NAMES THAT STEER: IDENTIFIER BIAS ACROSS TEXT AND VISION IN CODE UNDERSTANDING

Anonymous authors
Paper under double-blind review

ABSTRACT

Code is increasingly presented to multimodal models as rendered images, yet little is known about whether visual presentation changes how symbolic cues shape program understanding. We trace identifier bias across textual and visual code using a semantics-preserving source–donor construction that holds a source implementation fixed while replacing controlled identifiers with names from a functionally different donor. Across eight vision-capable model instances, donor-aligned names shift both functional choices and open-ended summaries toward the donor under Text and Readable Image. The Readable Image choice effect is positive in every instance (4.2–22.8 percentage points) and exceeds its Text counterpart in all eight, although carrier differences vary by task and readout. Readback and constrained visual controls make gross transcription failure insufficient to explain this persistence. In an open-weight focal model, we further examine carrier-conditioned output likelihood, cross-carrier representation geometry, and activation patching. Text and Image yield strongly corresponding representations, and identifier-induced changes align more closely for the same program than across programs. A matched Text direction steers Image judgments more than a mismatched direction, while a same-carrier Image direction remains stronger. Similar generated summaries nevertheless retain a self-carrier likelihood advantage. Together, these findings are consistent with a partial-anchoring account in which visual-code representations share text-compatible, program-specific components while retaining carrier-conditioned structure. Identifier bias therefore persists beyond textual input, and text-derived identifier directions can functionally steer image judgments.

1 INTRODUCTION

Source code increasingly reaches vision-language models (VLMs) through IDEs, documents, and screenshots. Recent work also renders programs as images to extend context coverage or reduce input cost (Shi et al., 2026; Zhong et al., 2026; Tang et al., 2026). These studies establish usability, recognition, compression, and efficiency, but say less about which semantic evidence a model retains and uses when the same program is presented as text or image. We call these input formats *carriers*. A visual carrier jointly presents glyphs, indentation, spatial proximity, and layout, potentially changing how lexical and structural evidence are combined.

Shared alignment and carrier-specific processing can coexist: visual code may align with text-compatible representations while retaining image-conditioned information through answer formation. Here, *text-compatible* means correspondence with states elicited by textual code, not membership in an independently identified text-token subspace. Prior work localizes visual-text pathways in language backbones (Steinberg & Gal, 2026) and separates text-aligned from residual visual components (Ou et al., 2026), but leaves open whether code alignment is content-specific and whether text-derived representations can affect image judgments. Answering these questions requires a paired intervention that changes interpretable semantic evidence while preserving program meaning, followed from behavior to representation and functional intervention.

Identifiers provide such a probe. Names are informative in statistically regular software (Hindle et al., 2012), yet code models can overweight them when they conflict with implementation evidence (Gao et al., 2023). We use the source–donor construction in Shen et al. (2026): Original (O)

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.

retains source-aligned identifiers, Confusing (C) replaces controlled identifiers with names from a functionally different donor, and Indexed Anonymous (A) replaces them with indexed placeholders. O-to-C predicts movement from source toward donor if names gain weight, and this direction can be followed through behavior, representation, and activation steering.

We construct repository-aware pairs from CodeSearchNet Python functions (Husain et al., 2019) and test Text and Readable Image (hereafter Image) using choice, one-sentence summary, and readback. Behavioral analysis covers eight model instances; internal analysis in Qwen3-VL-8B-Instruct compares program geometry, identifier-change directions, and norm-matched activation steering across carriers. A result is *content-matched* when same-program correspondence exceeds its mismatched-program control. We ask: **RQ1**, do semantics-preserving identifier conflicts continue to affect functional judgments under visual presentation, and how does their expression vary across tasks and outputs? **RQ2**, do text and image form content-matched representations of programs and identifier changes through answer formation? **RQ3**, can norm-matched, content-matched text-derived directions steer image judgments beyond mismatched controls?

The results form a three-step chain. Donor-directed influence persists in readable images across all eight instances, with choice effects of 4.2–22.8 percentage points; carrier differences vary by task and readout. The focal model shows content-matched program geometry and identifier-change alignment through answer start. A same-program Text direction also steers Image judgments more than a mismatched direction ($+.783$ raw logit, 95% CI $[.215, 1.352]$), while same-carrier and output-support controls show that the carriers do not become interchangeable.

We contribute a cross-carrier diagnostic that traces a controlled symbolic conflict from behavior to representation and intervention; evidence that identifier influence persists in visual code across eight model instances; and content-matched representation and steering evidence in an open-weight VLM. We call the resulting working account *partial anchoring with carrier retention*: visual code aligns with text-compatible, content-matched representations while answer formation remains carrier-conditioned.

2 Related Work

2.1 Visual Text and Code Processing

Pixel models process rendered language without a fixed word-piece vocabulary (Rust et al., 2023), and screenshot models extend visual language understanding to structured documents and interfaces (Lee et al., 2023; Gao et al., 2024). Visual-text compression uses the same capability to cover longer inputs (Li et al., 2025; Cheng et al., 2026), although rendering and fidelity affect representations and verbatim tasks more than higher-level judgments (Lotz et al., 2023; Tu et al., 2026). Visual code likewise ranges from AST images and codepoint grids (Bilgin, 2021; Shi et al., 2022) to readable screenshots (Rabin & Alipour, 2022). Recent systems feed rendered code to general-purpose VLMs (Shi et al., 2026) and optimize the trade-off between context coverage and symbol fidelity (Zhong et al., 2026; Tang et al., 2026). They establish usability and efficiency, but rarely hold executable semantics fixed while directionally changing one family of symbolic evidence.

2.2 Cross-Modal Alignment and Cue Conflict

Studies of visual text identify intervenable OCR-associated heads (Baek et al., 2025), routing bottlenecks in language backbones (Steinberg & Gal, 2026), and a progression from surface features to linguistic abstractions (Tatariya et al., 2024). Projection-based analyses also separate components absorbed by a text-token subspace from residual visual components (Ou et al., 2026). When pixels are weak or conflict with linguistic priors, VLMs can produce plausible but visually unsupported text (Shu et al., 2025), echoing shortcut-based diagnostic findings (Geirhos et al., 2020; McCoy et al., 2019). In code, lexical recovery likewise need not imply semantic understanding (Štorek et al., 2026). This literature does not yet connect content-matched cross-carrier representations to functional evidence in code.

2.3 Lexical Cues in Code Understanding

Identifiers are useful cues in statistically regular software (Hindle et al., 2012), but can also mislead functional judgments (Gao et al., 2023) and code generation (Yang et al., 2024). Low-readability inputs expose surface dependence (Hu et al., 2024), while semantics-preserving obfuscations separate naming cues from structural evidence (Le et al., 2025). A source–donor construction supplies a directional conflict between names and implementation (Shen et al., 2026). TokDrift traces some effects of textual rewrites to early embeddings and subword–grammar mismatches (Li et al., 2026); this identifies a text-side mechanism without determining whether visual code later aligns with text-compatible states. We examine whether identifier influence persists beyond direct text-token input, how identifier-induced changes correspond across carriers, and whether text-derived directions transfer functionally to image judgments.

3 Experimental Setup and Methodology

We use a source–donor identifier conflict as a cross-carrier semantic probe: we first measure the effects of the same controlled renaming under Text and Readable Image, then examine cross-carrier representation correspondence and functional transfer in one open-weight focal model. Behavioral, output-level, geometric, and interventional analyses retain separate estimands and are interpreted jointly in Section 4.

3.1 Source–Donor Design and Behavioral Measures

(a) Source–donor construction and identifier arms Each unit pairs a **source function** with a functionally different **donor**. The source provides the fixed implementation, source reference, and original identifiers; the donor provides the contrasting reference and names assigned to eligible local bindings by role and order. Donor code never enters the input. Binding-aware replacement updates definitions and uses together and passes binding, syntax, and executability audits. *Semantics-preserving* means unchanged executable meaning, control flow, and data flow; surface form and uncontrolled external names may differ.

Three arms follow: **Original (O)**, source-aligned; **Indexed Anonymous (A)**, using role-based placeholders such as `func0` and `var0`; and **Confusing (C)**, using donor-aligned identifiers that conflict with the implementation. The primary comparison is $C{-}O$; $A{-}O$ and $C{-}A$ help distinguish effects associated with controlled-name replacement and donor-aligned naming. A preserves indices, repetition, and binding structure. Additional neutral-natural and nonce controls appear in Appendix F.

(b) Presentation carriers and interventions **Text** presents the code as ordinary text; **Readable Image** renders it under a fixed font, syntax highlighting, page, and layout specification, and both carriers share references, task targets, and sample pairing (Figure 1).

A **Parity Image** provides a constrained, model-specific visual boundary that jointly changes page budget, optical density, layout, and verbatim availability. **Selective salience** factorially emphasizes recurrent identifier bindings, execution-relevant structural cues, or both, testing their main effects and interaction on donor-choice susceptibility under Confusing names.

(c) Tasks and behavioral endpoints **Functional choice.** The model chooses between the source and donor descriptions; the readout is the donor-choice indicator, the naming effect the within-sample $C{-}O$ donor-choice risk difference, and the carrier interaction the image-minus-text effect.

One-sentence summary. The model writes one functional sentence. For sample i, arm a, and carrier c, we define $\mathrm{DMS}_i^{a,c} = S(y_i^{a,c}, r_{D,i}) - S(y_i^{a,c}, r_{S,i})$, where y is the summary, r_D and r_S the donor and source references, and S a frozen similarity function. We report prespecified, surface-sensitive ROUGE-L F1 contrasts (Lin, 2004) and BERTScore F1 as a contextual-semantic complement (Zhang et al., 2020). $\mathrm{DMS} > 0$ means the output is closer to the donor reference; $\mathrm{DMS}_C - \mathrm{DMS}_O > 0$ denotes a donor-directed relative shift.

Auxiliary image-only readback. The model transcribes code from the image, diagnosing visual availability and naming-related recognition differences rather than functional understanding. We report character error rate and exact match/recall over controlled identifiers.

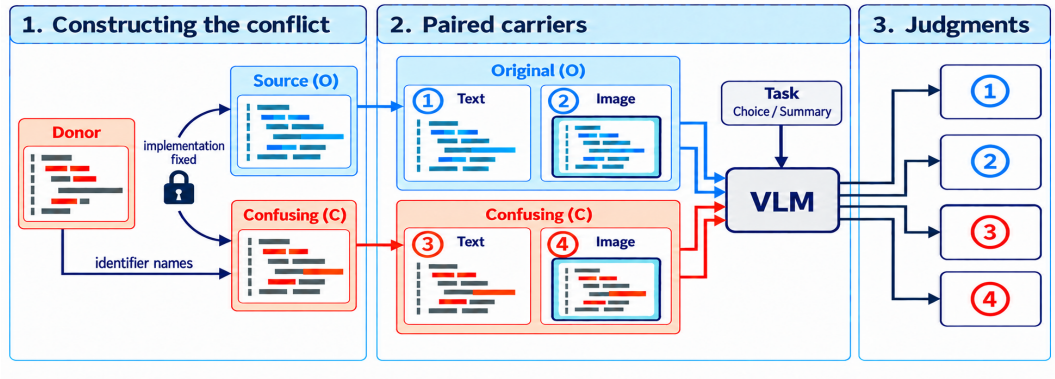


Figure 1: Source–donor construction and paired carriers. The implementation remains fixed; donor-aligned identifiers alone create Confusing (C). Original (O) and C are presented as Text or Readable Image and evaluated by choice or summary.

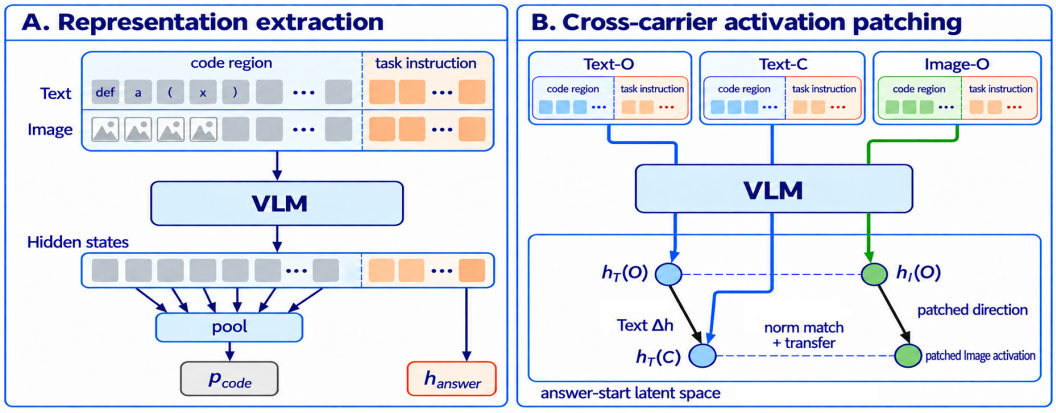


Figure 2: Representation readouts and cross-carrier patching. (A) `p_code` pools code-region states; `h_answer` is the pre-generation residual state. (B) Forward and restoration patches transfer a norm-matched Text $C-O$ direction to Image, with equal-dose Random, Mismatched Text, Matched Text, and Same-carrier Image controls.

3.2 Cross-Carrier Representation and Intervention

(a) Representation extraction Internal analysis uses a fixed local Qwen3-VL-8B-Instruct stack (Bai et al., 2025). Writing r_ℓ for residual-stream depth ℓ, **`p_code`** pools the Text code-token or Image visual-content span, while **`h_answer`** is the pre-generation residual state after code and task context. Development fixed the mid-depth `p_code r16` readout; `h_answer r36` is the architectural endpoint. Both were frozen before independent evaluation (Figure 2).

(b) Program geometry and identifier-change directions Centered kernel alignment (linear CKA) measures global Text–Image matrix correspondence under O, A, and C (Kornblith et al., 2019); representational similarity analysis (RSA) compares pairwise program-distance structures (Kriegeskorte et al., 2008). Label permutations compare CKA with random pairing; RSA is descriptive. Frozen r16/r36 positions carry the primary interpretation; other depth-wise variation is descriptive.

For $a \in \{A, C\}$, the identifier-change directions of sample i are $d^a_{T,i} = h_{T,i}(a) - h_{T,i}(O)$ and $d^a_{I,i} = h_{I,i}(a) - h_{I,i}(O)$. We compare the same-program cosine between $d^a_{T,i}$ and $d^a_{I,i}$ against cross-program Text–Image directions. The primary estimand is the matched-minus-mismatched gap; constrained mismatching excludes the same repository component and probes sensitivity to length, rename coverage, and direction norm.

(c) Output-facing carrier diagnostic For each arm, G_{NLL} averages, in both directions, the reduction in token-level negative log-likelihood (NLL) when each carrier-specific greedy summary is scored under its producing carrier rather than the other carrier. Cross-summary BERTScore measures semantic closeness; the exact matrix appears in Table 15.

(d) Answer-start activation patching The formal Text patch uses the sample-level $C-O$ direction scaled to the norm of the corresponding Image direction, $\widetilde{d}_{T,i} = \rho_i d_{T,i}$ with $\rho_i = \|d_{I,i}\|_2 / \|d_{T,i}\|_2$. The receiver is an Original or Confusing Image `h_answer`: forward adds the direction to Original, and restoration subtracts it from Confusing; both yield a donor-directed logit effect. Following activation-based causal analysis in code models (Lucchetti & Guha, 2025), this content-matched intervention transfers a norm-matched Text direction into the Image answer-start state without token-level alignment between unequal-length carrier inputs.

Four equal-dose interventions are compared: **Random** (equal-norm), **Mismatched Text** (another program's direction), **Matched Text** (the same program's direction), and **Same-carrier Image** (the same program's Image direction). With $L = \mathrm{logit}(\mathrm{donor}) - \mathrm{logit}(\mathrm{source})$, ΔL is each intervention's bidirectional average change in L; the primary contrast is $\Delta L_{\mathrm{Matched}} - \Delta L_{\mathrm{Mismatched}}$. Random and Same-carrier Image provide perturbation and receiver-aligned references.

A component-disjoint development set freezes the earliest layer passing prespecified identity, repeatability, and directional-sign checks; held-out evaluation uses only that layer. The complete gate appears in Table 16.

3.3 Data, Models, and Statistical Analysis

(a) Dataset construction We use CodeSearchNet Python functions (Husain et al., 2019) passing executability, length, renderability, and controllable-binding filters; functional references guide source–donor pairing. Splits respect repository components, while source/donor options, length bins, and coverage are balanced or stratified. All inputs use frozen code, with images generated from frozen rendering plans.

(b) Models and evidence scope The O/A/C Text/Readable Image experiments cover eight fixed model instances: Qwen3-VL-4B, Qwen3-VL-8B, Qwen3-VL-32B, and Qwen3-VL-Plus (Bai et al., 2025); Kimi K2.5 (Kimi Team, 2026); GPT-5 (OpenAI, 2025); Claude Sonnet 4.5 (Anthropic, 2025); and GLM-5V-Turbo (GLM-V Team, 2026). Parity and salience use the corresponding subsets. Geometry, output likelihood, and patching use a frozen local Qwen3-VL-8B-Instruct stack.

(c) Statistical analysis Primary contrasts are paired within sample. Behavioral and patching intervals use two-way source- and donor-repository clustering (Cameron et al., 2011); salience tests use within-model Holm adjustment (Holm, 1979); CKA uses sample-label permutation; and direction gaps use paired bootstrap. Prompts and response handling appear in Appendix A; model revisions, evidence splits, matching rules, and full uncertainty procedures appear in Appendix B.

4 Results and Analysis

We examine behavioral persistence, cross-carrier representation correspondence, and functional intervention in turn. Their separate estimands ask whether identifier influence persists, whether the carriers form content-matched internal representations, and whether that correspondence can change cross-carrier judgments.

4.1 RQ1: Observable behavior

(a) Identifier semantics affect both choice and summary We first fix the orientation of the summary readout. Donor-minus-source similarity (DMS) measures whether a summary lies closer to the donor or to the source; a positive $DMS_C - DMS_O$ therefore means that Confusing moves the output toward the donor reference, even when the absolute DMS of both naming conditions still favors the source.

Figure 3A reports the donor-choice change from Original to Confusing for eight fixed model instances. Every instance shows a positive effect under both Text and Readable Image; the Readable

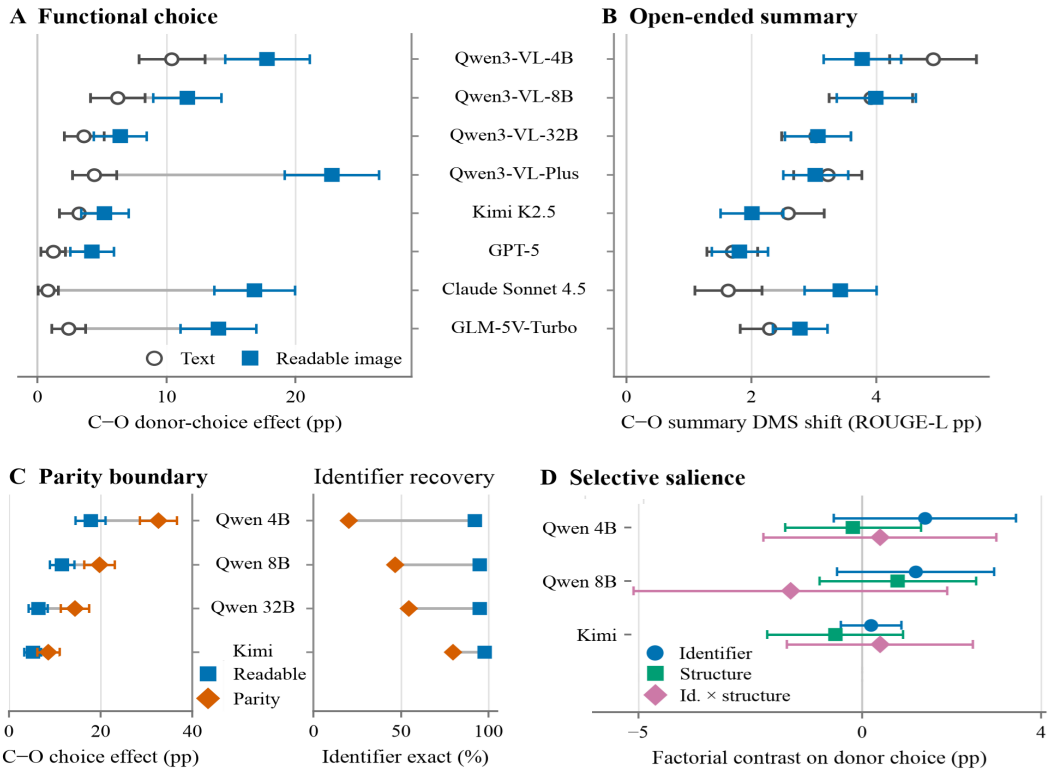


Figure 3: Identifier effects across presentation conditions. (A) $C-O$ choice effects. (B) ROUGE-L donor-minus-source summary shifts. (C) Parity recovery and choice. (D) Salience main effects and interaction. Bars are repository-clustered 95% CIs; salience tests use Holm adjustment. Main and Parity analyses use $N = 500$, salience $N = 250$, and readback nested $N = 100$.

effect ranges from +4.2 to +22.8 percentage points and exceeds the corresponding Text effect in all eight. Readable rendering therefore does not remove the identifier conflict. Because the Readable-Text difference varies across models, we treat it as a choice-task carrier difference rather than evidence of a uniform visual-amplification mechanism.

Open-ended summaries corroborate the direction. In Figure 3B, the prespecified ROUGE-L DMS shift is positive for all eight models under both carriers, ranging from +1.81 to +3.99 percentage points under Readable Image. BERTScore likewise shows lower source similarity and higher DMS in every instance, but its Readable-Text interaction is positive for only six instances. The influence of identifier conflict on open-ended output therefore persists across models, while the carrier difference depends on task and readout.

Anonymous serves mainly as a controlled-name replacement condition. Across 16 model-by-carrier cells, $A-O$ changes donor choice by at most 3 pp, lowers source-reference ROUGE-L in all 16, and raises donor-reference ROUGE-L in only three; $C-A$ raises the latter in all 16 (Table 9). This helps distinguish effects associated with controlled-name replacement from those associated with donor-aligned naming.

Image-only readback provides an auxiliary diagnostic on nominal nested N=100 samples per arm for the eight instances, with valid counts varying by instance. Across Readable cells, controlled-identifier multiset exact recovery ranges from 90.0% to 100.0% and mean CER from 0.63% to 1.86%; no Confusing-specific recovery collapse appears. These diagnostics do not support a simple account based on gross Confusing-specific transcription failure. Complete O/A/C behavior and readback cells appear in Tables 7 and 8.

(b) The effect persists as verbatim recovery declines Parity Image provides a joint lower-fidelity boundary for comparing verbatim recovery and functional judgment (Figure 3C). Original donor-

Table 1: Cross-carrier similarity of realized outputs and own-carrier support.

Arm	Text–Image BERTScore F1	F1 Δ vs. mismatch	Self-carrier NLL advantage
Original	.9187	+.0739	.604
Anonymous	.9124	+.0684	.710
Confusing	.9063	+.0614	.749

Notes. F1 compares same-program Text/Image summaries; the mismatch is repository-disjoint and length-matched. NLL is two-way own-path support, not full-distribution distance. $N = 96$.

choice rates remain only 0%–0.4%. Controlled-identifier exact recovery nevertheless falls from 92.3%–98.0% to 20.0%–79.7% and CER rises in all four models, while the $C - O$ choice effect remains positive and descriptively increases from +5.2 to +17.8 pp under Readable and from +8.6 to +32.6 pp under Parity. This constrains a broad task-collapse account, although Parity jointly changes several aspects of visual fidelity rather than isolating one factor.

Selective salience reweights identifier bindings and execution-relevant structural cues (Figure 3D). For Qwen3-VL-4B, Qwen3-VL-8B, and Kimi K2.5, the Confusing effect remains positive in all four cells, but all main-effect and interaction intervals cross zero and no model-wise Holm-adjusted test rejects a contrast. At the tested dose and precision, no salience-induced shift is detected. The complete Parity and salience results appear in Tables 10 and 11.

4.2 RQ2: Cross-carrier representation correspondence

(a) Program identity and renaming form cross-carrier correspondence We evaluate the held-out data at the development-frozen `p_code r16` and at `h_answer r36`, the architectural endpoint before generation. Depth-wise variation outside these targets is descriptive, and the more complex `p_code` trajectory is discussed in Appendix D.

Figure 4A shows Text–Image centered linear CKA for N=96 programs across all residual depths. At the prespecified `p_code r16`, CKA is .945, .938, and .931 for Original, Anonymous, and Confusing; at `h_answer r36`, it is .875, .852, and .720. RSA corroborates the fixed-position pattern (.694–.928; Table 13). All six fixed-position CKAs exceed the maxima of 1,024 sample-label permutations (.238–.308; upper-tail $p = .00098$), indicating strongly corresponding same-program geometry across carriers.

State changes induced by renaming likewise show program-specific cross-carrier correspondence. Figure 4B compares Text–Image identifier-change directions for the same program and for mismatched programs at `h_answer r36`. For $A-O$, the paired cosine is .571, the mismatched cosine .234, and the difference +.337 with bootstrap 95% CI $[.315, .360]$; the corresponding values for $C-O$ are .663, .111, and +.552 $[.529, .574]$. Both gaps are positive in 96/96 samples. This correspondence appears for donor-aligned C as well as for indexed replacement A, indicating that it tracks the identifier change matched to a specific program rather than an outcome produced only by donor choice.

(b) Similar realized outputs retain carrier-conditioned support Table 1 compares same-program Text/Image summary similarity with each carrier's NLL support advantage for its own generated path; Table 15 gives the underlying scoring matrix.

Under all three naming conditions, same-program similarity exceeds a repository-component-disjoint baseline matched on code and summary length, indicating that the carriers generate content-matched text. All three self-carrier NLL advantages are also positive: the producing carrier assigns greater token-level support to its realized summary path. Cross-carrier output content can therefore be similar while its conditional support remains carrier-dependent.

Text and Image thus exhibit content-matched program geometry at the code-region and answer-start readouts and program-specific identifier-change correspondence at answer start, while output support remains carrier-dependent.

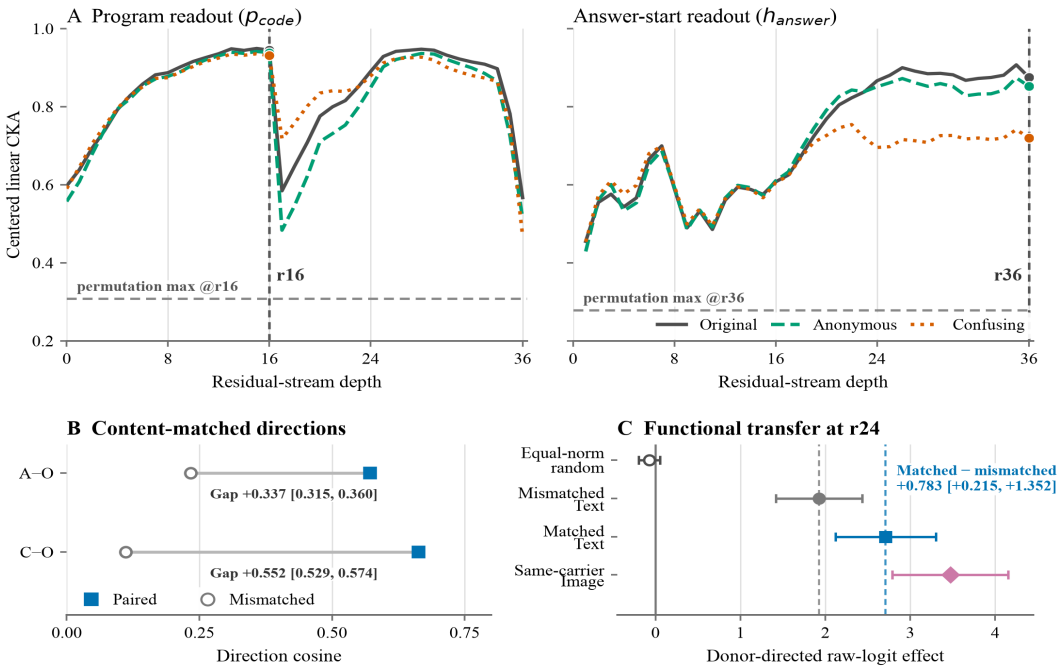


Figure 4: Cross-carrier correspondence and transfer. (A) Text–Image CKA for `p_code` and `h_answer`; markers show frozen readouts and permutation maxima. (B) Same-program and mismatched identifier-change cosines with paired gaps. (C) Equal-norm patching at held-out `h_answer r24`; dashed lines mark Mismatched and Matched Text. Geometry uses $N = 96$; development $N = 16$ and held-out patching $N = 64$, with repository-clustered 95% CIs.

4.3 RQ3: Representation Intervention

On the N=16 development set, the gate excludes r18 among the prespecified r18/r24/r30 candidates; r24 and r30 qualify, so the earliest-eligible rule fixes the held-out N=64 evaluation at `h_answer r24` rather than selecting r30 by effect size (Table 16).

Figure 4C reports bidirectional donor-directed raw-logit effects for the four equal-dose interventions. Equal-norm random is near zero (-.072 $[-.197, .053]$). Mismatched Text produces +1.926 $[1.416, 2.435]$, consistent with a cross-program shared steering component. Matched Text rises to +2.709 $[2.118, 3.301]$, and the prespecified Matched-minus-Mismatched contrast is +.783 $[.215, 1.352]$, indicating an additional content-matched increment. Same-carrier Image is strongest at +3.471 $[2.791, 4.152]$, exceeding Matched Text by +.762 $[.219, 1.305]$.

The forward increment is +.964 $[.412, 1.517]$ and restoration +.603 $[-.146, 1.351]$. Both are positive, although only the forward interval excludes zero; their prespecified equal-weight average remains primary. A same-program Text direction thus changes Image judgments more than another program's direction, while Same-carrier Image retains additional influence. The full directional decomposition and compact sample-level summary appear in Table 17.

4.4 Integrated Analysis: Partial Anchoring with Carrier Retention

We summarize the evidence with a simplified working decomposition. For carrier $m \in \{T, I\}$, let the identifier-change direction be

$$d_{m,i} = \alpha_{m,i} g + \beta_{m,i} s_i + u_{m,i}.$$

Here g captures a cross-program component motivated by mismatched steering over random; s_i is a same-program component corresponding to matched geometry and the Matched-minus-Mismatched increment; and $u_{m,i}$ retains carrier- and receiver-conditioned structure, consistent with self-carrier

NLL advantage and stronger Same-carrier Image patching. Coefficients allow shared components to vary by carrier and sample. This non-identified model accommodates functional cross-carrier correspondence without carrier invariance. We call the pattern **partial anchoring with carrier retention**: visual code aligns with text-compatible, content-matched directions while retaining carrier-conditioned structure.

5 Limitations and Threats to Validity

Scope and external validity. We study filtered CodeSearchNet Python functions and one controlled source–donor conflict. The conclusions therefore apply first to function-level code of this kind, rather than other languages, longer programs, naturally occurring naming defects, or arbitrary visual environments. Readable Images use one fixed renderer. Agreement across eight fixed model instances demonstrates breadth but does not estimate incidence in future models, and internal analyses use one open-weight VLM. Repository-separated splits and clustered uncertainty reduce leakage and corpus dependence without widening this population.

Intervention and measurement. Binding-aware renaming preserves executable behavior and local binding structure but cannot isolate name semantics from every correlated surface property: shape, length, tokenization, and layout may change, while uncontrolled attribute, method, and external-API names remain. O-to-C is therefore a controlled directional conflict. Choice targets source-versus-donor susceptibility; summary scores measure relative movement rather than program-level correctness; and readback constrains gross transcription accounts rather than measuring functional understanding. Parity jointly changes visual factors, so its descriptively larger $C-O$ effect cannot be assigned to one.

Mechanistic interpretation. Geometry establishes content-matched correspondence, not identical encodings or natural use. Equal-norm patching shows local causal sufficiency to shift answer logits, but does not identify necessity, the complete pathway, or mediation proportion. The self-carrier NLL advantage concerns realized summary paths rather than full output distributions. Accordingly, partial anchoring with carrier retention remains a working decomposition whose components are not separately identified and need not be orthogonal or globally linear. Full execution qualifications, denominators, and analysis classifications appear in Appendix G and the reproducibility materials.

6 Conclusion

Using a semantics-preserving source–donor conflict, we find that the influence of misleading identifiers persists under readable visual presentation across eight model instances, with carrier differences varying by task and readout. In the focal model, Text and Image exhibit content-matched program geometry and identifier-change correspondence, yet generated outputs retain a self-carrier support advantage. A same-program Text direction also steers Image judgments more than a cross-program direction.

Together, these findings support, without uniquely identifying, a working account of partial anchoring with carrier retention: visual code shares text-compatible, content-matched directions that can function across carriers, while answer formation remains carrier-conditioned. Robustness evaluations of visual code should therefore examine how models weight and use symbolic evidence alongside recognition and task accuracy.

AI Use Statement

In this work, we used generative AI tools for implementing methods, surveying related work, and editing the paper to improve readability. We have not used generative AI tools for generating synthetic datasets or cleaning and reformatting datasets. The remaining tasks with required disclosure are not applicable to this work. We have reviewed all AI-assisted work: AI-assisted code/method implementation was verified and tested by the authors, literature search results were checked against original sources, and AI-edited text was reviewed by all authors. We take responsibility for the final content of this work.

Reproducibility statement

Appendix A documents public-data reconstruction, source–donor pairing, binding-aware renaming, rendering, prompts, and response handling; Appendix B specifies model identities, development/held-out separation, estimands, and uncertainty procedures; and Appendices C–E report the complete estimates and sensitivity analyses supporting the three research questions. The anonymous supplementary package releases the dataset-construction and renaming tools, deterministic renderer and frozen profiles, exact prompt schemas, model-call and representation-intervention specifications, local verification records, and a complete checksum manifest. Table 21 summarizes this inventory and its redistribution boundary. Public CodeSearchNet inputs can be rebuilt with the released scripts. Provider orchestration, raw responses, model weights, hidden-state tensors, and derived result files are not redistributed; the appendix reports the corresponding aggregates and valid denominators, while the released model revisions, request settings, and capture and intervention specifications support equivalent re-execution.

References

Anthropic. Claude sonnet 4.5 system card, September 2025. URL https://assets.anthropic.com/m/12f214efcc2f457a/original/Claude-Sonnet-4-5-System-Card.pdf. Official system card.

Ingeol Baek, Hwan Chang, Sunghyun Ryu, and Hwanhee Lee. How do large vision-language models see text in image? unveiling the distinctive role of OCR heads, 2025. URL https://arxiv.org/abs/2505.15865. EMNLP 2025 Oral.

Shuai Bai et al. Qwen3-VL technical report, 2025. URL https://arxiv.org/abs/2511.21631.

Zeki Bilgin. Code2Image: Intelligent code analysis by computer vision techniques and application to vulnerability prediction, 2021. URL https://arxiv.org/abs/2105.03131.

A. Colin Cameron, Jonah B. Gelbach, and Douglas L. Miller. Robust inference with multiway clustering. *Journal of Business & Economic Statistics*, 29(2):238–249, 2011. doi: 10.1198/jbes.2010.07136. URL https://doi.org/10.1198/jbes.2010.07136.

Jiale Cheng, Yusen Liu, Xinyu Zhang, Yulin Fei, Wenyi Hong, Ruiliang Lyu, Weihan Wang, Zhe Su, Xiaotao Gu, Xiao Liu, Yushi Bai, Jie Tang, Hongning Wang, and Minlie Huang. Glyph: Scaling context windows via visual-text compression. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics*, pp. 37145–37158. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.1722. URL https://aclanthology.org/2026.acl-long.1722/.

Shuzheng Gao, Cuiyun Gao, Chaozheng Wang, Jun Sun, David Lo, and Yue Yu. Two sides of the same coin: Exploiting the impact of identifiers in neural code comprehension, 2023. URL https://arxiv.org/abs/2207.11104. Accepted to ICSE 2023.

Tianyu Gao, Zirui Wang, Adithya Bhaskar, and Danqi Chen. Improving language understanding from screenshots, 2024. URL https://arxiv.org/abs/2402.14073.

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

GLM-V Team. GLM-5V-Turbo: Toward a native foundation model for multimodal agents, 2026. URL https://arxiv.org/abs/2604.26752.

Abram Hindle, Earl T. Barr, Mark Gabel, Zhendong Su, and Premkumar T. Devanbu. On the naturalness of software. In *Proceedings of the 34th International Conference on Software Engineering*, pp. 837–847. IEEE, 2012. doi: 10.1109/ICSE.2012.6227135. URL https://doi.org/10.1109/ICSE.2012.6227135.

Sture Holm. A simple sequentially rejective multiple test procedure. *Scandinavian Journal of Statistics*, 6(2):65–70, 1979. URL https://www.jstor.org/stable/4615733.

Chao Hu, Yitian Chai, Hao Zhou, Fandong Meng, Jie Zhou, and Xiaodong Gu. How effectively do code language models understand poor-readability code? In *Proceedings of the 39th IEEE/ACM International Conference on Automated Software Engineering*, pp. 795–806. ACM, 2024. doi: 10.1145/3691620.3695072. URL https://doi.org/10.1145/3691620.3695072.

Hamel Husain, Ho-Hsiang Wu, Tiferet Gazit, Miltiadis Allamanis, and Marc Brockschmidt. CodeSearchNet challenge: Evaluating the state of semantic code search, 2019. URL https://arxiv.org/abs/1909.09436.

Kimi Team. Kimi K2.5: Visual agentic intelligence, 2026. URL https://arxiv.org/abs/2602.02276.

Simon Kornblith, Mohammad Norouzi, Honglak Lee, and Geoffrey Hinton. Similarity of neural network representations revisited. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3519–3529. PMLR, 2019. URL https://proceedings.mlr.press/v97/kornblith19a.html.

Nikolaus Kriegeskorte, Marieke Mur, and Peter A. Bandettini. Representational similarity analysis: Connecting the branches of systems neuroscience. *Frontiers in Systems Neuroscience*, 2:4, 2008. doi: 10.3389/neuro.06.004.2008. URL https://doi.org/10.3389/neuro.06.004.2008.

Cuong Chi Le, Minh V. T. Pham, Cuong Duc Van, Hoang N. Phan, Huy N. Phan, and Tien N. Nguyen. When names disappear: Revealing what LLMs actually understand about code, 2025. URL https://arxiv.org/abs/2510.03178. Preprint.

Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2Struct: Screenshot parsing as pretraining for visual language understanding. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 18893–18912. PMLR, 2023. URL https://proceedings.mlr.press/v202/lee23g.html.

Yanhong Li, Zixuan Lan, and Jiawei Zhou. Text or pixels? evaluating efficiency and understanding of LLMs with visual text inputs. In *Findings of the Association for Computational Linguistics: EMNLP 2025*, pp. 10564–10578. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.findings-emnlp.558. URL https://aclanthology.org/2025.findings-emnlp.558/.

Yinxi Li, Yuntian Deng, and Pengyu Nie. TokDrift: When LLM speaks in subwords but code speaks in grammar. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 47605–47629. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.2199. URL https://aclanthology.org/2026.acl-long.2199/.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out*, pp. 74–81, Barcelona, Spain, July 2004. Association for Computational Linguistics. URL https://aclanthology.org/W04-1013/.

Jonas Lotz, Elizabeth Salesky, Phillip Rust, and Desmond Elliott. Text rendering strategies for pixel language models. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, pp. 10155–10172. Association for Computational Linguistics, 2023. doi: 10.18653/v1/2023.emnlp-main.628. URL https://aclanthology.org/2023.emnlp-main.628/.

Francesca Lucchetti and Arjun Guha. Understanding how CodeLLMs (mis)predict types with activation steering. In *Proceedings of the 8th BlackboxNLP Workshop: Analyzing and Interpreting Neural Networks for NLP*, pp. 358–397. Association for Computational Linguistics, 2025. doi: 10.18653/v1/2025.blackboxnlp-1.22. URL https://aclanthology.org/2025.blackboxnlp-1.22/.

R. Thomas McCoy, Ellie Pavlick, and Tal Linzen. Right for the wrong reasons: Diagnosing syntactic heuristics in natural language inference. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pp. 3428–3448. Association for Computational Linguistics, 2019. doi: 10.18653/v1/P19-1334. URL https://aclanthology.org/P19-1334/.

OpenAI. GPT-5 system card, August 2025. URL https://openai.com/index/gpt-5-system-card/. Official system card, published 2025-08-07.

Congyang Ou, Ruike Song, Yang Zhou, Libo Sun, Haokui Zhang, and Zhenbo Luo. When vision becomes text: Visual token pruning via cross-modal residual guidance in VLMs, 2026. URL https://arxiv.org/abs/2608.10489.

Md. Rafiqul Islam Rabin and Mohammad Amin Alipour. Code2Snapshot: Using code snapshots for learning representations of source code. In *2022 21st IEEE International Conference on Machine Learning and Applications*. IEEE, 2022. doi: 10.1109/ICMLA55696.2022.00140. URL https://arxiv.org/abs/2111.01097.

Phillip Rust, Jonas F. Lotz, Emanuele Bugliarello, Elizabeth Salesky, Miryam de Lhoneux, and Desmond Elliott. Language modelling with pixels. In *International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=FkSp8VW8RjH.

Xin Shen, San-Zhuo Xi, Yali Du, and Ming Li. On the lexical superstition of large language models for code comprehension: Re-evaluation on code of low lexical quality, 2026. URL https://arxiv.org/abs/2609.26388. Preprint.

Ruibo Shi, Lili Tao, Rohan Saphal, Fran Silavong, and Sean J. Moran. CV4Code: Sourcecode understanding via visual code representations. In *Proceedings of the Asian Conference on Computer Vision*, pp. 3464–3480, 2022. URL https://openaccess.thecvf.com/content/ACCV2022/html/Shi_CV4Code_Sourcecode_Understanding_via_Visual_Code_Representations_ACCV_2022_paper.html.

Yuling Shi, Chaoxiang Xie, Zhensu Sun, Yeheng Chen, Chenxu Zhang, Longfei Yun, Chengcheng Wan, Hongyu Zhang, David Lo, and Xiaodong Gu. Seeing is coding: On the effectiveness of vision language models in code understanding, 2026. URL https://arxiv.org/abs/2602.01785. ISSTA 2026 camera ready; project commonly referred to as CodeOCR.

Yan Shu, Hangui Lin, Yexin Liu, Yan Zhang, Gangyan Zeng, Yan Li, Yu Zhou, Ser-Nam Lim, Harry Yang, and Nicu Sebe. When semantics mislead vision: Mitigating large multimodal models hallucinations in scene text spotting and understanding, 2025. URL https://arxiv.org/abs/2506.05551.

Jonathan Steinberg and Oren Gal. Where vision becomes text: Locating the OCR routing bottleneck in vision-language models, 2026. URL https://arxiv.org/abs/2602.22918.

Adam Štorek, Mukur Gupta, Samira Hajizadeh, Prashast Srivastava, and Suman Jana. Sense and sensitivity: Examining the influence of semantic recall on long context code understanding. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 480–498. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.19. URL https://aclanthology.org/2026.acl-long.19/.

Wenxin Tang, Jingyu Xiao, Zhenyu Liu, Zipeng Xie, Junliang Liu, Wang Luo, Yuan Jiang, Yintong Huo, and Michael R. Lyu. CodeShrink: Adaptive visual compression for efficient multimodal code understanding, 2026. URL https://arxiv.org/abs/2607.29637.

Kushal Tatariya, Vladimir Araujo, Thomas Bauwens, and Miryam de Lhoneux. Pixology: Probing the linguistic and visual capabilities of pixel-based language models. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing*, pp. 3307–3320. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.emnlp-main.194. URL https://aclanthology.org/2024.emnlp-main.194/.

Jianhong Tu, Nicholas Crispino, Kyle Montgomery, Chenguang Wang, and Dawn Song. Fico: Evaluating vision-language models under visual fidelity and compression at scale. In *Findings of the Association for Computational Linguistics: ACL 2026*, pp. 35261–35277. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.findings-acl.1758. URL https://aclanthology.org/2026.findings-acl.1758/.

Guang Yang, Yu Zhou, Wenhua Yang, Tao Yue, Xiang Chen, and Taolue Chen. How important are good method names in neural code generation? a model robustness perspective. *ACM Transactions on Software Engineering and Methodology*, 33(3):60:1–60:35, 2024. doi: 10.1145/3630010. URL https://doi.org/10.1145/3630010.

Tianyi Zhang, Varsha Kishore, Felix Wu, Kilian Q. Weinberger, and Yoav Artzi. BERTScore: Evaluating text generation with BERT. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=SkeHuCVFDr.

Jianping Zhong, Guochang Li, Chen Zhi, Junxiao Han, Zhen Qin, Xinkui Zhao, Nan Wang, Shuiguang Deng, and Jianwei Yin. Can vision-language models handle long-context code? an empirical study on visual compression, 2026. URL https://arxiv.org/abs/2602.00746. Introduces LongCodeOCR.

A Data and Experimental Construction

A.1 Public data, filtering, and source–donor pairs

We reconstruct the candidate pool from the public Python portion of CodeSearchNet. Records must contain one top-level function or asynchronous function, a usable functional reference, and code that passes the frozen syntax, compilation, length, renderability, and controllable-binding checks. The conservative binding analyzer rejects constructs for which local name resolution cannot be audited reliably, including nested scopes, reflective access, selected keyword-call cases, and ambiguous shadowing. Every exclusion is written to a manifest rather than silently dropped. The released scripts accept either the official JSONL layout or a Parquet conversion with the original CodeSearchNet fields and record the checksum of the source archive used in development.

Each experimental unit contains a source function and a functionally different donor reference. The source contributes the implementation, source reference, and Original identifiers; donor code is never shown to the model. Donors are selected under role-coverage and length constraints and from a different repository component. Source/donor option order is balanced 250/250 in the main set. Repository-component identifiers are retained for splitting, mismatch construction, and two-way clustered uncertainty. Table 2 shows the resulting branches; the smaller sets are not treated as one false linear nesting.

Table 2: Frozen experimental sample sets and their roles.

Set	N	Relation	Use
Accepted pool	561	Filtered candidate pool	Source selection after syntax, reference, binding, length, and rendering review
Main behavior	500	Drawn from accepted pool	O/A/C choice and summary under Text and Readable Image; Parity choice
Selective salience	250	Main-set subset	Four-cell identifier $\times$ structure visual intervention
Image readback	100	Salience-set subset	O/A/C Readable and Parity visual-availability diagnostic
Representation	96	Repository-separated formal set	Geometry, direction correspondence, summaries, and likelihood support
Patching development	16	Component-disjoint development set	Frozen r18/r24/r30 gate and earliest-eligible selection
Patching held-out	64	Disjoint from development	Formal answer-start activation intervention

Notes. The behavior subsets share the main source–donor construction. The representation and patching cohorts use repository-component separation for their stated comparisons. N denotes source programs, before multiplying by arm, carrier, task, or patch family.

A.2 Binding-aware identifier interventions

The analyzer resolves eligible bindings and records every definition and use. The default controlled set contains the top-level function name, ordinary parameters other than conventional `self`/`cls` receivers, and local bindings such as assignment, loop, exception, and context-manager targets. A deterministic plan assigns one collision-free replacement to each binding and propagates it to every resolved occurrence. Original (O) retains the source names. Indexed Anonymous (A) assigns role-indexed names such as `func0`, `para0`, and `var0`. Confusing (C) assigns eligible donor names by binding role and source order. When a donor lacks sufficient unique names for a role, the pair is not silently completed with source names.

Validation requires legal identifiers, collision freedom, successful parsing and compilation, preservation of each renamed binding's occurrence count, unchanged protected names, and equality of the original and transformed ASTs after both are canonicalized to the same binding labels. Thus, *semantics preserving* denotes binding-consistent transformation with unchanged executable control

Table 3: Binding-aware transformation scope.

Class	Treatment	Audit
Function binding	Rename, including direct recursive uses	Definition/use propagation and canonical-AST check
Parameters and local variables	Rename by role and source order	Collision, occurrence-count, parse, and compile checks
Conventional receiver	Preserve `self`/`cls`	Recorded as protected
Attributes and keyword labels	Preserve	Equality check over protected names
Imports, builtins, external APIs	Preserve	Recorded as unresolved or protected external symbols
Literals, operators, control flow	Preserve semantically; formatting may be normalized	Canonical-AST equivalence and execution checks where available
Unsupported scopes or reflection	Reject the record	Structured rejection reason in the construction manifest

Table 4: Readable rendering and derived visual interventions.

Profile	Frozen settings	Construction and invariant
Readable	Menlo face 0, 28 px; 38 px line height; 48 px margin; 1024–2240 px canvas on a 16 px grid; at most 8 pages; 4-space tabs; 300-DPI metadata	Black-on-white deterministic raster; shared token-boundary wrapping; balanced source-line pagination; font SHA-256 `dc256e0b...c1f6`
Parity	Model-specific visual-budget profile	Uniform aspect-preserving rescale of the frozen Readable raster; all arms share the same scale and page dimensions. This jointly changes fidelity, density, and visual budget.
Identifier salience	Up to four recurrent bindings with at least two occurrences	Binding-consistent colors from a four-color palette; sample-hash rotation fixes palette assignment without moving glyphs
Structure salience	Python control keywords and indentation	Bold structural keywords plus 1-px indentation guides every four columns; the same layout plan and character coordinates are retained
Both salience	Identifier and structure operations combined	Factorial 1/1 cell; plain, identifier, structure, and both share code and layout

and data flow within the supported scope. It does not mean surface identity or removal of every lexical cue. Table 3 states the boundary.

Natural and Nonce controls extend the same full-binding interface and are described in Appendix F.

A.3 Text and visual carriers

Text inputs place the code verbatim in the task template. Readable Image inputs are generated from a frozen shared-wrap plan so that identifier arms for one program use common wrap boundaries and page allocation. The renderer fails closed on a font-digest mismatch or missing glyph. Table 4 gives the principal profile. Each page is encoded once as PNG and transmitted as a base64 data URL without client-side resizing, re-encoding, or re-pagination; per-cell image digests are recorded. Provider-side preprocessing is not observable.

Table 5: Behavioral model instances and experimental coverage.

Reported instance	Requested identifier or revision	Execution	Coverage
Qwen3-VL-4B	`Qwen/Qwen3-VL-4B-Instruct`	Local vLLM	Main, Parity, salience, readback
Qwen3-VL-8B	`qwen3-vl-8b-instruct`	Provider; local	Main, Parity, salience, readback; all internal analyses
Qwen3-VL-32B	`qwen3-vl-32b-instruct`	Provider	Main, Parity, readback
Qwen3-VL-Plus	`qwen3-vl-plus-2025-12-19`	Provider	Main, readback
Kimi K2.5	`kimi-k2.5`	Provider	Main, Parity, salience, readback
GPT-5	`gpt-5-2025-08-07`	Provider	Main, readback
Claude Sonnet 4.5	`claude-sonnet-4-5-20250929`	Provider	Main, readback
GLM-5V-Turbo	`glm-5v-turbo`	Provider	Main, readback

A.4 Tasks, prompts, and response handling

Functional choice asks which of two descriptions best matches the implementation and requires {`"choice":"A"`} or `B`. Source and donor descriptions are balanced across A/B. Summary asks for one concise sentence describing the implementation and requires a single `summary` string. Image-only readback asks for an exact transcription and requires one `code` string. The anonymous archive releases the verbatim Text and Image templates and their JSON Schemas.

The main behavior family includes the system instruction "You are a helpful coding assistant." Other families use no system turn. One user turn carries one stimulus; for images, all pages occur in page order before the prompt text. Where supported, requests use temperature 0, top-p 1, disabled thinking, and strict JSON Schema. GPT-5 uses minimal reasoning and `max_completion_tokens`; other interfaces select their compatible output-token field. Main behavior records seed 0 or null according to interface support; other families omit seed. The client uses concurrency four and a 180-second timeout. At most one byte-equivalent structured-request retry is permitted for transport, timeout, rate-limit, or explicitly retryable runtime errors. Fenced, truncated, or invalid-JSON content is retained and never retried or repaired in the primary analysis.

The donor-choice endpoint is a within-sample risk difference. Summary DMS is donor-reference similarity minus source-reference similarity, using ROUGE-L F1 in percentage points and offline BERTScore F1 in its native scale. Readback reports character error rate (CER), normalized transcription exactness, and multiset-exact recovery of controlled identifiers. The choice-scoring and patching readout is the donor-option logit minus the source-option logit.

B Models, Evidence Design, and Statistical Procedures

B.1 Model instances and execution configurations

Table 5 lists the eight fixed behavior instances. Identifiers returned by the provider matched those requested, but the serving interfaces exposed no usable snapshot fingerprint; rolling aliases are therefore identifiers rather than immutable snapshot claims.

The local Qwen3-VL-4B-Instruct weight revision is `ebb281ec70b05090aa6165b016eac8ec08e71b17`. The focal local Qwen3-VL-8B-Instruct model uses weight revision `0c351dd01ed87e9c1b53cbc748cba10e6187ff3b` and processor revision `60595ebc30ec8e3b1d3b9e65d4943ca011c0006a`. Direct local execution uses CPython 3.12.3, PyTorch 2.9.0 with CUDA 12.8, Transformers 4.57.6, float16 weights, SDPA, batch size one, seed 314159, and a 32-GB Tesla V100S. Representation capture, teacher-forced likelihood, and patching use `use_cache=false`; greedy summary and readback generation use the generation cache and limits of 128 and 2048 new tokens. The local serving stack uses vLLM 0.11.2, PyTorch 2.9.0+cu128, Transformers 4.57.6, xgrammar 0.1.25, and float16; it serves both the 4B instance and an 8B endpoint bridge. Offline BERTScore uses `FacebookAI/roberta-large` revision

Table 6: Evidence status of the principal analyses.

Analysis	N	Status	Frozen or selected element
O/A/C choice and summary	500	Main confirmatory evidence	Arms, carriers, prompts, pairing, and directional contrasts
Parity boundary	500	Secondary boundary	Uniform rescale profile and Readable comparison
Selective salience	250	Prespecified factorial	Two main effects, interaction, and within-model Holm family
Program geometry/directions	96	Formal held-out	`p_code r16` and `h_answer r36`; other depth variation descriptive
Summary/NLL support	96	Formal supporting evidence	Same-program/mismatch summary pairs and two-way realized-path score
Patching gate	16	Development selection	Candidate set, nine checks, and earliest-eligible rule
Activation patching	64	Formal held-out	r24, four equal-dose families, bidirectional primary average

`722cf37b...f1d59`, layer 17, its slow tokenizer, no IDF, no baseline rescaling, maximum length 512, and batch size 16. The provider request client runs under CPython 3.14.5.

B.2 Development choices and held-out evidence

The formal representation set and patching sets are separated by repository component. The `p_code r16` readout was fixed from development analysis; `h_answer r36` is the architectural endpoint immediately before final normalization/generation. For patching, a component-disjoint $N = 16$ development cohort evaluates only r18, r24, and r30. A layer must pass identity and repeat exactness plus seven positive-direction checks. The rule selects the earliest eligible candidate, never the largest development effect. Layers r24 and r30 passed, so r24 was frozen before the held-out $N = 64$ run; Appendix E gives all gate results.

B.3 Estimands and uncertainty

Behavioral contrasts are paired within source program. Confidence intervals for behavior and patching use two-way source- and donor-repository clustering; when a two-way component is unavailable, the frozen analysis records the fallback rather than substituting an unreported denominator. Salience tests form a three-contrast family per model and use Holm adjustment. CKA significance uses 1,024 sample-label permutations with upper-tail value $(b + 1)/(1024 + 1)$; RSA is descriptive. Identifier-direction gaps use paired bootstrap intervals. Missing responses propagate to the metric-specific valid denominator and are not directionally imputed.

The mismatched representation baseline compares a program's Text direction with cross-program Image directions and is summarized over admissible pairs. Patching instead requires one deterministic sender per receiver: the sender differs in sample and repository component, shares source-option orientation, and belongs to a one-to-one derangement; the formal map is also length-band stratified. These controls answer different questions and are not interchangeable.

C Complete Behavioral Results for RQ1

C.1 Original, Anonymous, and Confusing behavior

Table 7 reports the raw O/A/C cells. Discrete choice uses the donor-choice percentage. Each summary cell gives ROUGE-L DMS in percentage points over BERTScore DMS. Less-negative DMS indicates movement toward the donor reference even when the output remains closer to the source in absolute terms. The rows are fixed instances; no cross-model population mean is estimated.

Table 7: Eight-model O/A/C main matrix: choice and directional summary statistics.

	Choice: donor-choice rate (%)						Summary: DMS (ROUGE-L pp / BERTScore F1)					
	Original		Anonymous		Confusing		Original		Anonymous		Confusing	
Model instance	T	I	T	I	T	I	T	I	T	I	T	I
Qwen3-VL-8B	0.0	0.0	0.0	0.2	6.2	11.6	-6.45 -.0232	-6.51 -.0221	-4.90 -.0173	-4.94 -.0176	-2.54 -.0090	-2.51 -.0100
Qwen3-VL-4B	0.0	0.2	0.2	0.4	10.4	18.0	-7.43 -.0238	-6.24 -.0217	-5.33 -.0185	-4.66 -.0172	-2.52 -.0112	-2.46 -.0098
Qwen3-VL-32B	0.0	0.2	0.0	0.2	3.6	6.6	-6.32 -.0239	-5.70 -.0216	-5.01 -.0196	-4.12 -.0169	-3.28 -.0125	-2.64 -.0091
Qwen3-VL-Plus	0.2	0.0	0.0	0.6	4.6	22.8	-6.00 -.0223	-4.58 -.0200	-4.50 -.0188	-3.17 -.0143	-2.78 -.0124	-1.56 -.0078
Kimi K2.5	0.0	0.0	0.0	0.0	3.2	5.2	-6.08 -.0231	-5.62 -.0227	-5.07 -.0201	-5.09 -.0204	-3.50 -.0154	-3.62 -.0146
GPT-5	0.2	0.2	0.4	0.2	1.4	4.4	-4.97 -.0214	-4.50 -.0206	-4.18 -.0189	-3.80 -.0174	-3.28 -.0149	-2.69 -.0113
Claude Sonnet 4.5	0.0	0.0	0.0	0.2	0.8	16.8	-6.29 -.0248	-5.78 -.0226	-5.87 -.0223	-4.23 -.0183	-4.66 -.0176	-2.36 -.0112
GLM-5V-Turbo	0.0	0.2	0.2	0.6	2.4	14.2	-5.93 -.0241	-4.74 -.0210	-4.89 -.0210	-3.80 -.0180	-3.64 -.0153	-1.96 -.0104

Notes. T/I denote Text/Readable Image. Choice cells contain 500/500 strict-valid responses. ROUGE-L uses the operational all-request estimand with nominal $N = 500$: a strict-invalid, empty, or truncated summary remains invalid and contributes zero overlap and zero output length; both-valid paired sensitivity excludes pairs with either response invalid. BERTScore is computed only for strict-valid, nonempty scorer-ready summaries, yielding cell $N = 487$–500; invalid rows remain recorded but unscored. The image C−O choice effects are 11.6, 17.8, 6.4, 22.8, 5.2, 4.2, 16.8, and 14.0 pp in row order; the small differences from raw C cells occur when Original is above zero.

C.2 Image-only readback

Table 8 reports the two diagnostics most relevant to the main interpretation. Identifier recovery remains high in every arm and no model exhibits a C-specific collapse. Anonymous identifier recovery is never below Original in the seven fully strict cohorts and is equal to or higher under the strict Qwen3-VL-Plus view. The latter model frequently returned otherwise usable transcriptions outside its requested JSON envelope; the table preserves the strict primary view, while conservative de-fencing recovers all 100 requests per arm and changes identifier exactness to 97/98/95% for O/A/C. Its recovered-view CER is much larger because the retained wrappers enter character scoring, so that view is a response-format sensitivity rather than the main visual diagnostic.

C.3 Lower-fidelity and selective-salience boundaries

Parity sharply reduces exact identifier recovery and increases CER, yet its C−O choice effect remains positive and is larger than the Readable effect in all four evaluated models (Table 10). Since the transformation jointly changes resolution, optical density, layout budget, and verbatim availability, the Parity-minus-Readable column is descriptive and does not identify a pure visual-token-dose effect.

The selective-salience cells in Table 11 show small, non-replicating changes. All prespecified contrast intervals cross zero, and no contrast rejects after the three-test within-model Holm adjustment. These are bounded null results at the tested dose and precision, rather than equivalence evidence.

Table 8: Readable Image readback by model and identifier arm.

	Valid N			Controlled-ID exact (%)			CER (%)		
Model	O	A	C	O	A	C	O	A	C
Qwen3-VL-4B	100	100	100	92.0	95.0	90.0	1.68	1.63	1.80
Qwen3-VL-8B	100	99	99	93.0	98.0	96.0	1.34	1.34	1.42
Qwen3-VL-32B	100	100	100	95.0	95.0	95.0	1.25	1.22	1.44
Qwen3-VL-Plus	51	51	52	98.0	98.0	92.3	1.09	1.03	.98
Kimi K2.5	100	99	96	100.0	100.0	99.0	.64	.65	.63
GPT-5	100	100	100	96.0	97.0	99.0	1.86	1.40	1.78
Claude Sonnet 4.5	100	100	100	99.0	99.0	99.0	.77	.82	.71
GLM-5V-Turbo	100	100	100	99.0	99.0	99.0	.74	.80	.79

Notes. Nominal $N = 100$ per arm. Primary rows require the frozen schema and scoring contract. Controlled-ID exact requires equality of the identifier multiset over controlled bindings; CER is computed against the expected transformed code. Readback diagnoses gross visual availability rather than functional understanding.

Table 9: Cross-cell directional checks supporting the RQ1 summary and Anonymous-control claims.

Metric contrast	Direction across model–carrier cells	Observed magnitude
C−O source-reference BERTScore F1	Decrease in 16/16	$-.0083$ to $-.0048$
C−O BERTScore DMS	Increase in 16/16	$+.0066$ to $+.0142$
A−O donor-choice rate	$\lvert\Delta\rvert \leq 3$ pp in 16/16	Sample churn 0.0–0.6%
A−O source-reference ROUGE-L	Decrease in 16/16	Directional check
A−O donor-reference ROUGE-L	Increase in 3/16	Maximum $\lvert\Delta\rvert = 0.40$ pp
C−A donor-reference ROUGE-L	Increase in 16/16	$+0.41$ to $+1.92$ pp

Notes. The 16 cells are eight fixed model instances × Text/Readable Image. DMS is donor-reference minus source-reference similarity. Choice churn is the fraction of samples whose selected option changes, rather than the difference between marginal donor-choice rates.

D Cross-Carrier Geometry and Output Support for RQ2

Throughout this appendix, *text-compatible* means correspondence with states elicited by textual code, not membership in an independently identified text-token subspace. *Content-matched* means that same-program correspondence exceeds the stated mismatched-program control.

D.1 Representation extraction

All capture points are forward pre-hooks. Positions r0–r35 are inputs to decoder blocks 0–35; r36 is the input to final normalization. Hooks assert shape $(1, T, 4096)$ and finite values, cast retained vectors to float32, and are removed in a `finally` block. Model, processor, tokenizer, and prompt-token digests are checked before collection.

D.2 Program geometry across depth

Figure 4 shows the complete depth trajectories used in the paper. The `p_code` curve is non-monotonic, so depth-wise changes outside the frozen r16 and r36 readouts remain descriptive. Table 13 gives the exact fixed-position values. Every observed CKA exceeds the maximum of 1,024 sample-label permutations; the corresponding upper-tail value is $1/1025 = .00098$. RSA gives the same qualitative same-program correspondence but was not assigned a separate inferential test.

Table 10: Parity choice effects and accompanying readback changes.

	C−O choice effect (pp)			ID exact (%)		CER (%)	
Model	Readable	Parity	P−R	Readable	Parity	Readable	Parity
Qwen3-VL-8B	11.6	19.8	+8.2	95.0	46.7	2.02	5.10
Qwen3-VL-4B	17.8	32.6	+14.8	92.3	20.0	1.70	12.63
Qwen3-VL-32B	6.4	14.4	+8.0	95.0	54.3	1.30	3.60
Kimi K2.5	5.2	8.6	+3.4	98.0	79.7	2.30	3.77

Notes. Choice uses $N = 500$ paired programs per carrier. Readback pools O/A/C within carrier, nominal $N = 300$ requests. Original donor-choice rates remain 0–0.4% in both visual conditions.

Table 11: Selective-salience cells and frozen factorial contrasts.

	Confusing donor-choice rate (%)				Factorial contrast (pp; 95% CI)		
Model	Plain	Identifier	Structure	Both	Identifier main	Structure main	Interaction
Qwen3-VL-8B	12.0	14.0	13.6	14.0	+1.20 [-.55, 2.95]	+.80 [-.95, 2.55]	-1.60 [-5.10, 1.90]
Qwen3-VL-4B	18.0	19.2	17.6	19.2	+1.40 [-.63, 3.43]	-.20 [-1.72, 1.32]	+.40 [-2.20, 3.00]
Kimi K2.5	5.2	6.4	5.6	6.0	+.20 [-.48, .88]	-.60 [-2.12, .92]	+.40 [-1.68, 2.48]

Notes. $N = 250$ per model and cell. Intervals use two-way repository clustering. Holm-adjusted p values for identifier/structure/interaction are .539/.741/.741 (8B), .532/1/1 (4B), and 1/1/1 (Kimi).

D.3 Identifier-change direction correspondence

For arm $a \in \{A, C\}$ and program i, let $d^a_{T,i} = h_{T,i}(a) - h_{T,i}(O)$ and define $d^a_{I,i}$ analogously. The matched statistic is the cosine of the Text and Image directions for the same program. The mismatch statistic uses cross-program directions; admissible comparisons exclude the same program and repository component and are checked against length, rename-coverage, and norm restrictions. Table 14 reports the formal `h_answer r36` results.

D.4 Realized outputs and carrier-conditioned support

The same-program summary comparison uses BERTScore between the Text- and Image-generated greedy summaries. Its mismatch baseline pairs repository-component-disjoint programs matched on code and summary length. For likelihood support, each carrier scores both carrier-specific summary paths. EOS is excluded consistently, target-token positions are checked against prompt-prefix IDs, and a prompt-end gate compares the scoring and generation states. Define

$$G_{\mathrm{NLL}} = \frac{1}{2}\left[\mathrm{NLL}_I(y_T) - \mathrm{NLL}_T(y_T) + \mathrm{NLL}_T(y_I) - \mathrm{NLL}_I(y_I)\right].$$

A positive value means that each realized path receives lower NLL under its producing carrier, averaged over directions. Table 15 gives both the raw scoring matrix and the derived advantage.

E Answer-Start Activation Patching for RQ3

E.1 Intervention construction

At candidate layer l, the Text and Image C−O directions are $d_T = r_l(C, T) - r_l(O, T)$ and $d_I = r_l(C, I) - r_l(O, I)$. Text-derived and random vectors are unit-normalized and scaled to the receiver-specific $\|d_I\|_2$. Forward patches add the direction to an Original Image receiver; restoration patches subtract it from a Confusing Image receiver. Both are signed so that a positive change in $L = \mathrm{logit}(D) - \mathrm{logit}(S)$ is donor directed. Patching occurs only at `h_answer`, after the full unequal-length carrier input and two-option scaffold, so it requires no token alignment.

Matched Text uses the receiver program's d_T. Mismatched Text uses the frozen cross-program sender. Equal-norm Random is seeded per cohort, sample, layer, and receiver arm and is orthogonalized against the matched axis and, when non-degenerate, the mismatched residual. Same-carrier

Table 12: Representation positions and pooling definitions.

View	Text	Image
p_code	Mean over code-only, non-special tokens	Mean over all visual tokens, including whitespace regions
h_answer	Last prompt token immediately before answer generation or scoring	Same answer-start position after the image pages and choice/summary scaffold

Table 13: Fixed-position Text–Image geometry and permutation controls ($N = 96$).

Readout	Arm	CKA	RSA	Permutation max	Upper-tail p
p_code r16	Original	.9447	.8914	.3077	.00098
	Anonymous	.9384	.9277	.2831	.00098
	Confusing	.9307	.9086	.3024	.00098
h_answer r36	Original	.8750	.8036	.2780	.00098
	Anonymous	.8518	.7903	.2578	.00098
	Confusing	.7200	.6936	.2377	.00098

Notes. Centered linear CKA is the primary matrix-correspondence statistic; RSA is Spearman correlation over Euclidean pairwise distances.

Image applies the exact Image replacement delta and is the receiver-aligned reference. The hook must fire, the unpatched receiver state must match its recorded baseline byte-wise, and every record stores the applied norm and patch digest.

The formal mismatched map is a one-to-one cyclic derangement within source-option $\times$ length-band strata, excluding equal sample IDs and repository components. Candidate shifts are selected by a SHA-256 tie-break under the frozen seed. Random seeds use the first 15 hexadecimal digits of `SHA256(seed|random|cohort|sample|layer|arm)`.

E.2 Development gate

Table 16 gives all nine checks. r18 fails the same-carrier directional checks, both Matched-minus-Random checks, and the leave-one-out stability check. r24 and r30 pass all criteria; the frozen earliest-eligible rule therefore selects r24 even though r30 has a larger development mean.

E.3 Held-out intervention estimates

Table 17 separates forward and restoration directions from their prespecified equal-weight average. Random is centered near zero. Mismatched Text produces a sizable shared steering component; Matched Text adds a positive same-program increment; Same-carrier Image remains stronger. The Matched-minus-Mismatched average is the primary contrast. At the sample level it is positive in 38/64 programs with median .508, indicating heterogeneity around the positive clustered mean.

F Additional Identifier Controls

F.1 Anonymous, Natural, and Nonce construction

Indexed Anonymous replaces controlled identifier words with indexed placeholders but retains role, binding repetition, index order, length/digit patterns, and residual uncontrolled names. It is therefore a controlled-name replacement condition rather than a semantic zero. Natural (N) replaces every eligible binding with an unrelated natural-looking identifier under the same role-aware, collision-free interface. Nonce (Z) splits snake, camel, Pascal, acronym, and digit structure into lexical components; components of length at least three use frozen Wuggy candidates and length-two components use the deterministic fallback. The adapter restores separators, digit strings, component length, and

Table 14: Cross-carrier identifier-change direction alignment at `h_answer r36`.

Direction	Matched cosine	Mismatched cosine	Gap	95% CI	Positive gaps
A−O	.5713	.2340	+.3373	[.3147, .3602]	96/96
C−O	.6629	.1110	+.5518	[.5285, .5736]	96/96

Notes. Intervals are paired bootstrap intervals over programs. All matched cosines are positive. The result establishes program-specific correspondence beyond cross-program alignment; it does not establish that the two vectors are identical or exclusively textual.

Table 15: Cross-summary correspondence and realized-path NLL support.

	Summary BERTScore			Mean token NLL: scoring carrier / target path				
Arm	Same program	Mismatch	Gap	T/y_T	I/y_T	T/y_I	I/y_I	G_{NLL}
Original	.9187	.8448	+.0739	.139	.689	.817	.160	.604
Anonymous	.9124	.8440	+.0684	.143	.794	.933	.163	.710
Confusing	.9063	.8449	+.0614	.168	.879	.965	.179	.749

Notes. Same-program $N = 96$ for O/A and cross-summary C; each mismatch mean uses 480 pairs. NLL has $N = 96$ for O/A and $N = 95$ complete two-way C rows; the displayed I/y_T C mean includes 96 available scores, while G_{NLL} uses complete rows. Gap CIs are [.0723,.0751], [.0669,.0698], and [.0597,.0631]. G_{NLL} is a realized-path support contrast, not KL/JS distance or summary correctness.

case pattern, then applies the standard full-binding validation and shared rendering audit. Attributes, methods, external APIs, literals, and protected identifiers remain available in every arm.

F.2 Compact behavioral check

Table 18 restricts all five arms to the common 250-program subset. Donor choice remains at floor for A, N, and Z, whereas C produces the directional shift. The summary cells do not form a single O–A–N–Z–C continuum. Z also has lower identifier recovery than N, so N−Z cannot be interpreted as a pure semantic-naturalness contrast: wordhood, tokenization, glyph form, and visual recovery change together.

G Main-Claim Audit and Artifact Map

G.1 Denominators and material deviations

Table 19 consolidates the denominator qualifications that can change an estimate or its interpretation. Transport failures are not directionally imputed. Content-invalid outputs remain invalid in the primary view. No result below uses an unreported retry with a modified payload.

G.2 Claim-to-artifact and release map

The result files named in Table 20 are authoritative internal audit sources. They are not part of the anonymous tool release; their complete reported aggregates appear in the referenced appendix tables. The release instead contains the code and configurations needed to reconstruct inputs and issue equivalent calls.

Local package verification runs the binding-aware renaming tests, dataset-construction tests, renderer tests, five-arm/Wuggy adapter tests, prompt/hash checks, and a synthetic end-to-end rendering fixture. The frozen verification record reports 42 renaming, 30 dataset, 38 renderer, and 11 adapter tests passing. The checksum manifest covers the anonymous archive; generated images, caches, raw datasets, credentials, endpoint identities, and machine-specific paths are excluded.

Table 16: Development layer gate ($N = 16$).

Check	r18	r24	r30
Identity-patch exactness	✓	✓	✓
Repeat exactness	✓	✓	✓
Same-carrier Image forward > 0	×	✓	✓
Same-carrier Image restoration > 0	×	✓	✓
Matched−Mismatched forward > 0	✓	✓	✓
Matched−Mismatched restoration > 0	✓	✓	✓
Matched−Random forward > 0	×	✓	✓
Matched−Random restoration > 0	×	✓	✓
Minimum leave-one-out Matched−Mismatched > 0	×	✓	✓
Eligible	no	yes	yes
Matched−Mismatched bidirectional mean	.0127	.2861	2.4702
Minimum leave-one-out mean	-.0083	.0417	1.6036

Table 17: Held-out answer-start patching at frozen r24 ($N = 64$).

Family or contrast	Forward	95% CI	Restoration	95% CI	Average	95% CI
Equal-norm Random	-.138	[-.319,.043]	-.006	[-.204,.191]	-.072	[-.197,.053]
Mismatched Text	2.546	[2.040,3.052]	1.306	[.572,2.039]	1.926	[1.416,2.435]
Matched Text	3.510	[2.855,4.166]	1.908	[1.066,2.751]	2.709	[2.118,3.301]
Same-carrier Image	3.729	[3.057,4.401]	3.213	[2.442,3.984]	3.471	[2.791,4.152]
Matched−Mismatched	.964	[.412,1.517]	.603	[-.146,1.351]	.783	[.215,1.352]
Same-carrier−Matched	.219	[-.063,.501]	1.305	[.362,2.247]	.762	[.219,1.305]

Notes. Entries are changes in donor-minus-source raw logit. Intervals use two-way source- and donor-repository clustering. The equal-weight forward/restoration average was fixed before the held-out run; directional rows are its decomposition.

Table 18: Common-250 five-arm results for the focal Qwen3-VL-8B instance.

	Choice donor rate (%)		Summary source / DMS ROUGE-L (pp)		Image readback	
Arm	Text	Image	Text	Image	ID exact (%)	CER (%)
			16.94	16.95		
O	0.0	0.0	-7.20	-7.54	93.0	1.34
			15.46	15.14		
A	0.0	0.0	-5.77	-5.59	98.0	1.34
			14.78	14.51		
C	6.4	12.0	-3.48	-3.25	96.0	1.42
			15.34	14.56		
N	0.0	0.0	-5.25	-4.43	96.0	1.95
			16.24	15.94		
Z	0.0	0.4	-5.91	-5.73	80.0	2.37

Notes. O/A/C choice and summary cells have $N = 250$. N/Z choice and summary have valid $N = 249$–250 because four technical cells are missing across the combined matrix. Readback uses the common nested 100-program Image subset: valid $N = 100$ for O/Z and $N = 99$ for A/C/N.

Table 19: Main-experiment denominators and material dispositions.

Family	Nominal unit	Analyzed	Material disposition
Main choice	500 per model-carrier-arm	500 strict-valid	All main choice responses passed strict validation
Main ROUGE-L	500 per model-carrier-arm	500 operational rows	Invalid summaries remain invalid and contribute zero overlap and zero output length; both-valid sensitivity is reported separately
Main BERTScore	500 summaries per cell	487–500 scorer-ready	Strict-invalid, empty, or missing summaries remain recorded but unscored; denominators are propagated by cell
Readable readback	100 per model-arm	96–100, except Plus 51/51/52 strict	Plus all-100 conservative recovery reported only as sensitivity
Parity choice / readback	500 / 100 per arm	Choice 500; pooled readback nominal 300	Six 4B Parity length truncations retained under the frozen diagnostic disposition
Salience	250 per model-cell	250	Three-contrast Holm family within each model
Geometry and directions	96 programs	96	Complete O/A/C Text–Image representations
Summary/NLL support	96 programs	Summary 96; NLL C 95 complete	One Confusing/Image non-EOS scoring item excluded from two-way G_{NLL}
Patching development / formal	16 / 64	16 / 64	r24 selected before formal run by earliest-eligible rule
Five-arm control	250; readback 100	249–250; 99–100	Small N/Z technical missingness and cross-run provider qualifier

Table 20: Main-paper claim and figure map.

Paper item	Authoritative source	Appendix reproduction record
Figure 1 / construction	Released dataset, renaming, renderer, and prompt packages	Tables 2–4; exact templates in `prompts/prompt_templates.json`
Figure 3 / RQ1	Frozen Phase-2 analysis results and `figure3-behavioral-persistence-v2-data.json`	Tables 7–11
Table 1 / output support	`formal96-analysis-v1.json`, likelihood scores, and frozen cross-summary-null scores	Table 15
Figure 4A–B	`formal96-analysis-v1.json` and `figure4-cross-carrier-mechanism-v2-data.json`	Tables 13–14
Figure 4C	Frozen layer verdict and formal patch analysis	Tables 16–17
Additional naming control	Common-250 decomposition and full-coverage N/Z analysis	Table 18

Table 21: Anonymous supplementary artifact inventory and redistribution boundary.

Artifact class	Availability	Reproduction route
Public CodeSearchNet inputs	Public / rebuildable	Official release plus `phase1-dataset/` inventory, selection, pairing, and materialization scripts
Renaming and identifier controls	Released	`renaming-toolchain/` and `phase3_tasks/`, with unit tests and binding manifests
Readable, Parity, salience rendering	Released	`codeocr_renderer/` with frozen JSON profiles, font checksum, and deterministic tests
Prompts and response schemas	Released	`prompts/prompt_templates.json` and materializer
Model/run/representation specifications	Released	`environment/ANONYMIZED-EXPERIMENT-CONFIGURATION.md`; `environment/EXECUTION-BOUNDARY.md`
Verification and file integrity	Released	`scripts/verify_package.py`; `scripts/scan_anonymity.py`; `MANIFEST.sha256`
Provider orchestration, responses, and derived results	Not redistributed	Released prompts and settings support equivalent calls; reported aggregates and valid denominators appear in Appendices C–G
Model weights and hidden-state tensors	Not redistributed	Exact public model/revision and capture/patch specifications in Appendix B and released configuration