RECONFIGURABLE EMBODIED REASONING: OFFLINE COMPOSITION OF REPLACEABLE CAPABILITY MODULES

Anonymous authors
Paper under double-blind review

ABSTRACT

Generalist embodied models typically integrate capabilities such as grounding and interaction, spatial reasoning, temporal understanding and planning, and video understanding into a single checkpoint. We explore how these capabilities can be independently maintained, updated, and recombined after training while retaining deployment as a single model. We organize independently maintained LoRA modules into a capability library and introduce ReCAP, a weight-only composition method that integrates selected modules into one static checkpoint. ReCAP incorporates capability updates through module replacement and recomposition, without retraining the remaining modules. ReCAP anchors composition in one complete module update and expresses the others as residual edits in shared coordinates. It aggregates these edits through magnitude-weighted voting in a joint residual subspace and controls their injection with a support mask. ReCAP achieves the highest Full-11 score among the evaluated training-free merging methods across 11 embodied reasoning benchmarks, two backbones, and three capability-library settings. On libraries trained from a shared base, it scores 66.53 on InternVL3.5-14B and 68.27 on Qwen3-VL-8B, exceeding the strongest single modules by 1.86 and 0.85 points. Under a shared evaluation pipeline, the Qwen3-VL-8B composition outperforms all nine public embodied models, leading the strongest comparator by 2.89 points. Further experiments show that task-specific recomposition improves target performance, local module improvements benefit the composed model without retuning, and incremental updates obtained through continued training of an individual capability can enter an existing capability library. These results demonstrate the viability of building high-performing embodied models from independently evolving capability modules while preserving single-checkpoint deployment.

1 INTRODUCTION

In embodied settings, models must ground instruction-specified interaction targets, infer spatial relations, understand event sequences and plan subsequent steps, and interpret video actions and scene changes. Existing research expands the tasks, scenes, and interactions they handle (Driess et al., 2023; Brohan et al., 2023; Mu et al., 2023; Huang et al., 2024b; Gemini Robotics Team, 2025). We further explore organizing development around individual capabilities: when one capability improves, can we reuse the other trained capabilities and recompose the updated module into a unified model?

Embodied systems require continued improvement because their existing capabilities may fall short of task requirements in familiar settings or reveal generalization gaps when encountering new objects, spatial layouts, and interaction patterns. These gaps motivate capability-specific data collection and training while other capabilities remain reusable. Pooling all capability data for joint retraining couples local improvements to whole-model optimization. We explore independent capability development and incorporation through recomposition.

We organize capability-specific low-rank adaptation (LoRA) modules on a shared vision-language model (VLM) backbone into a capability library. Each stores an independently maintainable parameter update, enabling task-specific combinations and independent capability updates with single-model deployment. For different embodied tasks, we select corresponding modules and compose their

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.

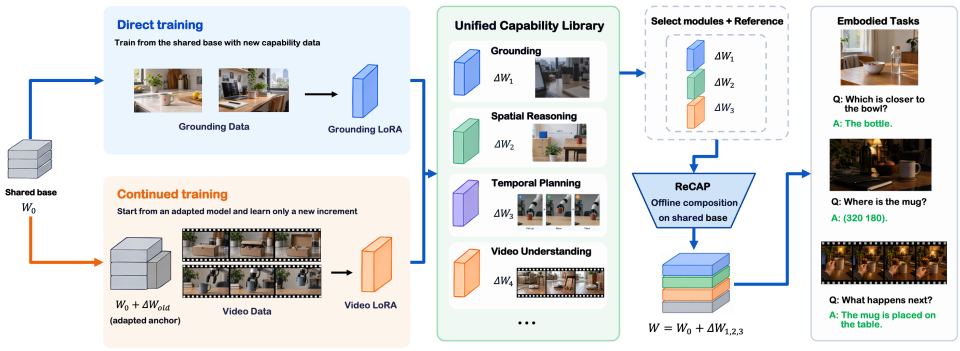

Figure 1: Reconfigurable embodied reasoning. Shared-base modules and increments from adapted anchors form a capability library. ReCAP composes selected modules around a reference offline into a single static checkpoint for deployment.

updates into one static checkpoint. An improved module replaces its predecessor for recomposition without retraining the remaining modules. We call this workflow **reconfigurable embodied reasoning** (Figure 1).

Realizing reconfigurable embodied reasoning requires effectively composing independently trained and updated capability modules. Opposite-sign updates at the same weights may cancel useful adaptations under direct addition (Yadav et al., 2023). Common representations improve LoRA alignment and aggregation (Stoica et al., 2025; Panariello et al., 2025). We investigate composition from a complete capability adaptation, selectively incorporating other modules' changes to integrate complementary capabilities and subsequent improvements into a unified deployable model.

We introduce **ReCAP** (Reconfigurable Capability Composition), a weight-only method for composing LoRA modules. Using Core Space's common representation, ReCAP takes a selected module's complete update as reference and expresses the remaining modules as reference-relative residual edits (Panariello et al., 2025). We construct a joint residual subspace, select and aggregate updates through magnitude-weighted voting, and regulate their injection into the reference update with a consensus support mask derived from complete module updates. These selective edits anchored in a reference adaptation produce a single static checkpoint without runtime routing or expert selection. The same composition procedure applies to different capability subsets and can be repeated after module updates, supporting task-specific capability combinations and continued capability improvement.

We evaluate ReCAP on InternVL3.5-14B and Qwen3-VL-8B. ReCAP outperforms joint training on the union of capability datasets at matched total LoRA rank on both VLM backbones. With the remaining modules and coefficients fixed, improving one module also benefits ReCAP, showing that targeted capability training can improve the unified model through recomposition.

Our contributions are:

- We formulate **reconfigurable embodied reasoning**: a capability-library framework for independent development, task-specific composition, and module replacement, reusing trained modules while retaining single-checkpoint deployment.
- We propose **ReCAP**, a weight-only LoRA composition method that uses a complete module update as reference and selectively aggregates and injects reference-relative residuals from the other modules.
- We extend capability libraries beyond a shared training origin by incorporating second-stage increments without retraining existing members; ReCAP leads the training-free methods in the tested extension.
- Across 11 benchmarks on the InternVL3.5-14B and Qwen3-VL-8B backbones, ReCAP leads the evaluated training-free merging methods on both aggregate metrics in all tested capability-

library settings. Both backbone-specific compositions also outperform all nine public embodied comparators on these metrics.

2 Related Work

Embodied models and capability integration. PaLM-E and RT-2 jointly train on vision-language and embodied supervision to develop unified capabilities (Driess et al., 2023; Brohan et al., 2023). Specialist-first approaches include ACE-Brain-0, which fine-tunes experts from a shared spatial foundation, combines them through task-vector optimization, then performs embodied fine-tuning (Gong et al., 2026); and Capek 0.5, which merges specialists with Trim, Elect Sign & Merge (TIES) before Multi-Teacher On-Policy Distillation (MOPD) (Chen et al., 2026). ReCAP composes independently maintained, replaceable LoRA capability modules for specific tasks and incorporates module updates without post-composition training.

Modular adaptation and capability updates. Existing approaches reuse trained modules in different ways: AdapterFusion learns to fuse adapter representations (Pfeiffer et al., 2021); CORAL routes among LoRA experts by task instruction (Luo et al., 2026); MergeVLA merges LoRA parameters while retaining task-specific masks and action expert heads (Fu et al., 2026). LoRAHub combines existing LoRAs into one module using a few target-task examples and gradient-free coefficient search (Huang et al., 2024a). To preserve knowledge as new tasks are learned, O-LoRA constrains updates to orthogonal subspaces (Wang et al., 2023), while PS-LoRA regularizes update directions and magnitudes and merges sequential adapters by magnitude (Zhou et al., 2025).

Model merging mechanisms. Model merging combines task-specific parameter updates to integrate the capabilities of separately fine-tuned models. Task-vector arithmetic represents adaptations as weight differences from a shared base and combines them arithmetically (Ilharco et al., 2023). To mitigate interference, subsequent methods filter or sparsify updates before aggregation, using magnitude trimming and sign-consistent merging (Yadav et al., 2023). Other methods align LoRA updates through joint SVD (Stoica et al., 2025) or representing them losslessly in a common low-dimensional core space (Panariello et al., 2025).

For capability organization, ReCAP composes independently trained and maintained LoRA capability modules offline into a single static model, without runtime routing or cross-task training constraints. For composition, ReCAP builds on the common representation of Core Space, starts from a complete reference update, aggregates reference-relative residuals by weighted magnitude in a joint residual subspace, and regulates residual injection through a consensus support mask.

3 ReCAP: Reconfigurable Capability Composition

We organize embodied capabilities as a library of LoRA modules on a shared backbone. A deployment configuration specifies the participating modules, a reference, and donor coefficients. Task reconfiguration changes the module subset and reference; capability updates replace module versions before recomposition, reusing the trained parameters of retained modules. ReCAP integrates the configured capabilities offline into one static deployment checkpoint. At each adapted layer, it starts from the complete reference update, aggregates reference-relative donor residuals in a joint residual subspace of common coordinates, adds low-weight outside-subspace terms, and regulates injection with a support mask (Figure 2).

3.1 A common coordinate system

For an adapted layer with base weights $W_0 \in \mathbb{R}^{m\times n}$, the effective LoRA update of module i is

$$\Delta W_i^{\text{eff}} = \frac{\alpha_i}{r_i} B_i A_i, \qquad B_i \in \mathbb{R}^{m\times r_i},\ A_i \in \mathbb{R}^{r_i\times n}. \tag{1}$$

Here r_i denotes rank and α_i/r_i scaling; all subsequent core representations and residuals use effective updates.

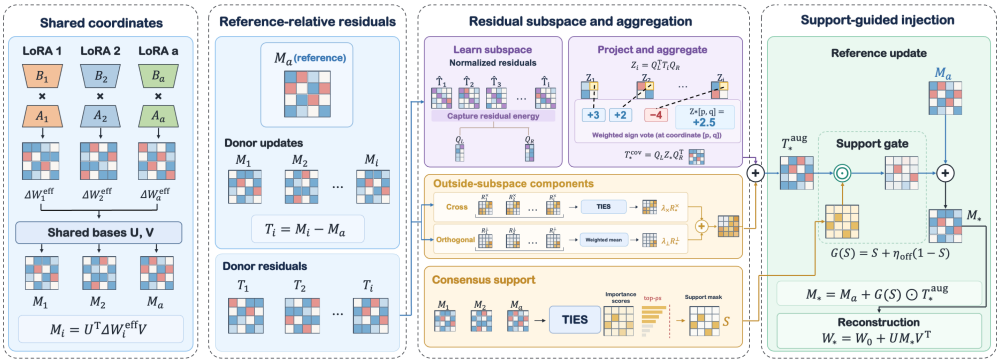

Figure 2: ReCAP at one adapted layer. Reference-relative residuals are aggregated in common coordinates, with reduced-weight outside-subspace contributions. A support mask from full updates regulates residual injection before reconstructing the weight update. Effective updates include LoRA scaling; the voting example uses equal weights.

Following Core Space (Panariello et al., 2025), we separately concatenate and orthogonalize selected left factors $\{B_i\}$ and transposed right factors $\{A_i^\top\}$, obtaining common orthonormal bases $U \in \mathbb{R}^{m\times d_L}$ and $V \in \mathbb{R}^{n\times d_R}$. Each module's core matrix and reconstructed update are

$$M_i = U^\top \Delta W_i^{\text{eff}} V, \qquad \widehat{\Delta W}_i = U M_i V^\top. \tag{2}$$

Reconstruction is exact if the common bases span all module updates; numerical truncation gives their projection (Appendix D.1). These core matrices represent reference-relative residuals for composition.

3.2 Reference-relative residuals

In common coordinates, ReCAP starts composition from the complete update M_a of a selected LoRA module a, groups the remaining modules into the donor set $\mathcal{D}$, and defines

$$T_i = M_i - M_a, \qquad i \in \mathcal{D}. \tag{3}$$

T_i is the candidate change from reference to donor. Common coordinates and the shared reference allow consistent residual comparisons: opposite signs indicate opposing adjustments at a coordinate. We selectively aggregate these residuals and inject the edit into M_a.

3.3 Learning a joint residual subspace

ReCAP first jointly learns compact row and column subspaces of the donor residuals, then determines signs and aggregate values at projected coordinates (Section 3.4). We normalize residuals as $\widehat{T}_i = T_i/(\|T_i\|_F + \epsilon_n)$ and the given donor coefficients m_i as $u_i = m_i / \sum_{j\in\mathcal{D}} m_j$, with $m_i \geq 0$ and a positive coefficient sum. We approximately maximize the weighted normalized-residual energy captured by the subspace:

$$\max_{Q_L^\top Q_L = I,\, Q_R^\top Q_R = I} \sum_{i\in\mathcal{D}} u_i \left\| Q_L^\top \widehat{T}_i Q_R \right\|_F^2. \tag{4}$$

Here $Q_L \in \mathbb{R}^{d_L\times k_L}$ and $Q_R \in \mathbb{R}^{d_R\times k_R}$ define the subspace in common coordinates U, V. At fixed dimensions, energy maximization minimizes weighted normalized-residual reconstruction error. Normalization reduces norm influence; summing per-residual projected energies avoids sign cancellation. After SVD initialization from the weighted mean normalized residual, we alternate eigendecomposition updates of Q_L, Q_R (at most five rounds; Appendix B).

3.4 AGGREGATING RESIDUALS BY MAGNITUDE-WEIGHTED VOTE

Once the subspace is determined, ReCAP projects the unnormalized residuals as $Z_i = Q_L^\top T_i Q_R$, preserving magnitude information along the selected directions. Following TIES (Yadav et al., 2023), ReCAP avoids opposite-sign cancellation by electing each coordinate's sign by total weighted magnitude, then averaging same-sign values with weights.

ReCAP adjusts donor weights by a bounded positive factor f_i of each residual's energy capture rate, yielding voting and aggregation weights:

$$\eta_i^{\mathrm{cap}} = \frac{\|Z_i\|_F^2}{\|T_i\|_F^2 + \epsilon_c}, \qquad \omega_i = \frac{u_i f_i}{\sum_{j\in\mathcal{D}} u_j f_j}. \tag{5}$$

The factor f_i is nondecreasing in capture rate; its positive lower bound prevents low capture alone from excluding positively weighted donors (definition and parameters: Appendix D.1).

The projected residuals Z_i are not magnitude-trimmed, and we use $\mathrm{sign}(0) = 0$. At each projected coordinate (p, q), we select

$$s_{pq} = \mathrm{sign}\left(\sum_i \omega_i Z_i[p,q]\right) \tag{6}$$

We then compute the weighted average of contributions matching the elected nonzero sign:

$$Z_\star[p,q] = \frac{\sum_i \omega_i Z_i[p,q]\mathbf{1}\{\mathrm{sign}(Z_i[p,q]) = s_{pq} \neq 0\}}{\sum_i \omega_i \mathbf{1}\{\mathrm{sign}(Z_i[p,q]) = s_{pq} \neq 0\} + \varepsilon}. \tag{7}$$

A tied vote or absence of matching contributions yields zero. Values of the numerical stability constants $\epsilon_n, \epsilon_c, \varepsilon > 0$ are given in Appendix D.1.

Mapping $Z_\star$ back to the common coordinates gives

$$T_\star^{\mathrm{cov}} = Q_L Z_\star Q_R^\top. \tag{8}$$

For changes outside the compact subspace, original donor weights u_i aggregate cross components through TIES and bilaterally orthogonal components through weighted averaging. Their respective smaller fixed gates scale these aggregates before addition to $T_\star^{\mathrm{cov}}$, yielding $T_\star^{\mathrm{aug}}$ for support-guided injection (definitions and rules: Appendix C).

3.5 SUPPORT-GUIDED INJECTION

Equal-weight TIES on all selected modules' full core updates M_i (including the reference) yields the consensus importance map C_{cons}. Thresholding its absolute magnitudes at a preset retention fraction gives binary mask S regulating the injection strength of aggregated edits (Appendix D.1). The final core update is

$$M_\star = M_a + \big(S + \eta_{\mathrm{off}}(1 - S)\big) \odot T_\star^{\mathrm{aug}}, \qquad 0 \leq \eta_{\mathrm{off}} \leq 1. \tag{9}$$

The residual enters at full strength on supported locations and is multiplied by η_{off} elsewhere. We then reconstruct and add the update to the base weights:

$$\Delta W_\star = U M_\star V^\top, \qquad W_\star = W_0 + \Delta W_\star. \tag{10}$$

Applying this procedure to each adapted layer produces a single static deployment model; configurations are given in Appendix D.

Recomposition after capability updates. After module replacement or reselection, ReCAP reuses retained weights without retraining and rebuilds common coordinates, reference-relative residuals, the joint residual subspace, and the support mask from the current modules. When LoRA training starts from a model adapted by an existing module, only the new increment enters the library for composition with the remaining modules (Section 4.4; Appendix E.2).

4 EXPERIMENTS

We test three aspects of ReCAP: complementarity among independently trained modules, task-specific recomposition of one library, and benefits from module improvements while reusing other modules. Component controls analyze gains; public-model comparisons assess the embodied reasoning performance of ReCAP's composed models.

4.1 EXPERIMENTAL SETUP

Models and libraries. We use rank-128 LoRA modules on InternVL3.5-14B and Qwen3-VL-8B. InternVL3.5-14B has grounded interaction (NL1), spatial reasoning (NL2), temporal planning (NL3), and video-semantic next-action reasoning (NL4); Qwen3-VL-8B omits NL3 (training data: Table 9). Both primary libraries train modules independently from W_0. The extended InternVL3.5-14B library retains NL1–NL3 and replaces NL4 with the LoRA increment trained from $W_0 + \Delta W_{\mathrm{NL1}}$. Within each setting, all methods use the same stored updates and produce single static models (origins and coefficients: Appendix E.2, Tables 10–11).

Benchmarks and comparisons. We evaluate on VSIBench, ERQA, OpenEQA, EmbSpatial, RoboVQA, Cosmos, CVBench, BLINK-4, RefSpatial, VABench-Point, and Where2Place (Appendix E.1). **Full-11** equally averages all scores; **Mean-9** averages the nine remaining after excluding the LLM-judged OpenEQA and RoboVQA. Comparators include four training-free methods (Task Arithmetic, TIES, KnOTS-TIES, Core-Space TIES), the strongest single module, a per-benchmark single-module oracle, and joint training on pooled capability data. The oracle selects each benchmark's best single module and is not one deployable checkpoint. Joint training matches total tokens and summed LoRA rank (512/384 for InternVL3.5-14B/Qwen3-VL-8B; Appendix E).

4.2 COMPOSITION QUALITY AND CAPABILITY COMPLEMENTARITY

Composing independently trained capabilities outperforms rank-matched joint training. With total tokens, LoRA rank, and experimental settings matched, ReCAP exceeds pooled-data joint training by $2.45/2.48$ Full-11/Mean-9 points on InternVL3.5-14B and $1.72/1.70$ on Qwen3-VL-8B (Table 1). These results show that independently developed capabilities can be composed offline into a single unified model, improving aggregate performance while retaining modular maintainability.

Complementary capabilities improve performance. ReCAP exceeds all four merging baselines on Full-11 and Mean-9 across the three libraries (Table 1). Its Full-11 margins over each library's strongest merging baseline are 0.67 for same-origin InternVL3.5-14B (Vanilla TIES), 0.74 for the extended library (Core-Space TIES), and 0.56 for Qwen3-VL-8B (KnOTS-TIES). Same-origin InternVL3.5-14B and Qwen3-VL-8B also exceed their strongest single modules by 1.86 and 0.85 Full-11 points; the former matches the per-benchmark single-module oracle at reported precision using one static checkpoint.

The complete composition design provides further gains. With core representations held fixed, ReCAP gains 1.03 Full-11 and 1.34 Mean-9 over Core-Space TIES on same-origin InternVL3.5-14B, supporting gains from the complete residual composition procedure beyond the shared representation. Component analyses appear in Section 4.5; per-benchmark scores appear in Appendix F.1.

Table 1: Evaluation results in the reported capability-library settings. Bold marks the highest training-free merging score in each library. Per-benchmark results: Appendix F.1.

	InternVL3.5-14B				Qwen3-VL-8B	
	Primary: same-origin		Extended library			
Method	Full-11	Mean-9	Full-11	Mean-9	Full-11	Mean-9
Training-free composition						
Vanilla TIES	65.86	66.87	64.97	65.64	65.48	65.49
Task Arithmetic	65.11	66.31	64.99	66.11	65.08	64.72
Core-Space TIES	65.50	66.32	65.82	66.73	67.35	67.53
KnOTS-TIES	65.42	66.37	65.35	66.31	67.71	67.77
ReCAP	**66.53**	**67.66**	**66.56**	**67.52**	**68.27**	**68.44**
Model-level references						
Strongest single module	64.67	65.71	65.34	66.28	67.42	67.35
Per-benchmark single-module oracle	66.53	67.91	66.71	67.95	68.79	68.98
Rank-matched joint training	64.08	65.18	64.08	65.18	66.55	66.74

4.3 TASK-SPECIFIC CAPABILITY COMPOSITION

One library supports target-specific recomposition. With module weights and donor coefficients fixed, ReCAP improves target performance by changing subsets and references (Table 2). Same-origin InternVL3.5-14B exceeds defaults and strongest singles on full-set spatial/video evaluations, with video 1.57 points above default; extended-library localization also exceeds both on the configuration study's evaluation samples. These gains show that existing modules can be recomposed for different targets.

Target gains come with lower non-target performance and Mean-9, revealing a trade-off between task specialization and overall performance. Reconfiguration does not always outperform the default composition or the strongest single module: the default four-module composition remains preferable for same-origin localization, while extended-library video matches the strongest single module. Appendix G provides candidate configurations, target definitions, and full comparisons.

Table 2: Task-specific reconfiguration of InternVL3.5-14B. Reference modules are bold; deltas are relative to the default composition. Non-target scores average the remaining Mean-9 benchmarks. See Appendix G.

Library	Target	Modules	Target score			ΔNon-target	ΔMean-9
			Subset	Default	Best single		
Primary	Spatial	NL2, NL3, **NL4**	74.58	73.51	74.15	-3.44	-2.44
Primary	Localization	**NL1**, NL2, NL4	66.87	66.99	65.86	-0.08	-0.10
Primary	Video	**NL2**, NL3, NL4	61.57	60.00	61.37	-1.16	-0.87
Extended	Spatial	NL2, NL3, **NL4**	74.18	72.74	73.32	-3.01	-2.02
Extended	Localization	NL1, **NL2**, NL3	68.54	66.65	66.60	-1.44	-0.33
Extended	Video	**NL2**, NL3, NL4	60.58	58.51	60.58	-0.38	-0.11

4.4 CAPABILITY-LIBRARY EXTENSION AND UPDATES

Keeping the four InternVL3.5-14B module slots fixed, we test ReCAP after local module improvement, incorporation of LoRA increments learned by continued training from an already adapted model, and full-library refresh.

This experiment tests whether a local capability improvement transfers through recomposition to the unified deployment model.

Local improvements benefit the composition. Both NL3 versions train for $3{,}000$ steps; other modules and composition coefficients stay fixed. Replacing only NL3 with a different-data-mixture version improves ReCAP's primary and extended compositions on mean RoboVQA/OpenEQA, though less than the standalone module (Table 3B). Independently improved capabilities thus transfer through recomposition without retraining retained modules or retuning coefficients.

We also compare direct coefficient reuse with norm calibration, keeping other modules and coefficients fixed. For the replaced donor i, let $n_i^{\mathrm{old}}, n_i^{\mathrm{new}} > 0$ denote the old and new overall Frobenius norms of effective LoRA updates across shared adapted layers. The norm ratio and calibrated coefficient are

$$r_i^{\mathrm{norm}} = \frac{n_i^{\mathrm{new}}}{n_i^{\mathrm{old}}}, \qquad m_i^{\mathrm{new}} = \frac{m_i^{\mathrm{old}}}{r_i^{\mathrm{norm}}}, \quad i \in \mathcal{D}. \tag{11}$$

The coefficient varies inversely with update magnitude (norm definition: Appendix H.3, Eq. (53)). On the module-replacement study's evaluation samples, direct reuse exceeds calibration on mean RoboVQA/OpenEQA for both NL3 seeds and on Cosmos for NL4 (Appendix H.2, Table 17).

Continued-training increments remain composable. Composing NL1–NL3 with the NL4 increment trained from $W_0 + \Delta W_{\mathrm{NL1}}$, ReCAP exceeds all four extended-library baselines. Under fixed modules and settings, increment entry improves Full-11 by 1.41 points over cumulative entry from W_0 (NL1 update plus increment; Table 3A). Both reference-relative residuals and support-map inputs change, so this tests the overall entry strategy (Appendix H.1).

Table 3: Library evolution with composition settings fixed. A: NL4 entry conventions. B: NL3 data-mixture update; RoboVQA/OpenEQA mean. Panel metrics differ.

A. Continued-training NL4

Stored entry	Full-11	Mean-9
Native increment	66.56	67.52
Flattened change	65.15	65.99

B. NL3 data update

Setting	Before	After	Change
Standalone	58.93	64.08	+5.15
Primary merge	61.62	62.13	+0.51
Extended merge	61.66	62.49	+0.83

Full refresh benefits from donor calibration. After all four extended-library modules are independently retrained, norm calibration (Eq. (11)) raises Full-11 from coefficient reuse's 62.23 to 64.25 ($+2.02$), with the refreshed reference unscaled (Table 4; configuration: Appendix H.3).

Table 4: Composition results after refreshing all four InternVL3.5-14B extended-library modules. Norm calibration leaves the refreshed reference unscaled.

Setting	Full-11	Mean-9
Reuse original donor coefficients	62.23	63.25
Best refreshed single module	64.21	65.17
Adjusted-coefficient control	63.66	65.02
Norm-calibrated donor coefficients	**64.25**	**65.71**
Control: calibration factors in wrong slots	58.05	58.70

Calibration moves the composition from below to slightly above the strongest refreshed module on both aggregates. Wrong-slot factors cost 6.20 Full-11 points relative to correct calibration, showing that factors must match donors. Both update types avoid joint retraining: single-module replacement favors coefficient reuse, whereas full-library refresh benefits from norm calibration (Appendices H.3–H.4).

4.5 Component and aggregation analysis

In this section, we examine design contributions through component removal (Table 5A) and vary only within-subspace aggregation, holding other operations fixed, to separate the retained pipeline's performance from the additional gains of sign selection (Table 5B).

Table 5: Component and aggregation controls. A: Full-11 changes; dashes denote untested settings. Reference–residual removal includes uniform donor weighting. B: InternVL3.5-14B Full-11 (change from ReCAP), with other operations fixed. Details: Appendix I.

A. Component removal

Removed component	InternVL3.5-14B		Qwen3-VL -8B
	Same-origin	Extended	
Common coordinates	-0.80	-0.97	-0.33
Reference–residual	-0.08	-0.91	-0.27
Joint residual subspace	-0.09	-0.58	-0.30
Support injection	—	—	-0.33

B. Aggregation rule

Rule	Same-origin	Extended
ReCAP signs	**66.53** (—)	**66.56** (—)
Weighted average	66.40 (-0.13)	66.19 (-0.37)
Majority signs	66.03 (-0.50)	65.52 (-1.04)
Reversed signs	64.36 (-2.17)	64.42 (-2.14)

Common coordinates and residual organization jointly support composition. Removing common coordinates lowers Full-11 in all three libraries (Table 5A), showing that a shared representation supports module composition. Jointly replacing reference-relative construction and donor weighting with uniformly weighted symmetric merging, or removing the joint residual subspace, lowers Full-11 in the InternVL3.5-14B extended library by 0.91 and 0.58 points, respectively. Both drops exceed the corresponding 0.08 and 0.09 points in the same-origin library, indicating that the tested continued-training extended library is more sensitive to these design choices.

The composition pipeline provides a foundation that sign selection further improves. Holding the joint residual subspace, donor coefficients, support mask, and other operations fixed, Table 5B varies only the within-subspace aggregation rule. Weighted averaging without sign filtering still beats

all four baselines, supporting the retained pipeline. Magnitude-weighted sign selection with same-sign aggregation improves further; unweighted majority and reversed signs do worse, showing that direction selection matters. These findings concern the learned bases (rotation controls: Appendix I.3).

Support guidance and outside-subspace terms improve residual injection. ReCAP adds low-weight outside-subspace terms before location-dependent, support-masked injection. Mask removal lowers Qwen3-VL-8B's Full-11, supporting location-dependent injection after aggregation (Table 5A). Removing outside-subspace terms also slightly lowers Full-11 on both backbones, showing that useful updates remain outside the finite-dimensional projection and provide supplementary gains under the tested low-weight setting (Appendix I.2).

4.6 Comparison with public embodied models

Both ReCAP backbones exceed all nine public embodied comparators on Full-11 and Mean-9 (Table 6). Qwen3-VL-8B leads the strongest comparator, Embodied-R1.5 (Yuan et al., 2026b), by 2.89 and 3.12 points, respectively. Independently developed LoRA capabilities thus compose into models leading this comparison. The comparison protocol and full per-benchmark results are in Appendix J (Table 18).

Table 6: Public-model comparison with main-result benchmarks and scoring. ReCAP rows match Table 1; public models use Qwen3-VL-8B settings, InternVL3.5-14B its own.

Model	Full-11	Mean-9
ReCAP (Qwen3-VL-8B)	**68.27**	**68.44**
ReCAP (InternVL3.5-14B)	66.53	67.66
Embodied-R1.5 (Yuan et al., 2026b)	65.38	65.32
RynnBrain-8B (Dang et al., 2026)	59.75	59.05
MiMo-Embodied-7B (Hao et al., 2025)	50.61	48.68
RoboBrain 2.0-7B (BAAI RoboBrain Team et al., 2025)	47.83	45.02
Pelican-VL-7B (Zhang et al., 2025)	47.69	44.05
EmbodiedBrain-7B (Zou et al., 2025)	46.67	43.87
Embodied-R1-3B (Yuan et al., 2026a)	45.82	43.31
Cosmos-Reason1-7B (NVIDIA et al., 2025)	45.15	42.11
SpaceR-7B (Ouyang et al., 2025)	43.47	41.03

5 Discussion and Conclusion

ReCAP's experiments show how independently maintained LoRA capability modules can be reused in continued development. Task-specific reconfiguration, continued-training increment incorporation, and module replacement experiments together show that the same capability library can accommodate changes in task requirements and module versions, while existing training outcomes remain useful in subsequent compositions. Offline composition thus becomes a repeatable step connecting independent capability development with unified deployment: modules remain independently maintainable during development, while each deployment uses a single static model checkpoint without runtime routing or expert selection.

Local improvements remain reusable, but their gains depend on other modules and composition settings. Different coefficient outcomes after single-module replacement and full-library refresh require configurations to accommodate library changes. Through offline recomposition, ReCAP retains the training investment in existing modules, incorporates capability updates, and adjusts capability emphasis to deployment goals.

ReCAP demonstrates high-performing embodied reasoning through independently developed and updated LoRA modules. On both backbones, ReCAP exceeds rank-matched joint training on pooled capability data; reconfiguration and update experiments demonstrate library reuse with single-model deployment. Future work will test larger capability libraries and closed-loop embodied tasks.

AI USE STATEMENT

Generative AI tools were used to assist with code implementation, literature retrieval and in-depth searches for related work, and manuscript language editing. The analysis of related work was carried out independently by the authors. The authors also worked with AI tools to check mathematical notation and formulas, and proofs. Language editing assistance focused on improving the clarity and presentation of content developed by the authors.

The creative research ideas and core methodological contributions were developed independently by the authors without using generative AI. The authors also independently designed the experiments and developed the substantive analyses, interpretations, and descriptions of the experimental results. These intellectual contributions and scientific judgments were made by the authors, rather than generated by AI.

The authors reviewed and verified the AI-assisted code and text and checked the cited references against their original sources. The authors take full responsibility for the final manuscript, code, mathematical claims, and reported results, including all content and artifacts developed with AI assistance.

REFERENCES

BAAI RoboBrain Team, Mingyu Cao, Huajie Tan, Yuheng Ji, Xiansheng Chen, Minglan Lin, Zhiyu Li, Zhou Cao, Pengwei Wang, et al. RoboBrain 2.0 technical report. *arXiv preprint arXiv:2507.02029*, 2025.

Anthony Brohan, Noah Brown, Justice Carbajal, Yevgen Chebotar, Xi Chen, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. *arXiv preprint arXiv:2307.15818*, 2023.

Ying Chen, Weizhen Li, Zhe Hu, Zhenjiang Li, Rui Jiang, Zhifeng Gu, Lihuang Fang, Jiangping Liu, Lei Yi, and Jie Chen. Capek 0.5: An execution-centric vision-language model for embodied intelligence. *arXiv preprint arXiv:2608.06756*, 2026.

Ronghao Dang, Jiayan Guo, Bohan Hou, Sicong Leng, Kehan Li, Xin Li, Jiangpin Liu, Yunxuan Mao, et al. RynnBrain: Open embodied foundation models. *arXiv preprint arXiv:2602.14979*, 2026.

Danny Driess, Fei Xia, Mehdi S. M. Sajjadi, Corey Lynch, Aakanksha Chowdhery, Brian Ichter, Ayzaan Wahid, Jonathan Tompson, Quan Vuong, Tianhe Yu, et al. PaLM-E: An embodied multimodal language model. In *International Conference on Machine Learning (ICML)*, 2023.

Ky Fan. On a theorem of Weyl concerning eigenvalues of linear transformations I. *Proceedings of the National Academy of Sciences*, 35(11):652–655, 1949.

Yuxia Fu, Zhizhen Zhang, Yuqi Zhang, Zijian Wang, Zi Huang, and Yadan Luo. MergeVLA: Cross-skill model merging toward a generalist vision-language-action agent. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2026.

Gemini Robotics Team. Gemini Robotics: Bringing AI into the physical world. *arXiv preprint arXiv:2503.20020*, 2025.

Ziyang Gong, Zehang Luo, Anke Tang, Zhe Liu, Shi Fu, Zhi Hou, Ganlin Yang, Weiyun Wang, Xiaofeng Wang, Jianbo Liu, Gen Luo, Haolan Kang, Shuang Luo, Yue Zhou, Yong Luo, Li Shen, Xiaosong Jia, Yao Mu, Xue Yang, Chunxiao Liu, Junchi Yan, Hengshuang Zhao, Dacheng Tao, and Xiaogang Wang. ACE-Brain-0: Spatial intelligence as a shared scaffold for universal embodiments. *arXiv preprint arXiv:2603.03198*, 2026.

Xiaoshuai Hao, Lei Zhou, Zhijian Huang, Zhiwen Hou, Yingbo Tang, Lingfeng Zhang, Guang Li, Zheng Lu, et al. MiMo-Embodied: X-Embodied foundation model technical report. *arXiv preprint arXiv:2511.16518*, 2025.

Chengsong Huang, Qian Liu, Bill Yuchen Lin, Tianyu Pang, Chao Du, and Min Lin. LoRAHub: Efficient cross-task generalization via dynamic LoRA composition. In *Conference on Language Modeling (COLM)*, 2024a.

Jiangyong Huang, Silong Yong, Xiaojian Ma, Xiongkun Linghu, Puhao Li, Yan Wang, Qing Li, Song-Chun Zhu, Baoxiong Jia, and Siyuan Huang. An embodied generalist agent in 3D world. In *International Conference on Machine Learning (ICML)*, 2024b.

Gabriel Ilharco, Marco Tulio Ribeiro, Mitchell Wortsman, Suchin Gururangan, Ludwig Schmidt, Hannaneh Hajishirzi, and Ali Farhadi. Editing models with task arithmetic. In *International Conference on Learning Representations (ICLR)*, 2023.

Yuankai Luo, Woping Chen, Tong Liang, and Zhenguo Li. CORAL: Scalable multi-task robot learning via LoRA experts. *arXiv preprint arXiv:2603.09298*, 2026.

Yao Mu, Qinglong Zhang, Mengkang Hu, Wenhai Wang, Mingyu Ding, Jun Jin, Bin Wang, Jifeng Dai, Yu Qiao, and Ping Luo. EmbodiedGPT: Vision-language pre-training via embodied chain of thought. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

NVIDIA, Alisson Azzolini, Junjie Bai, et al. Cosmos-Reason1: From physical common sense to embodied reasoning. *arXiv preprint arXiv:2503.15558*, 2025.

Kun Ouyang, Yuanxin Liu, Haoning Wu, Yi Liu, Hao Zhou, Jie Zhou, Fandong Meng, and Xu Sun. SpaceR: Reinforcing MLLMs in video spatial reasoning. *arXiv preprint arXiv:2504.01805*, 2025.

Aniello Panariello, Daniel Marczak, Simone Magistri, Angelo Porrello, Bartłomiej Twardowski, Andrew D. Bagdanov, Simone Calderara, and Joost van de Weijer. Accurate and efficient low-rank model merging in core space. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Jonas Pfeiffer, Aishwarya Kamath, Andreas Rücklé, Kyunghyun Cho, and Iryna Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In *Conference of the European Chapter of the Association for Computational Linguistics (EACL)*, 2021.

George Stoica, Pratik Ramesh, Boglarka Ecsedi, Leshem Choshen, and Judy Hoffman. Model merging with SVD to tie the KnOTS. In *International Conference on Learning Representations (ICLR)*, 2025. arXiv:2410.19735.

Xiao Wang, Tianze Chen, Qiming Ge, Han Xia, Rong Bao, Rui Zheng, Qi Zhang, Tao Gui, and Xuanjing Huang. Orthogonal subspace learning for language model continual learning. In *Findings of the Association for Computational Linguistics: EMNLP*, 2023.

Prateek Yadav, Derek Tam, Leshem Choshen, Colin Raffel, and Mohit Bansal. TIES-merging: Resolving interference when merging models. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Yifu Yuan, Haiqin Cui, Yaoting Huang, Yibin Chen, Fei Ni, Zibin Dong, Pengyi Li, Yan Zheng, Hongyao Tang, and Jianye Hao. Embodied-R1: Reinforced embodied reasoning for general robotic manipulation. In *International Conference on Learning Representations (ICLR)*, 2026a.

Yifu Yuan, Yaoting Huang, Xianze Yao, Yutong Li, Shuoheng Zhang, Linqi Han, Pengyi Li, et al. Embodied-R1.5: Evolving physical intelligence via embodied foundation models. *arXiv preprint arXiv:2606.11324*, 2026b.

Yi Zhang, Che Liu, Xiancong Ren, Hanchu Ni, Shuai Zhang, Zeyuan Ding, Jiayu Hu, Hanzhe Shan, et al. Pelican-VL 1.0: A foundation brain model for embodied intelligence. *arXiv preprint arXiv:2511.00108*, 2025.

Yueer Zhou, Yichen Wu, and Ying Wei. Resolving conflicts in lifelong learning via aligning updates in subspaces. *arXiv preprint arXiv:2512.08960*, 2025.

Ding Zou, Feifan Wang, Mengyu Ge, Siyuan Fan, Zongbing Zhang, Wei Chen, Lingfeng Wang, Zhongyou Hu, et al. EmbodiedBrain: Expanding performance boundaries of task planning for embodied intelligence. *arXiv preprint arXiv:2510.20578*, 2025.

A ReCAP Algorithm Summary

ReCAP composes each weight matrix equipped with LoRA separately; we call such a matrix an *adapted layer*. At a given adapted layer, the capability modules provide effective updates of the same shape, although their LoRA factor bases may differ. ReCAP therefore first constructs common bases and expresses the selected updates as directly comparable coordinate matrices M_i. It then retains the reference update M_a and forms donor residuals $T_i = M_i - M_a$, turning composition into the integration of additional changes relative to one starting solution.

To concentrate on the dominant directions of these changes, ReCAP learns a joint residual subspace from normalized residuals and projects the original residuals into it. Within these coordinates, magnitude-weighted sign election precedes averaging over sign-consistent values; components missed by the finite-dimensional subspace are added back with smaller weights. Finally, a support mask constructed from the full updates regulates residual injection at each coordinate. The filtered residual is added to the reference and reconstructed as an update to the original weight matrix. Repeating this process across adapted layers produces one deployable static checkpoint.

The main objects in this sequence are

$$\begin{aligned} M_i &= U^\top \Delta W_i^{\text{eff}} V, \qquad T_i = M_i - M_a, \\ Z_i &= Q_L^\top T_i Q_R, \qquad M_\star = M_a + \widetilde{T}_\star, \end{aligned} \tag{12}$$

Here $\widetilde{T}_\star$ is the residual after aggregation and support gating, so $M_\star = M_a + \widetilde{T}_\star$ represents the final injection step. The two earlier changes of representation serve different purposes: common bases U, V are built from all selected LoRAs to compare full updates, whereas Q_L, Q_R are learned from reference-relative residuals to concentrate on their directions. Table 7 lists the objects in this dependency order, connecting their construction to their use.

Table 7: The main objects used by ReCAP.

Object	Constructed from	Purpose
U, V	Selected LoRA factors	Put every selected update in the same coordinate system
M_i	$U^\top \Delta W_i^{\text{eff}} V$	Represent the complete update of module i
T_i	$M_i - M_a$	Express module i as an additional change to the reference
Q_L, Q_R	Normalized $\{\widehat{T}_i\}$	Identify a compact subspace that captures dominant changes across the residual set
Z_i	$Q_L^\top T_i Q_R$	Aggregate the residuals by magnitude-weighted vote
S	Full updates $\{M_i\}$	Control residual injection using consensus support

These objects specify the per-layer computation. Table 8 instantiates the participating modules and their reference or donor roles for the primary InternVL3.5-14B setting. Training emphasis identifies their capability sources, while the composition role determines whether an update is retained directly or first expressed as a residual.

Given the modules, reference, and donor coefficients, Algorithm 1 applies the sequence to every adapted layer and writes reconstructed weights into one base-model copy. The per-layer compositions thereby form a single static checkpoint.

Table 8: Capability modules in the primary InternVL3.5-14B setting. Training emphasis and supervision define each module, which serves as either the reference or a donor in the main composition.

Module	Training emphasis	Dominant and auxiliary supervision	Merge role
NL1	Grounded interaction	Grounded QA and pointing, with auxiliary spatial and contextual QA	Reference
NL2	Spatial reasoning	Spatial reasoning and geometric layout, with auxiliary grounding support	Residual
NL3	Temporal and task-planning reasoning	Temporal and action-sequence QA and planning supervision, with auxiliary scene understanding	Residual
NL4	Video-semantic next-action reasoning	Video next-action and subtask supervision, with auxiliary static-scene semantics	Residual

Algorithm 1 ReCAP

Require: Base weights W_0; selected LoRA modules; reference a; donor coefficients $\{m_i\}_{i\in\mathcal{D}}$; subspace, residual-gate, and support hyperparameters
Ensure: One static checkpoint $\theta_\star$
Initialize $\theta_\star$ as a copy of the base model
for each adapted layer **do**
  Compute effective updates $\Delta W_i^{\text{eff}} = \frac{\alpha_i}{r_i} B_i A_i$
  Build common bases U, V jointly and compute $M_i = U^\top \Delta W_i^{\text{eff}} V$
  Keep M_a and form residuals $T_i = M_i - M_a$ for $i \in \mathcal{D}$
  Learn the joint residual subspace Q_L, Q_R from normalized $\widehat{T}_i$ with normalized donor weights using Eq. 4
  Merge $Z_i = Q_L^\top T_i Q_R$ with Eqs. 5–7
  Merge residual tracks with Eq. 51 and add the gated components
  Build support mask S from all full updates using Eq. 52
  Form $M_\star$ with Eq. 9
  Reconstruct $\Delta W_\star = U M_\star V^\top$ and set $W_\star = W_0 + \Delta W_\star$
  Write $W_\star$ into the corresponding layer of $\theta_\star$
end for
return $\theta_\star$

B Joint Residual Subspace: Objective and Optimization

Appendix A reduces composition to integrating residuals around a reference. This section derives the shared low-dimensional representation used for those residuals in Section 3.3, with inputs $X_t = T_i = M_i - M_a$. We first pose a weighted reconstruction problem with shared bilateral bases, solve for the optimal coordinates at fixed bases, and eliminate those coordinates to obtain the captured-energy objective. We then derive the left- and right-basis updates, state the alternating procedure and its convergence limits, and explain how subspace discovery relates to subsequent sign aggregation.

B.1 Reconstruction in a Shared Bilateral Subspace

Let $\widehat{X}_1, \ldots, \widehat{X}_T \in \mathbb{R}^{d_L \times d_R}$ denote the normalized input residual matrices, with residual weights $w_1, \ldots, w_T > 0$ satisfying $\sum_t w_t = 1$. Here w_t corresponds to the normalized donor weight u_i in the main text; the positive-weight assumption identifies the residuals whose coordinates must be optimized. The implementation also permits zero-weight donors. They do not affect this reconstruction objective, and their coordinates, although not uniquely determined by it, are computed by the same bilateral projection $Q_L^\top \widehat{X}_t Q_R$. With this convention, we seek orthonormal bases $Q_L \in \mathbb{R}^{d_L \times k_L}$ and $Q_R \in \mathbb{R}^{d_R \times k_R}$ together with per-residual coefficient matrices $Z_t \in \mathbb{R}^{k_L \times k_R}$ that

jointly minimize the weighted reconstruction error:

$$\min_{\substack{Q_L \in \mathbb{R}^{d_L \times k_L},\, Q_R \in \mathbb{R}^{d_R \times k_R} \\ Z_1, \ldots, Z_T \in \mathbb{R}^{k_L \times k_R}}} \sum_{t=1}^{T} w_t \left\| \widehat{X}_t - Q_L\, Z_t\, Q_R^\top \right\|_F^2 \tag{13}$$

subject to the orthogonality constraints

$$Q_L^\top Q_L = I_{k_L}, \qquad Q_R^\top Q_R = I_{k_R}. \tag{14}$$

The product $Q_L Z_t Q_R^\top$ approximates $\widehat{X}_t$ through a common pair of orthonormal bases (Q_L, Q_R), where the per-residual "coordinates" Z_t capture how each residual is represented in the learned low-rank geometry.

B.2 Optimal Residual Coordinates for Fixed Bases

To reduce Eq. (13) to an optimization over the bases alone, first fix Q_L, Q_R satisfying Eq. (14) and solve for each Z_t. Because the coordinates of different residuals are independent and each weight is positive, this step separates into unconstrained least-squares problems. Define the error $\mathcal{E}_t \coloneqq \widehat{X}_t - Q_L Z_t Q_R^\top$ and use $\|X\|_F^2 = \operatorname{tr}(X^\top X)$ to expose its dependence on Z_t:

$$\begin{aligned} \|\mathcal{E}_t\|_F^2 &= \operatorname{tr}\big(\mathcal{E}_t^\top \mathcal{E}_t\big) \\ &= \operatorname{tr}\Big[\big(\widehat{X}_t - Q_L Z_t Q_R^\top\big)^\top \big(\widehat{X}_t - Q_L Z_t Q_R^\top\big)\Big] \\ &= \operatorname{tr}\Big[\big(\widehat{X}_t^\top - Q_R Z_t^\top Q_L^\top\big)\big(\widehat{X}_t - Q_L Z_t Q_R^\top\big)\Big]. \end{aligned} \tag{15}$$

Distributing the product inside the trace:

$$\|\mathcal{E}_t\|_F^2 = \operatorname{tr}\big(\widehat{X}_t^\top \widehat{X}_t\big) - \operatorname{tr}\big(\widehat{X}_t^\top Q_L Z_t Q_R^\top\big) - \operatorname{tr}\big(Q_R Z_t^\top Q_L^\top \widehat{X}_t\big) + \operatorname{tr}\big(Q_R Z_t^\top Q_L^\top Q_L Z_t Q_R^\top\big). \tag{16}$$

We now separate the constant, linear, and quadratic terms using trace identities and orthonormality.

First, $\operatorname{tr}(\widehat{X}_t^\top \widehat{X}_t) = \|\widehat{X}_t\|_F^2$ is constant with respect to Z_t.

For the first cross term, use the cyclic property of the trace, $\operatorname{tr}(ABC) = \operatorname{tr}(CAB) = \operatorname{tr}(BCA)$:

$$\begin{aligned} \operatorname{tr}\big(\widehat{X}_t^\top Q_L Z_t Q_R^\top\big) &= \operatorname{tr}\big(Q_R^\top \widehat{X}_t^\top Q_L Z_t\big) \\ &= \operatorname{tr}\big(Z_t^\top Q_L^\top \widehat{X}_t Q_R\big), \end{aligned} \tag{17}$$

where the last equality uses $\operatorname{tr}(X) = \operatorname{tr}(X^\top)$.

The other cross term admits the same simplification. By the same cyclic property:

$$\operatorname{tr}\big(Q_R Z_t^\top Q_L^\top \widehat{X}_t\big) = \operatorname{tr}\big(Z_t^\top Q_L^\top \widehat{X}_t Q_R\big). \tag{18}$$

Hence Terms 2 and 3 are identical.

Finally, for the quadratic term, apply the orthogonality constraints $Q_L^\top Q_L = I_{k_L}$ and $Q_R^\top Q_R = I_{k_R}$:

$$\begin{aligned} \operatorname{tr}\big(Q_R Z_t^\top \underbrace{Q_L^\top Q_L}_{I_{k_L}} Z_t Q_R^\top\big) &= \operatorname{tr}\big(Q_R Z_t^\top Z_t Q_R^\top\big) \\ &= \operatorname{tr}\big(Q_R^\top Q_R Z_t^\top Z_t\big) && \text{(trace cycling)} \\ &= \operatorname{tr}\big(\underbrace{Q_R^\top Q_R}_{I_{k_R}} Z_t^\top Z_t\big) \\ &= \operatorname{tr}\big(Z_t^\top Z_t\big) = \|Z_t\|_F^2. \end{aligned} \tag{19}$$

The two identical cross terms can now be combined with the simplified quadratic term. Substituting (17)–(19) into (16):

$$\boxed{\|\widehat{X}_t - Q_L Z_t Q_R^\top\|_F^2 = \|\widehat{X}_t\|_F^2 - 2\, \operatorname{tr}\big(Z_t^\top Q_L^\top \widehat{X}_t Q_R\big) + \|Z_t\|_F^2.} \tag{20}$$

The reconstruction error is thus an explicit quadratic function of Z_t. Treating (20) as a function of Z_t and differentiating using the standard matrix calculus identities $\frac{\partial}{\partial Z}\operatorname{tr}(Z^\top C) = C$ and $\frac{\partial}{\partial Z}\operatorname{tr}(Z^\top Z) = 2Z$:

$$\frac{\partial}{\partial Z_t}\|\mathcal{E}_t\|_F^2 = -2\,Q_L^\top \widehat{X}_t Q_R + 2\,Z_t. \tag{21}$$

Setting this to zero:

$$\boxed{Z_t^\star = Q_L^\top \widehat{X}_t\, Q_R.} \tag{22}$$

This is a global minimum since the Hessian $\frac{\partial^2}{\partial Z_t^2}\|\mathcal{E}_t\|_F^2 = 2\,I$ is positive definite.

Consequently, fixed bases require no additional iteration over coordinates: the optimal coefficient matrix $Z_t^\star$ is obtained by projecting $\widehat{X}_t$ onto the learned bases from both sides. When Q_L and Q_R span a joint residual subspace, $Z_t^\star$ captures how residual t is represented within that geometry.

B.3 Captured-Energy Objective after Eliminating Coordinates

Having determined the best coordinates for given bases, we eliminate them to compare how much residual energy different bases retain. Substituting $Z_t^\star = Q_L^\top \widehat{X}_t Q_R$ into (20):

$$\|\widehat{X}_t - Q_L Z_t^\star Q_R^\top\|_F^2 = \|\widehat{X}_t\|_F^2 - 2\,\operatorname{tr}\big(Z_t^{\star\top} Q_L^\top \widehat{X}_t Q_R\big) + \|Z_t^\star\|_F^2. \tag{23}$$

We first evaluate $\operatorname{tr}(Z_t^{\star\top} Q_L^\top \widehat{X}_t Q_R)$. Substituting $Z_t^\star = Q_L^\top \widehat{X}_t Q_R$:

$$\begin{aligned}\operatorname{tr}\big(Z_t^{\star\top} Q_L^\top \widehat{X}_t Q_R\big) &= \operatorname{tr}\big((Q_L^\top \widehat{X}_t Q_R)^\top (Q_L^\top \widehat{X}_t Q_R)\big)\\ &= \operatorname{tr}\big(Q_R^\top \widehat{X}_t^\top Q_L\, Q_L^\top \widehat{X}_t Q_R\big)\\ &= \|Q_L^\top \widehat{X}_t Q_R\|_F^2.\end{aligned} \tag{24}$$

Similarly, $\|Z_t^\star\|_F^2 = \|Q_L^\top \widehat{X}_t Q_R\|_F^2$. Substituting both into (23):

$$\begin{aligned}\|\widehat{X}_t - Q_L Z_t^\star Q_R^\top\|_F^2 &= \|\widehat{X}_t\|_F^2 - 2\,\|Q_L^\top \widehat{X}_t Q_R\|_F^2 + \|Q_L^\top \widehat{X}_t Q_R\|_F^2\\ &= \|\widehat{X}_t\|_F^2 - \|Q_L^\top \widehat{X}_t Q_R\|_F^2.\end{aligned} \tag{25}$$

Weighting this per-residual identity by w_t and summing eliminates all coordinate variables from the original objective. The total weighted reconstruction error at the optimum becomes:

$$\sum_{t=1}^{T} w_t \|\widehat{X}_t - Q_L Z_t^\star Q_R^\top\|_F^2 = \underbrace{\sum_{t=1}^{T} w_t \|\widehat{X}_t\|_F^2}_{\text{constant w.r.t. } Q_L, Q_R} - \sum_{t=1}^{T} w_t \|Q_L^\top \widehat{X}_t Q_R\|_F^2. \tag{26}$$

Since the first term is fixed, minimizing reconstruction error over (Q_L, Q_R) is equivalent to maximizing the captured energy:

$$\boxed{\max_{\substack{Q_L^\top Q_L = I_{k_L}\\ Q_R^\top Q_R = I_{k_R}}} \sum_{t=1}^{T} w_t \left\|Q_L^\top \widehat{X}_t\, Q_R\right\|_F^2.} \tag{27}$$

This is Eq. (4) in the main text.

B.4 Left-Basis Update with the Right Basis Fixed

The reduced objective still requires a computable update for each basis. We first isolate the dependence on the left basis while holding the right basis fixed.

Fix Q_R satisfying $Q_R^\top Q_R = I_{k_R}$. We rewrite each summand in (27) using trace notation. For any matrix X, $\|X\|_F^2 = \mathrm{tr}(X^\top X)$, so:

$$\begin{aligned}\|Q_L^\top \widehat{X}_t Q_R\|_F^2 &= \mathrm{tr}\big((Q_L^\top \widehat{X}_t Q_R)^\top (Q_L^\top \widehat{X}_t Q_R)\big) \\ &= \mathrm{tr}\big(Q_R^\top \widehat{X}_t^\top Q_L Q_L^\top \widehat{X}_t Q_R\big). \end{aligned} \tag{28}$$

For a form directly optimized over Q_L, cycle the trace to place $Q_L^\top$ and Q_L at the two ends:

$$\begin{aligned}\mathrm{tr}\big(Q_R^\top \widehat{X}_t^\top Q_L\, Q_L^\top \widehat{X}_t Q_R\big) &= \mathrm{tr}\big(Q_L^\top \widehat{X}_t Q_R\, Q_R^\top \widehat{X}_t^\top Q_L\big) \\ &\qquad \text{(by cyclicity of the trace)} \\ &= \mathrm{tr}\big(Q_L^\top\, \widehat{X}_t Q_R Q_R^\top \widehat{X}_t^\top\, Q_L\big). \end{aligned} \tag{29}$$

The last step groups $\widehat{X}_t Q_R Q_R^\top \widehat{X}_t^\top$ into a single $d_L \times d_L$ matrix.

Because all residuals share the same bases, their contributions to the left-basis objective must now be combined. Summing over t with weights w_t:

$$\begin{aligned}\sum_{t=1}^{T} w_t \|Q_L^\top \widehat{X}_t Q_R\|_F^2 &= \sum_{t=1}^{T} w_t\ \mathrm{tr}\big(Q_L^\top \widehat{X}_t Q_R Q_R^\top \widehat{X}_t^\top Q_L\big) \\ &= \mathrm{tr}\Big(Q_L^\top \underbrace{\Big[\sum_{t=1}^{T} w_t\, \widehat{X}_t Q_R Q_R^\top \widehat{X}_t^\top\Big]}_{=:\, A_L^{\mathrm{span}}} Q_L\Big). \end{aligned} \tag{30}$$

Define the *left span matrix*:

$$\boxed{A_L^{\mathrm{span}} \coloneqq \sum_{t=1}^{T} w_t\, \widehat{X}_t\, Q_R\, Q_R^\top\, \widehat{X}_t^\top \ \in\ \mathbb{R}^{d_L \times d_L}.} \tag{31}$$

Note that A_L^{span} is symmetric positive semidefinite: for any vector v,

$$v^\top A_L^{\mathrm{span}} v = \sum_t w_t\, \|Q_R^\top \widehat{X}_t^\top v\|_2^2 \geq 0. \tag{32}$$

With the right basis fixed, the energy objective therefore becomes a standard trace-maximization problem. The optimization over Q_L is:

$$\boxed{\max_{Q_L^\top Q_L = I_{k_L}} \mathrm{tr}\big(Q_L^\top A_L^{\mathrm{span}} Q_L\big).} \tag{33}$$

By the **Ky Fan trace maximization theorem** (Fan, 1949), the maximum is achieved when the columns of Q_L are the eigenvectors corresponding to the k_L largest eigenvalues of A_L^{span}. The optimal value equals the sum of these k_L eigenvalues:

$$\mathrm{tr}\big(Q_L^{\star\top} A_L^{\mathrm{span}} Q_L^\star\big) = \sum_{i=1}^{k_L} \lambda_i\big(A_L^{\mathrm{span}}\big), \tag{34}$$

where $\lambda_1 \geq \lambda_2 \geq \cdots$ denote the eigenvalues of A_L^{span} in descending order.

B.5 Right-Basis Update with the Left Basis Fixed

After the left-basis update, the same argument supplies the complementary right-basis update with the new left basis held fixed.

By an entirely symmetric argument, fixing Q_L and optimizing over Q_R, we start from:

$$\begin{aligned}\|Q_L^\top \widehat{X}_t Q_R\|_F^2 &= \mathrm{tr}\big((Q_L^\top \widehat{X}_t Q_R)^\top (Q_L^\top \widehat{X}_t Q_R)\big) \\ &= \mathrm{tr}\big(Q_R^\top \widehat{X}_t^\top Q_L Q_L^\top \widehat{X}_t Q_R\big). \end{aligned} \tag{35}$$

This time, the expression is already in the form $\mathrm{tr}(Q_R^\top \cdot [\cdot] \cdot Q_R)$. Define the *right span matrix*:

$$\boxed{A_R^{\mathrm{span}} \coloneqq \sum_{t=1}^{T} w_t \, \widehat{X}_t^\top \, Q_L \, Q_L^\top \, \widehat{X}_t \;\in\; \mathbb{R}^{d_R \times d_R}.} \tag{36}$$

The right subproblem is:

$$\max_{Q_R^\top Q_R = I_{k_R}} \mathrm{tr}\big(Q_R^\top A_R^{\mathrm{span}} Q_R\big), \tag{37}$$

solved by the top-k_R eigenvectors of A_R^{span} (again by the Ky Fan theorem).

B.6 Alternating Updates, Stopping, and Convergence Limits

The coupled problem (27) is non-convex in (Q_L, Q_R) jointly but becomes a standard eigenvalue problem when one variable is fixed. This motivates the following alternating scheme.

Initializing the subspace from the weighted mean. Compute the weighted mean of the normalized residual input matrices $\widehat{\bar{X}} = \sum_{t=1}^{T} w_t \, \widehat{X}_t$ and its SVD $\widehat{\bar{X}} = \bar{U} \, \bar{\Sigma} \, \bar{V}^\top$. Set $Q_L^{(0)} = \bar{U}_{[:,1:k_L]}$ and $Q_R^{(0)} = \bar{V}_{[:,1:k_R]}$, where $k_L \le \min(d_L, d_R)$ and $k_R \le \min(d_L, d_R)$ so that sufficiently many singular vectors are available. The first-order mean can cancel high-energy directions, and alternating updates may remain at a suboptimal solution.

Alternating basis updates and stopping check. After initialization, repeat the following steps for $\ell = 0, 1, 2, \ldots$.

1. **Left update.** Given $Q_R^{(\ell)}$, form $A_L^{(\ell)} = \sum_t w_t \, \widehat{X}_t Q_R^{(\ell)} {Q_R^{(\ell)}}^\top \widehat{X}_t^\top$ and set $Q_L^{(\ell+1)}$ to its top-k_L eigenvectors.
2. **Right update.** Given $Q_L^{(\ell+1)}$, form $A_R^{(\ell)} = \sum_t w_t \, \widehat{X}_t^\top Q_L^{(\ell+1)} {Q_L^{(\ell+1)}}^\top \widehat{X}_t$ and set $Q_R^{(\ell+1)}$ to its top-k_R eigenvectors.
3. **Convergence check.** Compute the Frobenius distance between successive subspace projectors:

$$\delta^{(\ell)} = \big\| Q_L^{(\ell+1)} {Q_L^{(\ell+1)}}^\top - Q_L^{(\ell)} {Q_L^{(\ell)}}^\top \big\|_F. \tag{38}$$

   Stop when $\delta^{(\ell)} < 10^{-6}$.

Objective convergence versus basis convergence. Each half-step solves a Rayleigh–Ritz subproblem exactly (by the Ky Fan theorem), so the objective (27) is nondecreasing after every block update. Because the objective is bounded above by $\sum_t w_t \|\widehat{X}_t\|_F^2$, the sequence of *objective values* converges monotonically. Note that this guarantees convergence of the objective, not necessarily of the iterates $(Q_L^{(\ell)}, Q_R^{(\ell)})$ themselves; when the top eigenspace of $A_L^{(\ell)}$ or $A_R^{(\ell)}$ is degenerate, multiple equally optimal bases exist. The implemented stopping check measures only the left projector; it does not guarantee that the right projector has stabilized.

Finite iteration budget in the experiments. The monotonicity argument concerns the objective sequence without a fixed iteration limit. The primary InternVL3.5-14B configuration instead sets `max_outer_iter` $= 5$, allowing at most five rounds; this finite budget does not imply that every adapted layer reaches the stopping tolerance.

B.7 Role of the Energy Objective and Its Relation to Aggregation

The preceding derivation explains how the energy objective is optimized. It remains to explain why subspace discovery uses this objective instead of simply retaining the mean-SVD initialization.

A natural alternative is to take the SVD of the weighted mean $\bar{X} = \sum_t w_t \widehat{X}_t$. ReCAP instead learns the subspace from the sum of per-residual projection energies.

Directions remain visible before sign resolution. A signed mean can cancel a direction when residuals use it with opposite signs. The energy objective sums $\|Q_L^\top \widehat{X}_t Q_R\|_F^2$, so opposite signs do not cancel in the objective. This property does not guarantee that weighted-mean initialization and finite alternating updates discover the direction.

Normalized residuals contribute on a comparable scale. The normalization $\widehat{X}_t = X_t/(\|X_t\|_F + \epsilon_n)$ reduces the influence of a single large-norm residual on subspace selection. When several residuals use similar row–column directions, their captured energy accumulates in the objective.

Subspace discovery precedes value aggregation. The bilateral objective determines where the residuals place substantial update energy, using normalized inputs $\widehat{T}_i = (M_i - M_a)/(\|M_i - M_a\|_F + \epsilon_n)$. Once the bases are fixed, aggregation instead uses the original-scale coordinates $Z_i = Q_L^\top T_i Q_R$. Equation (7) then chooses the better-supported sign and averages the corresponding values. Thus, the optimal coordinates of the normalized reconstruction problem above should be distinguished from the unnormalized coordinates used for merging.

C Orthogonal Residual Decomposition and Component Merging

Once the joint residual subspace is fixed, projection represents only part of each residual. Discarding the remainder would also discard update directions that the subspace does not capture. We therefore derive an exact decomposition that retains both parts and use it to define the merge tracks. The decomposition and orthogonality concern the input residuals; the subsequent composition rule determines how each track is aggregated and weighted.

C.1 Exact Decomposition by Bilateral Projection

To distinguish whether each side of a residual lies in the learned subspace, construct projections from the orthonormal bases $Q_L \in \mathbb{R}^{d_L \times k_L}$ and $Q_R \in \mathbb{R}^{d_R \times k_R}$. Because $Q_L^\top Q_L = I$ and $Q_R^\top Q_R = I$, the subspace and orthogonal-complement projections are

$$P_L = Q_L Q_L^\top \in \mathbb{R}^{d_L \times d_L}, \qquad P_L^\perp = I_{d_L} - P_L, \tag{39}$$

$$P_R = Q_R Q_R^\top \in \mathbb{R}^{d_R \times d_R}, \qquad P_R^\perp = I_{d_R} - P_R. \tag{40}$$

These projections partition each side into complementary parts. On the left, idempotence, mutual exclusion, and completeness give

$$P_L^2 = P_L, \quad {P_L^\perp}^2 = P_L^\perp, \quad P_L P_L^\perp = P_L^\perp P_L = 0, \quad P_L + P_L^\perp = I_{d_L}, \tag{41}$$

The right projections $P_R, P_R^\perp$ satisfy the same relations. Inserting both complete decompositions around any original residual $X_t \in \mathbb{R}^{d_L \times d_R}$ therefore yields

$$\begin{aligned} X_t &= I_{d_L}\, X_t\, I_{d_R} \\ &= (P_L + P_L^\perp)\, X_t\, (P_R + P_R^\perp) \\ &= P_L X_t P_R + P_L X_t P_R^\perp + P_L^\perp X_t P_R + P_L^\perp X_t P_R^\perp. \end{aligned} \tag{42}$$

The four terms sum exactly to the input; no low-rank approximation has yet been made. The first term, $S_t = P_L X_t P_R$, lies in the learned subspaces on both sides. Expanding its projections gives

$$\begin{aligned} S_t &= Q_L Q_L^\top X_t\, Q_R Q_R^\top \\ &= Q_L \underbrace{(Q_L^\top X_t Q_R)}_{Z_t^{\mathrm{raw}}} Q_R^\top \\ &= Q_L\, Z_t^{\mathrm{raw}}\, Q_R^\top, \end{aligned} \tag{43}$$

Here $Z_t^{\mathrm{raw}} = Q_L^\top X_t Q_R$ preserves the original residual scale and supplies the coordinates aggregated in Section 3.4. The other three terms describe what this projection omits: $P_L X_t P_R^\perp$ overlaps the learned subspace only on the left, $P_L^\perp X_t P_R$ only on the right, and $R_t^\perp = P_L^\perp X_t P_R^\perp$ lies in the complements on both sides. We group the two one-sided overlaps as the cross residual

$$R_t^\times \coloneqq P_L X_t P_R^\perp + P_L^\perp X_t P_R. \tag{44}$$

The four projection blocks thus form three components corresponding to the later in-span, cross, and orthogonal merge tracks:

$$\boxed{X_t = S_t + R_t^{\times} + R_t^{\perp} = \underbrace{Q_L Z_t^{\mathrm{raw}} Q_R^{\top}}_{\text{in-span}} + \underbrace{P_L X_t P_R^{\perp} + P_L^{\perp} X_t P_R}_{\text{cross residual}} + \underbrace{P_L^{\perp} X_t P_R^{\perp}}_{\text{orthogonal residual}}}. \tag{45}$$

C.2 Input Orthogonality and Energy Allocation

Beyond reconstructing the input, these components partition its energy without overlap. To establish this, consider the four primitive blocks under the Frobenius inner product $\langle X, Y\rangle_F = \mathrm{tr}(X^{\top}Y)$. Any two distinct blocks use complementary projections on at least one side; cyclic trace rearrangement then exposes either $P_L P_L^{\perp} = 0$ or $P_R P_R^{\perp} = 0$. The six pairings are

$$\begin{aligned}
\langle S_t, R_t^{\perp}\rangle_F &= \mathrm{tr}(P_R X_t^{\top} \underbrace{P_L P_L^{\perp}}_{0} X_t P_R^{\perp}) &&= 0, \\
\langle S_t, P_L X_t P_R^{\perp}\rangle_F &= \mathrm{tr}(X_t \underbrace{P_R^{\perp} P_R}_{0} X_t^{\top} P_L) &&= 0, \\
\langle S_t, P_L^{\perp} X_t P_R\rangle_F &= \mathrm{tr}(P_R X_t^{\top} \underbrace{P_L P_L^{\perp}}_{0} X_t P_R) &&= 0, \\
\langle P_L X_t P_R^{\perp}, P_L^{\perp} X_t P_R\rangle_F &= \mathrm{tr}(P_R^{\perp} X_t^{\top} \underbrace{P_L P_L^{\perp}}_{0} X_t P_R) &&= 0, \\
\langle P_L X_t P_R^{\perp}, R_t^{\perp}\rangle_F &= \mathrm{tr}(P_R^{\perp} X_t^{\top} \underbrace{P_L P_L^{\perp}}_{0} X_t P_R^{\perp}) &&= 0, \\
\langle P_L^{\perp} X_t P_R, R_t^{\perp}\rangle_F &= \mathrm{tr}(X_t \underbrace{P_R^{\perp} P_R}_{0} X_t^{\top} P_L^{\perp}) &&= 0.
\end{aligned} \tag{46}$$

Both blocks forming the cross residual are also orthogonal to the in-span and fully orthogonal components. Grouping them therefore introduces no cross-energy terms, so Eq. (45) satisfies

$$\|X_t\|_F^2 = \|S_t\|_F^2 + \|R_t^{\times}\|_F^2 + \|R_t^{\perp}\|_F^2. \tag{47}$$

This identity establishes that outside-subspace energy is part of the input that S_t does not account for. It motivates retaining auxiliary residuals, but does not determine their best composition weights.

C.3 Three Merge Tracks from the Input Decomposition

ReCAP concentrates conflict handling in the learned subspace: magnitude-weighted sign aggregation acts on Z_t^{raw}, and reconstruction yields $T_{\star}^{\mathrm{cov}}$. To supplement changes omitted by the finite-dimensional projection, mild TIES separately produces the cross-track output $R_{\times}^{\star}$, while weighted averaging produces $R_{\perp}^{\star}$. Appendix D.1 specifies these operators. Their outputs are combined through gates as

$$T_{\star}^{\mathrm{aug}} = T_{\star}^{\mathrm{cov}} + \lambda_{\times}\, R_{\times}^{\star} + \lambda_{\perp}\, R_{\perp}^{\star}, \tag{48}$$

The coefficients $\lambda_{\times}, \lambda_{\perp} \in [0, 1]$ regulate the two auxiliary contributions. The primary InternVL3.5-14B configuration uses $\lambda_{\times} = 0.25$ and $\lambda_{\perp} = 0.5$, assigning the orthogonal track a larger multiplier. This is a composition choice, not a consequence of orthogonality. Since the track norms can differ, the larger multiplier does not by itself imply a larger injected energy.

C.4 Limits of Transferring Input Properties to the Output

Equation (47) cannot be applied directly to the three merged outputs. It relies on orthogonality of the input projection blocks, whereas coordinate trimming, sign selection, and subsequent coordinate-wise support gating need not preserve those subspaces. Track names therefore identify input origins without guaranteeing mutually orthogonal outputs. Even when weighted averaging alone preserves the orthogonal track's linear subspace, the final support gate may change that membership.

The same distinction matters for the reference: its coefficient of one specifies the starting update, while the added residual still changes the final parameters. Retaining the reference in the composition

formula consequently does not guarantee unchanged reference capability; that effect must be assessed empirically.

D Composition Operators and Numerical Implementation

Appendix C identifies three residual components. We now specify their operators in computational order: capture rates first adjust within-subspace donor weights, trimming and sign aggregation then define the merge rule, and these operators are applied to residual tracks and full-update support construction. Input conditions and the effect of numerical-rank truncation complete the specification.

D.1 Residual Aggregation and Support Construction

D.1.1 Within-Subspace Weights from Energy Capture

Different donor residuals can place different fractions of their energy in the learned subspace. ReCAP therefore makes a bounded correction to their original weights according to the captured fraction. Since Q_L, Q_R have orthonormal columns, the squared norm of $Z_i = Q_L^\top T_i Q_R$ equals the energy of its reconstructed component:

$$\|Q_L Q_L^\top T_i Q_R Q_R^\top\|_F^2 = \|Q_L Z_i Q_R^\top\|_F^2 = \|Z_i\|_F^2.$$

This identity lets us define capture as a fraction of full residual energy, clip and interpolate that fraction into a correction factor, and renormalize the resulting weights:

$$\begin{aligned} \eta_i^{\mathrm{cap}} &= \frac{\|Z_i\|_F^2}{\|T_i\|_F^2 + \epsilon_c}, \\ f_i &= (1-\beta) + \beta\ \mathrm{clip}(\eta_i^{\mathrm{cap}}, \eta_{\min}, 1), \\ \omega_i &= \frac{u_i f_i}{\sum_{j\in\mathcal{D}} u_j f_j}. \end{aligned}$$

Here $\mathrm{clip}(x, a, b) = \min\{b, \max\{a, x\}\}$. Setting $\beta = 0.5$, $\eta_{\min} = 0.2$, and $\epsilon_c = 10^{-12}$ makes the correction nondecreasing in capture rate but bounded in $[0.6, 1]$. Higher-capture donors thus receive greater relative emphasis without excluding a positive-weight donor solely for low capture. The same equations cover the boundary cases: a zero residual has zero capture, while a donor with $u_i = 0$ retains $\omega_i = 0$ after correction.

D.1.2 From Matrix-Wise Trimming to Sign-Consistent Aggregation

With weights specified, aggregation must also decide which coordinates to retain and how to resolve opposing signs. For magnitude trimming, flatten $|X|$, define $\tau_\rho(X)$ as its $K_\rho(X)$-th largest entry, and set

$$K_\rho(X) = \max\{1, \lfloor \rho\ \mathrm{numel}(X) + 1/2 \rfloor\}, \qquad \mathcal{H}_\rho(X) = \mathbf{1}\{|X| \geq \tau_\rho(X)\}, \tag{49}$$

All entries tied at the threshold are retained, so actual support can exceed the nominal fraction. After applying this mask to each input, nonnegative weights w summing to one elect a sign, and only values consistent with that sign enter the weighted average:

$$\begin{aligned} \widetilde{X}_i &= \mathcal{H}_\rho(X_i) \odot X_i, & s[p,q] &= \mathrm{sign}\left(\sum_i w_i \widetilde{X}_i[p,q]\right), \\ I_i[p,q] &= \mathbf{1}\{\mathrm{sign}(\widetilde{X}_i[p,q]) = s[p,q] \neq 0\}, & \mathcal{T}_\rho(\{X_i\}; w)[p,q] &= \frac{\sum_i w_i \widetilde{X}_i[p,q] I_i[p,q]}{\sum_i w_i I_i[p,q] + \varepsilon}. \end{aligned} \tag{50}$$

Equation (50) specifies trimming, sign election, and denominator normalization together. Each merge track therefore only needs its inputs, density, and weights specified. Within-subspace aggregation uses $\mathcal{T}_1(\{Z_i\}; \omega)$, retaining all coordinates and using capture-corrected weights. The outside-subspace tracks instead use the original donor weights u:

$$R_\times^\star = \mathcal{T}_{0.2}(\{R_i^\times\}; u), \qquad R_\perp^\star = \sum_{i\in\mathcal{D}} u_i R_i^\perp. \tag{51}$$

This distinction confines capture-rate correction to the subspace it directly measures, while the cross and orthogonal tracks retain their respective merge rules.

D.1.3 Support from Full Updates for Residual Injection

Residual aggregation determines which change to inject; support construction determines where injection receives greater strength. Support therefore uses full updates M_i, including the reference, rather than the donor residuals above. Equal-weight TIES first forms a consensus map, from which the support mask is extracted:

$$C_{\mathrm{cons}} = \mathcal{T}_{0.1}(\{M_i\}_{i\in\mathcal{D}\cup\{a\}}; w^{\mathrm{eq}}), \quad w_i^{\mathrm{eq}} = \frac{1}{|\mathcal{D}|+1}, \quad S = \mathcal{H}_{0.1}(C_{\mathrm{cons}}). \tag{52}$$

The resulting S controls coordinate-wise residual injection in Eq. (9). Because support construction and residual aggregation use different inputs and weights, a zero residual-aggregation weight does not remove a module from full-update support construction.

D.2 Numerical Conditions and Common-Coordinate Reconstruction

These computations use float64 cores, stabilizers $\epsilon_n = \epsilon_c = \varepsilon = 10^{-12}$ for normalization, capture, and aggregation, and effective LoRA scaling $\alpha/r = 2$. The common coordinates come from SVDs of concatenated factors, retaining only directions with $\sigma_j > 10^{-10}\sigma_1$. This threshold determines the numerical core dimensions d_L, d_R. Requested subspace dimensions are clipped to the corresponding core dimensions; fixed-size thin-SVD initialization additionally requires $k_L, k_R \le \min(d_L, d_R)$ so that enough initial singular vectors are available on both sides.

The specification assumes nonempty numerical cores, at least one donor, and positive total donor weight, without which the stated projections or weight normalization cannot be applied. The code rejects missing donors and invalid weights. For permitted zero-weight donors, only the weighted residual contribution vanishes; their full updates still participate in common-coordinate and support construction.

Numerical-rank truncation also limits reconstruction exactness. With U, V denoting the retained orthonormal bases, reconstruction from $M_i = U^\top \Delta W_i^{\mathrm{eff}} V$ yields

$$\widehat{\Delta W}_i = U M_i V^\top = U U^\top \Delta W_i^{\mathrm{eff}} V V^\top.$$

Thus, reconstruction recovers the original update projected into the retained spaces. It equals $\Delta W_i^{\mathrm{eff}}$ only when those spaces contain every required update direction; components removed by the threshold cannot be recovered. This separates the numerical approximation in common-coordinate construction from the exact residual decomposition within given coordinates in Appendix C.

E Experimental Objects, Evaluation, and Configurations

This section specifies the capability libraries, metrics, configurations, and training controls underlying the result tables. Module sources, inference settings, and scoring are matched within each comparison.

E.1 Benchmark Scores, Aggregates, and Inference Settings

We evaluate on 11 benchmarks and average their scores as Full-11. Because OpenEQA and RoboVQA use the fixed `gpt-4o-2024-11-20` judge, we also report Mean-9, which excludes them and summarizes the nine rule-scored benchmarks. BLINK-4 first averages Relative Depth, Spatial Relation, Object Localization, and Multi-view Reasoning, then enters the aggregates as one benchmark.

The evaluated item counts are VSIBench 5130, ERQA 400, EmbSpatial 3640, Cosmos 510, CVBench 2638, BLINK-4 522, RefSpatial 277, VABench-Point 300, Where2Place 100, OpenEQA 1636, and RoboVQA 1921. BLINK-4's 522 items come from the four spatial subtasks of the full 1901-item BLINK evaluation; RefSpatial uses 277 items on both InternVL3.5-14B and Qwen3-VL-8B. Subsequent tables abbreviate the corresponding names as VSI, EmbS, Cos, CVB, BLK, RefS, VAB, W2P, OEQA, and RoboV, with ERQA unchanged.

Under this scoring protocol, headline comparisons within a backbone fix module sources and inference settings so that differences do not arise from these conditions. InternVL3.5-14B uses frame size 448, 32 video segments, bf16, and FlashAttention-2, with temperature 0.2 for ERQA and

RefSpatial. RefSpatial is a strict-format coordinate probe in this setting: answers must be integer points normalized to 0–1000 in `[[x,y]]` format. Its scores should therefore be interpreted with both the task and output-format requirements in view.

E.2 Library Entries Defined by Training Origins

Each library entry is the update stored by its own training run, but the relationship between that update and the final model depends on the training starting point. The two InternVL3.5-14B libraries differ only in how video module NL4 was trained, allowing us to distinguish same-origin composition from continued-training increment entry.

In the same-origin library, all four modules are trained independently from base checkpoint W_0. Each stored ΔW_i is therefore the complete change from the base, and $W_0 + \Delta W_i$ is its training endpoint. This is the primary InternVL3.5-14B block in Table 1. In the extended library, NL1–NL3 still start from W_0, but NL4 starts a second training stage from the NL1-adapted checkpoint $W_0 + \Delta W_{\mathrm{NL1}}$. Its stored rank-128 increment $\Delta W^{\mathrm{inc}}_{\mathrm{NL4}}$ records only the second-stage change, whereas the endpoint contains both stages: $W_0 + \Delta W_{\mathrm{NL1}} + \Delta W^{\mathrm{inc}}_{\mathrm{NL4}}$. The extended-library NL4 single-module row evaluates this complete endpoint.

The object consumed by composition must be distinguished from that evaluated endpoint. The extended library enters the stored increment itself as NL4 and supplies the same $\Delta W^{\mathrm{inc}}_{\mathrm{NL4}}$ to every merge method. With NL1 as reference, ReCAP therefore forms $T_{\mathrm{NL4}} = M^{\mathrm{inc}}_{\mathrm{NL4}} - M_{\mathrm{NL1}}$ and uses $M^{\mathrm{inc}}_{\mathrm{NL4}}$ in support construction. This residual measures a difference between entries, not the full NL4 model's change from W_0. That distinction is necessary for interpreting the full-change entry control in Appendix H.1; no method receives NL4-specific information in the comparison.

Qwen3-VL-8B has no continued-training entry: all three modules are trained from the same W_0. Its stored updates and endpoint models consequently follow the same-origin relationship above.

E.3 Capability Definitions and Training Supervision

Training origins specify where a module starts; its supervision specifies which capability it is intended to learn. Table 9 connects semantic types with training supervision and starting checkpoints. Types are assigned from training objectives before composition rather than inferred from evaluation scores, so subsequent experiments test reuse of capabilities defined in advance.

InternVL3.5-14B NL1–NL3 are shared by its two libraries, while the two video-semantic entries correspond to same-origin and continued-training NL4. Qwen3-VL-8B modules are listed separately in the same table to preserve the mapping between capability and training origin.

E.4 Evaluation Scores and Composition Configurations

Tables 1, 13, and 14 report the listed method configurations on the full evaluation set. Scores are aggregated within each benchmark and then averaged equally across benchmarks; Full-11 includes all 11, whereas Mean-9 excludes OpenEQA and RoboVQA.

The InternVL3.5-14B same-origin library uses NL1 as the reference and donor coefficients $[1.05, 0.15, 0.98]$; the extended library uses $[1.05, 0.30, 0.98]$. Task reconfiguration and module replacement are compared within their respective evaluation samples (Tables 15, 16, and 17). Qwen3-VL-8B module replacement keeps the composition configuration fixed.

E.5 Adopted InternVL3.5-14B Composition Configuration

Table 10 specifies the same-origin composition configuration. Continuous merge coefficients act only on donors, while the reference update is retained directly. Values at the reference position are metadata for symmetric ablations and do not enter ReCAP's reference–residual composition. The extended library retains the other settings and uses donor coefficients $[1.05, 0.30, 0.98]$ for NL2, NL3, and NL4.

Table 9: Supervision and training origins for both backbones. InternVL3.5-14B NL1–NL3 are common to both library regimes. The same-origin video module starts from the base checkpoint W_0; the extended video module starts from the NL1-adapted checkpoint.

Module	Semantic type	Training supervision	Training origin
InternVL3.5-14B			
NL1	Grounded interaction	Grounded QA, point answering, and place/action grounding	Base checkpoint W_0
NL2	Spatial reasoning	Spatial reasoning and geometric layout	Base checkpoint W_0
NL3	Temporal, action-sequence, and task-planning reasoning	Temporal and action-sequence QA and planning supervision	Base checkpoint W_0
NL4, same-origin	Video-semantic next-action reasoning	Video next-action and subtask supervision	Base checkpoint W_0
NL4, extended	Video-semantic next-action reasoning	Video next-action and subtask supervision	NL1-adapted checkpoint $W_0 + \Delta W_{\mathrm{NL1}}$
Qwen3-VL-8B, same-origin library			
NL1	Spatial and temporal reasoning	Spatial-QA mixture	Base checkpoint W_0
NL2	Spatial reasoning	Point answering and spatial hard-case mixtures	Base checkpoint W_0
NL4	Video-semantic next-action reasoning	Cosmos video data	Base checkpoint W_0

Table 10: Composition configuration for the InternVL3.5-14B same-origin library. The extended library uses donor coefficients $[1.05, 0.30, 0.98]$; all other entries are unchanged.

Component	Parameter	Value
Effective update	`lora_scaling_in_core`	2.0
Numerical union coordinates	—	All participating LoRA A/B factors
Subspace-input normalization	`normalize_span`	`true`
Joint residual subspace	`max_outer_iter`	5
Residual-subspace dimension	$k_L = k_R$	128
Z-space merge method	`method`	TIES
Z-space density	`density`	1.0 (no trim)
Sign method	`sign_method`	`total`
Capture-rate voting	`vote_beta`	0.5
Capture-rate floor	`vote_eta_min`	0.2
Residual mode	`residual_mode`	`both`
Cross residual merge	—	TIES
Cross residual gate	$\lambda_\times$	0.25
Orthogonal residual merge	—	weighted-mean
Orthogonal residual gate	$\lambda_\perp$	0.5
Reference	—	NL1
Residual set	—	{NL2, NL3, NL4}
Donor coefficients	$[m_{\mathrm{NL2}}, m_{\mathrm{NL3}}, m_{\mathrm{NL4}}]$	$[1.05, 0.15, 0.98]$ (active sum 2.18)
Symmetric-ablation-only entry	γ_{NL1}	0.52 (unused by the proposed method)
Consensus-map top-k	`baseline_topk`	10
Support density	`support_density`	0.10
Off-support gate	η_{off}	0.10
Export mode	`EXPORT_LORA`	`False` (direct apply)

E.6 Adopted Qwen3-VL-8B Composition Configuration

Qwen3-VL-8B retains the same subspace-learning and aggregation settings, but its different module library requires a separate reference and donor specification. Table 11 gives the configuration evaluated in Tables 1 and 14: video-semantic next-action module NL4 is the reference, and the off-support gate is 0.08. InternVL3.5-14B instead uses reference NL1 and gate 0.10. The table lists the Qwen3-VL-8B donors and coefficients; other shared settings follow Table 10.

Table 11: Composition hyperparameters for the main Qwen3-VL-8B setting.

Component	Parameter	Value
Effective update	`lora_scaling_in_core`	2.0
Numerical union coordinates	—	All participating LoRA A/B factors
Joint residual subspace	`max_outer_iter`	5
Residual-subspace dimension	$k_L = k_R$	128
Z-space merge method	`method`	TIES
Z-space density	`density`	1.0 (no trim)
Sign method	`sign_method`	`total`
Capture-rate voting	`vote_beta`	0.5
Capture-rate floor	`vote_eta_min`	0.2
Residual mode	`residual_mode`	`both`
Cross residual gate	$\lambda_\times$	0.25
Orthogonal residual gate	$\lambda_\perp$	0.5
Reference	—	NL4
Residual set	—	{NL1, NL2}
Donor coefficients	$[m_{\text{NL1}}, m_{\text{NL2}}]$	$[0.52, 1.14]$ (active sum 1.66)
Consensus-map top-k	`baseline_topk`	10
Support density	`support_density`	0.10
Off-support gate	η_{off}	0.08
Residual injection	`residual_injection_mode`	`masked`
Export mode	`EXPORT_LORA`	`False` (direct apply)

E.7 Rank-Matched Training Controls and Cost Accounting

Training-free comparisons fix the module library. The joint-training control additionally tests independent training followed by composition against one training run on pooled capability data. To match the adaptation-rank budget, each backbone's joint control uses the union of capability mixtures and a LoRA rank equal to the library's summed rank. Table 12 records the controls used in Table 1.

Table 12: Rank-matched monolithic controls used in the strict-protocol comparison.

Backbone	Joint LoRA rank	Capability-module capacity	Full-11
InternVL3.5-14B	512	4×128	64.08
Qwen3-VL-8B	384	3×128	66.55

The joint control pools and reshuffles all capability-training data and uses a LoRA rank equal to the sum of component ranks. Its total training tokens equal the sum used by the capability modules. Every module and joint control uses its final training checkpoint. Library construction requires training, whereas recomposition of existing modules requires no additional training.

F Per-Benchmark Composition Results

This section reports full-set per-benchmark scores for the methods in Table 1.

F.1 Per-Benchmark Method Results

Tables 13 and 14 report full-set per-benchmark scores for InternVL3.5-14B and Qwen3-VL-8B, retaining single-module, per-benchmark oracle, and joint-training controls. Appendix E.1 defines the benchmarks and evaluation counts.

Table 13: Full-set per-benchmark results for the two InternVL3.5-14B libraries. Shared NL1–NL3 and rank-matched controls appear once. The extended NL4 single row evaluates the training endpoint stacking NL1 and its continued-training increment. Bold marks the highest reported training-free result within each library.

Model	VSI	ERQA	EmbS	Cos	CVB	BLK	RefS	VAB	W2P	OEQA	RoboV	Full-11
Same-origin library												
ReCAP	**67.52**	**46.50**	79.51	**60.00**	87.68	66.71	56.32	**71.67**	**73.00**	**57.78**	65.15	**66.53**
Vanilla TIES	66.59	43.50	**80.08**	59.41	**88.17**	66.53	54.87	**71.67**	71.00	57.04	65.64	65.86
Task Arithmetic	61.12	45.00	77.61	55.88	88.02	**70.01**	**58.48**	68.67	72.00	57.73	61.67	65.11
Core-Space TIES	64.72	43.00	79.67	**60.00**	88.10	65.35	52.71	70.33	**73.00**	57.58	**66.03**	65.50
KnOTS-TIES	65.93	44.00	79.73	59.41	87.60	65.63	53.07	70.00	72.00	56.75	65.53	65.42
NL4 single (same-origin module)	54.73	44.00	77.36	61.37	87.15	69.23	55.23	69.33	73.00	56.98	62.94	64.67
Oracle (per-benchmark max of singles)	68.88	44.75	79.42	61.37	87.38	70.09	55.60	70.67	73.00	57.75	62.94	66.53
Extended library												
ReCAP	67.59	**46.75**	79.95	**60.00**	88.25	67.43	56.68	**71.00**	70.00	**58.56**	65.97	**66.56**
Vanilla TIES	**67.71**	43.25	80.25	59.41	87.91	64.21	52.71	70.33	65.00	57.90	**66.01**	64.97
Task Arithmetic	60.94	44.50	77.45	54.90	88.02	**69.94**	**59.21**	68.00	**72.00**	57.90	62.01	64.99
Core-Space TIES	66.68	45.00	**80.58**	59.80	**88.29**	64.69	54.51	69.00	**72.00**	58.25	65.20	65.82
KnOTS-TIES	67.40	42.75	80.36	59.80	87.83	65.54	53.79	68.33	71.00	56.68	65.37	65.35
NL4 single (extended module)	58.17	44.50	79.09	62.55	87.57	68.67	55.60	69.33	71.00	58.18	64.11	65.34
Oracle (per-benchmark max of singles)	68.88	44.75	79.42	62.55	87.57	70.09	55.60	70.67	72.00	58.18	64.11	66.71
Shared by both regimes												
NL1 single	60.03	44.75	77.80	53.53	87.38	68.98	55.60	70.67	66.00	57.75	62.40	64.08
NL2 single	68.88	42.00	79.42	51.57	87.30	61.96	50.90	69.67	70.00	54.45	60.19	63.30
NL3 single	56.26	41.25	77.91	52.94	87.00	70.09	53.07	69.33	72.00	57.67	60.37	63.44
Rank-matched joint training	68.44	40.75	79.53	60.39	87.87	59.89	53.79	66.00	70.00	55.67	62.55	64.08

Table 14: Full-set per-benchmark results for Qwen3-VL-8B. Bold marks the highest reported training-free result in each column, including ties.

Model	VSI	ERQA	EmbS	Cos	CVB	BLK	RefS	VAB	W2P	OEQA	RoboV	Full-11
ReCAP	**61.88**	45.25	82.45	**66.08**	87.41	72.83	**57.76**	67.33	**75.00**	61.90	73.09	**68.27**
NL1 single (best)	60.43	44.50	82.75	61.57	87.38	70.99	58.84	67.67	72.00	61.42	74.04	67.42
NL2 single	64.51	44.25	81.13	57.06	87.45	70.27	54.51	64.00	72.00	61.35	70.95	66.14
NL4 single	57.24	42.50	83.46	66.86	87.23	73.24	54.87	70.00	70.00	61.77	73.43	67.33
Oracle (per-benchmark max of singles)	64.51	44.50	83.46	66.86	87.45	73.24	58.84	70.00	72.00	61.77	74.04	68.79
Vanilla TIES	59.94	43.25	80.82	64.51	86.16	70.24	49.82	67.67	67.00	59.02	71.81	65.48
Task Arithmetic	60.74	43.50	81.70	64.51	**87.64**	**76.20**	45.85	53.33	69.00	**64.56**	68.81	65.08
Core-Space TIES	61.60	43.50	**82.99**	65.10	86.66	71.69	55.23	**68.00**	73.00	60.71	72.38	67.35
KnOTS-TIES	61.23	**46.00**	82.97	**66.08**	87.04	71.95	56.68	67.00	71.00	61.02	**73.88**	67.71
Rank-matched joint training	63.49	41.25	84.20	62.75	88.17	71.28	54.87	68.67	66.00	59.41	72.01	66.55
Base model	59.93	42.00	79.29	61.57	78.47	66.66	19.31	44.00	63.00	66.23	62.91	58.49

G Task Reconfiguration with a Fixed Capability Library

The preceding section compares methods at fixed configurations. This section reports task results for different module subsets and references within one library. Configurations retain existing module weights and donor coefficients and differ in the participating modules and reference.

Spatial reasoning averages VSIBench and EmbSpatial; localization averages RefSpatial, VABench-Point, and Where2Place; video understanding uses Cosmos. The tables report target scores, non-target scores, and Mean-9 for task configurations, the default four-module composition, and the corresponding strongest single module.

Table 15 first reports descriptive full-set results for the same-origin library. Reconfiguration improves its spatial and video targets, but default four-module localization at 66.99 still exceeds the best three-module subset at 66.87. Configuration selection therefore need not favor a smaller subset; it can also support retaining the full composition.

Table 16 reports the extended library on that configuration study's evaluation samples. The library and sample scope differ from the same-origin table, so gains are computed against the default composition

Table 15: Target-oriented deployments from the primary library on the full evaluation set. Target and non-target scores average the selected group and the remaining Mean-9 benchmarks, respectively.

Target	Deployment	Target	Non-target	Mean-9
Primary same-origin library: full evaluation set				
Spatial reasoning	**Highest: {NL2,NL3,NL4}, ref NL4**	**74.58**	62.54	65.22
	All-capability default	73.51	**65.98**	**67.66**
	Strongest single module (NL2)	74.15	61.91	64.63
Localization	Best 3-module subset: {NL1,NL2,NL4}, ref NL1	66.87	67.91	67.56
	All-capability default	**66.99**	**67.99**	**67.66**
	Strongest single module (NL4)	65.86	65.64	65.71
Video understanding	**Highest: {NL2,NL3,NL4}, ref NL2**	**61.57**	67.45	66.79
	All-capability default	60.00	**68.61**	**67.66**
	Strongest single module (NL4)	61.37	66.26	65.71

or single modules in this table. Together, the tables expand the task gains and non-target changes in Table 2.

Table 16: Task results for the continued-training extended library on that configuration study's evaluation samples. Gains are relative to the default composition in this table; its sample scope differs from the same-origin table, so absolute scores are not compared across tables.

Target	Deployment	Target	Non-target	Mean-9
Spatial reasoning	**Highest: {NL2,NL3,NL4}, ref NL4**	**74.18**	62.13	64.81
	All-capability default	72.74	**65.14**	**66.83**
	Strongest single module (NL2)	73.32	62.04	64.55
Localization	**Highest: {NL1,NL2,NL3}, ref NL2**	**68.54**	65.48	66.50
	All-capability default	66.65	**66.92**	**66.83**
	Strongest single module (NL4)	66.60	65.55	65.90
Video understanding	**Highest: {NL2,NL3,NL4}, ref NL2**	**60.58**	67.49	66.72
	All-capability default	58.51	**67.87**	**66.83**
	Strongest single module (NL4)	**60.58**	66.56	65.90

H Controls for Capability-Library Extension and Updates

Task reconfiguration keeps the modules unchanged, whereas library evolution also changes entry representations or training outcomes. Following Section 4.4, we examine three changes in sequence: entering a continued-training increment, replacing one module while reusing coefficients, and calibrating after all modules are retrained. Seed replications and wrong-slot controls then delimit what the coefficient-handling results establish.

H.1 Entering a Continued-Training Increment or Its Full Change

Appendix E.2 enters the stored continued-training NL4 increment into the library. To test that convention, Table 3A instead enters the member's full change from the base, $\Delta W_{\mathrm{NL1}} + \Delta W_{\mathrm{NL4}}^{\mathrm{inc}}$. Flattening requires no training: concatenating LoRA factors exactly constructs a rank-≤ 256 entry, represented losslessly by the full union coordinate basis. Apart from the entry representation, the main run's merge pipeline, configuration, code state, and evaluation protocol remain fixed.

Under these conditions, full-change entry lowers Full-11 from 66.56 to 65.15, a 1.41-point drop, and Mean-9 from 67.52 to 65.99. The largest per-benchmark drops are RefSpatial (-5.78), BLINK-4 (-2.53), and ERQA (-2.25). Direct entry of the stored increment therefore performs better end to end under this frozen configuration.

The intervention determines how far that result can be interpreted. With NL1 as reference, the full-change entry's residual reduces to the pure second-stage increment, whereas the stored-increment

entry is directly differenced against the reference. Changing the entry also changes the consensus-support inputs. The experiment consequently compares two complete entry conventions without separately attributing the gap to residual construction or support masking.

H.2 Coefficient Reuse and Calibration after Single-Module Replacement

The entry control fixes the trained updates. We next change one module to test whether existing composition coefficients remain useful. NL3 and NL4 are replaced independently, leaving the other three modules and untouched donors' coefficients fixed. Their donor-weight shares differ: $0.30/2.33 = 12.9\%$ for NL3 and $0.98/2.33 = 42.1\%$ for NL4. The replacements thus cover smaller and larger changes to the active donor mass.

NL3 replacement uses mean RoboVQA/OpenEQA as its target score and includes two training seeds; NL4 replacement uses Cosmos. Table 17 reports target scores for direct coefficient reuse and norm calibration on the module-replacement study's evaluation samples.

Table 17: Direct coefficient reuse versus donor norm calibration after module replacement. Single-slot rows report target scores on that replacement study's evaluation samples; the all-slot row reports Full-11 (Table 4), so levels are not directly comparable across the rule. Parentheses show changes from direct reuse; donor shares exclude the reference.

Replaced	Share of active donor mass	r_i^{norm}	Direct reuse	Calibrated
NL3, seed 1	12.9%	1.52	61.59	61.43 (-0.16)
NL3, seed 2	12.9%	1.50	61.50	61.31 (-0.20)
NL4	42.1%	0.79	55.19	53.53 (-1.66)
All four slots	100%	0.21–1.81	62.23	64.25 ($+2.02$)

The three single-slot rows favor direct coefficient reuse over norm calibration, whereas the all-slot row favors calibration. Rescaling therefore does not automatically help after a module replacement. The all-slot row reports Full-11 rather than the preceding target scores and is not directly comparable in score level, but motivates examining coefficient handling separately for a broader refresh.

H.3 Norm Calibration for Full-Library Refresh

Retraining the entire library changes the reference and several donors together, potentially shifting update scales away from the original composition configuration. Equation (11) adjusts donor coefficients by their new/old effective-update norm ratios. To specify this scale, fix the reference assignment, let $\mathcal{A}$ contain adapted layers shared by old and new modules, and define the full-update norm as

$$n_i = \|\Delta\Theta_i^{\mathrm{eff}}\|_F = \left(\sum_{\ell\in\mathcal{A}} \|\Delta W_{i,\ell}^{\mathrm{eff}}\|_F^2\right)^{1/2}. \tag{53}$$

For positive old and new norms, $r_i^{\mathrm{norm}} > 1$ reduces the coefficient and a ratio below one increases it, matching each donor's coefficient–full-update norm product. The rule adjusts magnitude rather than matching update directions or reference-relative residuals, and the reference remains unscaled. Whether it helps must therefore be assessed on the new library.

The reference's new/old norm ratio is 1.81, and donor ratios are 1.51, 0.21, and 0.24, with original coefficients $(1.05, 0.30, 0.98)$. Table 4 evaluates the calibrated configuration $(0.70, 1.44, 4.13)$. Substituting later norm records into the formula yields approximately $(0.6933, 1.4397, 4.1053)$. These slightly different computed values are distinguished from, and do not replace, the evaluated configuration.

Within the same refreshed library, calibration raises Full-11 from direct reuse's 62.23 to 64.25, the 2.02-point gain in Table 17. This improves composition of the new modules without restoring the old library's level: its pre-refresh 66.56 comes from a different module set and remains higher.

H.4 Seed Replication and Donor-Correspondence Diagnostics

The opposite single-module and full-library outcomes motivate checking sensitivity to randomness and calibration assignment. First, changing only the NL3 training seed moves the composed target mean by 0.09; its two norm ratios agree within 1.5%, and direct reuse leads both replacements. The single-slot observation is thus consistent across these two seeds, although the run count does not establish statistical significance.

Second, assigning calibration factors to incorrect donor slots loses 6.20 points when all four modules are refreshed, showing that correct factor–donor correspondence matters in that setting. In a single-slot update, the intervention only perturbs one untouched donor coefficient and does not hurt either tested slot. The full-library diagnostic therefore cannot be directly generalized to a need for single-slot calibration.

Finally, uniformly rescaling all donor coefficients cannot substitute for wrong-slot assignment. ReCAP normalizes the coefficients to sum to one, so multiplication by a common positive constant leaves the consumed weights unchanged in both update settings. The consequential calibration change is therefore in relative donor coefficients, not their common scale.

I Controls for Composition Components and Aggregation

An advantage for the full pipeline does not by itself establish each design's contribution. We therefore expand Section 4.5 at three levels: component removal checks the composition structure, auxiliary-term removal measures the smaller outside-subspace contribution, and fixed-pipeline comparisons isolate within-subspace aggregation. Because the last comparison operates on coordinates, we also test how basis rotations change its output.

I.1 Scope and Outcomes of Component Removal

Table 5A reports Full-11 changes under the listed interventions. The reference-related control jointly replaces reference-relative construction and selected donor weighting with uniformly weighted symmetric merging. Together with the common-coordinate and subspace-removal controls, this comparison evaluates the combined utility of the composition design. The larger gains from reference-relative weighted composition and the residual subspace in the extended library support their role in integrating continued-training increments.

Removing common coordinates, reference–residual construction, or the joint residual subspace lowers performance in all tested settings. The latter two interventions incur larger losses in the InternVL3.5-14B extended library than in the same-origin library, connecting them particularly to this continued-training increment composition. Support-guided injection is tested only in the listed Qwen3-VL-8B control: removing it reduces Full-11 by 0.33 points and Cosmos by 5.7. This links the aggregate change to a task-specific loss, without treating untested table entries as evidence of no effect.

I.2 Supplementary Gains from Outside-Subspace Components

The preceding controls concern the main composition structure. The outside-subspace terms in Appendix C instead supplement changes omitted by finite-dimensional projection. Removing their low-weight contribution changes Full-11 by -0.15 on InternVL3.5-14B and -0.05 on Qwen3-VL-8B. Retaining them therefore provides a positive but small gain in the tested settings, supporting their description as auxiliary residuals injected with reduced gates.

I.3 Aggregation Rules with the Remaining Pipeline Fixed

Component removal changes relatively broad parts of the computation. Table 5B narrows the intervention by fixing the learned subspace, donor coefficients, support mask, and all other operations, replacing only within-subspace coordinate aggregation. The reference configuration retains every coordinate, so trimming is not a varying factor. The branches instead assess hard sign gating, magnitude weighting, and the elected direction.

Weighted averaging without hard gating reaches 66.40 and 66.19 on the InternVL3.5-14B same-origin and extended libraries, still exceeding the other training-free baselines in Table 1. The retained design thus already provides a composition advantage, on top of which ReCAP's sign rule adds 0.13 and 0.37 points. Unweighted majority signs and reversed election perform worse; extended-library reversal losses concentrate in VSIBench (-7.85) and Cosmos (-5.49), connecting the elected direction to these task outcomes. These conclusions concern the tested four-module, untrimmed setting.

Beyond task gains, learning the same subspace must be distinguished from choosing the same coordinate basis. A subspace admits basis rotations, but coordinate-wise sign decisions depend on those coordinates. Haar-random rotations of the discovered bases change the merged shared term by a mean relative Frobenius distance of 0.57, whereas weighted averaging remains strictly invariant under the same test. Together with the scores above, this locates the sign-rule gains in the learned basis used for evaluation, rather than establishing a basis-independent property of the subspace alone.

J Per-Benchmark Comparison with Public Embodied Models

The preceding experiments compare composition methods within specified libraries. Section 4.6 additionally compares the resulting models with nine public embodied models. Table 6 summarizes that comparison by Full-11 and Mean-9; Table 18 below expands the same models into per-benchmark scores to distinguish aggregate performance from task-specific strengths.

Model inputs are constructed using the template types specified in each model's open-source release and checked for compatibility with multimodal inputs. Each benchmark follows the official task definitions and evaluation metrics in its open-source release; the tasks and scoring rules used in this study are described in Appendix E.1.

Public comparators and the Qwen3-VL-8B composition use the same task-prompt content and the following input and generation settings: at most 64 sampled frames per video, images upsampled to at least 1.04 megapixels, bf16, and FlashAttention-2. The generation temperature is 0 except for ERQA, where it is 0.2. The maximum generation length is set to at least 2,048 tokens for each benchmark; any higher existing limit is retained. InternVL3.5-14B retains its main-experiment settings specified in Appendix E.1.

Before scoring, thinking content is removed from model outputs, and the final model answers are extracted using each benchmark's answer-extraction rules. For OpenEQA and RoboVQA, the extracted final answers are submitted to the fixed `gpt-4o-2024-11-20` judge. The table reports model performance under this protocol; cross-backbone differences cannot be attributed entirely to the composition method.

Table 18: Comparison with nine public embodied models, ordered by Full-11. Both ReCAP variants match their respective main results. Bold marks the highest score in each column. Public comparators use Qwen3-VL-8B evaluation settings; InternVL3.5-14B retains its main-experiment settings.

Model	VSI	ERQA	EmbS	Cos	CVB	BLK	RefS	VAB	W2P	OEQA	RoboV	Mean-9	Full-11
ReCAP (Qwen3-VL-8B)	61.88	45.25	82.45	**66.08**	87.41	**72.83**	**57.76**	67.33	75.00	61.90	**73.09**	**68.44**	**68.27**
ReCAP (InternVL3.5-14B)	67.52	**46.50**	79.51	60.00	87.68	66.71	56.32	**71.67**	73.00	57.78	65.15	67.66	66.53
Embodied-R1.5	59.82	43.00	76.84	65.29	87.45	57.61	50.90	70.00	**77.00**	60.68	70.60	65.32	65.38
RynnBrain-8B	**67.90**	42.25	**83.02**	56.08	**88.82**	59.80	55.60	7.00	71.00	59.69	66.11	59.05	59.75
MiMo-Embodied-7B	45.44	43.50	79.67	53.73	88.67	60.55	11.91	25.67	29.00	**64.14**	54.45	48.68	50.61
RoboBrain 2.0-7B	47.90	42.00	73.57	35.88	85.71	61.68	10.11	14.33	34.00	56.14	64.82	45.02	47.83
Pelican-VL-7B	53.21	41.25	72.03	62.35	81.12	66.26	11.19	9.00	0.00	57.82	70.34	44.05	47.69
EmbodiedBrain-7B	34.44	43.00	73.98	57.45	80.29	65.35	7.67	9.67	23.00	52.15	66.40	43.87	46.67
Embodied-R1-3B	28.91	40.75	68.79	52.75	79.38	62.84	5.05	26.33	25.00	50.15	64.06	43.31	45.82
Cosmos-Reason1-7B	34.89	39.25	68.41	59.02	66.76	61.39	6.96	13.33	29.00	54.87	62.82	42.11	45.15
SpaceR-7B	46.42	40.50	68.93	55.88	75.82	66.53	11.19	4.00	0.00	43.77	65.12	41.03	43.47

Abbreviations: VSI, VSIBench; EmbS, EmbSpatial; Cos, Cosmos; CVB, CVBench; BLK, BLINK-4; RefS, RefSpatial; VAB, VABench-Point; W2P, Where2Place; OEQA, OpenEQA; RoboV, RoboVQA.

The Qwen3-VL-8B ReCAP version leads this comparison on Cosmos, BLINK-4, RefSpatial, and RoboVQA, while the InternVL3.5-14B version leads on ERQA and VABench-Point. Other public models still attain the highest scores on several benchmarks. The aggregate lead in the main text

therefore reflects an advantage across the combined tasks, without implying that either ReCAP version leads every task.