(12) INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(19) World Intellectual Property Organization
International Bureau

(43) International Publication Date
19 December 2024 (19.12.2024)

WIPO | PCT

(10) International Publication Number
WO 2024/258425 A1

(51) International Patent Classification:
H04N 23/617 (2023.01) *H04N 23/60* (2023.01)
G06N 5/00 (2023.01) *H04N 23/80* (2023.01)
G06N 20/00 (2019.01) *H04N 23/951* (2023.01)
G06T 5/00 (2024.01) *G06N 3/08* (2023.01)
G06T 19/00 (2011.01) *G06V 10/82* (2022.01)

(21) International Application Number:
PCT/US2023/068307

(22) International Filing Date:
12 June 2023 (12.06.2023)

(25) Filing Language: English

(26) Publication Language: English

(71) Applicant: **GOOGLE LLC** [US/US]; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US).

(72) Inventors: **MOTTA, Ricardo Jansson**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **IP, Katharine**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **WAN, Gordon**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **BOSSUT, Phillippe**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US). **TSENG, Bonnie**; 1600 Amphitheatre Parkway, Mountain View, California 94043 (US).

(74) Agent: **ENGLESBY, Michael K.**; 291 E. Shore Drive, Suite 200, Eagle, Idaho 83616 (US).

(81) Designated States *(unless otherwise indicated, for every kind of national protection available)*: AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CV, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IQ, IR, IS, IT, JM, JO, JP, KE, KG, KH, KN, KP, KR, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, MG, MK, MN, MU, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(84) Designated States *(unless otherwise indicated, for every kind of regional protection available)*: ARIPO (BW, CV, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SC, SD, SL, ST, SZ, TZ, UG, ZM, ZW), Eurasian (AM, AZ, BY, KG, KZ, RU, TJ, TM), European (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, ME, MK, MT, NL, NO, PL, PT, RO, RS, SE,

(54) Title: TRAINING A MACHINE-LEARNED IMAGE-PROCESSING MODULE USING SYNTHETIC DATA

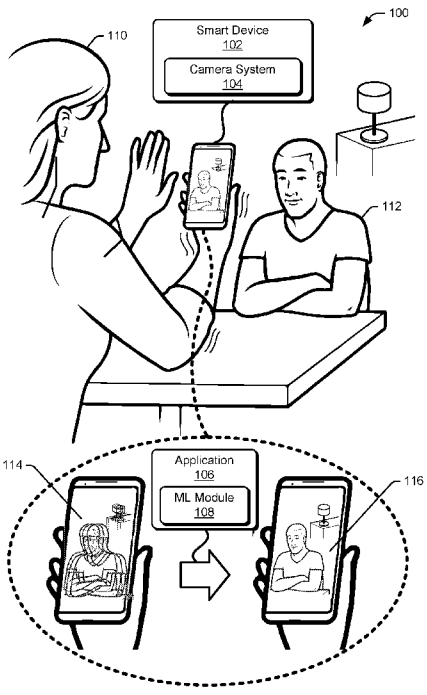


FIG. 1

(57) Abstract: Techniques and apparatuses are described that facilitate training a machine-learned image-processing module (108, 406) using synthetic data. The synthetic data (412), along with a computer model (410) of a configuration of a camera system (104), can be used to create a population of simulated images (420) by simulating image captures of a variety of scenes. These simulated image captures are themselves synthetic or artificial images that simulate what would be captured by the camera system based on the computer model. The modeled configuration can include computer models of the camera system's optics and software. These techniques and apparatuses enable training a machine-learned image-processing module for computational photography. Training with simulated images allows faster development and increased accuracy of the machine-learned image-processing module. Furthermore, the techniques and apparatuses enable a different workflow cycle for camera design, which can increase innovation and provide flexibility in developing the device/camera system.

WO 2024/258425 A1

[Continued on next page]

SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

Declarations under Rule 4.17:

— *as to applicant's entitlement to apply for and be granted a patent (Rule 4.17(ii))*

Published:

— *with international search report (Art. 21(3))*

TRAINING A MACHINE-LEARNED IMAGE-PROCESSING MODULE USING SYNTHETIC DATA

BACKGROUND

[0001] Many people use mobile devices, such as smartphones, to take pictures. Besides being readily available, the performance of cameras in mobile devices now approaches that of some dedicated cameras. Along with advancements in image-sensor technology, many mobile devices have cameras that can utilize computational photography. Computational photography allows a camera to use digital image-capture and image-processing techniques to compensate for hardware tradeoffs. Computational photography can be used to enhance images and offer features that are not available, in some cases, in dedicated cameras. As the market for smartphones becomes increasingly competitive and the smartphone development cycle shortens, however, it becomes more challenging to introduce new features associated with computational photography.

SUMMARY

[0002] This document describes techniques and apparatuses that facilitate training a machine-learned image-processing module using synthetic data. The synthetic data, along with a computer model (e.g., a camera-simulation model) of a configuration of a camera system (e.g., of a smartphone), can be used to create a population of simulated images by generating simulated image captures of a variety of scenes, which may also be computer-generated. These simulated image captures are themselves synthetic or artificial images that simulate what would be captured by the smartphone's camera system based on the computer model. The modeled configuration can include computer models of the camera system's optics (e.g., lenses), other hardware components (e.g. infra-red (IR) filters, color filters, or sensor arrays), image-processing algorithms, software, and other parameters that could be used in the camera system to generate a sensor image (e.g., a "RAW" format image), or a red-green-blue (RGB) image as would be seen by a user.

[0003] These techniques and apparatuses enable training a machine-learned image-processing algorithm or module for computational photography using, at least in part, the population of simulated images as training data. This training allows for faster development and increased accuracy of the machine-learned image-processing module. Furthermore, the techniques and apparatuses also enable a different workflow cycle for camera design, which can increase innovation and provide flexibility in developing the device/camera system. For example, once the computer model of a particular configuration is validated, engineers and designers can

easily model changes to components (e.g., lenses, filters, sensors, or algorithms) to determine how to solve problems or improve performance. In typical scenarios, this would require changes to hardware and software and manufacturing a new prototype, which would incur cost and lengthen the development cycle. Additionally, developers would need to generate a new set of training data for the machine-learned image-processing module that accounts for the changes. Thus, the described techniques and apparatuses can reduce costs, improve performance, save time, and enable more innovation.

[0004] Aspects described below include a method performed for training a machine-learned image-processing module using synthetic data. The method includes receiving parameters that describe an operational configuration of a camera system and generating, using the received parameters, a computer model of the camera system. The method also includes receiving synthetic source data describing one or more scenes and generating, by computer simulation based on the synthetic source data and the computer model of the camera system, a population of simulated images. The method further includes training a machine-learned module using training data that is at least partly based on the population of simulated images.

[0005] Aspects described below also include a system with a camera system and one or more processors. The processors are configured to process images captured by the camera system according to a machine-learned module. The machine-learned module is trained using the method of any of the previous examples.

[0006] Aspects described below also include a (computer-implemented) method performed for training a machine-learned image-processing module using synthetic data. The method includes receiving parameters that describe an operational configuration of a camera system (e.g., parameters that relate to, or are indicative of, a configuration of the camera system with respect to properties of (a) hardware components of the camera system and/or (b) software components of the camera system for capturing and processing a real-world image by the camera systems and for generating an adapted image of the real-world image for display and/or storage) and generating, using the received parameters, a computer model of the camera system.

[0007] The method also includes receiving synthetic source data describing one or more scenes (e.g., source data resulting from computer-generated imagery) and generating, by computer simulation based on (e.g., using) the synthetic source data and the computer model of the camera system, a population of simulated images (e.g., images that respectively simulate an image which would be captured by the camera system of a scene described by the source data). The method further includes training a machine-learned module using training data that is at least partly based

on the population of simulated images (e.g., training the machine-learned module using training data that includes the population of simulated images).

[0008] The trained machine-learned module may finally (e.g., when implemented to the camera system) be used to process a real-world image captured by the camera system to generate, based on the real-world image, an adapted image for display and/or storage. Before finally implementing the trained machine-learned model in the camera system the proposed method may in particular allows for re-training the machine-learned model based on (a) adjusted parameters describing another operational configuration of the camera system and (b) the same and/or other/new synthetic source data thereby facilitating provision of a machine-learned model adapted best to specifics of a camera systems not yet finally developed.

[0009] This summary is provided to introduce simplified concepts related to training a machine-learned image-processing module using synthetic data. The concepts are further described below in the Detailed Description. This summary is not intended to identify essential features of the claimed subject matter, nor is it intended for use in determining the scope of the claimed subject matter.

BRIEF DESCRIPTION OF DRAWINGS

[0010] Apparatuses for and techniques that facilitate training a machine-learned image-processing module using synthetic data are described with reference to the following drawings. The same numbers are used throughout the drawings to reference like features and components:

FIG. 1 illustrates an example environment 100 in which techniques and apparatuses that enable training a machine-learned image-processing module using synthetic data can operate;

FIG. 2 illustrates, generally, an example implementation of a camera as part of a smart device;

FIG. 3-1 illustrates, generally, operation of an example optical system of a camera system;

FIG. 3-2 illustrates operation of an example image processor of a camera system;

FIG. 4-1 illustrates, generally, an example image simulator for generating training data for a machine-learned image-processing module;

FIG. 4-2 illustrates an example hardware configuration and an example software configuration for a computer model of a camera system;

FIG. 5 depicts an example method 500 for implementing aspects of training a machine-learned image-processing module using synthetic data; and

FIG. 6 illustrates an example computing system embodying, or in which techniques may be implemented that enable use of, training a machine-learned image-processing module using synthetic data.

DETAILED DESCRIPTION

[0011] Driven to offer new features associated with smartphone cameras, developers are enhancing computational photography with artificial intelligence (e.g., machine learning). Developers can use machine-learned modules to allow a camera system to enhance image-processing abilities. For example, a machine-learned module can help the camera system perform image reconstruction, deblur images, remove unwanted objects from images, understand a scene, label elements of a scene, or reduce noise effects. The performance of a machine-learned module, however, is at least partially dependent on training data, and it can be expensive to train a machine-learned module associated with computational photography. Not only does it take a lot of training data to properly train algorithms of a machine-learned module, but the kind, variety, and quality of image data also matter. For example, one bottleneck for training the machine-learned module is the need for large datasets of real images captured with the particular camera module that is being developed.

[0012] Furthermore, shrinking development timeframes to integrate new camera systems in smartphones increase the difficulty and cost. For example, because the production cycle for smartphones has shortened (vendors often target releasing a new device yearly), there is frequently not enough time to collect enough actual real-world images from pre-release prototype smartphone versions to use for training machine-learned modules and algorithms that will be used on a new smartphone.

[0013] To produce a new camera system, developers must evaluate available hardware (e.g., measure gain, sensitivity, noise, and the image sensor), tune the image processing based on those measurements, and create tools and a development algorithm to produce the camera system. To do all of this, the developers need scenes and images, which is a problem because the characteristics of the device influence what the images look like. The developers are, thus, "stuck" with images based on an initial configuration of the system. And, if that configuration changes over the course of the development cycle, the images may not be appropriate for the task.

[0014] Current techniques may utilize a robotic camera system to capture various scenes in a lab environment. In addition to utilizing complex and expensive equipment, this process can be time-consuming. Using mannequins and robots to capture information using prototypes also presents other challenges. For example, robot-based solutions are limited by available quantities of objects, scenes, and lighting. While use of real subjects is possible, scheduling, cost, and privacy concerns can limit the use of real people.

[0015] One technique that some manufacturers use to overcome the expensive and time-consuming nature of these techniques is to use computer graphics or computer-generated imagery

(CGI) to simulate images. Images that are artificially created using CGI techniques can be useful for training purposes and can partially alleviate the problem. However, certain shortcomings become apparent when the resulting trained machine-learned modules and algorithms are applied to actual images produced by a real-world smartphone camera, because the camera hardware has characteristics that produce images that are different from those produced with CGI techniques. Even using real, non-synthetic images taken using another camera, including a previous generation of the same smartphone camera, to train a machine-learned module or algorithm can reduce the quality of image-processing results if the camera systems use different hardware.

[0016] To address these challenges and provide both improved and new features for smartphone camera systems, techniques are described for using parameters that describe an operational configuration of a camera system to generate a computer model of the camera system. The computer model is applied to synthetic source data that describes a variety of scenes to create a population of simulated images from the variety of scenes. The synthetic source data can be artificially generated (e.g., by a computer or with the aid of a computer or algorithm) based on rules, statistical modeling, simulation, or other techniques. These simulated images are thus images that simulate what would be captured by the smartphone's real-world camera system, based on the model. The modeled configuration can include computer models of the camera system's optics (e.g., lenses), other hardware components, image-processing algorithms, software, and other parameters that could be used in the camera system. The simulated images can be used to train a machine-learned image-processing algorithm or module for computational photography.

[0017] Training with the simulated images allows for faster development and increased accuracy of the machine-learned image-processing module. Furthermore, the techniques and apparatuses also enable a different workflow cycle for camera design, which can increase innovation and provide flexibility in developing the device/camera system. For example, once a computer model of a particular configuration is validated, engineers and designers can easily model changes to components. For instance, developers can "switch" lenses, filters, sensors, or algorithms in the modeled camera to solve problems or improve performance without having to make a new prototype. Further, developers can quickly generate a new set of training data for the machine-learned image-processing module that accounts for the changes.

Operating Environment

[0018] FIG. 1 illustrates an example environment 100 in which techniques and apparatuses that enable training a machine-learned image-processing module using synthetic data can operate. In the depicted environment 100, a smart device 102 includes a camera system 104 capable of

taking pictures. Although the smart device 102 is shown to be a smartphone in environment 100, the smart device 102 can generally be implemented as any type of device or object, as further described with respect to FIG. 2.

[0019] The smart device 102 can also include an application 106 and a machine-learned (ML) module 108. The application 106 can be any of a variety of applications suitable for operating the ML module 108 to provide image-processing functionality on the smart device 102. For example, as shown in FIG. 1, a user 110 may use the smart device 102 to take a photograph of a subject 112. In some cases, for example, the user 110 may not hold the smart device 102 steady while taking the photo, which can produce a blurry image 114 of the subject 112.

[0020] With the described techniques and apparatuses, the application 106 can use the ML module 108 to correct the blurry image 114. As shown in the dotted-line detail view inset of FIG. 1, the ML module 108 can operate on the blurred image 114 to produce a corrected image 116 with blurring removed. In some implementations, the user 110 may interact with the application 106 to produce the corrected image 116. In other implementations, the ML module 108 may automatically detect the blurring and produce the corrected image 116. In still other implementations, the ML module 108 may automatically detect the blurring and the user 110 may then interact with the application 106 produce the corrected image 116 (e.g., the application 106 may present the blurred image 114 with a prompt asking if the user 110 wants to edit or correct the blurred image 114). Additional features and details of the smart device 102, the camera system 104, and the ML module 108 are further described with respect to FIG. 2.

[0021] FIG. 2 illustrates, generally at 200, an example implementation of the camera system 104 as part of the smart device 102. The smart device 102 is illustrated with various non-limiting example devices, including a desktop computer 102-1, a tablet 102-2, a laptop 102-3, a television 102-4, a computing watch 102-5, computing glasses 102-6, a gaming system 102-7, a microwave 102-8, and a vehicle 102-9. Other devices may also be used, such as a home service device, a home security system or security camera, a smart speaker, a smart thermostat, a baby monitor, a Wi-Fi™ router, a drone, a trackpad, a drawing pad, a netbook, an e-reader, a home automation and control system, a wall display, or another home appliance. Note that the smart device 102 can be wearable, non-wearable but mobile, or relatively immobile (e.g., desktops and appliances). The camera system 104 can be used as a stand-alone camera system (e.g., a dedicated camera) or used with, or embedded within, many different smart devices 102 or peripherals, such as in control panels that control home appliances and systems, in automobiles to capture images inside or outside of the vehicle, or as an attachment to a laptop computer to control computing applications on the laptop.

[0022] The smart device 102 includes one or more computer processors 202 and at least one computer-readable medium 204, which includes memory media and storage media. Applications and/or an operating system (not shown) embodied as computer-readable instructions on the computer-readable medium 204 can be executed by the computer processor 202 to provide some or all of the functionalities described herein, including method(s) described with reference to FIG. 5. The computer-readable medium 204 also includes the application 106, which can further process images captured by the camera system 104. In at least some aspects, the application 106 utilizes the ML module 108 to provide one or more features of editing images captured by the camera system 104.

[0023] In some implementations, the ML module 108 relies on supervised learning and uses simulated (e.g., synthetic) data and, optionally, measured (e.g., real) data for machine-learning training purposes. Training enables the ML module 108 to learn image-processing techniques for improving the quality of captured images and for providing editing and processing features for a user (e.g., automatic editing and correction of possible defects in images or user editing with an interface). In other implementations, the ML module 108 relies on unsupervised learning.

[0024] In some implementations, the ML module 108 may be trained offline (e.g., during manufacturing or testing, prior to a user operating the smart device 102 for the first time). An example offline training procedure can use a population of simulated images as at least part of training data for training the ML module 108. The population of simulated images can be generated using a computer model of the camera system 104 and synthetic source data, including, for example, computer-generated scenes or images (e.g., using CGI techniques). The ML module 108 is then trained using the simulated images to enable various image-processing functions.

[0025] For example, the ML module 108 can be trained to recognize and remove unwanted objects from real images, deblur images when a user and/or an object being photographed is in motion, or re-light an image that appears over- or under-exposed. During the training, the ML module 108 adjusts machine-learning parameters (e.g., weights and biases) to improve results of these image-processing functions. Based on this offline training procedure, the determined weights and biases can be pre-programmed into the ML module 108 to enable subsequent image processing using machine learning. In some cases, the offline training procedure can improve image processing performance (e.g., by establishing baseline weights) and/or provide a significantly higher quantity of image data, with greater diversity of image parameters, for training the ML module 108.

[0026] Additionally or alternatively, a real-time training procedure can use image-processing capabilities of the smart device 102 to generate data for training the ML module108. In this case, a training procedure can be initiated by a user of the smart device 102. For example, the user can enter a training mode for the smart device 102 and label one or more images as training data. The images may be images as-captured by the camera system 104 or images edited by the user using editing features of the smart device 102. The user may designate features of the images as preferred (e.g., color, lighting, atmosphere). These selected and designated images are provided to the ML module 108. The ML module 108 can then determine or adjust machine-learning parameters to include the user's preferences. Using the real-time training procedure, the ML module 108 can be tailored to the user and account for changes in the user's preferences.

[0027] The ML module 108 can include one or more artificial neural networks (referred to herein as neural networks). A neural network includes a group of connected nodes (e.g., neurons or perceptrons), which are organized into one or more layers and may be implemented by hardware and/or software. As an example, the ML module 108 can include a deep neural network, which includes an input layer, an output layer, and one or more hidden layers positioned between the input layer and the output layer. The nodes of the deep neural network can be partially connected or fully connected between the layers.

[0028] In some cases, the deep neural network is a recurrent deep neural network (e.g., a long short-term memory (LSTM) recurrent deep neural network) with connections between nodes forming a cycle to retain information from a previous portion of an input data sequence for a subsequent portion of the input data sequence. In other cases, the deep neural network is a feed-forward deep neural network in which the connections between the nodes do not form a cycle. Additionally or alternatively, the ML module 108 can include another type of neural network, such as a convolutional neural network. The ML module 108 can also include one or more types of regression models, such as a single linear regression model, multiple linear regression models, logistic regression models, step-wise regression models, multi-variate adaptive regression splines, locally estimated scatterplot smoothing models, and so forth. Furthermore, a machine-learning architecture of the ML module 108 can be tailored based on available power, available memory, or computational capability.

[0029] Although shown to be included within the application 106, other implementations of the ML module 108 can be included, at least partially, within the computer-readable medium 204. In this case, at least some functionality of the ML module 108 can be performed by the computer processor 202.

[0030] The smart device 102 can also include a network interface 206 for communicating data over wired, wireless, or optical networks. For example, the network interface 206 may communicate data over a local-area network (LAN), a wireless local-area network (WLAN), a personal-area network (PAN), a wide-area network (WAN), an intranet, the Internet, a peer-to-peer network, a point-to-point network, a mesh network, and the like.

[0031] The smart device 102 can also include a display 208 to provide a kind of "viewfinder" to indicate the bounds of what subject matter the camera system 104 can capture (e.g., by what is shown on the display 208). The display 208 can also present the images captured by the camera system 104 for viewing or for editing (e.g., using the image-processing tools enabled by training a machine-learned image-processing module using synthetic data).

[0032] The camera system 104 includes an example optical system 210 and at least one image processor 212. The optical system 210 includes physical components used to generate images. The optical system 210 can include, for example, some or all of the following components (and may include one or more of each): lenses and/or lens arrays (e.g., a microlens array), image sensors, apertures, shutters, flash lighting, color filter arrays (CFA), or optical image stabilization (OIS) mechanisms. The image processor 212 can be any of a variety of processors or microprocessors suitable to convert raw image data from the image sensor into electronic data (or electronic/digital image data) useable to modify and present the image for later viewing. While not shown in FIG. 2, the camera system 104 may also include one or more computer-readable system media that can be used to store the electronic image data for processing or other data needed to operate the described techniques and apparatuses. Additional details regarding operation of the optical system 210 and the image processor 212 are further described with respect to FIGs. 3-1 and 3-2, respectively.

[0033] FIG. 3-1 illustrates, generally at 300, operation of an example optical system 302 of a camera system (e.g., the optical system 210 of the camera system 104). As shown, the example optical system 302 includes a lens 304, a CFA 306, an image sensor 308, and an analog-to-digital converter (ADC) 310. The lens 304 directs light towards the image sensor 308. For example, the lens 304 can direct light from a light source 312 that is reflected off an object 314 to the image sensor 308.

[0034] The lens 304 may include multiple lenses in various arrangements, such as in an array (e.g., a microlens array). As shown, the lens 304, or lenses, can receive photons 316 (e.g., light) reflected from the object 314 and direct focused photons 318 through the CFA 306 to the image sensor 308. The image sensor 308 can be any of a variety of devices that can convert the light provided through the lens 304 and the CFA 306 (e.g., at least a portion of the focused photons

318) into electrical signals 320 and transmit the electrical signals 320 to the ADC 310. For example, the image sensor 308 can be a charge-coupled device (CCD) sensor or a complementary metal-oxide-semiconductor (CMOS) sensor. The image sensor 308 includes a grid of many (e.g., thousands or millions) sensor pixels (also called pixels, photo-pixels, or photosites). The sensor pixels are parts of the image sensor 308 that are light-reactive and can convert the received photons 316 into the electrical signals 320 (e.g., convert charges produced by the focused photons 318 hitting the sensor pixels into voltages).

[0035] The focused photons 318 are focused using focal lengths of the component lenses, or of the microlens array, of the lens 304 to increase the quantity of photons that reach each sensor pixel through the CFA 306. The CFA 306 is a filter grid that covers each pixel or photosite of the image sensor 308 with a filter of a specific color or colors. In this way, the CFA 306 acts as a screen that only allows photons of that specific color or colors into each pixel or photosite of the image sensor 308. For example, the CFA 306 can be a red-green-blue (RGB) filter (e.g., a Bayer or Quad-Bayer filter that includes red, green, and blue filters), a red-green-blue-emerald (RGBE) filter, a cyan-yellow-green-magenta (CYGM) filter, or another type of color filter. The ADC 310 can be any of a variety of ADCs that can convert the electrical signals 320 into raw image data 322 that can be processed using an image processor 324 (e.g., digital data). The image processor 324 is further described with respect to FIG. 3-2.

[0036] In some implementations (not shown in FIG. 3-1), the example optical system 302 also includes one or more optical image stabilization (OIS) mechanisms. For example, the OIS mechanism can be or include a gyroscope that moves the lens 304 (or the microlens array) or the image sensor 308 to counteract minor movements and jitters of the smart device 102. In some implementations, the OIS mechanism can include a micro-electromechanical system (MEMS) gyroscope to detect movement and adjust the camera system accordingly. Other components that may be included with the optical system 302, but not shown in FIG. 3-1, include apertures (fixed or adjustable) through which the photons 316 can access the lens 304, shutters (mechanical or electronic–by activating and deactivating the image sensor 308 to simulate a mechanical shutter), or a flash (e.g., a lighting mechanism). In different implementations, the optical system 302 may include one or more of these not-shown components in various combinations.

[0037] FIG. 3-2 illustrates operation of an example image processor 324 of a camera system (e.g., the image processor 212 of the camera system 104). The image processor 324 can be implemented as one or more processors or microprocessors that can receive the raw image data 322 from the image sensor 308 (using, for example, the ADC 310) and convert it into an image format that can be used by an application 326 of the smart device 102. For example, the image

processor 324 can convert the raw image data 322 into a format that is compatible with or conforms to a Joint Photographic Experts Group (JPEG) standard and that can be viewed as a photographic image.

[0038] Converting the raw image data 322 can involve one or more processing steps 328. In some implementations, for example, the conversion can include one or more of demosaicing 328-1 (or debayering 328-1), which is determining a color of each pixel using an algorithm that analyzes neighboring pixels to determine the color of the subject pixel, color correction 328-2, noise reduction 328-3 (e.g., reducing shot noise, temporal noise, fixed-pattern noise, read noise), lens shade or vignette correction 328-4 (e.g., undesirable darkness at an edge or edges of an image, compared to the more central region), deblurring 328-5, or tone-mapping/tone-correction 328-6, which are algorithms that can improve the quality of HDR images when viewed using media with less dynamic range. The various processing steps 328 can be performed in any of a variety of orders, and one or more of the processing steps 328 may be performed one or more times.

[0039] The image processor 324 can also perform data encoding 330 to provide encoded image data 332 to the application 326. For example, the image processor 324 can encode the processed raw image data as a still image in any of a variety of formats, including JPEG, PNG, or TIFF, and provide that image to the application 326 for display. Similarly, the image processor 324 can encode the processed raw image data as a video in any of a variety of formats, including MPEG or AVI, and provide that video to the application 326 for replay.

[0040] FIG. 4-1 illustrates, generally at 400, an example image simulator 402 for generating training data 404 for an ML module 406. The training data 404 can be generated using simulated images created with an artificial scene generator 408 and a computer model 410 of a camera system (e.g., the camera system 104). The artificial scene generator 408 can be used to generate synthetic source data 412. The synthetic source data 412 can include CGI data 414, augmented image data 416, or a combination of both. The artificial scene generator 408 can also be used to transmit the synthetic source data 412 to the computer model 410. In some implementations, the ML module 406 can be, or be implemented as, all or part of the ML module 108.

[0041] The CGI data 414 can be generated using one or more of simple modeling rules, statistical modeling, simulation, or other techniques (e.g., in contrast to real data, which is obtained from direct measurements using a prototype of the camera system). Further, using CGI to generate the synthetic source data 412 provides flexibility to simulate different properties of the synthetic source data 412, including objects (including, for example, textures and simulated camera

positions), light sources (e.g., the number, location, and/or intensity of the light sources), and scene atmosphere or location (e.g., inside, outside, daylight, nighttime, and so forth).

[0042] The augmented image data 416 can be generated by using so-called "clean" images from another source (e.g., not from the camera system under development) and introducing variations. Augmentation is independent of how the camera works; in other words, the variations introduced are not based on operating parameters of the camera system. Rather, the variations are introduced so that an ML module can be trained using a diverse set of images. For example, a clean image set may be comprised mostly of well-exposed images, which will make the ML module less able to account for poorly exposed images. Augmentation may be used to modify some of the images in the clean image set to simulate noise that is created by reduced lighting.

[0043] This can be done by simulating a reduced lux, or illuminance, value in some of the images (lux is a measure of luminous flux per unit area). If, for example, a clean image exists at 256 lux (a typical value for a well-exposed image with a smartphone camera), the image (or image data) gets noisier as lux decreases (e.g., from 256 lux to 64 lux, 32 lux, 8 lux, 2 lux). Introducing this kind of variation can provide improved performance of the ML module. In some implementations, augmentation can be used to introduce other noise to some of the images (including to the clean images, the already-augmented images, or both). The other noise can include some or all of additive white Gaussian noise (AWGN), random noise, read noise, or shot noise.

[0044] In some implementations, all or part of the artificial scene generator 408 can be a separate component that is in electronic communication with the image simulator 402, rather than being included with the image simulator 402. In these implementations, the synthetic source data 412 is still transmitted to the computer model 410, using any of a variety of appropriate communication techniques.

[0045] The computer model 410 is generated using parameters that describe an operational configuration 418 of the camera system. In some implementations, the image simulator 402 receives the parameters that describe the operational configuration 418 and generates the computer model 410 based on the parameters. In other implementations, the computer model 410 may be generated by another entity, based on the parameters, and provided to the image simulator 402. The parameters that describe the operational configuration 418 of the camera system can include at least one hardware configuration 422, at least one software configuration 424, or one or more of both. The hardware configuration 422 and the software configuration 424 are described with additional details with reference to FIG. 4-2.

[0046] Consider FIG. 4-2, which illustrates an example hardware configuration 422 and an example software configuration 424 for a computer model of a camera system (e.g., the computer model 410). The hardware configuration 422 includes a property of at least one hardware component that is or could be or will be included with the camera system (e.g., the camera system 104). For instance, as shown in FIG. 4-2, the hardware configuration 422 includes lens properties 428, CFA properties 430, and image sensor properties 432. In some implementations, the hardware configuration 422 may not contain all of the component properties shown in FIG. 4-2, may contain additional and/or different component properties (shown or not), or may show a different quantity of component properties. Other properties that can be modeled in the hardware configuration 422, but not shown in FIG. 4-2, can include properties of components related to optical image stabilization (OIS), transport, or analog-to-digital conversion.

[0047] Example lens properties 428 can include one or more of a quantity of lenses in a compound lens package, a configuration of the compound lens package, a material of the lens or lenses (including, for example, an IR cut filter), a shape of the lens or lenses, a focal length of the lens or lenses, or a field-of-view of the lens or lenses. Other lens-related properties can include a configuration of an optical zoom mechanism (e.g., periscope zoom, voice coil motor (VCM) zoom, or multi-camera zoom), a configuration of a microlens array (and properties related to the individual lenses of the microlens array), an aperture configuration, a digital f-stop configuration, lens shade (correction) parameters, a material or color response of the color filter array, and/or a point spread function. Lens shading is a reduction in the quantity of photons (light) that strike the image sensor farther away from an optical axis. Lens shading can occur naturally by edge pixels being off-axis from the incident light, or be caused by physical obstructions or defects in the lens. The point spread function (PSF), or impulse response, of an imaging system describes the imaging system's response to a point source or point object. It expresses a degree of blurring (e.g., spreading) of a point source or object that will be present in an image of the point object.

[0048] Further, in implementations using microlenses, the aperture of the individual lenses in the microlens array can affect a sensitivity of pixels on the image sensor. For example, the aperture configuration affects the angle of incoming light (e.g., light arriving at some oblique angles can be refracted and strike a non-photosensitive portion of the sensor or even an adjacent or nearby pixel, which is known as crosstalk).

[0049] Example CFA properties 430 can include a color configuration (e.g., a red-green-blue (RGB) configuration, such as Bayer or quad-Bayer configurations, an RGB-plus-emerald (RGBE) configuration, a cyan-yellow-green-magenta (CYGM) configuration, and so forth).

Other CFA properties 430 can include a material of the filter or a material of a pigment or dye used to color elements of the CFA.

[0050] Example image sensor properties 432 can include pixel optics/resolution, sensitivity (ISO value), quantum efficiency, full well capacity, dynamic range, and noise. Pixel optics/resolution can include a size, quantity, density, and/or arrangement of pixels on the sensor. Sensitivity, or ISO value, is an adjustable value related to the sensor's sensitivity to light. ISO can be adjusted in digital cameras by adjusting an amplification of a signal generated by the sensor to make the image appear darker or brighter. Quantum efficiency is a measure of an ability of the pixels (or photosites) to absorb photons and generate electrons.

[0051] Full well capacity represents a maximum number of electrons (or photons) that can be detected in a pixel. As measured, full well capacity is often described as the capacity of each pixel to hold the electrons that are generated from the light (e.g., photons) that strikes the image sensor. Dynamic range describes a range of luminance values between brightest and darkest detectible (or perceptible) points in a digital or photographic image. For digital images, this may be expressed as a ratio between a maximum signal (e.g., electrons) that can be generated from the photons and a minimum signal. In practice, the minimum signal is likely to be a noise floor (because an "empty" (black) pixel is not typically measurable/detectable because of noise factors in the sensor, the circuitry, and/or other components). Noise parameters can be related to nearby circuitry or properties of the sensor itself, as described in this document.

[0052] The example software configuration 424 includes a property associated with at least one image-signal processing function performed by the image processor of the camera system (e.g., the image processor 212 of the camera system 104 of FIG. 2 or the image processor 324 of FIGs. 3-1 and 3-2). For instance, as shown in FIG. 4-2, the software configuration 424 includes demosaicing properties 434, color correction properties 436, and tone-mapping properties 438. Other properties 440 that can be modeled with the software configuration 424 can include properties related to the encoding process and/or analog-to-digital conversion, lens shade correction, defective pixel correction, auto-focus and/or exposure effects, noise, or gain. Further, in some implementations, the software configuration 424 may not contain all of the properties shown in FIG. 4-2 or may contain additional and/or different properties.

[0053] Example demosaicing properties 434 can include one or more algorithms that can be used to analyze electrical signals (e.g., raw image data) received from a pixel and from adjacent or nearby pixels on the image sensor to determine a color for each pixel. Any of a variety of algorithms can be used. Example algorithms include simple techniques, such as nearest-neighbor interpolation or bilinear interpolation. More-complex examples include algorithms that can

independently interpolate within individual color planes, such as Lanczos resampling, bicubic interpolation, or spline interpolation. Other suitable example algorithms include Pixel Grouping or Patterned Pixel Grouping (PPG), Adaptive Homogeneity-Directed (AHD), Variable Number of Gradients (VNG) interpolation, or Aliasing Minimization and Zipper Elimination (AMaZE).

[0054] Example color correction properties 336 can include one or more white-balance algorithms, which can help compensate for color effects introduced by lighting in order to help make colors in an image look as would be expected, regardless of lighting. For example, if photographed using a candle as a light source, an image of a plate of white rice might show the rice as a light orange color because of the candlelight. Using a white-balance algorithm, the image processor can make the sure rice looks white. Other color balancing algorithms can be used to address other lighting issues and help ensure that the colors in the image look like the colors of the real-world subject matter.

[0055] Example tone-mapping properties 438 can include one or more algorithms that can simulate an appearance of HDR images displayed in media with more-limited dynamic range by mapping one set of colors to another. Many kinds of screens (e.g., CRT or LCD monitors), projectors, and print media have a dynamic range that is insufficient for reproducing the range of light intensities that exists in real-world scenes. Tone mapping can mitigate problems related to contrast reduction from a real-world scene radiance to the displayable range but still preserve details and color appearance in the real-world scene. Example algorithms can include local tone mapping, global tone mapping, and tone correction.

[0056] Properties related to noise can include modeling noise signals to approximate camera performance or adding noise for diversity or modeling noise-reduction algorithms. For example, the properties can include a noise-simulation algorithm that can generate one or more simulated noise signals (e.g., simulated dark current noise, simulated shot noise, simulated temporal noise, simulated flicker noise, simulated fixed-pattern noise; or simulated read noise).

[0057] Multiple types of noise properties can affect the camera system, including the image sensor. Dark current noise (or dark noise) is related to exposure time and temperature. For example, during a longer exposure time, electrons can collect in the image sensor because of thermal processes (e.g., thermal dark current). These electrons add noise to the signal. Dark current level can be accounted for, but not the noise from the dark current. For shorter exposures and/or lower temperatures, dark noise is negligible compared to other kinds of noise. Shot noise, or photon noise, is a measure of randomness in the number of photons that hit a single pixel during the exposure time. Temporal noise is related to a time-based (temporal) varying of photosite (e.g., pixel) output at steady illumination. Temporal noise can be caused by the device, a power supply,

a substrate, circuitry, and/or quantization effects. Temporal noise generally increases with output (signal or photocurrent) but is more pronounced at lower signal values. Flicker noise can be associated with direct current (e.g., DC current) flow in conductive or depletion regions or in both. It is generally a result of defects and contaminants. Fixed-pattern noise is noise that presents a pattern of brightness variation across pixels, in which the pattern is fixed. Fixed pattern noise is generally caused by small differences in responsiveness (or responsitivity) of pixels in an array, which can be caused by differences in pixel size or material, or by interference from nearby circuitry. Read noise is an aggregated measure of noise signals introduced by reading the signal generated from the pixels through the circuitry from the pixels to the amplifiers to the ADC (including noise from conversion of the signal to a digital format).

[0058] Returning to FIG. 4-1, the example image simulator 402 can use the computer model 410 and the synthetic source data 412 to generate a population of simulated images 420. The population of simulated images 420 can be converted into the training data 404 using a conversion algorithm 426. The conversion algorithm 426 can be any suitable algorithm that can convert the population of simulated images 420 into a format that can be used for training the ML module 406.

Example Method

[0059] FIG. 5 depicts an example method 500 for implementing aspects of training an ML image-processing module using synthetic data in accordance with one or more implementations of the present disclosure. The method 500 is shown as sets of operations (or acts) performed but not necessarily limited to the order or combinations in which the operations are shown herein. Further, any of one or more of the operations may be repeated, combined, reorganized, or linked to provide a wide array of additional and/or alternate methods. In some implementations, the method 500 may be implemented using a computing device. In portions of the following discussion, reference may be made to the environment 100 of FIG. 1 and entities detailed in FIGs. 2 through 4-2, reference to which is made only for purposes of example. The operations are not limited to performance by one entity or multiple entities operating on one device.

[0060] At 502, parameters that describe an operational configuration of a camera system are received. In some implementations, the parameters that describe the operational configuration of the camera system can include one or more hardware configurations, one or more software configurations, or one or more of both. The hardware configuration(s) can include a property of at least one hardware component that is or could be or will be included or associated with the camera system. Similarly, the software configuration(s) can include a property associated with at

least one image-signal processing function performed by the camera system. For example, the image simulator 402 can receive the hardware configuration 422 and/or the software configuration 424, along with their respective properties, as described with reference to at least FIG. 4-1 and FIG. 4-2.

[0061] Properties included in the hardware configuration can include properties of, or associated with, a lens of the camera system, a microlens or microlens array of the camera system, a filter of the lens, a filter of the camera system, an image sensor of the camera system, or a CFA of the camera system. The properties included in the hardware configuration can also include one or more performance characteristics of the hardware component(s) or a noise signal associated with the hardware component(s). Properties included in the software configuration can include a demosaicing algorithm, a color-correction algorithm, a tone-mapping algorithm, or a noise-simulation algorithm. In some implementations, an indicator, launcher, or address for one or more of the demosaicing algorithm, the color-correction algorithm, the tone-mapping algorithm, and/or the noise-simulation algorithm can be received (e.g., rather than entire algorithms). Details and examples of these algorithms are described with references to FIG. 4-2.

[0062] At 504, a computer model of the camera system is generated using the received parameters. For example, either or both of the hardware configuration 422 (and the properties associated with the hardware configuration 422) and the software configuration 424 (and the properties associated with the software configuration 424), as described with reference to FIG. 4-1 and FIG. 4-2, can be used to generate the computer model 410.

[0063] At 506, synthetic source data describing one or more scenes is received. In some implementations, the synthetic source data can be a scene-image (or data representing the scene-image) that was generated using a CGI technique. Additionally or alternatively, the synthetic source data can be a scene-image (or data representing the scene-image) that was generated by another image capture device (e.g., another camera) and augmented using an image-processing technique. For example, the artificial scene generator 408 (or the image simulator 402) can receive the CGI data 414 and/or the augmented image data, as described with reference to at least FIG. 4-1 and FIG. 4-2. In some implementations, the synthetic source data can include data related to one or more light sources of the scene or scene-image, data related to one or more objects of the scene or scene-image (including textures or simulated camera positions), data related to one or more properties of an atmosphere of the scene, or a combination of one or more of these types of data.

[0064] At 508, a population of simulated images is generated, by computer simulation, based on the synthetic source data and the computer model of the camera system. For example,

the image simulator 402 can generate the simulated images 420. The image simulator 402 (e.g. a computer simulation tool or engine) can use the synthetic source data and the computer model 410 to generate the simulated images 420, as described with reference to at least FIG. 4-1 and 4-2.

[0065] In some implementations, generating the population of simulated images also includes modifying at least some of the simulated images using a noise-simulation algorithm. The noise-simulation algorithm can modify the simulation images by applying one or more of a simulated dark current noise, a simulated shot noise, a simulated temporal noise, a simulated flicker noise, or a simulated read noise. For example, the image simulator 402 can modify some or all of the simulated images 420 by applying some or all of the properties associated with the software configuration 424, as described with reference to FIG. 4-1 and 4-2.

[0066] Additionally or alternatively, in some implementations, generating the population of simulated images can also include modifying at least some of the simulated images using one or more properties associated with the lens of the camera system. The lens properties can include a quantity of lenses in a compound lens package, a configuration of the compound lens package, a material of the lens or lenses, a shape of the lens or lenses, a focal length of the lens or lenses, a field-of-view of the lens or lenses, a configuration of an optical zoom mechanism, a configuration of a microlens array, a configuration of an aperture, a configuration of a digital f-stop, a point spread function of the camera system, a material or color response of the CFA, or a lens shade correction algorithm. For example, the image simulator 402 can modify some or all of the simulated images 420 by applying some or all of the properties associated with the hardware configuration 422, as described with reference to FIG. 4-1 and 4-2.

[0067] At 510, an ML module is trained using training data that is at least partly based on the population of simulated images. For example, the image simulator 402 can convert the simulated images 420 into the training data 404, using the conversion algorithm 426. The training data 404 can then be used to train the ML module 406 (or the ML module 108). In some implementations, the ML module can be trained using both the training data and another population of images from another source, such as CGI-based images, real images, or simulated images from another source (e.g., a different camera model).

[0068] Additionally or alternatively, in some implementations, the ML module can be trained using the training data that is based on the population of simulated images. At least one performance metric related to the ML module can be measured (e.g., de-blurring or object removal). Based on the measuring, one or more parameters of the operational configuration can be adjusted. Another operational configuration can be generated using the adjusted parameter or parameters, and another population of simulated images can be generated based on the other

operational configuration. The ML module can then be re-trained using other training data that is at least partly based on the other population of simulated images.

[0069] Optionally, at 512, one or more images captured by the camera system can be processed according to the trained ML module. For example, the camera system 104 can use the ML module 406 (or the ML module 108) to process images captured by the camera system 104. An example of processing images captured by the camera system 104 using the trained ML module 406 (or 108), when implemented to the camera system 104 can include processing a real-world image captured by the camera system 104 to generate, based on the real-world image, an adapted image for display and/or storage. In some implementations, before finally implementing the trained machine-learned model in the camera system 104, the method 500 allows for re-training the ML module 406 (or 108) based on and adapted computer model using (a) adjusted parameters describing another operational configuration of the camera system and (b) the same and/or other/new synthetic source data. This facilitates providing a machine-learned module adapted to specifics of a camera system not yet finally developed.

Example Computing System

[0070] FIG. 6 illustrates an example computing system 600 embodying, or in which techniques may be implemented that enable use of, training an ML image-processing module using synthetic data. The example computing system 600 includes various components that can be implemented as any type of client, server, and/or computing device as described with reference to the previous figures (e.g., FIG. 2 through FIG. 5) to implement aspects of training an ML image-processing module using synthetic data in various environments (e.g., as described with reference to FIG. 1 and FIG. 2).

[0071] The computing system 600 includes communication devices 602 that enable wired and/or wireless communication of device data 604 (e.g., received data, data that is being received, data scheduled for broadcast, data packets of the data, or other data, including training data such as the training data 404). While a camera system 104 is shown as part of the computing system 600, the communication devices 602 and/or the computing system 600 can include one or more camera systems 104. The device data 604 or other device content can include configuration settings of the device, media content stored on the device, and/or information associated with a user of the device. Media content stored on the computing system 600 can include any type of audio, video, and/or image data. The computing system 600 includes one or more data inputs 606 via which any type of data, media content, and/or inputs can be received, such as training data for training an ML module (e.g., the training data 404), human utterances, user-selectable inputs

(explicit or implicit), messages, music, television media content, recorded video content, and any other type of audio, video, and/or image data received from any content and/or data source.

[0072] The computing system 600 also includes communication interfaces 608, which can be implemented as any one or more of a serial and/or parallel interface, a wireless interface, any type of network interface, a modem, and any other type of communication interface. The communication interfaces 608 provide a connection and/or communication links between the computing system 600 and a communication network by which other electronic, computing, and communication devices communicate data with the computing system 600.

[0073] The computing system 600 includes one or more processors 610 (e.g., any of microprocessors, controllers, and the like), which process various computer-executable instructions to control operation of the computing system 600. Alternatively or in addition, the computing system 600 can be implemented with any one or combination of hardware, firmware, or fixed logic circuitry that is implemented in connection with processing and control circuits, which are generally identified at 612. Although not shown, the computing system 600 can include a system bus or data transfer system that couples the various components within the device. A system bus can include any one or combination of different bus structures, such as a memory bus or memory controller, a peripheral bus, a universal serial bus, and/or a processor or local bus that utilizes any of a variety of bus architectures.

[0074] The computing system 600 also includes a computer-readable medium 614, such as one or more memory devices that enable persistent and/or non-transitory data storage (i.e., in contrast to mere signal transmission), examples of which include random access memory (RAM), non-volatile memory (e.g., any one or more of a read-only memory (ROM), flash memory, EPROM, EEPROM, etc.), and a disk storage device. The disk storage device may be implemented as any type of magnetic or optical storage device, such as a hard disk drive, a recordable and/or rewriteable compact disc (CD), any type of a digital versatile disc (DVD), and the like. The computing system 600 can also include a mass storage medium device (storage medium) 616.

[0075] The computer-readable medium 614 provides data storage mechanisms to store the device data 604, as well as various device applications 618 and any other types of information and/or data related to operational aspects of the computing system 600. For example, an operating system 620 can be maintained as a computer application with the computer-readable medium 614 and executed on the processors 610. The device applications 618 may include a device manager, such as any form of a control application, a software application, a signal-processing and control module, code that is native to a particular device, a hardware abstraction layer for a particular device, and so on.

[0076] The device applications 618 can also include any system components, engines, or managers to implement or enable training an ML image-processing module using synthetic data. In this example, the device applications 618 include the application 106 and the ML modules 108 and 406.

Conclusion

[0077] Although techniques facilitating, and apparatuses that can implement, training a machine-learned image-processing module using synthetic data have been described in language specific to features and/or methods, it is to be understood that the subject of the appended claims is not necessarily limited to the specific features or methods described. Rather, the specific features and methods are disclosed as example implementations of training a machine-learned image-processing module using synthetic data.

[0078] In the figures referenced in this document, it may be the case that only one of each numbered component (e.g., processors or converters or filters) is illustrated, but there may be one or more of any of the numbered components and the described functions and features of the numbered components may be performed by one component working alone, by one or more components working in tandem or in combination, or by one component doing a first operation and one or more other components doing one or more other operations. Further, in this document, the terms "at least one processor" or "one or more processors" (which may be referred to as "processor(s)" or "microprocessor(s)" in this document) include one processor working alone, one or more processors working in tandem or in combination, or one processor doing a first operation and one or more other processors doing one or more other operations.

[0079] Some Examples are described below.

[0080] Example 1: A method comprising: receiving parameters that describe an operational configuration of a camera system; generating, using the received parameters, a computer model of the camera system; receiving synthetic source data describing one or more scenes; generating, by computer simulation based on the synthetic source data and the computer model of the camera system, a population of simulated images; and training a machine-learned module using training data that is at least partly based on the population of simulated images.

[0081] Example 2: The method of example 1, wherein receiving the parameters that describe the operational configuration of the camera system further comprises at least one of the following: receiving at least one hardware configuration comprising a property of at least one hardware component included with the camera system; or receiving at least one software

configuration comprising a property associated with at least one image-signal processing function performed by the camera system.

[0082] Example 3: The method of example 2, wherein: receiving the at least one software configuration further comprises receiving one or more of: a demosaicing algorithm; a color-correction algorithm; a tone-mapping algorithm; or a noise-simulation algorithm.

[0083] Example 4: The method of example 3, further comprising: receiving the noise-simulation algorithm, the noise-simulation algorithm configured to generate one or more of: simulated dark current noise; simulated shot noise; simulated temporal noise; simulated flicker noise; simulated fixed-pattern noise; or simulated read noise.

[0084] Example 5: The method of example 4, further comprising: modifying at least some of the simulated images by applying at least one of: the simulated dark current noise; the simulated shot noise; the simulated temporal noise; the simulated flicker noise; simulated fixed-pattern noise; or the simulated read noise.

[0085] Example 6: The method of example 2, wherein: receiving the at least one hardware configuration further comprises receiving one or more of: a property associated with a lens of the camera system; a property associated with a color filter array of the camera system; or a property associated with an image sensor of the camera system.

[0086] Example 7: The method of example 6, wherein: receiving the property associated with the lens of the camera system further comprises receiving at least one of a property associated with: a quantity of lenses in a compound lens package; a configuration of the compound lens package; a material of the lens or lenses; a shape of the lens or lenses; a focal length of the lens or lenses; a field-of-view of the lens or lenses; a configuration of an optical zoom mechanism; a configuration of a microlens array; a configuration of an aperture; a configuration of a digital f-stop; a point spread function; a material or color response of the color filter array; or a lens shade correction algorithm.

[0087] Example 8: The method of example 7, further comprising: modifying at least some of the simulated images by applying at least one of the properties associated with: the quantity of lenses in the compound lens package; the configuration of the compound lens package; the material of the lens or lenses; the shape of the lens or lenses; the focal length of the lens or lenses; the field-of-view of the lens or lenses; the configuration of the optical zoom mechanism; the configuration of the microlens array; the configuration of the aperture; the configuration of the digital f-stop; the point spread function; the material or color response of the color filter array; or the lens shade correction algorithm.

[0088] Example 9: The method of any previous example, wherein: receiving the synthetic source data describing the one or more scenes further comprises: receiving a scene-image generated using a computer graphic imaging technique; or receiving a scene-image generated by another image capture device and augmented using an image-processing technique.

[0089] Example 10: The method of any previous example, wherein: receiving the synthetic source data describing the one or more scenes further comprises receiving one or more of: data related to one or more light sources of the one or more scenes; data related to one or more objects of the one or more scenes; or data related to one or more properties of an atmosphere of the one or more scenes.

[0090] Example 11: The method of any previous example, wherein: training the machine-learned module further comprises training the machine-learned module using the training data that is at least partly based on the population of simulated images and at least partly on other training data that is based on another population of images from another source.

[0091] Example 12: The method of any previous example, further comprising: training the machine-learned module using training data based on the population of simulated images; measuring at least one performance metric related to the machine-learned module; adjusting one or more parameters of the operational configuration, based on the measuring; generating another computer model of the camera system using at least the one or more adjusted parameters; generating another population of simulated images, based on the other computer model; and re-training the machine-learned module using other training data that is at least partly based on the other population of simulated images.

[0092] Example 13: The method of any previous example, further comprising: processing one or more images captured by the camera system according to the trained machine-learned module.

[0093] Example 14: A system comprising: a camera system; and one or more processors, the processors configured to process images captured by the camera system according to a machine-learned module, the machine-learned module trained using the method of any of examples 1-13.

[0094] Example 15: The system of example 14, wherein the camera system and the one or more processors are integrated within a smartphone.

[0095] Example 16: An apparatus comprising a camera system and a processor. The processor is configured to process images captured by the camera system according to a machine-learned module for image enhancement. The machine-learned module has been trained using training data that has been generated by: receiving parameters that describe an operational

configuration of the camera; generating, using the received parameters, a computer model of the camera; receiving synthetic source data describing a scene; and generating, by computer simulation based on the synthetic source data and the computer model of the camera, a population of simulated image captures. The training data is formed using at least a portion of the population of simulated image captures.

[0096] Example 17: An apparatus for image processing, comprising: a camera system; and one or more processors, the processors configured to process images captured by the camera system according to a machine-learned module, the machine-learned module trained using training data, the training data generated by: receiving parameters that describe an operational configuration of the camera system; generating, using the received parameters, a computer model of the camera system; receiving synthetic source data describing a scene; generating, by computer simulation based on the synthetic source data and the computer model of the camera system, a population of simulated images; and forming the training data using at least a portion of the population of simulated images.

[0097] Example 18: A method comprising: receiving parameters that that relate to, or are indicative of, a configuration of the camera system with respect to properties of (a) hardware components of the camera system and/or (b) software components of the camera system for capturing and processing a real-world image by the camera system and for generating an adapted image of the real-world image for display and/or storage; generating, using the received parameters, a computer model of the camera system; receiving synthetic source data resulting from computer-generated imagery; generating, by computer simulation using the synthetic source data and the computer model of the camera system, a population of simulated images; training a machine-learned module using training data that includes the population of simulated images; and using the trained machine-learned module with the camera system to process a real-world image captured by the camera system to generate, based on the real-world image, an adapted image for display and/or storage.

CLAIMS

What is claimed is:

1. A method comprising:

receiving parameters that describe an operational configuration of a camera system;

generating, using the received parameters, a computer model of the camera system;

receiving synthetic source data describing one or more scenes;

generating, by computer simulation based on the synthetic source data and the computer model of the camera system, a population of simulated images; and

training a machine-learned module using training data that is at least partly based on the population of simulated images.

2. The method of claim 1, wherein receiving the parameters that describe the operational configuration of the camera system further comprises at least one of the following:

receiving at least one hardware configuration comprising a property of at least one hardware component included with the camera system; or

receiving at least one software configuration comprising a property associated with at least one image-signal processing function performed by the camera system.

3. The method of claim 2, wherein:

receiving the at least one software configuration further comprises receiving one or more of:

a demosaicing algorithm;

a color-correction algorithm;

a tone-mapping algorithm; or

a noise-simulation algorithm.

4. The method of claim 3, further comprising:

receiving the noise-simulation algorithm, the noise-simulation algorithm configured to generate one or more of:

simulated dark current noise;

simulated shot noise;

simulated temporal noise;

simulated flicker noise;

simulated fixed-pattern noise; or

simulated read noise.

5. The method of claim 4, further comprising:

modifying at least some of the simulated images by applying at least one of:

the simulated dark current noise;

the simulated shot noise;

the simulated temporal noise;

the simulated flicker noise;

the simulated fixed-pattern noise; or

the simulated read noise.

6. The method of any one of claims 2 to 5, wherein:

receiving the at least one hardware configuration further comprises receiving one or more of:

a property associated with a lens of the camera system;

a property associated with a color filter array of the camera system; or

a property associated with an image sensor of the camera system.

7. The method of claim 6, wherein:

receiving the property associated with the lens of the camera system further comprises receiving at least one of a property associated with:

a quantity of lenses in a compound lens package;

a configuration of the compound lens package;

a material of the lens or lenses;

a shape of the lens or lenses;

a focal length of the lens or lenses;

a field-of-view of the lens or lenses;

a configuration of an optical zoom mechanism;

a configuration of a microlens array;

a configuration of an aperture;

a configuration of a digital f-stop;

a point spread function;

a material or color response of the color filter array; or

a lens shade correction algorithm.

8. The method of claim 7, further comprising:

modifying at least some of the simulated images by applying at least one of the properties associated with:

the quantity of lenses in the compound lens package;

the configuration of the compound lens package;

the material of the lens or lenses;

the shape of the lens or lenses;

the focal length of the lens or lenses;

the field-of-view of the lens or lenses;

the configuration of the optical zoom mechanism;

the configuration of the microlens array;

the configuration of the aperture;

the configuration of the digital f-stop;

the point spread function;

the material or color response of the color filter array; or

the lens shade correction algorithm.

9. The method of any one of claims 1-8, wherein:

the synthetic source data describing the one or more scenes further comprises:

a scene-image generated using a computer graphic imaging technique; or

a scene-image generated by another image capture device and augmented using an image-processing technique.

10. The method of any one of claims 1-9, wherein:

the synthetic source data describing the one or more scenes further comprises one or more of:

data related to one or more light sources of the one or more scenes;

data related to one or more objects of the one or more scenes; or

data related to one or more properties of an atmosphere of the one or more scenes.

11. The method of any one of claims 1-10, wherein:

training the machine-learned module further comprises training the machine-learned module using the training data that is at least partly based on the population of simulated images and at least partly on other training data that is based on another population of images from another source.

12. The method of any one of claims 1-11, further comprising:

training the machine-learned module using training data based on the population of simulated images;

measuring at least one performance metric related to the machine-learned module;

adjusting one or more parameters of the operational configuration, based on the measuring;

generating an adapted computer model of the camera system, using at least the one or more adjusted parameters;

generating another population of simulated images, based on the adapted computer model; and

re-training the machine-learned module using other training data that is at least partly based on the other population of simulated images.

13. The method of any one of claims 1-12, further comprising:

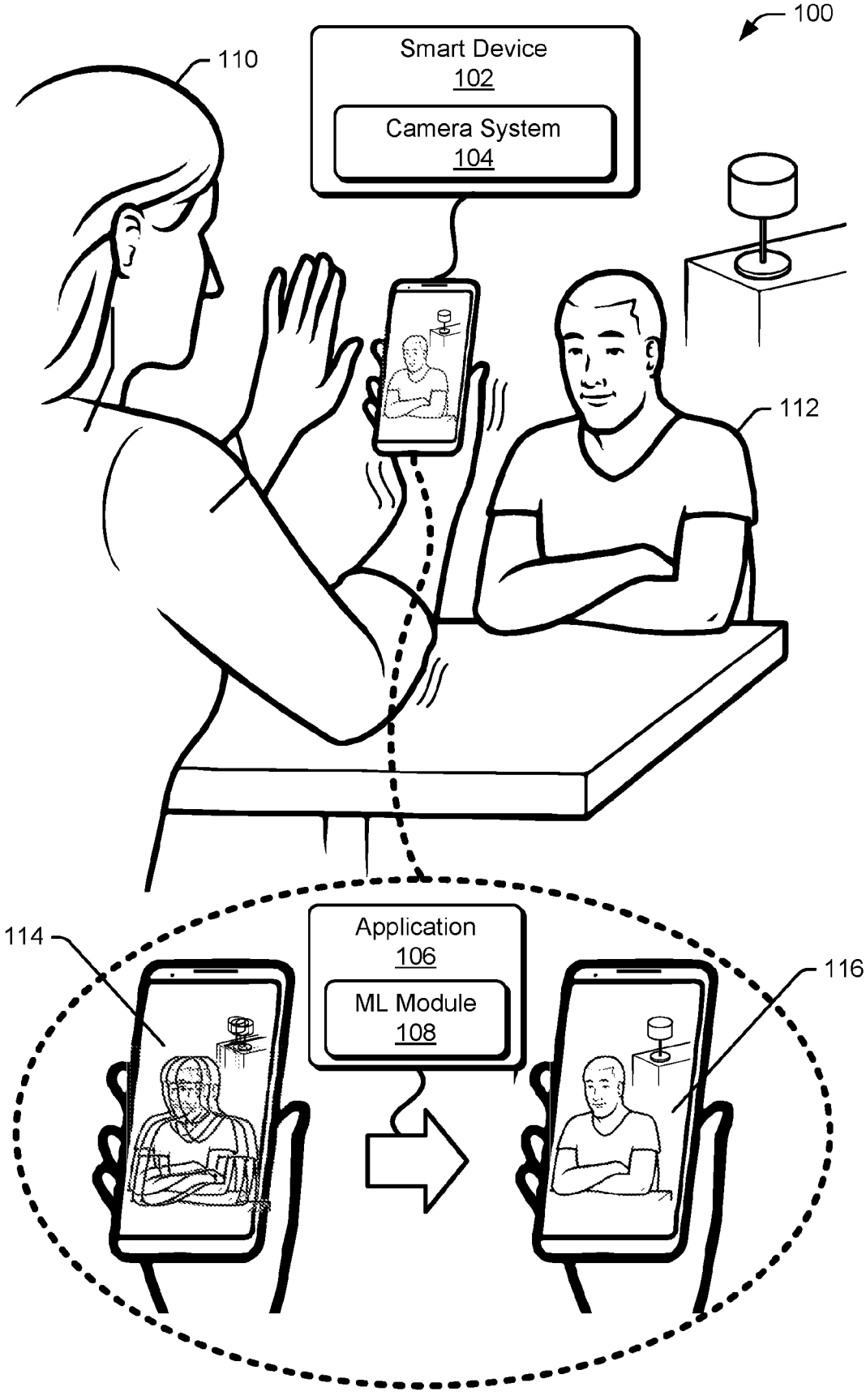
processing one or more images captured by the camera system according to the trained machine-learned module.

14. A system comprising:

a camera system; and

one or more processors, the processors configured to process images captured by the camera system according to a machine-learned module, the machine-learned module trained using the method of any one of claims 1-13.

15. The system of claim 14, wherein the camera system and the one or more processors are integrated within a smartphone.


100
110
Smart Device
102
Camera System
104
112
114
Application
106
ML Module
108
116

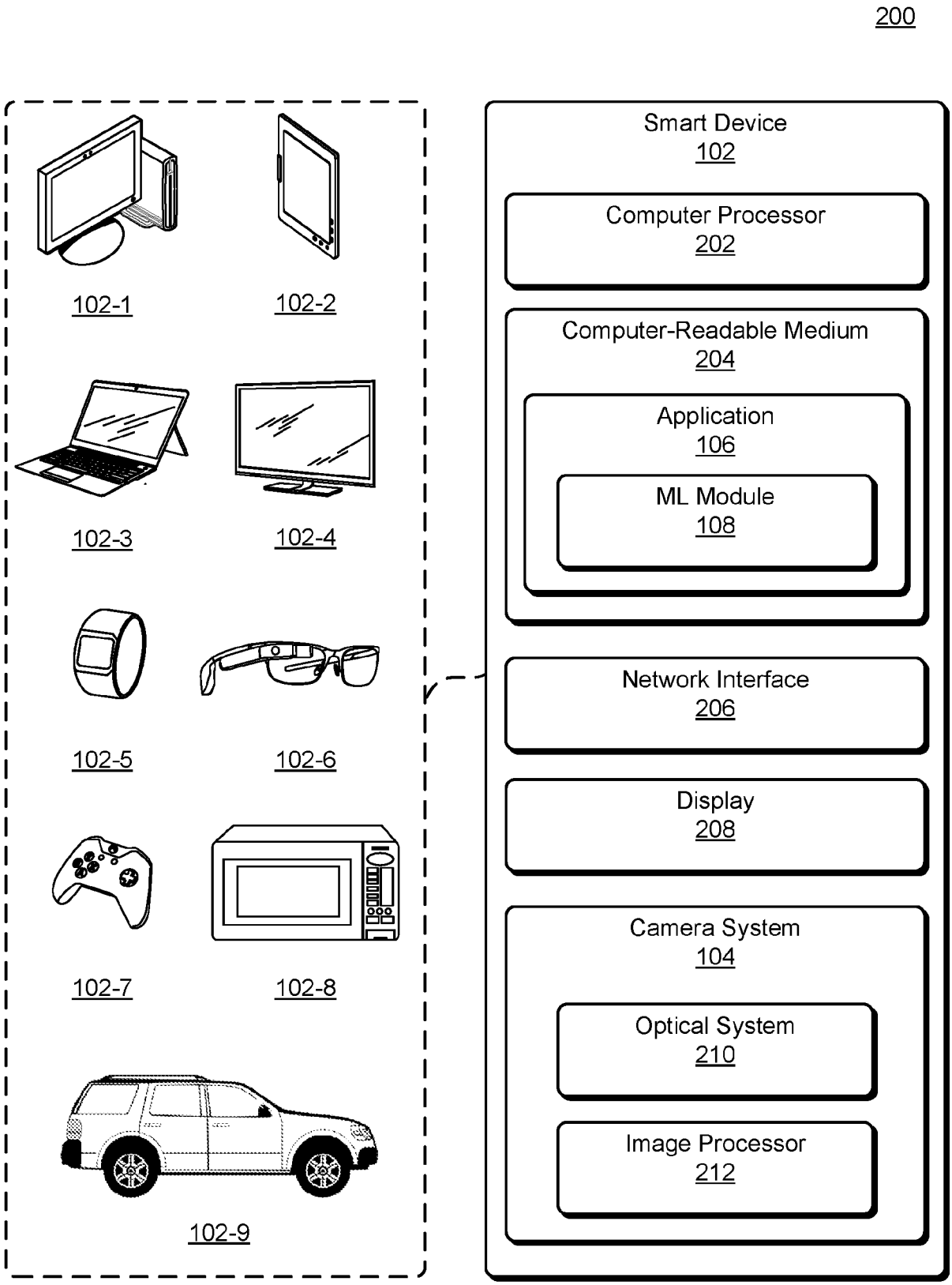

200
102-1
102-2
102-3
102-4
102-5
102-6
102-7
102-8
102-9
Smart Device
102
Computer Processor
202
Computer-Readable Medium
204
Application
106
ML Module
108
Network Interface
206
Display
208
Camera System
104
Optical System
210
Image Processor
212

FIG. 2

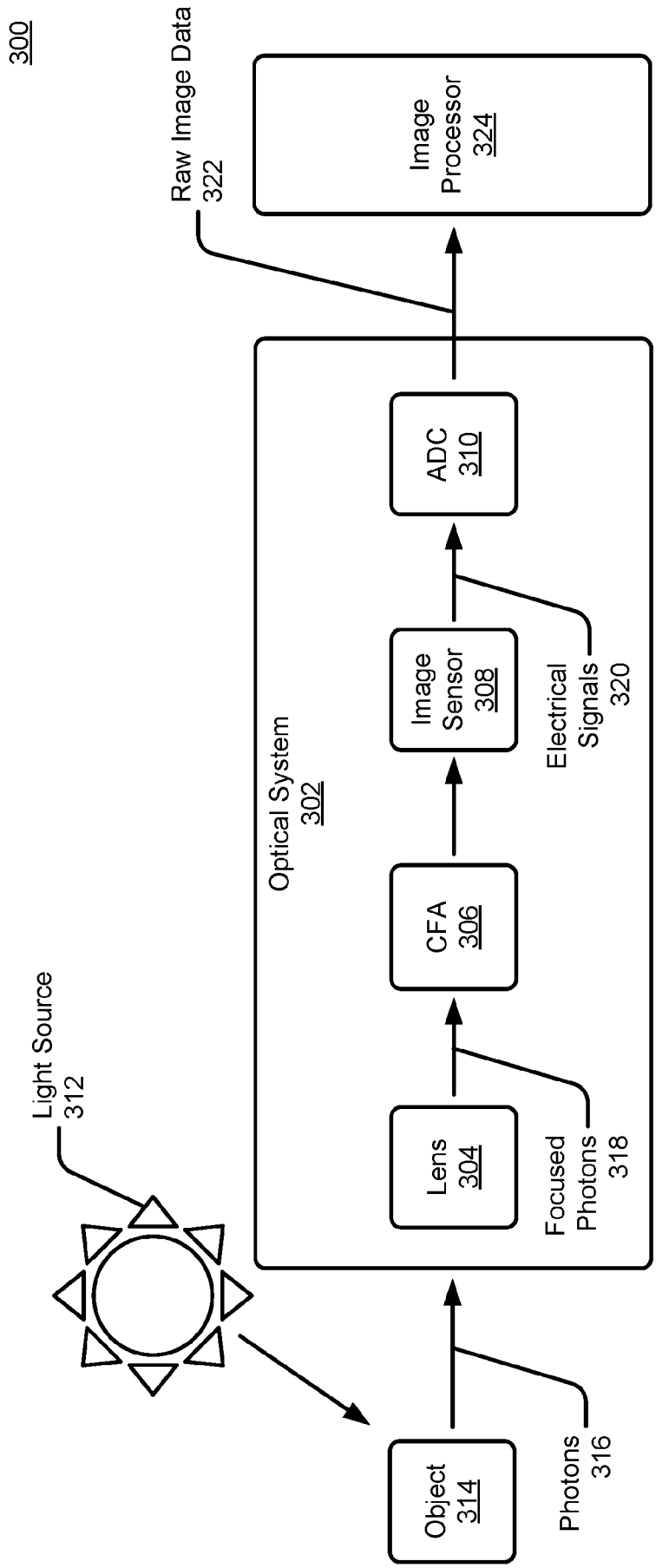

300
Raw Image Data
322
Image Processor
324
ADC
310
Image Sensor
308
Electrical Signals
320
Optical System
302
CFA
306
Lens
304
Focused Photons
318
Light Source
312
Object
314
Photons
316

FIG. 3-1

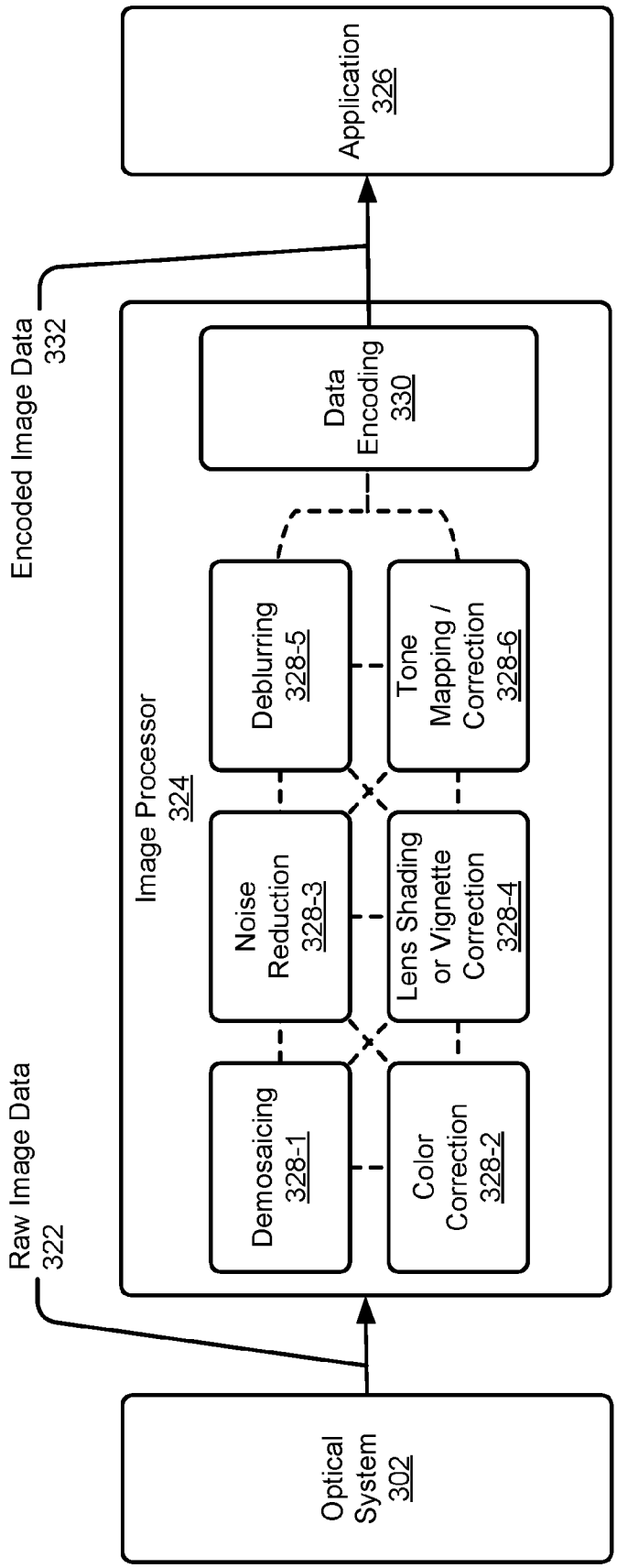

Application
326
Encoded Image Data
332
Image Processor
324
Data
Encoding
330
Deblurring
328-5
Tone
Mapping /
Correction
328-6
Noise
Reduction
328-3
Lens Shading
or Vignette
Correction
328-4
Demosaicing
328-1
Color
Correction
328-2
Raw Image Data
322
Optical
System
302

FIG. 3-2

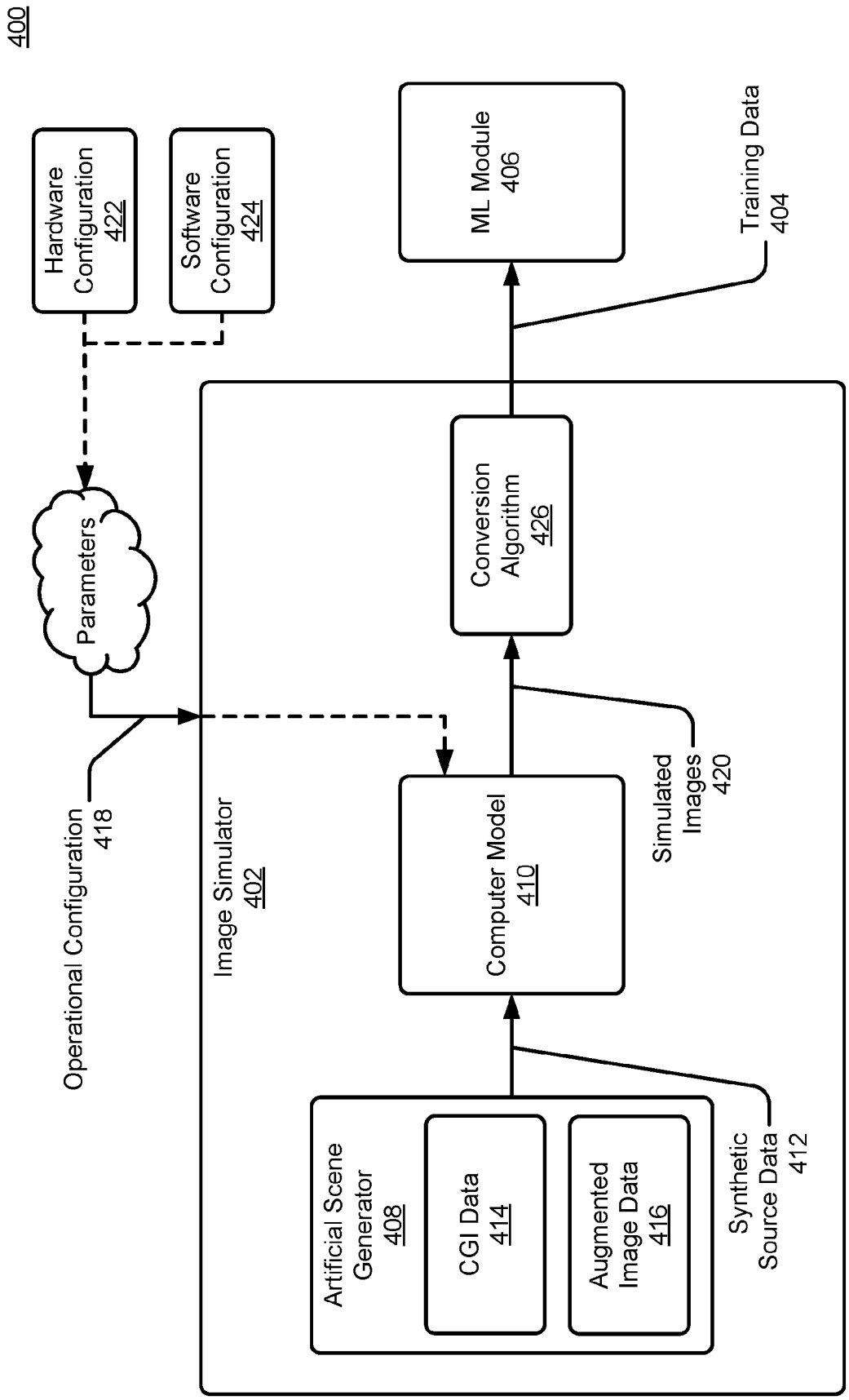

400
Hardware Configuration 422
Software Configuration 424
ML Module 406
Training Data 404
Parameters
Conversion Algorithm 426
Operational Configuration 418
Image Simulator 402
Computer Model 410
Simulated Images 420
Artificial Scene Generator 408
CGI Data 414
Augmented Image Data 416
Synthetic Source Data 412

FIG. 4-1

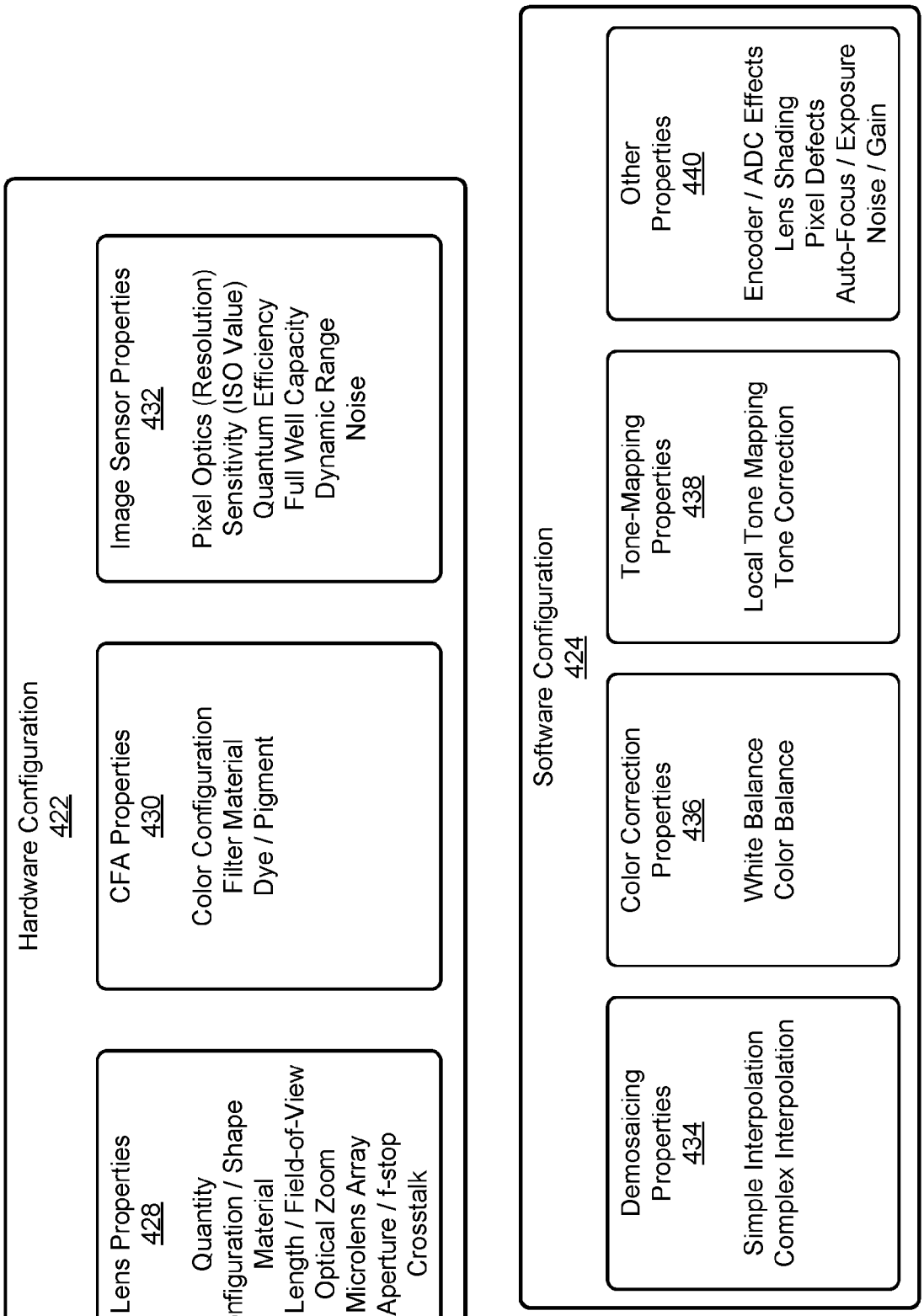

Hardware Configuration
422
Lens Properties
428
Quantity
Configuration / Shape
Material
Focal Length / Field-of-View
Optical Zoom
Microlens Array
Aperture / f-stop
Crosstalk
CFA Properties
430
Color Configuration
Filter Material
Dye / Pigment
Image Sensor Properties
432
Pixel Optics (Resolution)
Sensitivity (ISO Value)
Quantum Efficiency
Full Well Capacity
Dynamic Range
Noise
Software Configuration
424
Demosaicing
Properties
434
Simple Interpolation
Complex Interpolation
Color Correction
Properties
436
White Balance
Color Balance
Tone-Mapping
Properties
438
Local Tone Mapping
Tone Correction
Other
Properties
440
Encoder / ADC Effects
Lens Shading
Pixel Defects
Auto-Focus / Exposure
Noise / Gain

FIG. 4-2

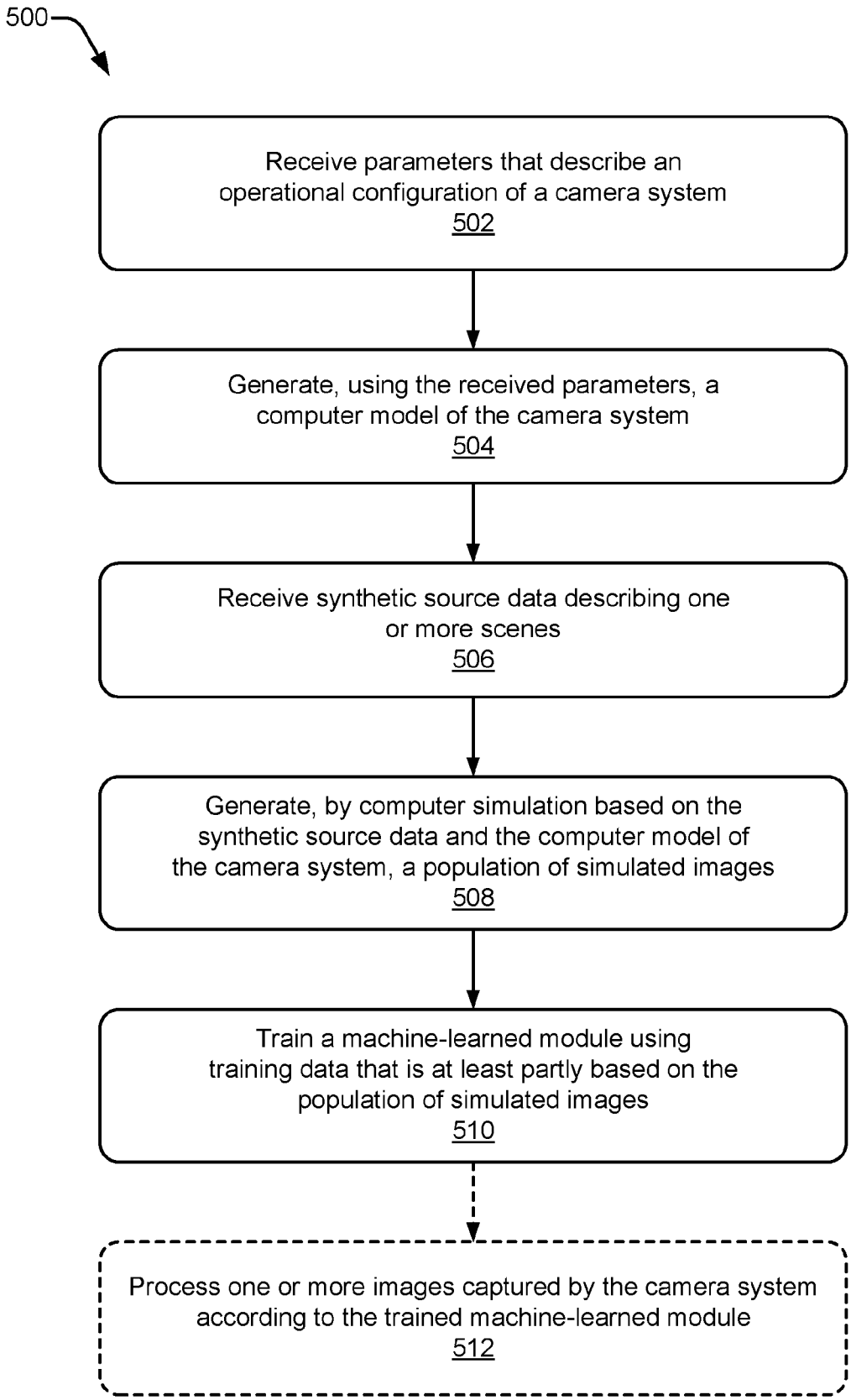

500
Receive parameters that describe an operational configuration of a camera system
502
Generate, using the received parameters, a computer model of the camera system
504
Receive synthetic source data describing one or more scenes
506
Generate, by computer simulation based on the synthetic source data and the computer model of the camera system, a population of simulated images
508
Train a machine-learned module using training data that is at least partly based on the population of simulated images
510
Process one or more images captured by the camera system according to the trained machine-learned module
512

FIG. 5

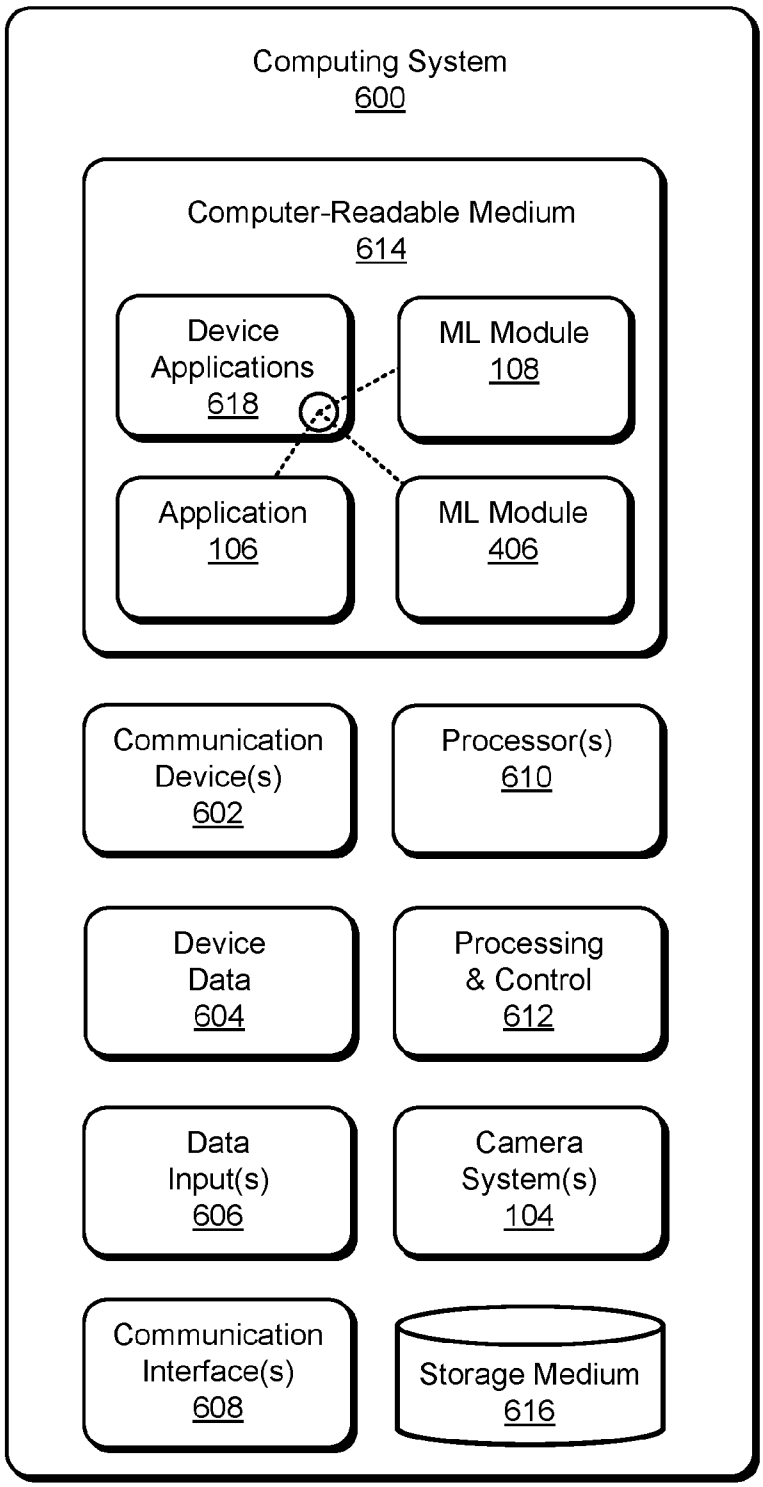

Computing System
600
Computer-Readable Medium
614
Device Applications
618
ML Module
108
Application
106
ML Module
406
Communication Device(s)
602
Processor(s)
610
Device Data
604
Processing & Control
612
Data Input(s)
606
Camera System(s)
104
Communication Interface(s)
608
Storage Medium
616

FIG. 6

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2023/068307

A. CLASSIFICATION OF SUBJECT MATTER
INV. H04N23/617 G06N5/00 G06N20/00 G06T5/00 G06T19/00
H04N23/60 H04N23/80 H04N23/951 G06N3/08 G06V10/82
ADD.
According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED
Minimum documentation searched (classification system followed by classification symbols)
H04N G06T G06N G06V

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)
EPO-Internal

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 2020/342652 A1 (ROWELL ADAM [US] ET AL) 29 October 2020 (2020-10-29) paragraph [0019] - paragraph [0173]	1-15
X	US 2021/390375 A1 (LUO CHENCHI [US] ET AL) 16 December 2021 (2021-12-16) paragraph [0030] - paragraph [0078]	1-15
X	US 2019/156151 A1 (WRENNINGE CARL MAGNUS [US] ET AL) 23 May 2019 (2019-05-23) paragraph [0013] - paragraph [0093]	1-15
X	US 2021/097341 A1 (MARCO TIMOTHY [US] ET AL) 1 April 2021 (2021-04-01) paragraph [0037] - paragraph [0081]	1,2,6-15

-/--

[X] Further documents are listed in the continuation of Box C.
[X] See patent family annex.

* Special categories of cited documents :

"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier application or patent but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance;; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance;; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
20 December 2023	19/01/2024

Name and mailing address of the ISA/	Authorized officer
European Patent Office, P.B. 5818 Patentlaan 2 NL - 2280 HV Rijswijk Tel. (+31-70) 340-2040, Fax: (+31-70) 340-3016	Gil Zamorano, Gunnar

1

Form PCT/ISA/210 (second sheet) (April 2005)

INTERNATIONAL SEARCH REPORT

International application No
PCT/US2023/068307

C(Continuation). DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	ELMQUIST ASHER ET AL: "Modeling Cameras for Autonomous Vehicle and Robot Simulation: An Overview", IEEE SENSORS JOURNAL, IEEE, USA, vol. 21, no. 22, 8 October 2021 (2021-10-08), pages 25547-25560, XP011887935, ISSN: 1530-437X, DOI: 10.1109/JSEN.2021.3118952 [retrieved on 2021-11-11] the whole document	1-15
X	US 11 256 958 B1 (SUBBIAH MELANIE S [US] ET AL) 22 February 2022 (2022-02-22) column 3, line 35 - column 18, line 64	1,2,6-14

1

INTERNATIONAL SEARCH REPORT

Information on patent family members

International application No
PCT/US2023/068307

Patent document cited in search report	Publication date	Patent family member(s)	Publication date
US 2020342652 A1	29-10-2020	NONE	
US 2021390375 A1	16-12-2021	US 2021390375 A1	16-12-2021
		WO 2021256812 A1	23-12-2021
US 2019156151 A1	23-05-2019	US 10235601 B1	19-03-2019
		US 2019156151 A1	23-05-2019
US 2021097341 A1	01-04-2021	US 2021097341 A1	01-04-2021
		WO 2021067234 A1	08-04-2021
US 11256958 B1	22-02-2022	NONE	