TaCOS: Task-Specific Camera Optimization with Simulation

Chengyang Yan Donald G. Dansereau
Australian Centre for Robotics, School of Aerospace, Mechanical and Mechatronic Engineering,
The University of Sydney
chengyang.yan, donald.dansereau@sydney.edu.au

Abstract

The performance of perception tasks is heavily influenced by imaging systems. However, designing cameras with high task performance is costly, requiring extensive camera knowledge and experimentation with physical hardware. Additionally, cameras and perception tasks are mostly designed in isolation, whereas recent methods that jointly design cameras and tasks have shown improved performance. Therefore, we present a novel end-to-end optimization approach that co-designs cameras with specific vision tasks. This method combines derivative-free and gradient-based optimizers to support both continuous and discrete camera parameters within manufacturing constraints. We leverage recent computer graphics techniques and physical camera characteristics to simulate the cameras in virtual environments, making the design process cost-effective. We validate our simulations against physical cameras and provide a procedurally generated virtual environment. Our experiments demonstrate that our method designs cameras that outperform common off-the-shelf options, and more efficiently compared to the state-of-the-art approach, requiring only 2 minutes to design a camera on an example experiment compared with 67 minutes for the competing method. Designed to support the development of cameras under manufacturing constraints, multiple cameras, and unconventional cameras, we believe this approach can advance the fully automated design of cameras. Code is available on our project page at https://roboticimaging.org/Projects/TaCOS/.

1. Introduction

The quality of camera captures directly affects perception tasks. An analogy in nature is how animals' visual perceptions impact their daily activities. It is believed that evolution designs distinct visual systems for different species to suit their habitats [30], suggesting a need for a sophisticated and application-based camera design approach.

Currently, designing cameras is cumbersome, typically requiring professionals to devise designs based on their experiences, followed by extensive experiments with various design decisions. Moreover, the camera hardware is often designed in isolation from perception tasks, despite literature showing the benefits of joint design.

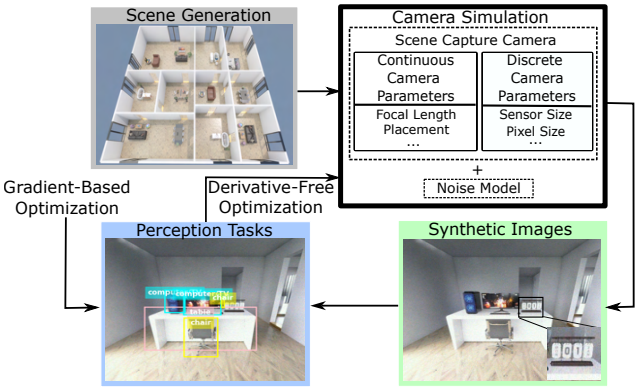

Figure 1. Our method combines a derivative-free optimizer and gradient-based optimizer to co-design the camera with perception tasks in simulation, which utilizes ray tracing and a physics-based noise model. Our approach supports optimising discrete and continuous camera parameters for manufacture constraints and the generalization to other camera design problems.

Existing methods optimize imaging systems for tasks through software, often using ray tracing renderers to simulate the physical image formation process. Ray tracing render has been widely used for designing and evaluating cameras for autonomous driving [5, 32–34, 58]. However, manual tuning of camera parameters for optimization is still required, and joint optimization of the cameras and the tasks is not addressed in these methods.

Other works propose propose automatic co-design of imaging systems and perception tasks using differentiable ray tracing or proxy neural networks [2, 6, 7, 9, 11, 25, 36, 39, 40, 46, 48, 50, 52, 53, 59, 60, 62]. Nevertheless, these methods focus primarily on optics design since their design space is restricted by using captured image datasets which prevents them from generalizing to more complex camera

design problems involving field-of-view (FOV), resolution, multi-cameras, or unconventional cameras.

Game engines like Unreal Engine (UE) [13] are repurposed for simulating cameras due to their ray tracing support and ability to produce video sequences and interact with virtual environments. Klinghoffer et al. [29] propose a reinforcement learning (RL) method for camera design evaluated with a UE-based simulator. While this method achieves high performance, it optimizes a complex neural network to design cameras, making it less efficient than directly optimizing camera parameters.

As illustrated in Fig. 1, we introduce an end-to-end camera design method that directly optimizes camera parameters for perception tasks. We propose a camera simulator that allows tuning various parameters and addresses image signal-to-noise ratio (SNR) with a physics-based noise model [18]. Additionally, we use a procedural generation algorithm to create indoor virtual environments in UE 5 with machine learning labels for our method.

Inspired by evolution, we employ a genetic algorithm [22], a derivative-free optimizer, to optimize discrete, continuous, and categorical variables, which supports manufacturing constraints and selecting parts (optics, image sensors, etc.) from catalogs where custom manufacture is infeasible. Finally, we adopt a quantized continuous approach [9] for discrete variables, considering their interdependencies.

We implement our approach on two design problems, demonstrating our method's efficiency compared to the state-of-the-art (SOTA) design method, and cameras designed by our approach achieve compelling performance compared to high-quality robotic/machine vision cameras. We also validate our simulation's accuracy by comparing it with physical cameras. In summary, our contributions are:

- We introduce an end-to-end camera design method that combines derivative-free and gradient-based optimization to automatically co-design cameras with perception tasks, allowing continuous, discrete, and categorical camera variables.
- We develop a camera simulation with a physics-based noise model and a virtual environment, and provide a procedurally generated virtual environment.
- We validate our simulator by establishing equivalence in both low-level image statistics and high-level task performance between synthetic and captured imagery.
- We validate our method with improved performance compared to other methods and off-the-shelf options.

This work is a key step in simplifying the process of designing cameras for autonomous systems, emphasizing task performance and manufacturability constraints.

Limitations Our work focuses exclusively on camera parameters set during the manufacturing stage. Parameters that can be dynamically adjusted during camera operation, such as exposure settings, will be addressed in future work. Nevertheless, our method can still select algorithms that control these dynamic parameters dynamically.

2. Related Work

Task-Specific Simulation-Based Camera Design Designing cameras tailored for vision tasks using simulation has gained popularity. Blasinski et al. [5] propose optimizing a camera design to detect vehicles. They used synthetic data generated with ISET [55, 56]. The method optimizes cameras by experimentally analyzing the impact of image postprocessing pipelines and auto-exposure algorithms on object detection tasks. This work continues in [32–34, 58] with larger datasets and an optimization framework for high dynamic range (HDR) imaging. Nevertheless, these methods require manual tuning and testing of camera parameters, whereas our method is an automatic end-to-end approach.

Other works have proposed end-to-end methods to optimize the imaging system based on tasks. Many use gradient-based optimization with differentiable camera simulators, including differentiable ray tracing and proxy neural networks for non-differentiable image formation processes. In these works, prerecorded images are used as input scenes to their pipeline, and their simulators convert the input images to images formed by their proposed cameras. Applications include extended depth of field (DOF) [48, 50, 59], depth estimation [2, 7, 20, 25], object detection [9, 46, 52, 53], HDR imaging [39], [36], image classification [6, 11, 60, 62], and motion deblurring [40]. However, these methods use precaptured images so that their camera simulation is restricted to the domains of the data. Key camera design decisions such as the FOV, resolution, use of multiple cameras, and the design of unconventional cameras (light field, etc.), are not addressed. Our method establishes a virtual environment to support the simulation and optimization of a much broader range of camera designs.

RL has also been explored for end-to-end optimization of imaging systems. Klinghoffer et al. [29] uses RL to train a camera designer, encompassing various camera parameters using the CARLA Simulator [12]. Hou et al. [24] introduce another RL-based approach for pedestrian detection. Although RL demonstrates impressive results in camera design, it involves optimizing complex neural networks that learn to design cameras, demanding more training data and time since the neural network contains a larger number of parameters that need to be optimized. In contrast, our approach directly optimizes the camera's parameters, yielding competitive results with much less computation.

Genetic Algorithms for Camera Design Genetic algorithms [22] are widely applied across many domains thanks to their flexibility, robustness, and ability to explore complex search spaces including continuous, discrete, and cat-

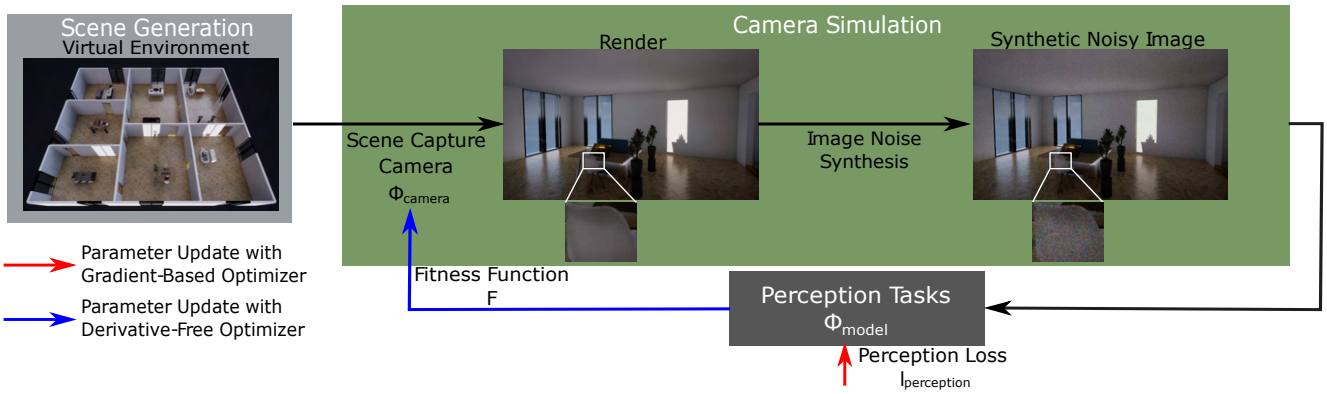

Figure 2. We establish a virtual environment and capture scene renders using a ray-traced scene capture camera. We then add physics-based, sensor-specific noise to the renders and input them into perception tasks for evaluation. In our optimization process, we jointly optimize the camera parameters Φ_{camera} using a fitness function F with a derivative-free optimizer (blue arrow), as well as the parameters of perception tasks Φ_{model} (if trainable) on their corresponding loss function $l_{perception}$ with gradient-based optimizers (red arrow).

egorical variables. For imaging systems design, existing works have applied genetic algorithms for optimizing camera placements in a network of cameras [1, 21, 41], and for optimizing optics design to improve image quality [4, 14, 15, 42, 51, 54, 61], a review can be found in [23]. To our knowledge, none of these existing methods combines genetic algorithms with gradient-based optimizers to co-design the camera hardware and perception tasks for automatic camera design and improved performance, which is the primary contribution of our work.

3. Method

Our method (Fig. 2) uses a simulated camera to capture a virtual environment and apply a physics-based noise model. We then evaluate the resulting images in perception tasks and use this data to optimize the camera design. Our method outputs the designed camera parameters and trained models for perception tasks if the task is trainable.

3.1. Simulation Environment

The simulation environment should be photorealistic to minimize the sim-to-real gap. It should support rendering with various camera parameters and illuminations to maximize the design space, as well as the deployment of multiple cameras for unconventional imaging systems.

Considering the desired features of the simulation environment, we chose to use UE for this work. UE utilizes real-time hardware ray tracing combined with software ray tracing for global illumination and reflection, aligning with the physics of light and providing realistic shadowing, ambient occlusion, illumination, reflections, etc. [13]. UE supports the alteration of various camera parameters, detailed in Sec. 3.2, and the application of multiple cameras. Nevertheless, our method is not limited to UE, other simulators with the desired features or suited to the downstream tasks can also be used to build the simulation environment.

We deploy cameras on an auto-agent simulating the platform that uses the camera. The agent navigates the virtual environment autonomously, enabling a fully automated design process. The cameras capture scene renders as the agent moves, which are then used in downstream processes.

To demonstrate our method, we implemented the procedural generation of random indoor virtual environments and their associated semantic labels with UE 5, with support for application-specific objects. The implementation details of the environment are further explained in Sec. 4.3.

3.2. Camera Simulation

Our camera simulation includes a scene capture component from the simulation environment and an image noise synthesis component to augment the scene captures to enhance the realism of our simulation.

Scene Capture The scene capture component captures scene irradiance from the virtual environment. We employ the camera in UE that captures the scene renders, which allows the configuration of parameters associated with camera placement (location, orientation), optics (focal length and aperture), the image sensor (width, height, and pixel count), exposure settings (shutter speed and ISO), and multi-camera designs (number of cameras and their poses). It also allows the configuration of algorithms in the image processing pipeline (auto-exposure, white balancing, tone mapping, color correction, gamma correction, and compression) and includes image effects like motion blur.

We experimentally validate our method using parameters supported by UE. Additional parameters like geometric distortion and defocus blur could be added by augmenting the renderer. The noise synthesis model described below serves as an example of such augmentation.

Noise Synthesis Image noise is a fundamental limiting factor for many vision tasks that is tightly coupled to camera parameters such as pixel size and exposure settings. As UE lacks a realistic noise model, we incorporate a post-render image augmentation that introduces noise. We employ thermal and signal-dependent Poisson noise following the affine noise model [18]. The noise model is calibrated with a physical camera following established methods [31, 43, 57] and then generalized for different exposure time and gain settings. The generalized form of the noise model defines the variation of intensity at a pixel as

$$\sigma^2 = \frac{G}{G_0}\sigma_p^2\bar{I} + \frac{G^2}{G_0^2}\sigma_r^2, \tag{1}$$

where σ_p and σ_r are photon and thermal noise respectively, G and G_0 are the new gain and calibrated gain, and $\bar{I}$ is the measured intensity. We detail the noise model calibration and generalization method in the supplementary material.

To generalize the noise model to different image sensors, we consider the ratio of their pixel sizes. For the same illumination, larger pixel sizes capture more photons, resulting in a higher measured intensity level, which is readily reflected by adjusting the gain in Eq. 1 inversely proportional to the pixel area. While we employ these observations to generalise noise characterisations, it is also possible to directly characterize multiple sensors and directly use these characterizations for more accurate noise characteristics.

3.3. Optimization

Optimizers We employ a derivative-free optimizer to optimize the camera parameters as many simulators and tasks used for designing cameras are non-differentiable, and numerous camera parameters are discrete or categorical, which makes gradient-based optimizers, such as gradient-descent, and gradient estimation based method, such as surrogate gradient and finite differences, inapplicable. Hence, we utilize the genetic algorithm [22] as the derivative-free optimizer, although other derivative-free optimizers can also be integrated into the proposed pipeline.

To enhance the performance of perception tasks, we jointly optimize the tasks (if applicable) along with the design of the camera hardware. The applicable tasks are the ones that involve machine learning models and are optimized using their corresponding optimizers. Given that neural networks are commonly used in SOTA methods for these tasks, gradient-based optimizers such as Stochastic Gradient Descent [45] or Adam [28] are typically employed. Therefore, our method simultaneously optimizes the camera hardware with a derivative-free optimizer and the trainable perception tasks with gradient-based optimizers. However, perception tasks that are not trainable, such as extracting Oriented FAST and Rotated BRIEF (ORB) features [47], are not jointly optimized.

Camera Parameters Our optimizer can handle the optimization of all parameters captured in the camera simulation, e.g. those outlined in Sec. 3.2, which can be both continuous and discrete. For instance, parameters related to optics and image sensors can be optimized as continuous variables if there are no manufacturing constraints on new optics/sensors. Alternatively, they can be selected from existing lens/sensor catalogs, allowing manufacturing and availability to be considered.

Discrete Variable Optimization Our approach offers two schemes to optimize the discrete camera variables: *fully discrete* and *quantized continuous*. In the fully discrete scheme, we optimize the parameter x, representing the discrete parameter in its available values, by constraining the mutation stage of the genetic algorithm to ensure that only available values of this variable are used.

In the quantized continuous scheme, we adopt the "quantized continuous variables" method introduced in [9]. Here, in each iteration, the discrete parameter x can freely change as a continuous variable from its current best value obtained in the previous iteration. However, it is then replaced with the closest value from its available range:

$$x^* = \arg\min_k ||x - x_k||_2^2, \tag{2}$$

where x^* is the parameter retained from its available range, x is the variable obtained from the optimization process, and x_k represents the k-th parameter in its available values.

A limitation of the fully discrete scheme arises when a variable x encompasses multiple interconnected parameters. For instance, if x represents available image sensors, it includes parameters such as width, height, and pixel size. Selecting x categorically does not leverage the relationships between these parameters. The quantized continuous scheme addresses this by optimizing all parameters within x freely and then replaced by it with the closest categorical x. Therefore, the fully discrete scheme is suitable for independent parameters like the number of cameras, while the quantized continuous scheme is beneficial for parameters with interdependencies like image sensors.

Fitness Function The fitness function is constructed based on the performance of tasks, which is used to optimize the camera parameters using the derivative-free optimizer. We demonstrate the construction of the fitness function incorporating various perception tasks in our experiments.

4. Experiments

We apply our proposed method to two design problems for demonstration: designing a stereo camera with two components for depth estimation on a vehicle and designing a monocular camera for multiple tasks on a Mixed Reality (MR) device. Additionally, we validate our camera simula-

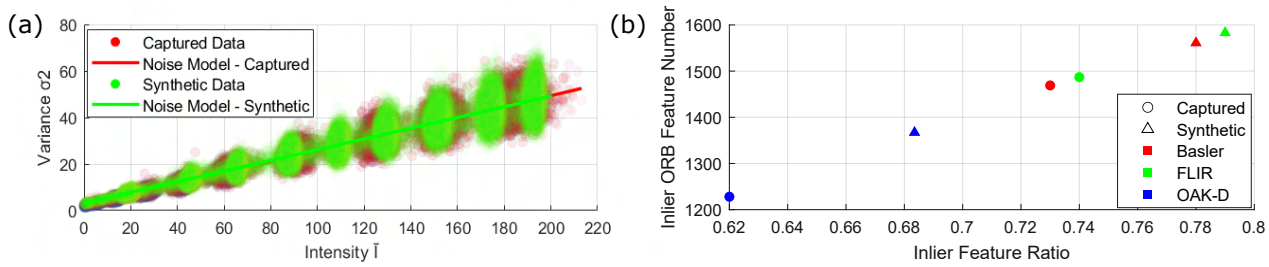

Figure 3. Comparison of captured and synthetic images in terms of (a) variance in pixel intensities and (b) perception task performance. In (a), despite differences in color intensities due to manufacturing variations of the test target, the variances of pixel values in synthetic images match those in captured images, validating the accuracy of our noise model. In (b), the ranking of camera performance in our simulation aligns with physical cameras, and the differences in their performance between captured and synthetic images are consistent.

tion approach by comparing it with physical cameras. Additional details and results are provided in the supplement.

Assumptions (1) Objects' distances to the camera exceed the camera's hyperfocal distance so that the DOF is safely neglected. (2) The lens of our camera is free of geometric distortion and chromatic aberration.

Noise Synthesis We apply the affine noise model calibrated using a FLIR Flea3 Camera [17] with a Sony IMX172 sensor to all synthetic images, except the simulations of off-the-shelf cameras used in Sec. 4.1 and 4.3, which are calibrated using their corresponding noise model.

4.1. Simulator Validation

To validate our simulator, we compare synthetic images from our simulator with those captured by physical cameras in terms of image statistics and task performance.

Image Statistics We use a test target with linear greyscale colorbars in a controlled illumination environment, capturing images with the FLIR Flea3 Camera [17]. The same test target is then recreated in our simulator, maintaining consistent illumination, camera position, and parameters. Fig. 3(a) shows intensity variances versus pixel intensities for both captured and synthetic images, with good alignment validating the accuracy of our noise model. The slight differences in mean intensity for each greyscale bar are caused by printer variations during manufacturing.

Perception Task Performance We further validate the cameras' performance on perception tasks in our simulation, which serves as the evaluation method in Sec. 4.3. In this experiment, we focus on a feature extraction task using the widely applied ORB feature extractor [47].

We use a test target from [10] with features of varying scales and depths, set against a texture-less background. Ten frames are captured with constant translational motion, ensuring consistent illumination, camera parameters, and relative positioning between real and virtual experiments. Fig. 3 (b) shows the number of inlier features (correctly matched points across consecutive frames) along with the ratio of inliers to total features (inliers plus outliers).

The comparison includes three robotic/machine vision cameras: the RGB camera of the Luxonis OAK-D Pro Wide [35], the FLIR Flea3 [17], and the Basler Dart DaA1280-54uc [3]. Specific noise models for these cameras, calibrated as described in Sec. 3.2, are applied in our simulator. The feature extraction task serves as a reliable measure of performance consistency between physical and simulated environments.

Fig. 3 (b) shows that synthetic images from our simulator yield more inlier features and a higher inlier ratio due to quality reduction in the physical test target from the manufacturing process. However, the relative performance of the cameras, crucial for optimization, remains consistent between simulated and physical settings. This validates that our simulations accurately reflect real-world camera performance and effectively evaluate camera designs.

4.2. Stereo Camera on Autonomous Vehicle

We apply our method to design a stereo camera on an autonomous vehicle for the task of depth estimation.

Environment and Data This experiment is conducted in CARLA [12], based on UE 4. The stereo camera is mounted to a car that moves automatically. The environment is configured with constant illumination, and motion blur enabled due to the moving platform. Images are captured during both training and testing. In training, each camera configuration captures one image per step, totalling 1000 images over 1000 steps. For testing, each camera design captures 1500 images to evaluate performance. Training and testing are performed on different urban outdoor maps.

Design Space We optimize the baseline ($b \in [0.01\text{m}, 3\text{m}]$) and horizontal FOV ($fov \in [50°, 120°]$) as continuous variables. We use two cameras with fixed sensor sizes of 1.536 mm$\times$0.768 mm, a pixel size of 1.55 μm, and a mounting height of 2 m, positioned forward-facing with no slant.

Stereo Matching Network We use the PSMNet [8] to pre-

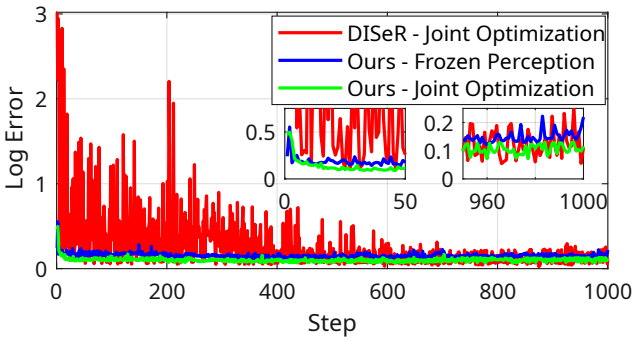

Figure 4. Training curves comparing the design of cameras using our method with and without joint optimization, and curve of DISeR [29], are plotted. Zoomed-in windows from 0 to 50 and 950 to 1000 timesteps are provided for visualization. The curves demonstrate that our method with joint optimization achieves superior task performance with fewer steps and smoother behaviour.

dict the disparity map, which is pretrained on the KITTI Stereo 2015 dataset [37, 38] and subsequently fine-tuned with the optimization of camera parameters using its loss function ($l_{disparity}$, Smooth L1 Loss), and the maximum disparity for this model is capped as 192 following [8]. We use an Adam optimizer [28] with a batch size of 4 and a learning rate of $1 \cdot 10^{-3}$ to train the network.

Fitness Function We use the inverse of the log error between the predicted and ground-truth disparity maps (normalized by baselines), which emphasizes the depth prediction accuracy in both large and small distances.

Derivative-Free Optimization The genetic algorithm uses 5 solutions per generation, retaining the top 2 for the next iteration. The top 3 generate offspring via uniform crossover, followed by mutation with a random factor (0.8 to 1.2) and an addition value (-5 to 5 for FOV, -0.2 to 0.2 for baseline). Hyperparameters are empirically chosen, and optimization parameters are randomly initialized. See the supplement for comparisons of hyperparameters and initialization.

Results Tab. 1 details the performance of stereo cameras designed by our method, compared with two off-the-shelf models, the Intel RealSense D450 [26] and ZED 2i [49], as well as a camera designed with the RL method DISeR [29]. We also compare results from jointly optimizing camera design and perception tasks against optimizing camera design alone while fixing perception model parameters (pretrained with 2500 images from CARLA using 50 camera configurations). ● indicates optimized parameters and ● indicates fixed parameters. Performance is evaluated using Average Log Error and Root Mean Square Error (RMSE) between estimated and ground-truth depths in meters.

Our camera design and DISeR perform best in terms of log error, which evaluates depth estimation accuracy across both short and long distances, and RMSE, which favors

Table 1. Depth estimation performance of stereo cameras designed using our method, the RL method (DISeR [29]), and two off-the-shelf cameras. The proposed method incorporating joint optimization of camera parameters and perception tasks shows similar performance as DISeR at a fraction of the training time.

Camera	Camera Parameters		Performance	
	Baseline b (m)	Horizontal FOV fov (°)	Log Error ↓	RMSE ↓
RealSense [26]	0.095 ●	87 ●	0.39 ●	178.94 ●
ZED [49]	0.12 ●	72 ●	0.29 ●	134.74 ●
DISeR [29]	1.84 ●	50 ●	0.16 ●	**75.05** ●
Ours - Frozen	1.41 ●	50 ●	0.19 ●	111 ●
Ours - Joint	1.6 ●	50 ●	**0.14** ●	79.81 ●

larger baselines for more accurate long-distance estimation as long-distance has a greater impact on this value. However, larger baselines struggle with short-distance accuracy due to limited overlapping FOVs and the perception model's maximum disparity setting. While increasing FOVs can improve overlap, it reduces accuracy by lowering disparity values. As a result, both our method and DISeR converge on small FOVs and moderate baselines to balance accuracy across all distances. See the supplementary material for qualitative results and analysis.

Fig. 4 compares the training curves for our joint optimization method, our method with fixed perception parameters, and the RL method. Our method converges faster than DISeR by directly optimizing camera parameters, whereas DISeR optimizes a policy network with many parameters to predict them. The RL method shows abrupt changes during training, as all network parameters are updated in each iteration, leading to significant fluctuations in camera parameters and performance. In contrast, the genetic algorithm makes smaller parameter adjustments, resulting in smoother performance changes. Our method converges with 45 steps in 2 min and completes 1000 steps in 38 minutes with an NVIDIA RTX4070 GPU, while DISeR takes 700 steps (67 min) to converge and 97 minutes to complete 1000 steps.

4.3. Monocular Camera on Mixed Reality Headset

In our second experiment, we apply our method to design a monocular RGB camera for an MR headset. Object detection, obstacle avoidance, and feature extraction for 3D reconstruction are selected as examples of tasks as they are essential for most MR devices. However, other tasks can be added depending on the target application.

Environment and Data We establish an indoor environment in UE 5 with 10 object classes. Floorplans and object locations are randomly generated to introduce variability and the camera is mounted on an auto-agent acting as a user. During training, each camera design captures 500 images, indicating 10000 images in total as we optimize for 20 generations, 10 solutions per generation. Testing is conducted with 1000 images using different scene configurations.

Table 2. We compared the parameters and performance of cameras designed with our method using both fully discrete and quantized continuous schemes alongside three robotic/machine vision cameras. Optimized parameters via our method are labeled with ●, covering all camera parameters, the object detection network in the joint optimization scheme, and mounting angles of off-the-shelf cameras. Our approach consistently designs cameras with higher performance in both scenarios. The quantized continuous scheme notably outperforms the fully discrete scheme, benefiting from its consideration of parameter interdependencies. Joint optimization of the object detector also enhances detection precision, while the performance of non-trainable tasks remains similar.

Scenario	Camera	Camera Parameters				Performance			
		Pitch Angle θ (°)	Focal Length f (mm)	Sensor Size $w \times h$ (mm)	Pixel Size p (μm)	Obstacle Detect. Accuracy ↑	Object Detect. AP ↑	Inlier Number ↑	Inlier Ratio ↑
Day	OAK-D [35]	-20.04 ●	2.75 ●	6.29×4.71 ●	1.55 ●	1	0.37 ●	115	0.07
	FLIR [17]	-23.28 ●	3.6 ●	6.2×4.65 ●	1.55 ●	1	0.37 ●	77	0.05
	Basler [3]	-25.90 ●	3.6 ●	4.8×3.6 ●	3.75 ●	1	0.23 ●	131	0.08
	Ours - Fully Discrete	-27.87 ●	4.01 ●	8.45×6.76 ●	6.6 ●	1	0.43 ●	191	**0.13**
		-24.69 ●	3.77 ●	8.45×6.76 ●	6.6 ●	1	0.51 ●	189	0.12
	Ours - Quantized Continuous	-21.86 ●	2.99 ●	8.45×6.76 ●	6.6 ●	1	0.46 ●	**230**	**0.13**
		-26.34 ●	2.88 ●	7.31×5.58 ●	4.5 ●	1	**0.64** ●	224	**0.13**
Night	OAK-D [35]	-19.72 ●	2.75 ●	6.29×4.71 ●	1.55 ●	1	0.34 ●	117	0.06
	FLIR [17]	-22.71 ●	3.6 ●	6.2×4.65 ●	1.55 ●	1	0.36 ●	69	0.03
	Basler [3]	-25.83 ●	3.04 ●	4.8×3.6 ●	3.75 ●	1	0.16 ●	122	0.07
	Ours - Fully Discrete	-23.65 ●	2.61 ●	7.2×5.4 ●	4.5 ●	1	0.37 ●	**179**	**0.11**
		-23.66 ●	3.10 ●	7.2×5.4 ●	4.5 ●	1	0.42 ●	170	**0.11**
	Ours - Quantized Continuous	-29.86 ●	3.33 ●	8.45×6.76 ●	6.6 ●	1	0.37 ●	165	**0.11**
		-22.58 ●	3.49 ●	14.48×9.94 ●	9 ●	1	**0.57** ●	172	**0.11**

Design Space We focus on designing geometric parameters that determine the camera's FOV and photometric parameters affecting resolution, crucial for user mobility and objects' effective resolution. For FOV, we optimize the mounting angle in pitch direction ($\theta \in [-30°, 30°]$, focal length ($f \in [1\text{mm}, 20\text{mm}]$), and image sensor dimensions (width w and height h). The number of pixels is decided by the sensor's pixel size (p) and its dimensions. The camera's height varies randomly between 1 m to 2 m per training step to ensure robustness across different user heights.

We restrict our design to readily available sensors by optimizing the image sensor (i) as a categorical variable, selecting from a catalog of 43 commercial CMOS image sensors from five manufacturers (i_{c1}, i_{c2},..., i_{c43}). Each sensor (i) comprises a set of sensor-related parameters (w, h, and p). We compare two optimization techniques for discrete variables, treating i as fully discrete and using the quantized continuous approach. Our catalog predominantly features 28 Sony sensors, reflecting their widespread use in machine vision cameras, and our noise model is more likely to generalize well across sensors from the same manufacturer. Hence, we also used a Sony sensor to calibrate.

Obstacle Avoidance To assess the camera's ability to detect obstacles affecting user mobility, we place thresholds at room entrances in our virtual environment. These low-height obstacles highlight the importance of an appropriate FOV and mounting angle. We focus on low-height obstacles because taller ones are constrained by object and feature detection tasks. Visibility to the camera is determined by whether the obstacle appears in rendered images. Additionally, the auto-agent is programmed to react when stepping on a threshold, regardless of its visual rendering.

Feature Extraction We extract ORB features [47] and employ the Brute-Force Matcher in OpenCV [27] to match features across consecutive frames, we then apply RANSAC [16] to find inlier features while estimating transformation matrix between frames.

Object Detector We utilize a Faster R-CNN [44] object detector with a ResNet-50 [19] backbone, pretrained on 2000 images generated by our simulator and then fine-tuned alongside the camera parameters. Training employs an Adam optimizer [28] with a batch size of 8 and a learning rate of $1 \cdot 10^{-4}$, which decays by 0.5 every 5 steps during both pre-training and fine-tuning.

Fitness Function The fitness function combines three perception tasks. For obstacle avoidance, it calculates the ratio of the number of obstacles seen by the camera (o_{seen}) to the total number of obstacles (o_{total}) in the user's path. The object detector is trained using its loss function (l_{OD}), and we take the Average Precision (AP) with a 0.5 Intersection-over-Union (IoU) threshold as a term in the fitness function. Feature extraction contributes through the number of inlier features (n_{inlier}) and the ratio of inlier features to total features (n_{total}), emphasizing both the quantity and accuracy of detected features. Thus, the total fitness function is

$$F = \lambda_{feature}(\lambda_{inlier} n_{inlier} + \lambda_{ratio} \frac{n_{inlier}}{n_{total}}) + \lambda_{OD}\text{AP@}0.5IoU + \lambda_{obstacle} \frac{o_{seen}}{o_{total}}, \quad (3)$$

where we balance the weights of all terms by setting λ_{inlier} to 0.0025 (inlier in an image is typically 100-250), λ_{ratio} to 0.5, $\lambda_{obstacle}$, λ_{OD}, and $\lambda_{feature}$ to 1 as we consider these tasks equally significant.

Derivative-Free Optimization We optimize the camera parameters using the genetic algorithm over 20 generations with 10 solutions per generation. A uniform crossover is applied using the top 5 solutions to produce offspring, while

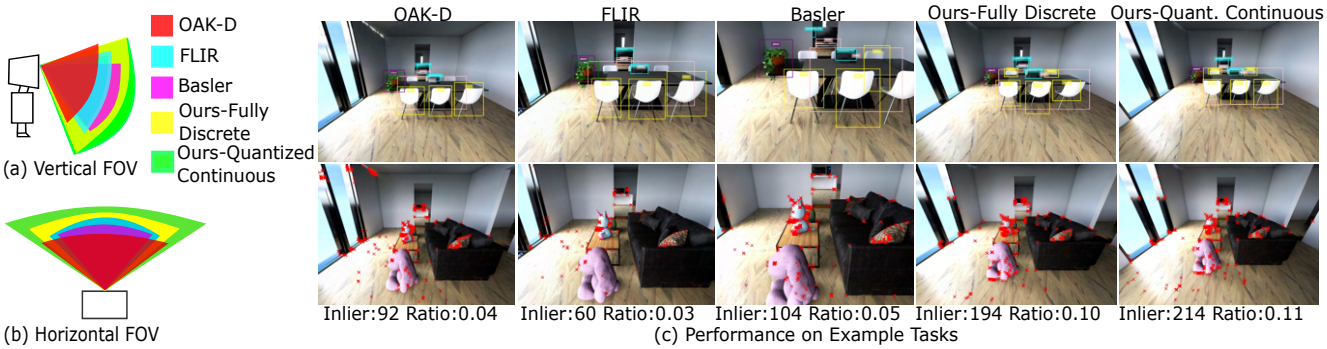

Figure 5. The vertical (a) and horizontal (b) FOVs of cameras optimized by our method with joint optimization, alongside those designed by humans under the daytime scenario, and example evaluation (c). Our method designs a camera with the largest FOV, enabling the capture of all obstacles and objects, extracting more features while maintaining sufficient resolution for effective object and feature detection.

the top 3 solutions are reused. The mutation process remains consistent with the previous experiment, except the addition value is randomly selected between -3 and 3. These hyperparameters are chosen empirically. Each solution involves collecting 500 images to evaluate both the feature extraction and obstacle (threshold) avoidance tasks.

Results We present the optimized set of camera parameters obtained through our approach and evaluate the camera's performance in Tab. 2, considering obstacle detection accuracy, AP score, average number of inlier ORB features across consecutive frames, and ratio of average inlier features to total features extracted. Additionally, we report optimized parameters under two application scenarios: daytime operation in a well-illuminated simulation environment (20 lux) with a lower baseline camera gain (5 dB), and nighttime operation in low-light conditions (2 lux) with a higher baseline camera gain (15 dB). We observed that different application scenarios led to distinct camera designs. Our approach designed a camera with a larger pixel size for nighttime applications, which is expected as a sensor with larger pixels delivers higher SNR since larger pixels gather more light. Hence, sensors with larger pixels require lower gain to capture images with the comparable measured intensity compared to sensors with smaller pixels.

We compare our optimized camera with three human-designed robotic/machine vision cameras, OAK-D, FLIR, and Basler, used in previous experiments in our simulation. Specific noise models are applied to these cameras, with optimized mounting angles while others remain fixed due to configurability. Additionally, we report performance without joint optimization, where parameters of the object detection network are frozen. Fig. 5 illustrates FOVs and example evaluations of our proposed cameras with joint optimization alongside robotic/machine vision cameras. Please refer to the supplementary material for more visualisations.

The results show performance improvements across all tasks with our method, while lower performance is observed under night scenarios due to reduced SNR. Fully discrete optimization schemes show lower performance, indicating the importance of considering parameter interdependencies. Similarly, freezing parameters of the perception model results in reduced performance. The performance of obstacle avoidance is always perfect because we optimized the pitch angle of all the cameras, including the off-the-shelf ones.

We note that the RL method was developed for and demonstrated on single-frame tasks [29]. Both feature matching and obstacle avoidance are episodic, multi-frame tasks requiring consecutive images for precise performance evaluation, and this data collection process is time-consuming. The RL method demands significantly more training steps and image data than ours, applying it to episodic tasks requires impractically long optimization times that prevent us from making a direct comparison. Hence, it is not compared in this experiment.

5. CONCLUSION

We presented a novel end-to-end approach that combines derivative-free and gradient-based optimizers to co-design cameras with perception tasks efficiently. Utilizing UE and an affine noise model, we constructed a camera simulator and validated its accuracy against physical cameras. Our method handles continuous, discrete, categorical camera parameters, and advances a quantized continuous approach for discrete variables to consider their interdependencies. We believe this work can be generalized easily and is an important step toward principled and automated camera design for autonomous systems that account for the interdependency between cameras and the algorithms that interpret them. For future work, we aim to develop a task-driven control algorithm that dynamically adjusts camera parameters, such as exposure settings, in an online manner.

Acknowledgements We would like to thank both ARIA Research Pty Ltd and the Australian government for their funding support via a CRC Projects Round 11 grant.

References

[1] Azeddine Aissaoui, Abdelkrim Ouafi, Philippe Pudlo, Christophe Gillet, Zine-Eddine Baarir, and Abdelmalik Taleb-Ahmed. Designing a camera placement assistance system for human motion capture based on a guided genetic algorithm. *Virtual reality*, 22:13–23, 2018. 3

[2] Seung-Hwan Baek and Felix Heide. Polka lines: Learning structured illumination and reconstruction for active stereo. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5757–5767, 2021. 1, 2

[3] Basler AG. *Basler dart daA1280-54uc*, 3 2024. Rev. 85. 5, 7

[4] Ellis I Betensky. Postmodern lens design. *Optical Engineering*, 32(8):1750–1756, 1993. 3

[5] Henryk Blasinski, Joyce Farrell, Trisha Lian, Zhenyi Liu, and Brian Wandell. Optimizing image acquisition systems for autonomous driving. *Electronic Imaging*, 2018(5):161–1, 2018. 1, 2

[6] Julie Chang, Vincent Sitzmann, Xiong Dun, Wolfgang Heidrich, and Gordon Wetzstein. Hybrid optical-electronic convolutional neural networks with optimized diffractive optics for image classification. *Scientific reports*, 8(1):12324, 2018. 1, 2

[7] Julie Chang and Gordon Wetzstein. Deep optics for monocular depth estimation and 3d object detection. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10193–10202, 2019. 1, 2

[8] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5418, 2018. 5, 6

[9] Geoffroi Côté, Fahim Mannan, Simon Thibault, Jean-François Lalonde, and Felix Heide. The differentiable lens: Compound lens search over glass surfaces and materials for object detection. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 20803–20812, 2023. 1, 2, 4

[10] Donald G Dansereau, Bernd Girod, and Gordon Wetzstein. LiFF: Light field features in scale and depth. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8042–8051, 2019. 5

[11] Steven Diamond, Vincent Sitzmann, Frank Julca-Aguilar, Stephen Boyd, Gordon Wetzstein, and Felix Heide. Dirty pixels: Towards end-to-end image processing and perception. *ACM Trans. Graph.*, 40(3):1–15, 2021. 1, 2

[12] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In *Conference on robot learning*, pages 1–16. PMLR, 2017. 2, 5

[13] Epic Games. Unreal engine. 2, 3

[14] Yi Chin Fang, Tung-Kuan Liu, Cheng-Mu Tsai, Jyh-Horng Chou, Han-Ching Lin, and Wei Teng Lin. Extended optimization of chromatic aberrations via a hybrid taguchi–genetic algorithm for zoom optics with a diffractive optical element. *Journal of Optics A: Pure and Applied Optics*, 11(4):045706, 2009. 3

[15] Yi-Chin Fang, Chen-Mu Tsai, John MacDonald, and Yang-Chieh Pai. Eliminating chromatic aberration in gauss-type lens design using a novel genetic algorithm. *Applied Optics*, 46(13):2401–2410, 2007. 3

[16] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981. 7

[17] FLIR Integrated Imaging Solutions Inc. *FLIR FLEA®3 USB3 Vision*, 1 2017. Rev. 8.1. 5, 7

[18] Alessandro Foi. Clipped noisy images: Heteroskedastic modeling and practical denoising. *Signal Processing*, 89(12):2609–2629, 2009. 2, 4

[19] 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. 7

[20] Lei He, Guanghui Wang, and Zhanyi Hu. Learning depth from single images with deep neural network embedding focal length. *IEEE Transactions on Image Processing*, 27(9):4676–4689, 2018. 2

[21] Andries M Heyns. Optimisation of surveillance camera site locations and viewing angles using a novel multi-attribute, multi-objective genetic algorithm: A day/night anti-poaching application. *Computers, Environment and Urban Systems*, 88:101638, 2021. 3

[22] John H Holland. *Adaptation in natural and artificial systems: an introductory analysis with applications to biology, control, and artificial intelligence*. MIT press, 1992. 2, 4

[23] Kaspar Höschel and Vasudevan Lakshminarayanan. Genetic algorithms for lens design: a review. *Journal of Optics*, 48(1):134–144, 2019. 3

[24] Yunzhong Hou, Xingjian Leng, Tom Gedeon, and Liang Zheng. Optimizing camera configurations for multi-view pedestrian detection. *arXiv preprint arXiv:2312.02144*, 2023. 2

[25] Hayato Ikoma, Cindy M Nguyen, Christopher A Metzler, Yifan Peng, and Gordon Wetzstein. Depth from defocus with learned optics for imaging and occlusion-aware depth estimation. In *2021 IEEE International Conference on Computational Photography*, pages 1–12. IEEE, 2021. 1, 2

[26] Intel. *Intel RealSense Product Family D400 Series*, 3 2024. Rev. 018. 6

[27] Itseez. Open source computer vision library. https://github.com/itseez/opencv, 2015. 7

[28] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *Proccedings of the 3rd International Conference on Learning Representations*, 2015. 4, 6, 7

[29] Tzofi Klinghoffer, Kushagra Tiwary, Nikhil Behari, Bhavya Agrawalla, and Ramesh Raskar. DISeR: Designing imaging systems with reinforcement learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 23632–23642, 2023. 2, 6, 8

[30] Michael F Land and Dan-Eric Nilsson. *Animal eyes*. OUP Oxford, 2012. 1

[31] Ce Liu, William T Freeman, Richard Szeliski, and Sing Bing Kang. Noise estimation from a single image. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 1, pages 901–908. IEEE, 2006. 4

[32] Zhenyi Liu, Trisha Lian, Joyce Farrell, and Brian Wandell. Soft prototyping camera designs for car detection based on a convolutional neural network. In *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*, pages 0–0, 2019. 1, 2

[33] Zhenyi Liu, Shen Minghao, Jiaqi Zhang, Shuangting Liu, Henryk Blasinski, Trisha Lian, and Brian Wandell. A system for generating complex physically accurate sensor images for automotive applications. *Electronic Imaging*, 2019:53–1, 01 2019. 1, 2

[34] Zhenyi Liu, Devesh Shah, Alireza Rahimpour, Devesh Upadhyay, Joyce Farrell, and Brian A Wandell. Using simulation to quantify the performance of automotive perception systems. *arXiv preprint arXiv:2303.00983*, 2023. 1, 2

[35] Luxonis. *OAK-D Pro (USB boot with 256Mbit NOR flash)*, 12 2021. 5, 7

[36] Julien NP Martel, Lorenz K Mueller, Stephen J Carey, Piotr Dudek, and Gordon Wetzstein. Neural sensors: Learning pixel exposures for hdr imaging and video compressive sensing with programmable sensors. *IEEE transactions on pattern analysis and machine intelligence*, 42(7):1642–1653, 2020. 1, 2

[37] Moritz Menze, Christian Heipke, and Andreas Geiger. Joint 3d estimation of vehicles and scene flow. In *ISPRS Workshop on Image Sequence Analysis*, 2015. 6

[38] Moritz Menze, Christian Heipke, and Andreas Geiger. Object scene flow. *ISPRS Journal of Photogrammetry and Remote Sensing*, 2018. 6

[39] Christopher A Metzler, Hayato Ikoma, Yifan Peng, and Gordon Wetzstein. Deep optics for single-shot high-dynamic-range imaging. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1375–1385, 2020. 1, 2

[40] Cindy M Nguyen, Julien NP Martel, and Gordon Wetzstein. Learning spatially varying pixel exposures for motion deblurring. In *2022 IEEE International Conference on Computational Photography*, pages 1–11. IEEE, 2022. 1, 2

[41] Gustavo Olague and Roger Mohr. Optimal camera placement for accurate reconstruction. *Pattern recognition*, 35(4):927–944, 2002. 3

[42] Isao Ono, Shigenobu Kobayashi, and Koji Yoshida. Optimal lens design by real-coded genetic algorithms using undx. *Computer methods in applied mechanics and engineering*, 186(2-4):483–497, 2000. 3

[43] Netanel Ratner and Yoav Y Schechner. Illumination multiplexing within fundamental limits. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007. 4

[44] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. 7

[45] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951. 4

[46] 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. 1, 2

[47] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. ORB: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. Ieee, 2011. 4, 5, 7

[48] Vincent Sitzmann, Steven Diamond, Yifan Peng, Xiong Dun, Stephen Boyd, Wolfgang Heidrich, Felix Heide, and Gordon Wetzstein. End-to-end optimization of optics and image processing for achromatic extended depth of field and super-resolution imaging. *ACM Trans. Graph*, 37(4):1–13, 2018. 1, 2

[49] StereoLabs. *ZED 2i Camera Overview & Datasheet*. Rev. 1.2. 6

[50] Qilin Sun, Congli Wang, Fu Qiang, Dun Xiong, and Heidrich Wolfgang. End-to-end complex lens design with differentiable ray tracing. *ACM Trans. Graph*, 40(4):1–13, 2021. 1, 2

[51] Cheng-Mu Tsai and Yi-Chin Fang. Improvement of filed curvature aberration in a projector lens by using hybrid genetic algorithm with damped least square optimization. *Journal of Display Technology*, 11(12):1023–1030, 2015. 3

[52] Ethan Tseng, Ali Mosleh, Fahim Mannan, Karl St-Arnaud, Avinash Sharma, Yifan Peng, Alexander Braun, Derek Nowrouzezahrai, Jean-Francois Lalonde, and Felix Heide. Differentiable compound optics and processing pipeline optimization for end-to-end camera design. *ACM Trans. Graph.*, 40(2):1–19, 2021. 1, 2

[53] 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 Trans. Graph.*, 38(4):27–1, 2019. 1, 2

[54] Derek C Van Leijenhorst, Carlos B Lucasius, and Jos M Thijssen. Optical design with the aid of a genetic algorithm. *BioSystems*, 37(3):177–187, 1996. 3

[55] Brian Wandell, David Cardinal, David Brainard, Zheng Lyu, Zhenyi Liu, Aaron Webster, Joyce Farrell, Trisha Lian, Henryk Blasinski, and Kaijun Feng. Iset/isetcam, 2 2024. 2

[56] Brian Wandell, David Cardinal, Trisha Lian, Zhenyi Liu, Zheng Lyu, David Brainard, Henryk Blasinski, Amy Ni, Joelle Dowling, Max Furth, Thomas Goossens, Anqi Ji, and Jennifer Maxwell. Iset/iset3d, 2 2024. 2

[57] Taihua Wang and Donald G Dansereau. Multiplexed illumination for classifying visually similar objects. *Applied Optics*, 60(10):B23–B31, 2021. 4

[58] Korbinian Weikl, Damien Schroeder, Daniel Blau, Zhenyi Liu, and Walter Stechele. End-to-end imaging system optimization for computer vision in driving automation. In *Proceedings of IS&T Int'l. Symp. on Electronic Imaging: Autonomous Vehicles and Machines*, 2021. 1, 2

[59] Xinge Yang, Qiang Fu, and Wolfgang Heidrich. Curriculum learning for ab initio deep learned refractive optics. *Nature communications*, 15(1):6572, 2024. 1, 2

[60] Xinge Yang, Qiang Fu, Yunfeng Nie, and Wolfgang Heidrich. Image quality is not all you want: Task-driven lens design for image classification. *arXiv preprint arXiv:2305.17185*, 2023. 1, 2

[61] Chih-Ta Yen and Jhe-Wen Ye. Aspherical lens design using hybrid neural-genetic algorithm of contact lenses. *Applied optics*, 54(28):E88–E93, 2015. 3

[62] Yuxuan Zhang, Bo Dong, and Felix Heide. All you need is raw: Defending against adversarial attacks with camera image pipelines. In *European Conference on Computer Vision*, pages 323–343. Springer, 2022. 1, 2