BEYOND ATTENTION-GUIDED WARPING: LEARNING QUESTION-GUIDED VISUAL REALLOCATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Multimodal large language models (MLLMs) operate under a limited visual-token budget, which can undersample question-relevant details while allocating substantial capacity to background regions. Cropping and active visual search can recover local detail, but may remove contextual evidence or require repeated model evaluations. Moreover, intermediate-layer attention does not reliably predict which finite image deformation will improve the decoded answer. We propose QGVR (question-guided visual reallocation), a two-stage proposal-and-selection framework that redistributes visual sampling density without changing the frozen MLLM or its token interface. In Stage I, a proposal model predicts a question-conditioned, full-frame sampling warp from frozen native-image ViT features, using only image–question–answer supervision without boxes, masks, or region labels. Stage II freezes the proposal model, constructs a candidate set comprising the native observation, the base warp, and eight fixed edits, and uses offline answer scores from the frozen MLLM to supervise native-relative ranking and fallback decisions. On InternVL, inference requires at most one additional ViT encoding of the selected warped image and a single answer-decoding pass. QGVR achieves the highest reported scores on all five InternVL benchmarks and four of five LLaVA benchmarks in our comparison. Across four annotated evaluation subsets, QGVR directs the model's post-decoding visual attention to target regions more accurately, improving target-hit accuracy by 4.00–12.18 percentage points over image warping guided by intermediate-layer attention maps. Anonymous project materials and a code framework are available at https://anonymous.4open.science/w/qgvr-framework-03F0/.

1 INTRODUCTION

Multimodal large language models (MLLMs) process images under a limited visual-token budget, yet the evidence needed to answer a question—such as small text, objects, or local spatial relations—often occupies only a small portion of the image. Question-agnostic image resizing and tiling may allocate visual tokens inefficiently when such evidence is sparse. Even when the relevant region is correctly identified, insufficient sampling of its fine-grained details may still limit answer accuracy (Zhang et al., 2025). Fine-grained perception therefore requires not only locating relevant evidence but also preserving sufficient detail in its visual representation. This motivates the following question: Can a fixed visual sampling budget be redistributed toward question-relevant regions while preserving full-image context, without modifying the underlying MLLM?

Figure 1 illustrates this idea: after a local continuous warp, the model responds more strongly around the specified book spine and changes its answer from incorrect to correct. This example suggests that changing sampling locations, rather than increasing the number of tokens, can make a fixed visual budget more useful for representing answer evidence.

Token compression methods such as FastV and VisionTrim prune or merge visual tokens to reduce computation (Chen et al., 2024; Yu et al., 2026), but cannot restore details lost during input sampling. ViCrop uses internal model responses to select crops, while SD-RPN predicts regions from denoised-attention supervision (Zhang et al., 2025; Shi et al., 2026). Cropping improves local detail but

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

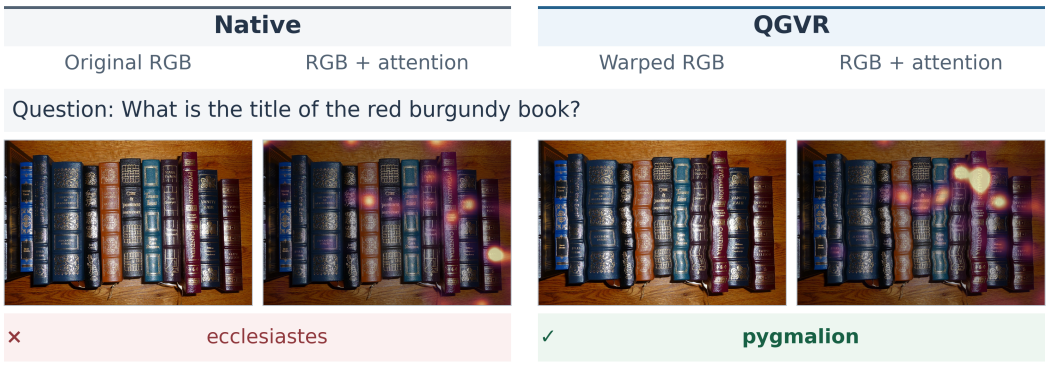

Figure 1: Question-guided visual reallocation. From left to right: the native image, its decoding attention, the QGVR warp, and its decoding attention, with actual answers below. Warping concentrates the response around the burgundy book spine specified in the question. Both paths share a color scale; brighter colors indicate stronger responses. Heatmaps are explanatory visualizations, not model inputs or direct measurements of token utilization. See Appendix B.6 for the display protocol.

excludes surrounding context; retaining the original image alongside crops adds visual tokens and encoding cost. DeepEyes interleaves reasoning with cropping and zooming, whereas ReFocus performs code-based visual editing (Zheng et al., 2026; Fu et al., 2025). These multi-step approaches introduce sequential model or tool calls, increasing inference latency.

Continuous image warping offers a way to enhance local detail while retaining the full field of view. AttWarp uses decoder attention to redistribute spatial resolution on a fixed output canvas, improving visual question answering without modifying the MLLM; its distilled variant reduces the cost of obtaining sampling guidance (Dalal et al., 2026). However, attention-based guidance does not directly measure whether a finite warp improves the answer. Moreover, magnifying one region reduces sampling density elsewhere, potentially weakening other evidence. These trade-offs motivate learning both *how to warp the image* from verified answer improvements and *whether to use the warp* instead of the original image.

We introduce QGVR, a framework that learns to redistribute visual sampling toward question-relevant evidence and select the view most likely to improve the answer (Figure 2). Using image–question–answer pairs, we train a proposal model to produce question-guided image warps that retain the full field of view. Its supervision comes from evaluating candidate warps with the frozen MLLM against reference answers, linking image reallocation to answering performance. To account for questions that benefit from different warps or from no warping at all, we train a lightweight selector on the same training data, using offline answer scores for the original image and alternative warped views. The selector learns to choose among these views before answer generation. At inference, the MLLM decodes only the selected view, avoiding repeated answer generation across candidates. Training requires no boxes, masks, or region labels, and the underlying MLLM remains unchanged throughout.

Across five benchmarks evaluated with InternVL3.5-8B and LLaVA-1.5-7B, QGVR generally outperforms the range of methods compared in this paper. Region-alignment and throughput experiments further examine where the model attends and the cost of deployment. Our contributions are:

- We propose QGVR, a framework that formulates visual reallocation as two complementary learning tasks: warp proposal and view selection. It improves the visual representation of question-relevant details while preserving the full field of view and leaving the underlying MLLM unchanged.
- We introduce answer-supervised warp learning that constructs training targets from the frozen MLLM's evaluation of candidate warps, enabling the proposal model to learn sampling patterns that support question answering without boxes, masks, or region annotations.
- We design a lightweight selector that uses the original image as a reference and learns from offline answer scores to select a warped view or retain the original. Selection precedes answer generation at inference, so only the chosen view is decoded.

2 Related Work

Task-driven sampling and continuous warping. Spatial transformer networks learn geometric controls from task losses, while Learning to Zoom converts saliency into nonuniform sampling on a fixed canvas (Jaderberg et al., 2015; Recasens et al., 2018). Adaptive patch sizes allocate visual computation by changing patch partitions (Choudhury et al., 2026). AttWarp converts axial marginals of decoder attention into rectilinear warps, with iterative refinement and distilled prediction offering further accuracy and efficiency gains, respectively (Dalal et al., 2026). However, attention-based sampling guidance does not directly establish whether a particular warp improves answering performance, and local magnification can reduce the detail available elsewhere. QGVR addresses this gap by using the frozen MLLM's answer loss to verify candidate warps during proposal learning and training a selector to compare warped views against the original image. Its contribution lies in verified proposal learning and native-relative selection.

Local observations and visual-budget utilization. ViCrop locates crops from internal attention or gradients; SD-RPN learns a lightweight region predictor from denoised-attention pseudo-labels. Both re-encode local views to improve detail perception (Zhang et al., 2025; Shi et al., 2026). V* and ZoomEye use hierarchical search to acquire additional visual observations (Wu & Xie, 2024; Shen et al., 2025), while FastV and SparseVLM remove redundant tokens to reduce downstream computation (Chen et al., 2024; Zhang et al., 2024). These approaches face different trade-offs: crops omit surrounding context unless combined with the original image at extra token cost; sequential search adds observation and inference overhead; token compression cannot recover details lost before encoding. QGVR instead redistributes sampling density over the full image and selects one view for final decoding, retaining the full field of view without concatenating additional local views or pruning visual tokens.

Output attribution and candidate selection. Integrated Gradients accumulates gradients along an input path to attribute model outputs, and GrAInS uses attribution-identified tokens to steer model activations (Sundararajan et al., 2017; Nguyen et al., 2026). RRHF learns answer-preference rankings, while BOND distills Best-of-N selection into a model to reduce online sampling (Yuan et al., 2023; Sessa et al., 2025). Attribution alone does not establish the benefit of a finite image transformation, and answer-level ranking does not directly solve the problem of choosing a visual input before decoding. QGVR uses a local VJP to propose warps and verifies their effects before learning from them. It then learns a native-relative set ranking over visual candidates (Lee et al., 2019), retaining the original image as an option and requiring answer generation only for the selected view at inference.

3 Method

3.1 Overview

Given an image I and question Q, QGVR reallocates spatial samples to better represent answer evidence within a fixed visual budget, preserving full-image context and the native visual-token interface. As shown in Figure 2, Stage I trains VJP-Warp to learn *how to generate useful warps*, and Stage II trains WarpRank to learn *which candidate merits replacing the native image*. At inference, proposal and selection precede a single answer-generation pass on the selected observation.

Let R denote the frozen target MLLM, and let θ and ϕ be the proposal and selector parameters. Stage I updates only θ; Stage II fixes θ and updates only ϕ. The visual encoder, connector, and language decoder of R remain frozen throughout. Both stages use image–question–answer supervision only, without boxes, masks, or region labels.

3.2 Stage I: VJP-Warp—Learning Useful Warp Proposals

Predicting where to sample. Along the forward path in Figure 2(a), the frozen ViT extracts native features V_0. The proposal network fuses them with question embeddings e_Q, and a warp head produces the sampling-control map Z_θ. Its positive, normalized form P specifies a spatial density,

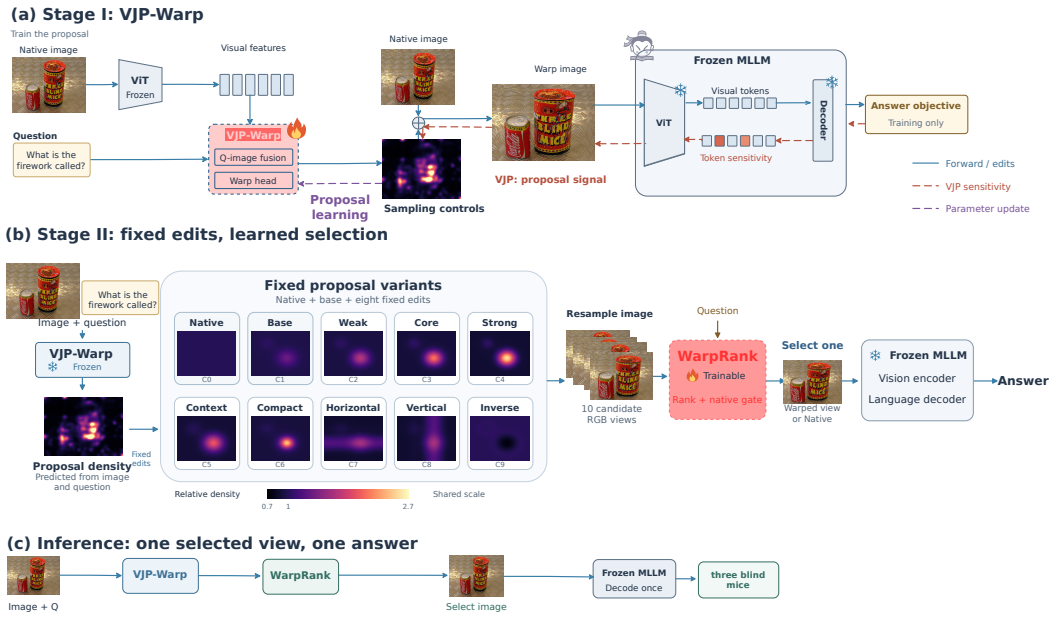

Figure 2: Proposal, selection, and inference in QGVR. (a) A VJP seeds proposals; the dashed purple loop summarizes proposal updates after candidate verification, not direct parameter updates from the answer loss. (b) Stage II freezes the proposal module and learns candidate ranking and native fallback; offline scoring supervision is detailed in the text. (c) Inference decodes only the selected observation. Flames and snowflakes indicate trainable and frozen modules, respectively. Images, density maps, and highlighted tokens are architectural illustrations, not measured intermediate outputs.

from which the resampling operator $\mathcal{W}$ generates a base warp:

$$Z_\theta = f_\theta(V_0, e_Q), \qquad I_1 = \mathcal{W}(I, Z_\theta),$$
$$P_i(Z) = \frac{\mathrm{softplus}(Z_i) + \varepsilon_0}{\sum_{j\in\Omega}(\mathrm{softplus}(Z_j) + \varepsilon_0)}, \qquad i \in \Omega. \tag{1}$$

Here, Ω is the control grid and $\varepsilon_0 > 0$ ensures positive density. Both V_0 and e_Q are fixed inputs; only f_θ is trainable. Higher-density regions receive more samples, while a uniform density yields the identity observation. Through a continuous coordinate mapping, $\mathcal{W}$ reallocates local scale while preserving the sample's native canvas and tile layout, without removing regions from the field of view. This density is a *sampling control*, not decoder attention; Appendix A.1 details its tempering and conversion to coordinates.

Proposing finite edits from answer sensitivity. To identify potentially useful sampling changes near the base warp, training uses native answer probabilities to construct an anchored objective $\mathcal{L}_{\mathrm{anc}}$ tilted toward the acceptable answer set $\mathcal{Y}$. Differentiating with respect to the current sampling controls gives

$$G = \left.\frac{\partial \mathcal{L}_{\mathrm{anc}}(\mathcal{W}(I, Z), Q, \mathcal{Y})}{\partial Z}\right|_{Z=Z_\theta}. \tag{2}$$

This vector–Jacobian product (VJP) follows the red backward path: the answer objective is differentiated through the frozen language decoder, visual encoder, and differentiable resampling operation back to Z. Frozen weights do not prevent input differentiation. We extract regions to adjust from G and vary magnification strength, spatial extent, and direction around the base proposal to form finite edits. A VJP describes only local sensitivity, not guaranteed improvements from finite warps; we therefore do not update θ directly using $\mathcal{L}_{\mathrm{anc}}$.

Verifying edits before learning proposals. The frozen MLLM evaluates the multi-answer negative log-likelihood $\ell(I_k)$ of the native image, base warp, and proposed edits. We retain edits that improve

on the base warp without degrading the native result beyond a tolerance, preferring smaller edits when their effects are similar. If the base warp is harmful and no edit is acceptable, a uniform density provides a corrective target. If neither improvement nor correction is needed, the update is skipped. The resulting target density P^* and sample weight w define

$$\mathcal{L}_{\mathrm{S1}}(\theta) = w\,\mathrm{KL}(P^* \,\|\, P(Z_\theta))\,. \tag{3}$$

Each update treats the verified P^* and w as fixed values and differentiates only the predicted density to update θ, not the candidate-selection procedure. This is the purple proposal-learning loop: *the VJP supplies search directions, MLLM verification provides supervision, and the proposal network learns the verified sampling pattern*. The anchored objective, perturbations, and target-selection rules are given in Appendices A.2–A.4.

3.3 Stage II: WarpRank—Learning Native-Relative Selection

Constructing deployment-consistent candidates. As shown on the left of Stage II in Figure 2, we freeze VJP-Warp after Stage I, derive focus regions from its predicted density rather than answer gradients, and apply the same family of fixed edit operators. Together with the native image $I_0 = I$ and base warp I_1, these form

$$\mathcal{C} = (I_0, I_1, \ldots, I_9), \qquad I_{j+1} = \mathcal{W}(I, Z_\theta + \Delta_j), \quad j = 1, \ldots, 8. \tag{4}$$

The eight edits are weak, core, strong, expanded-context, compact, horizontal, vertical, and inverse variants, corresponding to Weak through Inverse in the figure. They modify sampling controls, not attention weights. Candidate construction takes only the image and question, uses the same procedure for selector training and deployment, and requires neither ground-truth answers nor backward gradients.

Supervising selection with decoded answers. Stage I verifies local proposals using answer likelihood, but better likelihood need not improve decoded task scores. We therefore let the frozen MLLM answer each candidate offline, obtain its task-metric score s_k, and define native-relative gain as

$$u_k = s_k - s_0, \qquad k = 0, \ldots, 9. \tag{5}$$

If a positive gain exists, all tied best candidates form the target set $\mathcal{P}^*$; otherwise, $\mathcal{P}^* = \{0\}$ explicitly supervises native fallback. Ground-truth answers are used only for offline scoring, not as inputs to candidate decoding. The figure's decoding path after selection does not imply that only the selected candidate is evaluated during training.

Learning lightweight candidate ranking. WarpRank extracts candidate features with a trainable lightweight RGB encoder and aligns native features using each candidate's sampling coordinates, comparing changes in evidence at corresponding locations. Question features and geometric descriptors provide task and spatial information. A set-ranking network jointly outputs $(r_0, \ldots, r_9) = h_\phi(\mathcal{C}, Q)$ without running the full MLLM for each candidate. Training combines set ranking and a native-relative margin:

$$\begin{aligned} \mathcal{L}_{\mathrm{list}} &= -\log \frac{\sum_{k \in \mathcal{P}^*} e^{r_k}}{\sum_{j=0}^{9} e^{r_j}}, \\ \mathcal{L}_{\mathrm{margin}} &= \Big[m_{\mathrm{tr}} + r_0 - \max_{k \in \mathcal{P}^*} r_k \Big]_+ . \end{aligned} \tag{6}$$

Here, $[x]_+ = \max(x, 0)$ and m_{tr} is a ranking margin. The first term increases the total probability of the optimal set, allowing multiple equally useful observations. The second applies only to positive-gain examples and ranks useful edits above the native image. We average the terms over the full batch and the positive-gain subset, respectively, sum them, and update only ϕ. Offline answer scores thus become a lightweight pre-decoding comparison capability; network details and the batch objective appear in Appendices A.5–A.6.

3.4 Final Selection and Inference

Figure 2(c) summarizes online proposal, selection, and single-pass decoding. Let $k^+ = \arg\max_{1 \le k \le 9} r_k$ be the highest-ranked warp. We select

$$k^* = \begin{cases} k^+, & r_{k^+} - r_0 \ge m_d \text{ and } p_{k^+} \ge \tau_p, \\ 0, & \text{otherwise.} \end{cases} \tag{7}$$

Here, m_d is the minimum ranking margin over the native image, p_k is a gate score obtained by applying a sigmoid mapping to the benefit branch, and τ_p is its acceptance threshold. Ranking adaptation uses no additional gain regression objective, and p_k is not interpreted as a calibrated improvement probability; Appendix A.6 specifies the mapping and thresholds. If the highest-ranked warp fails the gate, selection returns directly to the native image, making "no modification" an explicit decision.

The final answer is $\hat{Y} = R(I_{k^*}, Q)$. Native selection reuses the ViT features already extracted for proposal generation; warp selection requires one additional visual encoding of that image only. Both paths invoke the language decoder once. Deployment performs no VJP, answer verification, or per-candidate decoding. Its additional cost consists of proposal generation, candidate construction, lightweight ranking, and at most one extra visual encoding.

4 EXPERIMENTS

Our evaluation moves from overall answer quality to controlled analyses of proposal learning and view selection. We first compare QGVR with existing methods across five benchmarks and two frozen MLLMs using the complete designated evaluation splits (Table 2). We then use fixed evaluation subsets to examine the sources of improvement. Proposal controls test whether a learned proposal outperforms a fixed Gaussian under the same selector, and whether VJP directions produce a stronger candidate pool than random directions under matched training and verification settings (Table 3). Selection analysis compares the original image, the Stage I warp, WARPRANK, and Oracle@10 to distinguish proposal quality from the gains and remaining limitations of learned selection (Table 4). Training ablations assess alternative objectives and training strategies (Table 5). Finally, we examine post-decoding attention on four spatially annotated subsets as an explanatory analysis of the complete system, and measure online inference cost to characterize the computational overhead of proposal and selection.

4.1 MODELS, DATA, AND EVALUATION SETUP

Models. We evaluate InternVL3.5-8B (Wang et al., 2025) and LLaVA-1.5-7B (Liu et al., 2024), representing dynamic multi-tile and fixed-resolution visual interfaces. Both use BF16 without quantization, and the target MLLM remains frozen. The main InternVL experiments use at most 12 local 448×448 tiles; LLaVA retains its native CLIP preprocessing. QGVR adds neither views nor region tokens to final decoding. Additional settings are in Appendix B.

Table 1: Shared training data for Stages I and II.

Dataset	QA pairs
VizWiz-VQA-Grounding	6,494
DocReason25K	17,847
ChartQA	28,299
DocVQA	39,463
InfoVQA	23,946
TextVQA	34,602
Total	150,651

Training and adaptation data. Stages I and II use the same collection of 150,651 question–answer pairs (Table 1). Stage I uses answers to verify proposal targets; Stage II scores offline candidate predictions to supervise selection. Evaluation uses the validation or test partitions below. The new direction-supervision control restarts both arms from the same training pool after excluding exact evaluation-image matches (Appendix B.4).

Evaluation tasks and comparisons. TextVQA (Singh et al., 2019), DocVQA (Mathew et al., 2021), InfoVQA (Mathew et al., 2022), ChartQA (Masry et al., 2022), and GQA (Hudson & Manning, 2019) cover scene text, documents, charts, and object relations. The main comparison in Table 2 evaluates all examples in each benchmark's designated evaluation split, without subsampling. The proposal, selection, and training ablations in Tables 3–5 instead use a shared, fixed evaluation subset of 500 questions per dataset (2,500 questions in total). All task scores are reported on a percentage scale; the aggregate scores in these subset experiments are macro-averages with equal weight assigned to each of the five datasets. Evaluation splits, metrics, and decoding settings are specified in Appendix B.3.

4.2 Answer Quality across Tasks and Models

Native (Base LLM) is the unmodified-image reference. AttWarp constructs attention-guided warps (Dalal et al., 2026); ViCrop and SD-RPN add crops and learned region inputs, respectively (Zhang et al., 2025; Shi et al., 2026). Table 2 compares answer quality with each method's observation strategy. QGVR preserves Native's final visual-token interface; the controls below additionally fix the proposal architecture or candidate-selection pipeline.

Table 2: Multi-task answer quality (%). Stage I decodes the base warp; Stage II adds selection. Bold marks the highest reported score in each row.

Dataset	Base LLM	AttWarp	ViCrop	SD-RPN	QGVR (Stage I)	QGVR
InternVL3.5-8B						
$\text{TextVQA}_{\text{val}}$	79.17	80.17	79.41	77.51	79.79	**81.65**
$\text{InfoVQA}_{\text{val}}$	74.56	76.53	77.75	73.38	73.15	**81.34**
$\text{DocVQA}_{\text{val}}$	91.28	92.54	92.35	91.58	92.80	**94.84**
$\text{ChartQA}_{\text{test}}$	87.00	87.60	86.96	86.80	86.92	**87.88**
$\text{GQA}_{\text{test-dev}}$	61.58	63.84	62.66	62.82	63.92	**64.65**
LLaVA-1.5-7B						
$\text{TextVQA}_{\text{val}}$	48.02	58.10	57.20	58.80	53.82	**60.60**
$\text{InfoVQA}_{\text{val}}$	21.27	19.88	21.40	22.30	23.93	**26.04**
$\text{DocVQA}_{\text{val}}$	23.70	25.50	27.00	**34.20**	24.96	31.19
$\text{ChartQA}_{\text{test}}$	18.60	20.04	19.56	20.60	17.52	**22.52**
$\text{GQA}_{\text{test-dev}}$	60.64	63.70	60.98	61.24	60.84	**63.86**

Reallocation helps beyond multi-tile coverage. On InternVL, QGVR leads all five tasks, improving over Native by 6.78 points on InfoVQA, 3.56 on DocVQA, and 2.48 on TextVQA. On ChartQA, it reaches 87.88, exceeding Stage I (86.92), Native, and AttWarp by 0.96, 0.88, and 0.28 points, respectively. These gains show that question-guided reallocation remains useful beyond multi-tile coverage.

Transfer across answer decoders. On LLaVA, QGVR leads four of five tasks. ChartQA reaches 22.52, improving over Native, AttWarp, and ViCrop by 3.92, 2.48, and 2.96 points, respectively; InfoVQA and DocVQA gain 4.77 and 7.49 points over Native. Proposals use InternVL visual and question features, while LLaVA retains its native CLIP encoder and answer decoder. Table 2 compares complete configurations, including LLaVA's auxiliary encoder (Appendix B.3). Runtime measurements concern InternVL (Table 7).

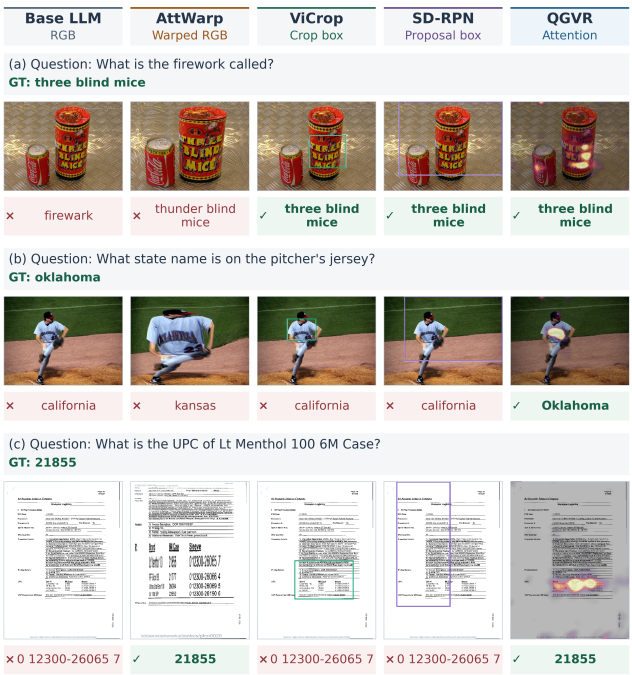

Figure 3: Visual processing strategies and actual answers. Checkmarks/crosses mark correctness; attention overlays are explanatory, not model inputs (Appendix B.6).

Qualitative examples. Figure 3 illustrates detail recognition and target-field selection: QGVR reads the jersey's "Oklahoma" and the document UPC.

4.3 Proposal Utility and Direction Supervision

Table 3 separates learned-proposal utility from direction-supervision quality. In (a), a fixed center Gaussian replaces the trained proposal while the warp, ten-candidate builder, and frozen WarpRank remain fixed. In (b), VJP and Random share architecture, initialization, data order, optimizer, 200-step training, and finite-verification budget; both retain density distillation. Oracle@10 is computed without loading WarpRank. Thus, (a) measures proposal utility within the deployed pipeline, and (b) tests VJP's contribution to candidate quality independently of Stage II.

Table 3: Proposal utility and direction supervision (five-task mean, %). (a) Proposal replacement with the fixed selector. (b) Oracle@10 after matched training; Δ is VJP minus Random. Panels use separate fixed checkpoints and decoding runs.

(a) Fixed-selector control

Proposal / path	Mean
Native	79.71
Fixed Gaussian + WarpRank	79.83
QGVR (learned)	**81.74**

(b) Matched direction control

Readout	VJP	Random	Δ [95% CI]
Oracle@10	**86.43**	85.79	+0.64 [0.14, 1.15]

2,500 questions; paired image-cluster bootstrap interval.

Learned spatial allocation improves on a fixed prior. With the same frozen WarpRank and candidate budget, the learned proposal raises the mean score from 79.83 to 81.74 (+1.91 points; 95% CI: [1.21, 2.60]). This gain establishes the value of learned spatial allocation over a question-independent Gaussian prior within the deployed selection pipeline.

VJP improves the candidate-selection upper bound. VJP raises Oracle@10 from 85.79 to 86.43 (+0.64 points; 95% paired CI: [0.14, 1.15]), improving four tasks and tying on ChartQA (Appendix B.4). Oracle@10 evaluates candidate answers directly, making this gain independent of learned selection. Under the matched training and verification budget, VJP therefore supplies more useful candidates for downstream selection within the same ten-candidate budget.

4.4 Selection Behavior and Training Ablations

Selection adds value with a fixed proposal. Table 4 holds the proposal model fixed and varies the candidate-selection rule. WarpRank raises the base-warp score from 80.34 to 81.74 (+1.40 points); Oracle@10 reaches 85.95, with 4.20 points of regret before rounding. Mean warp-selection rate, beneficial-switch precision, and harmful-switch rate are 18.08%, 19.37%, and 0.96%, respectively. Together with Table 3(b), this comparison separates candidate quality from the benefit of learned selection (Appendix B.8).

Table 4: Selection with a fixed proposal (five-task mean, %).

Path	Mean
Native	79.71
Stage I	80.34
WarpRank	81.74
Oracle@10	85.95

Decoded-answer selection complements proposal training. Table 5 compares complete training variants on the same evaluation set. QGVR exceeds the strongest alternative by 1.45 points. The comparison supports the combined training recipe; the separate matched direction control in Table 3(b) tests the specific contribution of VJP supervision.

Table 5: Training variants (five-task mean, %).

Variant / path	Mean
Native	79.71
NLL update	79.59
Anchor + feature credit	80.29
Anchor + competition	80.03
Joint distillation + ranking	79.43
Verified distillation	79.75
QGVR	81.74

4.5 Explanatory Analysis of Spatial Attention

We analyze post-decoding attention on TextVQA-X (Nagaraj Rao et al., 2021), VQS (Gan et al., 2017), TextOCR, and VizWiz-VQA-Grounding (Chen et al., 2022). *Focus-hit accuracy* tests whether the peak-attention token center falls within the target; *concentration* is one minus normalized attention entropy (Appendix B.7).

Table 6: Explanatory analysis: answer scores and post-decoding focus-hit accuracy (%) on four subsets.

Dataset	AttWarp		QGVR		Δ	
	Ans.	Focus hit	Ans.	Focus hit	Ans.	Focus hit
TextVQA-X	73.81	30.09	82.38	37.09	**+8.57**	**+7.00**
VQS	75.95	18.82	77.14	27.36	**+1.19**	**+8.55**
TextOCR	74.15	28.18	83.97	32.18	**+9.83**	**+4.00**
VizWiz	37.73	37.55	45.82	49.73	**+8.09**	**+12.18**

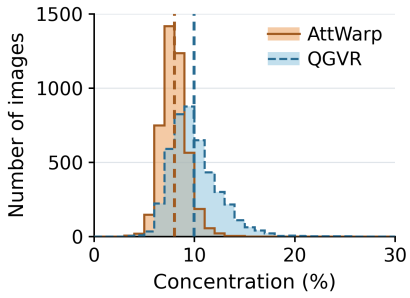

Figure 4: Concentration: AttWarp (orange) and QGVR (blue). Dashed vertical lines mark means.

Note: Δ uses unrounded scores. The histogram counts images in 1-point concentration bins, not target-region attention share. Both analyses concern the complete system (Appendix B.7).

Post-decoding response alignment. QGVR improves focus-hit accuracy by 4.00–12.18 percentage points on all four subsets (paired McNemar tests, $p < 0.05$). Answer scores also increase, although the VQS interval includes zero. Mean concentration rises from 7.99% to 9.95%, with overlapping distributions. These results characterize the complete system's response alignment; they do not directly measure proposal localization.

4.6 Efficiency

We time preprocessing, proposal and selection, visual encoding, and decoding on a fixed InternVL workload. The main configuration uses 180 A800 GPU-hours for offline labeling and two-stage training; the additional direction control is accounted for separately in Appendix B.4.

On this workload, QGVR retains 86.4% of Base LLM throughput, versus 29.6–66.8% for other methods, with 14.7% additional time and negligible peak-memory overhead (Table 7). These measurements reflect local implementations and module residency (Appendix C).

Table 7: Relative to Base LLM: throughput retained and cost increases (%).

Method	Throughput retained	Time increase	Memory increase
AttWarp	39.7	+150.9	+1.7
ViCrop	29.6	+235.2	+50.1
SD-RPN	66.8	+49.1	+5.0
QGVR	**86.4**	**+14.7**	**0.0**

5 Conclusion

QGVR combines verified VJP-guided proposal learning with native-relative selection, preserving the frozen MLLM's visual interface and a single decoding pass. Experiments support learned-proposal utility, candidate-pool quality, and selection gains with modest measured online overhead.

Limitations. Monotone warps can miss useful transformations or distort fine detail, and verified NLL gains need not improve decoding. Offline labeling requires multiple MLLM evaluations; transfer of the teacher-specific selector across models or domains remains a challenge.

AI USE STATEMENT

Generative AI tools assisted with manuscript drafting, translation, and revision; code generation and editing for implementation and evaluation; feedback on experimental design and interpretation of results; and figure preparation and paper formatting. The authors take responsibility for the final manuscript, implementation, experimental results, and conclusions, including all AI-assisted content and artifacts.

ETHICS STATEMENT

The method changes how visual evidence is sampled and selected, potentially suppressing small or atypical evidence. An MLLM-specific reranker may also fail under domain shift. High-stakes deployments should retain the native candidate, report harmful-switch rates, calibrate uncertainty, and avoid interpreting selected warps or heatmaps as causal explanations.

REPRODUCIBILITY STATEMENT

Method and evaluation details are provided in Sections 3–4 and Appendices A–B. The anonymous project page linked in the abstract provides a basic code framework with a symbolic workflow demonstration, together with interactive examples and their precomputed candidate outputs.

REFERENCES

Chongyan Chen, Samreen Anjum, and Danna Gurari. Grounding answers for visual questions asked by visually impaired people. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 19098–19107, 2022. URL https://openaccess.thecvf.com/content/CVPR2022/html/Chen_Grounding_Answers_for_Visual_Questions_Asked_by_Visually_Impaired_People_CVPR_2022_paper.html.

Liang Chen, Haozhe Zhao, Tianyu Liu, Shuai Bai, Junyang Lin, Chang Zhou, and Baobao Chang. An image is worth 1/2 tokens after layer 2: Plug-and-play inference acceleration for large vision-language models. In *European Conference on Computer Vision*, 2024.

Rohan Choudhury, JungEun Kim, Jinhyung Park, Eunho Yang, László A. Jeni, and Kris M. Kitani. Accelerating vision transformers with adaptive patch sizes. *arXiv preprint arXiv:2510.18091*, 2026.

Dwip Dalal, Gautam Vashishtha, Utkarsh Mishra, Jeonghwan Kim, Madhav Kanda, Hyeonjeong Ha, Svetlana Lazebnik, Heng Ji, and Unnat Jain. Constructive distortion: Improving mllms with attention-guided image warping. In *International Conference on Learning Representations*, 2026.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=YicbFdNTTy.

Xingyu Fu, Minqian Liu, Zhengyuan Yang, John Richard Corring, Yijuan Lu, Jianwei Yang, Dan Roth, Dinei Florencio, and Cha Zhang. Refocus: Visual editing as a chain of thought for structured image understanding. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 17783–17805, 2025. URL https://proceedings.mlr.press/v267/fu25d.html.

Chuang Gan, Yandong Li, Haoxiang Li, Chen Sun, and Boqing Gong. VQS: Linking segmentations to questions and answers for supervised attention in VQA and question-focused semantic segmentation. In *Proceedings of the IEEE International Conference on Computer Vision*, pp. 1811–1820, 2017. URL https://openaccess.thecvf.com/content_iccv_2017/html/Gan_VQS_Linking_Segmentations_ICCV_2017_paper.html.

Anwen Hu, Haiyang Xu, Jiabo Ye, Ming Yan, Liang Zhang, Bo Zhang, Ji Zhang, Qin Jin, Fei Huang, and Jingren Zhou. mPLUG-DocOwl 1.5: Unified structure learning for OCR-free document understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2024*, pp. 3096–3120. Association for Computational Linguistics, 2024. doi: 10.18653/v1/2024.findings-emnlp.175. URL https://aclanthology.org/2024.findings-emnlp.175/.

Drew A. Hudson and Christopher D. Manning. GQA: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 6700–6709, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Hudson_GQA_A_New_Dataset_for_Real-World_Visual_Reasoning_and_Compositional_CVPR_2019_paper.html.

Max Jaderberg, Karen Simonyan, Andrew Zisserman, and Koray Kavukcuoglu. Spatial transformer networks. In *Advances in Neural Information Processing Systems*, 2015.

Juho Lee, Yoonho Lee, Jungtaek Kim, Adam Kosiorek, Seungjin Choi, and Yee Whye Teh. Set transformer: A framework for attention-based permutation-invariant neural networks. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 3744–3753, 2019. URL https://proceedings.mlr.press/v97/lee19d.html.

Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. Improved baselines with visual instruction tuning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26296–26306, 2024. URL https://openaccess.thecvf.com/content/CVPR2024/html/Liu_Improved_Baselines_with_Visual_Instruction_Tuning_CVPR_2024_paper.html.

Ahmed Masry, Do Xuan Long, Jia Qing Tan, Shafiq Joty, and Enamul Hoque. ChartQA: A benchmark for question answering about charts with visual and logical reasoning. In *Findings of the Association for Computational Linguistics: ACL 2022*, pp. 2263–2279. Association for Computational Linguistics, 2022. doi: 10.18653/v1/2022.findings-acl.177. URL https://aclanthology.org/2022.findings-acl.177/.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. DocVQA: A dataset for VQA on document images. In *IEEE Winter Conference on Applications of Computer Vision*, pp. 2200–2209, 2021. URL https://openaccess.thecvf.com/content/WACV2021/html/Mathew_DocVQA_A_Dataset_for_VQA_on_Document_Images_WACV_2021_paper.html.

Minesh Mathew, Viraj Bagal, Rubèn Tito, Dimosthenis Karatzas, Ernest Valveny, and C. V. Jawahar. InfographicVQA. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 1697–1706, 2022. URL https://openaccess.thecvf.com/content/WACV2022/html/Mathew_InfographicVQA_WACV_2022_paper.html.

Varun Nagaraj Rao, Xingjian Zhen, Karen Hovsepian, and Mingwei Shen. A first look: Towards explainable TextVQA models via visual and textual explanations. In *Proceedings of the Third Workshop on Multimodal Artificial Intelligence*, pp. 19–29. Association for Computational Linguistics, 2021. doi: 10.18653/v1/2021.maiworkshop-1.4. URL https://aclanthology.org/2021.maiworkshop-1.4/.

Duy Nguyen, Archiki Prasad, Elias Stengel-Eskin, and Mohit Bansal. GrAInS: Gradient-based attribution for inference-time steering of LLMs and VLMs. In *Proceedings of the 64th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 46523–46543. Association for Computational Linguistics, 2026. doi: 10.18653/v1/2026.acl-long.2159. URL https://aclanthology.org/2026.acl-long.2159/.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Adrià Recasens, Petr Kellnhofer, Simon Stent, Wojciech Matusik, and Antonio Torralba. Learning to zoom: A saliency-based sampling layer for neural networks. In *European Conference on Computer Vision*, 2018.

Pier Giuseppe Sessa, Robert Dadashi, Léonard Hussenot-Desenonges, Johan Ferret, Nino Vieillard, Alexandre Rame, Bobak Shahriari, Sarah Perrin, Abram Friesen, Geoffrey Cideron, Sertan Girgin, Piotr Stanczyk, Andrea Michi, Danila Sinopalnikov, Sabela Ramos Garea, Amélie Héliou, Aliaksei Severyn, Matthew Hoffman, Nikola Momchev, and Olivier Bachem. BOND: Aligning LLMs with best-of-N distillation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/947f37882a394140f7add476bb99d1d3-Abstract-Conference.html.

Haozhan Shen, Kangjia Zhao, Tiancheng Zhao, Ruochen Xu, Zilun Zhang, Mingwei Zhu, and Jianwei Yin. Zoomeye: Enhancing multimodal llms with human-like zooming capabilities through tree-based image exploration. In *Conference on Empirical Methods in Natural Language Processing*, 2025.

Yuheng Shi, Xiaohuan Pei, Minjing Dong, and Chang Xu. Catching the details: Self-distilled RoI predictors for fine-grained MLLM perception. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=Cox6AaRyan.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards VQA models that can read. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 8317–8326, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Singh_Towards_VQA_Models_That_Can_Read_CVPR_2019_paper.html.

Mukund Sundararajan, Ankur Taly, and Qiqi Yan. Axiomatic attribution for deep networks. In *International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3319–3328, 2017. URL https://proceedings.mlr.press/v70/sundararajan17a.html.

Weiyun Wang, Zhangwei Gao, Lixin Gu, Hengjun Pu, Long Cui, Xingguang Wei, Zhaoyang Liu, Linglin Jing, Shenglong Ye, Jie Shao, et al. InternVL3.5: Advancing open-source multimodal models in versatility, reasoning, and efficiency. *arXiv preprint arXiv:2508.18265*, 2025. URL https://arxiv.org/abs/2508.18265.

Penghao Wu and Saining Xie. V*: Guided visual search as a core mechanism in multimodal llms. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2024.

Hanxun Yu, Wentong Li, Xuan Qu, Song Wang, Junbo Chen, and Jianke Zhu. Visiontrim: Unified vision token compression for training-free mllm acceleration. In *International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=57IXIg6nZ0.

Hongyi Yuan, Zheng Yuan, Chuanqi Tan, Wei Wang, Songfang Huang, and Fei Huang. RRHF: Rank responses to align language models with human feedback. In *Advances in Neural Information Processing Systems*, volume 36, pp. 10935–10950, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/23e6f78bdec844a9f7b6c957de2aae91-Abstract-Conference.html.

Jiarui Zhang, Mahyar Khayatkhoei, Prateek Chhikara, and Filip Ilievski. Mllms know where to look: Training-free perception of small visual details with multimodal llms. In *International Conference on Learning Representations*, 2025.

Yuan Zhang, Chun-Kai Fan, Junpeng Ma, Wenzhao Zheng, Tao Huang, Kuan Cheng, Denis Gudovskiy, Tomoyuki Okuno, Yohei Nakata, Kurt Keutzer, and Shanghang Zhang. Sparsevlm: Visual token sparsification for efficient vision-language model inference. *arXiv preprint arXiv:2410.04417*, 2024.

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. In *International Conference on Learning Representations*, 2026.

A METHOD DETAILS

This appendix expands the warp parameterization, answer objectives, structured perturbations, distillation targets, and ranking losses in Section 3. Data composition is given in Section 4, and training settings in Appendix B.

Parameter updates and differentiation. We use θ for proposal-network parameters and ϕ for selector parameters. Stage I updates only θ; Stage II freezes the proposal network and updates only ϕ. The target MLLM's parameters never change. Computing a VJP still requires differentiating the answer objective through the target MLLM with respect to the sampling-control map. This is distinct from updating the MLLM's weights.

Each training step first computes reference answer probabilities or candidate targets, then fixes these values while optimizing the current prediction. Thus, reference probabilities $q_t^{(m)}$ are constant when differentiating sampling variables, and the selected density P^* is constant when updating the proposal network. We state the optimization variables explicitly below rather than introducing additional computational-graph notation.

A.1 FULL-FIELD DENSITY AND CONDITIONAL INVERSE-CDF WARPING

For Z defined on a low-resolution spatial grid Ω, construct a positive density and its probability form:

$$B(Z) = \mathrm{softplus}(Z) + \varepsilon_0, \qquad A(Z) = \frac{B(Z)}{\mathrm{mean}_\Omega\, B(Z)}, \qquad P(Z) = \frac{B(Z)}{\sum_\Omega B(Z)}. \tag{8}$$

Here, $\varepsilon_0 = 10^{-9}$ maintains positivity, and $P(Z)$ is used for density distillation. To construct the sampling grid, we bicubically interpolate A to the source canvas, clamp numerical negative values, take its square root, and add a positive floor to obtain the transport density ρ. The square-root transform tempers strong density peaks; the distillation target P should be distinguished from the actual transport density ρ.

For clarity, express source coordinates on $[0,1]^2$ and define

$$\begin{aligned} p_y(y) &= \frac{\int_0^1 \rho(x,y)\,dx}{\int_0^1\int_0^1 \rho(x',y')\,dx'\,dy'}, & F_y(y) &= \int_0^y p_y(t)\,dt, \\ p_{x|y}(x \mid y) &= \frac{\rho(x,y)}{\int_0^1 \rho(t,y)\,dt}, & F_{x|y}(x \mid y) &= \int_0^x p_{x|y}(t \mid y)\,dt. \end{aligned} \tag{9}$$

The source coordinates and resampled image at output position (u, v) are

$$\begin{aligned} y_s = F_y^{-1}(v), \qquad x_s = F_{x|y}^{-1}(u \mid y_s), \\ I_Z(u,v) = \mathrm{Bilinear}(I; x_s, y_s). \end{aligned} \tag{10}$$

The implementation uses piecewise-linear inversion of the discrete marginal CDF and interpolates conditional column densities between neighboring source rows. A uniform density returns identity sampling; a nonuniform density induces monotone triangular conditional transport. Column sampling depends on the selected source row, rather than a separable warp from two independent axial marginals. Each candidate retains the native canvas and tile layout.

A.2 MULTI-ANSWER VERIFICATION AND THE ANCHORED VJP OBJECTIVE

For an answer $Y^{(m)}$, define the answer-token-averaged negative log-likelihood as

$$\begin{aligned} \ell_m(I') &= -\frac{1}{|Y^{(m)}|}\sum_t \log p_R(y_t^{(m)} \mid y_{<t}^{(m)}, Q, I'), \\ \ell(I') &= -T \log \sum_m w_m e^{-\ell_m(I')/T}. \end{aligned} \tag{11}$$

Here, $w_m \geq 0$ and $\sum_m w_m = 1$ are answer weights, and $T = 0.5$ controls multi-answer aggregation. This objective compares the answer utility of finite candidates.

The VJP uses an objective anchored to native predictions. Let $p_{0,t}^{(m)}$ be the native-image probability of the current answer token conditioned on the correct answer prefix. Construct the reference target

$$\begin{aligned} \alpha_t^{(m)} &= \mathrm{clip}(1 - p_{0,t}^{(m)}, 0.05, 0.50), \\ q_t^{(m)} &= \left[p_{0,t}^{(m)} + \alpha_t^{(m)} (1 - p_{0,t}^{(m)}) \right]. \end{aligned} \tag{12}$$

Grouping output probabilities into "the answer token" and "all remaining tokens" gives

$$\begin{aligned} a_m(Z) &= \frac{1}{|Y^{(m)}|} \sum_t \mathrm{KL}\Big(\mathrm{Ber}(q_t^{(m)}) \parallel \mathrm{Ber}(p_t^{(m)}(Z)) \Big), \\ \mathcal{L}_{\mathrm{anc}}(\mathcal{W}(I, Z), Q, \mathcal{Y}) &= -T \log \sum_m w_m e^{-a_m(Z)/T}. \end{aligned} \tag{13}$$

Here, $p_t^{(m)}(Z)$ is the corresponding answer-token probability under the warped observation; probabilities are clamped to positive values for numerical computation. Equation (2) differentiates this scalar objective through the frozen visual encoder, connector, and language decoder back to the sampling-control map Z. Only Z is differentiated; none of these model weights are updated. After directions and candidates are obtained, proposal optimization treats the candidates and their utilities as precomputed values, without differentiating candidate construction or filtering.

A.3 Eight Structured Perturbations and the Candidate Budget

Let $\mathcal{A}_j$ denote average pooling with a $j \times j$ window and $\mathcal{N}(F) = F / \max(\max_\Omega F, 10^{-6})$ normalize a nonnegative map. The training focus map is constructed from G:

$$F_{\mathrm{tr}} = \mathcal{N}(\mathcal{A}_3(S)), \qquad S = \begin{cases} \mathrm{ReLU}(-G), & \sum_\Omega \mathrm{ReLU}(-G) \geq 0.05 \sum_\Omega |G|, \\ |G|, & \text{otherwise.} \end{cases} \tag{14}$$

Stage II and deployment use $F_{\mathrm{dep}} = \mathcal{N}(\mathcal{A}_3(\mathrm{ReLU}(A(Z_\theta) - 1)))$. If its positive mass is insufficient, $|B(Z_\theta) - \mathrm{mean}\, B(Z_\theta)|$ replaces the focus map before pooling. Both sources supply only the focus input F to the same operator family.

Define centered RMS normalization as

$$\mathcal{U}(F) = \frac{F - \mathrm{mean}_\Omega\, F}{\max\Big(\sqrt{\mathrm{mean}_\Omega (F - \mathrm{mean}_\Omega\, F)^2}, 10^{-6} \Big)}. \tag{15}$$

The core, context, compact, and two axial directions are

$$\begin{aligned} C &= \mathcal{U}(F), \qquad C_{\mathrm{ctx}} = \mathcal{U}(\mathcal{A}_7(F)), \qquad C_{\mathrm{cmp}} = \mathcal{U}(F^2), \\ C_x &= \mathcal{U}(\mathrm{Broadcast}_x(\mathrm{mean}_x\, F)), \\ C_y &= \mathcal{U}(\mathrm{Broadcast}_y(\mathrm{mean}_y\, F)). \end{aligned} \tag{16}$$

C_x spreads each row's focus mass horizontally, while C_y spreads each column's focus mass vertically. The eight edits follow a fixed semantic order:

$$(\Delta_1, \ldots, \Delta_8) = \epsilon(0.5C, C, 1.5C, C_{\mathrm{ctx}}, C_{\mathrm{cmp}}, C_x, C_y, -C), \qquad \epsilon = 0.15. \tag{17}$$

The corresponding observations are $I_{j+1} = \mathcal{W}(I, Z_\theta + \Delta_j)$, $j = 1, \ldots, 8$. Together with the native image I_0 and base proposal I_1, the full candidate family contains ten observations.

Stage I samples without replacement from the eight edits, selecting $2/4/8$ with probabilities $0.7/0.2/0.1$, respectively. The native image and base proposal are always verified. Budget levels are synchronized within each distributed batch; sample identifiers and random seeds determine the specific subset. Native-first filtering retains all examples answered incorrectly on the native image and a deterministic 5% subset of those answered correctly. Stage II offline labeling and deployment both use the full ten-candidate set.

A.4 Valid Edits, Identity Fallback, and Proposal Distillation

Let $\mathcal{J} \subseteq \{2, \dots, 9\}$ contain the edit indices verified in the current step, and let $g_k = \ell(I_0) - \ell(I_k)$. The valid and near-optimal sets are

$$\begin{aligned} \mathcal{V} &= \{k \in \mathcal{J} : g_k - g_1 \geq \delta_{\text{rel}},\ g_k \geq -\eta\}, \\ \mathcal{V}_{\text{near}} &= \{k \in \mathcal{V} : g_k \geq \max_{j \in \mathcal{V}} g_j - \delta_{\text{near}}\}. \end{aligned} \tag{18}$$

We use $\delta_{\text{rel}} = 0.01$, $\eta = 0.01$, and $\delta_{\text{near}} = 0.02$. When $\mathcal{V}$ is nonempty, select $k^\dagger = \arg\min_{k \in \mathcal{V}_{\text{near}}} \text{RMS}(\Delta_{k-1})$. This measures edit magnitude in logit space, not pixel distortion. The target and sample weight are

$$(P^*, w) = \begin{cases} (P(Z_\theta + \Delta_{k^\dagger - 1}), 1), & \mathcal{V} \neq \varnothing, \\ (P_{\text{unif}}, 0.25), & \mathcal{V} = \varnothing,\ g_1 < -0.05, \\ (P(Z_\theta), 0), & \text{otherwise}. \end{cases} \tag{19}$$

P_{unif} is the uniform distribution. The target is determined from the pre-update proposal and verification results and remains fixed during the update. The KL matching in Equation (3) expands to

$$\mathcal{L}_{\text{S1}}(\theta) = w \sum_{i \in \Omega} P_i^* \log \frac{P_i^*}{P_i(Z_\theta)}.$$

Here, Ω is the spatial grid and i indexes its positions. When differentiating the loss with respect to θ, only the predicted density $P(Z_\theta)$ varies; P^* and w remain fixed. Training transfers the verified density to the proposal network without requiring it to reproduce the search procedure.

A.5 Proposal Network and RGB Proxy Features

Proposal network. Each 448×448 tile uses the raw $32 \times 32 \times 1024$ ViT grid from the final layer of frozen InternVL. Features pass through a $1024 \to 1024$ alignment adapter, spatial coordinate encoding, four spatial refinement blocks, and a tile Transformer. The question prefix's 4096-dimensional word embeddings are projected to 1024 dimensions and fused through two encoding layers and two tile–question cross-attention blocks. Prior, residual, and strength heads jointly construct Z_θ and the density in Equation (8). Answer tokens are excluded from the question prefix. Native ViT features and question embeddings are fixed inputs; training adjusts only the added proposal network.

Candidate-comparison features. The RGB proxy has width 384 and uses a 14×14 patch-embedding stem, four local ConvNeXt blocks, and spatial pooling to produce a 16×16 grid per tile. These grids are assembled into E_k according to the canvas layout. Let Γ_k denote the candidate's source sampling coordinates. First align the native grid as $\bar{E}_0^k = \text{GridSample}(E_0, \Gamma_k)$, then construct

$$X_k = \text{MLP}\big[E_k, \bar{E}_0^k, E_k - \bar{E}_0^k, |E_k - \bar{E}_0^k|, E_k \odot \bar{E}_0^k\big] + e(\Gamma_k). \tag{20}$$

Here, e projects coordinates, and E_k is the output of the trainable proxy, not candidate features from the full frozen ViT. Question cross-attention, feed-forward transformations, and attention pooling produce candidate embeddings, augmented with projected eight-dimensional geometric descriptors. For edited candidates, these describe magnitude, the two focus-center coordinates, spread along two axes, relative area, peak value, and normalized entropy. A two-layer permutation-equivariant set Transformer outputs ranking scores. Coordinates, labels, and native indicators are permuted together with the candidates.

A.6 Task-Ranking Objective and Deployment Gating

Let $u_k = s_k - s_0$ and $u^\star = \max_k u_k$. The set-valued supervision target is

$$\mathcal{P}^* = \begin{cases} \{k : u_k \geq u^\star - \varepsilon_s\}, & u^\star > \varepsilon_s, \\ \{0\}, & \text{otherwise}, \end{cases} \qquad \varepsilon_s = 10^{-8}. \tag{21}$$

Positive-gain examples may have multiple tied best observations; examples without positive gain explicitly target the native image. The set-ranking and native-relative margin terms are given in

Equation (6), with the margin applied only to positive-gain examples and $m_{\text{tr}} = 0.1$. For a training batch $\mathcal{B}$ and its positive-gain subset $\mathcal{B}_+$, the final objective is

$$\mathcal{L}_{\text{S2}} = \frac{1}{|\mathcal{B}|} \sum_{n \in \mathcal{B}} \mathcal{L}_{\text{list}}^{(n)} + \frac{1}{\max(1, |\mathcal{B}_+|)} \sum_{n \in \mathcal{B}_+} \mathcal{L}_{\text{margin}}^{(n)}. \tag{22}$$

The margin constrains ranking logits; it does not require task-score improvement greater than 0.1. This multi-dataset adaptation uses no additional gain-regression, gain-sign classification, or ViT-difference distillation loss.

The benefit-branch output b_k is mapped to a gate score by

$$p_k = \sigma(ab_k + c), \qquad a > 0. \tag{23}$$

Here, σ is the sigmoid function and a, c are affine coefficients. The highest-ranked edit is accepted or rejected by Equation (7). m_d is its minimum ranking margin over the native image, and τ_p is the benefit-branch acceptance threshold. Failure of the top edit leads directly to native fallback. The final observation is I_{k^*}, and $R(I_{k^*}, Q)$ generates the final answer. For InternVL, the benefit branch is initialized from an earlier selector checkpoint trained on cached native-relative task gains using gain regression and gain-sign and top-edit classification losses. It consumes candidate–native differences in pooled and set-encoded features. During the subsequent ranking adaptation, the branch parameters and affine coefficients remain fixed, but the shared feature extractor continues to update; hence b_k and p_k can change even with fixed branch parameters. The gate score is not interpreted as a calibrated probability of answer improvement.

B Experimental Reproducibility Details

B.1 Dataset Sources

The shared Stage I and Stage II data in Table 1 combine TextVQA (Singh et al., 2019), DocVQA (Mathew et al., 2021), InfographicVQA (abbreviated as InfoVQA) (Mathew et al., 2022), ChartQA (Masry et al., 2022), DocReason25K, and VizWiz-VQA-Grounding. DocReason25K is released with mPLUG-DocOwl 1.5 (Hu et al., 2024); VizWiz-VQA-Grounding is the answer-grounding dataset of Chen et al. (2022). Both stages use the same image–question–answer collection without spatial annotations. Table 1 reports the records used in our training collection, not the total sizes of the original datasets. The five answer-quality benchmarks use the source datasets cited in Section 4.1; grounding-subset provenance and derived annotations are specified in Appendix B.7.

B.2 Model and Optimization Settings

Table 1 lists the shared training data for both stages. The proposal module has 107.3M trainable parameters. Ground-truth answers construct the anchored VJP, and the NLL of the supplied answer sequence verifies the native image, base proposal, and sampled finite edits. Stage II freezes the proposal module and trains a 21.1M-parameter selector. Offline cached candidate task scores supervise set ranking; positive-gain examples additionally incur a margin loss between the best edit and the native image. The margin of 0.1 applies to learned ranking scores, not to a required task-score improvement of 0.1. Offline candidate labeling and training across both stages took 22.5 hours on eight NVIDIA A800 GPUs, for a total of 180 GPU-hours.

Main optimization. Table 8 summarizes the main InternVL optimization runs. Both use AdamW with $(\beta_1, \beta_2) = (0.9, 0.98)$ and $\epsilon = 10^{-8}$, with the reader frozen. The separate 200-step VJP–Random control is described in Appendix B.4.

The evaluated checkpoints are InternVL3.5-8B (Wang et al., 2025) and LLaVA-1.5-7B (Liu et al., 2024), rather than earlier releases in either family. We use ViT to denote the vision-transformer architecture (Dosovitskiy et al., 2021); each answer decoder retains its native image interface. The Stage I transfer evaluation additionally uses an InternVL auxiliary encoder for LLaVA proposals, as specified in Appendix B.3. InternVL uses BF16 without quantization, at most 12 dynamic local tiles per input, a maximum sequence length of 6,144, and an implementation generation cap of 1,024 tokens. The evaluations described below use the explicit 32-token cap in their protocols.

Table 8: Optimization settings for the main InternVL runs. Stage II entries refer to ranking adaptation with a frozen proposal. Update counts are optimizer steps.

Setting	Stage I proposal	Stage II ranking adaptation
Optimizer	AdamW	AdamW
Peak learning rate	2×10^{-6}	2×10^{-5}
Weight decay	0.01	0
Learning-rate schedule	Linear warmup + cosine	Linear warmup + cosine
Warmup ratio	0.02	0.02
Nominal global batch size	64	64

Preprocessing and batch settings are specified separately for the answer-quality, grounding, and runtime experiments. The selection gate threshold is 10^{-7} and the minimum ranking margin is zero, making this deployment configuration close to ranking-only selection. LLaVA uses a separately adapted selector with a gate threshold of 0.0875 and a zero minimum ranking margin, retaining native CLIP (Radford et al., 2021) image preprocessing. The training objective, gate mapping, and threshold usage are detailed in Appendix A.6.

B.3 Evaluation Protocols and Fixed-Prior Control

Scores in Table 2 are averaged over three evaluation runs. The Stage I column in Table 2 always decodes the base warp, without loading WarpRank. Both answer models use the same frozen proposal checkpoint. Each model is evaluated on 5,000 TextVQA validation, 2,801 InfoVQA validation, 5,349 DocVQA validation, 2,500 ChartQA test, and 12,578 GQA balanced test-dev questions. Stage I evaluation uses BF16 greedy decoding with at most 32 new tokens. For LLaVA, proposal inference uses the auxiliary InternVL visual encoder and question representation; the resulting RGB image is processed by LLaVA's native CLIP encoder and answer decoder. This evaluates transfer of the proposal across answer models, with the auxiliary encoder retained at deployment. The runtime measurements in Table 7 concern InternVL.

The task metrics are ten-answer leave-one-out VQA accuracy for TextVQA, ANLS for InfoVQA and DocVQA, 5% relaxed accuracy for ChartQA, and exact accuracy for GQA. The controlled comparisons in Tables 3–5 use a fixed set of 2,500 evaluation questions, 500 per dataset, with ChartQA sampled from test. All arms within each comparison share the prompt, scorer, preprocessing, and decoding settings. Appendix B.8 specifies sampling and aggregation; Appendix B.4 specifies the separate 200-step training control.

Question-independent replacement. On the normalized coordinates $(x, y) \in [-1, 1]^2$, define

$$g(x, y) = \exp\left(-\frac{x^2 + y^2}{2(0.5)^2}\right), \qquad d(x, y) = 0.35 + 0.65\frac{g(x, y)}{\mathrm{mean}(g)}. \tag{24}$$

The center, variance, and mixture weights are fixed before evaluation. The density uses the original proposal grid resolution and contains no image, question, answer, or learned-proposal information. We pass $\mathrm{softplus}^{-1}(d^2 - 10^{-9})$ through the unchanged candidate builder; the existing square-root density mapping then recovers d. The inverse-CDF warp, perturbation rules, and selector are otherwise identical across arms.

Paired decoding and scoring. The fixed Gaussian is evaluated on all 2,500 questions with the same frozen WarpRank, ten-candidate construction, preprocessing, and decoding settings as the common-candidate evaluation. Native and learned-proposal answers are reused from that run. When the Gaussian selector chooses candidate zero, it receives the same Native answer, solely according to its selection index. Independent raw Gaussian-path decodes are retained. No reference answer enters selection. The reported mean equally weights the five task-specific percentage scores. Confidence intervals use 10,000 paired image-cluster bootstrap resamples within each dataset, followed by equal-weight averaging across datasets. The learned proposal gains 1.91 mean task-score points over the Gaussian control (95% CI: [1.21, 2.60]).

Selector behavior under proposal substitution. The fixed Gaussian has a mean warp-selection rate of 11.12%, beneficial-switch precision of 12.51%, and harmful-switch rate of 1.12%; the learned proposal yields 18.08%, 19.37%, and 0.96%, respectively. These are equal-weight means of the five dataset rates, with denominators defined in Appendix B.8.

B.4 Matched VJP versus Random Direction Supervision

Controlled intervention. We train two 107.3M-parameter proposal networks from the same random initialization with a frozen InternVL3.5-8B reader. The Random arm replaces only the VJP field with an independent signed standard-Gaussian field, passed through the same focus, smoothing, normalization, and structured-edit operators. Both arms retain finite-edit answer-NLL verification, canonical target construction, and density KL distillation. Random directions use a local generator indexed by sample identity, execution index, and direction seed; they do not alter the global training random stream. Random therefore removes reader differentiation while preserving verified proposal learning.

Candidate evaluation. Both checkpoints are evaluated on the same five-task, 2,500-question set, using BF16 greedy decoding and at most 32 new tokens. The prompt suffix is "Answer the question using a single word or phrase." Each image uses at most 12 local tiles and a global context view. For ChartQA test edits, the global view is taken from the edited image; the other tasks retain the native global view. The candidate decode batch is 10 and the ViT microbatch is 24. Each arm generates 25,000 answers: Native, the base warp, and the same eight structured edits per question. Native answers are identical across arms. Reference answers enter only post-decoding scoring; no selector is loaded. Oracle@10 measures the best available candidate answer. These fresh checkpoints and decoding runs are compared within this experiment.

Table 9: Candidate Oracle@10 after matched 200-step direction supervision, including 50 warmup steps. Each dataset has 500 held-out questions. Scores are percentages; Δ is VJP minus Random, computed before rounding.

Dataset	VJP	Random	Δ
TextVQA	86.94	86.68	+0.26
InfoVQA	88.41	87.17	+1.24
DocVQA	97.41	96.90	+0.51
ChartQA	89.80	89.80	+0.00
GQA	69.60	68.40	+1.20
Mean	86.43	85.79	+0.64

Paired evaluation and candidate utility. For the fixed trained checkpoints, 95% confidence intervals use 10,000 paired image-cluster bootstrap resamples of the evaluation set within each dataset, followed by equal-weight averaging of the five task scores. Dataset streams are independently seeded from the SHA-256 digest of the string `20260913:dataset`. VJP improves mean Oracle@10 by 0.64 points (95% CI: [0.14, 1.15]), with higher scores on four tasks and a tie on ChartQA. With the candidate count and verification budget held fixed, this result supports VJP as a more effective direction signal for increasing the attainable answer score within the candidate pool.

B.5 Controlled Training Ablations

The five variants in Table 5 use the same 2,500-question evaluation set as Table 4, with 500 questions per dataset and equal-weight averaging of task scores. Saved checkpoints use the same reader, prompts, preprocessing, BF16 greedy decoding (32 new tokens; batch size 20), and scorers. Native and QGVR use the common-candidate run's corresponding answers; Native fallback is scored with the shared Native answer.

NLL update uses answer-NLL gradients. *Anchor + feature credit* adds visual-feature credit, and *Anchor + competition* adds Native-relative competition. *Joint distillation + ranking* learns proposal distillation and candidate ranking jointly; *Verified distillation* learns finite-warp targets verified by the frozen MLLM. QGVR combines proposal learning and selection. These comparisons evaluate the

resulting systems. Teacher-forced NLL and decoded task scores are distinct objectives; verified NLL improvement need not improve the decoded answer.

B.6 Qualitative Examples and Visualization Protocol

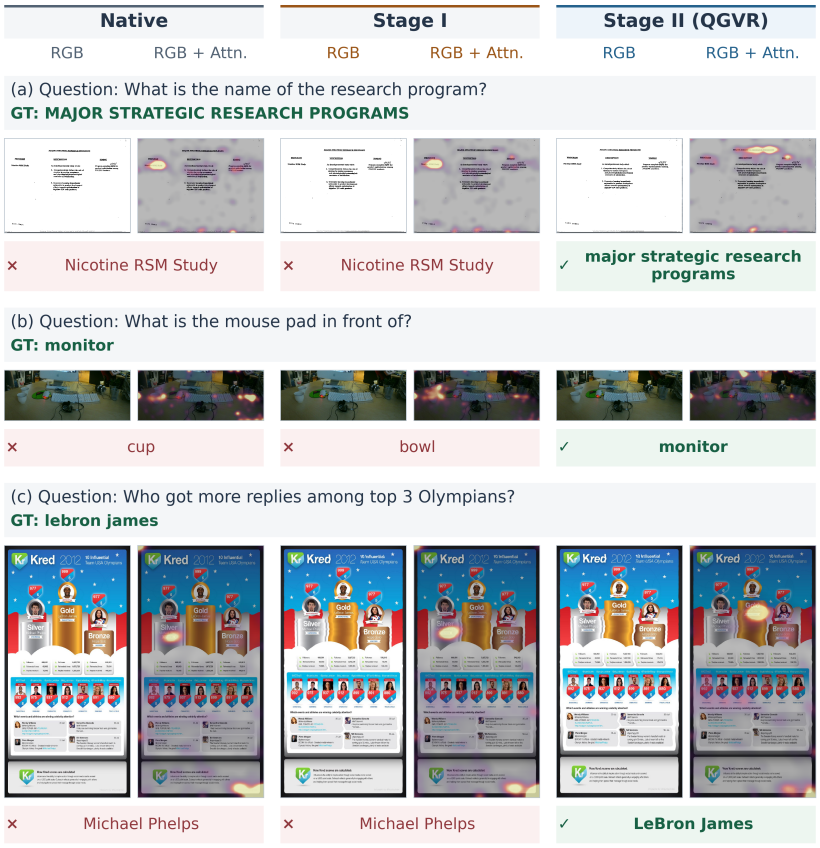

Figure 5: Stage II corrects errors from Native and Stage I on document, relation, and infographic questions. Each path shows its RGB input, decoding attention, and actual answer; Stage II is the deployed selection. These cases illustrate answer utility, rather than proving proposal localization.

Native–warp comparison in the introduction. Figure 1 uses the archived TextVQA example TV-05 (question ID 39317) and shows the non-native candidate actually selected at deployment. The four columns show the unwarped input canvas, its attention overlay, the selected warped canvas, and its corresponding attention overlay. Warped attention is not overlaid on an unaligned native image. Native and QGVR each use the frozen MLLM's attention during their own answer generation, from decoder layer 16 (zero-indexed). Both paths share a color scale. The archived display applies Gaussian smoothing with $\sigma = 1.2$ on the token grid followed by bicubic upsampling, and caps the scale at the larger of the two smoothed 99th-percentile values. It uses the magma colormap, a 1.35-power response transform, 72% background brightness, and a maximum overlay opacity of 78%. We only rearrange the archived overlays, without further hotspot enhancement. These display transforms affect neither model inference nor quantitative statistics.

Multi-method answer comparison. Figure 3 shows two TextVQA questions and one DocVQA question concerning a fireworks package, a jersey, and a document UPC. These are selected illustrations under method-specific preprocessing, not a random sample for estimating accuracy. AttWarp shows the exported warp; ViCrop shows att-CROP proposal boxes; SD-RPN boxes enclose proposed regions and need not equal its internal region-token mask. The QGVR column maps the selected path's decoding attention back to native-image coordinates through its sampling grid. It displays

Table 10: Answer scores (%) for three paths on grounding subsets. Stage I only directly uses the base proposal without candidate selection. AttWarp results are in Table 6.

Dataset	Native	Stage I only	QGVR
TextVQA-X	77.98	80.10	82.38
VQS	76.13	76.90	77.14
TextOCR	78.30	81.44	83.97
VizWiz-G	41.36	44.73	45.82

spatial responses rather than warped RGB. Overlays and smoothing affect neither inference nor quantitative attention metrics.

Two-stage correction examples. Figure 5 illustrates corrections on document, relation, and infographic questions. Native, Stage I, and Stage II show the unwarped input, base proposal, and deployed selection, respectively, with their own answers and aligned decoding attention. The three paths share a color scale within each example; hotspot intensity is independent of answer correctness. In these selected cases, Native and Stage I answer incorrectly and final selection answers correctly.

Visualization reproducibility. Figures use Matplotlib. The source RGB images, crop boxes, attention arrays, coordinate transforms, and histogram bins are retained with the plotting code. Display smoothing and color transformations are separate from quantitative scoring.

B.7 Explanatory Grounding Analysis and Path Comparison

This analysis uses spatial annotations to interpret the complete system's post-decoding attention. TextVQA-X provides visual and textual explanations for TextVQA (Nagaraj Rao et al., 2021), and VQS links questions and answers to image segmentations (Gan et al., 2017). TextOCR is our evaluation subset derived from TextVQA (Singh et al., 2019), not a separately published dataset: it uses only OCR boxes whose text matches the reference answer, rather than the union of all text boxes. Across the four subsets, all paths use the same tile grid, question template, and greedy-decoding settings for each example. The VizWiz subset uses original target masks from VizWiz-VQA-Grounding (Chen et al., 2022); its local records contain one reference answer, so the answer metric is normalized single-reference exact match.

Local AttWarp extracts attention from the model's own generated answer at decoder layer 16 (zero-indexed), replays aligned generated positions, and warps pixels using horizontal and vertical marginal inverse CDFs. This is not a best configuration retuned across all models, attention layers, or warp strengths in the original paper. Each path is evaluated post hoc using attention from its own answer. For warped paths, the target mask and image share the same sampling grid. Target attention mass is weighted by the fraction of each visual token covered by the target region, accounting for partial coverage. Focus-hit accuracy checks only the center of the maximum-attention token; it measures neither total target attention mass nor causal attribution.

Region expansion and attention enrichment. Expansion is the ratio of the target's warped area fraction to its native area fraction. Enrichment is the target's attention-mass fraction divided by its area fraction. Across the four subsets, mean expansion for final selection is approximately 1.00, versus 1.72–4.36 for local AttWarp; corresponding enrichment ranges are 2.82–9.50 and 1.39–3.64. Area normalization avoids equating larger targets with more relevant attention. These aggregate statistics describe the complete system; an overall mean near 1.00 does not establish uniformly small distortion for every warped example.

B.8 Oracle@10 and Selector Diagnostics

We sample 500 questions without replacement from each of TextVQA validation, InfoVQA validation, DocVQA validation, ChartQA test, and GQA balanced test-dev (seed 20260912). ChartQA is sampled directly from the complete test split. This 2,500-question set is shared by Tables 3, 4, 5, and 9. The direction control uses its own freshly trained checkpoints and the decoding batch specified in Appendix B.4; the settings below refer to the main trained proposal and selector.

Frozen InternVL decodes all ten candidates per question: Native, the base warp, and eight fixed edits. Proposal and selector checkpoints, exploration weight 0.15, gate threshold 10^{-7}, and zero minimum ranking margin are fixed. Selection receives only the image and question. All paths use the same prompt, BF16 greedy decoding with 32 new tokens, decode batch size 20, and ViT microbatches of 24. The selected path reuses its candidate answer. All 25,000 candidate answers are scored.

TextVQA uses ten-answer leave-one-out accuracy; InfoVQA and DocVQA use ANLS; ChartQA uses 5% relaxed accuracy; GQA uses exact accuracy. Tables 4 and 5 report the equal-weight mean of the five task scores on the percentage scale.

Let s_{dik} be candidate k's score for question i in dataset d, and $\hat{k}_{di}$ the selector's decision. For $D = 5$ and $N = 500$,

$$\overline{\text{Oracle @10}} = \frac{100}{DN} \sum_{d=1}^{D} \sum_{i=1}^{N} \max_{0 \le k < 10} s_{dik}, \qquad \overline{\text{Regret}} = \overline{\text{Oracle @10}} - \frac{100}{DN} \sum_{d=1}^{D} \sum_{i=1}^{N} s_{di\hat{k}_{di}}. \quad (25)$$

Oracle@10 uses reference answers only after decoding and measures the candidate pool's offline upper bound. Mean regret is 4.20 points (95% CI: [3.47, 4.96]). Intervals use 10,000 paired image-cluster resamples within each dataset followed by equal-weight averaging across datasets (seed 20260912).

Mean warp-selection rate is 18.08%, beneficial-switch precision is 19.37%, and harmful-switch rate is 0.96%. Within each dataset, warp and harmful-switch rates use all questions as the denominator; beneficial precision uses nonnative selections. Each reported mean averages the five dataset rates. Equal-score switches are neutral. Per-question predictions, candidate scores, sample manifests, and checkpoint hashes are retained in the experiment records.

C Runtime Protocol and Timing Breakdown

C.1 Timing Scope and Implementation Differences

The measurements underlying Table 7 use a fixed set of 128 examples across two GPUs, a per-GPU batch size of 8, and three repetitions. InternVL uses BF16 without quantization, at most 12 local tiles per image, and up to 32 generated tokens. CUDA-synchronized wall-clock time includes image loading, preprocessing, proposals, warping, reranking, all required ViT forward passes, and answer generation. It excludes model loading, warm-up, scoring, plotting, and disk output. Precomputed image features, attention maps, warps, and answers are not reused across examples or repetitions. Native ViT feature reuse within one QGVR inference is part of the measured algorithm. Operating-system file caches and model kernel caches remain warm.

We did not measure an exhaustive baseline that fully decodes all ten candidates and therefore report no measured speedup over that procedure. Speed comparisons in the main text use only the local measurements in Table 7.

Throughput is the median of three dual-GPU runs; single-GPU-equivalent throughput is half that value. Table 7 uses the same Base LLM (Native) reference for every method. Throughput retained is $100 r_m / r_{\text{Native}}$, where r is throughput and Native is 100%. Time and memory increases are $100(x_m / x_{\text{Native}} - 1)$, where x is amortized time or peak allocated memory. Higher retained throughput and smaller cost increases are better. Percentages are computed before rounding; QGVR's peak-memory increase is 0.0097%, displayed as 0.0%. Amortized time per example is the sample average of batch timings. Throughput and amortized time are therefore not exact reciprocals, because they use different aggregation. The Native-to-QGVR throughput ratio is 1.157, while mean time increases from 0.52295 to 0.59999 seconds, or 14.73%. This is not batch-one interactive latency. Initial tests used the cross-process timestamp span; supplementary ViCrop and SD-RPN tests used the slowest process after a process barrier, with a barrier polling interval of 0.1 seconds.

Local AttWarp sequentially performs native draft decoding, attention replay given the draft sequence, pixel warping, and final decoding, rather than the original paper's fused implementation. Local ViCrop uses att-CROP: the native and cropped images jointly use at most 24 local tiles and 8,192 context tokens. SD-RPN uses dynamic tile expansion, adding up to 256 region tokens with a context limit of 6,144, and retains the SDPA backend in parts of its decoder. These costs should be interpreted for the actual adapted implementations, not as inherent speeds of the algorithms on arbitrary systems.

The separate Native controls in the supplementary ViCrop and SD-RPN tests achieve dual-GPU throughputs of 3.951 and 3.356 examples/s, respectively. Relative to those controls, their slowdowns are $3.50\times$ and $1.32\times$, compared with $3.37\times$ and $1.50\times$ against the common Native control. Moreover, the initial Native, AttWarp, and QGVR tests keep proposal and selection modules resident, while supplementary tests load only the modules required by each method. Peak memory is therefore not a fully matched comparison of standalone deployments.

C.2 Component Timings

Table 11 reports mean amortized component times per example for Table 7. The *frontend* includes image preprocessing and method-specific proposal, warping, reranking, and required ViT operations. AttWarp's draft/replay column includes initial answer decoding and attention replay conditioned on that answer sequence.

Table 11: Amortized component times per example in the matched dual-GPU throughput experiment. Components sum to the reported total up to rounding.

Method	Frontend (ms)	Draft/replay (ms)	Final decode (ms)	Total (ms)
Base LLM	101.99	0	420.96	522.95
AttWarp (local)	193.80	698.11	420.17	1,312.09
Stage I only	191.58	0	419.40	610.98
QGVR	181.63	0	418.36	599.99