performing the retrieval that the *Tool* column already measures with CLIP and DINOv2 nearest neighbours, so the row would repeat a measurement rather than add one.

J The Five Evaluation Failures: Further Details

Section 7 gives each failure with its measurements; below are the details it leaves out.

(a) Scene shortcuts. NDD20 photographs come in sessions, and near-duplicate grouping does not remove the session. The tools were matching sessions, not dolphins. The masked greyscale crops are used for queries and gallery alike (Table C.1, Appendix C).

(b) Degenerate metrics. *Counting MAE.* The AI-TOD and LVIS splits keep only images in a narrow density band (Sec. 4), and in a band this narrow the training median beats every tool and every searched program (Appendix G). The searches themselves hill-climb on F1@0.5, but their final validation ranking used MAE alongside mIoU, so for each counting task we re-scored every candidate both searches produced on that task's validation split and re-ranked them by the two reported metrics; the delivered program is the pooled winner under that ranking. The searches were not re-run: only the selection step was recomputed, which changed the delivered program on AI-TOD and DroneVehicle and left LVIS unchanged.

(c) Selection noise. On PlantSeg pooling trades one ranking metric for the other: the first search delivers its own seed (48.4 Dice, 15.5 mIoU), the second a program that filters the anomaly regions with CLIP (45.1, 17.6), and the pooled Borda rank-sum prefers the second for its mIoU, so the delivered Dice falls 3.3 points below what the first search alone would have given. Pooling can also reorder what a single run selects, on Recod.ai/LUC, Magnetic Tile and DECIMER alike (Appendix D).

(d) The search harness scores its own failures. Three defects made candidates look bad for reasons that belonged to the harness we ran: an expiring per-candidate time budget, a sandbox missing two Python builtins, and a contract violation read as an empty prediction (Appendix F). All three were invisible in the aggregate scores. Task descriptions, finally, belong to one dataset each, and we re-ran a task only where its own runs came from a defective setting (Appendix F).

(e) Adaptation that does not transfer. Section 7(e) gives the case; the paragraph *Adaptation that does not transfer* of Appendix G gives the second alias, the ten test categories and the check of every test record.