Physically based synthetic image generation for machine learning - a review of pertinent literature

Dominik Schraml

Steinbeis Qualitätssicherung und Bildverarbeitung, Werner-von-Siemens Straße 9, Ilmenau, Germany

ABSTRACT

The term deep learning is almost on everyone's lips these days, in the area of computer vision mainly because of the great advances deep learning approaches have made amongst others in object detection and classification. For general object localization or classification tasks there do exist several giant databases containing several millions of labeled images and several thousands of different labels like COCO and ImageNet. In contrast in industrial applications like quality inspection there do hardly ever exist such training data not only for reasons of confidentiality of trade secrets. An obvious way to remedy this deficiency is the synthetic creation of image data. Physically based rendering attempts to achieve photorealistic images by accurately simulating the flow of light of the real world according to various physical laws. Therefor multiple techniques like Ray Tracing and Path Tracing have been implemented and are becoming increasingly widespread as hardware performance increases. The intent of this article is to give a wide but nevertheless preferably comprehensive overview of which approaches have been pursued in recent literature to generate realistic synthetic training images. The development of various rendering methods from rasterization to bidirectional monte carlo path tracing is outlined, as well as their differences and use. Along with the terminology a few mathematical foundations like the Bidirectional Reflectance Distribution Function (BRDF) are briefly described. Altogether special concern is given to industrial data and quality control, comparing literature and the practical application of its results.

Keywords: literature review, synthetic image generation, physically based rendering, deep learning, computer vision, machine learning

1. INTRODUCTION

The term deep learning is almost on everyone's lips these days, in the area of computer vision manly because of the great advances deep learning approaches have made amongst others in object detection and classification, since Alex Krizhevsky's deep neural network achieved a tremendously better result in ImageNet LSVRC-2010 contest than any other "classic" machine learning approach.[1]

Since then the research in the direction of deep learning to solve diverse computer vision tasks accelerated enormously. Successes have been achieved in numerous areas including Object detection, Image Classification, Instance Segmentation and even artificial Image generation.Today deep neural networks are slowed down mainly by the amount and quality of the available data. For this reason the focus of current research lies on the generation of data on the one hand to increase the data quantity and on the other hand to bypass the manual labeling necessity.

This review paper shall make a contribution to current research by providing a comprehensive overview which approaches have been pursued in recent literature to generate realistic synthetic training images and which successes therewith could have been achieved. Thereby the focus is on the area of application of industrial quality assessment.

Further author information:
E-mail: dominik.schraml@quick-image.de, Telephone: +49 3677 46905916

Photonics and Education in Measurement Science 2019, edited by Maik Rosenberger,
Paul-Gerald Dittrich, Bernhard Zagar, Proc. of SPIE Vol. 11144, 111440J
© 2019 SPIE · CCC code: 0277-786X/19/$21 · doi: 10.1117/12.2533485

2. DEFINITION OF TERMS AND METHODOLOGY OF LITERATURE RESEARCH

Rendering is the process of producing an image from the description of a 3D scene. "The goal of photorealistic rendering is to create an image of a 3D scene that is indistinguishable from a photograph of the same scene".[2] To this quote Pharr et al. note that the word "indistinguishable" itself is inaccurate, because it subjectively depends on the observer. But the goal is pursued by using natural laws of material-dependent light propagation reflection and refraction to model the scene appearance in a physically correct way.

Bidirectional Reflectance Distribution Function (BRDF):

$$f_r\left(p, \omega_0, \omega_i\right) = \frac{dL\left(p, \omega_0\right)}{dE\left(p, \omega_i\right)} = \frac{dL\left(p, \omega_0\right)}{L_i\left(p, \omega_i\right) \cos\theta_i d\omega_i} \tag{1}$$

The BRDF is a function for the reflection behavior of surfaces of a material at any angle of incidence. For each light beam incident on the material at a given angle of incidence, it provides the quotient of radiation density and radiance for each light beam emitted.

Generally the amount of light that reaches the camera from a point on an object is given by the sum of light emitted by the object the amount of reflected light and additionally the amount of light emitted by the object, if it is itself a light source. This idea transformed into a mathematical expression leads to the *rendering equation* (also known as *light transport equation*, which Kajiya first formulated 1986.[3]

Rendering Equation

$$L\left(p, \omega_0\right) = \int_{S^2} f\left(p, \omega_0, \omega_i\right) L_i\left(p, \omega_i\right) \left|\cos\theta_i\right| d\omega_i \tag{2}$$

The outgoing radiance $L\left(p, \omega_0\right)$ from a point p in direction ω_0 is equal to the emitted radiance at that point in that direction, $L_i\left(p, \omega_i\right)$ plus the incident radiance from all directions on the sphere S^2 around p scaled by the BSDF[*] $f\left(p, \omega_0, \omega_i\right)$ and a cosine term.[2–4]

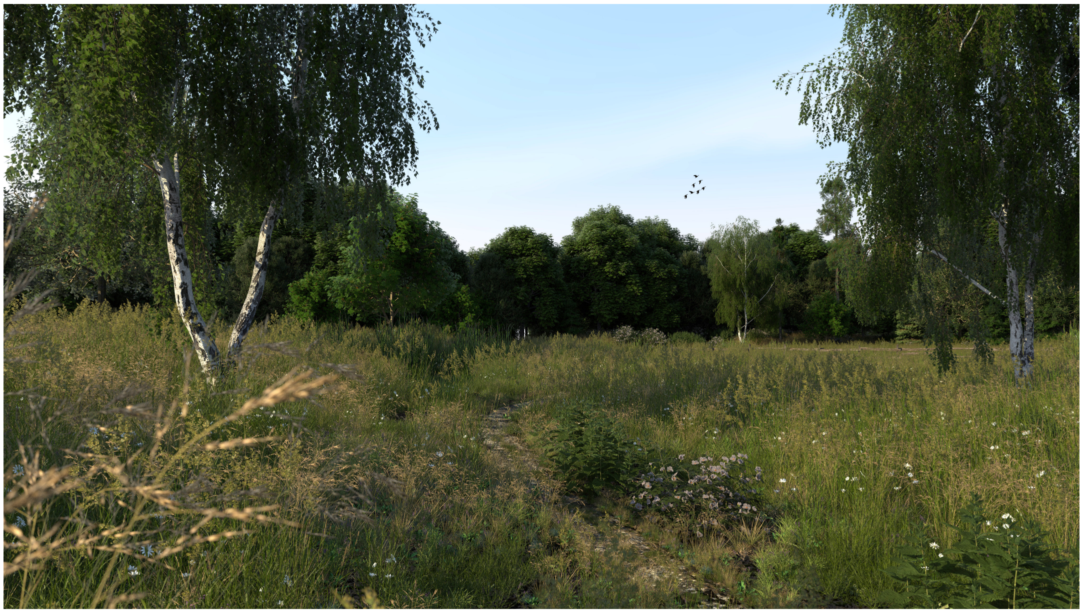

Figure 1: Computer generated image using physically based rendering like path tracing from Pharr et al.[2]

[*] Pharr et al. the BRDF generalized to all sorts of scattering Bidirectional Scattering Distribution Function (BSDF)

Ray Tracing

In ray tracing a ray is sent out from the virtual camera into the scene and traced until it intersects with a solid body. At this point a ray is cast to each of the light sources in the scene to calculate illumination and surface shading for the intersection point. Only if the surface is transparent the ray is sent out further into the scene, at the angle of refraction. If the surface is reflective a ray is radiated at the corresponding angle of reflection away from the object. Consequentially ray tracing comes closer to reality than triangle rasterization, but is no simulation of reality.[2,5]

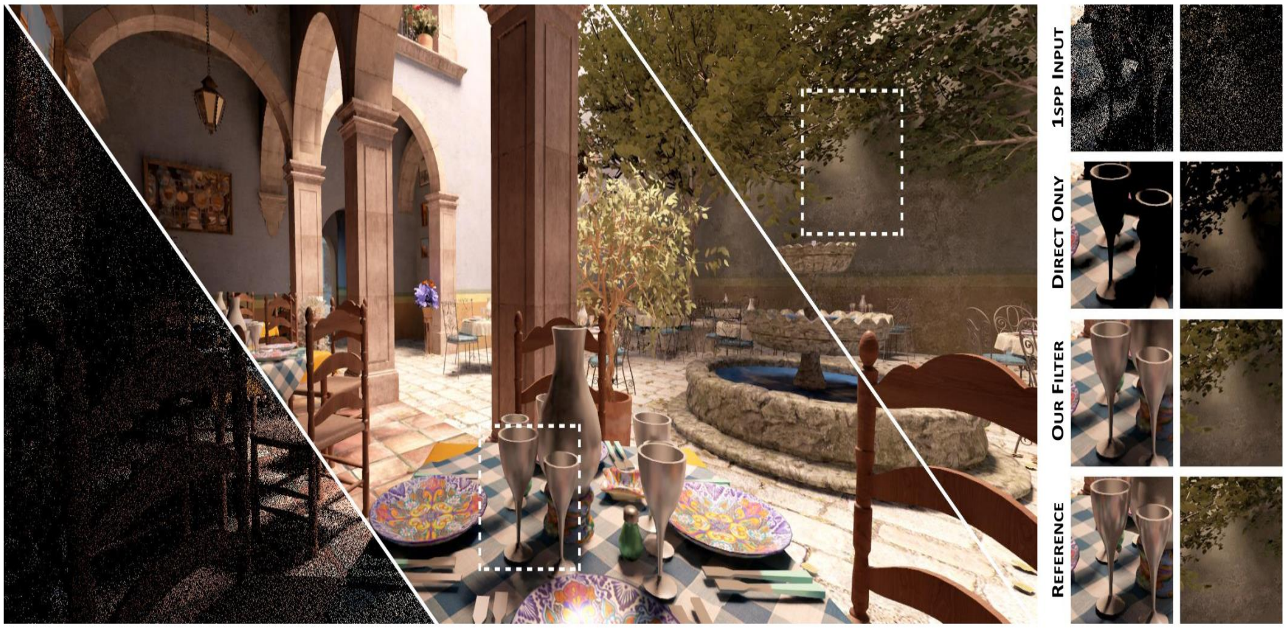

Figure 2: Representation of Monte Carlo Path Tracing with spatiotemporal variance-guided filtering to improve performance and rendering time from Schied et al.[6][†]

Path Tracing

A path tracer sends out hundreds or several thousands of rays for each pixel to be rendered. When it hits a surface it doesn' t trace a path to every light source, instead it bounces the ray off the surface and keeps bouncing it until it hits a light source or exhausts some bounce limit. Depending on the surfaces it refractured, the energy of the ray can be considered close to zero then. It then calculates the amount of light transferred all the way to the pixel, including any color information gathered from surfaces along the way. The values calculated from all the paths that were traced into the scene are considered (added up and averaged) to get the final pixel color value. If this approach uses a random draw to calculate the direction the ray has to go, when it hits a surface and is reflected, this technique is called Monte Carlo Path Tracing. [‡] To sum it all up, path tracing can produce the most realistic rendered images possible with soft shadows, caustics and global illumination, but is kind of a brute force approach that consequently needs lots of calculation time and performance.[5]

In practical application, like the unreal engine 4 game engine, there is a hybrid Ray Tracer that couples ray tracing capabilities with existing rasterization effect and a Path Tracer for generating reference renders offline. Thus the Ray Tracer can provide results for shadows, ambient occlusion (AO), reflections, interactive global illumination, and translucency in real-time by using a low number of samples couples with a denoising

[‡] It also requires light sources to have actual sizes, a bit of a departure from traditional point light sources that have a position but are treated like an infinitely small point in space (which works fine for ray tracing and rasterization because they only care about where the light is in space, but a path tracer needs to be able to intersect the light source).

algorithm, to mitigate the shortcomings of ray tracing like hard shadows. The Path Tracer, on the other hand, gathers samples over time and generates ground truth renders.[§] For this purposes it is useful to be able to set the maximum bounces that rays should travel and the number of samples per pixel that should be used for convergence.[7,8]

3. METHODOLOGY

The literature review is divided in the following **Research Questions:**

1. How useful is photo-realistic rendering for visual learning?
2. Which research areas are the main objectives of the training of AI with synthetic image data?
3. How is the synthetic training data generated?
4. What are the findings with regard to industrial applications, in particular quality testing?

Search Terms

Searching online scientific databases[¶], predominantly google scholar yielded nearly 100 results published since 2016. Most of them have quite different titles than each other like "Deep generative adversarial neural networks for realistic prostate lesion MRI synthesis", "FaceForensics++: Learning to Detect Manipulated Facial Images" and "Deep Underwater Image Enhancement". On the basis of these titles it can at least be concluded that the fields of application in which research is being conducted on combining synthetic data and deep learning are very numerous and diverse. As soon as the term quality inspection is added [‖] the results decrease immensely.

4. FINDINGS - ANSWERING THE RESEARCH QUESTIONS

4.1 How useful is photo-realistic rendering for visual learning

On of the former works that used the idea of training AI with synthetic data is from 2014 by Sun and Saenko. Therein domain adaption was performed on 3D models, rendering 2D images for the task of object detection. Sun and Saenko performed **domain adaptation**[**] "based on decorrelated features" and found that detectors trained on virtual data and adapted to real-image statistics perform comparably to detectors trained on real image datasets, including ImageNet. The results showed that non-photorealistic data works just as well as attempts to render more realistic images.[9]

Instead of domain adaptation **Tobin et. al**[10] attempted **domain randomization**, a fundamental different approach to surpass the great obstacle they named the ***reality gap***. They understood this to mean the impossibility of including all physical effects in current simulators, as well as the inability of simulated sensors to reproduce noise behavior in as much detail as their real-world counterparts.

The goal of their work was to localize objects on a table for the purpose of robotic control (to grab one object form a table) only from single monocular camera image from an uncalibrated camera. They used a randomly chosen number and shape of distractor objects on the table. Furthermore drawing from a random distribution was performed on position and texture of all objects on the table (also of the table, floor, skybox and robot), position orientation and field of view of the camera, number, position, orientation and specular characteristics of lights in the scene and type and amount fo random noise added to the images. Randomizing the camera

[§] For artists and programmers, the unbiased nature of the Path Tracer's ground truth image makes it invaluable to have built right into the Engine for comparison. It also removes the need for additional third-party software or plug-ins to generate these comparison results. For artists, it means being able to fine-tune materials and lighting setups more quickly. For programmers, it improves workflow and iteration times when tuning and validating the look of their real-time algorithms for techniques like denoising.

[¶] with the exact term "deep learning" AND "synthetic image generation"

[‖] more accurately "(quality control OR quality testing OR quality inspection)"

[**] Domain adaptation tries to replicate the statistical distributions of the source domain (the classifier is trained in) in the target domain (the objects it is tested and supposed to work with)

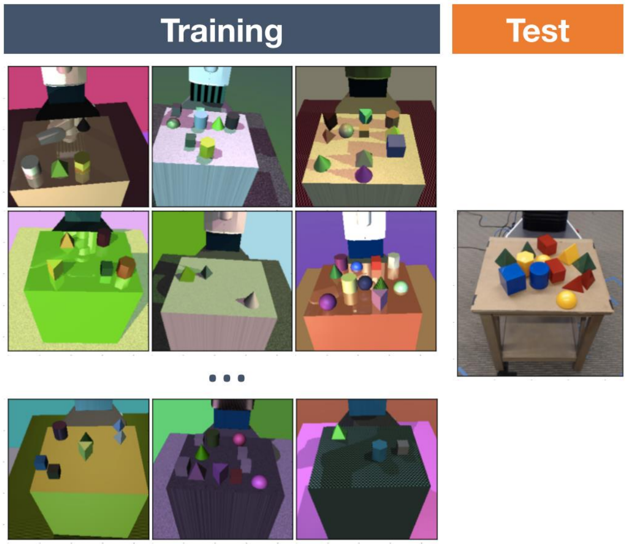

Figure 3: Various non-realistic training images and a real test image.

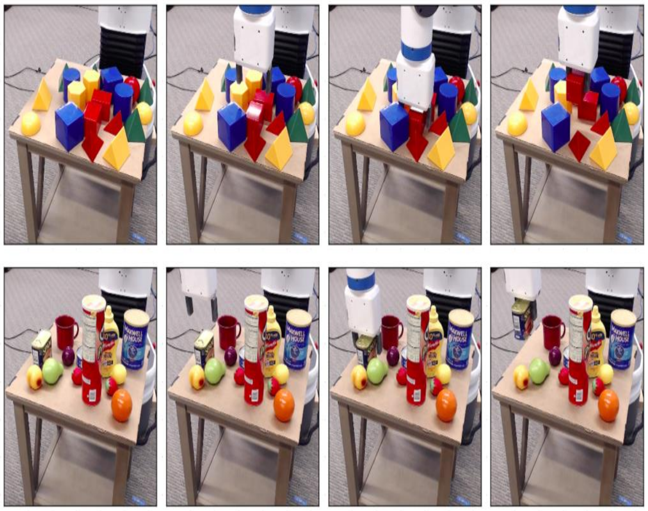

Figure 4: Robot grasping real objects on a table

characteristics included the placement of the camera ”randomly within a (10x5x10)cm box around its initial point” from the front of the table. From the assembled scene, not photorealistic images were rendered with MuJoCo Physics Engine and used for training a deep neural network. Cartesian coordinates of the center of the object in the world frame were used as labels. The trained model was evaluated in the real world with 480 webcam images of one or more geometric objects on a table at a distance of 70cm to 105cm from the camera. Further parameters were constant camera position, no control for lighting conditions. The 3D coordinates defined by the neural network were fed into a motion planning program which controlled the robot’s grasp.

Tobin et al. found that the method is at least some what sensitive to the following factors: number of training images, unique textures, presence of distractors in training data, randomization of camera position and pre-trained weights in the detection model. Only the amount of random noise had no significant impact on the model, as adding a small amount of random noise at training time improves convergence and makes the network less susceptible to local minima. Randomizing the position of the camera also consistently provides a slight accuracy boost. The author’s hypothesis that pre-training would be essential to generalizing to the real world proved to be false. It only took more training samples to archive similar small average errors. At around 5000 training samples the average error (in cm) of the network trained from scratch has significant decreased and is nearly as small as the pre-trained one. At around 16000 training samples, the network trained from scratch surpasses the pre-trained one.

The domain randomization technique was transferred to object detection and further investigated by **Tremblay et al.**[11] They placed a random number of synthetic cars in a 3D scene at random positions and orientations and added a random number of geometric shapes are added to the scene (flying distractors).A random number of lights of different types are inserted at random locations, and the scene is rendered from a random camera viewpoint, after which the result is composed over a random background image. The resulting images, with automatically generated ground truth labels (e.g., bounding boxes), are used for training the neural network. More accurate transfer learning was performed with diverse pre-trained common obj. detection networks: Faster R-CNN, R-FCN , SSD. The approach is evaluated on bounding box detection of cars on the KITTI dataset. Tremblay et al. claim that their domain randomization based car detector achieves better results on the KITTI dataset than the same architecture trained on virtual KITTI, even though the latter dataset is highly correlated with the test set. Furthermore, augmenting synthetic DR data by fine-tuning on real data yields better results than training on real KITTI data alone.

Hinterstoisser et al.[12] investigated the advantages of transfer learning combined with synthetic training data. Their papers goal is to exploit a large amount of available data in the source domain (synthetic) for training of a neural network to transfer the classification competence of the network to the target domain (real). The generation pipeline starts with placing the object at a random location in a randomly selected highly cluttered real background image using a uniform distribution. More randomness was introduced by swapping the three background image channels and randomly flipping and rotating the images in 90 degree steps. Attempts to use monochrome backgrounds of one randomly chosen color were abandoned after a few tests, as this lead to unsatisfying results. Random Gaussian noise was added to the rendered object and to better integrate the rendering with the background blurring with a Gaussian kernel was applied to the object inclusive its boundaries with the adjacent background image pixels. Hinterstoisser et al. make use of the fact that many object detectors like Faster-RCNN Mask-RCNN and R-FCN can be decoupled as a "meta-architecture" and a feature extractor such as VGG, Resnet, or InceptionResnet. The weights of this feature extractor, meaning the first layers (cut at some selected intermediate convolutional level) of Faster-RCNN architecture pre-learned on real images were "frozen". The remaining part of the architecture that can be used as part of the multi-way classification+localization of the object detector was trained on synthetic images only. The results of the Object localization and classification (semantic segmentation) task in cluttered environments come close (up to 95% of the performance) to detectors trained purely on real world data.

Hinterstoisser et al. state that images from different cameras lead to different results and that object detectors re-trained on synthetic data lead to poor performances, but contrary freezing the feature extractor always gives a huge performance boost. Moreover they claim that the results of their experiments suggest that simple rendering is sufficient to achieve good performances and that complicated scene composition does not seem necessary.[12]

Movshovitz-Attias[13] utilized large database of highly detailed, 3D models to create a large number of synthetic images. To diversify the generated data vary many of the rendering parameters were heavily varied. Among them the most important: randomization of light position, intensity, and temperature. Moreover F-stop and exposure time of the simulated Camera were sampled from a random distribution and occlusion was created by randomly sampling rectangles between 0.2 and 0.6 of the render size. A deep convolutional network whose architecture is based on AlexNet was trained with the rendered images of cars using a loss function that is optimized for viewpoint estimation. For evaluation viewpoint estimation was performed on CMU-Car and Pascal 3D+ datasets (containing real cars) labeled with the ground truth data.
Movshovitz-Attias et al. note that as the rendering process becomes more sophisticated the error decreases. They investigated the combination of synthetic and real data for training and found that when using low quality renders the error increases once the number of renders dominate the train set. But contrary this phenomenon is not observed with higher quality renders. Overall it is remarkable that there is an improvement when replacing up to 50% of the images with rendered data. One possible explanation the authors provide is the lack of balancing of the angle distribution the real training datasets have, as the cars are mainly photographed from two directions. The drop in performance when most data is rendered may probably be due to the circumstance that the overall image variability is smaller for renderings than for real images.
It is deduced that "generalizing from synthetic data is not harder than the domain adaptation required between two real-image datasets"[13] and that combining synthetic images with a certain amount of real data improves estimation accuracy. Moreover using complex materials and lighting is an important aspect of synthetic datasets.

Hodan at al.[14] did more in depth research on the use of highly photorealistic synthetic images generated via physically-based rendering for training a convolutional neural network (CNN)-based object detector. The datasets include 3D object models and real test RGB-D images of VGA resolution (only RGB channels). The images were annotated with ground-truth 6D object poses from which 2D bounding boxes where calculated and used for evaluation of the 2D object detection task. One the one hand physically based rendering with low medium and high rendering quality settings with commercial Autodesk MAYA and MAX software were used for rendering, on the other hand the synthetic data generation pipeline from Hinterstoisser et al. (4.1), for "baseline" image creation. With regard to AI Hodan et al. experimented with two underlying network architectures (Faster R-CNN: ResNet-101 and Inception-ResNet-v2). The networks were pre-trained on Microsoft COCO and fine-tuned on synthetic images for 100K iterations. To virtually increase diversity of the training set, the images were augmented by randomly adjusting brightness, contrast, hue, and saturation and by applying random Gaussian noise and blur.

Hodan at al. came to better results training on PBR images (following their three key aspects) than on the reference approach based on Hinterstoisser et al.s data generation pipeline, called Baseline rendering. The magnitude of the improvement varies over all combinations between 11% and around 35% (absolute improvement on five object classes and overall achieve a better performance on 12 out of 14 object classes). It stands out that the more complex the scene and lighting situation (complex reflections and not mostly materials with Lambertian surfaces), the better results the high quality PBR rendering yields over the low quality one. It is suggested that for scenes with simpler illumination and materials low quality rendering is sufficient, indicating to abandon PBR quality in favor of speed. However, even low-quality, physically based rendered images perform significantly better than their non-PBR counterparts.

While with some constraint Hinterstoisser et al where able to train state-of-the-art object detectors on synthetic data only, with results are close to approaches trained on real data only, the majority of researchers get improved results from extending the existing data with synthetically generated images.

4.2 Which research areas are the main objectives of the training of AI with synthetic image data

The purposes as well as the practical applications deep learning with synthetic data is researched, applied and used for are very diverse. In the following, some papers are briefly presented according to application areas.

Viewpoint estimation

Movshovitz et al. modified AlexNet DNN to investigate the usefulness of photo-realistic rendering for the means of camera viewpoint estimation. Performing various experiments on the effects varied data generation parameters like render quality, lighting and the mixture of synthetic and real training data have on estimation performance, they found that combining real images with synthetically generated ones improves performance and also concluded that using complex materials and lighting is an important aspect of synthetic datasets. It was reasoned that generalizing from synthetic data is not harder than the domain adaptation required between two real-image datasets and that combining synthetic images with a small amount of real data improves estimation accuracy and models trained on a combination of synthetic and real data outperform ones trained on natural images (see Chapter 4.1).[13]

Su et. al also did research on viewpoint estimation by overlaying images rendered from large 3D model collections on top of real images. Their findings lead to the conclusion that viewpoint estimation with the artificial images rendered "by carefully designing the data synthesis process" can significantly outperform state-of-the-art methods on PASCAL 3D+[*] benchmark. Furthermore it is claimed that the proposed "Render for CNN" pipeline can be extended to many tasks beyond viewpoint estimation.[15]

Pose Estimation

Pose estimation to obtain the coordinates of a desired object from only a single image of a 2D camera is a much desired research question. Especially in the industrial application the location of a real object obtained by a AI can be fed to a robot to perform grasping or other work processes without the need of a 3D imaging system. The already presented papers from Tobin et al. (see Chapter 4.1) and Mitash et al. (see Chapter 4.4) describe application examples in this context. Another form of pose estimation has been investigated by Chen et al,[16] who trained a cnn (AlexNet and VGG) on synthetic images of humans for the task of 3D pose estimation. The data generation is done with a sampled 3D pose model (SCAPE model) on which one of variaous clothes textures is mapped. After deforming the textured model, it is rendered using a variety of viewpoints and light sources, and finally composited over real image backgrounds. Their results show that the CNNs trained on this synthetic data with the aid of domain adaptation out-perform those trained with real photos on 3D pose estimation tasks.

Semantic Segmentation

Tsirikoglou et al.[17] claim "when analyzing the quality of a synthetic dataset, it is in general most telling to perform training on synthetic data alone, without any augmentation in the form of fine-tuning or weight initialization." They also conclude that a focus on maximizing variation and realism is well worth the effort. They estimate that their time investment in creating the dataset is at least three to four orders of magnitude smaller than the much larger virtual world from Richter et al., while still yielding state-of-the-art performance.

[*] http://cvgl.stanford.edu/projects/pascal3d.html

This was accomplished by ”ensuring that each image is highly varied as well as realistic, both in terms of low-level features such as anti-aliasing and motion blur, as well as higher-level features where realistic geometric models and light transport comes into play.”[17]

Concering segmentation of indoor scenes Zhang et al.[18] found that physically based rendering with realistic lighting and soft shadows is superior to other rendering methods. In addition they claim that pre-training on data obtained with physically based rendering with realistic lighting boosts the performance of indoor scene understanding tasks upon the state of the art methods. On the same subject McCormac et al.[?] show that a RGB-CNN pre-trained from scratch on synthetic RGB images can outperform an identical network initialized with the real-world VGG-16 ImageNet weights on a real-world indoor semantic labeling dataset, after fine-tuning. They concluded from this that large-scale high-quality synthetic RGB datasets with task-specific labels can be more useful for pre-training than real-world generic pre-training such as ImageNet.

Object Detection
Object detection or more precisely one of its multiple kinds is probably the most pursued topic of recent research in the area of training an artificial intelligence on synthetic data. Works in this direction includes Hinterstoisser 4.1, Hodan 4.1, Peng 4.3 and Mitash 4.4 to list just a few.

Rajpura et al.[19] use Blender and Cyles to generate synthetic image data for the purpose of object detection by transfer learning online 3D models from ShapeNet database (with everyday items like bottles, tins, cans and food items) and Archive3D database. They also encounter domain gap problem Jabbar et al. conducted experiments to detect drinking glasses in realistically rendered images compared to real images.[20]

Disparity and Optical Flow Estimation
Mayer et al.[21] came to the conclusion that for **disparity estimation** knowing and modeling the distortions of the camera in the training data largely improves the network's performance. For optical flow they stress the importance of domain adaption - a network trained on specialized data generalizes worse to other datasets than a network trained on diverse data. They claim that ”Realism is overrated”, because ”most of the learning task can be accomplished via simplistic data and data augmentation.” Even though realistic effects, such as sophisticated lighting models, induce minor improvements, they are not critical to learn basic optical flow. In the context of optical flow estimation it was found useful to train with learning schedules that combine multiple different datasets (simpler and more complex ones) greatly improve the generic performance of the trained networks.[21]

For **optical flow estimation** Dosovitskiy et al.[22] constructed neural network called FlowNet and tested it on several data sets. That largest one among these data sets was Flying Chairs. It was generated synthetically by overlaying random background images from Flickr with segmented images of chairs. Their results showed that it is possible to train a network to directly predict optical flow from two input images. For this purpose, the training data doe's not neet to be realistic. The artificial Flying Chairs data set ”including just affine motions of synthetic rigid objects, is sufficient to predict the optical flow in natural scenes with competitive accuracy.”[22]

Mayer et al. (2016)[23] extend this work with the intent to train large networks for disparity and scene flow estimation. [*] They generated a synthetic dataset containing over 35000 stereo image pairs with ground truth disparity, optical flow, and scene flow. The main part of this data collection consists of everyday objects flying along randomized 3D trajectories hence the name ”FlyingThings3D”. In the following examinations they found the network that was trained for disparity estimation on this data set is ”on par with the state of the art and runs 1000 times faster.”[23] This lead to the conclusion that the synthetic dataset can indeed be used to successfully train large convolutional networks.

Other application areas
Ekbatan et al.[24] used synthetic images containing a random set of pedestrians in a walkway to train a DNN. Their goal was to count the number of people in synthetic images and thereby, accurately predict the number of pedestrians. The data used for creation consists of low resolution images of groups of people walking towards and away from the camera extracted from 70 video samples. Their data generation process is quite similar to Dwibedi et al. and also light conditions were varied and artifacts that came by pasting people into the background image

[*] Mayer et al. provide the following definition: ”Estimating scene flow means providing the depth and 3D motion vectors of all visible points in a stereo video”

were corrected used morphological erosion. Tested on real images their approach performed similar to traditional approaches to the counting pedestrian task.

With regard to practical use of research in the last years many papers focused on the process of generating synthetic data with the intention that a **successful generation process** or the **synthetically generated data itself** can be adapted in diverse practical use cases like autonomous driving. For example Varol et al.[25] published the "SURREAL (Synthetic hUmans foR REAL tasks)" dataset. It contains 6 million synthetically-generated but realistic images together with ground truth pose, depth maps, and segmentation masks of people that where rendered from 3D sequences of human motion capture data.
Bhandari et. al[26] and the MIT Computer Graphics group presented a "configurable system to procedural generate synthetic street scene data", which they named "CoSy". Using 3D geometry of a whole city (.obj file), vehicles and other additional assets the system generates synthetic images including class level annotations of street scenes. It is intended to give researchers control over how data is generated, hence the system is designed to be configurable and extendable.

Regarding the **practical application** there are many different application areas in which synthetic data is used to train deep neural networks, including medicine,[27,28] agriculture,[29] and satellite or areal image analysis.[30] The majority of research papers[17,26,28,31,32] however deal with self driving car scenarios.

4.3 How the synthetic training data is generated

The majority of researches generate their training data by rendering images from 3D scenes via more or less photorealistic rendering.[10–12,14] Even simple/crude pasting of 2D objects in front of another background image and augmentation of the resulting boundaries is still used now and then like Dwibedi et al.[33] But besides common rendering software like Blender and Autodesk 3D Maya/Max with their embedded ray tracing engines like Octane, V-Ray, Cycles and Mitsuba renderer, more extraordinary methods are used to get to synthetic data. A few examples are mentioned here.

Richter et al. used **commercial video games** to generate realistic images that can be used to create "large-scale pixel-accurate ground truth data".[34] Expicitly named where Grand Theft Auto, Watch Dogs and Hitman (2016). Due to the commercial nature of Video games the source code is inaccessible. The key to still getting the data was to intercept the structured communication between the game and the graphics hardware.[*]

In their conclusion Richter et al. claim that their experiments have shown that data created with the presented approach can increase the performance of semantic segmentation models on real-world images. Furthermore that using their synthetically generated game data additionally for training yields a 2.6 percentage point improvement over training without this data.

Shafaei et al.[32] like Richter et al. use synthetic data in the form of commercial video games for semantic segmentation ("dense image classification") of street scenes. Every second, a sample from the in-game camera was collected containing. That sample consisted of RGB image, groundtruth semantic segmentation, depth image, and the surface normals. In comparison with real training data, their experiments showed that a convolution network trained on synthetic data achieves a similar test error as a network trained on real data for dense image classification. Furthermore if a simple domain adaptation technique is applied, they claim to get similar or even better results using synthetically generated RGB images than real data. As video games progress towards photorealistic environments, Shafaei et al. point out that they get the additional realism "at no extra cost".

Peng et al.[35] propose using **freely available 3D CAD models** to automatically generate synthetic 2D training images. Their comparatively simple generation process consists of non-photorealistic 3D CAD models of objects collected from the online source `3dwarehouse.sketchup.com`. Three to four poses (that best represent intraclass pro variance for real objects) of the object were sampled and augmented slightly by adding a small random rotation, texture, color, before adding a background rendering a virtual image. It was found that for novel object categories, adding synthetic variance to the existing dataset and fine-tuning the layers is useful. Furthermore Peng et al. marked that the outlined method outperforms detectors trained on real images when the real training data comes from a different domain. Realistic object texture, pose or background could not be found beneficial, as they lead to similar performance as training on synthetic images without these.

[*] A Software wrapper for the DirectX 9 API and used RenderDoc for wrapping Direct3D 11 was used followed by the implementation of a Injection method named "detouring".

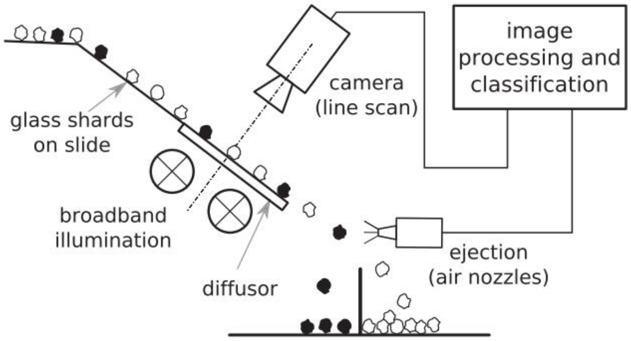

Figure 5: Scematic of on automated inspection system for the sorting of glass shards

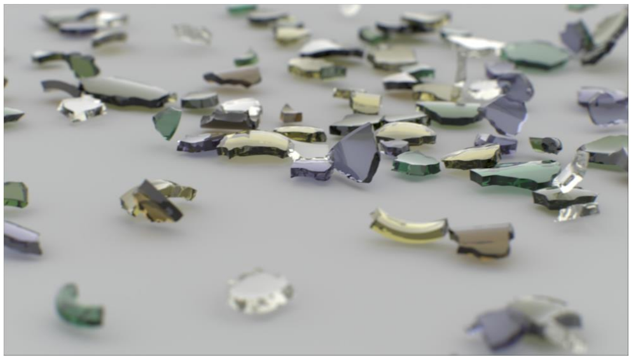

Figure 6: Rendered image of glass shards using monte carlo path tracing

4.4 Findings with regard to industrial applications - in particular quality testing

Despite this paper's intention to focus on the purpose of industrial quality inspection the literature research brought only sparse results.

In a broader sense, works such as that of Mitash et al.[36] can also be counted under ”industrial application”, since they combine the training of an AI with synthetic data with the concrete field of application, in this case the control of an industrial robot. Mitash et al. took CAD files, put them in a renderer with different lighting conditions and employed a physics engine to generate synthetic but physically realistic images. These were fed into a DNN to do object detection (in clutter) from multiple views and pose estimation. The process is evaluated with an industrial robotic arm and real objects placed on a table.[36]

Planche et al.[37] also used CAD data and modeled camera properties (distortion, motion blur, lens grain, noise) as well as motion, illumination, material properties, including micro-geometry. They also modeled the projector (lens, patterns and motion) and added this together to simulate rgb and depth data of industrial parts. They used their pipeline called DepthSynth to recognize 3D pose with six degrees of freedom (6-DOF) of one chair for testing. Also a classification task was done as a test, but only with 3 different chairs as classes and ”taking as final estimation the class of the nearest neighbor in the database for each extracted image representation.”[37] Despite the fact that Planche and his co-authors are from the industrial Company Siemens and have used their proposed pipeline for industrial parts, the test were only carried out on chairs, but not on industrial parts or assemblies.

Maybe closest to industrial quality inspection with Ai trained on synthetic image data are two papers from Retzlaff et al.[38,39] who did research on classification of glass shards. They rendered images of class glass shards once using monte carlo path tracing. This synthetic image generation method was contrasted with a real one that took images of different coloured glass shards which move on a slide past a RGB line scan camera that records the shards as they pass over a broadband illumination behind a diffusor (compare 5). The type of AI they trained with this data was an Support-Vector-Machine whose hyperparameters were chosen in a randomized search. They claimed that the classifier trained on synthetic images ”performed on par” with a classifier that was trained using physically acquired images, despite the synthetic images showing less variation than the physically acquired ones.[38]

In the second article of 2017, Retzlaff et al. attached great importance to the simulation of real lens systems and also took the simulation of (image) sensor properties into consideration. Procedural modeling techniques were used to generate virtual objects with varying appearance and properties, mimicking real objects and sample sets and different lenses more closely than before (compare figure 6). Both procedure and testing approach for Retzlaff et al.'s work match the practical use of automated optical (quality) inspection systems that are frequently seen in industry
To put it in a nutshell no specific industrial applications are sketched in papers. Closest to industry are (only) the kind of created data (CAD files for 3D geometry) and the purposes of application (robotic control, detection of glass shards, autonomous driving).

5. SUMMARY AND INTERPRETATION

First and foremost, it is undoubtedly possible to successfully solve problems in the ”real world”, i.e. on real data, with the help of AI trained on synthetic data. Over the majority of papers there is consent that one of the most important obstacles to overcome is the ”domain gap” it is undoubted from all the listed sources that domain randomization has a major influence and most sources even emphasize that it has the most impacts on the effectiveness and success of training with synthetic data. The reason for this can be seen in randomization in a non-realistic way, which forces the neural network to learn the essential features of the object of interest by varying important parameters of the simulator such as lighting, pose, object textures etc. in great detail.

Any type of data is used for the training of the AI, mostly 2D images, but also 3D point clouds, depth maps, data from time-of-flight sensors and 3D CAD data. Analogously at least this diverse data is used to create the synthetic training data and labels the AI needs for unsupervised learning. Concerning modification and augmentation of the training data, even the slight changes to the original image like image translations, reflections and manipulation of color intensity are common techniques even Krizhevsky et al.[1] used to prevent overfitting of the neuronal network. Even more important are the techniques of data modification and augmentation when working with synthetic training data only. If designed properly even simple data generation approaches like Dwibedi et al.[33] can work. They ”smoothend out” the remaining boundary artifacts after they crudely pasted objects from a real word image into another background image. At least in this specific case it was possible to train a network for object recognition without 3D data, although the more common and generalizable approach includes more or less realistic rendering of 3D scenes.

Comparing all the synthetic generated data to the real one, of course there are still the two most important benefits that synthetically generation is much faster and the labeling of the data does not need manual work. With regard to the use of real data only for training, it is advised to mix the training data with a small amount of real data to boost generalization and to get the ”best cost-to-benefit ratio”.[13] On the other hand, for the purpose of quality analysis of a synthetic data set, it is ”in general most telling to perform training on synthetic data alone, without any augmentation in the form of fine-tuning or weight initialization.”.[17] However, if not enough real data are available for the training of an AI, the use of exclusively synthetic data is not less promising. Many of the literature papers focused primarily on the process of data generation rather than on a more or less practical area of application (compare Bhandari,[26] Retzlaff,[39] Varol[25]) .

As the research could barely find specific papers to (practical) applications of training dnns for quality control purposes it may be beneficial to search for a specific technical area of application like printed circuit board (PCB) inspection or a specific product group like injection-moulded parts or aluminum bent parts. A plausible reason for the low number of articles published with focus on industrial quality inspection may be the urge to keep company secrets.

REFERENCES

[1] Krizhevsky, A., Sutskever, I., and Hinton, G. E., “Imagenet classification with deep convolutional neural networks,” *Advances in neural information processing systems* , 1097–1105 (2012).

[2] Pharr, M., Humphreys, G., and Jakob, W., [*Physically based rendering: From theory to implementation*], Morgan Kaufmann, Cambridge, MA, third edition ed. (2017).

[3] Kajiya, J. T., [*The Rendering Equation*], ACM, New York, NY (1986).

[4] Zwicker, M., Jarosz, W., Lehtinen, J., Moon, B., Ramamoorthi, R., Rousselle, F., Sen, P., Soler, C., and Yoon, S.-E., “Recent advances in adaptive sampling and reconstruction for monte carlo rendering,” *Computer Graphics Forum* , 667–681 (2015).

[5] “Path tracing vs ray tracing.” https://www.dusterwald.com/2016/07/path-tracing-vs-ray-tracing/ (2016). (Accessed: 03.05.2019).

[6] Schied, C., Salvi, M., Kaplanyan, A., Wyman, C., Patney, A., Chaitanya, C. R. A., Burgess, J., Liu, S., Dachsbacher, C., and Lefohn, A., “Spatiotemporal variance-guided filtering,” in [*Proceedings of High Performance Graphics on - HPG ’17*], McGuire, M. and Patney, A., eds., 1–12, ACM Press, New York, New York, USA (2017).

[7] "Rendering/raytracingaytracing." https://docs.unrealengine.com/en-us/Engine/Rendering/RayTracing (2019). (Accessed: 03.05.2019).

[8] Keller, A., Fascione, L., Fajardo, M., Georgiev, I., Christensen, P. H., Hanika, J., and ... & Nichols, G., [*The Path Tracing Revolution in the Movie Industry*], ACM, New York, NY (2015).

[9] Sun, B. and Saenko, K., "From virtual to reality: Fast adaptation of virtual object detectors to real domains," *BMVC* (2014).

[10] Tobin, J., Fong, R., Ray, A., Schneider, J., Zaremba, W., and Abbeel, P., "Domain randomization for transferring deep neural networks from simulation to the real world."

[11] Tremblay, J., Prakash, A., Acuna, D., Brophy, M., Varun, J., Cem, A., Thang, T., Eric, C., Shaad, B., and Stan, B., "Training deep networks with synthetic data: Bridging the reality gap by domain randomization," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (2018).

[12] Hinterstoisser, S., Lepetit, V., Wohlhart, P., and Konolige, K., "On pre-trained image features and synthetic images for deep learning."

[13] Movshovitz-Attias, Y., Kanade, T., and Sheikh, Y., "How useful is photo-realistic rendering for visual learning?."

[14] Hodan, T., Vineet, V., Gal, R., Shalev, E., Hanzelka, J., Connell, T., Urbina, P., Sinha, S. N., and Guenter, B., "Photorealistic image synthesis for object instance detection."

[15] Su, H., Qi, C. R., Li, Y., and Guibas, L. J., "Render for cnn: Viewpoint estimation in images using cnns trained with rendered 3d model views," (2015).

[16] Chen, W., Wang, H., Li, Y., Su, H., Wang, Z., Tu, C., Lischinski, D., Cohen-Or, D., and Chen, B., "Synthesizing training images for boosting human 3d pose estimation."

[17] Tsirikoglou, A., Kronander, J., Wrenninge, M., and Unger, J., "Procedural modeling and physically based rendering for synthetic data generation in automotive applications," *arXiv preprint arXiv:1710.06270* .

[18] Zhang, Y., Song, S., Yumer, E., Savva, M., Lee, J.-Y., Jin, H., and Funkhouser, T., "Physically-based rendering for indoor scene understanding using convolutional neural networks," *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2017).

[19] Rajpura, P., Aggarwal, A., Goyal, M., Gupta, S., Talukdar, J., Bojinov, H., and Hegde, R., "Transfer learning by finetuning pretrained cnns entirely with synthetic images," in [*Computer Vision, Pattern Recognition, Image Processing, and Graphics*], Rameshan, R., Arora, C., and Dutta Roy, S., eds., *Communications in Computer and Information Science* **841**, 517–528, Springer Singapore, Singapore (2018).

[20] Jabbar, A., Farrawell, L., Fountain, J., and Chalup, S. K., "Training deep neural networks for detecting drinking glasses using synthetic images," in [*Neural Information Processing*], Liu, D., Xie, S., Li, Y., Zhao, D., and El-Alfy, E.-S. M., eds., *Lecture Notes in Computer Science* **10635**, 354–363, Springer International Publishing, Cham (2017).

[21] Mayer, N., Ilg, E., Fischer, P., Hazirbas, C., Cremers, D., Dosovitskiy, A., and Brox, T., "What makes good synthetic training data for learning disparity and optical flow estimation?," *International Journal of Computer Vision* **126**(9), 942–960 (2018).

[22] Dosovitskiy, A., Fischer, P., Illg, E., Hausser, P., Hazirbas, C., Vladimir Golkov, van der Smagt, P., Cremers, D., and Brox, T., "Flownet: Learning optical flow with convolutional networks," **Proceedings of the IEEE international conference on computer vision** (2015).

[23] Mayer, N., Ilg, E., Häusser, P., Fischer, P., Cremers, D., Dosovitskiy, A., and Brox, T., "A large dataset to train convolutional networks for disparity, optical flow, and scene flow estimation," (2016).

[24] Ekbatani, H. K., Pujol, O., and Segui, S., "Synthetic data generation for deep learning in counting pedestrians," in [*Proceedings of the 6th International Conference on Pattern Recognition Applications and Methods*], 318–323, SCITEPRESS - Science and Technology Publications (24.02.2017 - 26.02.2017).

[25] Varol, G., Romero, J., Martin, X., Mahmood, N., Black, M. J., Laptev, I., and Schmid, C., "Learning from synthetic humans," **Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition**, 109–117 (2017).

[26] Bhandari, N., "Procedural synthetic data for self-driving cars using 3d graphics: Diss. massachusetts institute of technology, 2018.," (2018).

[27] Mahmood, F., Chen, R., Sudarsky, S., Yu, D., and Durr, N. J., “Deep learning with cinematic rendering: fine-tuning deep neural networks using photorealistic medical images,” *Physics in Medicine & Biology* **63**(18), 185012 (2018).

[28] Shin, H.-C., Tenenholtz, N. A., Rogers, J. K., Schwarz, C. G., Senjem, M. L., Gunter, J. L., Andriole, K. P., and Michalski, M., “Medical image synthesis for data augmentation and anonymization using generative adversarial networks,” in [*International Workshop on Simulation and Synthesis in Medical Imaging*], 1–11, Springer (2018).

[29] Rahnemoonfar, M. and Sheppard, C., “Deep count: fruit counting based on deep simulated learning,” *Sensors* **17**(4), 905 (2017).

[30] Han, S., Fafard, A., Kerekes, J., Gartley, M., Ientilucci, E., Savakis, A., Law, C., Parhan, J., Turek, M., Fieldhouse, K., and Rovito, T., “Efficient generation of image chips for training deep learning algorithms,” in [*Automatic Target Recognition XXVII*], Sadjadi, F. A. and Mahalanobis, A., eds., *SPIE Proceedings*, 1020203, SPIE (2017).

[31] Alhaija, H. A., Mustikovela, S. K., Mescheder, L., Geiger, A., and Rother, C., “Augmented reality meets computer vision: Efficient data generation for urban driving scenes,” *International Journal of Computer Vision* **126**(9), 961–972 (2018).

[32] Shafaei, A., Little, J. J., and Schmidt, M., “Play and learn: Using video games to train computer vision models.”

[33] Dwibedi, D., Misra, I., and Hebert, M., “Cut, paste and learn: Surprisingly easy synthesis for instance detection,” *Proceedings of the IEEE International Conference on Computer Vision* (2017).

[34] Richter, S. R., Vineet, V., Roth, S., and Koltun, V., “Playing for data: Ground truth from computer games.”

[35] Peng, X., Baochen, S., Karim, A., and Saenko, K., “Learning deep object detectors from 3d models,” *Proceedings of the IEEE International Conference on Computer Vision* , 1278–1286 (2015).

[36] Mitash, C., Bekris, K. E., and Boularias, A., “A self-supervised learning system for object detection using physics simulation and multi-view pose estimation.”

[37] Planche, B., Wu, Z., Ma, K., Sun, S., Kluckner, S., Lehmann, O., Chen, T., Hutter, A., Zakharov, S., Kosch, H., and Ernst, J., “Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5d recognition,” in [*2017 International Conference on 3D 2017*], 1–10.

[38] Retzlaff, M.-G., Richter, M., Längle, T., and Beyerer, J., “Combining synthetic image acquisition and machine learning: Accelerated design and deployment of sorting systems,” (2016).

[39] Retzlaff, M.-G., Hanika, J., Beyerer, J., and Dachsbacher, C., “Physically based computer graphics for realistic image formation to simulate optical measurement systems,” *Journal of Sensors and Sensor Systems* **6**(1), 171–184 (2017).