Learning to Exploit the Sequence-Specific Prior Knowledge for Image Processing Pipelines Optimization

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

Abstract

The hardware image signal processing (ISP) pipeline is the intermediate layer between the imaging sensor and the downstream application, processing the sensor signal into an RGB image. The ISP is less programmable and consists of a series of processing modules. Each processing module handles a subtask and contains a set of tunable hyperparameters. A large number of hyperparameters form a complex mapping with the ISP output. The industry typically relies on manual and time-consuming hyperparameter tuning by image experts, biased towards human perception. Recently, several automatic ISP hyperparameter optimization methods using downstream evaluation metrics come into sight. However, existing methods for ISP tuning treat the high-dimensional parameter space as a global space for optimization and prediction all at once without inducing the structure knowledge of ISP. To this end, we propose a sequential ISP hyperparameter prediction framework that utilizes the sequential relationship within ISP modules and the similarity among parameters to guide the model sequence process. We validate the proposed method on object detection, image segmentation, and image quality tasks.

1. Introduction

Hardware ISPs are low-level image processing pipelines that convert RAW images into high-quality RGB images. Typically, ISPs include a series of processing modules [5], each of which handles a subtask such as denoising, white balance, demosaicing, or sharpening. Compared to software image processing pipelines, hardware ISPs are faster, more power-efficient, and widely used in real-time products [7, 34], including cameras [28], smartphones, surveillance [21], IoT and driven-assistance systems. ISPs are highly modular and less programmable but with a set of tunable hyperparameters. The industry always relies on manual and costly hyperparameter tuning by image experts [1] to adapt the ISP to different application scenarios.

The ISP is always designed as a sequential pipeline [5] and the configurable hyperparameters of various modules from any ISP aggregate to be a complex parameter space (with tens to hundreds of parameters), making the manual tuning process time-consuming. It is also difficult to subjectively find optimal hyperparameters settings for various downstream tasks (such as object detection and image segmentation [29, 30]). Recently, several automatic ISP hyperparameter optimization methods [17, 31] using downstream evaluation metrics come into sight. These methods tuning hyperparameters for downstream tasks include derivative-free [18] or gradient methods [12, 25, 27] based on differentiable approximation. There are also methods to demonstrate the potential of predicting specific hyperparameters for each image or scene [22]. However, existing methods treat the high-dimensional parameter space as a global black-box space for optimization and prediction all at once, while ignoring the inherent sequence of the ISP modules and the critical intra-correlation among hyperparameters.

Inspired by the operating principles and structure knowledge of ISPs, we first propose a sequential ISP hyperparameter prediction framework (as shown in Fig. 1) that contains Sequential CNN model and Global Similarity Grouping. The sequential CNN model runs recurrently by predicting a group of parameters from several ISP modules, not all parameters, at each step. Meanwhile, the predicted parameters, along with the network's hidden layer and the input data, are in turn encoded as prior knowledge for predicting the following grouping of parameters. The Global Similarity Grouping module divides ISP parameters into multiple

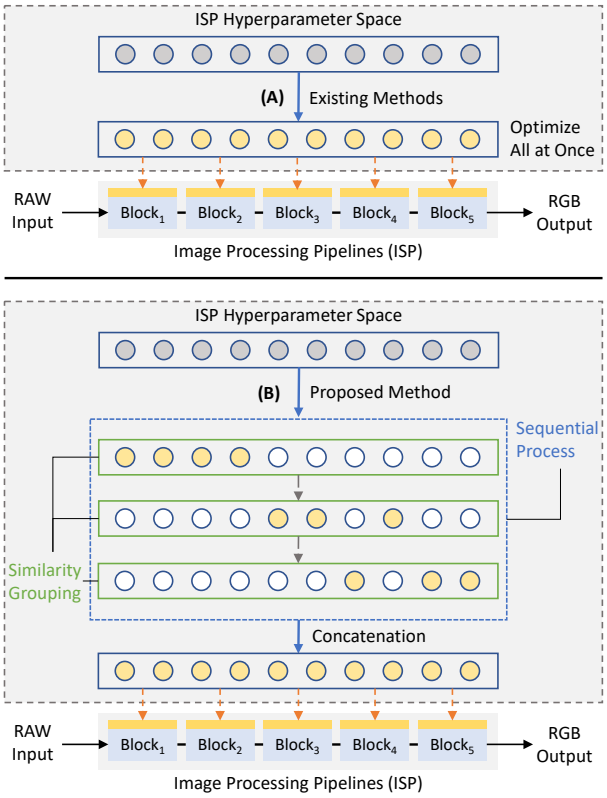

Figure 1. (A) Previous methods treat the hyperparameter space as a black box optimization problem and estimate all parameters at once without considering the prior knowledge of the ISP. (B) Our proposed method first decouples ISP structural knowledge and treats ISP tuning as a sequential prediction problem. It is effective to introduce sequence information and similarity relations in the high-dimensional hyperparameter space.

disjoint groupings. The sequence order between groupings is explored heuristically using prior knowledge of the order of ISP modules. Given the flexibility, groupings are determined based on similarity among parameters, not limited to the same module parameters. The correlation of parameter activation maps learned through the model is used as the basis for parameter groupings.

Our contributions can be summarized as the following:

- We propose a new sequential CNN structure to exploit the sequence processing knowledge within ISP. The potential sequential information among parameters is used to guide the processing of the model.
- We exploit the correlation among parameters by the proposed similarity grouping module. The flexible parameter groupings allow the exploration of cross-module relationships among parameters.
- We validate the effectiveness of our method in a variety of downstream tasks, including object detection, image segmentation, and image quality. In these applications, our method outperforms existing methods.

2. Related Work

ISPs contain several components, and each module carries a specific image-processing algorithm. These algorithms are permanently configured with tens to hundreds of handcrafted parameters, for instance, noise reduction, color space transformation, and sharpening. It is desirable to find parameter configurations for imaging experts that make ISP work well in real-world scenarios and output compressed images with good human perception. Recently, to tackle this challenging optimization problem, several automatic ISP tuning methods were proposed by optimizing the hyperparameters with downstream task feedback [3, 4, 35], for instance, object detection [27, 29], object segmentation [17], and image quality [2, 6, 30, 32].

With the high-level feedback information, recent works always leverage a differentiable mapping between the parameter space and high-level evaluation metrics. Some existing optimization methods try to mimic the whole hardware ISP pipeline as an approximate CNN proxy model and construct an end-to-end differentiable CNN framework to optimize the proxy network [8, 12, 33]. Tseng *et al*. [27] trained an approximate CNN proxy model to mimic hardware ISP and optimized the differentiable CNN model with Stochastic Gradient Descent. Onzon *et al*. [19] propose a neural network for exposure selection that is jointly end-to-end with an object detector and ISP pipeline. Pfister *et al*. [20] proposes to optimize sparsity regularization for denoising. These methods are based on the assumption that a single proxy network can mimic the whole ISP pipeline and cannot be used to optimize the hardware ISPs. Other methods can directly put the hardware ISP into the optimization stage and predict all hyperparameters at once. For instance, Mosleh *et al*. [17, 24] directly optimize hardware ISP by a novel CMA-ES strategy [9] with max-rank-based multi-objective scalarization and initial search space reduction. Kim *et al*. [14] utilize the objective function of multi-output regression for modeling the relation between ISP parameter and IQM score. Qin *et al*. [22] directly infer all ISP hyperparameters via an attention-based CNN method. However, these methods do not take sequential information into consideration. Only Nishimura *et al*. [18] optimize ISP components with sequential prior knowledge based on a 0-th order Nelder-Mead method. Nevertheless, it can only be used to optimize ISP modules one at a time, ignores cross-module connections while viewing the modules in isolation, and is difficult to apply to hardware ISPs.

Therefore, with the massive number of parameters in hardware ISPs, it needs to exploit the structural knowledge to tune ISPs in high-dimensional parameter space accurately. ISP tuning tasks should introduce ISP prior information and explore implicit relationships among parameters.

3. Image Processing Pipelines

For better comparison, we follow the most common ISP modules, and contains the following typical stages [5]:

(1) Optics and Sensor: The sensor captures the illumination through the optics, and the RAW image is the sensor output. The pixel value of RAW is linearly related to the intensity of illumination.

(2) Denoising: Noise is generally considered to be a random variable with zero mean. Denoising will reduce noise but also remove details by blurring the image.

(3) DigitalGain and White Balance: The RAW pixel values are gain-adjusted. Color correction is processed by automatically estimating illuminations or the specific white-balance matrices of common illuminations.

(4) Demosaicking: Convert the color filter array over pixel sensors to RGB values for each pixel by performing interpolation.

(5) Color Space Transform: Map the RAW pixel values to CIE XYZ color space using a 3x3 transform matrix.

(6) Sharpening: Detect edges in the image. Apply filtering measures to eligible edges to improve edge strength and enhance details.

(7) Color and Tone Correction: Applying gamma curves and adjusting image contrast by histogram operations to improve the image's global appearance.

(8) Compression: Image pixel values are compressed to JPEG and storage.

The ISP in this work models stages (2) to (8). This ISP f_{ISP} takes the RAW image $\mathbf{I}$ as input and converts into an RGB image $\mathbf{O}_{\text{ISP}}$:

$$\mathbf{O}_{\text{ISP}} = f_{\text{ISP}}(\mathbf{I}; \mathcal{P}),\ \mathcal{P} \in \mathbb{R}^{N}_{[0,1]}, \tag{1}$$

where $\mathbf{I} \in \mathbb{R}^{W \times H}$, $\mathbf{O}_{\text{ISP}} \in \mathbb{R}^{W \times H \times 3}$. The ISP is controlled by N continuous hyperparameters $\mathcal{P}$. For the discrete parameters, mapping them to continuous values within the range of values facilitates prediction [17].

4. Method

In this section, we devise the sequence model for predicting ISP hyperparameters, as shown in Fig. 2. We first introduce the ISP hyperparameter prediction task and the proposed model framework in Sec. 4.1. The Sequential CNN model design and the Global Similarity Grouping (GSG) for hyperparameters are introduced in Sec. 4.2 and Sec. 4.3.

4.1. Formulation

Given a RAW image $\mathbf{I}$ and a downstream task, the proposed method needs to maximize the following objectives:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{M} \mathcal{L}_{task}(f_{ISP}(\mathbf{I}_i; f_{pred}(\mathbf{I}_i; \boldsymbol{\theta}))), \tag{2}$$

where $\boldsymbol{\theta}$ are the weights of the model, M is the images' number, f_{pred} is the proposed method, and $\hat{\mathcal{P}} = f_{pred}(\mathbf{I}; \boldsymbol{\theta})$ is the prediction ISP parameters. Since ISP is a black-box hardware unit, we convert the optimization objectives to:

$$\boldsymbol{\theta}^* = \arg\min_{\boldsymbol{\theta}} \sum_{i=1}^{M} \mathcal{L}_2(\mathcal{P}^*, f_{pred}(\mathbf{I}_i; \boldsymbol{\theta})), \tag{3}$$

where $\mathcal{P}^*$ is the optimal ISP parameter for image $\mathbf{I}$ and the downstream task. Our method divides the parameter $\mathcal{P}$ into T disjoint subsets, $\mathcal{P} = \mathbf{P}_1 \cup \mathbf{P}_2 \cup \ldots \mathbf{P}_T$. There is an order between them, and $\mathbf{P}_t$ is predicted at step t. This division decomposes the loss function into ordered conditions.

$$\mathcal{L}_2(\mathcal{P}^*, \hat{\mathcal{P}}) = \sum_{t=1}^{T} \mathcal{L}_2(\mathbf{P}_t^*, \hat{\mathbf{P}}_t), \tag{4}$$

The proposed Sequential CNN model and Global Similarity Grouping achieve the above process. Sequential CNN model uses a fully convolutional structure and runs the loop T times to predict the sequential parameters groupings. Global Similarity Grouping exploits the implicit relations among parameters to achieve flexible parameter grouping.

4.2. Sequential CNN Model

In ISPs, one module's output is input to the next module. The effect of one set of parameters will be reflected in the image features and influence the next set of parameters. So we encode visual contextual features $\boldsymbol{c}_t$ as the following:

$$\boldsymbol{c}_t = f_{encoder}(\boldsymbol{v}, \mathbf{P}_1 \cup \mathbf{P}_2 \cup \ldots \mathbf{P}_{t-1}), \tag{5}$$

where $\boldsymbol{v}$ is the visual feature extracted by the feature extractor on image $\mathbf{I}$. The feature extractor is a pre-trained ResNet [11] model. At step t, the encoder encodes the visual features $\boldsymbol{v}$ and the predicted parameters sets at the time from 1 to $t-1$ into visual context features $\boldsymbol{c}_t$, after which $\boldsymbol{c}_t$ is used to predict of $\mathbf{P}_t$. Specifically, we concatenate the visual features $\boldsymbol{v}$ with parameters channels and convolve them. The number of parameters channels is the same as that of hyperparameter $\mathcal{P}$. Each channel replicates the value of the corresponding hyperparameter in the spatial dimension. Only the channels corresponding to $\{\mathbf{P}_1, \ldots, \mathbf{P}_{t-1}\}$ are assigned parameters' values, and the rest of the channels are assigned 0. The subsequent convolution module is designed as three parallel branching structures with 1x1, 3x3, and 5x5 convolutions. The three output features are concatenated to obtain the visual contextual features $\boldsymbol{c}_t$. Concatenating feature maps with different receptive fields generate multi-scale feature representations, which help encode parameters with different effect ranges.

There are five decoders to complete the decoding process. Each decoder contains a 3x3 convolution followed by a ReLU unit and a span2 upsampling. Each decoder reduces

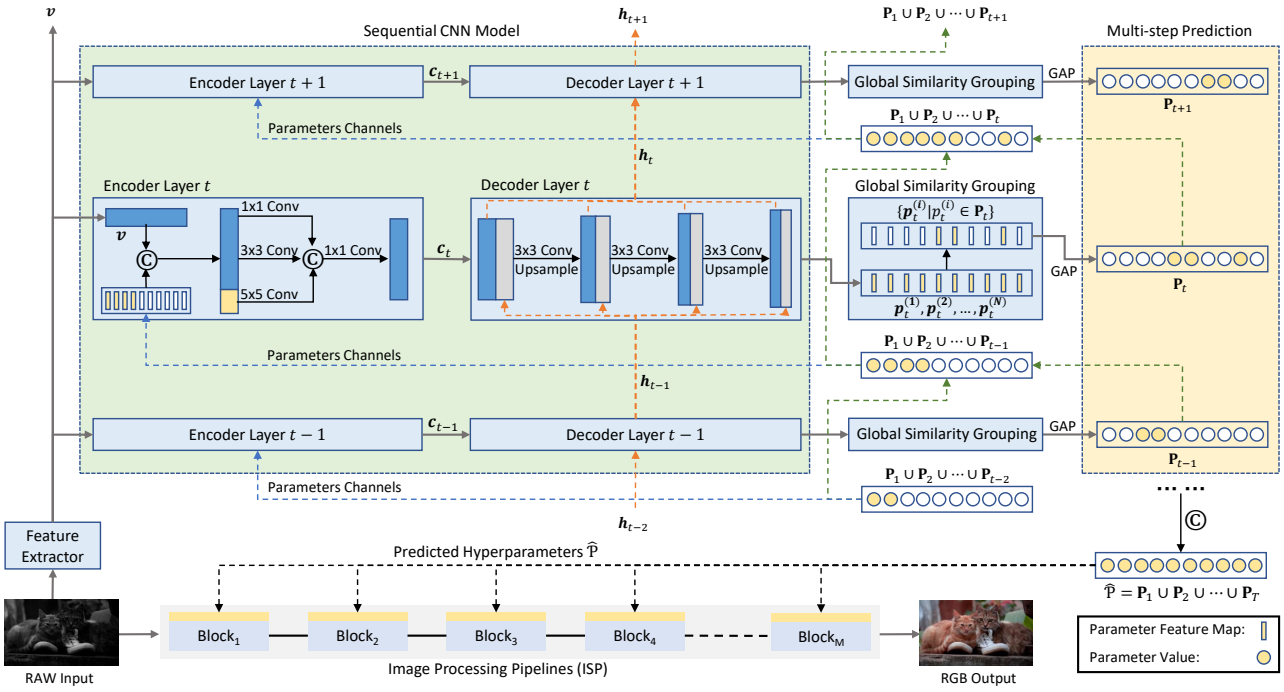

Figure 2. Proposed sequence parameters prediction framework and ISP tuning process. Sequential CNN model has a fully convolutional encoder-decoder structure and predicts parameter groupings sequentially. Global Similarity Grouping module divides the parameters into ordered disjoint sub-groupings by implicit relations among parameters. Multi-step prediction results are concatenated to obtain the predicted hyperparameters, which are set as the parameters setting of the ISP.

the number of feature channels by half, and the last one outputs a parameter feature map for N channels. We replicate the feature maps output by all decoders as hidden states $\boldsymbol{h}_t$ and concatenate them with the intermediate feature maps of the decoders at step $t+1$. The input feature maps of the decoder at step $t+1$ are the concatenate of the feature maps generated by the previous decoder at step $t+1$ and step t. In this process, the intermediate features of the convolutional network are passed as the hidden state $\boldsymbol{h}_t$ and combined with the visual context $\boldsymbol{c}_t$ for parameter prediction.

The visual context features $\boldsymbol{c}_t$ and the network hidden state $\boldsymbol{h}_{t-1}$ at time $t-1$ are input to the decoder, and the parameter feature maps for N channels are generated:

$$\begin{aligned}\boldsymbol{\Phi}_t &= f_{decoder}(\boldsymbol{c}_t, \boldsymbol{h}_{t-1}) \\ &= f_{decoder}(f_{encoder}(\boldsymbol{v}, \mathbf{P}_1 \cup \ldots \mathbf{P}_{t-1}), \boldsymbol{h}_{t-1}),\end{aligned} \tag{6}$$

where $\boldsymbol{\Phi}_t = \{\boldsymbol{p}_t^{(1)}, \boldsymbol{p}_t^{(2)}, \ldots, \boldsymbol{p}_t^{(N)}\}$ is the set of feature maps of all parameters at step t. And $\boldsymbol{p}_t^{(i)}$ is the feature map of parameter p_i at time t. And $\boldsymbol{\Phi}_t$ go through the GSG module and the global average pooling (GAP) layer to obtain the predicted parameter grouping $\mathbf{P}_t$ at step t:

$$\begin{aligned}\mathbf{P}_t &= f_{gap}(f_{gsg}(\boldsymbol{\Phi}_t)) \\ &= f_{gap}(f_{gsg}(f_{scnn}(\boldsymbol{v}, \boldsymbol{h}_{t-1}, \mathbf{P}_1 \cup \ldots \mathbf{P}_{t-1}))),\end{aligned} \tag{7}$$

For convenience, f_{scnn} is used to express the Sequential CNN model that contains the encoder and decoder mentioned above. Eq. 7 shows the function of the sequential model at one step. The model is run T times recurrently by passing the hidden layer $\boldsymbol{h}_i$ and the predicted parameter groupings $\mathbf{P}_i$, where $i = 1, \ldots, T$. All the predicted groupings are concatenated to obtain the predicted parameters.

4.3. Global Similarity Grouping

GSG runs once at the beginning of an iteration, and partitions hyperparameters into T disjoint subsets. Multiple steps of an iteration run mask operation (before GAP) according to the results of GSG. Parameters groups number T is less than or equal to the number of ISP modules M. Some modules in ISP have similar roles. Grouping these similar parameters can reduce the model steps and make it more efficient for a large number of parameters and modules.

Hyperparameters $\mathcal{P}$ have an original division $\mathcal{P} = \mathbf{P}'_1 \cup \mathbf{P}'_2 \cup \ldots \mathbf{P}'_M$, where $\mathbf{P}'_i$ is the set of parameters of module i. Details are described in Sec. 3 and supplement material. Considering the flexibility, we take the primal division as prior knowledge and use the correlation between parameters to explore the sequential divisions of parameters.

The Sequential CNN model generates feature maps of all parameters, and the feature maps are passed through the

Algorithm 1: Sequential initial clustering centers

Input: Model partition graphs: $\mathbf{P}'_1, \ldots, \mathbf{P}'_M$; Weighted adjacency matrix: $\mathbf{W}$; Steps in the prediction model: $T \leq M$.

Output: Sequential clustering centers: $c_1, \ldots, c_T$.

$vol(\mathbf{P}'_x) = \sum_{p^{(i)} \in \mathbf{P}'_x} \sum_{j=1}^{n} w_{ij}$;

$Ncut(\mathbf{P}'_x, \mathbf{P}'_y) = (\frac{1}{vol(\mathbf{P}'_x)} + \frac{1}{vol(\mathbf{P}'_y)}) \sum_{p^{(i)} \in \mathbf{P}_x, p^{(j)} \in \overline{\mathbf{P}_y}} w_{ij}$;

for $I \in [1, M-T]$ **do**

  $i \leftarrow i \; s.t. \; max\{Ncut(\mathbf{P}'_i, \mathbf{P}'_{i+1}) | i \in [1, M-I]\}$

  $\mathbf{P}'_i \leftarrow \mathbf{P}'_i \cup \mathbf{P}'_{i+1}$

  if $i \neq M-I$ **then**

    $\mathbf{P}'_{i+1}, \ldots, \mathbf{P}'_{M-I} \leftarrow \mathbf{P}'_{i+2}, \ldots, \mathbf{P}'_{M-I+1}$

$\mathbf{C}_1, \ldots, \mathbf{C}_T \leftarrow \mathbf{P}_1, \ldots, \mathbf{P}_T$;

for $J \in [1, T]$ **do**

  $c_J \leftarrow p^{(x)} \; s.t. \; max\{\sum_{p^{(y)} \in \mathbf{C}_J} w_{xy} | p^{(x)} \in \mathbf{C}_J\}$

GAP module to obtain the predicted parameters. The pixel values on the feature map represent the predicted parameter values in the corresponding region of the image $\mathbf{I}$. The feature map is a higher-dimension representation of the effect of the parameter on the image [22]. Therefore, we use the similarity between feature maps to perform the division of similar parameters. Here we directly use the cosine similarity to measure the feature map similarity as follows:

$$L_{cos}(p^{(i)}, p^{(j)}) = \frac{\boldsymbol{p}^{(i)} * \boldsymbol{p}^{(j)}}{|\boldsymbol{p}^{(i)}||\boldsymbol{p}^{(j)}|}, \tag{8}$$

where $\boldsymbol{p}^{(i)}$ and $\boldsymbol{p}^{(j)}$ are the feature maps of $p^{(i)}$ and $p^{(j)}$, respectively, and $L_{cos}(p^{(i)}, p^{(j)})$ is the similarity of $p^{(i)}$ and $p^{(j)}$. After the decoder outputs the parameter feature maps, We can obtain the similarity values between all parameters. All parameters can be expressed in the form of a graph:

$$G = (\mathbf{V}, \mathbf{E}), \mathbf{V} = \{p^{(1)}, p^{(2)}, \ldots, p^{(N)}\}. \tag{9}$$

Graph G is weighted, and $\mathbf{E}$ is the set of edges between the vertices. The weight of each edge between two vertices $p^{(i)}$ and $p^{(j)}$ is $w_{ij} = L_{cos}(p^{(i)}, p^{(j)}) \geq 0$. The weighted adjacency matrix is $\mathbf{W} = (w_{ij})_{i,j=1,\ldots,n}$.

The problem of partitioning the set $\mathcal{P}$ of parameters into T disjoint subsets is transformed into partitioning graph G:

$$\begin{aligned} \mathbf{V} =& \mathbf{P}_1 \cup \mathbf{P}_2 \cup \ldots \mathbf{P}_T, \\ & \mathbf{P}_i \cap \mathbf{P}_j = \varnothing, 1 \leq i < j \leq T. \end{aligned} \tag{10}$$

Graph G should be divided so that the parameters are dissimilar between groups and similar in the same group. That means the edges between different groups have low weights and the edges within a group have high weights. The above optimization objective can be expressed as follows:

$$min \sum_{t=1}^{T} \frac{W(\mathbf{P}_t, \overline{\mathbf{P}_t})}{vol(\mathbf{P}_t)} = \sum_{t=1}^{T} \frac{\sum_{p^{(i)} \in \mathbf{P}_t, p^{(j)} \in \overline{\mathbf{P}_t}} w_{ij}}{\sum_{p^{(i)} \in \mathbf{P}_t} \sum_{j=1}^{N} w_{ij}}. \tag{11}$$

For the above optimization objectives and sequence partitioning of the parameters, we use the prior structure information of ISP and perform spectral clustering of the parameters. Precisely, we first compute the normalized Laplacian matrix $\mathbf{L}$ of the adjacency matrix $\mathbf{W}$. The first T eigenvectors of $\mathbf{L}$ are used as columns to form $\mathbf{T} \in \mathbb{R}^{n \times T}$. After $\mathbf{T}$ is normalized by rows, the vector $\boldsymbol{y}^{(i)}$ of the parameter $p^{(i)}$ corresponds to the i-th row of $\mathbf{T}$.

To make cluster $\mathbf{P}_1, \ldots, \mathbf{P}_T$ with sequence information, we introduce serialized initial clustering centers $c_1, \ldots, c_T$. The sequential partition graphs $\mathbf{P}'_1, \ldots, \mathbf{P}'_M$ can be obtained based on the structure information of ISP. The graphs that are adjacent and related in order are merged to obtain T partitioned graphs $\mathbf{C}_1, \ldots, \mathbf{C}_T$, and c_i is the vertex in $\mathbf{C}_i$ that is most closely connected to other vertices. The specific process is described in Algorithm 1. With $c_1, \ldots, c_T$ as the initial clustering center, the points $(\boldsymbol{y}^{(i)})_{i=1,\ldots,N}$ are clustered using the k-means algorithm to obtain the clusters $\mathbf{P}_1, \ldots, \mathbf{P}_T$. $\mathbf{P}_t$ is the cluster with c_t as the initial cluster center and is predicted at step t in the sequential model.

The final obtained $\mathbf{P}_1, \ldots, \mathbf{P}_T$ for the division of parameters $\mathcal{P}$ is sequential, and its order is determined by the original ISP structure information. The parameters are similar within each grouping based on the similarity of the feature maps corresponding to the parameters. This flexible division enables cross-module grouping of similar parameters.

5. Experiments

5.1. Settings

In this section, we introduce the ISPs, datasets, and downstream tasks used for evaluation.

(1) Object detection on MS COCO [15] using [23]. For comparison with [22] and [17], we use the same synthetic (simulated) ISP to process simulated RAW as the upstream module for the downstream task. The ISP contains 20 hyperparameters, and detailed information is described in the supplementary material.

(2) Image segmentation using [10] on COCO with the same experimental setup as the object detection task.

(3) ISP parameter prediction for image quality. A SONY IMX766 CMOS sensor is used to collect the RAW dataset. The RAW data were processed using Qualcomm Spectra 580 ISP oriented to image quality. The ISP contains 144 hyperparameters, and detailed information is described in the supplementary material.

Figure 3. Image analytic vision tasks: Object Detection (left) and Image Segmentation (right) on COCO dataset. Default ISP hyperparameters (top), expert-tuned hyperparameters (middle), and the proposed method (bottom). The proposed method achieve better performance for downstream tasks.

Table 1. Synthetic ISP optimization for Object Detection on COCO. We borrow the results from [17] by constructing a similar processing pipeline with the ISP used in other methods.

	ISP Model	mAP 0.5	mAP 0.75	mAP 0.5:0.95
Default $\mathcal{P}$	ISP [17]	15.	-	-
blockwise-tuned [18]		20.	-	-
Hardware-tuned [17]		39.	-	-
Default $\mathcal{P}$	ISP Sec.3	34.1	22.4	21.4
Expert-tuned		56.8	40.8	37.7
Attention-ware [22]		61.0	44.1	41.0
Sequential-tuned		**62.8**	**45.2**	**42.3**

Table 2. Synthetic ISP optimization for Image Segmentation on COCO. We borrow the results from [17] by constructing a similar processing pipeline with the ISP used in other methods.

	ISP Model	mAP 0.5	mAP 0.75	mAP 0.5:0.95
Default $\mathcal{P}$	ISP [17]	12.	-	-
Hardware-tuned [17]		32.	-	-
Default $\mathcal{P}$	ISP Sec.3	22.7	13.2	12.0
Expert-tuned		46.9	28.2	27.1
Attention-ware [22]		52.3	33.4	31.5
Sequential-tuned		**54.2**	**35.1**	**32.6**

During the training stage, the RAW images are input into the model, which sequentially predicts several groups of hyperparameters. As described in Sec. 4.1, the predicted parameters and ground-truth parameters are used to calculate the losses for training the model. We use a combination of iterative search and manual annotation to batch annotate the training set, as described in detail in the supplementary material. The pre-trained Resnet [11] for feature extraction and the rest of the model is trained using loss defined in Eq. 4 and the RMSprop optimizer. The learning rate was initially set to 10^{-4} and reduced to 10^{-6} after 200 calendar hours. The training process was performed for 400 epochs.

In the evaluation stage, we used the test set as model input to predict the hyperparameters for each RAW image. The RAW image and predicted hyperparameters were input to the ISP to obtain RGB output as input to the downstream tasks. The upstream RGB output is used as input for the object detection and instance segmentation tasks, and the evaluation metric uses the mean accuracy (mAP) [15]. The PSNR and SSIM between the RGB output and the reference image are used as evaluation metrics for the image quality.

5.2. Optimization for Object Detection

In this task, we use an existing large sRGB dataset and a synthetic ISP to evaluate our ISP hyperparameter sequence

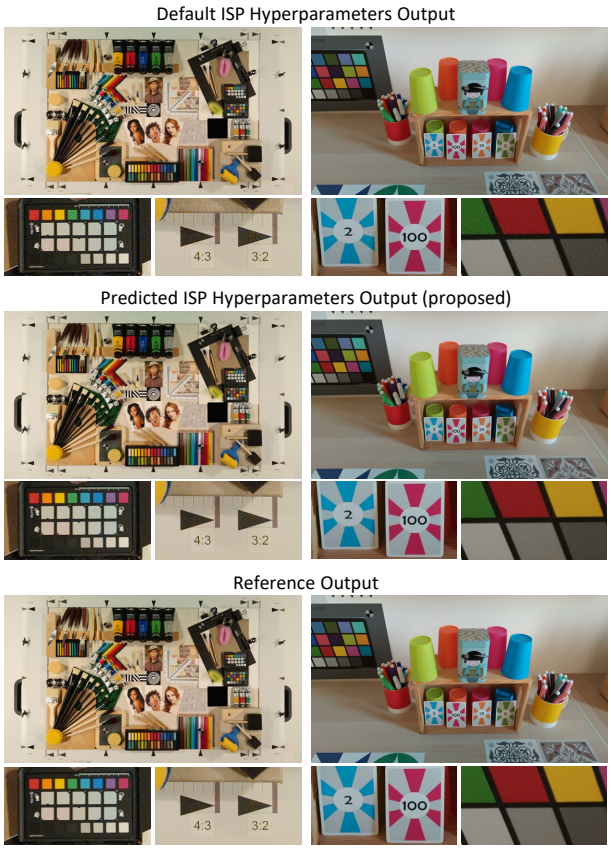

Figure 4. Default setting hyperparameters (top), proposed method (middle), and reference images (bottom). The proposed method achieves an image quality close to the reference image.

Table 3. Perceptual image quality using default parameters setting, recent method [22], and the proposed method.

	ISP Model	PSNR	SSIM
Default Parameters	Hardware ISP Sec. 5.1	30.73	0.720
Attention-ware [22]		40.08	0.965
Sequential-tuned		**45.29**	**0.982**

prediction method. The RGB images are converted to RAW as input to the ISP by simulating the processing of sRGB images with RAW data [13]. We use pre-trained [23] for object detection on the COCO dataset [15] as a downstream task. The proposed model is trained for the object detection task and evaluated using the mAP [15]. The quantitative evaluation results are shown in Table 1. Compared to the default parameter settings of the untuned ISP, our method shows a more significant improvement over current methods and better results than non-sequential prediction methods. The subjective results are shown in Fig. 3. The proposed method emphasizes the image's texture details and color features compared to the expert-tuned image. Although expert-tuned images are more consistent with human visual preferences, the predicted parameters allow for better results in these images for the object detection tasks.

5.3. Optimization for Image Segmentation

Next, we use pre-trained [10] for image segmentation task on the COCO dataset as the downstream task. The proposed method is trained and tested on the simulated RAW COCO dataset and synthetic ISP. The results in Table 2 show that the proposed method has the best final results for the instance segmentation task compared to the recent methods. It also significantly improves over the default parameters (baseline). Fig. 3 shows the default parameters, the expert tuning parameters, and examples of the proposed method. Similar to the results obtained in the object detection task, the predicted parameters improve the performance of the downstream task (image segmentation) by adjusting the texture and color of the image. The predicted parameters of our model achieve an improvement of 0.3 mAP and 0.2 mAP compared to the default parameters and the expert-tuned parameters, respectively, and 0.1 mAP compared to the non-sequential prediction method.

5.4. Optimization for Image Quality

The image quality optimization task aims to achieve a specific image rendering preference [16]. This specific rendering preference allows the output RGB to match the human visual preference by combining the visibility of various distortions in the image (noise, color shifts, blockiness) [26] and the imaging style. In this task, we aim to make the RAW image undergoes ISP to produce an RGB as close as possible to the reference image by the hyperparameters predicted by the proposed method. We collected a dataset for the image quality optimization task. The dataset contains images from 251 scenes acquired by the IMX766 sensor. The reference images are generated through the Spectra 580 ISP with the RAW images and the expert-tuned parameters as input. We used PSNR and SSIM metrics to evaluate the consistency between the images generated by the predicted parameters and the reference images. The quantitative results are shown in Table 3, and the proposed method consists of better PSNR and SSIM results than recent methods. Our method brings more improvements based on the default parameter settings. Fig. 4 shows some examples of image quality tasks. Our method produces images with better visual quality that conform to human preferences regarding texture details, noise control, and color representation.

5.5. Ablation Study

Sequential Structure. Recent ISP hyperparameter tuning methods, both iterative optimization and direct prediction methods, treat the set of ISP hyperparameters as a black box space and estimate all the parameters at once. In contrast, the proposed method allows step-by-step sequential

Table 4. Ablation study of evaluating the proposed sequential structure and similarity grouping. The first column is the results for multiple step settings of the proposed method, and the second is the results of multiple changes in Global Similarity Grouping module.

	Steps	Sequence	Prior Knowledge	Similarity Group	PSNR	SSIM
Sequence prediction (Multi step-settings)	one	-	-	-	39.52	0.957
	two	✓	✓	✓	40.05	0.962
	three	✓	✓	✓	42.66	0.971
	four	✓	✓	✓	**45.29**	**0.982**
	five	✓	✓	✓	45.23	0.982
Blockwise sequence & similarity group	**four**	✓	✓	✓	**45.29**	**0.982**
Blockwise sequence & blockwise group		✓	✓	-	43.57	0.969
Random sequence & similarity group		✓	-	✓	42.62	0.960
Random sequence & random group		✓	-	-	39.30	0.947

prediction of ISP parameters, achieving more refined predictions by modeling the sequential relationships among parameter groupings. We verify that this sequential tuning method has better results than the non-sequential method.

The effectiveness of sequence prediction is tested on the image quality dataset and a Spectra 580 ISP with 144 parameters. The results are shown in Table 4. We tested the non-sequential and sequential methods under multiple settings, and the test metrics are the PSNR and SSIM results of the output and the reference image. The number of model running steps equals the number of parameter groupings. For the same number of hyperparameters, the fewer steps of the model, the more parameters are predicted in each step. When the step is set to one, the model predicts all hyperparameters simultaneously, equivalent to the non-sequential method. The sequential prediction results have better results than the non-sequential ones. We also validate the results of several experiments with different step counts. The method loop steps size larger to generate long-distance dependencies, which will be more challenging to learn and converge. Endlessly increasing the size of steps with the same dataset may suffer performance degradation. Experimental results show that a compromise step count setting achieves the best results for a particular experimental setting.

Global Similarity Grouping. In sequential prediction, each step predicts a sub-grouping of the hyperparameters. Based on the block-wise structure of the ISP and the correlation between parameters, the Global Similarity Grouping module defines the order between different groups and how the parameters within the groups are divided. We validated the effectiveness of this adaptive grouping approach. The experimental setup is consistent with the ablation experiments above, and the results are shown in Table 4.

To verify the effectiveness of the prior structural information of ISP, we changed the Global Similarity Grouping in Sec. 4.3 by removing the block-wise sequence clustering centers and directly performing spectral clustering for all parameters. The model uses random order for prediction for multiple parameter groupings clustered out. The results in Table 4 show that the approach introducing the ISP prior structure information has a better result. To verify the effectiveness of cross-module grouping achieved by parameter similarity, we define the parameter grouping pattern directly with the block-wise structural information of ISP. Parameters of the same module are grouped into the same group, while the order between groups is consistent with the ISP module sequence. The results in Table 4 show the effectiveness of the similarity grouping with the introduction of parameter correlation. Similarity group compared to block-wise group is obtained from GSG by parameter similarity while predicting similar parameters at once will improve the accuracy. Finally, we remove both prior knowledge and similarity grouping by randomly defining the hyperparameter division and the order among groups. As shown in Table 4, such a setup has the worst results.

6. Conclusions

This work proposes a novel sequential method for ISP tuning that uses module relation and parameter similarity to guide the process. It decouples the ISP structure and treats tuning as a sequential prediction problem. It groups parameters to tune ISP sequentially. Experiments show that introducing sequence information and similarity relations in the high-dimensional parameter space improves performance.

Acknowledgments This work was supported by the National Key Research and Development Program of China (Grant No. 2020AAA0106800), the National Natural Science Foundation of China (No. 62192785, No. 61972071, No. U1936204, No. 62122086, No. 62036011, No. U2033210, No. 62172413, No. 62192782 and No. 61721004), the Beijing Natural Science Foundation No. M22005, and L223003, the Major Projects of Guangdong Education Department for Foundation Research and Applied Research (Grant: 2017KZDXM081, 2018KZDXM066), Guangdong Provincial University Innovation Team Project (Project No. 2020KCXTD045). The work of Bing Li was also supported by the Youth Innovation Promotion Association, CAS.

References

[1] Ieee standard for camera phone image quality. IEEE Std 1858-2016 (Incorporating IEEE Std 1858-2016/Cor 1-2017), pages 1–146, 2017. 1

[2] Rémi Bardenet, Mátyás Brendel, Balázs Kégl, and Michele Sebag. Collaborative hyperparameter tuning. In International conference on machine learning, pages 199–207, 2013. 2

[3] James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. Advances in neural information processing systems, 24, 2011. 2

[4] James Bergstra, Daniel Yamins, and David Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In International conference on machine learning, pages 115–123, 2013. 2

[5] Michael S Brown and SJ Kim. Understanding the in-camera image processing pipeline for computer vision. In IEEE International Conference on Computer Vision, volume 3, 2019. 1, 3

[6] Mark Buckler, Suren Jayasuriya, and Adrian Sampson. Reconfiguring the imaging pipeline for computer vision. In Proceedings of the IEEE International Conference on Computer Vision, pages 975–984, 2017. 2

[7] Yue Cao, Xiaohe Wu, Shuran Qi, Xiao Liu, Zhongqin Wu, and Wangmeng Zuo. Pseudo-isp: Learning pseudo in-camera signal processing pipeline from a color image denoiser. arXiv preprint arXiv:2103.10234, 2021. 1

[8] Chen Chen, Qifeng Chen, Jia Xu, and Vladlen Koltun. Learning to see in the dark. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 3291–3300, 2018. 2

[9] Nikolaus Hansen, Sibylle D Müller, and Petros Koumoutsakos. Reducing the time complexity of the derandomized evolution strategy with covariance matrix adaptation (cma-es). Evolutionary computation, 11(1):1–18, 2003. 2

[10] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In Proceedings of the IEEE international conference on computer vision, pages 2961–2969, 2017. 5, 7

[11] 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, pages 770–778, 2016. 3, 6

[12] Andrey Ignatov, Luc Van Gool, and Radu Timofte. Replacing mobile camera isp with a single deep learning model. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops, pages 536–537, 2020. 1, 2

[13] 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. 7

[14] Younghoon Kim, Jungmin Lee, Sung-Su Kim, Cheoljong Yang, TaeHyung Kim, and JoonSeo Yim. Dnn-based isp parameter inference algorithm for automatic image quality optimization. Electronic Imaging, 2020(9):315–1, 2020. 2

[15] 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. 5, 6, 7

[16] Rafał K Mantiuk, Anna Tomaszewska, and Radosław Mantiuk. Comparison of four subjective methods for image quality assessment. In Computer graphics forum, volume 31, pages 2478–2491. Wiley Online Library, 2012. 7

[17] 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, 2, 3, 5, 6

[18] 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, pages 2471–2475. IEEE, 2018. 1, 2, 6

[19] Emmanuel Onzon, Fahim Mannan, and Felix Heide. Neural auto-exposure for high-dynamic range object detection. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 7710–7720, 2021. 2

[20] Luke Pfister and Yoram Bresler. Learning filter bank sparsifying transforms. IEEE Transactions on Signal Processing, 67(2):504–519, 2018. 2

[21] Buu Phan, Fahim Mannan, and Felix Heide. Adversarial imaging pipelines. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 16051–16061, 2021. 1

[22] Haina Qin, Longfei Han, Juan Wang, Congxuan Zhang, Yanwei Li, Bing Li, and Weiming Hu. Attention-aware learning for hyperparameters prediction in image processing pipelines. In European Conference on Computer Vision, pages 271–287. Springer, 2022. 1, 2, 5, 6, 7

[23] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. arXiv preprint arXiv:1804.02767, 2018. 5, 7

[24] Nicolas Robidoux, Luis E Garcia Capel, Dong-eun Seo, Avinash Sharma, Federico Ariza, and Felix Heide. End-to-end high dynamic range camera pipeline optimization. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 6297–6307, 2021. 2

[25] 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, 2018. 1

[26] Kim-Han Thung and Paramesran Raveendran. A survey of image quality measures. In IEEE international conference for technical postgraduates, pages 1–4, 2009. 7

[27] Ethan Tseng, Felix Yu, Yuting Yang, Fahim Mannan, Karl ST Arnaud, Derek Nowrouzezahrai, Jean-François Lalonde, and Felix Heide. Hyperparameter optimization in black-box image processing using differentiable proxies. ACM Transactions on Graphics, 38(4):27–1, 2019. 1, 2

[28] Peter van Beek, Chyuan-Tyng Roger Wu, Baishali Chaudhury, and Thomas R Gardos. Boosting computer vision performance by enhancing camera isp. Electronic Imaging, 2021(17):174–1, 2021. 1

[29] Chyuan-Tyng Wu, Leo F Isikdogan, Sushma Rao, Bhavin Nayak, Timo Gerasimow, Aleksandar Sutic, Liron Ainkedem, and Gilad Michael. Visionisp: Repurposing the image signal processor for computer vision applications. In IEEE International Conference on Image Processing, pages 4624–4628. IEEE, 2019. 1, 2

[30] Lucie Yahiaoui, Ciarán Hughes, Jonathan Horgan, Brian Deegan, Patrick Denny, and Senthil Yogamani. Optimization of isp parameters for object detection algorithms. Electronic Imaging, 2019(15):44–1, 2019. 1, 2

[31] Cheoljong Yang, Jinhyun Kim, Jungmin Lee, Younghoon Kim, Sung-Su Kim, TaeHyung Kim, and JoonSeo Yim. Effective isp tuning framework based on user preference feedback. Electronic Imaging, 2020(9):316–1, 2020. 1

[32] Dani Yogatama and Gideon Mann. Efficient transfer learning method for automatic hyperparameter tuning. In Artificial intelligence and statistics, pages 1077–1085. PMLR, 2014. 2

[33] Ke Yu, Zexian Li, Yue Peng, Chen Change Loy, and Jinwei Gu. Reconfigisp: Reconfigurable camera image processing pipeline. arXiv preprint arXiv:2109.04760, 2021. 2

[34] Syed Waqas Zamir, Aditya Arora, Salman Khan, Munawar Hayat, Fahad Shahbaz Khan, Ming-Hsuan Yang, and Ling Shao. Cycleisp: Real image restoration via improved data synthesis. In Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 2696–2705, 2020. 1

[35] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 8697–8710, 2018. 2