Learning to Exploit the Sequence-Specific Prior Knowledge for Image Processing Pipelines Optimization (Supplemental Material)

Haina Qin[1,2] Longfei Han[3] Weihua Xiong[1] Juan Wang[1] Wentao Ma[4]
Bing Li[1(✉)] Weiming Hu[1,2]
[1]State Key Laboratory of Multimodal Artificial Intelligence Systems,
Institute of Automation, Chinese Academy of Sciences
[2]School of Artificial Intelligence, University of Chinese Academy of Sciences
[3]Beijing Technology and Business University [4]Zeku Technology, Shanghai
{qinhaina2020@,bli@nlpr.,wmhu@nlpr.}ia.ac.cn longfeihan@btbu.edu.cn Wallace.xiong@gmail.com

1. Additional Details on the ISP used for Experiments

1.1. Synthetic ISP

For better comparison, we follow the most common ISP modules and construct a similar processing pipeline with the ISP used in other methods [3, 4]. A series of processing modules in the synthetic ISP converts the input RAW data into the output RGB image; here, the synthetic ISP consists of *Denoising*, *White Balance*, *DigitalGain*, *Demosaicking*, *Color Space Transform*, *Sharpening*, *Color and Tone Correction* and *Compression* module. It replaces the hardware ISP for object detection and image segmentation experiments in the main document. The proposed method performs these parameters in the modules mentioned above. The details of each processing module in the synthetic ISP are described as follows.

- Denoising: For simple overall BM3D filtering strength adjustment, use the 'filter-strength' parameter. The type of noise is determined by 'noise-type', which has a list of accepted types. Noise variance of the resulting noise is controlled by 'noise-var'. 'lambda-thr3d' is threshold parameter for the hard-thresholding in 3D transform domain. PSDs multiplied with the value of 'mu2'.
- DigitalGain and White Balance: In this block, 'gain-r', 'gain-g' and 'gain-b' are multiplied by the r, g and b pixels on the bayer filter respectively. To adjust the gain of the imaging sensor signal, the image signal is multiplied by 'digital-gain'.
- Color Space Transform: To transform native RGB values of sensors, the original R, G, B is multiplied with Color Correction Matrix (CCM) to obtain R', G', B':

$$\begin{bmatrix} R' \\ G' \\ B' \end{bmatrix} = \begin{bmatrix} c00 & c01 & c02 \\ c10 & c11 & c12 \\ c20 & c21 & c22 \end{bmatrix} \begin{bmatrix} R \\ G \\ B \end{bmatrix} \tag{1}$$

- Sharpening: Amount of sharpening can be controlled via 'usm-amount', a multiplication factor for the sharpened signal. 'use-sigma' is a scalar or sequence of scalars.

The table 1 shows that the operational range of the hyperparameters in synthetic ISP that we predicted in our experiments. For the discrete parameters, we encode them continuously. The prediction result which have minimal distance from the predicted value is chosen from the operational ranges.

Table 1. Hyperparameters of the synthetic ISP and their operational ranges.

(a) Denoising

Parameter	Operational Range
filter-strength	$\{0, 0.1, \ldots, 2.0\}$
noise-type	$\{\text{gw}, \text{g0}, \ldots, \text{g1w}, \ldots\}$
noise-var	$\{0.01, 0.02, \ldots, 1.0\}$
lambda-thr3d	$\{1.0, 2.0, \ldots, 100.0\}$
mu2	$\{0.1, 0.2, \ldots, 100.0\}$

(b) DigitalGain and White Balance

Parameter	Operational Range
gain-r	$\{0.01, 0.02, \ldots, 5.0\}$
gain-g	$\{0.01, 0.02, \ldots, 5.0\}$
gain-b	$\{0.01, 0.02, \ldots, 5.0\}$
digital-gain	$\{0.01, 0.02, \ldots, 2.0\}$

(c) Color Space Transform

Parameter	Operational Range
ccm 00	$\{0.01, 0.02, \ldots, 4.0\}$
ccm 01	$\{-4.0, -3.99, \ldots, 4.0\}$
ccm 02	$\{-4.0, -3.99, \ldots, 4.0\}$
ccm 10	$\{-4.0, -3.99, \ldots, 4.0\}$
ccm 11	$\{0.01, 0.02, \ldots, 4.0\}$
ccm 12	$\{-4.0, -3.99, \ldots, 4.0\}$
ccm 20	$\{-4.0, -3.99, \ldots, 4.0\}$
ccm 21	$\{-4.0, -3.99, \ldots, 4.0\}$
ccm 22	$\{0.01, 0.02, \ldots, 4.0\}$

(d) Sharpening

Parameter	Operational Range
usm-amount	$\{0.1, 0.2, \ldots, 3.0\}$
usm-sigma	$\{0.1, 0.2, \ldots, 10.0\}$

1.2. Spectra 580 ISP

The Qualcomm Spectra 580 ISP is a hardware ISP, used on state-of-the-art consumer devices such as smartphones. Its processing stages also include *Denoising*, *White Balance*, *DigitalGain*, *Demosaicking*, *Color Space Transform*, *Sharpening*, *Color and Tone Correction* and *Compression*. Our approach optimizes 144 of all hyperparameters, optimizing the image quality tasks in the main document. Considering that it is a commercially available ISP, we cannot describe the specific functionality of each processing block. The table 2 shows that the operational range of the hyperparameters in the hardware ISP that we predicted in our experiments. For the discrete parameters, we encode them continuously. The prediction result which have minimal distance from the predicted value is chosen from the operational ranges.

2. Additional Details on Training and Inference Stages

2.1. Dataset Making

Figure 1. Samples of sRGB images and simulated RAW images

The training dataset of the proposed sequential ISP hyperparameter prediction model has two components: input RAW images and ISP hyperparameter labels for downstream tasks. For the image quality task, the input RAW images are collected by IMX766 CMOS sensor, and the hyperparameters labels are labeled by the image experts. For the object detection and image segmentation tasks, we collect the dataset as follows:

Table 2. Hyperparameters of the hardware ISP and their operational ranges.

(a) BNR

Parameter	Operational Range
BNR_0_dns_strength_G	$[0, 63]$
BNR_0_dns_strength_RB	$[0, 63]$
BNR_0_noise_level_gain_G	$[0, 127]$
BNR_0_noise_level_gain_RB	$[0, 127]$
BNR_0_win_LPF_blend_coef	$[0, 32]$
BNR_0_shading_LUT[33]	$[0, 63]$
BNR_0_np_poi_calib[7]	$[0, 9999]$
BNR_0_np_gau_calib[7]	$[0, 9999]$
BNR_0_np_dark_coef_LE	$[0, 100]$
BNR_0_np_light_coef_LE	$[0, 100]$
BNR_0_noise_level_gain_delta	$[0, 63]$
BNR_0_cross_G_strength	$[0, 7]$
BNR_0_dark_pxl_level	$[0, 65535]$
BNR_0_blk_reshape_en	$[0, 1]$
BNR_0_blk_reshape_thd	$[0, 255]$
BNR_0_blk_pxl_blend_coef	$[0, 32]$
BNR_0_region_enhance_en	$[0, 1]$
BNR_0_edge_thd	$[0, 255]$
BNR_0_edge_internal_thd	$[0, 255]$
BNR_0_edge_mad_slope	$[0, 127]$
BNR_0_edge_mad_adj	$[0, 255]$
BNR_0_patch_thd	$[0, 255]$
BNR_0_patch_mad_slope	$[0, 127]$
BNR_0_patch_mad_adj	$[0, 255]$
BNR_0_flat_thd	$[0, 255]$
BNR_0_flat_thd_slope	$[0, 127]$
BNR_0_flat_mad_adj	$[0, 255]$

(b) Optical_Flow

Parameter	Operational Range
OpticalFlow_mv_penalty_h_0_L0	$[0, 1023]$
OpticalFlow_mv_penalty_h_1_L0	$[0, 1023]$
OpticalFlow_mv_penalty_h_2_L0	$[0, 1023]$
OpticalFlow_mv_penalty_h_3_L0	$[0, 1023]$
OpticalFlow_mv_penalty_h_4_L0	$[0, 1023]$
OpticalFlow_mv_penalty_h_5_L0	$[0, 1023]$
OpticalFlow_mv_penalty_h_0_L1	$[0, 1023]$
OpticalFlow_mv_penalty_h_1_L1	$[0, 1023]$
OpticalFlow_mv_penalty_h_2_L1	$[0, 1023]$
OpticalFlow_mv_penalty_h_4_L1	$[0, 1023]$
OpticalFlow_mv_penalty_h_5_L1	$[0, 1023]$
OpticalFlow_mv_luma_penalty_offset	$[0, 255]$
OpticalFlow_mv_penalty_sad_rate_L0	$[0, 63]$
OpticalFlow_mv_penalty_sad_rate_L1	$[0, 63]$
OpticalFlow_mv_penalty_neighbor_rate_L0	$[0, 15]$
OpticalFlow_mv_penalty_neighbor_rate_L1	$[0, 15]$
OpticalFlow_mv_delta_penalty_base_L0	$[0, 1023]$
OpticalFlow_mv_delta_penalty_base_L1	$[0, 1023]$
OpticalFlow_mv_prop_conf_clip_rate_L0	$[0, 255]$
OpticalFlow_mv_prop_conf_clip_rate_L1	$[0, 255]$
OpticalFlow_mv_conf_iir_rate	$[0, 255]$
OpticalFlow_mv_conf_grad_rate	$[0, 63]$
OpticalFlow_conf_prop_base_rate	$[0, 15]$
OpticalFlow_conf_TNR_decay_clip	$[0, 15]$
OpticalFlow_prop_conf_rate	$[0, 65535]$

- The RGB images [2] are converted to RAW $\mathbf{I}$ as input to the ISP by simulating the processing of sRGB images with RAW data [1].
- Using iterative search to batch annotate the label hyperparameters $\mathcal{P}^*$ of the training set using images $\mathbf{I}$ and Ground-truth information on downstream tasks

We invert the ISP to get RAW from the sRGB images in COCO. A way to convert sRGB images back to their original CCD RAW response is described in [1]. Examples of simulated RAW images from the dataset are shown in Figure 1.

In the training dataset, $\mathcal{P}^*$ is the labeled ISP hyperparameters for the image $\mathbf{I}$ and the downstream task. With image $\mathbf{I}$, the downstream task, and the Ground-truth label $\mathbf{b}^{gt}$ on the downstream task, we find the label hyperparameters by iterative search. Specifically, $\mathcal{P}^*$ should meet the following objectives:

$$\mathcal{P}^*_{\text{task}} = \underset{\{\mathcal{P}\}}{argmin}\, \mathcal{L}_{\text{task}}(f_{ISP}(\mathbf{I}_i, \mathcal{P}), \mathbf{b}^{gt}). \tag{2}$$

For each RAW image and downstream task with Ground-truth labels, we chose to use the Artificial Bee Colony (ABC) method to search for the label hyperparameters $\mathcal{P}^*_{task}$. It is a swarm intelligence algorithm with good exploration capability and applicability to numerous practical problems, and it is simple, stable, and does not require complex tuning. We perform parameter labeling on the training dataset of COCO and use the RAW images and label parameters for the model training. The trained model can directly use the RAW image as input for ISP hyperparameters inference without GT information of downstream tasks.

Table 2. Hyperparameters of the hardware ISP and their operational ranges, continued.

(c) MCTF

Parameter	Operational Range
MCTF_tf_exp_base	$[0, 65535]$
MCTF_tf_exp_base_uv	$[0, 65535]$
MCTF_tf_exp_alpha_0	$[0, 1023]$
MCTF_tf_exp_alpha_1	$[0, 1023]$
MCTF_tf_lut_y_input_shift	$[0, 7]$
MCTF_tf_lut_uv_input_shift	$[0, 7]$
MCTF_tf_alpha_max_upper_bound	$[0, 255]$
MCTF_tf_fs_global_smoothness	$[0, 15]$
MCTF_tf_fs_similarity_bound	$[0, 255]$
MCTF_tf_local_enhance_rate	$[0, 15]$
MCTF_tf_var_score_base	$[0, 63]$

(d) NYNR

Parameter	Operational Range
NYNR_max_sad_L1_LL	$[0, 255]$
NYNR_max_sad_L1_HL	$[0, 255]$
NYNR_max_sad_L1_HH	$[0, 255]$
NYNR_luma_lmt_up	$[0, 100]$
NYNR_luma_lmt_lo	$[0, 100]$
NYNR_grad_lmt_up_L0	$[0, 255]$
NYNR_grad_lmt_lo_L0	$[0, 255]$
NYNR_grad_lmt_up_L1	$[0, 255]$
NYNR_grad_lmt_lo_L1	$[0, 255]$

(e) PNR

Parameter	Operational Range
PNR_rad_slope_y	$[0, 15]$
PNR_rad_slope_uv	$[0, 15]$
PNR_nr_enhance_UV_Y0	$[0, 255]$
PNR_nr_enhance_UV_Y1	$[0, 255]$
PNR_nr_enhance_UV_U0	$[0, 255]$
PNR_nr_enhance_UV_U1	$[0, 255]$
PNR_nr_enhance_UV_V0	$[0, 255]$
PNR_nr_enhance_UV_V1	$[0, 255]$
PNR_nr_enhance_texture_epsilon	$[0, 255]$
PNR_L0_nr_strength_Y	$[0, 32]$
PNR_L1_nr_strength_Y	$[0, 32]$
PNR_L2_nr_strength_Y	$[0, 32]$
PNR_L3_nr_strength_Y	$[0, 32]$
PNR_L4_nr_strength_Y	$[0, 32]$
PNR_L1_nr_strength_UV	$[0, 32]$
PNR_L2_nr_strength_UV	$[0, 32]$
PNR_L3_nr_strength_UV	$[0, 32]$
PNR_L4_nr_strength_UV	$[0, 32]$
PNR_L1_nr_strength_UV_Y	$[0, 32]$
PNR_L2_nr_strength_UV_Y	$[0, 32]$
PNR_L3_nr_strength_UV_Y	$[0, 32]$
PNR_L4_nr_strength_UV_Y	$[0, 32]$

(f) EE

Parameter	Operational Range
EE_enhance_strength	$[0, 1]$
EE_high_enhance_factor_max	$[0, 255]$
EE_high_enhance_clamp_max	$[0, 1023]$
EE_high_enhance_clamp_min	$[-1024, 0]$
EE_middle_enhance_clamp_max	$[0, 1023]$
EE_middle_enhance_clamp_min	$[-1024, 0]$

2.2. Training Stage

The training dataset used for the proposed method contains two parts.

- RAW image $\mathbf{I}$. $\mathbf{I}$ is also the input of ISP.
- The label hyperparameters $\mathcal{P}^*_{task}$ corresponding to $\mathbf{I}$ and downstream tasks. For the object detection and image segmentation tasks, the label hyperparameters $\mathcal{P}^*_{task}$ using iterative search to batch annotate on the training set. For the image quality task, the ISP hyperparameter labels are labeled by the image experts.

The training process of the proposed method is described in the main manuscript, and its flow is shown in Figure 2. During the training stage, the weights of the proposed method are trained by minimizing the loss of ordered conditions.

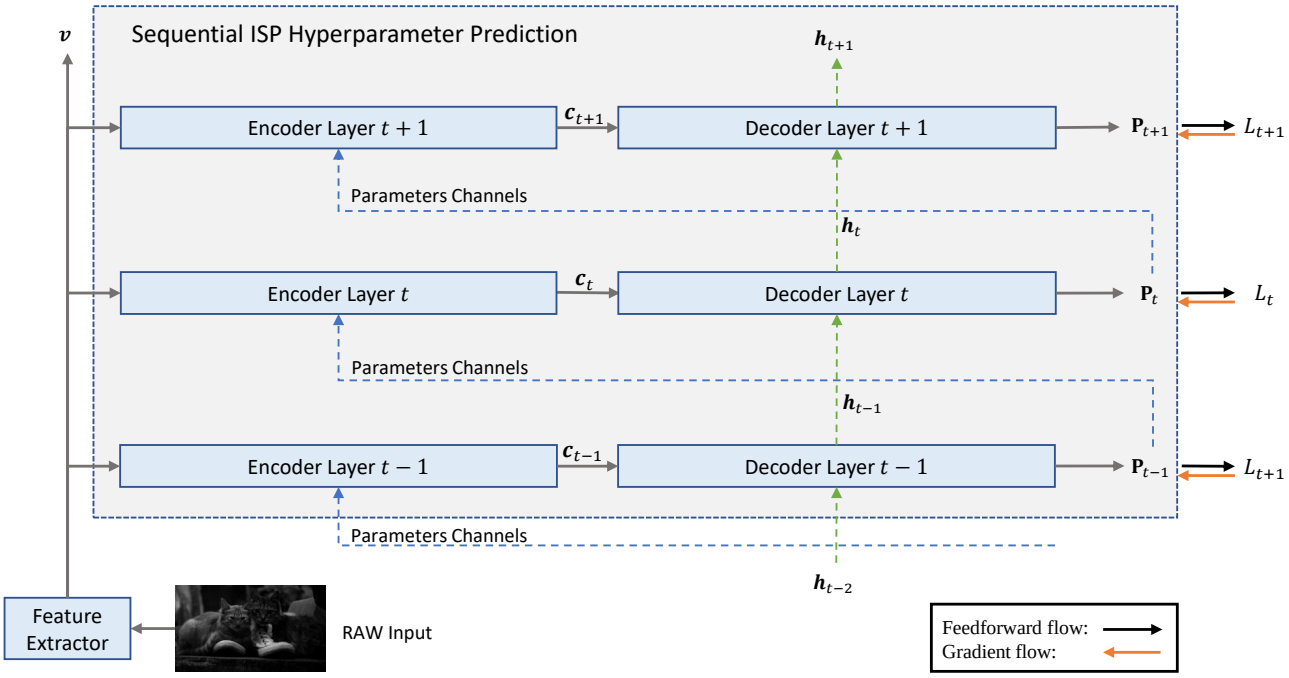

Figure 2. The training process of the proposed method.

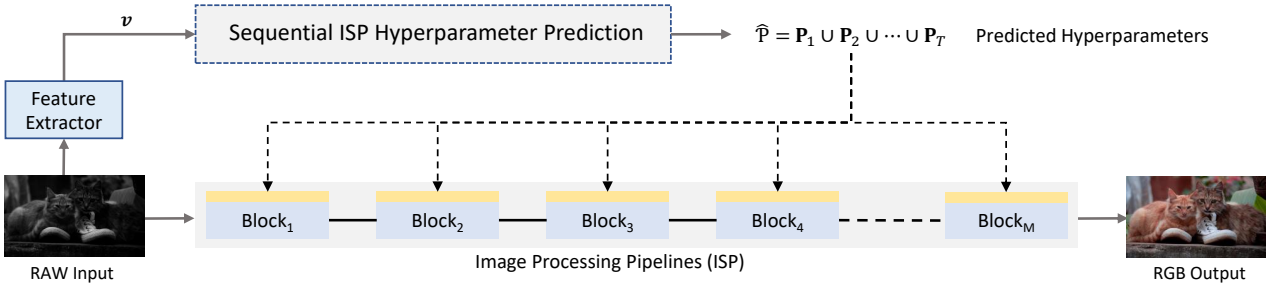

Figure 3. Prediction of ISP parameters using the proposed method. The RGB output is used as input for downstream tasks.

2.3. Inference Stage

In the inference stage, the RAW image $\mathbf{I}$ is used as the input of the proposed method, and the predicted parameters are obtained. The predicted parameters and the RAW image $\mathbf{I}$ are used as the input of the ISP to get the RGB image. The flow is shown in Fig. 3. In the evaluation process of downstream tasks, we transform the RAW test set into an RGB test set by the above process, and the RGB test set is used as the input of downstream tasks to get the evaluation results.

2.4. Additional objective experiments

In this section, we demonstrate additional experiments for the object detection task. To validate the effectiveness of the proposed method towards different models, we use EfficientDet [5] on the COCO dataset [2] as the downstream task of our method to predict hyperparameters on the synthetic ISP. We trained and tested our hyperparameter prediction results. The results in Table 3 show that our method has a great improvement over the default parameters and the expert-tuned parameters. We also conducted experiments on different backbones of the downstream model, and the experimental results show that our method is effective for different models.

Table 3. Object Detection Task using [5] and COCO [2]

	EfficientDet Model	$mAP_{0.5}$	$mAP_{0.75}$	$mAP_{0.5:0.95}$
Default $\mathcal{P}$	D0	0.309	0.200	0.192
Predicted $\hat{\mathcal{P}}$ (Our)		0.491	0.336	0.318
Default $\mathcal{P}$	D1	0.327	0.216	0.207
Predicted $\hat{\mathcal{P}}$ (Our)		0.556	0.394	0.371
Default $\mathcal{P}$	D2	0.353	0.238	0.229
Predicted $\hat{\mathcal{P}}$ (Our)		0.590	0.428	0.402
Default $\mathcal{P}$	D3	0.362	0.251	0.239
Predicted $\hat{\mathcal{P}}$ (Our)		0.618	0.457	0.431
Default $\mathcal{P}$	D4	0.451	0.314	0.299
Predicted $\hat{\mathcal{P}}$ (Our)		0.662	0.496	0.464
Default $\mathcal{P}$	D5	0.412	0.289	0.275
Predicted $\hat{\mathcal{P}}$ (Our)		0.671	0.510	0.475
Default $\mathcal{P}$	D6	0.427	0.302	0.287
Predicted $\hat{\mathcal{P}}$ (Our)		0.674	0.519	0.482
Default $\mathcal{P}$	D7	0.456	0.324	0.308
Predicted $\hat{\mathcal{P}}$ (Our)		0.697	0.560	0.502
Default $\mathcal{P}$	D7X	0.494	0.353	0.335
Expert-tuned		0.689	0.534	0.498
Predicted $\hat{\mathcal{P}}$ (Our)		0.712	0.558	0.517

References

[1] Seon Joo Kim, Hai Ting Lin, Zheng Lu, Sabine Süsstrunk, Stephen Lin, and Michael S Brown. A new in-camera imaging model for color computer vision and its application. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(12):2289–2302, 2012. 3

[2] 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*, pages 740–755, 2014. 3, 5, 6

[3] 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 *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7529–7538, 2020. 1

[4] Haina Qin, Longfei Han, Juan Wang, Congxuan Zhang, Yanwei Li, Bing Li, and Weiming Hu. Attention-aware learning for hyper-parameters prediction in image processing pipelines. In *European Conference on Computer Vision*, pages 271–287. Springer, 2022. 1

[5] Mingxing Tan, Ruoming Pang, and Quoc V Le. Efficientdet: Scalable and efficient object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10781–10790, 2020. 5, 6