VISUALCODING: EXECUTABLE SKILLS FOR RELIABLE AGENTIC VISUAL REASONING

Anonymous authors
Paper under double-blind review

ABSTRACT

Learning reusable skills can improve visual reasoning agents, yet applying a skill still requires the agent to implement or invoke a procedure and produce the final answer. Errors and hallucinations at these steps can prevent acquired experience from translating into reliable single-run performance. We propose VISUALCODING, which combines execution-guided acquisition of reusable visual programs with direct adoption of their answers on applicable inputs. *Execution-Guided Program Acquisition* composes visual primitives and additional code into family-level solvers, then iteratively tests and revises them on labeled training instances. *Scope-Aware Direct Execution* runs a retained solver and returns its valid in-scope answer. We evaluate VISUALCODING on four models across four benchmarks spanning 69 task families. VISUALCODING exceeds the strongest evaluated baseline in mean accuracy by 4.45 percentage points with Gemini-3.1-Pro and by 14.78 points when reusing the frozen Pro-acquired artifacts with Flash-Lite. Ablations support execution-guided acquisition and show that direct execution outperforms textual or callable reuse of the same programs. Code is available at https://anonymous.4open.science/r/VisualCoding.

1 INTRODUCTION

Agent skills allow reasoning agents to retain and reuse problem-solving experience across tasks. Existing methods distill or refine instructions, code templates, and scripts from execution trajectories without updating model parameters (Jiang et al., 2026; Yang et al., 2026b; Ni et al., 2026). These skills create reusable textual prompts that guide subsequent tool use and reasoning, reducing the need to develop each solution from scratch. However, their effectiveness depends on how agents implement the retained procedures and use their results to answer new questions.

For visual reasoning, however, trajectory-derived textual skills still leave critical execution decisions and final-answer generation to the agent. Even with visual tools and coding interfaces, applying a

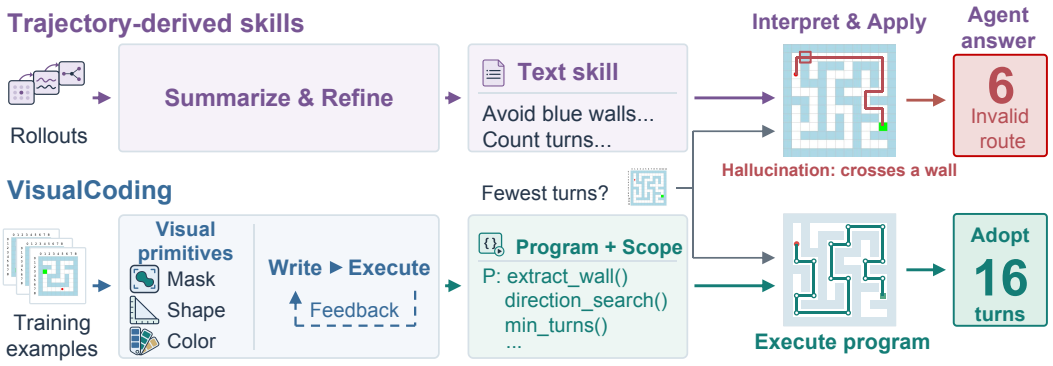

Figure 1: **From text skills to executable experience.** Agents can still make errors when applying text skills distilled from rollouts (top). VISUALCODING refines reusable programs through execution feedback and directly adopts their valid answers for more reliable visual reasoning (bottom).

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.

skill requires translating its guidance into measurements and computations that respect the current image's spatial constraints (Zhao et al., 2025; Hong et al., 2025; Zhao et al., 2026). Figure 1 illustrates this gap: the injected skill explicitly requires avoiding blue wall cells, yet the agent describes a wall-crossing route and answers 6. The skill states the correct constraint, but the agent fails to enforce it in the proposed path. Even when a computation yields a correct result, final-answer generation creates another chance for hallucination or misinterpretation to produce an incorrect response. Text skills therefore leave room for errors both in executing a visual strategy and in reporting its result.

These failures are not inevitable. Many visual reasoning tasks can be expressed as compositions of visual primitives, including spatial transformations, image enhancement, and element matching. Within a task family, exploring these compositions can turn abstract textual guidance into executable solution procedures. Labeled training images provide a basis for this exploration: candidate programs can be executed across instances and revised using correctness feedback, execution errors, and intermediate diagnostics. Such iterative exploration can uncover visual computations absent from the initial attempts and preserve them as reusable implementations, extending experience acquisition beyond summarizing existing trajectories.

Building on this insight, we propose VISUALCODING, a framework that acquires deployable solvers for known visual task families and directly adopts their valid in-scope answers (Figure 1). **Execution-Guided Program Acquisition** combines visual primitives with additional Python code to construct a shared solver for each task family. It iteratively tests and revises candidate solvers on training instances, checking changes to visual measurements, matching criteria, and search procedures for both gains and regressions. Training-side scope fitting and selection retain an eligible solver and its empirical applicability rule, both of which are frozen before deployment. **Scope-Aware Direct Execution** runs the frozen solver and directly returns its answer when validity and scope checks pass. This direct path preserves the computed result without requiring the deployment agent to reconstruct the computation, decide whether to invoke the solver, or generate the final answer. Rejected outputs fall back to the base agent. For task families without a deployable program, we distill text skills from available execution trajectories for use on subsequent test instances.

Our contributions are:

- We introduce VISUALCODING, which couples execution-guided acquisition of deployable visual solvers with direct answer adoption on valid in-scope inputs. It refines executable implementations across training instances and reuses the retained computation without requiring the deployment agent to reconstruct it or decide whether to adopt its answer.
- Across four benchmarks, VISUALCODING achieves the best mean accuracy among evaluated adaptations with Gemini-3.1-Pro and in transfer to Flash-Lite. Frozen Flash-Lite routes also improve DeepSeek and GPT deployment.
- Ablations show higher mean accuracy from VISUALCODING than from textual or callable reuse of the same programs. On some task families, single-run accuracy also exceeds the base agent's pass@4, suggesting that iterative program refinement can learn solutions missed by repeated sampling

2 RELATED WORK

2.1 EXPERIENCE AND SKILL LEARNING

Agent Workflow Memory extracts reusable workflows, Dynamic Cheatsheet accumulates problem-solving strategies, and Voyager retains experience from exploration in Minecraft (Wang et al., 2024; Suzgun et al., 2025; Wang et al., 2023). ReasoningBank distills successful and failed trajectories into reasoning memory, ACE evolves contextual playbooks, and MetaFlowLLM organizes experience into hierarchical workflows (Ouyang et al., 2026; Zhang et al., 2026b; Fan et al., 2025).

Agent Skills organize procedural knowledge for reuse, including multimodal guidance in XSkill and MMSkills (Jiang et al., 2026; Zhang et al., 2026a), trajectory distillation in Trace2Skill (Ni et al., 2026), and skill or context optimization in SkillOpt and Meta Context Engineering (Yang et al., 2026b; Ye et al., 2026). SkillGen refines skills through paired evaluations with and without

the skill, while SkillsBench evaluates their utility across tasks (Ma et al., 2026; Li et al., 2026b). Skill-Pro (ProcMEM) learns procedural memory without updating model weights (Mi et al., 2026). Executable reuse includes polymorphic skills in PolySkill, logic-grounded programs in NSI, and verified tool creation or compilation in RefTool and EvoC2F (Yu et al., 2026; Shao et al., 2026; Liu et al., 2026; Wei et al., 2026). Retaining reusable experience or text skills does not ensure a correct final answer: model-mediated implementation, invocation, or answer generation can still introduce execution errors or hallucinations. We instead encode the acquired computation in an executable solver and directly adopts its valid in-scope output.

2.2 Agentic Visual Reasoning

VisProg and ViperGPT compose visual modules into programs, while PyVision constructs tools dynamically during inference (Gupta & Kembhavi, 2022; Surís et al., 2023; Zhao et al., 2025). PIPS introduces program-based computation alongside chain-of-thought reasoning (Stein et al., 2025). DeepEyes, Pixel Reasoner, PyVision-RL, VLM-R^3, and AdaReasoner use reinforcement learning to improve visual tool use and orchestration (Zheng et al., 2025; Wang et al., 2025; Zhao et al., 2026; Jiang et al., 2025; Song et al., 2026). TVP abstracts solved programs into validated tools for subsequent program generation, while Code-with-Image refines textual skills through executable reflection (Wu et al., 2025; Yang et al., 2026a). Perception Programs convert tool outputs into structured cues for further model reasoning (Janjua et al., 2026). Existing coding-based approaches primarily construct tools or synthesize solutions for individual inputs. VisualCoding refines a task-family solver through execution feedback across labeled training instances and freezes it for reuse. Its valid in-scope outputs directly supply the final answer.

3 Method

VisualCoding acquires reusable visual programs from training samples and deploys them on applicable inputs. As shown in Figure 2, in *program acquisition*, a writer composes visual primitives and Python code, then revises candidates using execution feedback. Training-side scope fitting and selection retain a program with an empirical scope rule. In *scoped execution*, the frozen program runs on a new image and question, and its answer is returned when validity and scope checks pass.

3.1 Overview and Executable Skills

For a VQA, let $x = (I, q)$ contain an image and the full question text, and let y be the reference answer. For each known task family f, acquisition uses labeled examples $\mathcal{D}_f^{\mathrm{tr}}$ and outcome from a base agent π_0. We denote its empirical single-run accuracy on training example i by b_i, estimated from independent rollouts . Labels supervise acquisition and scoring but are excluded from program inputs to prevent label-based shortcuts.

An executable skill consists of a solver P_f, an empirical scope rule g_f, and an answer adapter a_f. Its interface is

$$P_f(x; \mathcal{L}) = (v, z), \qquad \tilde{y} = a_f(v; x), \tag{1}$$

where $\mathcal{L}$ is a fixed visual primitive library, v is the computed value, and z contains diagnostics. The adapter converts v to the task's answer convention. The scope rule uses program outputs and label-free task information to decide whether to adopt $\tilde{y}$. We retain at most one deployable program per family, which may contain internal routes.

The writer combines visual primitives for masks, components, color, geometry, and periodicity with ordinary Python control flow and numerical libraries. Perception interfaces support detection and model queries for discrete perceptual inputs. Acquisition learns the composition, additional code, and scope while keeping the primitives fixed.

Diagnostics support both revision and applicability checking. For example, periodicity estimation returns a period and its strength. During acquisition, diagnostics, substep checks, and intermediate images help locate failed computations. The scope fitter uses selected diagnostics to identify inputs on which to adopt the program's answer.

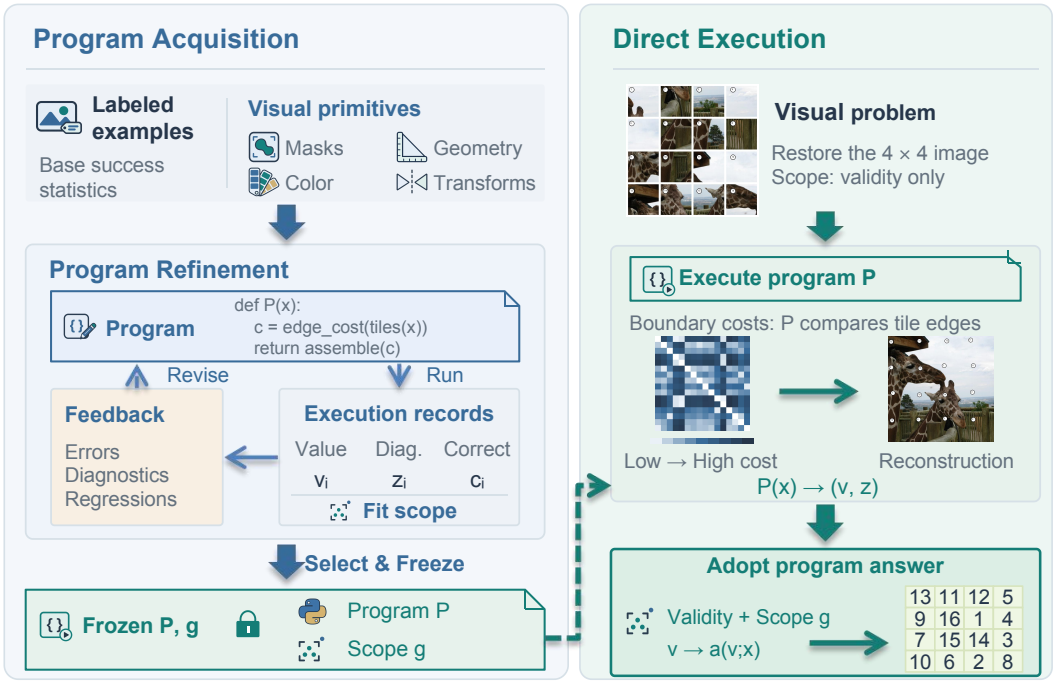

Figure 2: **VISUALCODING overview. Program acquisition:** training examples, baseline statistics, and visual primitives support program search. Execution feedback guides revision, and scope fitting and selection retain a frozen program P and scope g. **Direct execution:** valid in-scope outputs directly supply the answer through an adapter.

3.2 EXECUTION-GUIDED PROGRAM ACQUISITION

Generate, execute, and revise. Inspired by counterexample-guided inductive synthesis (Solar-Lezama, 2008), we develop a computation shared by a task family through repeated execution and revision. It starts from a profile of labeled training examples, baseline statistics, and bounded image probes, then proposes a program, an answer convention, and diagnostic features. The harness executes each candidate and returns per-example correctness, execution errors, diagnostics, and intermediate visualizations. Subsequent proposals receive the current best candidate, this feedback, and a ledger of previous attempts. Comparisons with the best candidate identify newly solved examples and regressions, guiding repairs and extensions (Algorithm 1). All feedback only comes from training instances.

Fit an empirical scope rule. For a fixed candidate, let $c_i = \mathbb{I}[\tilde{y}_i = y_i]$ denote correctness under the task evaluator and let $s_i(g)$ indicate a valid answer accepted by rule g. Within each program-declared route, or the full family without routing, we search a compact rule class $\mathcal{G}$ for maximum training coverage subject to minimum precision τ and support $n_{\min}$:

$$\max_{g \in \mathcal{G}} \sum_i s_i(g) \quad \text{s.t.} \quad \frac{\sum_i s_i(g) c_i}{\sum_i s_i(g)} \geq \tau, \qquad \sum_i s_i(g) \geq n_{\min}. \tag{2}$$

Rules combine numeric, Boolean, or categorical diagnostic comparisons. Required features that are missing, unsupported routes, invalid answers, and execution errors trigger fallback. Appendix A.2 specifies the rule class and threshold search.

After synthesizing the program, we check whether its scope rule depends too heavily on individual training tasks. We evaluate the scope rule by leaving out one task at a time, refitting the rule on the stored program outputs from the remaining tasks, and measuring precision and coverage on the left-out task. To assess rule stability, we measure how often the refitted rule makes the same scope decisions as the rule fitted on all training outputs. The program remains fixed throughout these checks.

Select and freeze. The high-precision tier requires training and leave-one-out precision, sufficient support, and rule stability. Otherwise, a gain tier applies its own precision, support, and stability checks and requires a positive bootstrap lower bound on paired improvement over the base-agent reference b_i. Within a run, high-precision candidates rank ahead of gain-tier candidates. Deployment additionally requires at least 50% training coverage. We freeze the best eligible program and scope under the training-side selection rule. Acquisition budgets, stopping criteria, and ranking details appear in Appendix A.2 and A.4. If no program is eligible, a separate summarizer builds a text skill from baseline trajectories and labeled outcomes. Skill construction and formatting checks are specified in Appendix A.5.

Algorithm 1 Execution-guided acquisition within one family run

Require: Labeled training set $\mathcal{D}^{\mathrm{tr}}$, base statistics b, writer W, library $\mathcal{L}$

$F \leftarrow \mathrm{ProfileAndProbe}(\mathcal{D}^{\mathrm{tr}})$
$\mathcal{I}^{\star} \leftarrow \varnothing$ ▷ Current best candidate
feedback $E \leftarrow \varnothing$; ledger $H \leftarrow \varnothing$
while the search stopping rule is not met **do**
  $(P, a, Z) \leftarrow W(F, b, \mathcal{L}, \mathcal{I}^{\star}, E, H)$ ▷ Propose or revise
  $R \leftarrow \mathrm{ExecuteAndScore}(P, a, \mathcal{D}^{\mathrm{tr}}, \mathcal{L})$
  $g \leftarrow \mathrm{FitScope}(R, Z, b)$ ▷ Use declared diagnostics Z
  $m \leftarrow \mathrm{EvaluateCandidate}(P, g, a, R, b)$ ▷ Evaluation metadata
  $\mathcal{I} \leftarrow (P, g, a, m)$
  $E \leftarrow \mathrm{Feedback}(R, m, \mathcal{I}^{\star})$ ▷ Errors, diagnostics, regressions
  $H \leftarrow \mathrm{AppendAttempt}(H, \mathcal{I}, E)$
  $\mathcal{I}^{\star} \leftarrow \mathrm{KeepBest}(\mathcal{I}^{\star}, \mathcal{I})$
end while
return $\mathcal{I}^{\star}$ ▷ Deployment eligibility is checked across runs

3.3 Scope-Aware Direct Execution

Deployment uses selective prediction with agent fallback (Geifman & El-Yaniv, 2017). Each family has a frozen route r_f: a deployable program takes priority, followed by a text skill S_f when available, and otherwise the base agent. On the program route, we execute P_f and let $s_f(x)$ indicate that its answer is valid. For the collection of frozen family routes $\mathcal{A}$, the policy is

$$\pi_{\mathcal{A}}(x) = \begin{cases} a_f(v; x), & r_f = \text{program and } s_f(x) = 1, \\ \pi_0(x \mid S_f), & r_f = \text{skill}, \\ \pi_0(x), & \text{otherwise}. \end{cases} \tag{3}$$

An accepted program answer is returned directly. A rejected output falls back to the base agent without the program output or a text skill. Text skills enter the agent's system prompt only for families without deployable programs. Routes and scope rules remain frozen during evaluation. Appendix A.6 separates the policy's program and text-skill contributions.

4 Experiments

4.1 Experimental Setup

Benchmarks and supervision. We evaluate on four widely used benchmarks introduced at top-tier conferences: TIR-Bench, MIRA, GameQA, and PuzzleVQA (Li et al., 2026a; Zhou et al., 2026; Tong et al., 2026; Chia et al., 2024). Our evaluation covers 69 task families and 2,960 test questions. Appendix C.1 specifies the training/test partitions.

Models and baselines. Gemini-3.1-Pro authors the retained artifacts, which we evaluate with Gemini 3.1 Pro, Gemini 3.1 Flash-Lite, DeepSeek V4.1 Flash, and GPT-5.4-mini. Gemini-3.1-Pro and Flash-Lite use executor-specific training-side deployment checks. DeepSeek and GPT reuse frozen Flash-Lite (FL) routes without target-model training or recalibration. Program readings are shared across executors; Appendix C.4 specifies the evaluation protocol.

Table 1: **Main comparison with Gemini 3.1 Pro.** Accuracy (%); Avg. equally weights four task-weighted dataset accuracies. GQA: GameQA-17; PVQA: PuzzleVQA; TF: tool-free inference. Bold/underline mark best/second-best distinct displayed accuracies within each executor (ties included). Small green/red numbers are gains/losses in percentage points: GEPA/TextGrad versus Direct; XSkill/Trace2Skill/SkillOpt/VC versus Tool agent. Protocol details: Appendix C.4.

Method	TIR	MIRA	GQA	PVQA	Avg.
Direct (TF)	48.9	33.3	68.3	93.6	61.0
GEPA (TF)	52.9$_{+4.0}$	35.0$_{+1.7}$	73.7$_{+5.4}$	<u>97.8</u>$_{+4.3}$	64.9$_{+3.8}$
TextGrad (TF)	54.0$_{+5.1}$	33.3$_{0.0}$	75.1$_{+6.8}$	96.7$_{+3.1}$	64.8$_{+3.8}$
Tool agent	62.9	37.2	73.3	93.7	66.8
XSkill	<u>68.9</u>$_{+6.0}$	42.4$_{+5.2}$	78.4$_{+5.1}$	96.9$_{+3.2}$	<u>71.7</u>$_{+4.9}$
Trace2Skill	65.8$_{+2.9}$	<u>45.2</u>$_{+8.0}$	74.9$_{+1.6}$	95.7$_{+2.0}$	70.4$_{+3.6}$
SkillOpt	67.6$_{+4.7}$	37.7$_{+0.6}$	<u>78.8</u>$_{+5.4}$	97.6$_{+3.9}$	70.4$_{+3.6}$
VISUALCODING	**76.3**$_{+13.4}$	**47.1**$_{+9.9}$	**82.5**$_{+9.1}$	**98.6**$_{+4.9}$	**76.1**$_{+9.3}$

We compare XSkill, SkillOpt, Trace2Skill, GEPA, and TextGrad (Jiang et al., 2026; Yang et al., 2026b; Ni et al., 2026; Agrawal et al., 2025; Yuksekgonul et al., 2024). XSkill, SkillOpt, Trace2Skill, and VISUALCODING share the same tool suite. GEPA and TextGrad use tool-free inference.

Evaluation. Main comparisons and ablations use the same rollout for each base-agent or text-skill fallback and frozen program readings; each agent intervention uses one rollout. Dataset accuracy weights tasks, and Avg. equally weights the four datasets. The sampling analysis separately uses four recorded base-agent attempts per question. Agent answers are scored by model judges. Complete family-level scores appear in Appendix D.

4.2 MAIN COMPARISON

With Gemini-3.1-Pro, VISUALCODING improves single-run accuracy over the tool agent on all four benchmarks (Table 1). VISUALCODING raises mean accuracy from 66.78% to 76.11% (+9.33 points), compared with 71.67% for XSkill, the strongest external average under the evaluated adaptations. The largest gains are on TIR-Bench and MIRA, where the base agent leaves more questions unsolved, while PuzzleVQA rises from an already high 93.72% to 98.58%. The ablations below examine which parts of acquisition and deployment contribute to these gains.

4.3 CROSS-EXECUTOR REUSE

In addition, we compare the effectiveness of different methods for transferring the training outputs of Gemini-3.1-Pro to the less powerful Gemini-3.1-Flash-Lite. VISUALCODING raises mean accuracy from the tool agent's 43.42% to 68.57% (+25.14 points), compared with 53.78% for XSkill (Table 2(a)). The program branch preserves its stored computation when the deployment agent changes, while text and base-agent branches continue to depend on the target executor. This setting uses Flash-Lite-calibrated routes, so the larger gain than with Pro combines executor differences with route calibration. Appendix C.4 compares this configuration with the frozen Pro deployment policy to examine reuse without Flash-Lite-specific deployment calibration.

Panel (b) tests reuse of the frozen FL routes with DeepSeek and GPT, without training rollouts or deployment checks for either target. DeepSeek improves from 60.74% to 71.66%, and GPT from 34.70% to 63.70% (+10.92 and +29.00 points). Both targets use the same configuration across all four datasets. These gains demonstrate reuse of the retained artifacts and stored readings beyond the source executor, under each target's scoring protocol.

4.4 SUCCESS BEYOND THE SAMPLING POOL

VISUALCODING explores and refines reusable programs to improve single-run reliability, which enables performance that even surpasses that of multiple independent attempts. On the 746-question TIR-Bench panel, its estimated pass@1 exceeds the base agent's observed pass@4 in four of twelve

Table 2: **Cross-executor reuse.** Accuracy (%); annotations follow Table 1, ranked separately per executor. (a) External Pro-learned artifacts deploy with Flash-Lite; ours uses Flash-Lite training-side calibration. (b) DeepSeek/GPT reuse frozen FL routes and program readings without target calibration. Judge assignments and evaluation protocol: Appendix C.4.

Method	TIR-Bench	MIRA	GameQA	PuzzleVQA	Avg.
(a) Gemini 3.1 Flash-Lite					
Direct (TF)	28.3	17.9	28.6	78.8	38.4
GEPA (TF)	36.3$_{+8.0}$	18.5$_{+0.6}$	40.9$_{+12.9}$	91.8$_{+12.9}$	46.9$_{+8.4}$
TextGrad (TF)	33.5$_{+5.2}$	16.3$_{-1.7}$	32.5$_{+3.9}$	84.4$_{+5.5}$	41.7$_{+3.3}$
Tool agent	39.0	20.9	30.4	83.4	43.4
XSkill	48.7$_{+9.7}$	28.4$_{+7.4}$	45.1$_{+14.8}$	93.0$_{+9.6}$	53.8$_{+10.4}$
Trace2Skill	44.8$_{+5.8}$	24.5$_{+3.6}$	40.9$_{+10.5}$	86.9$_{+3.5}$	49.3$_{+5.8}$
SkillOpt	48.5$_{+9.5}$	22.0$_{+1.1}$	49.8$_{+19.5}$	93.0$_{+9.6}$	53.3$_{+9.9}$
VISUALCODING	**68.5**$_{+29.5}$	**41.6**$_{+20.7}$	**65.4**$_{+35.0}$	**98.8**$_{+15.4}$	**68.6**$_{+25.1}$
(b) Frozen FL routes; no target-model calibration					
DeepSeek V4.1 Flash					
Tool agent	59.1	33.3	60.7	89.8	60.7
VISUALCODING	**75.6**$_{+16.5}$	**44.4**$_{+11.0}$	**68.9**$_{+8.2}$	**97.8**$_{+8.0}$	**71.7**$_{+10.9}$
GPT-5.4-mini					
Tool agent	33.6	20.1	31.1	53.9	34.7
VISUALCODING	**64.9**$_{+31.2}$	**39.4**$_{+19.3}$	**59.7**$_{+28.6}$	**90.8**$_{+36.9}$	**63.7**$_{+29.0}$

task families (Table 7), including maze (100% vs. 95%), OCR (87.5% vs. 72.5%), and rotation game (100% vs. 98%).

Of the 194 questions on which all four base attempts fail, acquired programs take over 88 and answer 81 correctly. On the 211 intermittently solved questions, program-only policy accuracy reaches 62.80%, compared with the base agent's 52.25% estimated pass@1. These results suggest that deeper exploration and refinement within task families can improve single-run reliability enough to outperform repeated independent attempts in some families.

4.5 Execution Cases and Supporting Analyses

Figure 3 presents successful representative cases on four benchmarks. On TIR-Bench, the program extracts puzzle pieces, generates candidate assemblies using boundary compatibility, and ranks complete layouts to recover the image. On MIRA, it compares feature matches before and after reflection to identify the mirrored pattern. On GameQA, it converts the maze into a traversable grid and searches position–direction states to minimize turns. On PuzzleVQA, it matches measured circle areas to their associated numbers, using one model call to transcribe characters. Together, these examples illustrate how visual measurements become answers through task-specific scoring, search, and matching procedures.

The baseline traces illustrate why trajectory-based text skill fail in execution. XSkill and Trace2Skill compute local boundary matches but then manually specify puzzle layouts without optimizing the complete arrangement; SkillOpt repeatedly visualizes proposed layouts without executing its suggested matching procedure. In the other cases, unsuccessful runs misidentify petal order, describe a maze route that crosses an obstacle, or substitute an assumed numerical relation for size measurement. These failures show that relevant high-level guidance can leave critical implementation details and visual judgments to the deployment agent and lead to failure. Our acquisition instead repeatedly tests and refines a complete procedure across a task family. Execution feedback guides revisions to preprocessing, matching criteria, candidate generation, and search, while comparisons across training instances identify improvements and regressions. The retained program therefore captures the implementation details needed to connect visual measurements to the answer. Reusing this procedure reduces reliance on per-input implementation and repeated visual interpretation, while direct execution preserves the computed result instead of subjecting it to another model judgment. Together, these choices reduce opportunities for visual hallucination during deployment. The same-

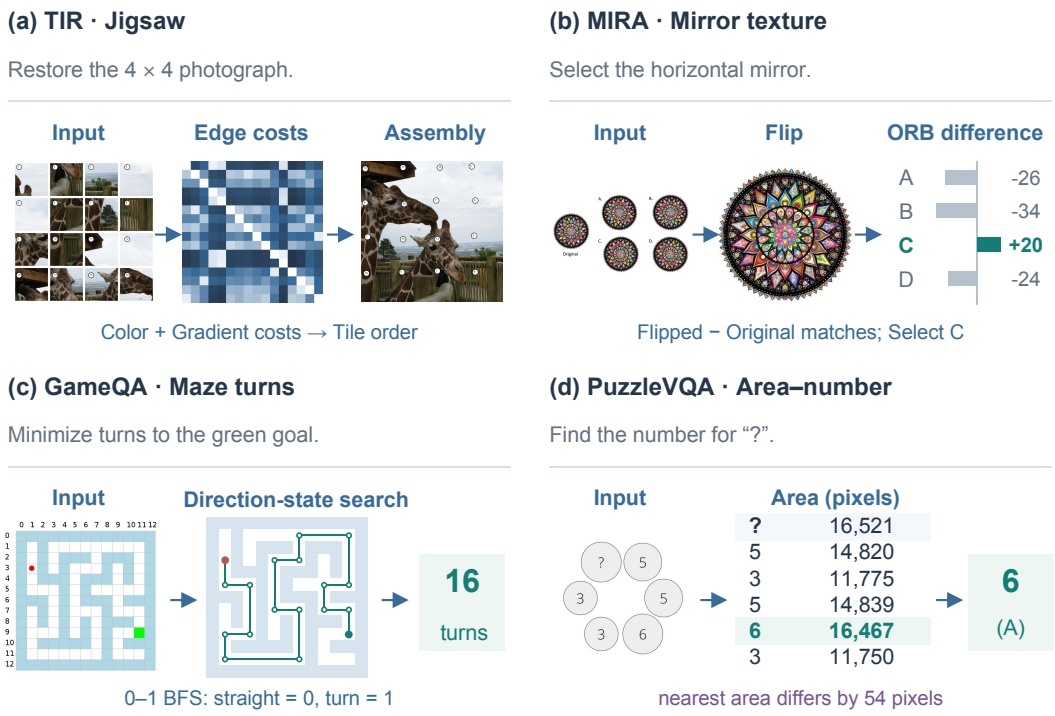

Figure 3: **Representative cases.** (a) TIR-Bench: lighter cells indicate lower boundary-matching costs. (b) MIRA: scores show the difference between reflected and original ORB match counts; C has the largest increase. (c) GameQA: the displayed path has 16 turns and is verified on the extracted grid. (d) PuzzleVQA: measured component areas identify six (A) as the answer.

program ablations in Section 4.6 examine the benefits of retaining and directly executing these computations.

4.6 Ablation Studies

Acquiring a reusable computation. Table 3(a) compares execution-guided revision with independent candidate search on TIR-Bench and MIRA using Gemini-3.1-Pro. Since at least five families have available programs across the benchmark, the feedback comparison re-acquires five families per benchmark, sharing initial facts, candidate slots, and selection rules. On these five families, text-skill fallback and the additional deployment check are disabled, and failed acquisitions revert to the tool agent; all other families retain their full-system outputs. Full feedback supplies earlier candidates, per-example outcomes, and diagnostics; independent search omits feedback from previous candidates. The no-scope acquisition arm retains full execution feedback while omitting scope fitting during program acquisition. All three conditions report mean accuracy over three independent acquisition runs. Compared with independent search, full feedback yields higher complete-panel accuracy: 75.9% versus 73.5% on

Table 3: **Acquisition and deployment ablations with Gemini-3.1-Pro.** Complete-panel accuracy (%): TIR-Bench has 746 questions and MIRA 363. Panel (a) compares feedback conditions on five re-acquired families under controlled deployment settings. Panel (b) changes how frozen main-experiment programs are used. Both panels retain full-system outputs on unaffected questions. Feedback-granularity and deployment-configuration controls appear in the appendix.

Variant	TIR-Bench	MIRA
Tool agent	62.9	37.2
(a) Acquisition		
Full execution feedback	75.9	46.6
Independent search	73.5	46.0
No scope during acquisition	68.3	42.0
(b) Deployment		
Frozen main-experiment programs		
Program-derived text procedure	69.4	44.4
Callable program	73.2	44.9
Reading injection	75.5	45.5
Full method	76.3	47.1

TIR and 46.6% versus 46.0% on MIRA. The full-feedback reference therefore uses different artifacts and deployment settings from the full method in panel (b). The no-scope arm achieves a mean complete-panel accuracy of 68.3% on TIR-Bench and 42.0% on MIRA.

Reusing the acquired implementation. Panel (b) holds the main experiment's acquired programs and deployed routes fixed while changing how the agent receives the computation. Direct execution in the full method reaches 76.3% on TIR and 47.1% on MIRA. Supplying the acquired programs as textual procedures yields 69.4% and 44.4%, respectively, while exposing them as callable code yields 73.2% and 44.9%. Callable code preserves the implementation, but leaves invocation and answer adoption to the agent. These controls compare deployment forms of the same acquired experience, showing that converting the program into text or enabling the agent to perform selective call introduces additional errors.

Preserving computed answers. Reading injection supplies the same frozen reading for a tool-free agent continuation, isolating answer use from program construction and invocation. Accuracy falls to 75.5% on TIR and 45.5% on MIRA. Among the 268 affected TIR takeovers, continuation changes 10 correct readings into incorrect answers and repairs four incorrect readings. On MIRA's 71 affected takeovers, the corresponding counts are seven and one. In each panel, six net lost answers account for the observed full-panel difference. These transitions identify answer continuation as a source of deployment loss after the computation is already available.

5 Discussion and Limitation

Acquiring and preserving experience. The ablations test acquisition and deployment separately (Table 3). Feedback-guided revision yields higher complete-panel accuracy than independent search, while direct execution outperforms textual and callable reuse on TIR-Bench and MIRA. Reading injection identifies losses from answer continuation; the broader reuse comparisons also involve implementation and invocation decisions. Programs additionally solve questions missed by all four observed baseline attempts, supporting recovery beyond a finite sampling pool.

Applicability and relative utility. Scope governs answer adoption within a family, while the deployment check determines which families use programs. With programs and scopes fixed, Flash-Lite benefits from both Pro and Flash-Lite policies, with higher accuracy under the latter (Appendix C.4). Larger transfer gains thus reflect both baseline headroom and deployment-policy choices.

Limitations. Due to API cost constraints, we use Gemini 3.1 Pro as the sole model for the program acquisition pipeline and do not explore other SOTA models in this role. For the same reason, our evidence of larger transfer gains for weaker models is limited to Gemini 3.1 Flash-Lite. Our main DeepSeek and GPT evaluations use the deployment policy calibrated for Flash-Lite. The Pro-calibrated deployment check may be overly conservative when reused with a weaker executor; relaxing it may improve accuracy while admitting programs whose relative benefit is less certain (Appendix C.4).

6 Conclusion

We present VisualCoding, a framework for learning and deploying executable skills for visual reasoning. It iteratively refines family-level programs using execution feedback on labeled training instances, then directly adopts their valid in-scope answers. Across four benchmarks, VisualCoding improves mean single-run accuracy over the tool agent by 9.33 percentage points with Gemini-3.1-Pro and 25.14 points with Gemini-3.1-Flash-Lite. The retained artifacts and stored program readings also improve DeepSeek and GPT deployment. Ablations on TIR-Bench and MIRA support feedback-guided acquisition and show higher mean accuracy from direct execution than from textual or callable reuse of the same programs. Acquired programs also solve questions missed by all four recorded base-agent attempts on some tasks. Together, these findings suggest that learning reusable computations can improve single-run reliability by reducing opportunities for execution errors and hallucinations during deployment.

AI USE STATEMENT

In this work, we have used AI tools for polishing writing. We have not used AI tools to support retrieval and discovery, to support research ideation and execution, to generate synthetic datasets, or to prove mathematical claims.

REPRODUCIBILITY STATEMENT

We release the anonymous code link at https://anonymous.4open.science/r/VisualCoding, which includes all code for data extraction, VisualCoding method and other five baseline methods.

Appendices A–C document implementation and evaluation protocols. Appendix D reports complete family-level results for the main comparisons, transfer evaluations, and TIR/MIRA ablations.

REFERENCES

Lakshya A Agrawal, Shangyin Tan, Dilara Soylu, Noah Ziems, Rishi Khare, Krista Opsahl-Ong, Arnav Singhvi, Herumb Shandilya, Michael J Ryan, Meng Jiang, Christopher Potts, Koushik Sen, Alexandros G. Dimakis, Ion Stoica, Dan Klein, Matei Zaharia, and Omar Khattab. GEPA: Reflective prompt evolution can outperform reinforcement learning. *arXiv preprint arXiv:2507.19457*, 2025. URL https://arxiv.org/abs/2507.19457.

Yew Ken Chia, Vernon Toh, Deepanway Ghosal, Lidong Bing, and Soujanya Poria. PuzzleVQA: Diagnosing multimodal reasoning challenges of language models with abstract visual patterns. In *Findings of the Association for Computational Linguistics: ACL 2024*, pp. 16259–16273, 2024. URL https://aclanthology.org/2024.findings-acl.962/.

Shengda Fan, Xin Cong, Zhong Zhang, Yuepeng Fu, Yesai Wu, Hao Wang, Xinyu Zhang, Enrui Hu, and Yankai Lin. Generalizing Experience for Language Agents with Hierarchical MetaFlows. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 64103–64132. Curran Associates, Inc., 2025. doi: 10.52202/085713-2149. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/5c882988ce5fac487974ee4f415b96a9-Paper-Conference.pdf.

Yonatan Geifman and Ran El-Yaniv. Selective classification for deep neural networks. *arXiv preprint arXiv:1705.08500*, 2017. URL https://arxiv.org/abs/1705.08500.

Tanmay Gupta and Aniruddha Kembhavi. Visual programming: Compositional visual reasoning without training. *arXiv preprint arXiv:2211.11559*, 2022. URL https://arxiv.org/abs/2211.11559.

Jack Hong, Chenxiao Zhao, ChengLin Zhu, Weiheng Lu, Guohai Xu, and Xing Yu. DeepEyesV2: Toward agentic multimodal model. *arXiv preprint arXiv:2511.05271*, 2025. URL https://arxiv.org/abs/2511.05271.

Muhammad Kamran Janjua, Hugo Silva, Di Niu, and Bahador Rashidi. Don't show pixels, show cues: Unlocking visual tool reasoning in language models via perception programs. *arXiv preprint arXiv:2604.12896*, 2026. URL https://arxiv.org/abs/2604.12896.

Chaoya Jiang, Yongrui Heng, Wei Ye, Haiyang Xu, Ming Yan, Ji Zhang, Fei Huang, and Shikun Zhang. Vlm-r3: Region recognition, reasoning, and refinement for enhanced multimodal chain-of-thought. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 63841–63869. Curran Associates, Inc., 2025. doi: 10.52202/085713-2143. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/5c4e0e38691e2aa08bba4cefc4c6e852-Paper-Conference.pdf.

Guanyu Jiang, Zhaochen Su, Xiaoye Qu, and Yi R. Fung. XSkill: Continual learning from experience and skills in multimodal agents. *arXiv preprint arXiv:2603.12056*, 2026. URL https://arxiv.org/abs/2603.12056.

Ming Li, Jike Zhong, Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Yuxiang Lai, Chen Wei, Konstantinos Psounis, and Kaipeng Zhang. Tir-bench: A comprehensive benchmark for agentic thinking-with-images reasoning. In *European Conference on Computer Vision*, pp. 464–481. Springer, 2026a.

Xiangyi Li, Yimin Liu, Wenbo Chen, Bingran You, Zonglin Di, Yifeng He, Shenghan Zheng, Kyoung Whan Choe, Jiankai Sun, Shuyi Wang, Chujun Tao, Binxu Li, Xuandong Zhao, Hejia Geng, Xiaojun Wu, Junwei Zhou, Xiaokun Chen, Hanwen Xing, Yubo Li, Qunhong Zeng, Di Wang, Yuanli Wang, Roey Ben Chaim, Penghao Jiang, Haotian Shen, Luyang Kong, Xinyi Liu, Runhui Wang, Xuanqing Liu, Jiachen Li, Xin Lan, Yueqian Lin, Wengao Ye, Junwei He, Songlin Li, Yue Zhang, Yipeng Gao, Yijiang Li, Ze Ma, Liqiang Jing, Tianyu Wang, Kaixin Li, Yiqi Xue, Haoran Lyu, Yizhuo He, Yuchen Tian, Shutong Wu, Bowei Wang, Yixuan Gao, Bo Chen, Litong Liu, Sikai Cheng, Jiajun Bao, Shuaicheng Tong, Shuwen Xu, Terry Yue Zhuo, Tinghan Ye, Qi Qi, Miao Li, Longtai Liao, Zelin Tan, Chang Shi, Xilin Tang, Srinath Tankasala, Boqin Yuan, Yaoyao Qian, Jianhong Tu, Chenguang Wang, Yizhou Sun, Wei Wang, Aaron Taylor, Ziyue Yang, Changkun Guan, Zhikang Dong, Xinyu Zhang, Steven Dillmann, Han chung Lee, and Dawn Song. Skillsbench: Benchmarking how well agent skills work across diverse tasks, 2026b. URL https://arxiv.org/abs/2602.12670.

Xiao Liu, Da Yin, Zirui Wu, and Yansong Feng. RefTool: Reference-Guided Tool Creation for Knowledge-Intensive Reasoning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 130663–130686, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/d453490ada2b1991852f053fbd213a6a-Paper-Conference.pdf.

Yuchen Ma, Yue Huang, Han Bao, Haomin Zhuang, Swadheen Shukla, Michel Galley, Xiangliang Zhang, and Stefan Feuerriegel. Skillgen: Verified inference-time agent skill synthesis, 2026. URL https://arxiv.org/abs/2605.10999.

Qirui Mi, Zhijian Ma, Mengyue Yang, Haoxuan Li, Yisen Wang, Haifeng Zhang, and Jun Wang. Skill-Pro: Learning reusable skills from experience via non-parametric PPO for LLM agents. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=9kJQjx2B80.

Jingwei Ni, Yihao Liu, Xinpeng Liu, Yutao Sun, Mengyu Zhou, Pengyu Cheng, Dexin Wang, Erchao Zhao, Xiaoxi Jiang, and Guanjun Jiang. Trace2Skill: Distill trajectory-local lessons into transferable agent skills. *arXiv preprint arXiv:2603.25158*, 2026. URL https://arxiv.org/abs/2603.25158.

Siru Ouyang, Jun Yan, I-Hung Hsu, Yanfei Chen, Ke Jiang, Zifeng Wang, Rujun Han, Long Le, Samira Daruki, Xiangru Tang, Vishy Tirumalashetty, George Lee, Mahsan Rofouei, Hangfei Lin, Jiawei Han, Chen-Yu Lee, and Tomas Pfister. ReasoningBank: Scaling Agent Self-Evolving with Reasoning Memory. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 94327–94354, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/980ea04d23d1f6908964eba2a74afe45-Paper-Conference.pdf.

Jie-Jing Shao, Haiyan Yin, Yueming Lyu, Xingrui Yu, Lan-Zhe Guo, Ivor Tsang, James Kwok, and Yu-Feng Li. Lifting traces to logic: Programmatic skill induction with neuro-symbolic learning for long-horizon agentic tasks. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=D0Dg8ISjq0.

Armando Solar-Lezama. *Program Synthesis by Sketching*. PhD thesis, University of California, Berkeley, 2008. URL https://www2.eecs.berkeley.edu/Pubs/TechRpts/2008/EECS-2008-176.html.

Mingyang Song, Haoyu Sun, Jiawei Gu, Linjie Li, Ranjay Krishna, and Yu Cheng. AdaReasoner: Dynamic Tool Orchestration for Iterative Visual Reasoning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 142731–142754,

2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e6f8759254d86ea9c197d30b92b313ca-Paper-Conference.pdf.

Adam Stein, Neelay Velingker, Mayur Naik, and Eric Wong. Once Upon an Input: Reasoning via Per-Instance Program Synthesis. In D. Belgrave, C. Zhang, H. Lin, R. Pascanu, P. Koniusz, M. Ghassemi, and N. Chen (eds.), *Advances in Neural Information Processing Systems*, volume 38, Main Conference, pp. 136940–136973. Curran Associates, Inc., 2025. doi: 10.52202/085713-4571. URL https://proceedings.neurips.cc/paper_files/paper/2025/file/c828f33af3c3f669690c2e28ae7af5e2-Paper-Conference.pdf.

Dídac Surís, Sachit Menon, and Carl Vondrick. ViperGPT: Visual inference via Python execution for reasoning. *arXiv preprint arXiv:2303.08128*, 2023. URL https://arxiv.org/abs/2303.08128.

Mirac Suzgun, Mert Yuksekgonul, Federico Bianchi, Dan Jurafsky, and James Zou. Dynamic cheatsheet: Test-time learning with adaptive memory, 2025. URL https://arxiv.org/abs/2504.07952.

Jingqi Tong, Jixin Tang, Hangcheng Li, Yurong Mou, Ming Zhang, Jun Zhao, Yanbo Wen, Fan Song, Jiahao Zhan, Yuyang Lu, Chaoran Tao, Zhiyuan Guo, Jizhou Yu, Tianhao Cheng, Zhiheng Xi, Changhao Jiang, Zhangyue Yin, Yining Zheng, Weifeng Ge, Guanhua Chen, Tao Gui, Xipeng Qiu, Qi Zhang, and Xuanjing Huang. Game-rl: Synthesizing multimodal verifiable game data to boost vlms'general reasoning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 34689–34755, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/3ad98e6306c5ed8a228b7240152f65af-Paper-Conference.pdf.

Guanzhi Wang, Yuqi Xie, Yunfan Jiang, Ajay Mandlekar, Chaowei Xiao, Yuke Zhu, Linxi Fan, and Anima Anandkumar. Voyager: An open-ended embodied agent with large language models, 2023. URL https://arxiv.org/abs/2305.16291.

Haozhe Wang, Alex Su, Weiming Ren, Fangzhen Lin, and Wenhu Chen. Pixel reasoner: Incentivizing pixel-space reasoning with curiosity-driven reinforcement learning. *arXiv preprint arXiv:2505.15966*, 2025. URL https://arxiv.org/abs/2505.15966.

Zora Zhiruo Wang, Jiayuan Mao, Daniel Fried, and Graham Neubig. Agent workflow memory, 2024. URL https://arxiv.org/abs/2409.07429.

Lei Wei, Qi Liu, Ruiyang Huang, Xiao Peng, Sicong Xie, Lanbo Lin, Chenhao Jiang, Yuanwu Xu, Tianyuan Yang, Jiayao Liu, Li Cai, Yuying Li, and Bin Wang. EvoC2F: Compiling tool orchestration for efficient and evolvable LLM agents. In *International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=ZSGB91kMOG.

Shengguang Wu, Xiaohan Wang, Yuhui Zhang, Hao Zhu, and Serena Yeung-Levy. Transductive visual programming: Evolving tool libraries from experience for spatial reasoning. *arXiv preprint arXiv:2512.20934*, 2025. URL https://arxiv.org/abs/2512.20934.

Tianze Yang, Liang Wu, Ruitong Sun, Yucheng Shi, Yanqiao Wang, Mayank Darbari, Ninghao Liu, Jin Sun, and Liangjie Hong. Self-evolving code-with-image reasoning. *arXiv preprint arXiv:2608.11292*, 2026a. URL https://arxiv.org/abs/2608.11292.

Yifan Yang, Ziyang Gong, Weiquan Huang, Qihao Yang, Ziwei Zhou, Zisu Huang, Yan Li, Xuemei Gao, Qi Dai, Bei Liu, Kai Qiu, Yuqing Yang, Dongdong Chen, Xue Yang, and Chong Luo. SkillOpt: Executive strategy for self-evolving agent skills. *arXiv preprint arXiv:2605.23904*, 2026b. URL https://arxiv.org/abs/2605.23904.

Haoran Ye, Xuning He, Vincent Arak, Haonan Dong, and Guojie Song. Meta context engineering via agentic skill evolution, 2026. URL https://arxiv.org/abs/2601.21557.

Simon Yu, Gang Li, Weiyan Shi, and Peng Qi. PolySkill: Learning Generalizable Skills Through Polymorphic Abstraction For Continual Learning. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 140298–140326,

2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/e350897ed832f9cad6f2e223e7acad6d-Paper-Conference.pdf.

Mert Yuksekgonul, Federico Bianchi, Joseph Boen, Sheng Liu, Zhi Huang, Carlos Guestrin, and James Zou. TextGrad: Automatic "differentiation" via text. *arXiv preprint arXiv:2406.07496*, 2024. URL https://arxiv.org/abs/2406.07496.

Kangning Zhang, Shuai Shao, Qingyao Li, Jianghao Lin, Lingyue Fu, Shijian Wang, Wenxiang Jiao, Yuan Lu, Weiwen Liu, Weinan Zhang, and Yong Yu. Mmskills: Towards multimodal skills for general visual agents, 2026a. URL https://arxiv.org/abs/2605.13527.

Qizheng Zhang, Changran Hu, Shubhangi Upasani, Boyuan Ma, Fenglu Hong, Vamsidhar Kamanuru, Jay Rainton, Chen Wu, Mengmeng Ji, Hanchen Li, Urmish Thakker, James Y Zou, and Kunle Olukotun. Agentic Context Engineering: Evolving Contexts for Self-Improving Language Models. In C. Vondrick, B. Hariharan, C. Raffel, L. Pinto, D. Yang, and A. Faust (eds.), *International Conference on Learning Representations*, volume 2026, pp. 86069–86100, 2026b. URL https://proceedings.iclr.cc/paper_files/paper/2026/file/8a94ff6f922d995d7d3f4ebf4143e442-Paper-Conference.pdf.

Shitian Zhao, Haoquan Zhang, Shaoheng Lin, Ming Li, Qilong Wu, Kaipeng Zhang, and Chen Wei. PyVision: Agentic vision with dynamic tooling. *arXiv preprint arXiv:2507.07998*, 2025. URL https://arxiv.org/abs/2507.07998.

Shitian Zhao, Shaoheng Lin, Ming Li, Haoquan Zhang, Wenshuo Peng, Kaipeng Zhang, and Chen Wei. PyVision-RL: Forging open agentic vision models via RL. *arXiv preprint arXiv:2602.20739*, 2026. URL https://arxiv.org/abs/2602.20739.

Ziwei Zheng, Michael Yang, Jack Hong, Chenxiao Zhao, Guohai Xu, Le Yang, Chao Shen, and Xing Yu. DeepEyes: Incentivizing "thinking with images" via reinforcement learning. *arXiv preprint arXiv:2505.14362*, 2025. URL https://arxiv.org/abs/2505.14362.

Yiyang Zhou, Haoqin Tu, Zijun Wang, Zeyu Wang, Niklas Muennighoff, Fan Nie, Chaorui Deng, Shen Yan, Haoqi Fan, Yejin Choi, et al. When visualizing is the first step to reasoning: Mira, a benchmark for visual chain-of-thought. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26154–26164, 2026.

A Implementation and Acquisition Details

A.1 Interfaces and Primitive Inventory

A deployable program is stored as $\mathcal{I}_f = (P_f, g_f, a_f, m_f)$, where m_f records dependencies and acquisition decisions. The frozen family record $\mathcal{A}_f = (r_f, \mathcal{I}_f, S_f)$ identifies its deployment route, program artifact, and text skill when available.

The acquired program exposes `measure(img, task, lib)` and returns a dictionary with `value` and `diag`. The task view contains the full question text, including answer choices when present, and permitted metadata, excluding the reference answer. A separate evaluator applies the answer adapter and compares with the label during acquisition or offline scoring. The adapter can interpret a numeric reading, a choice key, a coordinate, or a sequence according to the frozen answer convention. It is kept identical across deployment ablations. A failed execution or unparseable answer cannot become a direct answer.

The generic library contains 30 documented entries. Table 4 groups the generic entries by purpose. Numerical libraries, including NumPy, OpenCV, PIL, SciPy, and scikit-image, remain available to the program within the runner's restrictions.

Table 4: The fixed generic primitive library. Diagnostic returns can be included in the program's `diag`; debug images and substep checks support offline feedback.

Role	Documented entries
Image preparation	`arr`, `gray`, `background`, `fg_mask`, `ink_mask`, `hue_mask`, `stretch`, `upscale`, `rotate`
Structure and geometry	`components`, `fill_holes`, `holes`, `contour_stats`, `classify`, `min_area_rect`
Color measurements	`color_name`, `n_colors`, `color_share`, `mean_color`
Spatial patterns	`projection`, `period`, `panels8`, `apply_variant`
Perceptual interfaces	`detect`, `count_objects`, `ask`
Question processing	`question`, `numbers_in`
Execution inspection	`debug`, `check`

Programs can use pure code, detectors, or model-backed perception. The representation ablations give text and code controls the same coding access and primitives.

`ask` makes one model request per invocation.

A.2 Scope Fitting and Acceptance Criteria

Base-agent outcome statistics provide the acquisition reference. With K_{tr} rollouts for training example i, its empirical single-run accuracy is

$$b_i = \frac{1}{K_{\mathrm{tr}}} \sum_{j=1}^{K_{\mathrm{tr}}} \mathbb{I}[\hat{y}_{ij}^{(0)} = y_i]. \tag{4}$$

Equality denotes correctness under the fixed evaluator, with per-example denominators when rollout counts differ. The program writer receives labeled examples, these outcome statistics, and feedback from its candidate executions. Full stored baseline trajectories $\mathcal{T}_f^{\mathrm{tr}}$, including reasoning text and tool sequences, are inputs only to the separate text-skill summarizer.

For each numeric feature, the fitter sorts unique observed values and enumerates adjacent midpoints, subsampling to at most 30 thresholds per feature. It considers both upper and lower inequalities. If neighboring values are u_j, u_{j+1}, the associated separation is

$$t_j = \frac{u_j + u_{j+1}}{2}, \qquad m_j = \frac{u_{j+1} - u_j}{\max(u) - \min(u)}. \tag{5}$$

Constant numeric features do not generate thresholds. Boolean equality tests and categorical equality/inequality tests are also considered, with categorical enumeration limited to features having at

most ten observed values. Features are derived without ground-truth access. Raw output values and answer-option features are excluded by default unless explicitly permitted by the diagnostic declaration.

Each supported route receives a conjunction of up to two learned atoms in the default configuration, in addition to fixed requirements. The empty conjunction is allowed: a valid answer can be accepted without further learned comparisons. Fixed requirements are restricted to Boolean or string equality/inequality predicates so a writer-specified numeric threshold cannot bypass the fitter through that field. This restriction does not prevent program-internal numerical branching. Routes with no feasible rule are omitted from the accepted scope.

The default fitting objective maximizes coverage subject to the tier's precision and support requirements. Ties prefer larger paired gain, fewer learned atoms, higher precision, larger separation, and then a deterministic enumeration order. A gain-maximizing fitting objective is supported as a separate configuration; it is not the default. Rule stability is the fraction of leave-one-task-out folds whose fitted rule exactly matches the rule fitted on all stored training readings.

Table 5: Acceptance hyperparameters.

Criterion	High-precision tier	Gain tier
Minimum fitted precision	0.95	0.50
Minimum fitted support per route	3	5
Minimum pooled leave-one-out precision	0.90	0.50
Minimum rule stability	0.80	0.50
Maximum learned atoms per route	2	2
Minimum numeric separation	0.02	0.02
Minimum leave-one-out support	Nonempty	5
Positive paired-gain lower bound	Not required	Required

A preliminary hypothesis check tests whether the candidate's readings agree with training labels before fitting a selective scope. The standard path requires at least three parsed examples and agreement of at least 0.60, assessed over the pooled readings or a supported route. The gain path permits a five-example, 0.50-agreement criterion. Failed routes can be excluded from subsequent fitting.

For the gain tier, the harness forms paired differences $d_i = c_i - b_i$ on the leave-one-out accepted readings with a base-agent reference. It requires at least five such pairs and a positive one-sided 95% bootstrap lower bound on their mean, using 2,000 resamples and the implementation's fixed bootstrap seed of zero. This is a training-side selection heuristic under adaptive candidate search. It is not a multiple-search-adjusted guarantee of positive test gain. The same caution applies to the high-precision tier, since synthesis and diagnostic design have already used the training examples.

A.3 Computational Thresholds and Deployment Eligibility

Program-internal thresholds define a computation and may be proposed by the writer or computed adaptively from an input. Scope thresholds are fitted by the harness from training readings. Acceptance thresholds specify the candidate gate. A separate family-level coverage floor determines eligibility for deployment under the current adaptive policy. The gate and coverage floor are protocol hyperparameters, distinct from the fitted diagnostic thresholds.

A.4 Search Budgets and Artifact Selection

The budgets and selection rules below complement Algorithm 1 by specifying when search stops and which artifact is deployed.

The current simple profile has six base rounds with three candidate slots per round; the solver profile has eighteen base rounds with one candidate each and a higher reasoning setting. The default workflow can stop or extend according to progress and acceptance, so neither configuration implies exactly eighteen evaluated candidates. Production runs can extend after acceptance to improve coverage, while controlled acquisition ablations fix the candidate budget. All probes, repair attempts,

invalid proposals, and extensions contribute to acquisition cost. Candidate execution is subject to per-task resource limits.

Within a run, the ranking distinguishes high-precision, gain-tier, and unaccepted candidates, then uses leave-one-out gain, the hypothesis check, parsed accuracy, fitted gain, and coverage. The current selector considers artifacts accepted by the gate whose fitted scope also covers at least 50% of training examples. It orders eligible artifacts by leave-one-out gain, coverage, and run index. The retained artifact contains code, scope, answer convention, provenance, and dependency configuration. Saved candidates that fail either acceptance or deployment eligibility are not deployed.

The adaptive workflow stops when the latest run yields an eligible program or after three runs, then selects among eligible artifacts. If none is eligible, it uses text-skill fallback when available. The coverage floor controls both stopping and deployment eligibility without modifying the internal acceptance tiers.

A.5 Trajectory-Derived Skills and Frozen Family Routes

The text-skill summarizer filters stored rollouts by source, family, executor model, and training task IDs. The main protocol uses Gemini-3.1-Pro experience and a Gemini-3.1-Pro summarizer. It reads saved `trajectory_text`, removing the system prompt and adding outcome, final answer, reference answer, and termination status. It does not replay all intermediate image files. Per-task analysts inspect summaries, the question, label, and original image. Successes and failures yield up to three entries per applicable analyst. Tips are fused in batches of twelve and refined once, using temperature 0.3, top-$p = 1$, and an 8192-token output limit per request.

The summarizer has no access to acquired code, search ledgers, or validation/test performance. Existing family skills are reused rather than regenerated for each program-acquisition run. Formatting failures retain the last usable fused document; absent trajectories or a failure to produce a usable skill leave the base route. Unlike programs, skills undergo no accuracy or gain acceptance test.

A training-side route file records the family, selected program, skill path, and fallback mode. Under `fallback`, a deployment-eligible program has priority; its out-of-scope inputs use the unmodified executor. A family with no eligible program uses its text skill when available, otherwise the base agent. The driver reads the frozen route. `always` additionally applies skills outside program scope; `off` disables skills for core ablations. Transfer with a Flash-Lite-derived skill remains a transfer of that source's experience even when the deployment agent is Gemini-3.1-Pro; independently acquired Gemini-3.1-Pro artifacts are a separate configuration.

A.6 Policy Gain Decomposition

For an evaluation set of N examples, let b_i and h_i denote the measured single-run accuracies of the base and skill-guided agents, respectively. Let c_i denote program correctness, s_i indicate direct program adoption, and $t_i = \mathbb{I}[r_{f(i)} = \text{skill}]$ indicate the text-skill route. Since $s_i t_i = 0$, the policy score is $p_i = s_i c_i + t_i h_i + (1 - s_i - t_i) b_i$. Its mean gain over the base agent is

$$\Delta = \underbrace{\frac{1}{N} \sum_i s_i (c_i - b_i)}_{\text{program contribution}} + \underbrace{\frac{1}{N} \sum_i t_i (h_i - b_i)}_{\text{skill-fallback contribution}} . \tag{6}$$

The program term separates into correction mass $N^{-1} \sum_i s_i c_i (1 - b_i)$ minus harmful takeover mass $N^{-1} \sum_i s_i (1 - c_i) b_i$. This decomposition tracks what direct execution repairs and what it overrides incorrectly. The second term measures text-skill contribution on a disjoint set of inputs.

B Sampling Analysis

B.1 Family-Level Performance and Observed Sampling Recovery

Separate training and evaluation attempts. The program writer uses K_{tr} training rollouts only through their outcome statistics, together with labeled instances and candidate execution feedback. This TIR analysis uses $K_{\text{te}} = 4$ stored Pro base-agent attempts on each of 746 evaluation questions,

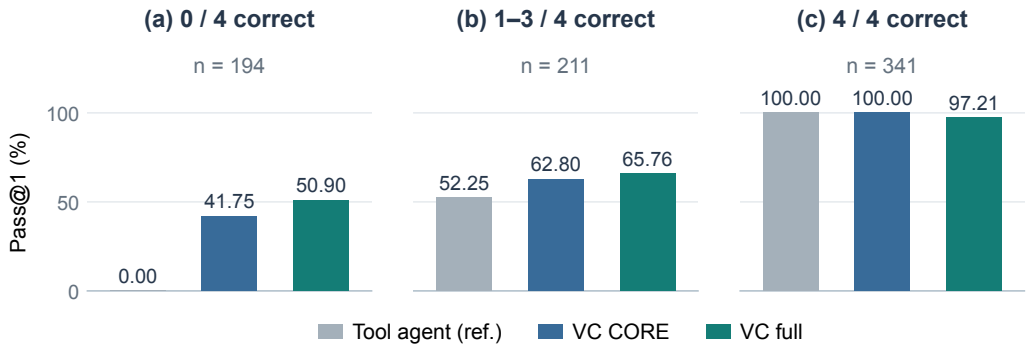

Figure 4: **Single-run accuracy across observed sampling strata.** TIR-Bench tasks are grouped by the number of correct answers among four stored Pro tool-agent attempts. Bars reproduce Table 6; CORE retains programs and scopes with text fallback disabled. Grouping fixes the reference rates at 0% and 100% in the extreme strata.

sampled at temperature 0.6. Policy estimates reuse frozen program readings and per-task skill/base means; skill repetitions are four except word search (one). Neither evaluation labels nor these evaluation attempts enter program writing, scope fitting, route selection, or skill construction.

Let $u_{ij} \in \{0, 1\}$ indicate correctness of evaluation attempt j on task i, and define $C_i = \sum_{j=1}^{K_{\text{te}}} u_{ij}$ and $O_i = \mathbb{I}[C_i > 0]$. On the same N tasks,

$$\widehat{\text{pass@1}}_0 = \frac{1}{N} \sum_i \frac{C_i}{K_{\text{te}}}, \qquad \widehat{\text{pass@}K_{\text{te}}}_0 = \frac{1}{N} \sum_i O_i. \tag{7}$$

The second statistic uses oracle correctness to identify whether any of the four observed answers succeeds. Every question has four recorded base attempts; the family comparisons and strata below use this complete pool.

Comparison on shared task identifiers. The base attempts and full VisualCoding policy are paired on the same 746 questions, including failed acquisition, rejected inputs, and text/base fallback. The frozen programs retain their acquisition references (Appendix C.5). Agent answers use the Pro judge, as documented in Appendix C.4.

Three observed strata. Partition tasks into $\mathcal{S}_0 = \{i : C_i = 0\}$, $\mathcal{S}_{\text{part}} = \{i : 0 < C_i < K_{\text{te}}\}$, and $\mathcal{S}_{\text{all}} = \{i : C_i = K_{\text{te}}\}$. Per-task single-run accuracy $\hat{p}_i$ averages the stored policy outcomes without best-output selection. Table 6 reports the task count and task-weighted mean accuracy in each stratum, along with complete-panel accuracy.

Table 6: **Success stratified by four observed Pro attempts on TIR.** Entries are pass@1 (%) on common baseline-defined strata. CORE disables text skills but keeps the same programs, scopes, and family eligibility; the full policy retains skill fallback. The tool-agent row uses the reference pool that defines the strata.

Method	$C_i = 0$	$0 < C_i < K_{\text{te}}$	$C_i = K_{\text{te}}$	Full panel
Task count	194	211	341	746
Tool agent (reference pool)	0.00	52.25	100.00	60.49
VisualCoding CORE	41.75	62.80	100.00	74.33
VisualCoding full	50.90	65.76	97.21	76.27

Figure 4 visualizes these three policies under the common baseline-defined strata.

TIR recovery and retained success. Among $\mathcal{S}_0$'s 194 tasks, 88 receive program takeovers and 81 are correct: 69 jigsaw, four maze, seven OCR, and one rotation question. These are successful program answers on questions for which all four recorded base-agent attempts fail. Of the remaining tasks in this stratum, 105 receive skills, contributing 17.75 summed per-task success estimates, and one receives base fallback. The fractional skill total sums estimated probabilities rather than counting distinct solved tasks. On the 211 intermittently solved tasks, CORE attains 62.80% single-run accuracy, compared with 52.25% for the base agent. In $\mathcal{S}_{\text{all}}$, CORE preserves all observed successes; the full policy attains 97.21%, with its 2.79-point conditional loss arising on skill routes.

Family-level comparison. Table 7 reports all twelve families. Full-policy estimated pass@1 exceeds the base agent's observed pass@4 in four: jigsaw (95.00% versus 11.25%), maze (100.00% versus 95.00%), OCR (87.50% versus 72.50%), and rotation game (100.00% versus 98.00%). Together with the per-question recovery counts, these comparisons show successful program answers beyond those observed in the four recorded baseline attempts.

Table 7: **Family-level Pro sampling reference and full-policy accuracy on TIR.** Percentages; all families are reported. OCR uses stored model-dependent program readings. Other rows can include program, skill, or base routes; these are complete-family scores, not in-scope precision.

Family	N	Base @1	Base @4	Full @1
Color	67	71.64	85.07	72.39
Contrast	34	63.24	79.41	63.97
Instrument	54	22.69	37.04	25.00
Jigsaw	80	4.38	11.25	95.00
Math	80	74.69	83.75	75.62
Maze	80	88.12	95.00	100.00
OCR	40	55.62	72.50	87.50
RefCOCO	80	44.06	70.00	58.12
Rotation game	50	89.50	98.00	100.00
Symbolic	34	78.68	94.12	88.24
Visual search	80	66.88	81.25	67.81
Word search	67	79.48	97.01	79.10

Interpretation of the sampling reference. The four recorded attempts provide a finite sampling reference rather than a capability upper bound. Additional attempts may move questions out of $\mathcal{S}_0$.

C Evaluation Accounting and Controls

C.1 Dataset Partitions

Table 8 lists the training and evaluation partitions. Program writing, scope fitting, and deployment checks use training data only.

Table 8: Training and evaluation partitions.

Dataset	Families	Train	Test
TIR-Bench	12	369	746
MIRA	20	183	363
GameQA	17	259	514
PuzzleVQA	20	663	1337

C.2 Model Roles and External Baseline Adaptations

Gemini-3.1-Pro authors the external artifacts, which remain fixed for deployment with Gemini-3.1-Pro and Gemini-3.1-Flash-Lite. XSkill, SkillOpt, and Trace2Skill have tool access; GEPA and TextGrad use tool-free inference. Direct answering and the tool agent provide their respective controls.

We extend Trace2Skill with a visual component for image-based tasks. Trace2Skill, SkillOpt, XSkill, and VisualCoding use the same tool suite. Appendix C.4 gives aggregation and evaluation details.

C.3 Frozen Transfer versus Deployment Calibration

Gemini-3.1-Pro and Gemini-3.1-Flash-Lite retain their own training-calibrated routes. DeepSeek and GPT reuse the frozen FL route without target-specific training checks. Programs, scopes, adapters, and stored readings are shared; the target agent executes text skills and the base-agent branch. Additional Pro-policy evaluations with Flash-Lite and DeepSeek appear in Appendix C.4.

Replacing the deployment agent with executor e changes the agent branches of Equation 3. Equation 6 uses that executor's $b_i^{(e)}$ and $h_i^{(e)}$, while the programs, scopes, and internal dependencies remain fixed. With identical stored readings and routes, the direct takeover set and program answers are shared by construction. Their utility remains relative to the target baseline.

A deployment-calibration variant retains P_f, g_f, and skills but recomputes the family route using target-executor training statistics. The training-side check requires at least five paired in-scope training readings and a positive one-sided 95% bootstrap lower bound for mean program-minus-executor correctness. A rejected program uses a Pro-trajectory text skill when available. This variant consumes target-executor training experience and changes the route.

C.4 Four-Benchmark Results: Aggregation and Evaluation

Aggregation. The four-benchmark suite comprises 69 task families, with 1,474 training questions and 2,960 test questions. Dataset accuracy is $\sum_f n_f a_f / \sum_f n_f$, where n_f is the number of test questions and a_f is the accuracy in family f. Avg. is the unweighted mean of the four dataset accuracies. Tables report accuracy to one decimal percentage point.

Model configuration and transfer. Pro-learned external artifacts remain fixed when deployed with Flash-Lite. Our Pro and Flash-Lite deployments use their respective training-calibrated family routes. Pro uses low thinking; Flash-Lite and DeepSeek disable thinking. DeepSeek and GPT reuse the frozen FL source route across all four datasets, without target-model training rollouts or deployment checks. Each target baseline and skill response is sampled once.

Answer scoring. For agent answers, the model judge compares the prediction with a supplied reference answer. In the main comparisons, Gemini 3.1 Pro experiments use Gemini 3.1 Pro as the judge; Flash-Lite, DeepSeek, and GPT experiments use Gemini 3.1 Flash-Lite. Because grading is a reference-based answer comparison, we use the lighter Flash-Lite judge for these subsequent evaluations.

Reported estimates and sampling. Main comparisons and ablations report point estimates. Acquisition ablations evaluate re-acquired family programs under a shared deployment protocol; deployment ablations use frozen artifacts with one recorded agent response per intervention task. The main TIR comparison and ablations share the fixed first rollout for base-agent and text-skill branches, giving a Pro tool-agent reference of 62.87%.

Results and interpretation. VisualCoding's four-dataset averages are 76.11% (Pro) and 68.57% (Flash-Lite), gains of 9.33 and 25.14 points over their own tool controls. Among the evaluated external methods, XSkill gives the highest averages, 71.67% and 53.78%; individual datasets can favor another external method. SkillOpt's four-dataset averages are 70.43% (Pro) and 53.33% (Flash-Lite).

Table 9: **Flash-Lite sensitivity to the family deployment policy.** Accuracy (%); every column uses Flash-Lite as the executor. Pro and FL denote the executor used for deployment calibration; no check removes the additional executor-specific check from the Pro configuration. The final column reports Pro-policy minus FL-policy accuracy in percentage points, with paired task-bootstrap 95% intervals from 4,000 resamples for each dataset and the pooled sample. Avg. equally weights datasets; Pooled weights all 2,960 questions.

Dataset (n)	Tool agent	Pro policy	Pro no check	FL policy	Pro − FL [95% CI]
TIR-Bench (746)	39.0	61.5	65.0	68.5	-7.0 $[-9.4, -4.7]$
MIRA (363)	20.9	36.4	38.3	41.6	-5.2 $[-8.0, -3.0]$
GameQA (514)	30.4	52.7	58.6	65.4	-12.6 $[-16.0, -9.5]$
PuzzleVQA (1,337)	83.4	97.1	98.7	98.8	-1.7 $[-2.5, -0.9]$
Avg. (datasets)	43.4	61.9	65.1	68.6	-6.6
Pooled (2,960)	55.3	73.0	75.8	78.3	-5.4 $[-6.3, -4.5]$

Source-policy sensitivity. Here, a deployment policy specifies whether each family uses its retained program or a fallback skill. Table 9 compares Flash-Lite under the frozen Pro policy, its own calibrated policy, and the Pro policy with the additional executor-specific deployment check removed. Programs, scopes, and retained skills remain fixed. The checked Pro and FL policies enable programs in 27 and 40 of the 69 families, respectively. For the 13 families that switch from programs to text skills under the Pro policy, we collect 649 Flash-Lite skill-guided responses; unchanged outputs and frozen program readings are reused.

The frozen Pro policy improves Flash-Lite on all four benchmarks, raising equal-dataset mean accuracy from 43.42% to 61.92%. Removing the executor-specific check raises this mean to 65.15%, while Flash-Lite-specific calibration reaches 68.57%. These results suggest that a check calibrated against Pro can exclude programs useful to a weaker executor. The no-check condition retains source eligibility and program scope, so its gain concerns family-level program selection.

DeepSeek's mean is 69.76% with the Pro policy and 71.66% with the FL policy, versus 60.74% for its tool agent. Source-policy effects vary: the Pro policy is better on TIR (76.54% versus 75.60%), whereas the FL policy is better on GameQA (68.87% versus 63.23%), MIRA (44.35% versus 41.87%), and PuzzleVQA (97.83% versus 97.38%).

C.5 TIR-Bench Ablations

Panel and acquisition. TIR-Bench contains 12 families, 746 test questions, and 369 training questions. The paired test baseline uses the same fixed first Pro rollout per question as Table 1; agent answers use the Pro judge (Appendix C.4). The acquisition-feedback comparison evaluates Full execution feedback (FULLC), Independent searc (IID), SCALAR and no scope on jigsaw, RefCOCO, word search, symbolic, and maze. Within each family, these arms share initial facts, training references, candidate budgets, and selection rules. FULLC uses earlier candidates and their execution feedback; IID proposes candidates independently while retaining the same execution and selection process. Pro training references use four rollouts per question except for jigsaw, which uses one. Jigsaw has 18 candidate slots; the other families have 36.

For both benchmarks, complete-panel evaluation combines the re-acquired family programs with the main experiment's fixed single-rollout outputs on unaffected families. Base-agent fallback uses the fixed first rollout on each question; unchanged skill branches use their fixed first skill-guided rollout. On re-acquired families, eligibility requires source acceptance and at least 50% training coverage; text skills and the additional executor-specific check are disabled, and failed acquisitions revert to the base agent. SCALAR retains the best code and aggregate execution outcomes but omits per-example feedback and error-selected images. Its complete-panel accuracy is 69.7% on TIR. Appendix D.2 reports the complete-family acquisition and deployment results.

Shared representation and continuation protocols. The representation and answer-continuation comparisons use the frozen main-experiment programs and their original acquisition references; the shared protocols below apply to both benchmarks. REP-TEXT supplies a procedure translated

by Pro from the frozen program, checked for omissions and interface consistency on five training questions per family before freezing. These procedures are distinct from the system's trajectory-derived fallback skills. REP-CODE supplies the complete source and a callable implementation that the agent may inspect, call, modify, or ignore. Both arms retain coding access and share the primitive library, task interface, answer adapter, original-image access, and a budget of 30 turns and 65,536 tokens per task.

DEC-DIRECT returns the answer from the frozen program reading. DEC-INJECT supplies Pro with the question, image, computed value, permitted diagnostics, applicability status, and candidate answer for continuation without tools or additional perception calls. Readings, scopes, and prompts remain fixed. Each agent intervention has one rollout per question and produces the final answer.

TIR's representation intervention replaces 228 deployed takeovers: jigsaw 80, maze 80, rotation 50, and symbolic 18. OCR's 40 takeovers remain direct because its model interface is unavailable in the agent kernel. The answer-continuation comparison includes these OCR takeovers, affecting 268 questions. All other system outputs stay fixed. The text-skill fallback control appears in Table 10.

C.6 MIRA Ablations

Acquisition protocol. MIRA contains 20 families and 363 test questions. The acquisition comparison selects the five pure-pixel program families with the lowest Pro training pass@1 using training-side information: billiards, puzzle, mirror pattern, trailer cubes count, and mirror clock. Acquisition uses 36 candidate slots and 7–10 training examples per family. The writer and four-rollout training reference are Pro; test fallback uses one stored Pro rollout. Other acquisition settings follow Appendix C.5. SCALAR's complete-panel accuracy is 46.5%. Deployment interventions use frozen programs acquired with Flash-Lite training references, separately from the Pro-referenced acquisitions in the feedback comparison.

Deployment interventions. The representation and answer-continuation comparisons follow the shared protocol and use the Pro judge. They replace 71 deployed takeovers: 14 billiards, 20 mirror clock, 20 mirror pattern, and 17 puzzle. All other outputs remain fixed, including the original direct outputs of model-read families. In the CODE condition, the agent calls the supplied mirror-clock program on 10% of questions.

Table 10 isolates text-skill fallback on both benchmarks. The no-text-fallback condition replaces the text-skill branch with the base agent while keeping program routes and their out-of-scope fallback unchanged. Both conditions use the same fixed first base-agent and skill-guided rollouts as the main comparison. Text-skill fallback adds 0.3 percentage points on TIR and 0.8 on MIRA under this shared evaluation protocol.

Table 10: **Text-skill fallback control.** Complete-panel accuracy (%) with Pro on the same TIR-Bench and MIRA panels as Table 3.

Variant	TIR-Bench	MIRA
Full method	76.3	47.1
No text fallback	76.0	46.3

D Complete Family-Level Results

All 69 families in the four-benchmark main suite are included below, in the same benchmark order as Tables 1 and 2. Entries are complete-family accuracy (%), including program rejection and fallback; n counts test questions. Dataset totals are task-weighted, not averages of family percentages. Unrounded family scores reproduce every corresponding main-table dataset entry. Family identifiers omit the dataset prefixes `mr_`, `gq_`, and `pz_`; underscores are displayed as spaces.

D.1 Main Comparison and Cross-Executor Reuse

Dir. denotes tool-free direct answering; Tool the tool agent; XS, SO, T2S, GEPA, and TG denote XSkill, SkillOpt, Trace2Skill, GEPA, and TextGrad; VC denotes VisualCoding. GEPA and TG are tool-free adaptations. Pro and Flash-Lite retain their respective training-calibrated routes. DeepSeek and GPT reuse the FL route and stored program readings.

Bold/underline mark best/second-best distinct displayed accuracies within each executor (ties included). Small green/red numbers are gains/losses in percentage points: GEPA/TextGrad versus Direct; XSkill/Trace2Skill/SkillOpt/VC versus Tool agent. In these dense family tables the change is placed below the accuracy; shaded cells identify VC. Rankings are within each family and executor, with dataset totals ranked separately.

Table 11: **TIR-Bench: Gemini 3.1 Pro.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
color	67	50.7	59.7 +9.0	52.2 +1.5	<u>74.6</u>	<u>74.6</u> 0.0	**77.6** +3.0	73.1 -1.5	70.1 -4.5
contrast	34	64.7	47.1 -17.6	58.8 -5.9	64.7	**73.5** +8.8	64.7 0.0	<u>67.7</u> +2.9	52.9 -11.8
instrument	54	24.1	<u>25.9</u> +1.9	24.1 0.0	24.1	<u>25.9</u> +1.9	24.1 0.0	**31.5** +7.4	<u>25.9</u> +1.9
jigsaw	80	1.2	3.8 +2.5	5.0 +3.8	5.0	2.5 -2.5	7.5 +2.5	<u>10.0</u> +5.0	**95.0** +90.0
math	80	58.8	58.8 0.0	63.7 +5.0	<u>77.5</u>	<u>77.5</u> 0.0	72.5 -5.0	72.5 -5.0	**81.2** +3.7
maze	80	91.2	<u>98.8</u> +7.5	<u>98.8</u> +7.5	90.0	97.5 +7.5	88.8 -1.3	96.2 +6.2	**100.0** +10.0
ocr	40	30.0	47.5 +17.5	40.0 +10.0	55.0	77.5 +22.5	<u>82.5</u> +27.5	80.0 +25.0	**87.5** +32.5
refcoco	80	36.2	56.2 +20.0	<u>61.3</u> +25.0	47.5	**70.0** +22.5	56.2 +8.8	56.2 +8.8	55.0 +7.5
rotation game	50	44.0	52.0 +8.0	62.0 +18.0	92.0	**100.0** +8.0	**100.0** +8.0	<u>96.0</u> +4.0	**100.0** +8.0
symbolic	34	73.5	64.7 -8.8	64.7 -8.8	88.2	**94.1** +5.9	88.2 0.0	<u>91.2</u> +2.9	<u>91.2</u> +2.9
visual search	80	65.0	66.2 +1.2	61.3 -3.7	70.0	**73.8** +3.8	67.5 -2.5	<u>71.2</u> +1.3	70.0 0.0
word search	67	52.2	46.3 -6.0	50.7 -1.5	80.6	82.1 +1.5	<u>85.1</u> +4.5	**88.1** +7.5	79.1 -1.5
Dataset total	746	48.9	52.9 +4.0	54.0 +5.1	62.9	<u>68.9</u> +6.0	65.8 +2.9	67.6 +4.7	**76.3** +13.4

Table 12: **TIR-Bench: Gemini 3.1 Flash-Lite.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
color	67	41.8	46.3 +4.5	52.2 +10.4	61.2	64.2 +3.0	<u>68.7</u> +7.5	**73.1** +11.9	56.7 -4.5
contrast	34	41.2	44.1 +2.9	47.1 +5.9	55.9	<u>61.8</u> +5.9	**64.7** +8.8	58.8 +2.9	55.9 0.0
instrument	54	14.8	20.4 +5.6	14.8 0.0	13.0	<u>24.1</u> +11.1	<u>24.1</u> +11.1	**27.8** +14.8	16.7 +3.7
jigsaw	80	<u>0.0</u>	<u>0.0</u> 0.0	<u>0.0</u> 0.0	<u>0.0</u>	<u>0.0</u> 0.0	<u>0.0</u> 0.0	<u>0.0</u> 0.0	**95.0** +95.0
math	80	33.8	45.0 +11.2	38.8 +5.0	51.2	**56.2** +5.0	53.8 +2.5	<u>55.0</u> +3.8	41.2 -10.0
maze	80	23.8	32.5 +8.8	23.8 0.0	<u>37.5</u>	35.0 -2.5	35.0 -2.5	28.7 -8.8	**100.0** +62.5
ocr	40	15.0	22.5 +7.5	15.0 0.0	55.0	65.0 +10.0	52.5 -2.5	<u>70.0</u> +15.0	**87.5** +32.5

Continued on next page

Table 12 (continued)

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
refcoco	80	17.5	56.2 +38.8	46.2 +28.8	22.5	35.0 +12.5	27.5 +5.0	35.0 +12.5	**60.0** +37.5
rotation game	50	46.0	50.0 +4.0	46.0 0.0	44.0	78.0 +34.0	70.0 +26.0	84.0 +40.0	**100.0** +56.0
symbolic	34	35.3	35.3 0.0	38.2 +2.9	67.6	73.5 +5.9	76.5 +8.8	67.7 0.0	**79.4** +11.8
visual search	80	48.8	52.5 +3.8	52.5 +3.8	61.3	63.7 +2.5	61.3 0.0	**65.0** +3.7	60.0 -1.3
word search	67	31.3	28.4 -3.0	29.9 -1.5	28.4	65.7 +37.3	43.3 +14.9	56.7 +28.4	**71.6** +43.3
Dataset total	746	28.3	36.3 +8.0	33.5 +5.2	39.0	48.7 +9.7	44.8 +5.8	48.5 +9.5	**68.5** +29.5

Table 13: **TIR-Bench: DeepSeek and GPT with frozen FL routes.** Complete-family accuracy (%).

Family	n	DS Tool	DS VC	GPT Tool	GPT VC
color	67	**74.6**	64.2 -10.4	50.7	**62.7** +11.9
contrast	34	58.8	**61.8** +2.9	26.5	**32.4** +5.9
instrument	54	14.8	**25.9** +11.1	16.7	**31.5** +14.8
jigsaw	80	5.0	**95.0** +90.0	1.2	**95.0** +93.8
math	80	**81.2**	77.5 -3.7	**37.5**	35.0 -2.5
maze	80	68.8	**100.0** +31.2	57.5	**100.0** +42.5
ocr	40	40.0	**87.5** +47.5	37.5	**87.5** +50.0
refcoco	80	53.8	**60.0** +6.2	38.8	**60.0** +21.2
rotation game	50	80.0	**100.0** +20.0	42.0	**100.0** +58.0
symbolic	34	76.5	**79.4** +2.9	41.2	**61.8** +20.6
visual search	80	**70.0**	62.5 -7.5	35.0	**40.0** +5.0
word search	67	**86.6**	**86.6** 0.0	19.4	**65.7** +46.3
Dataset total	746	59.1	**75.6** +16.5	33.6	**64.9** +31.2

Table 14: **MIRA: Gemini 3.1 Pro.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
billiards	14	14.3	28.6 +14.3	50.0 +35.7	35.7	64.3 +28.6	50.0 +14.3	50.0 +14.3	**71.4** +35.7
convex hull	20	55.0	55.0 0.0	25.0 -30.0	90.0	90.0 0.0	**95.0** +5.0	**95.0** +5.0	90.0 0.0
cubes count	20	**5.0**	**5.0** 0.0	0.0 -5.0	**5.0**	**5.0** 0.0	**5.0** 0.0	0.0 -5.0	**5.0** 0.0
cubes missing	20	**5.0**	0.0 -5.0	**5.0** 0.0	**5.0**	0.0 -5.0	0.0 -5.0	**5.0** 0.0	0.0 -5.0
defuse a bomb	17	17.6	17.6 0.0	**35.3** +17.6	23.5	**35.3** +11.8	23.5 0.0	29.4 +5.9	17.6 -5.9
electric charge	14	78.6	**85.7** +7.1	**85.7** +7.1	78.6	**85.7** +7.1	78.6 0.0	**85.7** +7.1	**85.7** +7.1

Continued on next page

Table 14 (continued)

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
gear rotation	26	**100.0**	96.2 -3.8	**100.0** 0.0	96.2	**100.0** +3.8	**100.0** +3.8	**100.0** +3.8	**100.0** +3.8
localizer	16	**0.0**	**0.0** 0.0	**0.0** 0.0	**0.0**	**0.0** 0.0	**0.0** 0.0	**0.0** 0.0	**0.0** 0.0
mirror clock	20	80.0	75.0 -5.0	40.0 -40.0	75.0	80.0 +5.0	80.0 +5.0	75.0 0.0	**100.0** +25.0
mirror pattern	20	25.0	35.0 +10.0	25.0 0.0	20.0	65.0 +45.0	90.0 +70.0	55.0 +35.0	**100.0** +80.0
multi piece puzzle	20	10.0	10.0 0.0	**15.0** +5.0	5.0	5.0 0.0	**15.0** +10.0	10.0 +5.0	5.0 0.0
overlap	20	45.0	40.0 -5.0	50.0 +5.0	65.0	**80.0** +15.0	75.0 +10.0	40.0 -25.0	60.0 -5.0
paper airplane	17	23.5	**47.1** +23.5	29.4 +5.9	17.6	41.2 +23.5	35.3 +17.6	29.4 +11.8	29.4 +11.8
puzzle	17	47.1	47.1 0.0	**58.8** +11.8	52.9	47.1 -5.9	41.2 -11.8	35.3 -17.7	23.5 -29.4
rolling dice sum	17	0.0	5.9 +5.9	0.0 0.0	5.9	11.8 +5.9	17.6 +11.8	0.0 -5.9	**58.8** +52.9
rolling dice top	17	41.2	35.3 -5.9	41.2 0.0	29.4	17.6 -11.8	**58.8** +29.4	29.4 0.0	47.1 +17.6
rolling dice two	17	0.0	0.0 0.0	**11.8** +11.8	0.0	0.0 0.0	0.0 0.0	0.0 0.0	5.9 +5.9
trailer cubes count	17	70.6	58.8 -11.8	64.7 -5.9	70.6	64.7 -5.9	64.7 -5.9	64.7 -5.9	**76.5** +5.9
trailer cubes missing	17	**0.0**	**0.0** 0.0	**0.0** 0.0	**0.0**	**0.0** 0.0	**0.0** 0.0	**0.0** 0.0	**0.0** 0.0
unfolded cube	17	17.6	35.3 +17.6	17.6 0.0	**41.2**	29.4 -11.8	**41.2** 0.0	23.5 -17.6	**41.2** 0.0
Dataset total	363	33.3	35.0 +1.7	33.3 0.0	37.2	42.4 +5.2	45.2 +8.0	37.7 +0.6	**47.1** +9.9

Table 15: **MIRA: Gemini 3.1 Flash-Lite.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
billiards	14	28.6	0.0 -28.6	28.6 0.0	35.7	42.9 +7.1	28.6 -7.1	28.6 -7.1	**71.4** +35.7
convex hull	20	15.0	20.0 +5.0	10.0 -5.0	85.0	**90.0** +5.0	**90.0** +5.0	80.0 -5.0	85.0 0.0
cubes count	20	0.0	0.0 0.0	0.0 0.0	0.0	**5.0** +5.0	0.0 0.0	0.0 0.0	0.0 0.0
cubes missing	20	**5.0**	0.0 -5.0	0.0 -5.0	0.0	**5.0** +5.0	0.0 0.0	0.0 0.0	**5.0** +5.0
defuse a bomb	17	35.3	**47.1** +11.8	17.6 -17.6	17.6	29.4 +11.8	41.2 +23.5	23.5 +5.9	11.8 -5.9
electric charge	14	64.3	57.1 -7.1	71.4 +7.1	**85.7**	64.3 -21.4	78.6 -7.1	**85.7** 0.0	78.6 -7.1
gear rotation	26	42.3	**80.8** +38.5	57.7 +15.4	38.5	42.3 +3.8	65.4 +26.9	42.3 +3.8	65.4 +26.9
localizer	16	**0.0**	**0.0** 0.0	**0.0** 0.0	**0.0**	**0.0** 0.0	**0.0** 0.0	**0.0** 0.0	**0.0** 0.0
mirror clock	20	15.0	0.0 -15.0	5.0 -10.0	15.0	25.0 +10.0	35.0 +20.0	15.0 0.0	**100.0** +85.0
mirror pattern	20	35.0	20.0 -15.0	25.0 -10.0	25.0	50.0 +25.0	55.0 +30.0	25.0 0.0	**100.0** +75.0
multi piece puzzle	20	0.0	0.0 0.0	0.0 0.0	**5.0**	0.0 -5.0	0.0 -5.0	0.0 -5.0	0.0 -5.0
overlap	20	35.0	40.0 +5.0	30.0 -5.0	30.0	**80.0** +50.0	35.0 +5.0	35.0 +5.0	25.0 -5.0
paper airplane	17	29.4	**35.3** +5.9	23.5 -5.9	11.8	23.5 +11.8	23.5 +11.8	17.6 +5.9	17.6 +5.9
puzzle	17	29.4	17.6 -11.8	23.5 -5.9	23.5	**52.9** +29.4	5.9 -17.6	29.4 +5.9	23.5 0.0

Continued on next page

Table 15 (continued)

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
rolling dice sum	17	0.0	5.9 +5.9	0.0 0.0	5.9	11.8 +5.9	0.0 -5.9	5.9 0.0	**58.8** +52.9
rolling dice top	17	11.8	5.9 -5.9	23.5 +11.8	35.3	17.6 -17.6	11.8 -23.5	41.2 +5.9	**47.1** +11.8
rolling dice two	17	0.0	**11.8** +11.8	0.0 0.0	0.0	0.0 0.0	0.0 0.0	0.0 0.0	0.0 0.0
trailer cubes count	17	5.9	0.0 -5.9	0.0 -5.9	5.9	5.9 0.0	0.0 -5.9	0.0 -5.9	**94.1** +88.2
trailer cubes missing	17	0.0	0.0 0.0	0.0 0.0	0.0	**5.9** +5.9	0.0 0.0	**5.9** +5.9	**5.9** +5.9
unfolded cube	17	5.9	5.9 0.0	5.9 0.0	0.0	5.9 +5.9	0.0 0.0	5.9 +5.9	**35.3** +35.3
Dataset total	363	17.9	18.5 +0.6	16.3 -1.7	20.9	28.4 +7.4	24.5 +3.6	22.0 +1.1	**41.6** +20.7

Table 16: **MIRA: DeepSeek and GPT with frozen FL routes.** Complete-family accuracy (%).

Family	n	DS Tool	DS VC	GPT Tool	GPT VC
billiards	14	57.1	**71.4** +14.3	14.3	**71.4** +57.1
convex hull	20	**95.0**	90.0 -5.0	**65.0**	60.0 -5.0
cubes count	20	**5.0**	**5.0** 0.0	**0.0**	**0.0** 0.0
cubes missing	20	**0.0**	**0.0** 0.0	**0.0**	**0.0** 0.0
defuse a bomb	17	**35.3**	17.6 -17.6	23.5	**29.4** +5.9
electric charge	14	**71.4**	**71.4** 0.0	50.0	**64.3** +14.3
gear rotation	26	**50.0**	46.2 -3.8	42.3	**50.0** +7.7
localizer	16	**0.0**	**0.0** 0.0	**0.0**	**0.0** 0.0
mirror clock	20	55.0	**100.0** +45.0	25.0	**100.0** +75.0
mirror pattern	20	30.0	**100.0** +70.0	30.0	**100.0** +70.0
multi piece puzzle	20	**10.0**	0.0 -10.0	**5.0**	0.0 -5.0
overlap	20	60.0	**80.0** +20.0	**40.0**	35.0 -5.0
paper airplane	17	11.8	**41.2** +29.4	**11.8**	**11.8** 0.0
puzzle	17	**52.9**	23.5 -29.4	**35.3**	23.5 -11.8
rolling dice sum	17	5.9	**58.8** +52.9	0.0	**58.8** +58.8
rolling dice top	17	35.3	**47.1** +11.8	35.3	**47.1** +11.8
rolling dice two	17	**0.0**	**0.0** 0.0	**0.0**	**0.0** 0.0
trailer cubes count	17	76.5	**94.1** +17.6	11.8	**94.1** +82.4
trailer cubes missing	17	**0.0**	**0.0** 0.0	0.0	**5.9** +5.9
unfolded cube	17	11.8	**35.3** +23.5	0.0	**35.3** +35.3
Dataset total	363	33.3	**44.4** +11.0	20.1	**39.4** +19.3

Table 17: **GameQA-17: Gemini 3.1 Pro.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
jewel2 q5	40	**90.0**	87.5 -2.5	**90.0** 0.0	87.5	**90.0** +2.5	87.5 0.0	87.5 0.0	87.5 0.0
jewel2 q6	20	60.0	65.0 +5.0	70.0 +10.0	75.0	75.0 0.0	70.0 -5.0	65.0 -10.0	**80.0** +5.0
maze q4	35	51.4	45.7 -5.7	60.0 +8.6	45.7	77.1 +31.4	60.0 +14.3	57.1 +11.4	**100.0** +54.3
minesweeper q5	40	87.5	**90.0** +2.5	**90.0** +2.5	**90.0**	**90.0** 0.0	**90.0** 0.0	80.0 -10.0	**90.0** 0.0
pacman q1	20	20.0	20.0 0.0	15.0 -5.0	35.0	50.0 +15.0	45.0 +10.0	**95.0** +60.0	**95.0** +60.0
pacman q3	20	55.0	60.0 +5.0	60.0 +5.0	65.0	90.0 +25.0	55.0 -10.0	75.0 +10.0	**100.0** +35.0
pacman q4	20	80.0	80.0 0.0	80.0 0.0	75.0	65.0 -10.0	**85.0** +10.0	80.0 +5.0	70.0 -5.0
rhythm game q2	27	85.2	85.2 0.0	85.2 0.0	88.9	81.5 -7.4	88.9 0.0	**92.6** +3.7	**92.6** +3.7
rhythm game q7	27	92.6	**100.0** +7.4	**100.0** +7.4	85.2	**100.0** +14.8	96.3 +11.1	**100.0** +14.8	**100.0** +14.8
rubiks cube q3	27	77.8	88.9 +11.1	85.2 +7.4	85.2	85.2 0.0	**92.6** +7.4	88.9 +3.7	**92.6** +7.4
rubiks cube q4	31	74.2	87.1 +12.9	87.1 +12.9	83.9	87.1 +3.2	83.9 0.0	**90.3** +6.4	74.2 -9.7
sokoban q1	40	82.5	**100.0** +17.5	97.5 +15.0	97.5	**100.0** +2.5	**100.0** +2.5	97.5 0.0	**100.0** +2.5
sokoban q2	20	95.0	**100.0** +5.0	**100.0** +5.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
tetris q2	40	85.0	95.0 +10.0	**97.5** +12.5	80.0	**97.5** +17.5	**97.5** +17.5	**97.5** +17.5	72.5 -7.5
ultra tictactoe q6	27	51.9	70.4 +18.5	77.8 +25.9	81.5	77.8 -3.7	63.0 -18.5	**85.2** +3.7	70.4 -11.1
zuma q3	40	32.5	30.0 -2.5	37.5 +5.0	35.0	52.5 +17.5	35.0 0.0	45.0 +10.0	**57.5** +22.5
zuma q6	40	35.0	42.5 +7.5	35.0 0.0	42.5	20.0 -22.5	27.5 -15.0	30.0 -12.5	**45.0** +2.5
Dataset total	514	68.3	73.7 +5.4	75.1 +6.8	73.3	78.4 +5.1	74.9 +1.6	78.8 +5.4	**82.5** +9.1

Table 18: **GameQA-17: Gemini 3.1 Flash-Lite.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
jewel2 q5	40	**72.5**	62.5 -10.0	32.5 -40.0	30.0	57.5 +27.5	52.5 +22.5	70.0 +40.0	65.0 +35.0
jewel2 q6	20	20.0	35.0 +15.0	30.0 +10.0	45.0	**50.0** +5.0	45.0 0.0	40.0 -5.0	**50.0** +5.0
maze q4	35	20.0	2.9 -17.1	25.7 +5.7	14.3	14.3 0.0	8.6 -5.7	28.6 +14.3	**100.0** +85.7
minesweeper q5	40	50.0	82.5 +32.5	75.0 +25.0	55.0	70.0 +15.0	62.5 +7.5	72.5 +17.5	**85.0** +30.0
pacman q1	20	10.0	10.0 0.0	5.0 -5.0	10.0	20.0 +10.0	30.0 +20.0	45.0 +35.0	**95.0** +85.0
pacman q3	20	10.0	15.0 +5.0	30.0 +20.0	10.0	15.0 +5.0	15.0 +5.0	15.0 +5.0	**100.0** +90.0
pacman q4	20	55.0	**65.0** +10.0	50.0 -5.0	55.0	50.0 -5.0	50.0 -5.0	40.0 -15.0	45.0 -10.0
rhythm game q2	27	22.2	18.5 -3.7	11.1 -11.1	37.0	37.0 0.0	40.7 +3.7	40.7 +3.7	**100.0** +63.0
rhythm game q7	27	29.6	40.7 +11.1	33.3 +3.7	37.0	63.0 +25.9	51.8 +14.8	51.8 +14.8	**92.6** +55.6

Continued on next page

Table 18 (continued)

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
rubiks cube q3	27	**44.4**	37.0 -7.4	14.8 -29.6	37.0	40.7 +3.7	18.5 -18.5	22.2 -14.8	**44.4** +7.4
rubiks cube q4	31	12.9	32.3 +19.4	9.7 -3.2	19.4	29.0 +9.7	22.6 +3.2	**48.4** +29.0	19.4 0.0
sokoban q1	40	30.0	77.5 +47.5	55.0 +25.0	42.5	77.5 +35.0	77.5 +35.0	87.5 +45.0	**97.5** +55.0
sokoban q2	20	25.0	90.0 +65.0	30.0 +5.0	30.0	70.0 +40.0	85.0 +55.0	**100.0** +70.0	45.0 +15.0
tetris q2	40	22.5	70.0 +47.5	62.5 +40.0	25.0	72.5 +47.5	50.0 +25.0	**80.0** +55.0	70.0 +45.0
ultra tictactoe q6	27	3.7	22.2 +18.5	11.1 +7.4	18.5	**25.9** +7.4	14.8 -3.7	18.5 0.0	22.2 +3.7
zuma q3	40	27.5	10.0 -17.5	22.5 -5.0	35.0	30.0 -5.0	37.5 +2.5	40.0 +5.0	**57.5** +22.5
zuma q6	40	10.0	7.5 -2.5	20.0 +10.0	12.5	**22.5** +10.0	**22.5** +10.0	17.5 +5.0	20.0 +7.5
Dataset total	514	28.6	40.9 +12.3	32.5 +3.9	30.4	45.1 +14.8	40.9 +10.5	49.8 +19.5	**65.4** +35.0

Table 19: **GameQA-17: DeepSeek and GPT with frozen FL routes.** Complete-family accuracy (%).

Family	n	DS Tool	DS VC	GPT Tool	GPT VC
jewel2 q5	40	55.0	**65.0** +10.0	60.0	**65.0** +5.0
jewel2 q6	20	**60.0**	**60.0** 0.0	**35.0**	20.0 -15.0
maze q4	35	97.1	**100.0** +2.9	34.3	**100.0** +65.7
minesweeper q5	40	70.0	**85.0** +15.0	35.0	**72.5** +37.5
pacman q1	20	**95.0**	**95.0** 0.0	5.0	**95.0** +90.0
pacman q3	20	70.0	**100.0** +30.0	0.0	**100.0** +100.0
pacman q4	20	70.0	**75.0** +5.0	10.0	**15.0** +5.0
rhythm game q2	27	**100.0**	**100.0** 0.0	66.7	**100.0** +33.3
rhythm game q7	27	63.0	**92.6** +29.6	25.9	**92.6** +66.7
rubiks cube q3	27	**48.1**	14.8 -33.3	**29.6**	25.9 -3.7
rubiks cube q4	31	25.8	**35.5** +9.7	35.5	**41.9** +6.5
sokoban q1	40	40.0	**97.5** +57.5	25.0	**97.5** +72.5
sokoban q2	20	**50.0**	**50.0** 0.0	**15.0**	5.0 -10.0
tetris q2	40	52.5	**70.0** +17.5	67.5	**70.0** +2.5
ultra tictactoe q6	27	**74.1**	70.4 -3.7	0.0	**11.1** +11.1
zuma q3	40	**85.0**	57.5 -27.5	27.5	**57.5** +30.0
zuma q6	40	7.5	**17.5** +10.0	**12.5**	**12.5** 0.0
Dataset total	514	60.7	**68.9** +8.2	31.1	**59.7** +28.6

Table 20: **PuzzleVQA: Gemini 3.1 Pro.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
circle size number	67	70.1	95.5 +25.4	**100.0** +29.9	77.6	**100.0** +22.4	88.1 +10.4	**100.0** +22.4	**100.0** +22.4
color grid	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
color hexagon	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
color number hexagon	67	92.5	**100.0** +7.5	97.0 +4.5	91.0	**100.0** +9.0	**100.0** +9.0	**100.0** +9.0	**100.0** +9.0
color overlap squares	67	98.5	**100.0** +1.5	**100.0** +1.5	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
color size circle	66	98.5	**100.0** +1.5	**100.0** +1.5	89.4	98.5 +9.1	87.9 -1.5	98.5 +9.1	**100.0** +10.6
grid number	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	98.5 -1.5
grid number color	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
polygon sides color	67	91.0	**100.0** +9.0	98.5 +7.5	95.5	98.5 +3.0	97.0 +1.5	**100.0** +4.5	97.0 +1.5
polygon sides number	67	**100.0**	98.5 -1.5	**100.0** 0.0	98.5	**100.0** +1.5	**100.0** +1.5	**100.0** +1.5	**100.0** +1.5
rectangle height color	67	86.6	98.5 +11.9	98.5 +11.9	88.1	92.5 +4.5	**100.0** +11.9	**100.0** +11.9	**100.0** +11.9
rectangle height number	66	93.9	**100.0** +6.1	97.0 +3.0	98.5	95.5 -3.0	97.0 -1.5	97.0 -1.5	98.5 0.0
shape morph	67	68.7	68.7 0.0	55.2 -13.4	62.7	53.7 -9.0	49.2 -13.4	56.7 -6.0	**80.6** +17.9
shape reflect	67	86.6	**100.0** +13.4	**100.0** +13.4	85.1	**100.0** +14.9	97.0 +11.9	**100.0** +14.9	**100.0** +14.9
shape size grid	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
shape size hexagon	67	95.5	**100.0** +4.5	97.0 +1.5	94.0	**100.0** +6.0	**100.0** +6.0	**100.0** +6.0	**100.0** +6.0
size cycle	67	92.5	95.5 +3.0	94.0 +1.5	98.5	**100.0** +1.5	**100.0** +1.5	**100.0** +1.5	**100.0** +1.5
size grid	66	97.0	**100.0** +3.0	98.5 +1.5	97.0	**100.0** +3.0	**100.0** +3.0	**100.0** +3.0	**100.0** +3.0
triangle	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
venn	67	**100.0**	**100.0** 0.0	98.5 -1.5	98.5	**100.0** +1.5	98.5 0.0	**100.0** +1.5	97.0 -1.5
Dataset total	1337	93.6	97.8 +4.3	96.7 +3.1	93.7	96.9 +3.2	95.7 +2.0	97.6 +3.9	**98.6** +4.9

Table 21: **PuzzleVQA: Gemini 3.1 Flash-Lite.** Complete-family accuracy (%).

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
circle size number	67	74.6	**100.0** +25.4	**100.0** +25.4	71.6	92.5 +20.9	71.6 0.0	**100.0** +28.4	**100.0** +28.4
color grid	67	79.1	92.5 +13.4	79.1 0.0	95.5	**100.0** +4.5	97.0 +1.5	**100.0** +4.5	**100.0** +4.5
color hexagon	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
color number hexagon	67	88.1	91.0 +3.0	86.6 -1.5	91.0	98.5 +7.5	95.5 +4.5	**100.0** +9.0	**100.0** +9.0
color overlap squares	67	82.1	**100.0** +17.9	**100.0** +17.9	91.0	**100.0** +9.0	98.5 +7.5	**100.0** +9.0	**100.0** +9.0
color size circle	66	50.0	**100.0** +50.0	89.4 +39.4	72.7	95.5 +22.7	69.7 -3.0	81.8 +9.1	**100.0** +27.3
grid number	67	95.5	98.5 +3.0	95.5 0.0	97.0	**100.0** +3.0	95.5 -1.5	97.0 0.0	98.5 +1.5

Continued on next page

Table 21 (continued)

Family	n	Dir.	GEPA	TG	Tool	XS	T2S	SO	VC
grid number color	67	95.5	**100.0** +4.5	92.5 -3.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
polygon sides color	67	80.6	88.1 +7.5	91.0 +10.4	92.5	98.5 +6.0	95.5 +3.0	97.0 +4.5	**100.0** +7.5
polygon sides number	67	94.0	94.0 0.0	92.5 -1.5	94.0	92.5 -1.5	**95.5** +1.5	**95.5** +1.5	94.0 0.0
rectangle height color	67	55.2	65.7 +10.4	80.6 +25.4	46.3	61.2 +14.9	64.2 +17.9	95.5 +49.3	**100.0** +53.7
rectangle height number	66	48.5	93.9 +45.5	47.0 -1.5	59.1	97.0 +37.9	87.9 +28.8	89.4 +30.3	**100.0** +40.9
shape morph	67	65.7	67.2 +1.5	55.2 -10.4	59.7	59.7 0.0	61.2 +1.5	65.7 +6.0	**83.6** +23.9
shape reflect	67	98.5	98.5 0.0	**100.0** +1.5	97.0	**100.0** +3.0	95.5 -1.5	**100.0** +3.0	**100.0** +3.0
shape size grid	67	98.5	98.5 0.0	**100.0** +1.5	98.5	**100.0** +1.5	98.5 0.0	**100.0** +1.5	**100.0** +1.5
shape size hexagon	67	65.7	97.0 +31.3	68.7 +3.0	77.6	98.5 +20.9	77.6 0.0	92.5 +14.9	**100.0** +22.4
size cycle	67	58.2	77.6 +19.4	67.2 +9.0	64.2	92.5 +28.4	74.6 +10.5	74.6 +10.5	**100.0** +35.8
size grid	66	56.1	72.7 +16.7	59.1 +3.0	77.3	72.7 -4.5	74.2 -3.0	69.7 -7.6	**100.0** +22.7
triangle	67	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0**	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0	**100.0** 0.0
venn	67	89.6	**100.0** +10.4	82.1 -7.5	82.1	**100.0** +17.9	85.1 +3.0	**100.0** +17.9	**100.0** +17.9
Dataset total	1337	78.8	91.8 +12.9	84.4 +5.5	83.4	93.0 +9.6	86.9 +3.5	93.0 +9.6	**98.8** +15.4

Table 22: **PuzzleVQA: DeepSeek and GPT with frozen FL routes.** Complete-family accuracy (%).

Family	n	DS Tool	DS VC	GPT Tool	GPT VC
circle size number	67	74.6	**100.0** +25.4	29.9	**100.0** +70.1
color grid	67	95.5	**100.0** +4.5	52.2	**100.0** +47.8
color hexagon	67	**100.0**	98.5 -1.5	17.9	**38.8** +20.9
color number hexagon	67	91.0	**98.5** +7.5	37.3	**97.0** +59.7
color overlap squares	67	82.1	**88.1** +6.0	14.9	**37.3** +22.4
color size circle	66	90.9	**100.0** +9.1	47.0	**100.0** +53.0
grid number	67	95.5	**98.5** +3.0	62.7	**98.5** +35.8
grid number color	67	**100.0**	**100.0** 0.0	94.0	**97.0** +3.0
polygon sides color	67	94.0	**100.0** +6.0	46.3	**100.0** +53.7
polygon sides number	67	**100.0**	94.0 -6.0	91.0	**94.0** +3.0
rectangle height color	67	70.1	**100.0** +29.9	61.2	**100.0** +38.8
rectangle height number	66	93.9	**100.0** +6.1	36.4	**100.0** +63.6
shape morph	67	58.2	**79.1** +20.9	56.7	**86.6** +29.9
shape reflect	67	97.0	**100.0** +3.0	80.6	**100.0** +19.4
shape size grid	67	**100.0**	**100.0** 0.0	77.6	**82.1** +4.5

Continued on next page

Table 22 (continued)

Family	n	DS Tool	DS VC	GPT Tool	GPT VC
shape size hexagon	67	86.6	**100.0** +13.4	52.2	**100.0** +47.8
size cycle	67	86.6	**100.0** +13.4	31.3	**100.0** +68.7
size grid	66	81.8	**100.0** +18.2	36.4	**100.0** +63.6
triangle	67	**100.0**	**100.0** 0.0	**86.6**	85.1 -1.5
venn	67	98.5	**100.0** +1.5	65.7	**100.0** +34.3
Dataset total	1337	89.8	**97.8** +8.0	53.9	**90.8** +36.9

D.2 Complete-Family Ablation Scores

The following tables expand Table 3 over all 12 TIR and 20 MIRA families, including unaffected families. $*$ marks the five re-acquired families. Other families retain the frozen full policy. The acquisition and deployment comparisons use different artifacts. Deployment TEXT/CODE/INJECT replace only the affected program takeovers of the full policy; all other outputs stay fixed. SCALAR supplements the main acquisition comparison with aggregate feedback. The no-text-fallback condition removes the trajectory-derived text fallback. Both benchmarks use the same fixed single-rollout Tool and fallback records as the main comparison. All acquisition entries report mean accuracy over three independent acquisition runs for every re-acquired family. Dataset totals weight these family scores by the number of questions.

Table 23: **TIR-Bench: complete-family acquisition ablations.** Accuracy (%).

Family	n	Tool	FULLC	IID	SCALAR	No scope
color	67	74.6	70.1	70.1	70.1	70.1
contrast	34	64.7	52.9	52.9	52.9	52.9
instrument	54	24.1	25.9	25.9	25.9	25.9
jigsaw*	80	5.0	90.4	75.0	47.1	81.2
math	80	77.5	81.2	81.2	81.2	81.2
maze*	80	90.0	96.7	97.1	90.4	43.8
ocr	40	55.0	87.5	87.5	87.5	87.5
refcoco*	80	47.5	52.1	47.5	47.5	54.2
rotation game	50	92.0	100.0	100.0	100.0	100.0
symbolic*	34	88.2	92.2	88.2	88.2	85.3
visual search	80	70.0	70.0	70.0	70.0	70.0
word search*	67	80.6	87.6	86.1	85.1	78.1
Dataset total	746	62.9	75.9	73.5	69.7	68.3

Table 24: **TIR-Bench: complete-family deployment ablations.** Accuracy (%).

Family	n	Tool	Full	TEXT	CODE	INJECT	No text fallback
color	67	74.6	70.1	70.1	70.1	70.1	74.6
contrast	34	64.7	52.9	52.9	52.9	52.9	64.7
instrument	54	24.1	25.9	25.9	25.9	25.9	24.1
jigsaw	80	5.0	95.0	33.8	72.5	92.5	95.0
math	80	77.5	81.2	81.2	81.2	81.2	77.5
maze	80	90.0	100.0	97.5	100.0	100.0	100.0
ocr	40	55.0	87.5	87.5	87.5	82.5	87.5
refcoco	80	47.5	55.0	55.0	55.0	55.0	47.5
rotation game	50	92.0	100.0	100.0	90.0	100.0	100.0
symbolic	34	88.2	91.2	91.2	91.2	85.3	91.2

Continued on next page

Table 24 (continued)

Family	n	Tool	Full	TEXT	CODE	INJECT	No text fallback
visual search	80	70.0	70.0	70.0	70.0	70.0	70.0
word search	67	80.6	79.1	79.1	79.1	79.1	80.6
Dataset total	746	62.9	76.3	69.4	73.2	75.5	76.0

Table 25: **MIRA: complete-family acquisition ablations.** Accuracy (%).

Family	n	Tool	FULLC	IID	SCALAR	No scope
billiards*	14	35.7	47.6	42.9	61.9	50.0
convex hull	20	90.0	90.0	90.0	90.0	90.0
cubes count	20	5.0	5.0	5.0	5.0	5.0
cubes missing	20	5.0	0.0	0.0	0.0	0.0
defuse a bomb	17	23.5	17.6	17.6	17.6	17.6
electric charge	14	78.6	85.7	85.7	85.7	85.7
gear rotation	26	96.2	100.0	100.0	100.0	100.0
localizer	16	0.0	0.0	0.0	0.0	0.0
mirror clock*	20	75.0	88.3	83.3	80.0	90.0
mirror pattern*	20	20.0	98.3	96.7	95.0	25.0
multi piece puzzle	20	5.0	5.0	5.0	5.0	5.0
overlap	20	65.0	60.0	60.0	60.0	60.0
paper airplane	17	17.6	29.4	29.4	29.4	29.4
puzzle*	17	52.9	52.9	52.9	52.9	13.7
rolling dice sum	17	5.9	58.8	58.8	58.8	58.8
rolling dice top	17	29.4	47.1	47.1	47.1	47.1
rolling dice two	17	0.0	5.9	5.9	5.9	5.9
trailer cubes count*	17	70.6	70.6	70.6	70.6	94.1
trailer cubes missing	17	0.0	0.0	0.0	0.0	0.0
unfolded cube	17	41.2	41.2	41.2	41.2	41.2
Dataset total	363	37.2	46.6	46.0	46.5	42.0

Table 26: **MIRA: complete-family deployment ablations.** Accuracy (%).

Family	n	Tool	Full	TEXT	CODE	INJECT	No text fallback
billiards	14	35.7	71.4	57.1	57.1	71.4	71.4
convex hull	20	90.0	90.0	90.0	90.0	90.0	90.0
cubes count	20	5.0	5.0	5.0	5.0	5.0	5.0
cubes missing	20	5.0	0.0	0.0	0.0	0.0	5.0
defuse a bomb	17	23.5	17.6	17.6	17.6	17.6	23.5
electric charge	14	78.6	85.7	85.7	85.7	85.7	78.6
gear rotation	26	96.2	100.0	100.0	100.0	100.0	96.2
localizer	16	0.0	0.0	0.0	0.0	0.0	0.0
mirror clock	20	75.0	100.0	80.0	70.0	80.0	100.0
mirror pattern	20	20.0	100.0	80.0	90.0	85.0	100.0
multi piece puzzle	20	5.0	5.0	5.0	5.0	5.0	5.0
overlap	20	65.0	60.0	60.0	60.0	60.0	65.0
paper airplane	17	17.6	29.4	29.4	29.4	29.4	17.6
puzzle	17	52.9	23.5	23.5	35.3	29.4	23.5
rolling dice sum	17	5.9	58.8	58.8	58.8	58.8	58.8
rolling dice top	17	29.4	47.1	47.1	47.1	47.1	47.1
rolling dice two	17	0.0	5.9	5.9	5.9	5.9	0.0
trailer cubes count	17	70.6	76.5	76.5	76.5	76.5	70.6
trailer cubes missing	17	0.0	0.0	0.0	0.0	0.0	0.0
unfolded cube	17	41.2	41.2	41.2	41.2	41.2	41.2
Dataset total	363	37.2	47.1	44.4	44.9	45.5	46.3

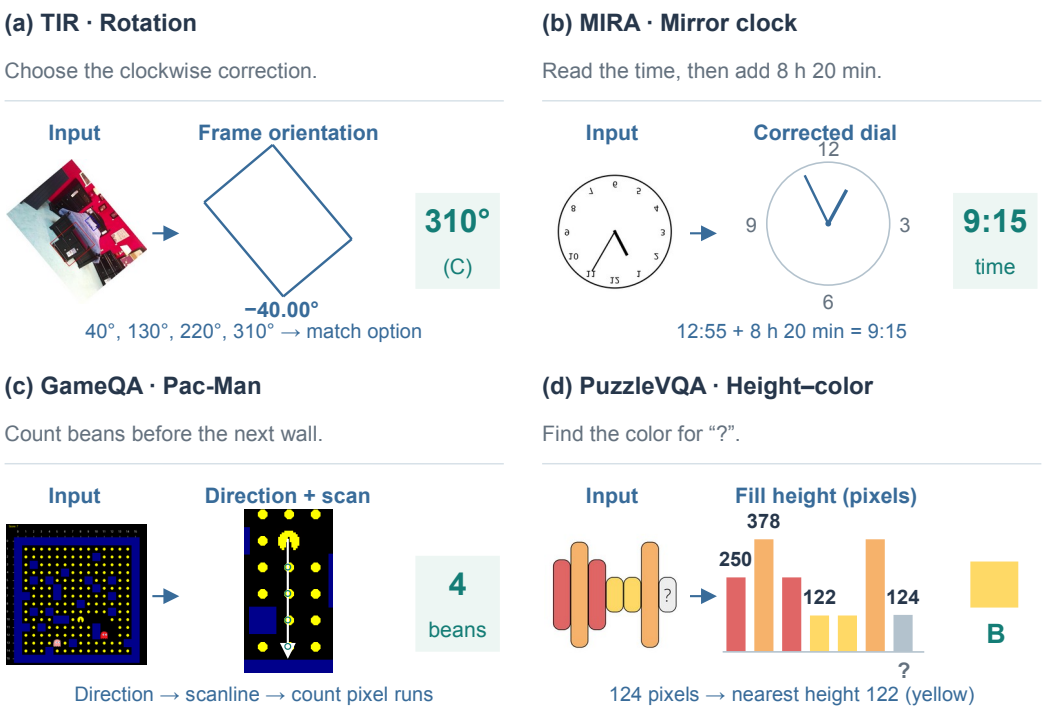

Figure 5: **Additional executable procedures.** (a) Minimum-area frame orientation yields four angle candidates; the answer adapter matches the provided options to select C (310° clockwise). (b) Measured hand directions, mirror/dial correction, and time arithmetic yield 9:15; the central dial depicts corrected angles schematically. (c) Centroid offset determines Pac-Man's direction; the enlarged scanline counts four beans before a wall. (d) Component fill heights map the query bar to yellow (B). All four examples use programs without model calls.

Related protocols and sampling results. Acquisition and deployment protocols appear in Appendices C.5 and C.6; Table 7 reports all twelve TIR family sampling scores under their separate four-attempt sampling protocol.

D.3 Illustrative Program Executions

Figure 3 in the main text shows one selected successful execution per dataset. Figure 5 adds one complementary operation per dataset: TIR rotation, MIRA mirror-clock reading, GameQA Pac-Man scanning, and PuzzleVQA height–color lookup. All four additional programs make no model calls.

D.3.1 From Skill Guidance to Executed Decisions

Completing the suggested computation. On the illustrated jigsaw instance, XSkill computes masked boundary MSE and prints the three best right and bottom neighbors for each piece. It then specifies complete layouts directly in plotting calls, without a layout optimizer. Trace2Skill first encounters a boundary-array shape mismatch, resizes the pieces, and prints 30 candidate edge pairs. Its remaining calls likewise plot manually specified layouts. SkillOpt's six calls extract and plot pieces but never compute the pairwise affinities or assembly search described in its skill. The three final answers match their last proposed layouts and are incorrect. These traces locate failures before a correct complete reconstruction has been obtained. In the maze case, Trace2Skill and SkillOpt use no tools and report six turns along a route that crosses an obstacle. XSkill instead extracts the grid, searches direction states, and verifies a path with 17 direction segments, yielding the correct 16 turns.

Selecting a measurable relation. SkillOpt's mirror-pattern skill correctly invokes chirality, but its tool-free response selects A from an incorrect reading of the petal sequence. XSkill and Trace2Skill compute SIFT correspondences and affine transformations, identifying C through a negative determinant. The acquired program uses reflected-minus-original ORB match counts of -26, -34, $+20$, and -24 for A–D, respectively, and returns C. On the area–number case, Trace2Skill retains both quantitative size comparison and opposite-number arithmetic as possible strategies, but executes no measurement and extrapolates an opposite-pair sum of eight to the queried pair, incorrectly answering two. XSkill and SkillOpt measure circle radii and correctly answer A. Our program fixes the area relation: the queried component has 16,521 foreground pixels, versus 16,467 for the component labeled six. One model call transcribes characters, and numerical matching returns six (A). These contrasts distinguish knowing a relevant principle from selecting and implementing the right test on an input.

Connection to the deployment comparisons. On jigsaw, Gemini-3.1-Pro family accuracy is 2.5% for XSkill, 7.5% for Trace2Skill, 10.0% for SkillOpt, and 95.0% for VISUALCODING (Table 11). Holding the acquired program fixed, textual description, callable code, and direct execution yield 33.8%, 72.5%, and 95.0%, respectively. On mirror patterns, the corresponding scores are 80%, 90%, and 100%; reading injection yields 85% on the same 20 questions. These controls complement the selected traces by varying how the same acquired experience is deployed. For the area–number family, XSkill, SkillOpt, and TextGrad also attain 100% with Gemini-3.1-Pro.

D.3.2 TRAINING FEEDBACK AND RETAINED IMPLEMENTATIONS

The solver in Figure 3(a) is the frozen sixth-round candidate from the acquisition run used by the main experiments. Table 27 traces selected candidates on the same 40 training puzzles. The initial program already used boundary continuity, excluded overlaid number regions from edge comparisons, and assembled pieces with spatial constraints. Failure feedback led the second candidate to try Prim-style assembly from every starting piece, recovering outputs where the first search became stuck. The sixth candidate generated assemblies from several color and gradient cost variants and fixed-seed perturbations, then ranked unique complete grids by the root-mean-square of their boundary costs. This ranks the available assemblies under a shared criterion and increases the penalty for a large discontinuity. It is a reusable heuristic for this task family.

Table 27: **Selected candidates in the deployed jigsaw solver's acquisition history.** Correct counts use all 40 training puzzles before scope filtering; missing outputs count as incorrect. These are candidates within one historical acquisition run, separate from the controlled acquisition ablations.

Round	Revision	Correct / 40
1	Boundary matching with constrained Kruskal/Prim-style assembly.	35
2	Run Prim-style assembly from every starting piece and compare complete grids.	37
4	Jointly trim two boundary pixels and replace RMSE costs with MSE; rejected after regression.	18
6	Combine multiple cost variants and fixed-seed perturbations; rank complete grids by RMS boundary cost.	39

The first candidate produced 35 valid answers, all correct; the second produced 40, of which 37 were correct. The fourth candidate's joint preprocessing and scoring change reduced correctness to 18/40 and was rejected. The sixth retained all 35 initially correct answers and solved four additional puzzles. Later candidates did not improve the selection score, so the sixth was frozen. The resulting program uses no model calls and answers 76/80 evaluation puzzles correctly. This trajectory illustrates refinement of an executable procedure through cross-instance feedback, including detection of regressions.