Physics-based Differentiable Depth Sensor Simulation

Benjamin Planche Rajat Vikram Singh[‡]
Siemens Technology
benjamin.planche@siemens.com, rajats@alumni.cmu.edu

Abstract

Gradient-based algorithms are crucial to modern computer-vision and graphics applications, enabling learning-based optimization and inverse problems. For example, photorealistic differentiable rendering pipelines for color images have been proven highly valuable to applications aiming to map 2D and 3D domains. However, to the best of our knowledge, no effort has been made so far towards extending these gradient-based methods to the generation of depth (2.5D) images, as simulating structured-light depth sensors implies solving complex light transport and stereo-matching problems. In this paper, we introduce a novel end-to-end differentiable simulation pipeline for the generation of realistic 2.5D scans, built on physics-based 3D rendering and custom block-matching algorithms. Each module can be differentiated w.r.t. sensor and scene parameters; e.g., *to automatically tune the simulation for new devices over some provided scans or to leverage the pipeline as a 3D-to-2.5D transformer within larger computer-vision applications. Applied to the training of deep-learning methods for various depth-based recognition tasks (classification, pose estimation, semantic segmentation), our simulation greatly improves the performance of the resulting models on real scans, thereby demonstrating the fidelity and value of its synthetic depth data compared to previous static simulations and learning-based domain adaptation schemes.*

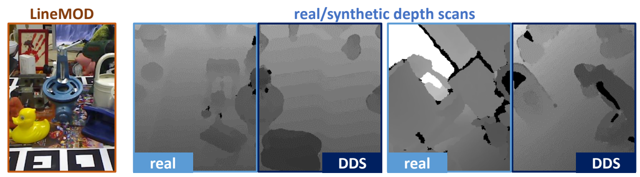

Figure 1: **Differentiable Depth Sensor Simulation (*DDS*) for the generation of highly-realistic depth scans.** *DDS* works off-the-shelf, but can be further optimized unsupervisedly against real data, yielding synthetic depth scans valuable to the training of recognition algorithms (demonstrated here on LineMOD dataset [21]).

1. Introduction

Progress in computer vision has been dominated by deep neural networks trained over large amount of data, usually labeled. The deployment of these solutions into real-world applications is, however, often hindered by the cost (time, manpower, access, *etc*.) of capturing and annotating exhaustive training datasets of target objects or scenes. To partially or completely bypass this hard data requirement, an increasing number of solutions are relying on synthetic images rendered from 3D databases for their training [15, 51, 36, 46, 61, 45], leveraging advances in computer graphics [50, 44]. Indeed, physics-based rendering methods are slowly but surely closing the visual gap between real and synthetic color image distributions, simulating complex optical phenomena (*e.g*., realistic light transport, lens aberrations, Bayer demosaicing, *etc*.). While these extensive tools still require domain knowledge to be properly parameterized for each new use-case (w.r.t. scene content, camera properties, *etc*.), their positive impact on the training of color-based visual recognition algorithms has been well documented already [9, 22].

The same cannot be said about depth-based applications. Unlike color camera that captures light intensity, structured-light depth sensors rely on stereo-vision mechanisms to measure the per-pixel distance between their focal plane and elements in the scene. They are useful for geometry-sensitive applications (*e.g*., robotics), but little effort has been made towards closing the realism gap w.r.t. synthetic depth (2.5D) scans or understanding their impact on the training of depth-based recognition methods. Some simulation pipelines [19, 33, 46] and domain adaptation schemes [55, 16, 54, 5, 63, 61] have been proposed; but the former methods require extensive domain knowledge [46, 63] to be set up whereas some of the latter need relevant real images for their training [55, 16, 54, 4], and all fail to generalize to new sensors [19, 33] or scenes [4, 63].

Borrowing from both simulation and learning-based principles, we propose herein a novel pipeline that virtually replicates depth sensors and can be optimized for new use-cases either manually (*e.g*., providing known intrinsic pa-

[‡]Now at NVIDIA.

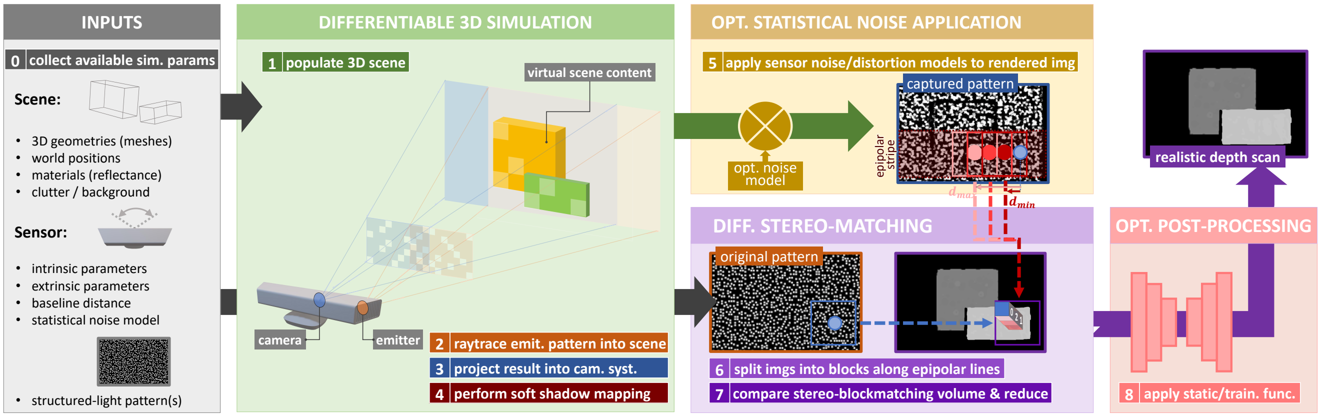

Figure 2: **Pipeline overview.** *DDS* differentiably simulates the physics and algorithmic mechanisms of real depth sensors.

rameters of a new sensor) or automatically via supervised or unsupervised gradient descent (*e.g.*, optimizing the pipeline over a target noise model or real scans). Adapting recent differentiable ray-tracing techniques [35, 64, 27] and implementing novel *soft* stereo-matching solutions, our simulation is differentiable end-to-end and can therefore be optimized via gradient descent, or integrated into more complex applications interleaving 3D graphics and neural networks. As demonstrated throughout the paper, our solution can off-the-shelf render synthetic scans as realistic as non-differentiable simulation tools [19, 33, 46], outperforming them after unsupervised optimization. Applied to the training of deep-learning solutions for various visual tasks, it also outperforms unconstrained domain adaptation and randomization methods [53, 5, 63, 61], *i.e.*, resulting in higher task accuracy over real data; with a much smaller set of parameters to optimize. In summary, our contributions are:

Differentiable Depth Sensor Simulation (*DDS*) – we introduce *DDS*, an end-to-end differentiable, physics-based, simulation pipeline for depth sensors. As detailed in Section 3, *DDS* reproduces the structured-light sensing and stereo-matching mechanisms of real sensors, off-the-shelf generating realistic 2.5D scans from virtual 3D scenes.

Optimizable Simulation through Gradient Descent – Because *DDS* is differentiable w.r.t. most of the sensor and scene parameters, it can learn to better simulate new devices or approximate unaccounted-for scene properties in supervised or unsupervised settings. It can also be tightly incorporated within larger deep-learning pipeline, *e.g.*, as a differentiable 3D-to-2.5D mapping function.

Benefits to Deep-Learning Recognition Methods – we demonstrate in Section 4 that *DDS* is especially beneficial to recognition solutions that must rely on synthetic data. The various methods (for depth-based object classification, pose estimation, or segmentation) trained with *DDS* performed significantly better when tested on real data, compared to the same methods trained with previous simulation tools or domain adaptation algorithms surveyed in Section 2.

2. Related work

Physics-based Simulation for Computer Vision. Researchers have already demonstrated the benefits of physics-based rendering of color images to deep-learning methods [22, 9], leveraging the extensive progress of computer graphics in the past decades. However, unlike color cameras, the simulation of depth sensors have not attracted as much attention. While it is straightforward to render synthetic 2.5D maps from 3D scenes (*c.f. z-buffer* graphics methods [52]), such *perfect* scans do not reflect the structural noise and measurement errors impairing real scans, leaving recognition methods trained on this synthetic modality ill-prepared to handle real data [46, 63, 45].

Early works [28, 14] tackling this *realism gap* tried to approximate the sensors' noise with statistical functions that could not model all defects. More recent pipelines [19, 33, 46, 49] are leveraging physics-based rendering to mimic the capture mechanisms of these sensors and render realistic depth scans, comprehensively modeling vital factors such as sensor noise, material reflectance, surface geometry, *etc*. These works also highlighted the value of proper 2.5D simulation for the training of more robust recognition methods [46, 45]. However, extensive domain knowledge (w.r.t. sensor and scene parameters) is required to properly configured these simulation tools. Unspecified information and unaccounted-for phenomena (*e.g.*, unknown or patented software run by the target sensors) can only be manually approximated, impacting the scalability to new use-cases.

With *DDS*, we mitigate this problem by enabling the pipeline to learn missing parameters or optimize provided ones by itself. This is made possible by the recent progress in differentiable rendering, with techniques modelling complex ray-tracing and light transport phenomena with continuous functions and adequate sampling [37, 35, 64, 27]. More specifically, we build upon Li *et al.* rendering framework [35] based on ray-tracing and Monte-Carlo sampling.

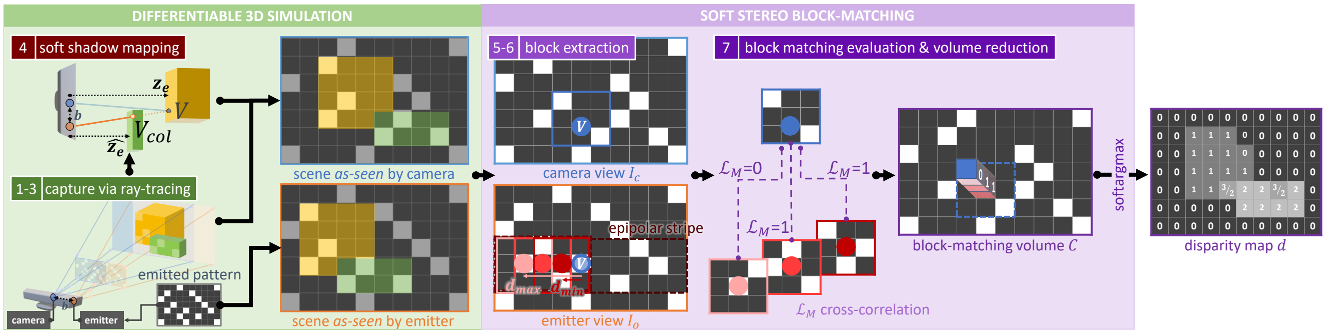

Figure 3: **Gradient-based light transport and block-matching**, proposed in this paper to approximate the original methods.

Domain Adaptation and Randomization. Similar to efforts w.r.t. color-image domains, scientists have also been proposing domain-adaptation solutions specific to depth data, replacing or complementing simulation tools to train recognition methods. Most solutions rely on unsupervised conditional generative adversarial networks (GANs) [18] to learn a mapping from synthetic to real image distributions [5, 60, 34] or to extract features supposedly domain-invariant [17, 63]. Based on deep neural architectures trained on an unlabeled subset of target real data, these methods perform well over the specific image distribution inferred from these samples, but do not generalize beyond (*i.e.*, they fail to map synthetic images to the real domain if the input images differ too much w.r.t. training data).

Some attempts to develop more scalable domain adaptation methods, *i.e.*, detached from a specific real image domain (and therefore to the need for real training data), led to *domain randomization* techniques [53]. These methods apply randomized transformations (handcrafted [53, 62, 63] or learned [61]) to augment the training data, *i.e.*, performing as an adversarial noise source that the recognition methods are trained against. The empirically substantiated claim behind is that, with enough variability added to the training set, real data may afterwards appear just as another noisy variation to the models. We can, however, conceptually understand the sub-optimal nature of these unconstrained domain adaptation techniques, which consider any image transform in the hope that they will be valuable to the task, regardless of their occurence probability in real data.

By constraining the transforms and their trainable parameters to the optical and algorithmic phenomena actually impacting real devices, *DDS* can converge much faster towards the generation of images that are both valuable to learning frameworks and photorealistic.

3. Methodology

As illustrated in Figure 3, structured-light devices measure the scene depth in their field of view by projecting a light pattern onto the scene with their emitter. Their camera—tuned to the emitted wavelength(s)—captures the pattern's reflection from the scene. Using the original pattern image I_o and the captured one I_c (usually filtered and undistorted) as a stereo signal, the devices infer the depth at every pixel by computing the discrepancy map between the images, *i.e.*, the pixel displacements along the epipolar lines from one image to the other. The perceived depth z can be directly computed from the pixel disparity d via the formula $z = \frac{f_\lambda b}{d}$, with b baseline distance between the two focal centers and f_λ focal length shared by the device's emitter and camera. Note that depth sensors use light patterns that facilitate the discrepancy estimation, usually performed by block-matching algorithms [12, 30]. Finally, most depth sensors perform some post-processing to computationally refine their measurements (*e.g.*, using hole-filling techniques to compensate for missing data).

In this paper, we consider the simulation of structured-light depth sensors as a function $Z = G(\Phi)$, with $\Phi = \{\Phi_s, \Phi_c, \Phi_e\}$ set of simulation parameters. G virtually reproduces the aforementioned sensing mechanisms, taking as inputs a virtual 3D scene defined by Φ_s (*e.g.*, scene geometry and materials), the camera's parameters Φ_c (*e.g.*, intrinsic and extrinsic values) and the emitter's Φ_e (*e.g.*, light pattern image or function γ_e, distance b to the camera); and returns a synthetic depth scan Z *as seen* by the sensor, with realistic image quality/noise. We propose a simulation function G differentiable w.r.t. Φ, so that given any loss function $\mathcal{L}$ computed over Z (*e.g.*, distance between Z and equivalent scan $\widehat{Z}$ from a real sensor), the simulation parameters Φ can be optimized accordingly through gradient descent. The following section describes the proposed differentiable pipeline step by step, as shown in Figures 2 and 3.

3.1. Pattern Capture via Differentiable Ray-Tracing

To simulate realistic pattern projection and capture in a virtual 3D scene, we leverage recent developments in physics-based differentiable rendering [37, 35, 64, 27]. Each pixel color γ_c observed by the device camera is formalized as an integration over all light paths from the scene passing through the camera's pixel filter (modelled as a continuous function k), following the rendering equation:

$$\gamma_c(\Phi) = \iiint k(x, y, \omega, \Phi_c) L(x, y, \omega\,;\,\Phi)\,dx\,dy\,d\omega, \quad (1)$$

with (x, y) continuous 2D coordinates in the viewport system, ω light path direction, and L the radiance function modelling the light rays coming from the virtual scene (*e.g.*, from ambient light and emissive/reflective surfaces) [35]. At any unit surface V projected onto (x, y) (in viewport coordinate system), the radiance L with direction ω is, therefore, itself integrated over the scene content:

$$L(x, y, \omega\,;\,\Phi) = \int_{\mathbb{S}^2} L_i(x, y, \omega\,;\,\Phi) f_s(V, \omega, \omega_i)\,d\sigma(\omega_i) + L_V(x, y, \omega\,;\,\Phi_s), \quad (2)$$

with L_V radiance emitted by the surface (*e.g.*, for the structured-light emitter or other light sources embodied in the scene), L_i incident radiance, f_s bidirectional reflectance distribution function [43], $d\sigma$ solid-angle measure, and $\mathbb{S}^2$ unit sphere [64]. As proposed by Li *et al.* [35], Monte Carlo sampling is used to estimate these integrals and their gradients: for continuous components of the integrand (*e.g.*, inner surface shading), usual area sampling with automatic differentiation is applied, whereas discontinuities (*e.g.*, surface edges) are handled via custom edge sampling.

More specific to our application, we simulate the structured-light pattern projection onto the scene and its primary contribution L_e to L for each unit surface V as:

$$L_e(x, y, \omega, \Phi) = \gamma_e(x_e, y_e, \Phi_e)\eta(V, \Phi_e), \quad (3)$$

with $(x_e, y_e, z_e)^\top = M_e V$ projection of V into the pattern image coordinate system defined by the projection matrix M_e, γ_e continuous representation of the structured-light pattern emitted by the sensor, and η light intensity (*e.g.*, as a function of the distance to the emitter). In other words, for surfaces visible to the camera, we trace rays from them to the light emitter to measure which elements of its pattern are lighting the surfaces (*c.f.* steps 1-3 in Figure 3).

As highlighted in various studies [33, 32, 46, 45], due to the baseline distance between their emitter and camera, depth sensors suffer from shadow-related capture failure, *i.e.*, when a surface V contributing to γ_c does not receive direct light from the emitter due to occlusion of the light rays by other scene elements (*c.f.* step 4 in Figure 3). Therefore, we propose a soft *shadow mapping* procedure [57, 1] that we model within the light intensity function η as follows:

$$\eta(V) = \frac{\eta_c}{z_e^2}\left(1 - \sigma(z_e - \widehat{z}_e - \xi)\right), \quad (4)$$

with σ sigmoid operator (replacing the discontinuous step function used in traditional shadow mapping), η_c emitter intensity, and $\widehat{z}_e$ computed as $(x_e, y_e, \widehat{z}_e)^\top = M_e V_{col}$ where V_{col} is the first surface hit by the virtual ray thrown from the emitter focal center toward V (*i.e.*, V_{col} superposed to V but closer in the emitter 2D coordinate system). We add an optimizable bias $\xi \in \mathbb{R}$ to prevent *shadow acne* (shadow artifacts due to distance approximations) [8].

Estimating $\gamma_c(\Phi)$ accounting for the scene and sensor properties Φ, we obtain the rasterized image I_c. To cover non-modelled physics phenomena (*e.g.*, lens defects) and according to previous works [19, 46], we also adopt an optional noise function f_n applied to I_c, *e.g.*, $f_n(I_c) = I_c + \Delta I$, with $\Delta I = \epsilon\sigma_n + \mu_n$, $\{\mu_n, \sigma_n\} \in \Phi_c$, and $\epsilon \sim \mathcal{N}(0, 1)$ (*c.f.* reparameterization trick [13, 39]).

3.1.1 Differentiable Stereo Block-Matching

Similar to real depth sensors, our pipeline then compares the computed I_c with a rasterized version I_o of the original pattern (both of size $H \times W$) to identify stereo-correspondences and infer the disparity map. Differentiable solutions to regress disparity maps from stereo signals have already been proposed, but these methods rely on CNN components to perform their task either more accurately [38, 6, 11] or more efficiently [29]. Therefore, they are bound to the image domain that they were trained over. Since our goal is to define a scene-agnostic simulation pipeline, we proposed instead an improved continuous implementation [29] of the classic stereo block-matching algorithm applied to disparity regression [30, 31], illustrated in Figure 3. The algorithm computes a matching cost volume $C \in \mathbb{R}^{H \times W \times N_d}$ by sliding a $w \times w$ window over the two images, comparing each block in I_c with the set of N_d blocks in I_o extracted along the same epipolar line. Considering standard depth sensors with the camera and emitter's focal planes parallel, the epipolar lines appear horizontal in their image coordinate systems (with $N_d = W$), simplifying the equation into:

$$C(x, y, \delta) = \sum_{i=x+u}^{x+w} \sum_{j=y+v}^{y+w} \mathcal{L}_M\left(I_{c;\,i,j}, I_{o;\,i,j-\delta}\right), \quad (5)$$

with $\delta \in [y - N_d - w, y]$ horizontal pixel displacement and $\mathcal{L}_M$ matching function (we opt for cross-correlation). Matrix unfolding operations are applied to facilitate volume inference. Formulating the task as a *soft* correspondence search, we reduce C into the disparity map d as follows: $d(x, y) = \mathrm{softargmax}_\delta\, C(x, y, \delta)$ with $\mathrm{softargmax}_i\, X = \sum_i \frac{i e^{\beta X_i}}{\sum_i e^{\beta X_i}}$ and $\beta \in \mathbb{R}$ optimizable parameter controlling the temperature of the underlying probability map. From this, we can infer the simulated depth scan $Z = \frac{f_\lambda b}{d}$.

However, as it is, the block-matching method would rely on an excessively large cost volume $H \times W \times W$ (*i.e.*, with $N_d = W$) making inference and gradient computation impractical. We optimize the solution by considering

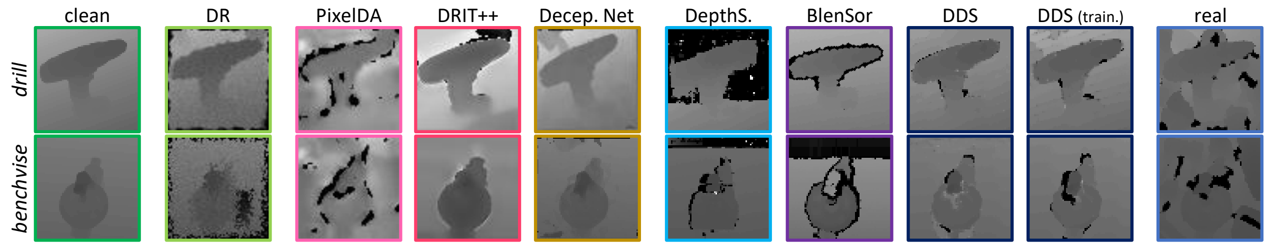

Figure 4: **Domain adaptation and simulation results**, on *Cropped LineMOD* [21, 5, 61] (real scene clutter not reproduced).

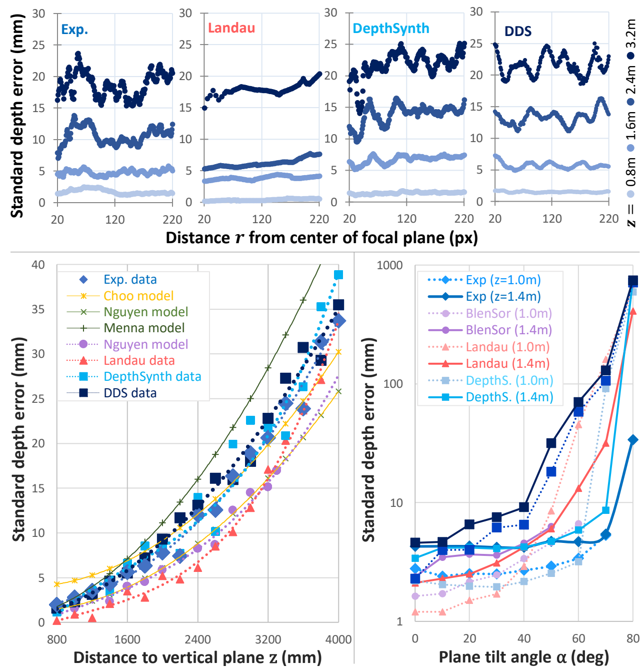

Figure 5: **Sensor noise study.** Given a flat surface placed at various distances z and tilt angles α w.r.t. the sensor, we plot the standard depth error as a function of r distance to the focal center in screen space, of z, and of α; for actual and simulated *Kinect V1* scans and statistical sensor models.

the measurement range $[z_{min}, z_{max}]$ of the actual sensor (*e.g.*, provided by the manufacturer or inferred from focal length), reducing the correspondence search space accordingly, *i.e.*, with $\delta \in [d_{min}, d_{max}] = [\lfloor \frac{f_\lambda b}{z_{max}} \rceil, \lfloor \frac{f_\lambda b}{z_{min}} \rceil]$ (dividing N_d tenfold for most sensors). The effective disparity range can be further reduced, *e.g.*, by considering the min/max *z-buffer* values in the target 3D scene.

The computational budget saved through this scheme can instead be spent refining the depth map. Modern stereo block-matching algorithms perform fine-tuning steps to achieve sub-pixel disparity accuracy, though usually based on global optimization operations that are not directly differentiable [24, 41]. To improve the accuracy of our method without trading off its differentiability, we propose the following method adapted from [33]: Let n_{sub} be an hyperparameter representing the desired pixel fraction accuracy. We create $\{I_{o,i}\}_{i=1}^{n_{sub}}$ lookup table of pattern images with a horizontal shift of i/n_{sub} px. Each $I_{o,i}$ is pre-rendered (once) via Equation 1 with $\Phi_{s,i}$ defining a virtual scene containing a single flat surface parallel to the sensor focal planes placed at distance $\frac{f_\lambda b}{d_{min,i}}$ with $d_{min,i} = d_{min} + \frac{i}{n_{sub}}$ (hence a global disparity of i/n_{sub} between I_o and $I_{o,i}$). At simulation time, block-matching is performed between I_c and each $I_{o,i}$, interlacing the resulting cost volumes and reducing them at once into the refined disparity map.

Finally, similar to the noise function optionally applied to I_c after capture, our pipeline allows Z to be post-processed, if non-modelled functions need to be accounted for (*e.g.*, device's hole-filling operation). In the following experiments, we present different simple post-processing examples (none, normal noise, or shallow CNN).

4. Experiments

Through various experiments, we evaluate the photorealism of depth images rendered by *DDS* and their value w.r.t. training recognition method or solving inverse problems.

4.1. Realism Study

First, we qualitatively and quantitatively compare *DDS* results with real sensor scans and data from other pipelines.

Qualitative Comparison. Visual results are shared in Figures 1 and 4 (w.r.t. *Microsoft Kinect V1* simulation), as well as in the supplementary material (w.r.t. *Matterport Pro2*). We can observe that off-the-shelf *DDS* reproduces the image quality of standard depth sensors (*e.g.*, *Kinect V1*): *DDS* scans contain shadow noise, quantization noise, stereo block-mismatching, *etc.*, similar to real images and previous simulations [19, 46] (*c.f.* empirical study of depth sensors' noise performed by Planche *et al.* [46]). Figure 4 and supplementary material further highlight how, unlike static simulations, ours can learn to tune up or down its inherent noise to better model sensors of various quality.

Quantitative Comparison. Reproducing the experimental protocol of previous 2.5D simulation methods [32, 46], we statistically model the depth error incurred by *DDS* as function of various scene parameters, and compare with empirical and statistical models from real sensor data.

• *Protocol.* Studying the *Microsoft Kinect V1* sensor, Landau *et al*. [33, 32] proposed the following protocol (further illustrated in the supplementary material). In real and simulated world, a flat surface is placed in front of the sensor. The surface is considered as a plane $(P, \overrightarrow{u}, \overrightarrow{v})$ with $P = (0, 0, z)$, $\overrightarrow{u} = (1, 0, 0)$, and $\overrightarrow{v} = (0, \sin\alpha, \cos\alpha)$ in camera coordinate system (*i.e*., a plane at distance z and tilt angle α w.r.t. focal plane). For each image captured in this setup, the standard depth error for each pixel q is computed as function of the distance z, the tilt angle α, and the radial distance r to the focal center. Like Landau *et al*. [33, 32] and Planche *et al*. [46], we compare the noise functions of our method with those of the actual *Kinect V1* sensor, as well as the noise functions computed for other state-of-the-art simulation tools (*BlenSor* [19], Landau's [33], and *DepthSynth* [46]) and noise models proposed by researchers studying this sensor (Menna *et al*. [40], Nguyen *et al*. [42] and Choo *et al*. [7, 32]).

• *Results.* In Figure 5, we first plot the error as a function of the radial distance r to the focal center. *DDS* performs realistically: like other physics-based simulations [19, 46], it reproduces the noise oscillations, with their amplitude increasing along with distance z—a phenomenon impairing real sensors, caused by pattern distortion.

We also plot the standard error as a function of the distance z and of the incidence angle α. While our simulated results are close to the real ones w.r.t. distance, we can observe that noise is slightly over-induced w.r.t. tilt angle. The larger the angle, the more stretched the pattern appears on the surface, impairing the block-matching procedure. Most algorithms fail matching overly-stretched patterns (*c.f*. exponential error in the figure), but our custom differentiable block-matching solution is unsurprisingly less robust to block skewing than the multi-pass methods used in other simulations [19, 46]. This could be tackled by adopting some more advanced block-matching strategies from the literature and rewriting them as continuous functions. This would however increase the computational footprint of the overall simulation and would only benefit applications where high photorealism is the end target. In the next experiments, we instead focus on deep-learning applications.

4.2. Applications to Deep Learning

We now illustrate how deep-learning solutions can benefit from our simulation method. We opt for various key recognition tasks over standard datasets, comparing the performance of well-known CNNs as a function of the data and the domain adaptation framework used to train them.

2.5D Semantic Segmentation. We start by comparing the impact of simulation tools on the training of a standard CNN for depth-based semantic segmentation.

• *Dataset.* For this task, we choose the *2D-3D-Semantic* dataset by Armeni *et al*. [3] as it contains RGB-D indoor scans shot with a *Matterport Pro2* sensor, as well as the camera pose annotations and the reconstructed 3D models of the 6 scenes. It is, therefore, possible to render synthetic images aligned with the real ones. We split the data into training/testing sets as suggested by *2D-3D-S* authors [3] (fold #1, *i.e*., 5 training scenes and 1 testing one). For the training set, we assume that only the 3D models, images and their pose labels are available (not the ground-truth semantic masks). Note also that for the task, we consider only the 8 semantic classes (out of 13) that are discernible in depth scans (*e.g*., *board* are indistinguishable from *wall* in 2.5D scans) and present in the training scenes.

Table 1: **Comparative study w.r.t. training usage**, measuring the accuracy of a CNN [20, 56, 59] performing semantic segmentation on real 2.5D scans from the indoor *2D-3D-S* dataset [3], as a function of the method used to render its training data (↑ = the higher the value, the better).

Train. Data Source	**Mean Intersection-Over-Union (mIoU)↑**								**Pixel Acc.↑**
	bookc.	*ceili.*	*chair*	*clutter*	*door*	*floor*	*table*	*wall*	
clean	.003	.018	.002	.087	.012	.052	.091	.351	35.3%
BlenSor [19]	.110	.534	.119	.167	.148	.561	.082	.412	51.6%
DepthS. [46]	.184	.691	.185	.221	.243	.722	.235	.561	65.3%
DDS	.218	.705	.201	.225	.240	.742	.259	.583	62.9%
DDS (train.)	**.243**	.711	**.264**	.255	.269	.794	.271	.602	69.8%
real	.135	**.770**	.214	**.277**	**.302**	**.803**	**.275**	**.661**	**73.5%**

• *Protocol.* Using the 3D models of the 5 training scenes, we render synthetic 2.5D images and their corresponding semantic masks using a variety of methods from the literature [2, 19, 46]. *DDS* is both applied off-the-shelf (only entering the *Pro2* sensor's intrinsic information), and after being optimized via supervised gradient descent (combining Huber and depth-gradient losses [23, 26]) against the real scans from one training scene (scene #3). Each synthetic dataset, and the dataset of real scans as upper-bound target, is then used to train an instance of a standard ResNet-based CNN [20, 56, 59] for semantic segmentation (we choose the *Dice* loss to make up for class imbalance [10]).

• *Results.* We measure the performance of each model instance in terms of per-class mean intersection-over-union [25, 48] and pixel accuracy. Results are shared in Table 1. We can observe how data from both untrained and trained *DDS* result in the most accurate recognition models (among those trained on purely synthetic data), with values on par or above those of the models trained on real annotated data for some classes. Even though *DDS* may not perfectly simulate the complex, multi-shot *Matterport* sensor, its ability to render larger and more diverse datasets can be easily leveraged to achieve high recognition accuracy.

Classification and Pose Estimation. We now perform an extensive comparison, as well as partial ablation study, w.r.t.

the ubiquitous computer vision task of instance classification and pose estimation (ICPE) [58, 5, 62, 63].

• *Dataset.* For this task, we select the commonly-used *Cropped LineMOD* dataset [21, 58, 5], composed of 64×64 RGB-D image patches of 11 objects under various poses, captured by a *Kinect V1* sensor, in cluttered environments. Disregarding the RGB modality for this experiment, we split the dataset into a non-annotated training set X^r_{trn} of 11,644 depth images, and a testing set X^r_{tst} of 2,919 depth images with their class and pose labels. The *LineMOD* dataset also provides a reconstructed 3D model of each object, used to render annotated synthetic training images. For fair comparison, all 3D rendering methods considered in this experiment are provided the same set of 47,268 viewpoints from which to render the images. These viewpoints are sampled from a virtual half-icosahedron centered on each target object, with 3 different in-plane rotations (*i.e.*, rotating the camera around its optical axis) [58, 62, 63, 47].

• *Protocol.* For this experiment, we opt for the generic task CNN from [16], trained for object classification and rotation estimation via the loss $\mathcal{L}_{icpe} = \mathbb{E}_{x,(y,q)}\left[-y^\top \log \hat{y} + \xi \log\left(1 - |q^\top \hat{q}|\right)\right]$, where the first term is the class-related cross-entropy and the second term is the log of a 3D rotation metric for quaternions [5, 61], with ξ pose loss factor, x input depth image, $\{y, q\}$ resp. ground-truth one-hot class vector and quaternion, and $\{\hat{y}, \hat{q}\}$ resp. predicted values. Again, we measure the network's classification accuracy and rotational error as a function of the data that it was trained on, extending the comparison to different online or offline augmentation and domain adaptation schemes (*c.f.* Figure 4 for visual comparison).

For domain adaptation solutions such as *PixelDA* [5] and *DeceptionNet* [61], the recognition network T is trained against a generative network G whose task is to augment the input synthetic images before passing them to T. This adversarial training framework, with G trained unsupervisedly against T [61] and/or a discriminator network D [5, 63] using non-annotated real images X^r_{trn}, better prepares T for its task on real data, *i.e.*, training it on noisier and/or more realistic synthetic images. To further demonstrate the training of our simulation, this time in a less constrained, unsupervised setting, we reuse *PixelDA* training framework, replacing its *ResNet*-based [20] generator by *DDS*. Our method is, therefore, unsupervisedly trained along with the task network, so that *DDS* learns to render synthetic images increasingly optimized to help T with its training. Three instance of *DDS* are thus compared: (a) off-the-shelf, (b) with $\Phi = \{\xi, \mu_n, \sigma_n, \beta\}$ (*i.e.*, parameters w.r.t. shadows, normal noise, and softargmax) optimized unsupervisedly, and (c) same as the previous but adding 2 trainable convolution layers as post-processing ($|\Phi|$ = 2,535 only in total).

• *Results.* Table 2 presents a detailed picture of state-of-the-art training solutions for scarce-data scenarios (basic or simulation-based image generation, static or GAN-based offline or online image transformations, *etc.*) and their performance on the task at hand. The various schemes are further sorted based on their requirements w.r.t. unlabeled real images and on the size of their parameter space.

Table 2: **Comparative and ablative study**, measuring the impact of unsupervised domain adaptation, sensor simulation (Sim), and domain randomization (DR, *i.e.*, using randomized 2.5D transforms *c.f.* [63, 61]) on the training of a CNN [16] for depth-based instance classification and pose estimation on the *Cropped LineMOD* dataset [21, 5, 61].

		Augmentations		**Sim/DA Req.**		**Class. Accur.**$^\uparrow$	**Rot. Error**$^\downarrow$
		offline	**online**	X^r_{trn}	$\lvert\Phi\rvert$		
	Basic					46.8%	67.0°
			DR			70.7%	53.1°
Dom. Adap.	*PixelDA* [5]		GAN	✓	1.96M	85.7%	40.5°
	DRIT++ [34]	GAN		✓	12.3M	68.0%	60.8°
		GAN	DR	✓	12.3M	87.7%	39.8°
	Decep.Net [61]		DR		1.54M	80.2%	54.1°
Sensor Sim.	*DepthS.* [46]	Sim				71.5%	52.1°
		Sim	DR			76.6%	45.4°
	BlenSor [19]	Sim				67.5%	63.4°
		Sim	DR			82.6%	41.4°
	DDS (untrained)	Sim				69.7%	67.6°
		Sim	DR			89.6%	39.7°
Combined	*DDS*	Sim		✓	4	81.2%	49.1°
		Sim	DR	✓	4	90.5%	39.4°
		Sim+conv		✓	2,535	85.5%	45.4°
		Sim+conv	DR	✓	2,535	93.0%	31.3°
	DDS + $(X,Y)^r_{trn}$	Sim+conv	DR	✓	2,535	**97.8%**	**25.1°**
	$(X,Y)^r_{trn}$			✓		95.4%	35.0°

The table confirms the benefits of rendering realistic data, with the recognition models trained against previous simulation methods [19, 46] performing almost as well as the instances trained with GAN-based domain adaptation techniques [5, 34] having access to a large set of relevant real images. In contrast to the latter methods, simulation tools have, therefore, superior generalization capability. A second interesting observation from the table is the value of online data augmentation (*e.g.*, random distortion, occlusion, *etc.*) [63], regardless of the quality of synthetic images. It provides a significant accuracy boost on both tasks, virtually and inexpensively increasing the training set size and variability *c.f.* domain randomization theory [53]. In that regard, *DeceptionNet* [61], a learning-based domain randomization framework, performs satisfyingly well without the need for real data (though domain knowledge is required to adequately set the 2.5D transforms' hyperparameters).

But overall, results highlight the benefits of combining all these techniques, which *DDS* can do seamlessly thanks to its gradient-based structure. Off-the-shelf, manually-parameterized *DDS* yields results similar to previous simulation tools when images are not further augmented but rises above all other methods when adding online augmen-

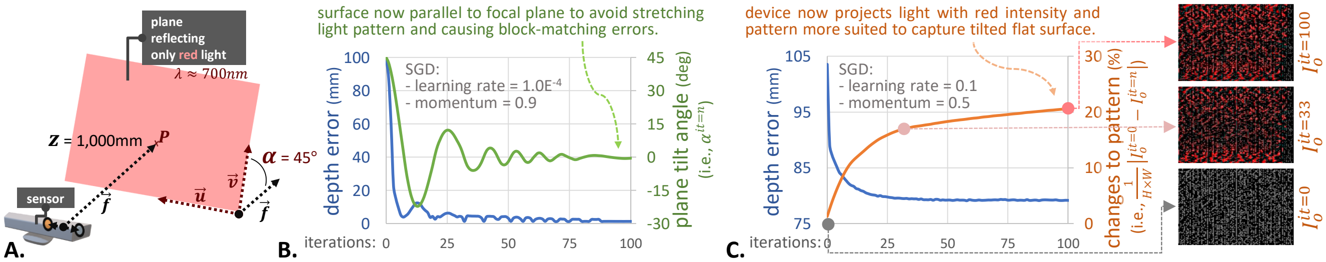

Figure 6: **Optimization of scene and sensor parameters** via simulation, to improve sensor precision in controlled use-cases. A. Experimental setup *c.f*. Section 4.1; B. Optimization of the scene parameters (*e.g*., pose) to reduce sensor standard error; C. Optimization of the sensor (*e.g*. pattern structure and frequencies) to improve its accuracy w.r.t. such scenes/materials.

tations. Training *DDS* unsupervisedly along with T further increases the performance, especially when intermittently applying a learned post-processing composed only of two convolutions. Opting for simple post-processing modules to compensate for non-modelled phenomena, we preserve the key role of simulation within *DDS* and, therefore, its generalization capability. Finally, we can note that, while the instance of T trained with *DDS* still performs slightly worse than the one trained on real annotated images w.r.t. the classification task, it outperforms it on the pose estimation task. This is likely due to the finer pose distribution in the rendered dataset (47,268 different images covering every angle of the objects) compared to the smaller real dataset. The best performance w.r.t. both tasks is achieved by combining the information in the real dataset with simulation-based data (*c.f*. penultimate line in Table 2).

Though computationally more intensive (a matter that can be offset by rendering images offline), our differentiable solution outperforms all other learning-based domain adaptation schemes, with a fraction of the parameters to train (therefore requiring fewer iterations to converge). Moreover, it is out-of-the-box as valuable as other depth simulation methods and outperforms them too when used within supervised or unsupervised training frameworks.

4.3. Optimization of Scene and Sensor Parameters

So far, we mostly focused on optimizing the simulation itself (*e.g*., shadow bias and noise parameters) in order to render more realistic images and improve CNNs training, rather than optimizing the scene or sensor parameters. To illustrate *DDS* capability w.r.t. such use-cases, we developed and performed a toy experiment, presented in Figure 6.

• *Protocol.* We consider the same scene setup as in Subsection 4.1 but assume that the target surface is tilted w.r.t. optical plan and only reflects *red* light frequencies, and that the depth sensor relies on a randomly generated dot pattern emitted with pseudo *white* light (mixture of wavelengths).

• *Results.* First, in Figure 6.B, we demonstrate how the scene geometry (*i.e*., the pose of the flat surface here) can be optimized via gradient descent to reduce the standard error of the simulated device (*i.e*., using the L1 distance between simulated depth maps and ground-truth noiseless ones as loss function). As expected, the surface is rotated back to be parallel to the focal plane, effectively preventing the stretching of the projected pattern and, therefore, block-matching issues (*c.f*. discussion in Subsection 4.1). In a second experiment, we consider the scene parameters fixed and instead try optimizing the depth sensor, focusing on its light pattern (*i.e*., to reduce sensing errors w.r.t. this kind of scenes, composed of tilted, red surfaces). Figure 6.C shows how the pattern image is optimized, quickly switching to red light frequencies, as well as more slowly adopting local patterns less impacted by projection-induced stretching.

We believe these toy examples illustrate the possible applications of simulation-based optimization of scene parameters (*e.g*., to reduce noise from surroundings when scanning an object) or sensor parameters (*e.g*., to build a sensor optimized to specific scene conditions).

5. Conclusion

In this paper we presented a novel simulation pipeline for structured-light depth sensors, based on custom differentiable rendering and block-matching operations. While directly performing as well as other simulation tools w.r.t. generating realistic training images for computer-vision applications, our method can also be further optimized and leveraged within a variety of supervised or unsupervised training frameworks, thanks to its end-to-end differentiability. Such gradient-based optimization can compensate for missing simulation parameters or non-modelled phenomena. Through various studies, we demonstrate the realistic quality of the synthetic depth images that *DDS* generates, and how depth-based recognition methods can greatly benefit from it to improve their end performance on real data, compared to other simulation tools or learning-based schemes used in scarce-data scenarios. Our results suggest that the proposed differentiable simulation and its standalone components further bridge the gap between real and synthetic depth data distributions, and will prove useful to larger computer-vision pipelines, as a *transformer* function mapping 3D data and realistic 2.5D scans.

References

[1] Tomas Akenine-Möller, Eric Haines, and Naty Hoffman. *Real-time rendering*. Crc Press, 2019. 4

[2] Aitor Aldoma, Zoltan-Csaba Marton, Federico Tombari, Walter Wohlkinger, Christian Potthast, Bernhard Zeisl, Radu Bogdan Rusu, Suat Gedikli, and Markus Vincze. Point cloud library. *IEEE Robotics & Automation Magazine*, 1070(9932/12), 2012. 6

[3] Iro Armeni, Sasha Sax, Amir R Zamir, and Silvio Savarese. Joint 2d-3d-semantic data for indoor scene understanding. *arXiv preprint arXiv:1702.01105*, 2017. 6

[4] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. *arXiv preprint arXiv:1612.05424*, 2016. 1

[5] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3722–3731, 2017. 1, 2, 3, 5, 7

[6] Jia-Ren Chang and Yong-Sheng Chen. Pyramid stereo matching network. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5410–5418, 2018. 4

[7] Benjamin Choo, Michael Landau, Michael DeVore, and Peter A Beling. Statistical analysis-based error models for the microsoft kinecttm depth sensor. *Sensors*, 14(9):17430–17450, 2014. 6

[8] Joey de Vries. LearnOpenGL - Shadow Mapping. https://learnopengl.com/Advanced-Lighting/Shadows/Shadow-Mapping. Accessed: 2021-03-10. 4

[9] Maximilian Denninger, Martin Sundermeyer, Dominik Winkelbauer, Youssef Zidan, Dmitry Olefir, Mohamad Elbadrawy, Ahsan Lodhi, and Harinandan Katam. Blenderproc. *arXiv preprint arXiv:1911.01911*, 2019. 1, 2

[10] Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal. The importance of skip connections in biomedical image segmentation. In *Deep learning and data labeling for medical applications*, pages 179–187. Springer, 2016. 6

[11] Shivam Duggal, Shenlong Wang, Wei-Chiu Ma, Rui Hu, and Raquel Urtasun. Deeppruner: Learning efficient stereo matching via differentiable patchmatch. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4384–4393, 2019. 4

[12] Nils Einecke and Julian Eggert. A multi-block-matching approach for stereo. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 585–592. IEEE, 2015. 3

[13] Otto Fabius, Joost R van Amersfoort, and Diederik P Kingma. Variational recurrent auto-encoders. In *ICLR (Workshop)*, 2015. 4

[14] Maurice F Fallon, Hordur Johannsson, and John J Leonard. Point cloud simulation & applications, 2012. http://www.pointclouds.org/assets/icra2012/localization.pdf. Accessed: 2020-09-23. 2

[15] Sanja Fidler, Sven Dickinson, and Raquel Urtasun. 3d object detection and viewpoint estimation with a deformable 3d cuboid model. In *Adv. Neural Inform. Process. Syst.*, pages 611–619, 2012. 1

[16] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015. 1, 7

[17] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The Journal of Machine Learning Research*, 17(1):2096–2030, 2016. 3

[18] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, pages 2672–2680, 2014. 3

[19] Michael Gschwandtner, Roland Kwitt, Andreas Uhl, and Wolfgang Pree. Blensor: blender sensor simulation toolbox. In *Advances in Visual Computing*, pages 199–208. Springer, 2011. 1, 2, 4, 5, 6, 7

[20] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, pages 770–778, 2016. 6, 7

[21] Stefan Hinterstoisser, Vincent Lepetit, Slobodan Ilic, Stefan Holzer, Gary Bradski, Kurt Konolige, and Nassir Navab. Model based training, detection and pose estimation of texture-less 3d objects in heavily cluttered scenes. In *ACCV*. Springer, 2012. 1, 5, 7

[22] Tomáš Hodaň, Vibhav Vineet, Ran Gal, Emanuel Shalev, Jon Hanzelka, Treb Connell, Pedro Urbina, Sudipta N Sinha, and Brian Guenter. Photorealistic image synthesis for object instance detection. In *2019 IEEE International Conference on Image Processing (ICIP)*, pages 66–70. IEEE, 2019. 1, 2

[23] Peter J Huber. Robust estimation of a location parameter. In *Breakthroughs in statistics*, pages 492–518. Springer, 1992. 6

[24] Martin Humenberger, Christian Zinner, Michael Weber, Wilfried Kubinger, and Markus Vincze. A fast stereo matching algorithm suitable for embedded real-time systems. *Computer Vision and Image Understanding*, 114(11):1180–1202, 2010. 5

[25] Paul Jaccard. The distribution of the flora in the alpine zone. 1. *New phytologist*, 11(2):37–50, 1912. 6

[26] Jianbo Jiao, Ying Cao, Yibing Song, and Rynson Lau. Look deeper into depth: Monocular depth estimation with semantic booster and attention-driven loss. In *Proceedings of the European conference on computer vision (ECCV)*, pages 53–69, 2018. 6

[27] Hiroharu Kato, Deniz Beker, Mihai Morariu, Takahiro Ando, Toru Matsuoka, Wadim Kehl, and Adrien Gaidon. Differentiable rendering: A survey. *arXiv preprint arXiv:2006.12057*, 2020. 2, 3

[28] Maik Keller and Andreas Kolb. Real-time simulation of time-of-flight sensors. *Simulation Modelling Practice and Theory*, 17(5):967–978, 2009. 2

[29] Alex Kendall, Hayk Martirosyan, Saumitro Dasgupta, Peter Henry, Ryan Kennedy, Abraham Bachrach, and Adam Bry.

End-to-end learning of geometry and context for deep stereo regression. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 66–75, 2017. 4

[30] Kurt Konolige. Small vision systems: Hardware and implementation. In *Robotics Research*, pages 203–212. Springer, 1998. 3, 4

[31] Kurt Konolige. Projected texture stereo. In *2010 IEEE International Conference on Robotics and Automation*, pages 148–155. IEEE, 2010. 4

[32] Michael J Landau. *Optimal 6D Object Pose Estimation with Commodity Depth Sensors*. PhD thesis, University of Virginia, 2016. http://search.lib.virginia.edu/catalog/hq37vn57m. Accessed: 2020-10-20. 4, 5, 6

[33] Michael J Landau, Benjamin Y Choo, and Peter A Beling. Simulating kinect infrared and depth images. *IEEE transactions on cybernetics*, 46(12):3018–3031, 2015. 1, 2, 4, 5, 6

[34] Hsin-Ying Lee, Hung-Yu Tseng, Qi Mao, Jia-Bin Huang, Yu-Ding Lu, Maneesh Singh, and Ming-Hsuan Yang. Drit++: Diverse image-to-image translation via disentangled representations. *International Journal of Computer Vision*, 128(10):2402–2417, 2020. 3, 7

[35] Tzu-Mao Li, Miika Aittala, Frédo Durand, and Jaakko Lehtinen. Differentiable monte carlo ray tracing through edge sampling. *ACM Transactions on Graphics (TOG)*, 37(6):1–11, 2018. 2, 3, 4

[36] Jasmine J Lim, Hamed Pirsiavash, and Antonio Torralba. Parsing ikea objects: Fine pose estimation. In *Int. Conf. Comput. Vis.*, pages 2992–2999. IEEE, 2013. 1

[37] Matthew M Loper and Michael J Black. Opendr: An approximate differentiable renderer. In *European Conference on Computer Vision*, pages 154–169. Springer, 2014. 2, 3

[38] Wenjie Luo, Alexander G Schwing, and Raquel Urtasun. Efficient deep learning for stereo matching. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5695–5703, 2016. 4

[39] Alireza Makhzani, Jonathon Shlens, Navdeep Jaitly, Ian Goodfellow, and Brendan Frey. Adversarial autoencoders. *arXiv preprint arXiv:1511.05644*, 2015. 4

[40] Fabio Menna, Fabio Remondino, Roberto Battisti, and Erica Nocerino. Geometric investigation of a gaming active device. In *SPIE Optical Metrology*, pages 80850G–80850G. International Society for Optics and Photonics, 2011. 6

[41] Matthias Michael, Jan Salmen, Johannes Stallkamp, and Marc Schlipsing. Real-time stereo vision: Optimizing semi-global matching. In *2013 IEEE Intelligent Vehicles Symposium (IV)*, pages 1197–1202. IEEE, 2013. 5

[42] Chuong V Nguyen, Shahram Izadi, and David Lovell. Modeling kinect sensor noise for improved 3d reconstruction and tracking. In *3D Imaging, Modeling, Processing, Visualization and Transmission (3DIMPVT), 2012 Second International Conference on*, pages 524–530. IEEE, 2012. 6

[43] Fred E Nicodemus. Directional reflectance and emissivity of an opaque surface. *Applied optics*, 4(7):767–775, 1965. 4

[44] Matt Pharr, Wenzel Jakob, and Greg Humphreys. *Physically based rendering: From theory to implementation*. Morgan Kaufmann, 2016. 1

[45] Benjamin Planche. *Bridging the Realism Gap for CAD-Based Visual Recognition*. PhD thesis, University of Passau, 2020. 1, 2, 4

[46] Benjamin Planche, Ziyan Wu, Kai Ma, Shanhui Sun, Stefan Kluckner, Terrence Chen, Andreas Hutter, Sergey Zakharov, Harald Kosch, and Jan Ernst. Depthsynth: Real-time realistic synthetic data generation from cad models for 2.5 d recognition. In *3DV*. IEEE, 2017. 1, 2, 4, 5, 6, 7

[47] Benjamin Planche, Sergey Zakharov, Ziyan Wu, Andreas Hutter, Harald Kosch, and Slobodan Ilic. Seeing beyond appearance-mapping real images into geometrical domains for unsupervised cad-based recognition. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2579–2586. IEEE, 2019. 7

[48] Md Atiqur Rahman and Yang Wang. Optimizing intersection-over-union in deep neural networks for image segmentation. In *International symposium on visual computing*, pages 234–244. Springer, 2016. 6

[49] Stefan Reitmann, Lorenzo Neumann, and Bernhard Jung. Blainder—a blender ai add-on for generation of semantically labeled depth-sensing data. *Sensors*, 21(6):2144, 2021. 2

[50] Christophe Schlick. An inexpensive brdf model for physically-based rendering. In *Computer graphics forum*, volume 13, pages 233–246. Wiley Online Library, 1994. 1

[51] Richard Socher, Brody Huval, Bharath Bath, Christopher D Manning, and Andrew Y Ng. Convolutional-recursive deep learning for 3d object classification. In *Adv. Neural Inform. Process. Syst.*, pages 665–673, 2012. 1

[52] Wolfgang Straßer. *Schnelle kurven-und flächendarstellung auf grafischen sichtgeräten*. PhD thesis, 1974. 2

[53] Josh Tobin, Rachel Fong, Alex Ray, Jonas Schneider, Wojciech Zaremba, and Pieter Abbeel. Domain randomization for transferring deep neural networks from simulation to the real world. In *IROS*, pages 23–30. IEEE, 2017. 2, 3, 7

[54] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. *arXiv preprint arXiv:1702.05464*, 2017. 1

[55] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014. 1

[56] Panqu Wang, Pengfei Chen, Ye Yuan, Ding Liu, Zehua Huang, Xiaodi Hou, and Garrison Cottrell. Understanding convolution for semantic segmentation. In *2018 IEEE winter conference on applications of computer vision (WACV)*, pages 1451–1460. IEEE, 2018. 6

[57] Lance Williams. Casting curved shadows on curved surfaces. In *Proceedings of the 5th annual conference on Computer graphics and interactive techniques*, pages 270–274, 1978. 4

[58] Paul Wohlhart and Vincent Lepetit. Learning descriptors for object recognition and 3d pose estimation. In *CVPR*, pages 3109–3118, 2015. 7

[59] Zifeng Wu, Chunhua Shen, and Anton Van Den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *Pattern Recognition*, 90:119–133, 2019. 6

[60] Simiao Yu, Hao Dong, Felix Liang, Yuanhan Mo, Chao Wu, and Yike Guo. Simgan: Photo-realistic semantic image manipulation using generative adversarial networks. In *2019*

IEEE International Conference on Image Processing (ICIP), pages 734–738. IEEE, 2019. 3

[61] Sergey Zakharov, Wadim Kehl, and Slobodan Ilic. Deceptionnet: Network-driven domain randomization. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 532–541, 2019. 1, 2, 3, 5, 7

[62] Sergey Zakharov, Wadim Kehl, Benjamin Planche, Andreas Hutter, and Slobodan Ilic. 3d object instance recognition & pose estimation using triplet loss with dynamic margin. In *IROS*, 2017. 3, 7

[63] Sergey Zakharov, Benjamin Planche, Ziyan Wu, Andreas Hutter, Harald Kosch, and Slobodan Ilic. Keep it unreal: Bridging the realism gap for 2.5 d recognition with geometry priors only. pages 1–11, 2018. 1, 2, 3, 7

[64] Shuang Zhao, Wenzel Jakob, and Tzu-Mao Li. Physics-based differentiable rendering: from theory to implementation. In *ACM SIGGRAPH 2020 Courses*, pages 1–30. 2020. 2, 3, 4