Hardware-in-the-loop End-to-end Optimization of Camera Image Processing Pipelines

Ali Mosleh[1] Avinash Sharma[1] Emmanuel Onzon[1]
Fahim Mannan[1] Nicolas Robidoux[1] Felix Heide[1,2]

[1]Algolux [2]Princeton University

Abstract

Commodity imaging systems rely on hardware image signal processing (ISP) pipelines. These low-level pipelines consist of a sequence of processing blocks that, depending on their hyperparameters, reconstruct a color image from RAW sensor measurements. Hardware ISP hyperparameters have a complex interaction with the output image, and therefore with the downstream application ingesting these images. Traditionally, ISPs are manually tuned in isolation by imaging experts without an end-to-end objective. Very recently, ISPs have been optimized with 1st-order methods that require differentiable approximations of the hardware ISP. Departing from such approximations, we present a hardware-in-the-loop method that directly optimizes hardware image processing pipelines for end-to-end domain-specific losses by solving a nonlinear multi-objective optimization problem with a novel 0th-order stochastic solver directly interfaced with the hardware ISP. We validate the proposed method with recent hardware ISPs and 2D object detection, segmentation, and human viewing as end-to-end downstream tasks. For automotive 2D object detection, the proposed method outperforms manual expert tuning by 30% mean average precision (mAP) and recent methods using ISP approximations by 18% mAP.

1. Introduction

Hardware ISPs are ubiquitous low-level image processing pipelines present in nearly all commodity cameras and a wide variety of applications like digital still photography, camera phones, video surveillance, robotics, driver-assistance systems, and self-driving vehicles. ISPs transform RAW sensor data into images suitable for human viewing or downstream analytic tasks. This transformation typically includes several processing blocks that operate power-efficiently at real-time rates, which is critical for applications in robotics or self-driving vehicles. Growing sensor resolutions mandate efficient processing pipelines in hardware. For example, the 8MPix Sony IMX324 sensor demands processing of about 1.5GB of RAW high dynamic range data per second.

Existing hardware ISPs typically consist of proprietary black box blocks with little information exposed to users except for a set of registers with their operational ranges. The behavior of an ISP is configurable with a set of user-adjustable hyperparameters. Hyperparameter values not only affect the output image but also the domain-specific application. Traditionally, imaging experts manually tune the hyperparameters of the ISP on a small dataset, using a combination of visual inspection and image quality metrics [9]. The resulting handcrafted hyperparameter settings are consequently biased towards human perception, and do not necessarily benefit analytic higher-level vision tasks [36, 38], see Fig. 1.

Optimizing a hardware ISP for an end-to-end loss is challenging because the parameter space is formed by tens to hundreds of categorical and continuous parameters that can affect the downstream task in a complex and nonlinear manner via the intermediate ISP image output. For example, a single binary parameter may switch between completely different algorithmic branches. A full grid search is not an alternative due to the large combinatorial space of hyperparameters. Several recent works aim at automating this process by solving ISP hyperparameter optimization problems. However, to tackle this challenging optimization problem, they rely on software approximations of the hardware ISP to allow either block coordinate descent [30], which requires detailed knowledge of ISP block internals, or end-to-end gradient-based methods, which requires the approximation to be differentiable [35]. We demonstrate that such approximations and first-order methods are prone to local minima for a number of recent hardware ISPs.

In this work, we depart from optimizing software approximations and demonstrate that it is possible to directly optimize low-level hardware ISP pipelines for an end-to-end domain-specific loss. The proposed hardware-in-the-loop optimization method revisits Covariance Matrix Adaptation Evolution Strategy (CMA-ES) [16]. We optimize ISP

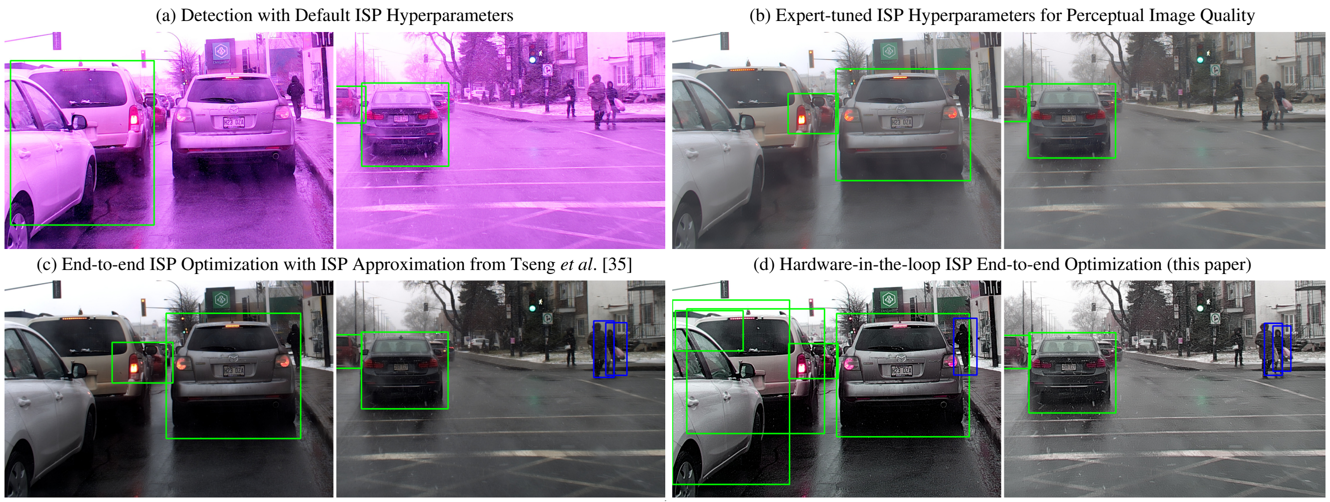
(a) Detection with Default ISP Hyperparameters
(b) Expert-tuned ISP Hyperparameters for Perceptual Image Quality
(c) End-to-end ISP Optimization with ISP Approximation from Tseng *et al.* [35]
(d) Hardware-in-the-loop ISP End-to-end Optimization (this paper)

Figure 1: Automotive object detection using [34] on real-world Sony IMX249 sensor captures processed with the ARM Mali-C71 hardware ISP, see text for details. The proposed hardware-in-the-loop end-to-end detection loss optimization approach (d) outperforms default ISP hyperparameters (a), ISP expert-tuned for perceptual image quality (b), and ISP optimized using a differentiable approximation [35] (c).

hyperparameters by solving a multi-objective black box optimization problem with a novel CMA-ES variant with max-rank-based multi-objective scalarization and initial search space reduction. The proposed method finds a set of best compromise solutions, *i.e.* Pareto front, over multiple objectives. We validate the proposed optimization method for a variety of applications including 2D object detection, segmentation and human viewing. For these applications, we demonstrate that the proposed hardware-in-the-loop method produces improved end-to-end losses compared to manual adjustment and existing approximation-based approaches.

Specifically, we make the following contributions:

- We propose an end-to-end hyperparameter optimization method for camera ISPs that integrates the hardware directly into the optimization loop.
- We propose a novel CMA-ES strategy for black box single and multi-objective ISP hyperparameter optimization robust with respect to parameter quantization and boundary effects.
- We validate end-to-end camera ISP optimization for 2D object detection, segmentation, and human viewing. For all applications, our approach outperforms existing state-of-the-art methods, including manual expert tuning and optimization via first-order approximations.

Limitations In this work, when considering scene understanding on ISP-processed images as downstream tasks, we assume that the hyperparameters and parameters of this higher-level block, *e.g.*, SIFT-based [27] or learned detector, have been fine-tuned to ISP outputs manually tuned for human viewing. Although additional fine-tuning of the downstream application module following ISP optimization might further improve the end-to-end performance, in this work, we assume the downstream block to be fixed. The challenging problem of joint optimization of hyperparameters and algorithm parameters along the full vision stack, including the image understanding module and potentially including optical and sensor design parameters, is an exciting area of future research that this work makes a first step towards.

2. Related Work

Optimization of Image Processing Pipelines Recently, several approaches [28, 30, 31, 38, 35] have explored the automatic optimization of image processing module hyperparameters. Note that ISP hyperparameter optimization should not be confused with adaptive capture control [12, 10, 24, 29, 37]. Adaptive capture control algorithms like auto-exposure (AE) extract statistics from the RAW image data to modify the capture process. We keep the camera control algorithms and their hyperparameters fixed.

Mittal *et al.* [28] propose to regress the noise standard deviation parameter of the BM3D denoiser using the Multi-Scale Structural Similarity index (MS-SSIM). Pfister *et al.* [31] propose to optimize sparsity regularization for denoising. Nishimura *et al.* [30] recently proposed a 0^{th}-order white box ISP optimization method using Nelder-Mead; their approach, however, can only be used to optimize a small number of hyperparameters, optimizes one ISP block at a time, and requires information about the internals of the ISP. Blockwise methods are not suitable for the joint optimization of multiple ISP modules, and white box approaches cannot be used to optimize black box proprietary ISPs.

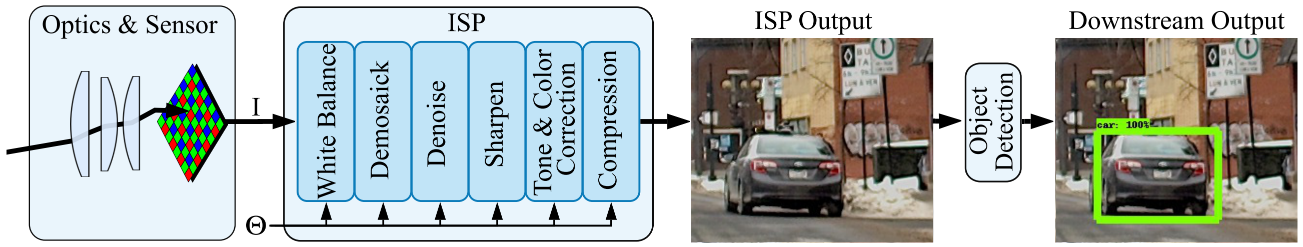

Figure 2: Typical hardware ISP architecture (similar to the automotive ARM Mali ISP). The ISP is a function mapping an input RAW measurement $\mathbf{I}$ to an output color image using a sequence of image reconstruction blocks. These blocks process the image depending on hyperparameters $\boldsymbol{\Theta}$. In this work, we optimize $\boldsymbol{\Theta}$ for an end-to-end objective, such as a detection loss for an object detector applied on the ISP output stream.

Hyperparameter Optimization of Vision Algorithms Making an image understanding module, such as a 2D object detector, robust with respect to changes in lighting and other environmental conditions is a challenging problem.

Bayesian optimization methods have been used for the optimization of dozens of hyperparameters for costly loss functions. Such methods have been used to optimize the hyperparameters of computer vision models [39, 1], building on existing hyperparameter optimization algorithms [2, 42, 3, 26]. Evolutionary strategies [8, 40] have also been proposed to optimize fast computer vision algorithms.

These works ignore the role of the ISP in the image formation process. The impact of ISP components in vision systems has been examined in [6, 35, 38]. For example, Buckler *et al*. [6] suggested that ISPs switch between a human-viewable mode and computer vision mode in order to produce perceptually lower-quality image-data only suitable for computer vision.

Manual ISP hyperparameter tuning for an image understanding downstream loss, for example intersection over union, is very challenging. To ease this difficult task, some camera and ISP manufacturers build simulation environments early within the design process [4]. Unfortunately, such simulated environments often have large domain gaps [20]. The proposed hardware-in-the-loop optimization method bridges domain gaps by processing RAW field data images with the actual hardware ISP.

Optimization Using Differentiable Approximations Recently Tseng *et al*. [35] proposed an automatic optimization method for black box ISPs based on learning a differentiable proxy that approximates the entire ISP as a RAW-to-output image transfer function. This is similar to the proposed method in that it allows the optimization of black box ISPs. In addition, the authors demonstrated that their approach can be applied to various downstream tasks, for example image processing for human viewing and image understanding. Unlike the proposed hardware-in-the-loop optimization method however, Tseng *et al*. [35] rely on an approximate model (trained on the hardware) instead of the hardware itself.

3. Image Formation Model

We present a model representative of the digital camera image formation process, from sensing to post-processing [5]:

1. *Optics and Sensor*: A lens or an assembly of lenses (*i.e*., optical system) focuses the scene radiance onto the sensor, which produces RAW pixel values by converting the photo-current measured in every pixel to discrete numbers using an analog-to-digital converter.
2. *White Balance*: The black level offset is subtracted. Defect pixels are detected and their pixel values corrected. Lens shading correction is performed. Devignetted pixel values are then gain-adjusted to match an estimated or preset white balance.
3. *Demosaicking*: RAW pixel values are typically captured using a color filter array sensor with a Bayer mosaic pattern. Trichromatic (*e.g*., RGB) pixel values are reconstructed from the mosaicked RAW image data.
4. *Denoising*: Pixel values are filtered to reduce noise, for example with edge-preserving filters or nonlocal patch matching.
5. *Sharpening*: Image details are boosted using unsharp masking or deconvolution.
6. *Color and Tone Correction*: Trichromatic pixel values are remapped based on the estimated illuminant. The image is tone mapped for contrast via global or local histogram manipulation and for display using gamma or sigmoid curve remapping.
7. *Compression*: Pixels values are converted to a specific color space (*e.g*., sRGB) and then compressed (*e.g*., JPEG).

Treating integer values as real numbers for simplicity, we model an ISP that reconstructs trichromatic color images $\mathbf{O}$ from RAW color filter array measurements $\mathbf{I}$ of size $\mathrm{W} \times \mathrm{H}$ as follows:

$$f : \mathbb{R}^{\mathrm{W}\times\mathrm{H}} \times \mathbb{R}^{\mathrm{P}}_{[0,1]} \to \mathbb{R}^{\mathrm{W}\times\mathrm{H}\times 3}, \quad (\mathbf{I}, \boldsymbol{\Theta}) \mapsto \mathbf{O}. \qquad (1)$$

The reconstruction is modulated by the values of P continuous hyperparameters $\boldsymbol{\Theta}$ with range of values normalized to the unit interval $\mathbb{R}_{[0,1]}$. Hyperparameters are generally discrete, each with its own operational range, for example $\{0,1\}$ for an algorithmic branch toggle and $\{0,\dots,2^{10}-1\}$ for a noise threshold [35]. We relax discrete parameters and map them to continuous parameters in the unit interval with an affine mapping (see Supplemental Material for details).

4. End-to-end Loss Functions

Existing end-to-end evaluation metrics include, for example, Mean-Average-Precision (mAP) and Mean-Average-Recall (mAR) for object detection, Panoptic Quality (PQ) for panoptic segmentation, and PSNR and CIELAB ΔC for human viewing. We use such metrics directly in the loss functions optimized by the proposed 0^{th}-order method. One or more metric might be used to optimize an ISP for a specific downstream task or an ensemble of potentially competing tasks. To this end, we pose ISP hyperparameter optimization as a Multi-Objective Optimization (MOO) [23] problem with optimal solutions

$$\boldsymbol{\Theta}^* = \underset{\boldsymbol{\Theta}\in\mathbb{R}^{\mathrm{P}}_{[0,1]}}{\operatorname{argmin}} \boldsymbol{\mathcal{L}}(\boldsymbol{\Theta}) := \underset{\boldsymbol{\Theta}\in\mathbb{R}^{\mathrm{P}}_{[0,1]}}{\operatorname{argmin}} (\mathcal{L}_1(\mathbf{s}(\boldsymbol{\Theta})),\dots,\mathcal{L}_{\mathrm{L}}(\mathbf{s}(\boldsymbol{\Theta}))) \quad (2)$$

where

$$\mathbf{s}(\boldsymbol{\Theta}) = (f(\mathbf{I}_1,\boldsymbol{\Theta}),\dots,f(\mathbf{I}_{\mathrm{S}},\boldsymbol{\Theta})) \quad (3)$$

is the *output image stack* consisting of a collection of images produced by the ISP using the same hyperparameter values but S different RAW image inputs from the *input image stack* $\mathbf{I}_1,\dots,\mathbf{I}_{\mathrm{S}}$. The *objective* is the vector $\boldsymbol{\mathcal{L}}(\boldsymbol{\Theta})$ of Eq. (2), where the L *end-to-end loss* vector components $\mathcal{L}_l(\mathbf{s}(\boldsymbol{\Theta}))$ are real-valued quality metrics measured on the output image stack. Each end-to-end loss component $\mathcal{L}_l(\mathbf{s}(\boldsymbol{\Theta}))$ assigns a score to an hyperparameter setting $\boldsymbol{\Theta}$ by evaluating the loss for images produced by the ISP modulated by $\boldsymbol{\Theta}$.

There may be more than one optimal solution $\boldsymbol{\Theta}^*$. First, the mapping between a hyperparameter setting $\boldsymbol{\Theta}$ and the output image stack $\mathbf{s}(\boldsymbol{\Theta})$ may have a nontrivial kernel, meaning that different hyperparameter settings may produce the exact same output images, and consequently the same losses. Such kernels should be distinguished by reducing the number of degrees of freedom of the hyperparameter search space: Ideally, $\mathbf{s}$ should be one-to-one, at least near hyperparameter settings that give near-optimal results. Second, MOO problems often have multiple solutions. For example, a first optimal solution may improve the value of $\mathcal{L}_1$ but make the value of $\mathcal{L}_2$ worse than another equally optimal solution. This manifests a different tradeoff between the conflicting loss functions. The set of optimal solutions to the MOO problem is called *Pareto front* [23].

We convert evaluation metrics into losses for which the value 0 means an optimal (or good enough) loss value. For example, if values of the evaluation metric $\mathcal{P}$ are to be treated as good enough if they fall within the range $[a,b]$, the corresponding loss can be defined as

$$\mathcal{L}(\mathbf{s}(\boldsymbol{\Theta})) = \begin{cases} 0 \text{ if } \mathcal{P}(\mathbf{s}(\boldsymbol{\Theta})) \in [a,b], & \text{and otherwise} \\ \min(|a-\mathcal{P}(\mathbf{s}(\boldsymbol{\Theta}))|\,,|\mathcal{P}(\mathbf{s}(\boldsymbol{\Theta}))-b|). \end{cases} \quad (4)$$

The proposed method facilitates weighting left distances differently than right distances and the use of other distance metrics than ℓ_1.

ISP hyperparameters typically affect the output image in strongly coupled, nonlinear, even discontinuous ways. As such, block coordinate-descent optimization is unfortunately prone to local minima as we demonstrate in this work. In addition, many important evaluation metrics respond nonlinearly or discontinuously to changes in their image inputs. As a result, $\mathcal{L}_l(\mathbf{s}(\boldsymbol{\Theta}))$ generally fails to have well-defined gradients, and a rugged search landscape, with local optima, outliers, discontinuities etc., is typical [35]. Moreover, ISP internals are often opaque, mandating black box optimization. Consequently, commonly used solvers are not adequate for the solution of MOO problem (2), see Supplemental Material.

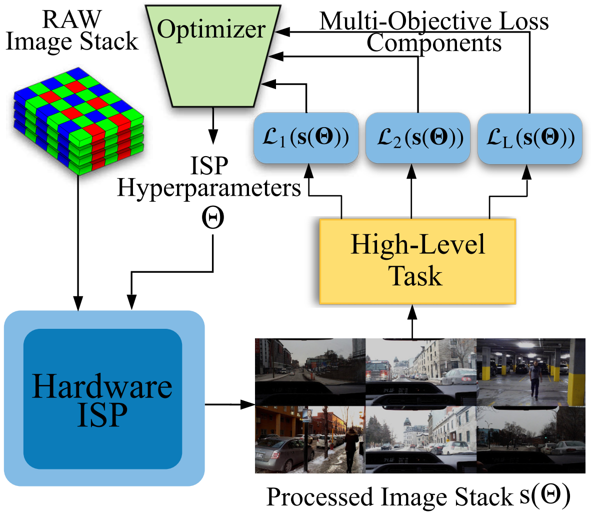
RAW
Image Stack
Optimizer
Multi-Objective Loss
Components
$\mathcal{L}_1(\mathbf{s}(\boldsymbol{\Theta}))$
$\mathcal{L}_2(\mathbf{s}(\boldsymbol{\Theta}))$
$\mathcal{L}_{\mathrm{L}}(\mathbf{s}(\boldsymbol{\Theta}))$
ISP
Hyperparameters
$\boldsymbol{\Theta}$
High-Level
Task
Hardware
ISP
Processed Image Stack $\mathbf{s}(\boldsymbol{\Theta})$

Figure 3: Illustration of the proposed hardware-in-the-loop method for multi-objective end-to-end ISP optimization.

5. Hardware-in-the-loop Optimization

The proposed hardware-in-the-loop ISP optimization method is illustrated in Fig. 3. Objective function evaluation begins by feeding a set of RAW images to the ISP, the output of which is then passed to the downstream task module. The output of the task module is evaluated by domain-specific evaluation metrics which are combined into the overall vectorial loss $\boldsymbol{\mathcal{L}}$. We propose a nonlinear non-convex black box optimizer that uses CMA-ES [16] strategies. Although CMA-ES is a 0^{th}-order stochastic evolutionary search method, it can be viewed as 2^{nd}-order since it estimates a covariance matrix closely related to the inverse Hessian [17, 15]. This feature allows CMA-ES to handle non-separable and badly conditioned problems, including (2). Specifically, we use the evolutionary Algorithm 1 with generation size λ and total number of iterations N_{iter}.

Dynamic Weighted Max-rank Loss Single-objective CMA-ES ranks each generation's trials based on a single loss function. The loss function exposed to CMA-ES, critical when performing MOO, is the *weighted max-rank loss*, a novel rank-based variant of weighted Chebyshev Scalarization [11]. It is computed as follows. Let a *trial* be a hyperparameter setting for which losses are known. First, the rank of each trial with respect to each of the corresponding losses among the trials of the current *and earlier* generations is computed. Then, the weighted max-rank loss associated with $\mathbf{\Theta}_j^{(t)}$ within generation t is

$$\mathcal{M}_j^{(t)} = \max_{l\in\{1,\ldots,\mathrm{L}\}} w_l^{(t)} \cdot \operatorname{rank}_{\left\{\mathcal{L}_l(\mathbf{\Theta}^{(0)}),\ldots,\mathcal{L}_l(\mathbf{\Theta}_\lambda^{(t)})\right\}}(\mathcal{L}_l(\mathbf{\Theta}_j^{(t)})).$$

(Losses are assumed known for the initial centroid $\mathbf{\Theta}^{(0)}$.) The nonnegative weight $w_l^{(t)}$, fixed within each generation, is used to modulate the importance of the loss function $\mathcal{L}_l$. The scalarization is adaptive: $w_l^{(t)}$ is adjusted at the end of each generation based on the proportion of trials in recent generations which have attained the target loss value 0.

Randomly drawn hyperparameter values that fall outside of the $\mathbb{R}_{[0,1]}$ search interval are mapped back in by reflection. In order to prevent an ISP hyperparameter from getting stuck at a fixed discrete value, Gaussian noise is added to each drawn hyperparameter value with a standard deviation chosen so that the proportion of trials without a bit change is approximately the proportion of trials with positive centroid ($\mathbf{\Theta}$) weights.

We rely on statistics generated by a tracking Monte Carlo simulation to estimate deviations from randomness within the results. CMA-ES parameters like σ, that affect the drawing of the trials of the next generation, are updated in this fashion. See Supplementary Material for details.

Warm-starting using Search Space Reduction Another critical component of the proposed optimization method is warm-starting using a novel search space reduction based on (approximately) Latin Hypercube sampling and the Kolgomorov-Smirnov independence test. The primary function of the search space reduction is to remap $\mathbb{R}_{[0,1]}^{\mathrm{P}}$ in such a way that hyperparameter ranges which impact positively the values of the losses occupy a larger volume within the search hypercube $\mathbb{R}_{[0,1]}^{\mathrm{P}}$. Search space reduction also provides an improved $\mathbf{\Theta}^{(0)}$ to Algorithm 1.

Search space reduction starts by sampling $\mathbb{R}_{[0,1]}^{\mathrm{P}}$. Trial coordinates are randomly generated by drawing from a mixture of Gaussian and uniform distributions. The hypercube sampling is consequently approximately Latin. Statistical

Algorithm 1 ISP Hyperparameter Optimization Method.

```
Require: $\mathbf{\Theta}^{(0)} \in \mathbb{R}_{[0,1]}^{\mathrm{P}}$, $\lambda \in \mathbb{N}^*$, $N_{\text{iter}} \in \mathbb{N}^*$
 1: initialize CMA-ES, $\mathbf{\Theta} \leftarrow \mathbf{\Theta}^{(0)}$, $t \leftarrow 1$
 2: while stopping criterion not satisfied & $t \leq N_{\text{iter}}$ do
 3:   for $j = 1$ to $\lambda$ do
 4:     $\mathbf{\Theta}_j^{(t)} \leftarrow$ randomly draw from Gaussian at $\mathbf{\Theta}$
 5:     Add random noise to $\mathbf{\Theta}_j^{(t)}$
 6:     Reflect $\mathbf{\Theta}_j^{(t)}$ back to $\mathbb{R}_{[0,1]}^{\mathrm{P}}$
 7:     $\mathbf{s}(\mathbf{\Theta}_j^{(t)}) \leftarrow$ run the ISP on $\mathbf{I}_1, \ldots, \mathbf{I}_{\mathrm{S}}$
 8:     $\left(\mathcal{L}_l(\mathbf{s}(\mathbf{\Theta}_j^{(t)})), \ldots, \mathcal{L}_{\mathrm{L}}(\mathbf{s}(\mathbf{\Theta}_j^{(t)}))\right) \leftarrow$ losses
 9:   end for
10:   $\mathbf{\Theta} \leftarrow$ update CMA-ES, $t \leftarrow t + 1$
11: end while
12: return $\mathbf{\Theta}$
```

analysis is then performed on the loss components of the trials, one hyperparameter at a time. The dependence between each parameter and loss component is quantified with the p-value of a Kolgomorov-Smirnov independence test with number of observations set to a small value. For each trial, the max-rank loss, over loss components impacted by the hyperparameter, is computed. The target interval for this hyperparameter is then taken to be the smallest interval that contains all the values of the hyperparameter for the $q\%$ trials with the best corresponding max-ranks. This is repeated multiple times, restricting the sampling of each hyperparameter to the previous stage's interval and centering its Gaussian component at the minimizer of the max-rank over the affected loss components. Before passing the result to CMA-ES, we construct a cubic spline for each hyperparameter so that the final interval occupies a large percentage of $\mathbb{R}_{[0,1]}$. This spline is used to remap the values of the hyperparameter. This separable procedure yields a bijective mapping of $\mathbb{R}_{[0,1]}^{\mathrm{P}}$ onto itself. Because the entire hypercube is still reachable, the proposed CMA-ES variant can recover from a tainted statistical analysis.

Max-rank Loss Initialization Following search space reduction, we compute the max-rank loss (over all loss components) of all trials with known losses, and we use the minimizer as initial hyperparameter setting $\mathbf{\Theta}^{(0)}$ for Algorithm 1.

Pareto Front Selection The very last $\mathbf{\Theta}$ may not be the very best one. At the end of Algorithm 1, a Pareto sort of all of the trials for which losses are known is performed, and the one encountered last is chosen as champion.

6. Assessment

In this section, three ISP-sensor combinations are used to validate the proposed end-to-end hardware-in-the-loop op-

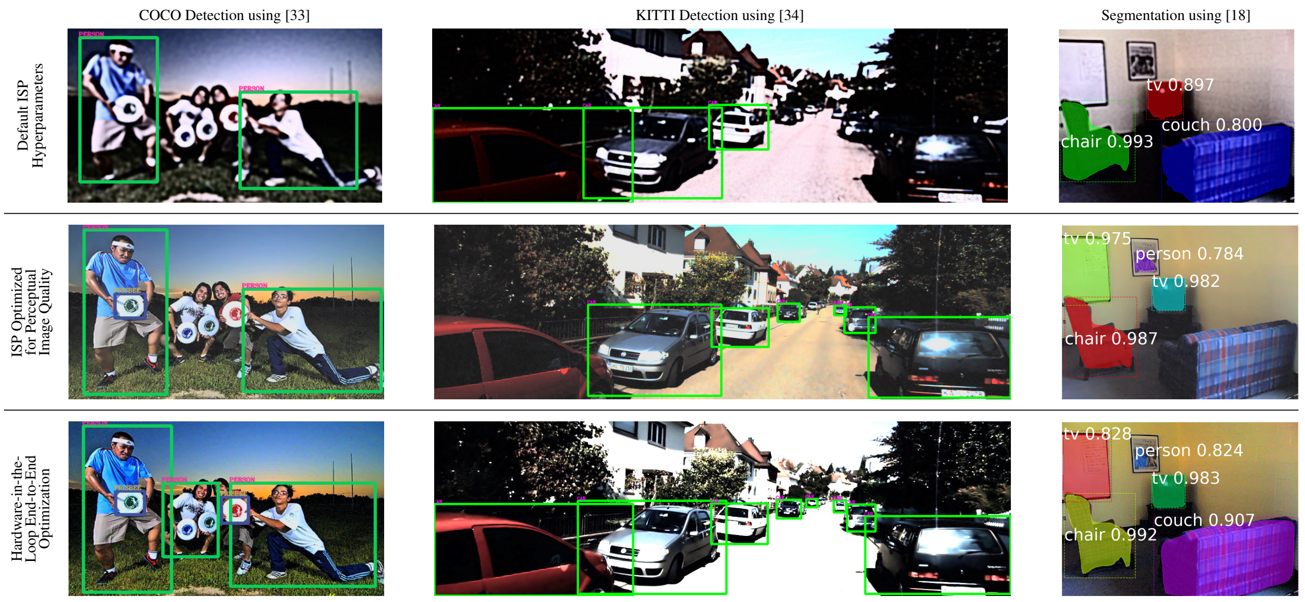
COCO Detection using [33]
KITTI Detection using [34]
Segmentation using [18]
Default ISP Hyperparameters
ISP Optimized for Perceptual Image Quality
Hardware-in-the-Loop End-to-End Optimization
tv 0.897
couch 0.800
chair 0.993
tv 0.975
person 0.784
tv 0.982
chair 0.987
tv 0.828
person 0.824
tv 0.983
couch 0.907
chair 0.992

Figure 4: Image understanding evaluation: From left to right, eighty-class object detection on COCO using [33], automotive object detection on KITTI using [34], and instance segmentation on COCO using [18]; from top to bottom, default ISP settings, ISPs expert-tuned for perceptual image quality, and task-specific-optimized ISPs, respectively. While the task-specific-optimized ISPs do not achieve the same perceptual quality as the expert-tuned ones, they substantially improve downstream image understanding tasks.

timization method for various downstream tasks:

- ARM Mali-C71 ISP and SONY IMX249 CMOS sensor, with 28 ISP hyperparameters,
- OnSemi AP0202AT ISP and AR0231AT CMOS sensor, with 12 ISP hyperparameters,
- A synthetic (simulated) ISP processing simulated RAW, with processing blocks shown in Fig. 2, with 14 ISP hyperparameters.

See the Supplemental Material for details. The performance of the proposed ISP optimization method is assessed for the following downstream tasks:

- Automotive 2D object detection using [34] with Resnet101 [19] backbone.
- Eighty-class 2D object detection using [33].
- Instance segmentation using [18].
- Panoptic segmentation with [18] for instance segmentation and [7] for semantic segmentation.
- Perceptual image quality for human viewing.

The training and evaluation of an image understanding algorithm generally require a large number of annotated images. Annotated RAW data is split into separate optimization and test sets. When using an existing dataset, *e.g.*, KITTI [13], the optimization set is taken from the training dataset and we leave the test set untouched. We have found that relatively small but representative optimization datasets, with

Table 1: Downstream applications and corresponding end-to-end evaluation metrics optimized in this assessment.

Task	**End-to-end Loss Component**
Object Detection	mAP and mAR [25]
Instance Segmentation	mAP [25]
Panoptic Segmentation	PQ [22]
Perceptual Image Quality	$5\hat{\mathcal{L}}_{\text{PERC}} + \ell_1$ [35]

about $\text{S} = 100$ images, are sufficient for ISP hyperparameter optimization. This is unsurprising given that ISPs have orders of magnitudes fewer parameters than higher-level neural networks.

Evaluation metrics are listed in Table 1: mean average precision with IoU 0.5 (mAP) [25], mean average recall for 10 detections per image (mAR) [25], Panoptic-Quality (PQ) [22] and, for human viewing Image Quality (IQ), the ℓ_1-norm of the difference between the ISP output and a reference image combined with the Perceptual image difference metric introduced in [41] and denoted by $\hat{\mathcal{L}}_{\text{PERC}}$.

6.1. ISP Optimization for Object Detection

Synthetic Evaluation Assessing methods using RAW data for an image understanding task requires a large amount of annotated training and test data. We first evaluate our approach with simulated RAW data using existing large sRGB datasets and a synthetic ISP. To this end, we processed sRGB images with RAW data simulation [21] and used a synthetic ISP that processes the RAW images, see Supplemental Material. We optimize the ISP for two different ob-

Table 2: Synthetic ISP optimization for domain-specific tasks.

Task: Eighty-class Object Detection using [33] and COCO [25]

Optimization Method	**mAP**	**mAR**
Default Parameters	0.15	0.13
Expert-tuned for Perceptual Image Quality	0.35	0.32
Proposed for Object Detection Loss	**0.39**	**0.36**
blockwise for Object Detection [30]	0.20	0.17

Task: Automotive Object Detection using [34] and KITTI [13]

Optimization Method	**mAP**	**mAR**
Default Parameters	0.14	0.10
Expert-tuned for Perceptual Image Quality	0.61	0.57
Proposed for Object Detection Loss	**0.75**	**0.71**

ject detection tasks as shown in Fig. 4: eighty-class object detection with the MS COCO dataset [25] using [33]; and two-class pedestrian-vehicle detection with the KITTI [13] dataset using [34]. Although expert-tuned ISP hyperparameters produce less noisy and perceptually better images (see Fig. 4), contrast around object boundaries is significantly higher with the proposed end-to-end optimization method, which leads to better performance for the downstream task. Table 2 shows a margin of 0.04 in both mAP and mAR for eighty-class object detection, and 0.14 in both mAP and mAR for automotive object detection, thus validating the proposed method.

Experimental Evaluation We next validate the proposed hardware-in-the-loop end-to-end optimization method with the ARM Mali-C71 ISP. We capture a set of RAW images of street scenes with a SONY IMX249 sensor. The experimental data contains annotated day and night scenes suitable for the evaluation of object detection models designed for automotive 2D object detection with [34] (see Supplemental Material). Quantitative evaluation results are shown in Table 3. The proposed method outperforms the ISP expert-tuned for perceptual image quality by margins of 0.30 mAP and 0.25 mAR. The ISP task-specific optimized using the proposed hardware-in-the-loop approach boosts local image gradients based on semantics, which results in improved detection, even though the image is more noise-contaminated (Fig. 1). This insight led us to manually boost local gradients to improve detection within a month-long manual tuning campaign led by an ISP expert. The resulting hyperparameters achieved a better performance than other methods but still fell short of the proposed optimization method by 0.12 mAP and 0.10 mAR.

6.2. ISP Optimization for Image Segmentation

Next, we demonstrate end-to-end ISP optimization for instance segmentation and panoptic segmentation [22] tasks. The evaluation is performed using the synthetic ISP and existing image segmentation datasets. We optimize the synthetic ISP using the simulated RAW COCO dataset to

Table 3: ARM Mali-C71 ISP optimization for automotive object detection using [34].

Optimization Method	**mAP**	**mAR**
Default Parameters	0.13	0.12
Expert-tuned for Perceptual Image Quality	0.14	0.13
Expert-tuned for Object Detection Loss	0.32	0.28
Tseng *et al.* [35] for IoU loss	0.26	0.23
Proposed for Object Detection Loss	**0.44**	**0.38**

Table 4: Evaluation for segmentation tasks on images generated using default ISP, ISP expert-tuned for perceptual image quality, and ISP optimized using the proposed method.

Optimization Method	**mAP**	**PQ**
	Instance [18]	**Panoptic** [7]
Default Parameters	0.12	0.15
Expert-tuned for Perceptual Image Quality	0.26	0.28
Proposed for Segmentation Loss	**0.32**	**0.35**

maximize mAP using [18] for instance segmentation. Fig. 4 shows examples of instance segmentation using three different ISP hyperparameter sets. For instance segmentation, the proposed end-to-end optimization achieves an improvement of 0.06 in mAP compared to the ISP expert-tuned for perceptual image quality. We perform a separate ISP optimization to minimize the PQ loss for panoptic segmentation. The proposed optimization for PQ loss achieves an improvement of 0.07 compared to the expert-tuned ISP for perceptual image quality (see Table 4).

6.3. ISP Optimization for Human Viewing

Perceptual image quality attributes [32], like sharpness, noise, detail, tone reproduction, contrast, and artifacts like zippering and staircasing impact human viewing applications. Like Tseng *et al.* [35], we optimize ISP hyperparameters for human viewing by minimizing the distance between the ISP output and a reference image, namely the central region of an aligned and resampled variant of the rainbow chart of [35]. Precise alignment of the reference and the ISP output is required. The reference is enhanced so that the optimized ISP reproduce the corresponding perceptual image quality attributes: It is tone mapped with a sigmoid-like curve to boost contrast and, at low gain, it is sharpened by unsharp masking to raise artificial acutance. The distance is computed with the perceptual loss $\mathcal{L}_{\text{PERC}} = 5\hat{\mathcal{L}}_{\text{PERC}} + \ell_1$, where $\hat{\mathcal{L}}_{\text{PERC}}$ is the AlexNet variant of the Perceptual metric [41] and the ℓ_1-norm of the difference is measured in normalized sRGB.

Captures of the rainbow chart were taken at two gains, high and low, and each corresponding $\mathcal{L}_{\text{PERC}}$ was fed to the optimizer (see Supplemental Material). Table 5 shows the values of three standard perceptual image quality metrics as measured on camera outputs of the rainbow chart using an expert-tuned ISP for perceptual image quality and an ISP optimized using the proposed method. Fig. 5 shows light

Table 5: Expert tuning and optimization of ARM Mali-C71 and OnSemi AP0202AT ISPs for perceptual image quality using Tseng *et al*. [35] and the proposed methods.

	ARM Mali-C71			**OnSemi AP0202AT**		
	Expert	Tseng *et al*. [35]	Proposed	Expert	Tseng *et al*. [35]	Proposed
SSIM	0.77	0.85	**0.89**	0.84	0.85	**0.86**
PSNR	16.50	19.53	**21.52**	17.97	17.67	**18.41**
CIELAB ΔE	13.81	10.54	**9.94**	10.58	10.40	**9.88**

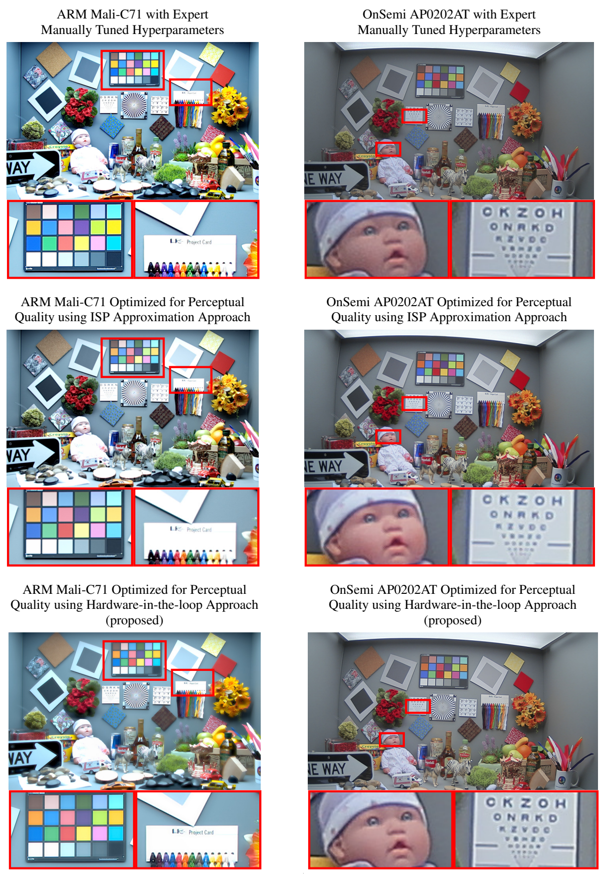
ARM Mali-C71 with Expert Manually Tuned Hyperparameters
OnSemi AP0202AT with Expert Manually Tuned Hyperparameters
ARM Mali-C71 Optimized for Perceptual Quality using ISP Approximation Approach
OnSemi AP0202AT Optimized for Perceptual Quality using ISP Approximation Approach
ARM Mali-C71 Optimized for Perceptual Quality using Hardware-in-the-loop Approach (proposed)
OnSemi AP0202AT Optimized for Perceptual Quality using Hardware-in-the-loop Approach (proposed)

Figure 5: Optimization of ARM Mali-C71 and OnSemi AP0202AT ISPs for perceptual image quality.

booth captures taken with the same hyperparameters. The results show that the proposed optimization method produces ISP hyperparameter settings with more detail, less noise, and better colors.

6.4. Blockwise ISP Optimization

Next, we compare the proposed method to the blockwise ISP optimization method of Nishimura *et al*. [30]. The authors optimize the blocks of an ISP sequentially, one group of hyperparameters at a time, a process that assumes ISP blocks to be functionally independent and separately accessible. Blockwise optimization is evaluated with the synthetic ISP and eighty-class object detection. For experiment details, see Supplemental Material. Table 2 shows that mAP and mAR scores are substantially lower with blockwise optimization than with the proposed joint hardware-in-the-loop optimization method. While blockwise ISP optimization [30] mitigates combinatorial explosion, this experiment demonstrates that ISP processing blocks are not necessarily independent and validates the proposed simultaneous optimization of all hyperparameters.

6.5. Optimization using ISP Approximations

We also compare the proposed method to the first-order ISP approximation method of Tseng *et al*. [35]. Fig. 5 shows images processed with the ARM Mali-C71 and OnSemi AP0202AT ISPs using hyperparameters tuned for perceptual image quality by an expert, optimized with the ISP approximation-based method, and optimized with the proposed hardware-in-the-loop method. We use the same end-to-end loss for the proposed method and Tseng *et al*. (Table 1). Table 5 shows that the proposed method outperforms Tseng *et al*. [35] in perceptual image quality. Next, we compare the two methods for the automotive 2D object detection task with the differentiable bounding-box regression and classification loss from [14]. As shown in Table 3, hardware-in-the-loop outperforms ISP-approximation by 0.18 mAP and 0.15 mAR. Qualitatively, the method of Tseng *et al*. [35] achieves this by finding a hyperparameter set that boosts local gradients (Fig. 1); the proposed method finds a hyperparameter set that further increases them.

7. Conclusions

We present a hardware-in-the-loop approach for optimizing black box ISPs. Specifically, we formulate hardware ISP hyperparameter tuning as a multi-objective nonlinear optimization problem which we solve using CMA-ES strategies with a max-rank-based scalarization and initial search space reduction. We validate the proposed method for end-to-end camera ISP optimization on 2D object detection, segmentation, and human viewing as task domains. For all applications, the proposed method outperforms existing state-of-the-art approaches, including expert manual tuning and first-order approximations. The proposed method optimizes the hyperparameters of ISP blocks not in isolation but within the full image processing stack, with the hardware in the loop. Exciting directions for future research include design parameters of the optics and sensor in the end-to-end optimization, or searching the design space of the ISP algorithms and downstream network architecture themselves, potentially allowing for joint image processing, sensor, optics and downstream network architecture design.

8. Acknowledgments

The authors thank Amrit Krishnan, Dong-eun Seo, Federico Ariza, Karl St-Arnaud and Luis Garcia for many fruitful discussions and assistance with the experiments.

References

[1] Rémi Bardenet, Mátyás Brendel, Balázs Kégl, and Michele Sebag. Collaborative hyperparameter tuning. In *International Conference on Machine Learning (ICML)*, pages 199–207, 2013. 3

[2] James Bergstra, Daniel Yamins, and David Daniel Cox. Making a science of model search: Hyperparameter optimization in hundreds of dimensions for vision architectures. In *International Conference on Machine Learning (ICML)*, 2013. 3

[3] James S. Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems*, pages 2546–2554, 2011. 3

[4] Henryk Blasinski, Joyce Farrell, Trisha Lian, Zhenyi Liu, and Brian Wandell. Optimizing image acquisition systems for autonomous driving. *Electronic Imaging*, 2018(5):1–7, 2018. 3

[5] Michael S. Brown. Understanding the in-camera image processing pipeline for computer vision. *IEEE International Conference on Computer Vision (ICCV) - Tutorial*, 2019. 3

[6] Mark Buckler, Suren Jayasuriya, and Adrian Sampson. Reconfiguring the imaging pipeline for computer vision. In *IEEE International Conference on Computer Vision (ICCV)*, pages 975–984, 2017. 3

[7] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2017. 6, 7

[8] Ernest C. H. Cheung, Jiali Wong, Joceyln Chan, and Jia Pan. Optimization-based automatic parameter tuning for stereo vision. In *IEEE International Conference on Automation Science and Engineering (CASE)*, pages 855–861, 2015. 3

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

[10] Jingming Dong, Iuri Frosio, and Jan Kautz. Learning adaptive parameter tuning for image processing. *Electronic Imaging*, 2018(13):1–8, 2018. 2

[11] Michael T. Emmerich and André H. Deutz. A tutorial on multiobjective optimization: Fundamentals and evolutionary methods. 17(3):585–609, Sept. 2018. 5

[12] Eduardo S. L. Gastal and Manuel M. Oliveira. Adaptive manifolds for real-time high-dimensional filtering. *ACM Transactions on Graphics (SIGGRAPH)*, 31(4):33, 2012. 2

[13] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the KITTI vision benchmark suite. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3354–3361, 2012. 6, 7

[14] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 580–587, 2014. 8

[15] Nikolaus Hansen. Benchmarking a BI-population CMA-ES on the BBOB-2009 function testbed. In *Workshop Proceedings of the GECCO Genetic and Evolutionary Computation Conference*, pages 2389–2395, July 2009. 5

[16] Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *IEEE International Conference on Evolutionary Computation*, pages 312–317, 1996. 1, 4

[17] Nikolaus Hansen and Andreas Ostermeier. Completely derandomized self-adaptation in evolution strategies. *Evolutionary Computation*, 9(2):159–195, 2001. 5

[18] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask R-CNN. In *IEEE International Conference on Computer Vision (CVPR)*, pages 2961–2969, 2017. 6, 7

[19] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. 6

[20] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *arXiv preprint arXiv:1711.03213*, 2017. 3

[21] 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. 6

[22] Alexander Kirillov, Kaiming He, Ross Girshick, Carsten Rother, and Piotr Dollár. Panoptic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9404–9413, 2019. 6, 7

[23] Marco Laumanns, Lothar Thiele, Kalyanmoy Deb, and Eckart Zitzler. Combining convergence and diversity in evolutionary multiobjective optimization. *Evolutionary Computation*, 10(3):263–282, 2002. 4

[24] Anat Levin, Boaz Nadler, Fredo Durand, and William T Freeman. Patch complexity, finite pixel correlations and optimal denoising. In *European Conference on Computer Vision (ECCV)*, pages 73–86, 2012. 2

[25] 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)*, pages 740–755, 2014. 6, 7

[26] Ilya Loshchilov and Frank Hutter. CMA-ES for hyperparameter optimization of deep neural networks. *arXiv preprint arXiv:1604.07269*, 2016. 3

[27] David G. Lowe. Object recognition from local scale-invariant features. In *IEEE International Conference on Computer Vision (ICCV)*, volume 2, pages 1150–1157, 1999. 2

[28] Anish Mittal, Anush K Moorthy, and Alan C Bovik. Automatic parameter prediction for image denoising algorithms using perceptual quality features. In *Human Vision and Electronic Imaging XVII*, volume 8291, page 82910G, 2012. 2

[29] Shree K. Nayar and Vlad Branzoi. Adaptive dynamic range imaging: Optical control of pixel exposures over space and

time. In *IEEE International Conference on Computer Vision (ICCV)*, page 1168, 2003. 2

[30] Jun Nishimura, Timo Gerasimow, Rao Sushma, Aleksandar Sutic, Chyuan-Tyng Wu, and Gilad Michael. Automatic ISP image quality tuning using nonlinear optimization. In *IEEE International Conference on Image Processing (ICIP)*, pages 2471–2475, 2018. 1, 2, 7, 8

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

[32] Jonathan B. Phillips and Henrik Eliasson. *Camera Image Quality Benchmarking*. Wiley Publishing, 1st edition, 2018. 7

[33] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. 6, 7

[34] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *Advances in Neural Information Processing Systems*, pages 91–99, 2015. 2, 6, 7

[35] 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 (SIGGRAPH)*, 38(4):27, 2019. 1, 2, 3, 4, 6, 7, 8

[36] 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 (ICIP)*, pages 4624–4628, 2019. 1

[37] Qi Xie, Qian Zhao, Deyu Meng, Zongben Xu, Shuhang Gu, Wangmeng Zuo, and Lei Zhang. Multispectral images denoising by intrinsic tensor sparsity regularization. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1692–1700, 2016. 2

[38] 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, 3

[39] Dani Yogatama and Gideon Mann. Efficient transfer learning method for automatic hyperparameter tuning. In *International Conference on Artificial Intelligence and Statistics*, pages 1077–1085, 2014. 3

[40] Steven R. Young, Derek C. Rose, Thomas P. Karnowski, Seung-Hwan Lim, and Robert M. Patton. Optimizing deep learning hyper-parameters through an evolutionary algorithm. In *Workshop on Machine Learning in High-Performance Computing Environments*, pages 1–5, 2015. 3

[41] Richard Zhang, Phillip Isola, Alexei A. Efros, Eli Shechtman, and Oliver Wang. The unreasonable effectiveness of deep features as a perceptual metric. *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 586–595, 2018. 6, 7

[42] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 8697–8710, 2018. 3