

Soft Prototyping Camera Designs for Car Detection Based on a Convolutional Neural Network

Zhenyi Liu^{1,2}, Trisha Lian¹, Joyce Farrell¹, and Brian Wandell¹

¹Stanford University, USA, ²Jilin University, China
 {zhenyiliu, tlian, jefarrel, wandell}@stanford.edu

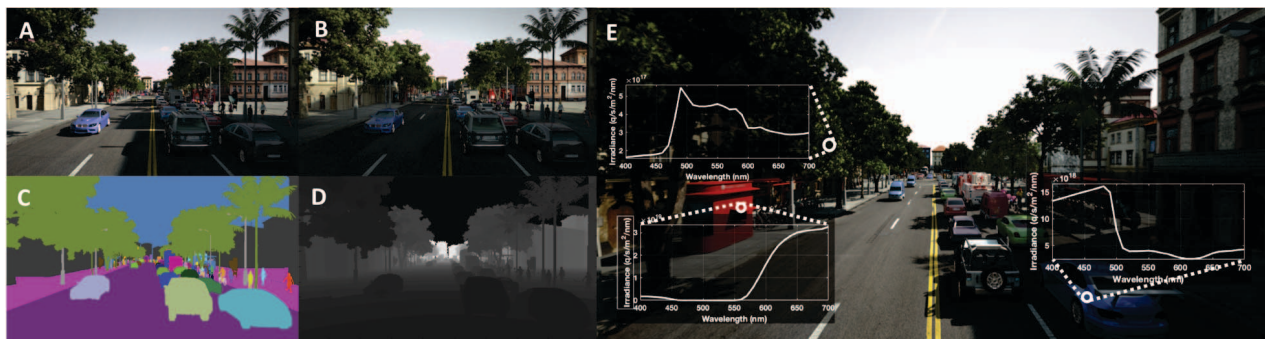


Figure 1. **Simulated scenes and metadata.** An ISE3d generated scene with sky maps measured in (A) early afternoon and (B) late afternoon. The simulation creates pixel-level labeled images including (C) a panoptic segmentation, and (D) a depth map. (E) A simulated scene with superimposed graphs showing the spectral radiance at different image locations (circles). The spectral irradiance is used by the camera simulation to account for the wavelength effects of the lens and color filters.

Abstract

Imaging systems are increasingly used as input to convolutional neural networks (CNN) for object detection; we would like to design cameras that are optimized for this purpose. It is impractical to build different cameras and then acquire and label the necessary data for every potential camera design; creating software simulations of the camera in context (soft prototyping) is the only realistic approach. We implemented soft-prototyping tools that can quantitatively simulate image radiance and camera designs to create realistic images that are input to a convolutional neural network for car detection. We used these methods to quantify the effect that critical hardware components (pixel size), sensor control (exposure algorithms) and image processing (gamma and demosaicing algorithms) have upon average precision of car detection. We quantify (a) the relationship between pixel size and the ability to detect cars at different distances, (b) the penalty for choosing a poor exposure duration, and (c) the ability of the CNN to perform car detection for a variety of post-acquisition processing algorithms. These results show that the optimal choices for car detection are not constrained by the same metrics used for image

quality in consumer photography. It is better to evaluate camera designs for CNN applications using soft prototyping with task-specific metrics rather than consumer photography metrics.

1. Introduction

An imaging system should be evaluated with respect to a task. For example, we judge the image quality of digital cameras by how pleasing the final images appear to consumers. Medical imaging systems are evaluated by how well they differentiate biological substrates and diagnose a disease. One of the most critical aspects for automotive imaging systems is performance on object detection tasks.

The enormous diversity of biological eyes is strong evidence that different tasks are best served by different image system designs [20]. Even a single biological system, such as human vision, is comprised of multiple subsystems specialized for low and high light levels (rods, cones) or for high acuity tasks versus invoking attention (fovea, periphery) [32]. Just as there is no optimal biological visual system for all tasks and conditions, no single camera design

will be optimal.

Designing and building an imaging system for object detection is expensive, collecting image data is time-consuming, and annotating the images is labor-intensive. Hence, an empirical design-build-test loop is impractical for co-design of imaging systems and neural networks, and soft prototyping is required to reduce the time and expense. A prototyping system must combine quantitative computer graphics for creating accurate scene radiance with quantitative methods for simulating the imaging system. Such a system can simulate realistic camera images with accurate labels at each pixel (Figure 1); these images can be used to train neural networks for object recognition and detection [33, 24].

In this paper, we perform computational experiments to assess specific co-designs of cameras and a convolutional neural network for car detection. We first quantify how average precision measured for car detection varies with a key parameter of the camera hardware: pixel size. Next, we analyze performance for variations in a critical sensor control algorithm: exposure duration. Finally, we compare system performance for different choices of the image processing system: gamma correction and demosaicing.

2. Related work

A number of groups have described the value in using computer graphics to create realistic images for machine-learning (ML). Two approaches have been used to generate training images: repurposing game engines [15, 19] and using physically based ray tracing methods [31, 34, 1, 24]. Game engines have the advantage of speed and the ability to efficiently produce video sequences. The ray-tracing methods have the ability to produce quantitatively realistic scene spectral radiance that can be coordinated with camera simulations. Given our objective, we use quantitative ray-tracing for our application.

Previous publications described an initial implementation of ISET3d, which includes procedural methods for generating complex automotive scenes [1, 24]. These scenes are rendered into images using a version of PBRT [26] and ISETCam. Those papers included preliminary evaluations of how camera design impacts CNN performance for pixel size and sensor type. We advance that work in several ways. The paper by [1] did not include procedural modeling and was restricted to relatively simple scenes; rendering was based on a version of PBRT [26] that was subsequently improved with regards to material modeling. The paper by [24] used Faster RCNN that was pre-trained using camera data from BDD100k [36] and tested on ISET3D synthetic data. The work described here uses (a) a much larger and more complex collection of synthetic scenes, (b) network training with the appropriate synthetic data, (c) a new network, Mask R-CNN with a ResNet backbone, and (d) more

extensive camera algorithm analyses.

There are alternative approaches to creating synthetic data. For example, it is possible to use a game engine and then apply domain adaptation methods to make the images more realistic [35, 6]. The realism is evaluated by assessing how well training on the synthetic dataset generalizes to a camera dataset. Domain randomization introduces random variations into the synthetic image with the hope that such perturbations force the network to focus on critical information [30]. Domain stylization uses photorealistic image style transfer algorithms to transform synthetic images so that an independent network cannot discriminate synthetic and measured images [7, 22, 37]. Neither of these methods represent the scene spectral radiance, which is required to account for the impact of wavelength-dependent components, including the optics and sensors [1]. This severely limits the value of this approach for evaluating camera design.

To assess the impact of the image processing pipeline on CNN performance, [2] used RGB images and an invertible model of the image processing pipeline to create a nominal sensor image. This approach does not account for optics and sensor properties, such as pixel size or exposure algorithm.

Data augmentation can improve the generalization between synthetic and camera images. In this approach, a set of synthetic images are transformed by image processing operations that approximate camera effects, including blur, chromatic aberration, and color processing [3, 4]. The network becomes less sensitive to camera differences with augmented data. Data augmentation is complementary to our goal, which is to explore camera design to provide optimal network performance.

Finally, there is a standardization effort by the IEEE-P2020 to address attributes that contribute to image quality for automotive Advanced Driver Assistance Systems applications, as well as identifying existing metrics relating to these attributes [16, 38]. Multiple metrics are under review, including consumer photography metrics such as signal-to-noise (SNR) and spatial frequency measures (MTF50). The soft prototyping environment we describe here clarifies limitations in using such metrics for assessing imaging systems designed for car detection by neural networks.

3. Methods

3.1. Automotive scene simulation

Simulated scene radiance data and sensor irradiance were generated for a collection of about 5000 city scenes using the ISET3d open-source software [24]. The scenes were assembled stochastically from a database of nearly 100 car shapes, 3 bus shapes, 8 truck shapes, 80 pedestrians, 10 bicycles, more than 50 different static objects (trees,

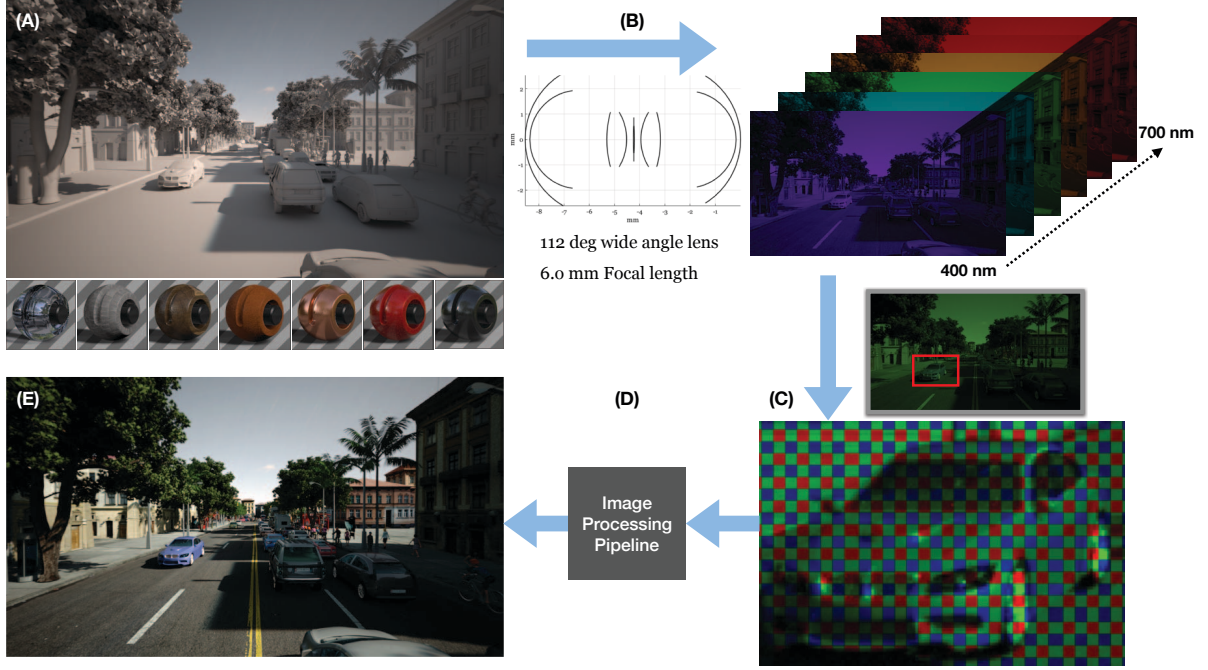


Figure 2. **The computational pipeline.** (A) A three-dimensional scene, including objects and materials, is defined in the format used by Physically Based Ray Tracing (PBRT) software [26]. (B) The rays pass through a lens modeled as a series of surfaces with wavelength-dependent indices of refraction and a Monte Carlo based model of diffraction [14, 28]. (C) Using the sensor specifications, ISETCam converts the spectral irradiance into an array of pixel voltages and then digital values [13, 11]. (D) Sensor data are converted into images by demosaicing and image processing. (E) The images with pixel level annotation are used to train and evaluate the car detection network.

trash cans, traffic signs, billboards, etc.), four city blocks, suburban and more than 200 buildings. The scenes are rendered with sky maps captured from 7:00 am to 7:00 pm at one minute intervals [21]. The source code for creating the images is available in the ISET3d package (<https://github.com/iset/iset3d>). The ray-tracing software is based on the Physically Based Ray Tracing [26] and is available as a Docker container.

3.2. Camera simulation

The ISETCam software converts the sensor spectral irradiance to RGB images (Figure 2) [13, 11, 12]. The sensor parameters for the simulations reported in this paper were based on the MT9V024 sensor manufactured by ON Semiconductor. This sensor is configured with 6-micron pixels, providing relatively high light sensitivity and signal-to-noise with a linear dynamic range of 55 dB. The MT9V024 is designed for automotive machine vision applications with options for monochrome, RGB Bayer and RCCC color filter arrays. This sensor is used in commercial ADAS systems (e.g., MobilEye 630).

We modeled a range of pixel sizes, starting with a configuration that is typical for automotive sensors (1280×720 pixels, RGB Bayer configuration). The sensor dye size is 2.1×3.85 mm. We analyzed system performance over a

range of pixel sizes and scene brightness levels. For the pixel size experiments, we simulated a sensor with a fixed field-of-view (dye size). Consequently, the number of pixels varies inversely with the square of pixel size. The camera lens for these simulations was a wide angle (112 degree) multi-element design with 6 mm focal length. The on-axis point spread of the lens has a full-width at half maximum of approximately $1.5 \mu\text{m}$, which was sufficient to support the smallest RRGB superpixel size ($3.0 \mu\text{m}$) in our simulations.

3.3. Soft prototyping validation

Full system validation is not practical: this would require that we collect and label images with a simulated sensor and then synthesize scenes that match the collected images. It is possible, however, to validate quantitatively critical components of the soft-prototyping system, and we have done so.

Sensor simulations were validated using a scene whose lights and surfaces were measured with a spectroradiometer and compared predicted data with data from a camera whose lens and sensor were modeled [10, 5]. The accuracy of the optics simulations were validated by comparing the simulated data with physical laws (diffraction-limited, Snells Law) and compared with Zemax calculations based on multi-element lens designs [23]. The geometry and

materials models, in particular the light-surface interaction function captured in the bidirectional reflectance distribution function (BRDF), were derived from measurements of real materials. The 3D car models are derived from 3D scans or CAD designs of real cars. The spectral characteristics of the sky maps used to simulate daylight were measured using a multispectral lighting capture system and are also validated [21].

3.4. Object Detection Network and Training

For car detection on the camera, we chose Mask R-CNN [9] with ResNet50-FPN as the network backbone. Mask R-CNN is a state-of-the-art region-proposed convolutional neural network(R-CNN) designed to solve instance segmentation problems in computer vision [17]. It extends Faster R-CNN [27], by adding a third branch that outputs the pixel level segmentation. Mask R-CNN also includes an ROIAlign layer to extract the feature map; the addition of this layer significantly improves detection accuracy. In this paper, we evaluate detection performance by measuring the overlap of the bounding box from Mask R-CNN with the bounding box of the labeled image data.

A pretrained model can be useful in some contexts, but to investigate the impact of camera design it is best to train the network from scratch using the simulated camera data (e.g., pixel size, exposure algorithm, post-processing algorithm). Hence, the neural network only learns to interpret images from a specific camera under the specific imaging conditions. The sensor images were calculated from the spectral irradiance simulations using ISET3d and the ISETCam camera model. We divided the sensor images obtained from the scenes into three independent groups used for training, validation, and testing. We used 3000 sensor images for training, 700 held-out sensor images for validation at each checkpoint, and 750 held-out sensor images to measure system performance.

We trained the network for 60 epochs, saving the checkpoint every 5 epochs. We evaluated the network using the validation images at each of 12 checkpoints and we use the network parameters from the checkpoint with highest score. We then run this network on the test dataset at this checkpoint. We trained on 4 Nvidia P100 GPUs with batch size equal to 8 and evaluated on 1 GPU with batch size equal to 4. We started training with the learning rate set to 0.02, and we decreased the learning rate to 0.002 after 30 epochs. The model is trained and evaluated with only one class: car.

3.5. Object detection metrics

Car detection performance was assessed using the PASCAL AP@0.5IOU [8], which we refer to as average precision (AP). When the network identifies a car within a bounding box, and that box overlaps with at least half of the area of a bounding box of the labeled pixels, we score the

detection as correct (a hit), and otherwise the box is scored as an error (false alarm). The AP combines these two values and is equivalent to measuring the area under the receiver operating characteristic defined in classic signal detection theory [29].

In many cases we measure AP as a function of distance between the camera and the car. We can obtain this function because the soft-prototyping tools provide both the instances labels and a depth map of the scene for every pixel(Figure 1). We use the label (car) and depth (meters) to sort the cars in the test dataset. We trace the curve by calculating the AP for all the cars within a 10 meter range. The distance (meters) at which the AP curve crosses 0.50 is a scalar summary value which we denote as the OD50.

4. Experiments

4.1. Pixel size

To assess how pixel size impacts the car detection AP, we simulated sensors with a range of pixel sizes that are typical for automotive applications: 1.5 μm to 6.0 μm . The simulation labels the pixels arising from any car within 150 m. The number of image pixels labeled as 'car' depends significantly on pixel size, occlusions, and distance to the car. There are roughly 33,000 labeled cars in the collection of training images. The number of labeled cars decreases with distance (Figure 3, histogram), but is the same for all camera pixel sizes. As the pixel size increases, however, the number of labeled pixels per car decreases (Figure 3, images).

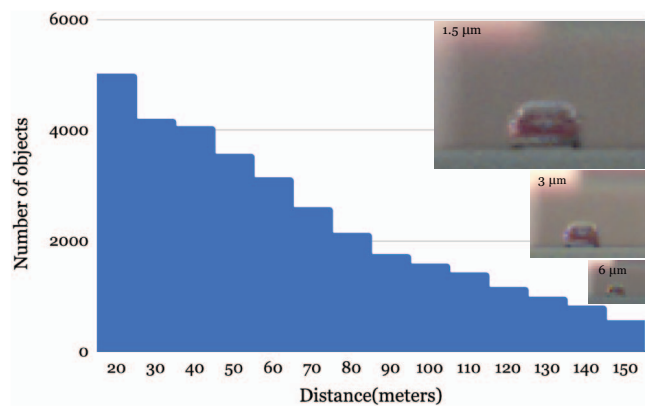


Figure 3. **Number of cars for network training at different distances** The histogram represents the number of cars in the training set at each distance. The images at the right show simulations of a clear view of a car at 150 m for each of the pixel sizes. The simulated pixel sizes span the range of typical modern day CMOS sensors; 3 and 6 μm pixel sizes are common for current automotive sensors.

We trained the network separately using images from each pixel size, and we measured how AP declines with

distance (Figure 4). Detection performance using different pixel sizes declines gradually with distance. The rate of decline depends on pixel size. The distance where detection reaches 0.50 increases non-linearly with pixel size, varying from 55 m to 110 m.

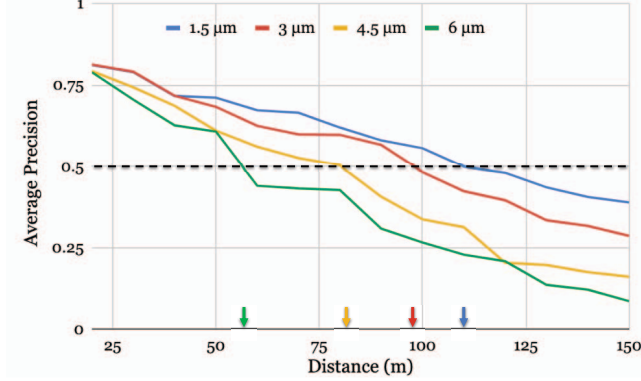


Figure 4. **Car detection for cameras with different pixel sizes.** The solid curves shows the PASCAL AP@0.5IOU for car detection as a function of the distance between the camera and the car. Each curve simulates a different pixel size. The corresponding colored arrows indicate the object distance for 50% detection (OD50).

The gradual decline with distance is likely to be a consequence of the reduction in the number of pixels corresponding to each car: more distant cars occupy less of the field of view. This principle is similar to the Johnson criterion that is used to summarize device resolution [18].

The choice of pixel size may be more significant for some applications than others. For example, if the application is to survey cars within a distance between 1 and 40 meters (e.g., at a toll booth), pixel size may not be a significant factor. We make extensive use of the AP as a function of distance in this paper; it will be necessary to consider other task-specific metrics in future work.

4.2. Exposure algorithms

The following experiments differ from the pixel size evaluations in two ways. First, we simulate a fixed $3\mu\text{m}$ pixel size. Second, we change the labeling policy to align with the conventional practice based on single camera data. The pixel size analysis used the knowledge from the simulation to label pixels: every car was labeled at every pixel size. Modern applications based on camera data typically label a pixel only if a human observer perceives the pixel as belonging to a car. We implemented this visibility constraint by labeling pixels only if they are part of a group of pixels with a bounding box of 10×15 pixels, big enough for a person to recognize. Including perception as a criterion of the labeling policy removes 24 % of the car instances from the training and evaluation data but only 0.14 % of labeled pixels. These are mainly distant or highly occluded cars. This

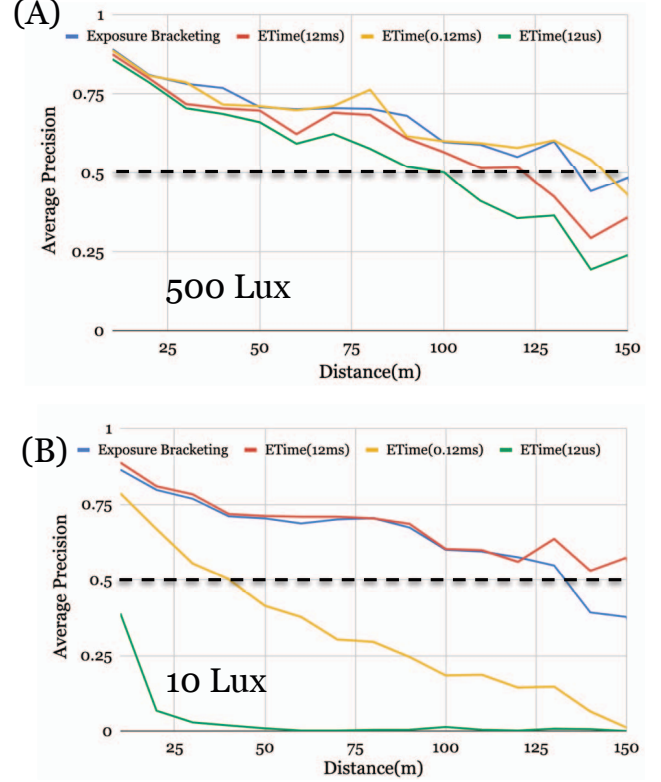


Figure 5. **Exposure duration has a significant impact on detection performance.** Average precision (AP) for car detection plotted as a function of the distance between the camera and the car with exposure time (12 ms, 0.12 ms, 12 μs) as the parameter. AP data are calculated for networks trained on sensor data with mean illuminance of 500 lx (top) or with mean sensor illuminance of 10 lx (bottom).

restriction brings the evaluation into compliance with current practice and increases network performance (AP and OD50) by removing difficult-to-detect cars from the evaluation data.

Exposure value algorithms adjust the integration time and lens aperture to bring pixel responses into their operating range: pixel responses should be above the dark noise level and below the saturation level. In this section, we analyze the impact of exposure-controlling algorithms on car detection, when the lens aperture is fixed. We measure the impact using the average precision of CNN car detection in experiments with simulate scenes. The scenes are modeled at different times of the day and with a range of mean luminance levels, from extremely bright sunlight (500 cd/m^2) to very dark late afternoon (10 cd/m^2). Under these conditions, the sensor illuminance of an $f/\# 4$ lens ranges from 10-500 lux. The diversity of surfaces and illumination in the simulated scenes generates images with dynamic ranges of 2-3 log units.

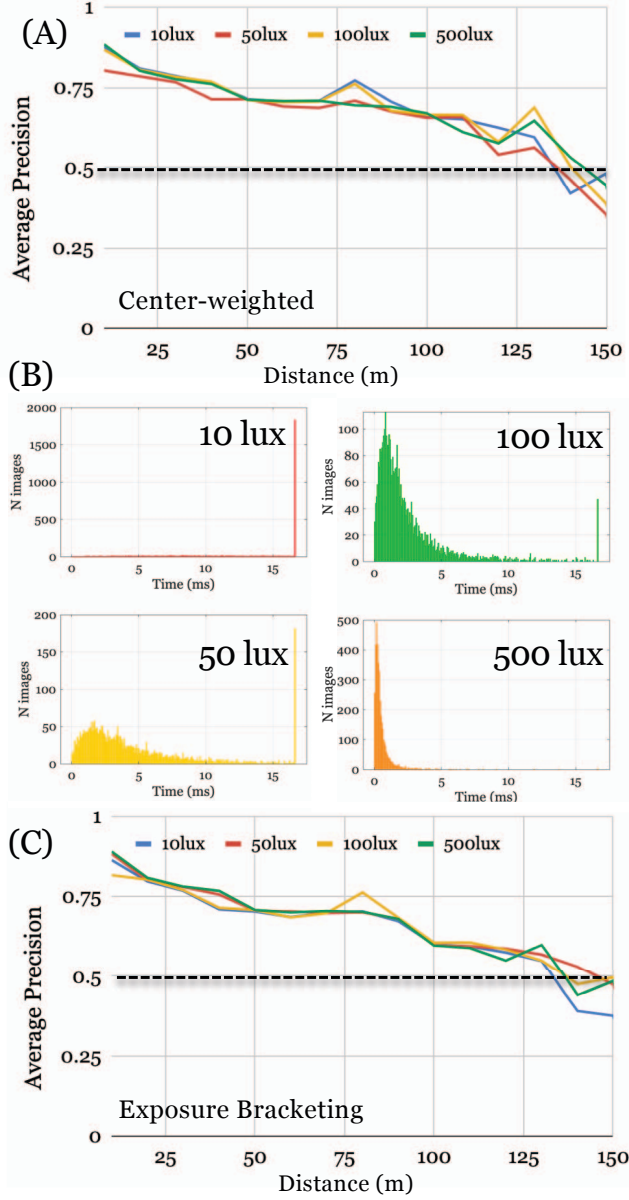


Figure 6. **Comparison of center-weighted and exposure-bracketing exposure algorithms.** (A) Average precision for detecting cars in sensor images captured using a center-weighted auto-exposure algorithm, plotted as a function of the distance between the camera and the car for different sensor illuminance levels (B) Histogram of center-weighted exposure times for scenes with different sensor illuminance levels. (C) Average precision for detecting cars in sensor images captured using exposure-bracketing algorithms, plotted as a function of object distance for different sensor illuminance levels.

We quantified the performance of CNN models trained on images processed with two different exposure duration algorithms. The first is a center-weighted algorithm - the exposure duration is set so that a central region of the scene,

about 100th of the whole image, falls within the pixel operating range. We impose a 16 ms upper bound based on an assumption that images are acquired at a rate of 60 frames per second (fps). The second is an exposure-bracketing algorithm - we combine the data from three frames with different exposure times (12 ms, 0.12 ms, 12 μ s). These images are combined by selecting the highest value prior to saturation, accounting for the exposure duration. The exposure bracketing method with these durations extends the sensor dynamic range by two orders of magnitude.

Selecting an appropriate exposure duration has a significant impact on average precision; performance is significantly reduced when the algorithm chooses a poor exposure time (Figure 5). The curves in the two panels quantify the impact of choosing too long an exposure for a bright scene (Figure 5A) or too short an exposure in a dark scene (Figure 5B). The penalty for choosing a poor exposure duration in the bright scene can be measured in terms of the OD50. The best duration (0.12 ms) has an OD50 of about 140 m, while the OD50 for the longer and shorter exposures are 125 m and 100 m. Under the low light conditions, the best exposure duration (12 ms) OD is about 130 m and choosing the wrong duration incurs an even larger penalty, with OD50 values of just 35 m and worse.

During informal experiments, we found that the region of interest of the exposure algorithm is important. Choosing a poor region or using the entire image led to poor results. The images used typically have a dynamic range of about 500:1, which fills up most of the pixel response range. Sub-optimal choice of the exposure duration puts part of the image beyond the pixels response range. For example, if one uses the entire scene a short exposure time is often chosen to avoid saturation from the bright sky, and this choice reduces the visibility of cars within a shadowed portion of the image. A center-weighted algorithm reduces the chance of an exposure duration error that impacts car detection. The exposure-bracketing algorithm reduces the chances of setting the wrong exposure duration: the algorithm acquires multiple captures at the cost of reading multiple frames and assembling the data into a single output at video rate. The three exposure durations used in the simulation were chosen to leave enough time for reading and assembling the output at 60 Hz. We compared the center-weighted algorithm and exposure bracketing for the collection of simulated images. We plot car detection performance for scene illumination levels ranging from 10 lx to 500 lx (Figure 6). The average performance on car detection does not differ substantially between the exposure-bracketing algorithm and the best exposure duration (0.12 ms for panel A: yellow curve; and 12 ms for panel B: red curve). The center-weighted exposure algorithm chooses many different exposure durations at each mean illumination level (see inset histograms).

The center-weighted algorithm chooses an appropriate

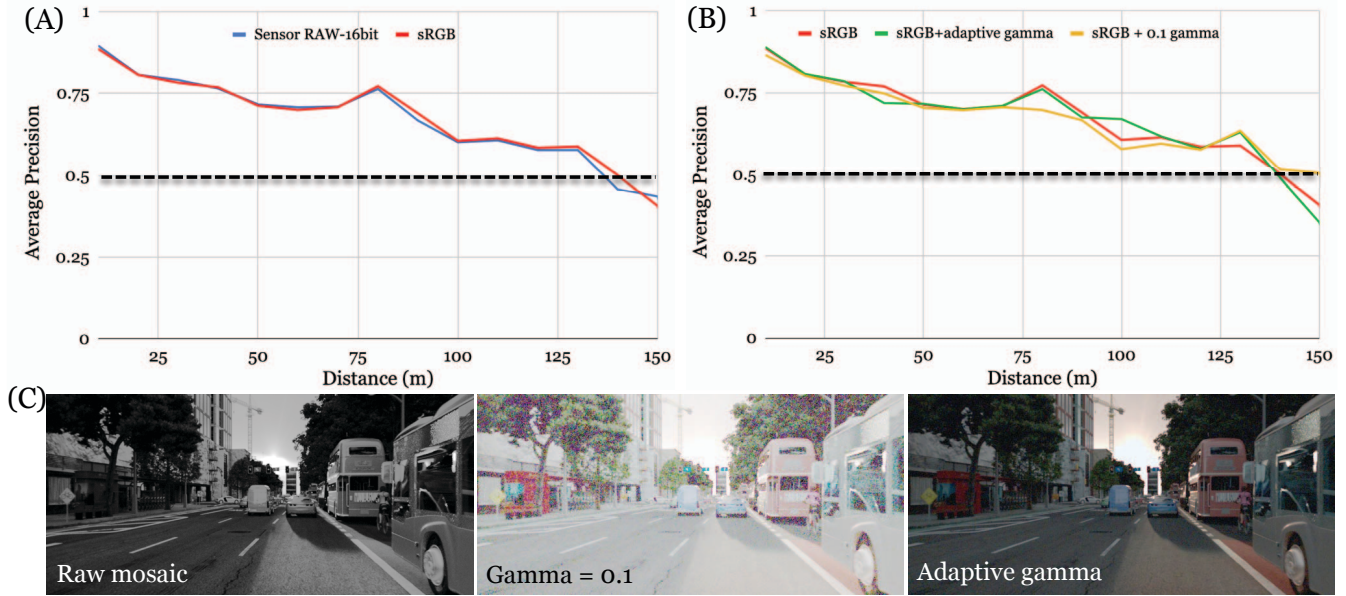


Figure 7. **The impact of image processing parameters on CNN performance.** (A) Average precision for detecting cars by networks trained on raw sensor data (blue) compared to average precision for networks trained on processed (sRGB) sensor data (red) plotted as a function of the distance between the camera and the car. (B) Average precision for networks trained on processed sensor data rendered with different gamma values plotted as a function of object distance. (C) Images of unprocessed sensor data (left image), and processed sensor data rendered with different gamma settings.

exposure time for the scenes in our collection with sensor illumination ranging between 10 and 500 lx (Figure 6). On average, these choices bring the image information about the cars into the pixel response range and car detection performance is not significantly different from the exposure-bracketing algorithm. While the averages agree, in certain cases using exposure-bracketing to increase the image dynamic range should have an advantage. Below we examine one such edge case - for example system performance in specific cases such as a highly reflective (specular) surface in the middle of the road.

4.3. Image processing

Consumer photography imaging incorporates many post-acquisition processing algorithms to render sensor data. The algorithms include demosaicing, conversion of sensor data to a calibrated color space, illuminant correction algorithms, noise reduction methods, and mapping for dynamic range and color gamut. The goal of these algorithms is to make an image that is pleasing to humans. Preliminary studies have begun to assess whether designing systems for these image quality measures improves CNN performance. Because these algorithms take time and can be energy consuming, understanding their impact is likely to be an ongoing topic of investigation. To date there are no firm conclusions about their impact [2, 1].

It is straightforward to implement image processing al-

gorithms as part of the prototyping system, and we have quantified several post-acquisition processing steps with respect to car detection. Figure 7 compares the average precision of car detection in two cases. In both cases we compare performance with a standard image processing pipeline (demosaicing, sensor conversion) that transforms the sensor data to JPEG format in sRGB space [25]). The first alternative eliminates the processing stream and the second alters the parameters of the processing.

Neither alternative post-processing has a significant impact on the car detection performance in daytime. Results are very similar when the network is trained with an sRGB image or the unprocessed sensor mosaic data (center-weighted exposure algorithm, Figure 7A). Maintaining the processing pipeline but using two different power functions (gamma mapping) to adjust the sRGB data also had no significant effect (Figure 7B). One method applied a fixed value (0.1) that brightened the high dynamic range images. The second method chose gamma adaptively so that the mean value of the sensor data raised to a power (gamma) equals 20 % of the voltage swing. The CNN training achieved the same performance level in each of these cases. This finding differs from [2], but they began with a network pre-trained on RGB images to evaluate the different image processing pipelines. These post-acquisition transformations produce images that are extremely different to the human eye (Figure 7C) and yet they had no effect on the precision of CNN

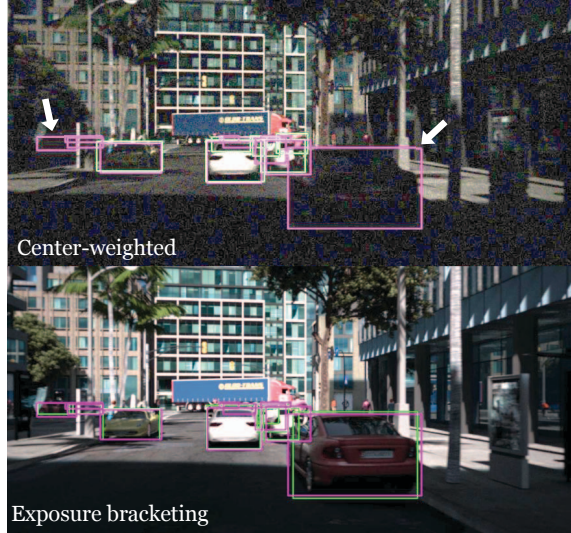


Figure 8. **Examining individual cases.** Images of the same scene captured with center-weighted (top) and exposure-bracketed (bottom) algorithms. Magenta bounding boxes indicate the locations of cars in the scene (ground truth) and green bounding boxes overlay the locations of cars that were detected.

car detection. Measures of image quality are not a good guide for predicting image system performance for car detection.

4.4. Edge cases

For driving applications, the high cost of certain errors make relatively rare events very problematic. Hence, it is important to consider specific cases in addition to average precision. Soft prototyping tools enable us to isolate and analyze conditions that require special attention.

An important case that arises in our simulations is shown in Figure 8. A white car in the center of the scene has a shiny (specular) surface that falls within the center-weighted region of interest. On the right side there is a darker (red) car within a shadow cast by a building. The overlaid green bounding box indicates that the car is detected by the CNN when using the exposure-bracketing algorithm but not the center-weighted algorithm. The reason is that the center-weighted algorithm selects a brief exposure duration and renders a noisy image in the shadowed region. The exposure-bracketing algorithm avoids this problem. This situation does not arise often enough to reduce the average precision in our simulations, but such cases can be identified by searching for examples where the exposure-bracketing performance is better than center-weighted performance.

5. Discussion and summary

The design-build-test loop for automotive image systems is expensive and complex and this makes a soft prototyping

tool necessary. Additionally, it is typically impossible to know how variations in an isolated component will impact system performance. These requirements call for an end-to-end prototyping environment. Such a system can greatly speed innovation of image system architectures including choice of lens, sensors, and camera placement by measuring whole system performance on the relevant task.

The open-source and freely available prototyping tools in ISET3d enables us to perform simulations that span scene definition, camera design, and network design. Our focus is on camera design and the experiments carried out in this paper explore aspects of the sensor hardware (pixel size), sensor control algorithms (exposure algorithms) and post-processing (demosaicing and gamma correction). Using the quantitative measure of average precision of car detection, and the summary measure of OD50, we quantified system performance. These experiments show that car detection algorithms identify cars at a distance approaching 115 meters for small ($1.5\mu\text{m}$) pixels and 98 meters for larger ($3\mu\text{m}$) pixels. Further experiments with lens and sensor design might lead to improved performance. Center-weighted exposure produced good average precision, but exposure-bracketing algorithms have higher reliability and handle certain edge cases that are problematic for the single-exposure center-weighted acquisition. Finally, we found no advantage for post-acquisition image processing for detecting cars in daytime conditions. When the network is trained from scratch on the relevant data, performance is nearly identical whether using digital values directly from the sensor mosaic or RGB values after post-acquisition processing.

The consumer photography industry has developed many image quality tools, including SNR and MTF50. In principle these metrics might be used to assess image systems for car detection. We find, however, that there is a divergence between image quality metrics and CNN detection performance. Images that would be intolerable in consumer photography - with extreme gamma values or without any post-acquisition processing - are effective inputs to CNN systems for car detection. End-to-end prototyping is preferable to using perceptual image quality metrics when predicting CNN performance.

The diversity of biological systems inspires us to design and test increasingly specialized cameras. Soft prototyping can support experimentation with cameras optimized for motion, certain colors or shapes. It can also support experimentation using camera arrays with wide baselines that are specialized for depth estimation, or cameras that incorporate data about vehicle motion to identify still from moving objects and to use parallax to estimate depth. We hope that using end-to-end simulations in combination with task-specific performance metrics will yield insights that improve image system designs for this new generation of imaging systems.

References

- [1] H. Blasinski, J. Farrell, T. Lian, Z. Liu, and B. Wandell. Optimizing image acquisition systems for autonomous driving. *Electronic Imaging*, 2018(5):161–1–161–7, Jan. 2018.
- [2] M. Buckler, S. Jayasuriya, and A. Sampson. Reconfiguring the imaging pipeline for computer vision, 2017.
- [3] A. Carlson, K. A. Skinner, and M. Johnson-Roberson. Modeling camera effects to improve deep vision for real and synthetic data. *arXiv preprint arXiv:1803.07721*, 2018.
- [4] A. Carlson, K. A. Skinner, R. Vasudevan, and M. Johnson-Roberson. Sensor transfer: Learning optimal sensor effect image augmentation for Sim-to-Real domain adaptation. *IEEE Robotics and Automation Letters*, pages 1–1, 2019.
- [5] J. Chen, K. Venkataraman, D. Bakin, B. Rodricks, R. Gravelle, P. Rao, and Y. Ni. Digital camera imaging system simulation. *IEEE Trans. Electron Devices*, 56(11):2496–2505, Nov. 2009.
- [6] L. Duan, I. W. Tsang, D. Xu, and T.-S. Chua. Domain adaptation from multiple sources via auxiliary classifiers. In *Proceedings of the 26th Annual International Conference on Machine Learning, ICML '09*, pages 289–296, New York, NY, USA, 2009. ACM.
- [7] A. Dundar, M.-Y. Liu, T.-C. Wang, J. Zedlewski, and J. Kautz. Domain stylization: A strong, simple baseline for synthetic to real image domain adaptation. *arXiv [cs.CV]*, July 2018.
- [8] M. Everingham, L. van Gool, C. Williams, J. Winn, and Z. A. Pascal visual object classes. <http://host.robots.ox.ac.uk/pascal/VOC/>. Accessed: 2019-NA-NA.
- [9] Facebook. maskrcnn-benchmark.
- [10] J. Farrell, M. Okincha, and M. Parmar. Sensor calibration and simulation. In *Digital Photography IV*, volume 6817, page 68170R. International Society for Optics and Photonics, Mar. 2008.
- [11] J. E. Farrell, P. B. Catrysse, and B. A. Wandell. Digital camera simulation. *Appl. Opt.*, 51(4):A80–90, Feb. 2012.
- [12] J. E. Farrell and B. A. Wandell. Image systems simulation. *Handbook of Digital Imaging*, 1:373–400, 2015.
- [13] J. E. Farrell, F. Xiao, P. B. Catrysse, and B. A. Wandell. A simulation tool for evaluating digital camera image quality. In *Image Quality and System Performance*, volume 5294, pages 124–131. International Society for Optics and Photonics, Dec. 2003.
- [14] E. R. Freniere, G. Groot Gregory, and R. A. Hassler. Edge diffraction in monte carlo ray tracing. In *Optical Design and Analysis Software*, volume 3780, pages 151–158. International Society for Optics and Photonics, Sept. 1999.
- [15] A. Gaidon, Q. Wang, Y. Cabon, and E. Vig. Virtual worlds as proxy for multi-object tracking analysis. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4340–4349, 2016.
- [16] M. Geese, U. Seger, and A. Paolillo. Detection probabilities: Performance prediction for sensors of autonomous vehicles. *Electronic Imaging*, 2018.
- [17] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. *arXiv [cs.CV]*, Mar. 2017.
- [18] J. Johnson. Analysis of image forming systems. In W. L. W. R. Barry Johnson, editor, *Selected Papers on Infrared Design, Part One and Part Two.*, volume 513 of *Selected papers on infrared design. Part I and II*, page 761. Publisher, SPIE-The International Society for Optical Engineering, Bellingham, Washington, 1985.
- [19] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan. Driving in the matrix: Can virtual worlds replace Human-Generated annotations for real world tasks? Oct. 2016.
- [20] M. F. Land. THE OPTICS OF ANIMAL EYES, 2001.
- [21] C. LeGendre, X. Yu, D. Liu, J. Busch, A. Jones, S. Pattanaik, and P. Debevec. Practical multispectral lighting reproduction. *ACM Trans. Graph.*, 35(4):32, July 2016.
- [22] Y. Li, M.-Y. Liu, X. Li, M.-H. Yang, and J. Kautz. A Closed-Form solution to photorealistic image stylization: 15th european conference, munich, germany, september 8–14, 2018, proceedings, part III. In V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, editors, *Computer Vision – ECCV 2018*, volume 11207 of *Lecture Notes in Computer Science*, pages 468–483. Springer International Publishing, Cham, 2018.
- [23] T. Lian, K. J. MacKenzie, D. H. Brainard, N. P. Cottaris, and B. A. Wandell. Ray tracing 3D spectral scenes through human optics models. Mar. 2019.
- [24] Z. Liu, M. Shen, J. Zhang, S. Liu, H. Blasinski, T. Lian, and B. Wandell. A system for generating complex physically accurate sensor images for automotive applications. Feb. 2019.
- [25] M. Nielsen and M. Stokes. The creation of the sRGB ICC profile. *Color and Imaging Conference*, 1998(1):253–257, 1998.
- [26] M. Pharr, W. Jakob, and G. Humphreys. *Physically Based Rendering: From Theory to Implementation*. Morgan Kaufmann, Sept. 2016.
- [27] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time object detection with region proposal networks. In C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems 28*, pages 91–99. Curran Associates, Inc., 2015.
- [28] W. Smith. *Modern Lens Design*. McGraw Hill Professional, Oct. 2004.
- [29] J. A. Swets and D. M. Green. Applications of signal detection theory. In H. L. Pick, H. W. Leibowitz, J. E. Singer, A. Steinschneider, and H. W. Stevenson, editors, *Psychology: From Research to Practice*, pages 311–331. Springer US, Boston, MA, 1978.
- [30] J. Tremblay, A. Prakash, D. Acuna, M. Brophy, V. Jampani, C. Anil, T. To, E. Cameracci, S. Bochoon, and S. Birchfield. Training deep networks with synthetic data: Bridging the reality gap by domain randomization. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 1082–10828, June 2018.
- [31] A. Tsirikoglou, J. Kronander, M. Wrenninge, and J. Unger. Procedural modeling and physically based rendering for synthetic data generation in automotive applications. Oct. 2017.
- [32] B. A. Wandell. *Foundations of vision*. Sinauer Associates, Sunderland, MA, 1995.

- [33] M. Wrenninge and J. Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. Oct. 2018.
- [34] M. Wrenninge and J. Unger. Synscapes: A photorealistic synthetic dataset for street scene parsing. Oct. 2018.
- [35] P. Wu and T. G. Dietterich. Improving SVM accuracy by training on auxiliary data sources. In *Proceedings of the Twenty-first International Conference on Machine Learning, ICML '04*, pages 110–, New York, NY, USA, 2004. ACM.
- [36] F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, and T. Darrell. BDD100K: A diverse driving video database with scalable annotation tooling. *arXiv [cs.CV]*, May 2018.
- [37] J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired Image-to-Image translation using Cycle-Consistent adversarial networks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2242–2251, Oct. 2017.
- [38] V. Zlokolica, M. Griffin, A. Casey, D. Solera, B. Deegan, P. Denny, and B. Dever. Visual quality evaluation of the multi-camera visualization in automotive surround view systems. *Electronic Imaging*, 2018(17):1–6, 2018.