MENTOR-ISP: STAGE-WISE CREDIT MENTORING FOR SINGLE-AGENT ISP TUNING

Anonymous authors
Paper under double-blind review

ABSTRACT

An image signal processor (ISP) is a sequential pipeline in which upstream decisions affect downstream modules. Single-agent reinforcement learning efficiently optimizes ISP parameters jointly, but typically relies on a global reward that provides coarse credit across coupled controls. Serialized multi-agent approaches provide finer stage-wise supervision, but require repeated ISP execution at deployment. We present Mentor-ISP, a modular single-agent RL framework that decouples structured credit assignment and inter-stage dependency modeling from serialized ISP execution. Mentor-ISP factorizes the joint action into pipeline-ordered groups, uses sequential action handoffs to condition downstream decisions on upstream predictions, and applies stage-wise credit during training. At inference, all decisions are assembled into a single joint action, requiring only one ISP rendering per optimization step. Across two real-camera image-quality datasets, Mentor-ISP substantially improves peak signal-to-noise ratio (PSNR) over a matched single-agent baseline while approaching serialized multi-agent performance at near-single-agent inference cost. Ablations show that stage-wise credit provides the primary gain, with action handoff offering complementary benefit.

1 INTRODUCTION

Image signal processors (ISPs) transform RAW sensor measurements into visually meaningful images through a cascade of processing stages, including white balance, tone adjustment, color processing, denoising, and detail enhancement (Liang et al., 2021; Schwartz et al., 2019). The parameters of these stages are strongly coupled: modifying an upstream control can alter both the input distribution and the desirable configuration of downstream stages. In practice, ISP tuning therefore remains a highly iterative process that relies heavily on experienced imaging engineers, who repeatedly adjust parameters, render images, inspect the resulting quality, and refine subsequent controls. Such a workflow is labor-intensive, time-consuming, and difficult to scale across sensors, imaging conditions, and target applications (Nishimura et al., 2018; Mosleh et al., 2020; Sun et al., 2024). Automatic ISP optimization has accordingly been explored through black-box parameter search, differentiable ISP proxies, direct parameter prediction, and reinforcement-learning-based tuning (Nishimura et al., 2018; Tseng et al., 2019; Mosleh et al., 2020; Qin et al., 2022; 2023; Sun et al., 2024).

Among these approaches, reinforcement learning (RL) is particularly attractive for image-adaptive ISP tuning because it naturally supports iterative, feedback-driven optimization. Rather than predicting a fixed parameter configuration in a single pass, an RL controller iteratively updates ISP parameters from rendering feedback and progressively refines subsequent decisions (Wang et al., 2020; Shin et al., 2022; Sun et al., 2024). This naturally casts ISP tuning as a sequential decision-making problem.

A natural solution is single-agent reinforcement learning (SARL), in which a single controller jointly updates the full ISP parameter vector over multiple optimization steps (Sun et al., 2024; 2026). This formulation has an important efficiency advantage. All parameter groups are predicted within a single policy evaluation, and the resulting joint update requires only one ISP execution per optimization step. However, the complete action is typically evaluated through a global image-level reward. When several strongly coupled ISP controls change simultaneously, such a reward provides

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.

limited information about which functional stage caused an observed improvement or degradation. SARL therefore provides efficient environment interaction, but its learning signal offers only coarse credit assignment across ISP modules.

Sequential multi-agent reinforcement learning (MARL) provides a complementary solution. MARL-ISP (Sun et al., 2026) assigns different agents to functional ISP modules and organizes their decisions according to the physical order of the imaging pipeline. The output produced after an upstream decision becomes part of the information available to subsequent agents, allowing downstream controls to adapt explicitly to preceding ISP decisions. Furthermore, the quality change observed as each module is introduced provides a more direct signal for identifying the contribution of individual functional stages. By exposing intermediate effects along the ISP chain, serialized MARL therefore models both inter-module dependency and stage-wise credit more explicitly than a monolithic joint controller.

This finer-grained structure, however, comes at the cost of finer-grained environment interaction. To obtain the observation or quality change produced by an individual module, its parameter update must be applied and the ISP must be rendered before the subsequent stage is evaluated. A serialized controller can therefore require multiple ISP executions within a single optimization step, with the interaction cost increasing with the number of functional stages. Prior work partially mitigates this training overhead by combining serialized and parallel prediction during exploration (Sun et al., 2026), but the basic trade-off remains: SARL performs joint environment interaction but provides coarse module-level credit, whereas serialized MARL couples explicit inter-module dependency modeling and stage-wise supervision to repeated ISP execution during both learning and deployment.

This trade-off motivates the central question of this work: *does exploiting the sequential structure of an ISP require serialized ISP execution at deployment?* More specifically, can the dependencies among ISP modules be represented inside a single policy, while sequential ISP execution is used only where it is most useful—for assigning credit during learning? Our key observation is that the structure of decision making, the structure of credit assignment, and the structure of environment interaction need not be identical. The ISP sequence can be encoded within the policy itself, while intermediate ISP evaluations can be restricted to training-time supervision.

Based on this observation, we introduce **Mentor-ISP**, a modular single-agent reinforcement-learning controller that combines *sequential action handoff* with *pipeline-ordered stage-wise credit assignment*. Mentor-ISP factorizes the joint ISP action into functional groups following the ISP pipeline. Each policy head predicts the controls owned by its functional stage, while downstream heads receive a bounded representation of preceding action predictions. This allows inter-stage dependencies to be represented within a single policy without requiring intermediate ISP rendering between policy heads.

During training, Mentor-ISP progressively evaluates the same candidate joint update in pipeline order. The incremental quality change introduced by each stage provides a structured learning signal for the corresponding action group. At inference, these intermediate evaluations are removed: all stage decisions are assembled before environment interaction and applied as a single joint ISP update. Mentor-ISP therefore separates sequential policy computation and training-time credit assignment from serialized environment interaction at deployment. We evaluate Mentor-ISP on two paired real-camera datasets and on task-driven ISP optimization for object detection and instance segmentation. Under matched ISP and optimization protocols, Mentor-ISP substantially improves PSNR over SARL while approaching serialized MARL, with inference cost close to the single-agent baseline. Ablations further show that stage-wise credit provides the primary optimization gain and action handoff offers complementary benefit.

2 RELATED WORK

2.1 AUTOMATIC ISP OPTIMIZATION

Automatic ISP optimization has been explored through several complementary directions, including learned ISP proxies, direct parameter prediction, and iterative parameter optimization. Proxy-based approaches replace or approximate parts of the conventional imaging pipeline with differen-

tiable or learned mappings, enabling image quality to be optimized through gradient-based learning (Tseng et al., 2019). Parameter-prediction methods instead retain explicit ISP controls and learn to infer their values directly from image features, with recent approaches further exploiting dependencies or sequence-specific priors among parameter groups (Qin et al., 2022; 2023). Black-box and hardware-in-the-loop methods directly search the parameter space of an explicit ISP according to image-quality or downstream-task objectives (Nishimura et al., 2018; Mosleh et al., 2020), but typically require repeated ISP evaluations for each input. Our work likewise retains an explicit, parameterized ISP, but focuses on learning a feedback-driven controller that progressively adapts its parameters from observed image quality rather than replacing the ISP or performing per-image search from scratch. This formulation naturally connects ISP parameter optimization with reinforcement learning, which we discuss next.

2.2 Reinforcement Learning for ISP Control

Reinforcement learning has been used to formulate image-processing control as a sequential decision-making problem, including exposure selection and adaptive image-processing operations (Wang et al., 2020; Shin et al., 2022). More recent approaches directly optimize ISP configurations through iterative interaction with the imaging environment. RL-SeqISP (Sun et al., 2024) progressively refines ISP parameters from rendering feedback, and its subsequent SARL-ISP formulation (Sun et al., 2026) jointly updates parameters across multiple ISP modules using a single agent. While this enables efficient joint ISP interaction, the resulting global reward provides only coarse credit across coupled functional stages.

MARL-ISP (Sun et al., 2026) instead decomposes ISP control across pipeline-ordered agents, allowing downstream decisions to use intermediate observations produced by preceding module decisions. It further introduces module-specific rewards based on intermediate quality changes, providing finer-grained credit assignment. However, obtaining these intermediate observations and rewards requires serialized ISP execution as decisions propagate through the pipeline, including at deployment.

2.3 Structured Policy and Credit Assignment for Modular Control

Structured action representations provide a complementary way to exploit internal structure within a high-dimensional action space. Action-branching architectures use a shared representation followed by separate branches for individual action dimensions, reducing the complexity of multidimensional control while retaining shared context (Tavakoli et al., 2018). Factorized policies similarly represent a joint action through dimension-wise policy components and have been shown to work effectively with on-policy algorithms such as Proximal Policy Optimization (PPO) (Tang & Agrawal, 2020). Beyond independent factorization, sequential action models predict action components one at a time, allowing later components to depend on preceding decisions (Metz et al., 2017). These approaches demonstrate that structure within a joint action can be represented explicitly inside the policy or value model rather than treating the complete action as an indivisible output.

Credit assignment presents a related but distinct challenge when multiple actions or decision makers jointly affect a shared outcome. In cooperative multi-agent reinforcement learning, a global return alone may provide limited information about the contribution of an individual agent, motivating methods that explicitly decompose or estimate agent-specific utility. Value-decomposition approaches such as Value-Decomposition Networks (VDN) and QMIX factorize a team value function into agent-level components (Sunehag et al., 2018; Rashid et al., 2018), while counterfactual methods such as Counterfactual Multi-Agent (COMA) estimate an agent's contribution by comparing its action against alternative actions under a shared team objective (Foerster et al., 2018). These methods establish structured credit assignment as an important mechanism for learning coordinated policies without requiring the environment to provide an explicit reward for every internal decision.

Mentor-ISP builds on these structured-policy and credit-assignment perspectives while exploiting the known physical ordering of ISP stages. Unlike serialized ISP controllers, it models inter-stage dependencies within a shared policy and uses intermediate ISP evaluations only for training-time credit assignment.

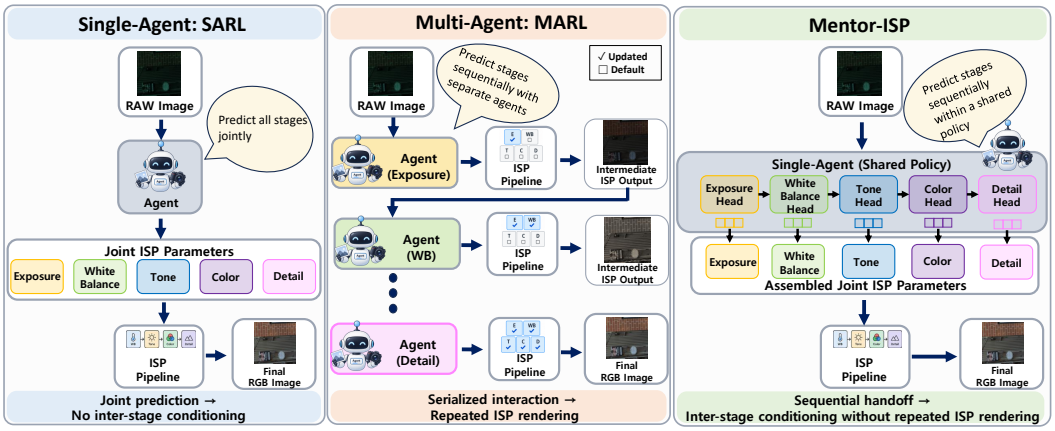

Figure 1: High-level comparison of SARL, serialized MARL, and Mentor-ISP. Mentor-ISP represents inter-stage dependencies within a shared policy and assembles all stage decisions into one joint ISP action, avoiding intermediate ISP execution at inference.

3 METHODS

3.1 OVERVIEW

As illustrated in Fig. 1, Mentor-ISP retains a single actor–critic framework while separating three aspects that are jointly serialized in MARL: action prediction, credit assignment, and environment interaction. The actor represents inter-stage dependencies through sequential action handoffs, while pipeline-ordered stage-wise credit provides module-specific learning signals during training. All stage decisions are assembled into a single joint action before environment interaction. Consequently, intermediate ISP evaluations are used only during training, whereas inference requires one ISP execution per optimization step. The overall Mentor-ISP architecture is shown in Fig. 2.

3.2 RL-BASED ISP PARAMETER TUNING

Let x denote a RAW input image and $F_{\mathrm{ISP}}(x;\mathbf{p})$ an ISP parameterized by the normalized parameter vector $\mathbf{p} \in [-1,1]^D$. At optimization step t, the current ISP output $I_t = F_{\mathrm{ISP}}(x;\mathbf{p}_t)$ is encoded by a feature extractor $\phi(\cdot)$ and provided to an actor–critic controller.

Controller architecture. The controller uses a frozen ResNet-50 feature extractor followed by an actor–critic network. In SARL, a single actor jointly predicts all active ISP controls, while a single critic estimates the global state value. Mentor-ISP retains the same feature extractor and critic architecture, but factorizes the actor into a shared trunk and five stage-specific policy heads. These heads constitute a single shared policy rather than independent agents.

The actor predicts a stochastic update for the active ISP controls and is optimized using Proximal Policy Optimization (PPO) (Schulman et al., 2017). The sampled continuous actions are bounded by a $\tanh$ transformation before being applied to the ISP.

During image-quality training, we adopt the Dynamic Parameter Optimization Module (DPOM) of Sun et al. (2026) for parameter transition,

$$\mathbf{p}_{t+1} = \mathcal{T}_{\mathrm{DPOM}}\left(\mathbf{p}_t, \mathbf{a}_t\right). \tag{1}$$

When DPOM is disabled, we directly apply $\mathbf{p}_{t+1} = \mathrm{clip}(\mathbf{p}_t + \mathbf{a}_t, -1, 1)$. Whenever DPOM is used, the same transition rule is applied to all compared RL controllers; DPOM itself is not a contribution of this work.

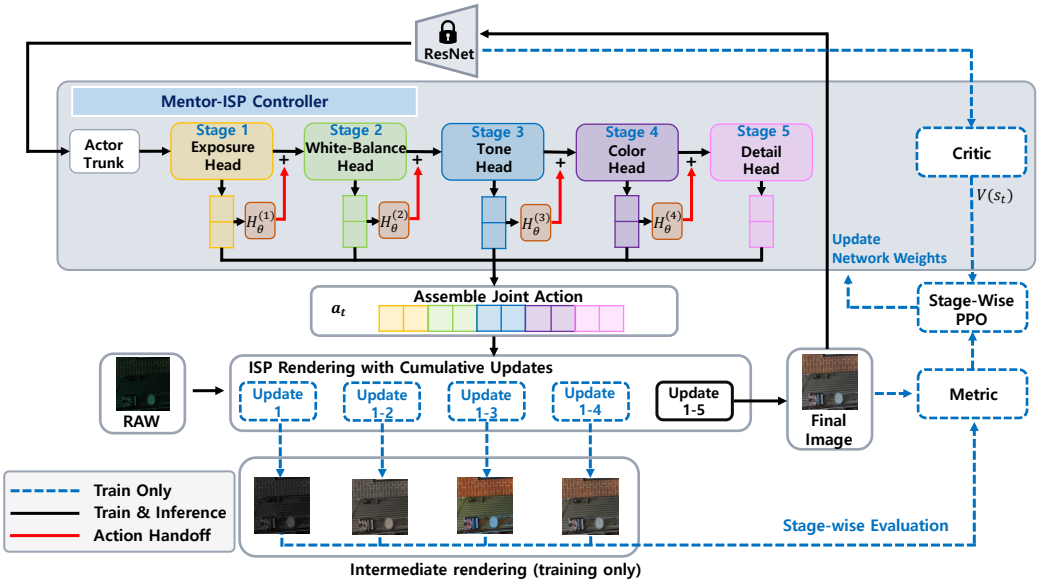

Figure 2: **Overall architecture of Mentor-ISP.** Stage-specific policy heads sequentially exchange action information and produce a single joint ISP action. During training, cumulative partial updates are additionally rendered in pipeline order to obtain stage-wise credits. These intermediate renderings are removed at inference, where the assembled joint action requires only one ISP execution per optimization step.

3.3 Sequential Action Handoff

The active ISP controls are divided into M disjoint functional groups ordered according to the ISP pipeline. Unlike a conventional single-agent controller that predicts all controls from the same latent representation, Mentor-ISP connects the stage-specific policy heads sequentially.

A shared actor trunk first produces the latent context $\mathbf{h}_t^{(1)}$. At stage m, the corresponding head predicts the mean $\boldsymbol{\mu}_t^{(m)}$ of the stochastic action distribution for the controls owned by that stage. A bounded representation of this prediction is then transferred to the next stage:

$$\begin{aligned} \boldsymbol{\mu}_t^{(m)} &= g_\theta^{(m)}\left(\mathbf{h}_t^{(m)}\right), \\ \mathbf{d}_t^{(m)} &= \eta \tanh\left(H_\theta^{(m)}\left(\tanh\left(\boldsymbol{\mu}_t^{(m)}\right)\right)\right), \\ \mathbf{h}_t^{(m+1)} &= \mathbf{h}_t^{(m)} + \mathbf{d}_t^{(m)}. \end{aligned} \tag{2}$$

Here, $H_\theta^{(m)}$ is a lightweight projection and η bounds the handoff magnitude. The handoff is computed from the predicted action mean rather than a sampled action, providing deterministic inter-stage context without propagating sampling noise.

After all stages produce their controls, the stage-wise actions are assembled into a single joint action $\mathbf{a}_t$ and applied once using the active parameter-transition rule. Thus, downstream decisions explicitly depend on preceding action predictions without requiring intermediate ISP rendering between policy heads.

Baseline-preserving initialization. For a controlled comparison, Mentor-ISP is initialized to reproduce the same initial policy distribution as SARL. The shared actor trunk, critic, action statistics, and corresponding output rows use matched initialization, while the handoff projections are initialized to zero. The two models therefore start from equivalent policy distributions before PPO training. No pretrained SARL policy is used; the two controllers are trained independently from matched initializations. This allows the ablations in Sec. 4.5 to isolate the effects of action handoff and stage-wise credit assignment.

3.4 STAGE-WISE CREDIT AND POLICY OPTIMIZATION

A conventional single-agent controller evaluates the complete joint update using a global reward,

$$R_t = \mathcal{M}(I_{t+1}) - \mathcal{M}(I_t), \tag{3}$$

where $\mathcal{M}(\cdot)$ denotes the task objective. Although this reward evaluates the final ISP output, it does not identify the contribution of individual functional stages.

During training, Mentor-ISP progressively reveals the same candidate joint parameter update in ISP pipeline order. After computing $\mathbf{p}_{t+1}$ using the active parameter-transition rule, the partial parameter vector for stage m is

$$\mathbf{p}_{t+1}^{(m)} = \mathbf{p}_t + \mathbf{b}^{(\leq m)} \odot (\mathbf{p}_{t+1} - \mathbf{p}_t), \tag{4}$$

where $\mathbf{b}^{(\leq m)}$ activates the dimensions belonging to stages $1, \ldots, m$. The corresponding image is $I_{t+1}^{(m)} = F_{\text{ISP}}(x; \mathbf{p}_{t+1}^{(m)})$, with $I_{t+1}^{(0)} = I_t$.

The credit assigned to stage m is the incremental quality change obtained when that stage is introduced after all preceding stages:

$$c_t^{(m)} = \mathcal{M}\left(I_{t+1}^{(m)}\right) - \mathcal{M}\left(I_{t+1}^{(m-1)}\right). \tag{5}$$

Because the final revealed state equals the complete joint update, the unmixed stage credits telescope to the global reward, $\sum_m c_t^{(m)} = R_t$. Thus, the credits decompose the same joint transition rather than defining separate stage-specific objectives. This decomposition is intentionally pipeline-ordered. When interactions exist among ISP stages, the individual credits are therefore not intended to provide order-invariant or causal attribution to each module. Instead, they serve as structured learning signals aligned with the physical ISP order: each credit measures the incremental effect of introducing one stage after its upstream stages have already been applied. The telescoping property ensures that these signals remain consistent with the observed global transition.

Following Sun et al. (2026), we augment each stage-wise credit with the global reward:

$$\tilde{c}_t^{(m)} = c_t^{(m)} + \omega R_t, \tag{6}$$

where ω controls the contribution of the shared global objective. The local term provides targeted feedback for the action dimensions directly controlled by each stage, whereas the global term retains sensitivity to cross-stage effects that cannot be uniquely attributed to one module. This is particularly important because an upstream prediction can also influence downstream decisions through the action-handoff pathway. We therefore use $\tilde{c}_t^{(m)}$ as a structured policy-learning signal rather than interpreting $c_t^{(m)}$ as an isolated causal contribution.

For stage m, the PPO probability ratio $r_t^{(m)}(\theta)$ is computed using only the action dimensions owned by that stage. The mixed stage credits are temporally accumulated across the outer ISP optimization steps as

$$\hat{G}_t^{(m)} = \sum_{l=0}^{T-t-1} (\gamma\lambda)^l \tilde{c}_{t+l}^{(m)}, \tag{7}$$

where γ is the discount factor and λ controls the temporal decay. Unlike standard generalized advantage estimation (GAE), $\hat{G}_t^{(m)}$ does not use a stage-specific value baseline; it directly accumulates the stage-wise credits.

The actor objective is

$$\mathcal{L}_{\text{actor}} = -\sum_{m=1}^{M} \mathbb{E}_t \left[\min\left(r_t^{(m)}(\theta)\hat{G}_t^{(m)}, \text{clip}\left(r_t^{(m)}(\theta), 1-\epsilon, 1+\epsilon\right)\hat{G}_t^{(m)} \right) \right]. \tag{8}$$

Each stage therefore receives its own policy-learning signal, while gradients from all stages jointly update the same shared actor.

The critic remains global and is trained from returns derived from the complete reward R_t using standard GAE (Schulman et al., 2016). No stage-specific critics or independent agents are introduced, and the remaining PPO optimization follows the standard formulation. The intermediate ISP evaluations used to construct stage-wise credits are required only during training; at inference, all stage decisions are assembled before a single ISP execution.

Table 1: ISP optimization results on the S24 Dataset and S7 Dataset in terms of structural similarity index (SSIM) and peak signal-to-noise ratio (PSNR). Results are reported as mean ± standard deviation over three seeds. Best and second-best results are shown in **bold** and underlined, respectively.

	S24 Dataset		S7 Dataset	
Method	SSIM ↑	PSNR ↑	SSIM ↑	PSNR ↑
SARL	0.706 ± 0.024	20.012 ± 0.594	0.497 ± 0.057	20.532 ± 0.694
Mentor-ISP	$\underline{0.741 \pm 0.017}$	$\underline{23.679 \pm 0.329}$	$\mathbf{0.631 \pm 0.032}$	$\underline{22.187 \pm 0.267}$
MARL	$\mathbf{0.780 \pm 0.054}$	$\mathbf{23.835 \pm 0.284}$	$\underline{0.593 \pm 0.057}$	$\mathbf{22.297 \pm 0.366}$

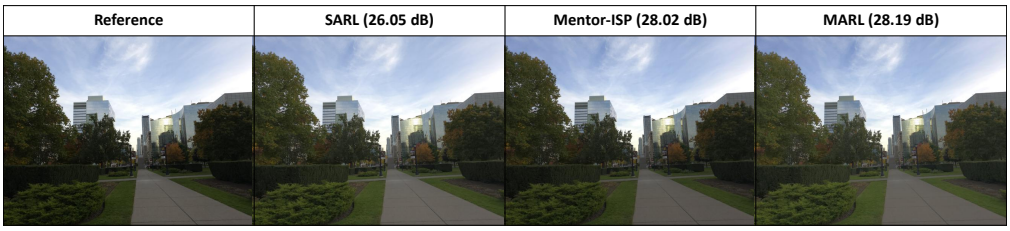

Figure 3: Qualitative comparison on S24 dataset. PSNR values are shown for SARL, Mentor-ISP, and serialized MARL.

4 EXPERIMENTS

4.1 EXPERIMENTAL SETUP

We evaluate Mentor-ISP on reference-based image-quality optimization and downstream detection and segmentation objectives. We evaluate ISP optimization on two paired real-camera datasets, S24 (Afifi et al., 2025) and S7 (Schwartz et al., 2019). The S24 dataset contains 2,619 training pairs and 205 validation pairs, of which 64 are used for checkpoint selection and 141 for final evaluation. S7 contains 110 same-exposure pairs, split into 90/10/10 pairs for training, checkpoint selection, and final evaluation. Fixed splits are shared across all methods, and results are averaged over three seeds (42, 43, and 44).

For image-quality optimization, DPOM is enabled during training for all methods and disabled during inference, ensuring reference-free deployment. Structural similarity index (SSIM) and peak signal-to-noise ratio (PSNR) are computed at the native resolution without resizing or cropping. For the task objective $\mathcal{M}$ in equation 3, we use SSIM for image quality, $-\log(1 + \mathcal{L}_{\text{det}})$ for detection (classification, box-regression, and RPN losses; mask loss excluded), and an image-level Mask AP proxy for segmentation over IoU thresholds 0.50–0.95.

4.2 IMAGE-QUALITY OPTIMIZATION

Table 1 compares SARL, Mentor-ISP, and serialized MARL on S24 and S7. On S24, Mentor-ISP improves PSNR from 20.012 dB to 23.679 dB over SARL and approaches MARL at 23.835 dB, while MARL achieves the highest structural-similarity scores. On S7, Mentor-ISP improves both SSIM and PSNR over SARL and exceeds MARL in SSIM (0.631 vs. 0.593), while remaining within 0.110 dB in PSNR. A representative qualitative comparison on S24 is shown in Fig. 3, exhibiting a trend consistent with the quantitative results.

4.3 DOWNSTREAM VISION TASKS

To evaluate whether the proposed formulation extends beyond reference-based image-quality optimization, we conduct object detection and instance segmentation experiments on Microsoft Common Objects in Context (COCO) 2017 (Lin et al., 2014). Following the unprocessing principle of Brooks et al. (2019), we generate 16-bit three-channel RAW-like inputs using an inverse transfor-

Table 2: Object detection and instance segmentation results. Values are mean ± standard deviation over three seeds. AP_{50} and AP_{75} use IoU thresholds of 0.50 and 0.75; best and second-best results are shown in bold and underlined.

	Object Detection			Instance Segmentation		
Method	AP ↑	AP_{50} ↑	AP_{75} ↑	Mask AP ↑	Mask AP_{50} ↑	Mask AP_{75} ↑
SARL	46.15 ± 0.24	66.41 ± 0.26	$\underline{50.47 \pm 0.24}$	45.91 ± 0.12	67.88 ± 0.03	50.80 ± 0.11
Mentor-ISP	$\mathbf{46.34 \pm 0.17}$	$\mathbf{66.72 \pm 0.18}$	$\mathbf{50.58 \pm 0.18}$	$\underline{45.96 \pm 0.16}$	$\underline{67.94 \pm 0.13}$	$\underline{50.85 \pm 0.13}$
MARL	$\underline{46.22 \pm 0.29}$	$\underline{66.47 \pm 0.31}$	50.46 ± 0.37	$\mathbf{46.02 \pm 0.15}$	$\mathbf{67.96 \pm 0.03}$	$\mathbf{51.05 \pm 0.22}$

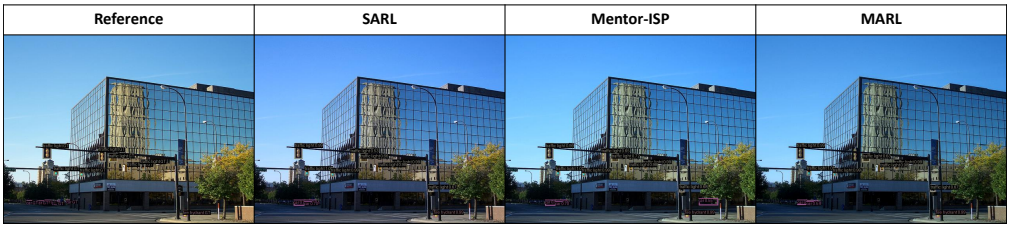

Figure 4: Qualitative comparison for object detection on the COCO-based downstream evaluation.

Table 3: Model size and inference efficiency under reference-free inference (DPOM off). Model size assumes 32-bit floating-point (FP32) parameter storage and excludes runtime memory. Latency is reported as mean ± standard deviation.

Method	Total Params (M)	Trainable Params (M)	Model Size (MB)	S24 Latency (ms) ↓	S7 Latency (ms) ↓
SARL	24.694	1.186	98.78	208.80 ± 0.29	193.70 ± 0.90
Mentor-ISP	24.698	1.190	98.79	214.58 ± 1.40	199.29 ± 1.47
MARL	34.665	11.157	138.66	905.31 ± 1.33	828.17 ± 2.60

mation tailored to our SimpleISP pipeline with a fixed S24 color-correction matrix. No learned reverse-RAW network is used. Details are provided in Appendix A.3.

All controllers optimize the same SimpleISP pipeline using PPO (Schulman et al., 2017), with a frozen COCO-pretrained Mask R-CNN (He et al., 2017) providing the downstream objective. Detection and segmentation policies use 1,000 and 200 training images, respectively, with 200 validation images for checkpoint selection and 4,773 for final evaluation.

DPOM is disabled during training and inference. Detection and segmentation performance is reported as mean ± standard deviation over three seeds.

As shown in Table 2, Mentor-ISP yields small but consistent numerical improvements over SARL while performing comparably to serialized MARL. Given the small differences among the controllers, we interpret these results as evidence that the proposed formulation remains applicable to task-driven ISP objectives rather than as a substantial downstream-task performance gain. A representative object-detection example is shown in Fig. 4.

4.4 Inference efficiency

We measure inference latency on a single NVIDIA L40S GPU in FP32 with batch size one, four control steps, and a 384×384 policy-state resolution. SARL and Mentor-ISP perform one full-resolution ISP rendering per outer step, compared with five for serialized MARL. Training and inference ISP execution counts are detailed in Appendix A.4.

With DPOM disabled, latency is averaged over 30 synchronized measurements after 10 warm-up runs. Measurements include state resizing, ResNet-50 feature extraction (He et al., 2016), actor

Table 4: Ablation study of module-wise PPO, sequential action handoff, and stage-wise credit assignment on the S24 and S7 datasets.

Dataset	Variant	Module-wise PPO	Action Handoff	Stage-wise Credit	PSNR ↑	SSIM ↑
S24	SARL	–	–	–	20.012	0.706
	A: Factorized SARL	✓	–	–	20.871	0.708
	B: A + Action Handoff	✓	✓	–	21.270	0.728
	C: A + Stage-wise Credit	✓	–	✓	<u>22.412</u>	<u>0.722</u>
	Mentor-ISP	✓	✓	✓	**23.679**	**0.741**
S7	SARL	–	–	–	20.532	0.497
	A: Factorized SARL	✓	–	–	21.271	0.533
	B: A + Action Handoff	✓	✓	–	21.772	0.563
	C: A + Stage-wise Credit	✓	–	✓	<u>22.049</u>	<u>0.615</u>
	Mentor-ISP	✓	✓	✓	**22.187**	**0.631**

Table 5: Sensitivity to functional ISP stage ordering. Results are reported as mean ± standard deviation over three seeds. Best and second-best results are shown in bold and underlined, respectively.

	S24 Dataset		S7 Dataset	
Order	PSNR ↑	SSIM ↑	PSNR ↑	SSIM ↑
Physical	$\mathbf{23.679} \pm 0.329$	$\mathbf{0.741} \pm 0.017$	$\mathbf{22.187} \pm 0.267$	$\mathbf{0.631} \pm 0.032$
Reverse	22.348 ± 0.573	0.725 ± 0.013	$\underline{21.952} \pm 0.300$	0.612 ± 0.023
Shuffle A	22.834 ± 0.580	$\underline{0.738} \pm 0.013$	21.939 ± 0.225	0.617 ± 0.030
Shuffle B	22.846 ± 0.542	0.734 ± 0.016	21.865 ± 0.168	0.613 ± 0.033
Shuffle C	$\underline{23.077} \pm 0.665$	$\underline{0.738} \pm 0.016$	21.785 ± 0.291	$\underline{0.623} \pm 0.022$

inference, parameter updates, and full-resolution ISP rendering, but exclude model loading, I/O, input transfer, and metric computation.

As shown in Table 3, Mentor-ISP remains nearly identical in model size to SARL, whereas MARL requires substantially more trainable parameters. Mentor-ISP incurs only 2.77%/2.89% latency overhead over SARL on S24/S7 while being 4.22×/4.16× faster than serialized MARL. Thus, inter-stage conditioning introduces little deployment overhead while avoiding MARL's repeated ISP rendering.

4.5 Ablation Study

Table 4 analyzes Mentor-ISP from the SARL baseline. Factorizing SARL into module-specific heads with module-wise PPO improves PSNR on both datasets. Stage-wise credit provides the larger individual PSNR gain, while action handoff offers a complementary benefit; their combination achieves the best performance among the ablated variants.

We also compare the physical ISP order with its reverse and three fixed shuffles, with the exact stage orderings defined in Appendix A.5. As shown in Table 5, the physical order achieves the highest mean PSNR and SSIM on both datasets, providing empirical support for its use in sequential handoff and stage-wise credit.

5 Conclusion

We presented Mentor-ISP, a modular single-agent RL framework that decouples structured credit assignment and inter-stage dependency modeling from serialized ISP execution. Pipeline-ordered stage-wise credit provides module-specific learning signals during training, while sequential action handoffs condition downstream decisions on upstream predictions within a shared policy. At inference, all stage decisions are assembled into a single joint action, avoiding intermediate ISP execution. Across two real-camera image-quality datasets, Mentor-ISP substantially improves over matched SARL and approaches serialized MARL at near-SARL inference cost. Ablations show that stage-wise credit provides the primary optimization gain, with action handoff offering complementary benefit. These results demonstrate that sequential ISP structure can be exploited without serialized deployment.

AI USE STATEMENT

We used generative AI tools to polish author-written text for grammar, clarity, and readability, assist with the retrieval and discovery of related work, and help draft portions of the manuscript. We did not use these tools for research ideation or execution, generating synthetic datasets, or proving mathematical claims. All AI-assisted outputs were reviewed by the authors to ensure that they preserved the intended meaning and accurately reflected our methods and results. We take full responsibility for the final content of this work, including all AI-assisted material.

REPRODUCIBILITY STATEMENT

We provide the methodological and experimental details necessary to reproduce the reported results in the main paper and Appendix. These include the model architecture, training procedure, reward formulation, dataset preprocessing and splits, and evaluation protocols. The settings for the image-quality and downstream vision experiments, including the relevant hyperparameters, are also documented. Main experiments are conducted using fixed data splits and three random seeds.

REFERENCES

Mahmoud Afifi, Luxi Zhao, Abhijith Punnappurath, Mohamed Ashraf Abdelsalam, Ran Zhang, and Michael S. Brown. Time-aware auto white balance in mobile photography. In *The IEEE International Conference on Computer Vision (ICCV)*, pp. 5038–5047, 2025.

Tim Brooks, Ben Mildenhall, Tianfan Xue, Jiawen Chen, Dillon Sharlet, and Jonathan T. Barron. Unprocessing images for learned RAW denoising. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 11036–11045, 2019.

Jakob Foerster, Gregory Farquhar, Triantafyllos Afouras, Nantas Nardelli, and Shimon Whiteson. Counterfactual multi-agent policy gradients. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. doi: 10.1609/aaai.v32i1.11794.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016. doi: 10.1109/CVPR.2016.90.

Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2961–2969, 2017.

Zhetong Liang, Jianrui Cai, Zisheng Cao, and Lei Zhang. CameraNet: A two-stage framework for effective camera ISP learning. *IEEE Transactions on Image Processing*, 30:2248–2262, 2021.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: Common objects in context. In *European Conference on Computer Vision (ECCV)*, pp. 740–755. Springer, 2014.

Luke Metz, Julian Ibarz, Navdeep Jaitly, and James Davidson. Discrete sequential prediction of continuous actions for deep RL. *arXiv preprint arXiv:1705.05035*, 2017.

Ali Mosleh, Avinash Sharma, Emmanuel Onzon, Fahim Mannan, Nicolas Robidoux, and Felix Heide. Hardware-in-the-loop end-to-end optimization of camera image processing pipelines. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7526–7535, 2020. doi: 10.1109/CVPR42600.2020.00755.

Jun Nishimura, Timo Gerasimow, Rao Sushma, Aleksandar Sutic, Chyuan-Tyng Wu, and Gilad Michael. Automatic ISP image quality tuning using nonlinear optimization. In *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 2471–2475, 2018. doi: 10.1109/ICIP.2018.8451818.

Haina Qin, Longfei Han, Juan Wang, Congxuan Zhang, Yanwei Li, Bing Li, and Weiming Hu. Attention-aware learning for hyperparameter prediction in image processing pipelines. In *Computer Vision – ECCV 2022*, volume 13679 of *Lecture Notes in Computer Science*, pp. 271–287. Springer, 2022. doi: 10.1007/978-3-031-19800-7_16.

Haina Qin, Longfei Han, Weihua Xiong, Juan Wang, Wentao Ma, Bing Li, and Weiming Hu. Learning to exploit the sequence-specific prior knowledge for image processing pipelines optimization. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 22314–22323, 2023. doi: 10.1109/CVPR52729.2023.02137.

Tabish Rashid, Mikayel Samvelyan, Christian Schroeder, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. QMIX: Monotonic value function factorisation for deep multi-agent reinforcement learning. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pp. 4295–4304. PMLR, 2018.

John Schulman, Philipp Moritz, Sergey Levine, Michael I. Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. In *International Conference on Learning Representations*, 2016.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL `https://arxiv.org/abs/1707.06347`.

Eli Schwartz, Raja Giryes, and Alex M. Bronstein. DeepISP: Toward learning an end-to-end image processing pipeline. *IEEE Transactions on Image Processing*, 28(2):912–923, 2019.

Ukcheol Shin, Kyunghyun Lee, and In So Kweon. DRL-ISP: Multi-objective camera ISP with deep reinforcement learning. In *2022 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7044–7051, 2022. doi: 10.1109/IROS47612.2022.9981361.

Xinyu Sun, Zhikun Zhao, Lili Wei, Congyan Lang, Mingxuan Cai, Longfei Han, Juan Wang, Bing Li, and Yuxuan Guo. RL-SeqISP: Reinforcement learning-based sequential optimization for image signal processing. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 5025–5033, 2024. doi: 10.1609/aaai.v38i5.28307.

Xinyu Sun, Bing Li, Congyan Lang, Zhikun Zhao, Juan Wang, Weihua Xiong, Weiming Hu, and Long Cheng. Reinforcement learning-based sequential parameter tuning for image signal processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 48(4):4114–4131, 2026. doi: 10.1109/TPAMI.2025.3642837.

Peter Sunehag, Guy Lever, Audrunas Gruslys, Wojciech Marian Czarnecki, Vinicius Zambaldi, Max Jaderberg, Marc Lanctot, Nicolas Sonnerat, Joel Z. Leibo, Karl Tuyls, and Thore Graepel. Value-decomposition networks for cooperative multi-agent learning based on team reward. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, pp. 2085–2087, 2018.

Yunhao Tang and Shipra Agrawal. Discretizing continuous action space for on-policy optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 5981–5988, 2020. doi: 10.1609/aaai.v34i04.6059.

Arash Tavakoli, Fabio Pardo, and Petar Kormushev. Action branching architectures for deep reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 4131–4138, 2018. doi: 10.1609/aaai.v32i1.11798.

Ethan Tseng, Felix Yu, Yuting Yang, Fahim Mannan, Karl Arnaud, Derek Nowrouzezahrai, Jean-François Lalonde, and Felix Heide. Hyperparameter optimization in black-box image processing using differentiable proxies. *ACM Transactions on Graphics*, 38:1–14, 07 2019. doi: 10.1145/3306346.3322996.

Zhouxia Wang, Jiawei Zhang, Mude Lin, Jiong Wang, Ping Luo, and Jimmy Ren. Learning a reinforced agent for flexible exposure bracketing selection. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1817–1825, 2020. doi: 10.1109/CVPR42600.2020.00189.

A Appendix

A.1 Training and Optimization Settings

Table 6 summarizes the training hyperparameters used for image-quality optimization and downstream vision tasks. Unless otherwise stated, the same optimization settings are used for SARL, Mentor-ISP, and MARL to ensure a controlled comparison. For Mentor-ISP, the global reward weight ω controls the contribution of the shared global objective to the stage-wise credit, while the action-handoff bound η limits inter-stage communication.

For Mentor-ISP, stage-specific policy signals are computed by recursively accumulating the mixed stage credits with discount $\gamma\lambda$, as described in Sec. 3.4. The global critic is trained using standard GAE with the same γ and λ. The handoff bound $\eta = 0.1$ limits the magnitude of the latent update passed between consecutive policy heads. All methods use a control horizon of four outer ISP optimization steps.

A.2 ISP Parameter Grouping

The SimpleISP interface represents ISP parameters using a 26-dimensional parameter vector. Among these, 17 controls are actively optimized by all compared RL methods. Mentor-ISP partitions the active controls into five disjoint functional groups following the ISP processing order. The same parameter ownership and stage ordering are used for the serialized MARL baseline, while SARL predicts the same 17 controls jointly as a single action. This ensures that differences among the methods arise from the policy and credit-assignment structures rather than from different controllable ISP parameters.

Table 7 summarizes the functional grouping and number of active controls assigned to each ISP stage.

A.3 Dataset and Preprocessing Details

S24 and S7 image-quality datasets. For image-quality optimization, we use two paired real-camera datasets. The S24 dataset contains 2,619 training pairs and 205 validation pairs of RAW-like linear camera RGB inputs and rendered sRGB targets. From the 205 validation pairs, a fixed subset of 64 images is used for checkpoint selection, while the remaining 141 images are reserved for final evaluation. The same split is used for all methods and random seeds.

The S7 dataset contains 110 same-exposure pairs consisting of medium-exposure DNG inputs and camera-rendered JPEG targets. We use 90 pairs for training, 10 for checkpoint selection, and 10 for final evaluation. Again, the same split is shared across all compared methods and random seeds. For both datasets, results are averaged over three random seeds (42, 43, and 44). During image-quality training, DPOM is enabled for all methods, whereas it is disabled during inference to enable reference-free deployment.

COCO reverse-RAW generation. For the downstream object-detection and instance-segmentation experiments, we generate synthetic RAW-like inputs from COCO 2017 RGB images. Following the unprocessing principle of Brooks et al. (2019), we apply an inverse transformation tailored to our SimpleISP pipeline using a fixed S24 color-correction matrix. We do not use a learned reverse-RAW network.

Reverse-RAW generation uses deterministic per-image rejection sampling. For each image, the random seed is derived from a base seed of 42 and the image file path, producing a reproducible sequence of inverse-ISP parameter candidates. At most 24 candidates are evaluated, and the first candidate satisfying all of the following round-trip reconstruction criteria is retained:

$$\mathrm{PSNR} \geq 38\ \mathrm{dB}, \qquad \mathrm{SSIM} \geq 0.95, \tag{9}$$

with no more than 0.1% of reconstructed RAW values falling outside the normalized $[0, 1]$ range. The resulting inputs are stored as 16-bit three-channel linear camera RGB images rather than Bayer RAW images. The original COCO image resolution and annotations are preserved.

Table 6: Training hyperparameters used for image-quality optimization and downstream vision tasks.

Hyperparameter	Image Quality (S24/S7)	Detection / Segmentation
Learning rate	5×10^{-5}	5×10^{-5}
Learning-rate schedule	Cosine decay (final ratio 0.3)	Constant
Discount factor γ	0.95	0.95
Temporal weighting parameter λ	0.95	0.95
PPO clip ϵ	0.2	0.2
Entropy coefficient	0.001	0.0
Value-loss coefficient	0.5	0.5
PPO epochs	4	4
Global reward weight ω	0.2	0.2
Handoff bound η	0.1	0.1
Action scale	0.25	0.1
Control horizon T	4	4
Batch size	4	1
Training epochs	8	10
Policy/reward resolution	384×384	224×224
Training crop	Random 512×512	None
Optimizer	AdamW	Adam (0.9, 0.999)
Weight decay	10^{-4}	0

Table 7: Functional grouping of the 17 active SimpleISP controls.

Stage	Functional group	No. of parameters
1	Exposure / Black Level	2
2	White Balance	3
3	Tone	5
4	Color	5
5	Detail	2
	Total	**17**

COCO training and evaluation split. For object detection, the RL policies are trained using 1,000 images sampled from COCO train2017. For instance segmentation, 200 train2017 images are used. Model selection and final evaluation are performed on COCO val2017. Of the original 5,000 validation images, 27 do not satisfy the reverse-RAW quality criteria within the maximum 24 candidate trials and are excluded. Among the remaining 4,973 valid images, 200 are used as a development set for checkpoint selection, and the remaining 4,773 images are reserved for final evaluation.

A COCO-pretrained Mask R-CNN with a ResNet-50-FPN backbone is used as the frozen downstream model for both object detection and instance segmentation. All ISP controllers use the same SimpleISP pipeline, a batch size of one, a 224×224 policy-state resolution, four control steps, and an action scale of 0.1. DPOM is disabled during both training and inference for these downstream experiments.

A.4 Training and Inference Execution

Table 8 summarizes the number of ISP executions required during training and inference. All methods first render the initial ISP output once. SARL then performs one ISP execution after each joint action, resulting in $1 + T$ executions.

During training, Mentor-ISP evaluates the progressively revealed joint update after each of its five functional stages to compute stage-wise credits. It therefore requires $1 + 5T$ ISP executions per episode. These intermediate stage evaluations are used only during training. At inference, Mentor-

Table 8: Number of ISP executions over one complete T-step optimization episode. The initial ISP rendering is included.

Method	Training	Inference
SARL	$1 + T = 5$	$1 + T = 5$
Mentor-ISP	$1 + 5T = 21$	$1 + T = 5$
MARL	$1 + 5T = 21$	$1 + 5T = 21$

ISP assembles all five module decisions before environment interaction and therefore requires only $1 + T$ executions, matching the interaction pattern of SARL.

Serialized MARL renders the ISP after each of the five module decisions during both training and inference, resulting in $1 + 5T$ executions.

Following prior work (Sun et al., 2026), which identified $T = 4$ as an effective control horizon for image-quality optimization, we use $T = 4$ throughout the main experiments. We additionally verified this setting on S7 using a single-seed sensitivity analysis over $T \in \{1, 2, 4, 8, 16\}$, where Mentor-ISP achieved its best PSNR and SSIM at $T = 4$.

With $T = 4$, this corresponds to 5 ISP executions for SARL, 21 and 5 for Mentor-ISP during training and inference, respectively, and 21 for MARL.

A.5 Stage-Order Ablation Details

To evaluate whether the physical ISP processing order provides a useful inductive bias, we compare the default stage sequence with its exact reverse and three fixed shuffled permutations. The functional groups are identical across all variants; only their execution order for sequential action handoff and stage-wise credit assignment is changed. The exact stage sequences used in this ablation are summarized in Table 9.

Table 9: Functional-stage orders used in the stage-order sensitivity study.

Order	Stage sequence
Physical	Exposure/Black → WB → Tone → Color → Detail
Reverse	Detail → Color → Tone → WB → Exposure/Black
Shuffle A	Color → WB → Tone → Detail → Exposure/Black
Shuffle B	Tone → WB → Color → Detail → Exposure/Black
Shuffle C	WB → Tone → Color → Exposure/Black → Detail