PX-NET: Simple, Efficient Pixel-Wise Training of Photometric Stereo Networks

Preprint · August 2020

CITATIONS
0

READS
84

4 authors, including:

Fotios Logothetis
Toshiba Research Europe Limited
17 PUBLICATIONS **99** CITATIONS
SEE PROFILE

Roberto Cipolla
University of Cambridge
492 PUBLICATIONS **36,915** CITATIONS
SEE PROFILE

PX-NET: Simple, Efficient Pixel-Wise Training of Photometric Stereo Networks

Fotios Logothetis[1], Ignas Budvytis[2], Roberto Mecca[1,2], and Roberto Cipolla[2]

[1] Cambridge Research Laboratory, Toshiba Europe, Cambridge, UK
fotios.logothetis@crl.toshiba.co.uk
[2] University of Cambridge, Cambridge, UK
{ib255,rm822,rc10001}@cam.ac.uk

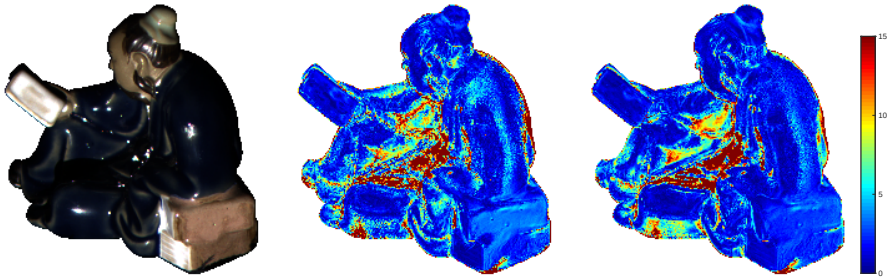
Abstract. Retrieving accurate 3D reconstructions of objects from the way they reflect light is a very challenging task in computer vision. Despite more than four decades since the definition of the Photometric Stereo problem, most of the literature has had limited success when global illumination effects such as cast shadows, self-reflections and ambient light come into play, especially for specular surfaces.
Recent approaches have leveraged the power of deep learning in conjunction with computer graphics in order to cope with the need of a vast number of training data in order to invert the image irradiance equation and retrieve the geometry of the object. However, rendering global illumination effects is a slow process which can limit the amount of training data that can be generated.
In this work we propose a novel pixel-wise training procedure for normal prediction by replacing the training data of globally rendered images with independent per-pixel renderings. We show that robustness to global physical effects can be achieved via data-augmentation which greatly simplifies and speeds up the data creation procedure. Our network, PX-NET, achieves the state-of-the-art performance on synthetic datasets, as well as the Diligent real dataset.

Keywords: Photometric Stereo, CNN, rendering

1 Introduction

Photometric Stereo (PS) is a classical problem in computer vision since the early '80s [1]. PS assumes multiple images from the same viewpoint along with varied illumination and calculates local geometrical features (e.g. normal or depth) at each pixel by exploiting the relation between surface orientation and intensity of reflected light. This is essentially an inverse rendering problem and the multitude of input images aims at disambiguating the contribution of the intrinsic surface color (albedo ρ) from the effect of the shading.

One of the most difficult feature to disambiguate while retrieving 3D shape from light reflected off the object is the type of reflection. The reason why this is so crucial is the strict dependency between light reflection and material surface



'Reading' CNN-PS[14], MAE=13^o Proposed, MAE=11.4^o

Fig. 1. Comparison of the proposed approach versus the state of art [14] on 'Reading' of the DiLiGenT real benchmark [15]. The evaluation metric is the mean angular error (MAE) of the computed normal compared with the ground truth.

of the object. Over the last forty years a very wide spectrum of image irradiance equations have been proposed. Starting from the basic linear light response for diffuse reflection [2,3], more specular behaviour of reflected light have been proposed [4,5,6,7,8,9]. Comparison among numerous irradiance equations can be found in [10,11,12,13].

Lately, [16,17] proposed a formulation capable to handle not only a wide variety of real material reflections, but they also included a non-pointwise behaviour of light reflected off the surface (*e.g.*scattering), namely Bidirectional Scattering Distribution Function (BSDF).

Recent advancements in computer graphics have allowed convolutional neural network (CNN)-based approaches to work by rendering large number of images of various surfaces under numerous light and material configurations. They often parametrise the PS problem as normal regression from light intensity observations (i.e. observational map [14]), effectively performing an inversion of the irradiance equation CNN-based approaches have been shown to outperform classical optimisation based methods [18,19] mainly due to the ability of CNNs to learn how to deal with a great variety of realistic reflectances which lead classical optimisation methods into intractable computations and thus simplifications (for example, the majority of classical literature still assumes Lambertian reflection). In addition, CNNs can gain robustness to deviations from the irradiance equation such as global illumination effects (cast shadows, self reflections) if the training data includes them ([14]). This can be achieved using rendering engines (like Blender [20]) which provide training data containing that level of realism. However, exhaustively sampling global illumination effects (which are a function of the overall surface geometry) requires a huge number of meshes to be rendered. In addition, the rendering requirements grow exponentially in order to cover materials/lights configurations as well. Finally, it is noted that rendering full objects is relatively slow and somehow inefficient, as there is a large amount of correlation among neighbouring pixels (especially for shadow/self reflection patterns).

In order to maximise the combinations of sampled materials, lights and normal directions, instead of pre-generating training data, we train our CNN with per-pixel rendered (using [16]) observational maps. Furthermore, given the pixel-wise type of rendering, we compensate the lack of global physical phenomena by augmenting the training data with a set of realistic effects including light source brightness variation, ambient light, cast shadows, self-reflections and reflectance mixing in discontinuity boundaries. Because replicating the distribution of the real world objects is challenging, we instead rely on broad enough augmentations to reduce the synthetic to real gap in the observation map space. We note that this is probably an easier task than attempting to render fully realistic full images as the observation map parameterisantion compacts information into a more constrained space compared to the set of real images.

Contribution: Our CNN based approach for solving PS problem has the following main contribution: we propose a per-pixel data generation and augmentation strategy which can replace slow-to-obtain full image rendering while still allowing the network to learn global rendering effects. In addition we also propose an improvement to the CNN-PS [14] architecture termed PX-NET which benefits from the increase in the training data variation. Finally, we extend the observation map parameterisation to include an additional channel with raw observations.

The rest of this work is divided as follows. Section 2 discusses relevant work in Photometric Stereo. Section 3 provides details of our proposed CNN approach. Sections 4 and 5 describe the experiment setup and corresponding results.

2 Related Work

An extensive research has been carried out attempting to retrieve 3D geometry using multiple lights reflected off an object. We now provide an overview of the latest PS improvements that mostly focus on deep learning approaches. For a fairly recent survey of PS techniques, refer to [21].

As Deep Learning (DL) has recently become very popular in the computer vision community, dominating the vast majority of ongoing research, several approaches have been proposed for retrieving 3D geometry of objects using PS. Most of the challenges in PS come from the difficulty of inverting non-linear and complex irradiance equations. Therefore, DL seems like an obvious way to go as it is known to be able to approximate highly non-linear mappings. However, regressing very dense and accurate depth maps is not a trivial task. The key point to make DL suitable for our purposes is to adapt the classification problem where DL excels, to the 3D reconstruction one.

A preliminary study proposed by Tang *et al.*[22] connected the usability of Deep Belief Networks [23] with light reflected off an object. However, this approach was limited by assuming diffuse reflection only. Recently, some approaches have been presented addressing the PS problem with DL architectures. Yu and Smith [24] built layers capable of modeling photometric image formation (in a unsupervised manner) that can be embedded into existent architectures such as encoder-decoder ones for establishing correspondences among light reflections

and geometry. Santo *et al.*[25] proposed a network where a real-world MERL database is used for training the network. They simplified the training by constraining the light directions to be uniform and most importantly to have the same directions during training and test.

Ju *et al.*[26], instead of retrieving the geometry with DL, used a learning procedure for separating the RGB information from multiplectral images when solving the RGB-PS problem. This demultiplexing procedure allowed to improve the accuracy of the reconstruction when using a minimal number of light sources. Taniai and Maehara [27] proposed an unsupervised training methods that does not require any training as they minimise the reconstruction loss between the rendered images and the input images at test time. This makes the approach slower with respect to usual DL based methods as the training computational time is partially transferred to the shape reconstruction pipeline. Chen *et al.*[28] firstly introduced PS-FCN, a fully convolutional network able of being flexible regarding the knowledge of light directions. Indeed, by using similar concepts, Chen *et al.*[29] proposed a DL approach a more challenging scenario for the PS problem where the light sources are unknown. In this case, a two-stage model approximates first the uniform light directions (LCNet) and then a second step estimates the normals (NENet). Zheng *et al.*[30] proposed SplineNet to solve the Sparse PS (e.g. PS with low number of images) by generalising light directions via interpolation and exploiting symmetric and asymmetric loss functions.

Finally, Ikehata [14] introduced the observation map parameterisation (32 by 32 gray-scale image) that merges information of multiple lights on a single tensor allowing a fixed network to be used under a varied number of light sources. The training data was obtained by rendering fifteen meshes with a dense variation of material properties under a number of light directions. The purpose of this pre-rendered training dataset was to allow the network to learn the effect of global physical phenomena with simulated data with computer graphics. Although a large amount of data was sampled, the choice of selecting specific meshes limits the possible light-normal-material configurations and constrains the patterns of the global illumination effects (cast shadows, self reflections) which are a direct function of the global surface geometry. In addition, training on purely synthetic images introduces the risk of over-fitting to the synthetic distribution with potential drop of performance in real images containing noise and other effects that may not be directly modeled by the rendering engine.

In order to overcome these limitation, we propose a DL based approach for the PS problem with a better coverage of physical effects than [14] without relying on pre-rendered meshes. To do so, we implemented a pixelwise renderer (shader) so that we can generate a way bigger number of observation maps than [14]. In order to deal with global effects, invisible at single pixel level, as well as minimise the synthetic to real gap, we use an augmentation strategy to enrich the observation maps with realistic features such as shadows, ambient light, self-reflection, etc.

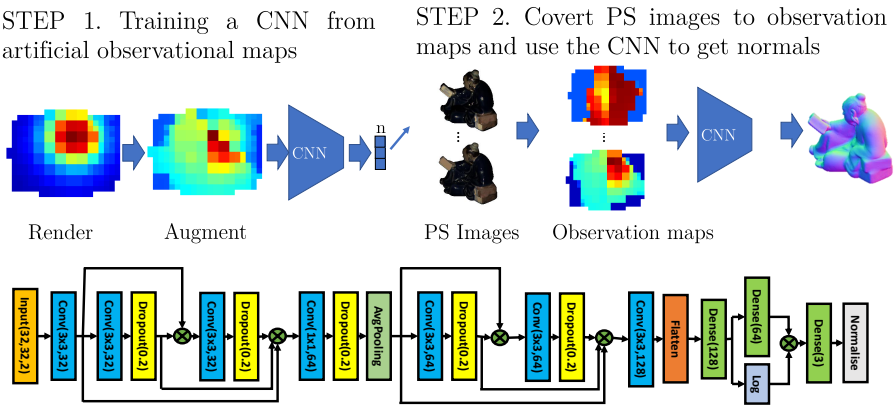


Fig. 2. This figure illustrates two key steps of our proposed approach. On the top left,the network training is illustrated consisting of rendering synthetic observation maps (3.2) and augmenting them (3.3) so as to build robustness to global illumination and other realistic effects. On the top right, the normal estimation process is shown: for each pixel, the observation map is computed by combining the information from all PS images into a single tensor (3.1). The maps are then processed by a CNN which regresses a normal (orientation) map. The complete CNN architecture is shown on the bottom row.

3 Method

This section explains our PS approach aimed at recovering surface normals from a set of PS images. In particular, (3.1) describes the mathematical formulation of the normal estimation problem; (3.2) provides a detailed explanation of how pixel-wise training data is rendered and (3.3) augmented.

3.1 Normal estimation

Our calibrated PS approach works on a set of m varied illumination images with the light directions $\mathbf{L}_j$ and the brightness ϕ_j assumed to be known for all light source j. For each pixel p, the pixel value at image j is denoted as $i_{j,p}$. The objective is to recover the normal at each pixel $\mathbf{N}_p$. This is achieved by combining all *observations* of the pixel in the images with varied illumination into a single $d \times d \times 2$ tensor which in turn is fed into a CNN which regresses normals. A high level diagram of this procedure is shown in Figure 2. A more detailed explanation is given below.

Observation map. [14] introduced the concept of the observation map as a way to merge information of a variable number of images into a single $d \times d$ image tensor. The mapping procedure follows two steps: Firstly, normalised observations $\hat{i}_{j,p}$ are computed by compensating for the light sources brightness variation and dividing with the maximum:

$$\hat{i}_{j,p} = \frac{i_{j,p}/\phi_j}{\max_j(i_{j,p}/\phi_j)}. \tag{1}$$

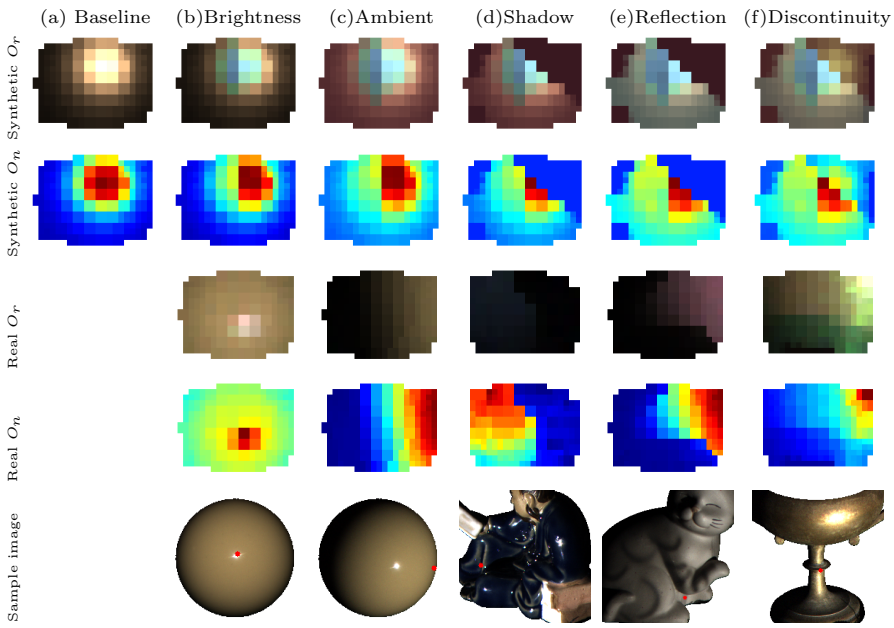


Fig. 3. Demonstration of the observation map augmentation process that builds a level of realism to the pixelwise rendered observation maps. RAW RBG maps and normalised gray (Components O_r and O_n in Equation 3) are shown. (a) is the baseline map before any augmentations are applied. (b) shows the change when variable light source brightness are considered (different pixels saturate at different levels and so the normalisation with the brightness distorts the specular highlight). (c) shows the addition of ambient light that acts as an additive offset everywhere. (d) cast shadow block all light expect ambient in regions of the map. (e) Self reflection further increases the brightness in non shadow regions. In the real data, it can be noticed by the fact that the mostly gray cat contains red pixels at the reflection point. (f) Points at the sharp edge of the cup exhibit discontinuity (which looks like the mixing of two different maps).

This normalisation observation is aimed at compensating for the albedo variation of different pixels, hence reducing the range of data. Secondly, the normalised observations $\hat{i}_j$ (omitting dependence on p for clarity) are placed on an image, the normalised observation map O_n, by projecting the light source direction $\mathbf{L}_j = [l_j^x, l_j^y, l_j^z]$ to a square of size d as follows:

$$O_n\Big(\Big\lfloor d\frac{l_j^x+1}{2}\Big\rfloor, \Big\lfloor d\frac{l_j^y+1}{2}\Big\rfloor\Big) = \hat{i}_j. \tag{2}$$

One of the main problems of the normalised observation map is that the division operation can corrupt the data in two cases. Firstly, if the maximum value is saturated, the map values are overestimated. Secondly, for very dark points, the ratio operation becomes numerically unstable and any amount of noise (or just discritisation inaccuracy) is greatly amplified. In order to overcome

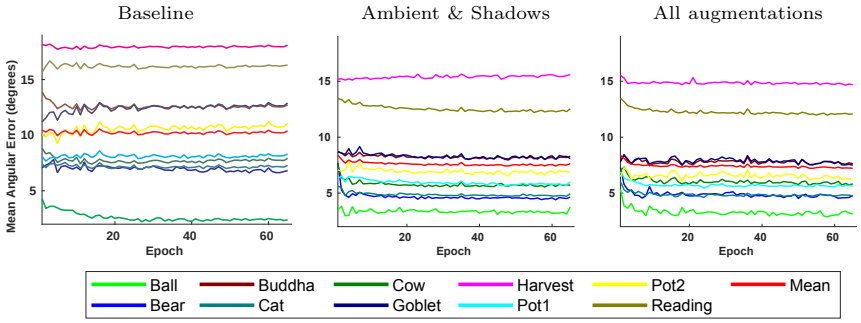


Fig. 4. MAE evolution (during training) curves illustrating the performance of the networks trained with successive augmentations. The accuracy is measured in MAE in the real DiLiGenT objects. The networks compared here are: (1) baseline/no augmentations, (2) ambient and global shadows only, (3) all augmentations. It is observed that successive augmentations improve performance by sifting downwards the error curves with only notable exception being the Ball as it suffers the least from the global illumination effects.

these limitations, we extend the observation map concept to a 3D tensor O_r which also includes a RAW gray-scale [3]channel map such as:

$$O_r\Big(\Big\lfloor d\frac{l_j^x+1}{2}\Big\rfloor, \Big\lfloor d\frac{l_j^y+1}{2}\Big\rfloor\Big) = i_j/\phi_j, \quad O=[O_r\,;O_n] \tag{3}$$

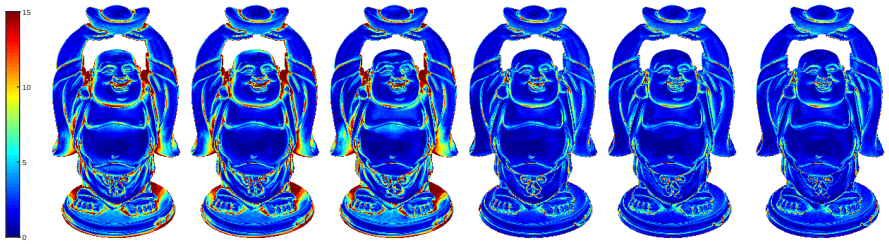
where O is a concatenation on the 3rd axis so defining a $w\times w\times 2$ tensor. Finally, these observation maps are fed into a CNN which regresses surface normal $\mathbf{N}_p$.

Network training. As in [14], we use a CNN for regressing normals from observational maps. The architecture is an extension of [14] (which in turn is is a simplified variant of DenseNet [31]) and consists of 7 convolutional layer, 3 fully connected layers and a layer computing a logarithm. The convolutional layer are used in order to learn robust features for dealing with real data (augmentation strategy, see below) and the fully connected plus logarithm layer at the end is mimicking the Blinn-Phong [5] refletance model which is shown to be a good first level approximation of many real BRDFs. The network has around 4.5M parameters in total. A full diagram of the architecture is shown in Figure 2.

3.2 Reflectance rendering

Our CNN is trained by synthetically per-pixel rendered observation maps. This is done by independently sampling normals, materials and lights which are fed to a pixelwise renderer (shader) generating a basic reflectance component b. No global rendering of objects is required and the maps are rendered on the fly during the network training. The first part of the map rendering process is achieved by using our implementation of the *Dinsey* BRDF from [16]. In addition, in order to be able to deal with more general reflections, a portion of

[3] We investigated RAW RGB vs RAW gray and found that gray was better.



(a)Baseline 12.80^o (b)Brightness 12.50^o (c)Ambient 12.37^o (d)Shadow 8.19^o (e)Reflection 8.01^o (f)Discontinuity 7.70^o

Fig. 5. Demonstration of the effect of incremental augmentations in the performance of our PX-CNN-PS on the 'Buddha' of DiLiGenT real dataset. (a) shows the result with the baseline network. For the rest of the augmentations, most of the improvement is at: (b) specular highlights middle of pot, (c) top of head, (d) significantly in most concave regions, (e) middle of head (f) sleeves.

the training data was rendered using BRDF data from the MERL real BRDF reflectance database [32]. Finally, in order to expand our material coverage even further, we also included virtual materials by blending MERL materials with a Lamberian reflection component ($\mathbf{N} \cdot \mathbf{L}$).

3.3 Rendering augmentations

In order for the proposed approach to be applicable to real data, the synthetically rendered observation maps are augmented so as to include deviation from the pure BRDF reflectance (output of the pixelwise renderer). These deviations include global illumination effect due to interaction of the incoming/reflected light with other parts of the surface. In addition, we also model some local effects as explained in detail bellow. The most important **global effects** are:

- **Cast shadows.** They are the result of a part of the surface blocking the light for a number of light sources, essentially turning the reflectance to zero for a region of the map. This can be indicated with a binary variable $s \in \{0, 1\}$ which is 0 in case of shadow, 1 otherwise. This is similar to the blacking out a random region of map performed by [33], with a notable difference being that the other augmentations are still applied in the shaded region (see bellow) resulting to non-zero values at these pixels.
- **Self reflections.** Self reflections occur in specular object as the result of light using parts of the surface as auxiliary light sources. This is harder to exactly model in practice compared to the shadow augmentation, as potentially hundreds of surface points could be contributing into this effect. To simulate this effect, we employ a simple piece-wise constant approximation as self reflection is likely to be smooth ([34]). In addition, we assert that cast shadows are likely to be partially blocking self reflections are well. Thus, we chose to model self reflections with two constants, one in the cast shadow regions, termed r_L, and a higher one, r_H, for the rest of the map.

– **Surface discontinuity.** It is common to assume that each pixel corresponds to the reflection of a single surface point with a specific normal (e.g. differential approaches like [35] assuming C^2 continues surface). However, in practice, pixels have a finite size and thus it is likely that they are the superposition of the reflectance of multiple surface points with potentially different surface normals. This effect is mostly relevant at surface discontinuity points, i.e. occlusion boundaries and sharp edges (Lipschitz continuity). As the reflectance is a non-linear function of the surface normal, this mixing effect needs to be accounted as well. This is implemented by sampling multiple ($t \in \{1, 2, 3\}$) normals $\mathbf{N}_k$ per pixel and then average out the respective intensity ($b(\mathbf{N}_k)$ see bellow).

We also consider the following **local effects**:

– **Ambient light.** Real images contain some amount of ambient light, mostly due to light dispersing into the atmosphere and reflecting on other objects in the environment. Even if the PS images are captured in a dark room with no reflective objects, this effect still persists even though it can be very small ($\approx$ 0.1% of the maximum intensity). We follow the standard practice in the literature (e.g. [36]) and model it with an additive offset. Note the similarity between ambient and self reflection; thus from a practical perspective, these two effects can be jointly addressed with the two[4] self-reflection constants (r_H, r_L).
– **Light source brightness.** Different real light sources have varied brightness ϕ. The observation map parameterisation aims to compensate for that through the division compensation however, in practice, pixel saturation makes this compensation imperfect and thus needs to be augmented for. The practical implementation involves sampling a brightness value ϕ, multiply the reflectance with this value, apply the rest of the augmentation and then apply discretisation and saturation[5]. The discretisation and saturation function will be referred to with the symbol $D(\cdot)$. Thus for saturated pixels, the final division to create an observation map does not fully compensate the light source brightness. Note that as the brightness is different for different channels , this results into specular highlights not being completely white in the brightness compensated images.
– **Noise.** Various noises are always present in real images. We assume four different types of additive and multiplicative, uniform and Gaussian noise. We empirically observed that there was a high level of correlation of noise with image brightness, therefore the most important component seems to be the multiplicative ones. We refer to the additive and multiplicative noise components as n_A, n_M respectively.

Combining all the above augmentetion effects with the base reflectance b (the superscipt k is indicating the different components that are averaged for the discontinuity compensation), the overall augmented pixel value i_j (for light source j) is calculated as follows:

$$i_j = D\Big(\frac{1}{t}\sum_{k=1}^{t}\big((\phi_j * b_j^k + r_H^k - r_L^k)s_j^k + r_L^k\big)n_{M,j}^k + n_{A,j}^k\Big). \tag{4}$$

[4] For the experiment in Table 1 including ambient and no self reflection, we use $r_H = r_L$ to get a single constant for all map pixels.
[5] We convert to 16bit integer to mimic real cameras.

Augmentation	Ball	Bear	Buddha	Cat	Cow	Goblet	Harvest	Pot1	Pot2	Reading	Mean
Baseline	2.38	6.82	12.8	7.28	7.92	12.9	18.09	8.31	11.09	16.31	10.39
+Brightness	2.79	7.09	12.5	7.02	8	12.85	18.03	7.72	10.71	16.29	10.3
+Ambient	3.14	6.54	12.37	7.23	7.4	11.43	17.25	8.16	9.31	16.14	9.89
+Shadow	3.76	4.64	8.19	4.93	5.69	8.08	15.6	5.99	6.82	12.53	7.64
+Reflection	2.64	4.54	8.01	4.81	5.87	8.32	15.08	5.88	7.5	12.5	7.51
+Discontinuity	3.13	4.8	7.7	4.76	5.82	7.49	14.68	5.62	6.25	12.08	7.23

Table 1. Demonstration of the effect of incremental augmentations in the performance of our PX-CNN-PS on the real data of DiLiGenT. It is observed that performance is almost monotonic for all objects; one notable exception is the Ball which is perfectly smooth (no discontinuities) and has no shadows so these augmentation decrease the performance for this object.

Finally, i_j are converted into an observation map as explained in Section 3.1. Visual illustration of these effects in real image maps and our synthetically rendered are shown in Figure 3. Note that the synthetic maps at Figure 3 are rendered with Diligent light to be comparable to real ones - we used random lights during train time. Detailed explanation for all the augmentation steps as well as the relevant hyperparameters can be found in the supplementary material.

4 Experimental Setup

This section describes our experimental setup including the datasets used, training procedure and evaluation protocol.

4.1 Datasets

We use two synthetic and one real dataset for evaluation. The real dataset used for the experiments is DiLiGenT [15] consisting of 10 objects of varied materials and geometry. For each object, 96 images (612×512 px) are provided along with ground truth light source directions, brightness and normal maps. For the 'Bear' object, the first 20 images are corrupted and thus were removed as reported by [14]. In addition, we generated a synthetic dataset of four objects QUEEN, PUMPKIN, ARMADILLO, DRAGON (see Figure 6). These objects are non-convex and were rendered with Blender (16 bit 512x512 px images) including realistic global illumination effects using the 96 light sources from DiLiGenT. We chose to generate this dataset instead of using CyclePSTest from [14] as the later one contains unrealistic surface material distribution (each superpixel has a different random material). In contrast all four of our objects are single material/albedo. QUEEN is purely specular, PUMPKIN is purely metallic (using the category definition of [14]) whereas ARMADILLO, DRAGON are more intermediate materials. Finally, the ability of the network to learn materials is evaluated on a synthetic dataset of spheres using the MERL materials [32]. For all 100 materials, we (pixelwise) rendered 96 spheres using the DiLiGenT lights.

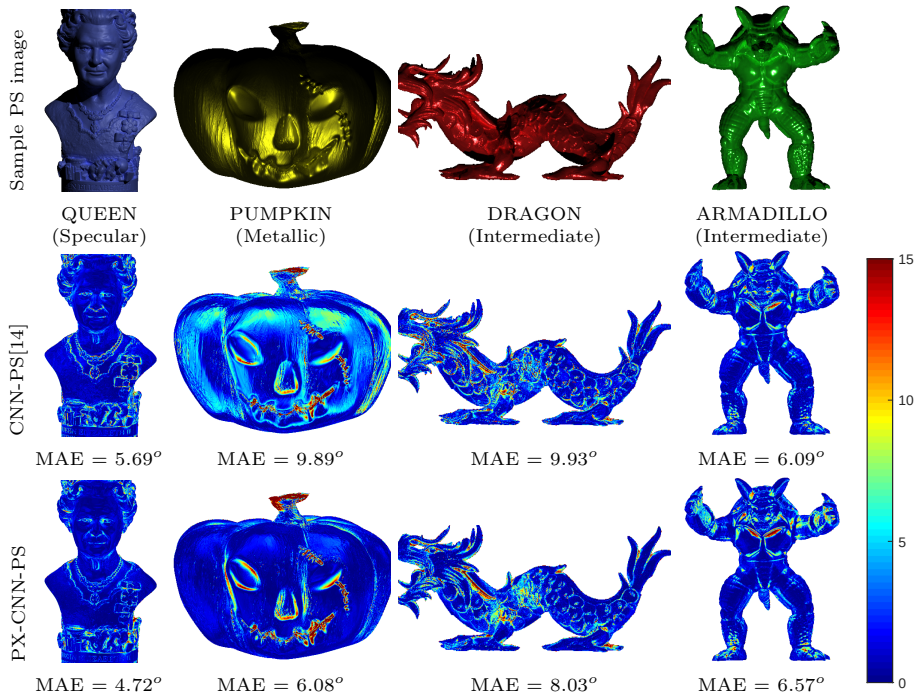


Fig. 6. Comparison of our PX-CNN-PS with CNN-PS [14] on synthetic, globally rendered objects. The proposed approach outperforms the competition in 3 out of these 4 objects and on average MAE (7.90^o vs 6.35^o).

4.2 CNN details

Training details. Baseline experiments are performed using the exact architecture of [14]. This version will be termed PX-CNN-PS. The only difference compared to [14] for this version is the training data that were made using our rendering and augmentation procedure explained in section 3.2. The purpose of PX-CNN-PS is to show that our augmentation strategy can compensate for the real and global effects and even achieve state of the art results. Final experiments are performed with our modified version of the architecture called PX-NET which achieves much better results.

Implementation. The network was implemented in Keras of Tensorflow 2.0. The rendering engine was implemented in Python and c++ with the only external dependence being opencv for basic vector algebra and i/o. We trained the network using the mean squared error loss function on the normalised normals using the default settings of the Adam optimiser.

Hyper parameters. The training batch size was set at 2400 with 5000 batches per epoch (12 million maps). We trained for 65 epochs which took around one day on a NVIDIA GeForce RTX 2080Ti (i.e. a little over 20 min per epoch). The train light distribution was set to 50-1000 random lights in order to have a

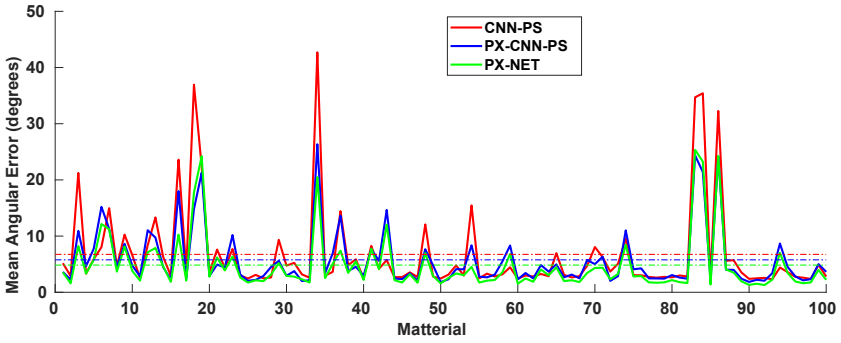


Fig. 7. Comparison of CNN-PS [14] versus our two networks (PX-CNN-PS ,PX-NET) artificially rendered MERL images using DiLiGenT 96 lights. All results are shown for a single prediction (k=1) with corresponding mean errors 6.7 5.8 4.8 also shown as horizontal lines.

fair comparison with [14]. The exact numbers for the rest of the augmentations are described in the supplementary material.

4.3 Evaluation protocol

The evaluation metric for all the experiments is the mean angular error (MAE) between the predicted normals and the ground truth ones (in degrees). Normal error maps showing normal error (in degrees) per pixel offer a qualitative evaluation of the method.

Rotation psedo-invariance: [14] notes that observation maps can be rotated in order to perform a test time augmentation (using 10 rotation which is termed as K=10). If this augmentation is not used (which is the default choice in the paper unless otherwise stated), the single network evaluation is termed K=1.

5 Experiments

In this section we present several experiments demonstrating state of the art performance in the datasets described in Section 4.1.

Local to Global-Augmentations. Our first experiment aimed at evaluating the effect of the different augmentations and demonstrating how the network trained with per-pixel rendered data can even outperform the network trained with globally rendered train data. For that, we first trained a series of networks with the exact same architecture of CNN-PS [14], which we refer to as PX-CNN-PS, and observed the effect of incrementally applying the series of augmentations. Note that we only used materials sampled from the Dinsey BRDF.The evaluation is performed on the real DiLiGenT dataset. This can be seen in Figure 4 and Figures 5 as well as in Table 1. We observed that the of augmentations monotonically improve performance on most objects as well as the average error across the whole dataset.

It is noted that the final step (including all geometric augmentations) outperform CNN-PS (see Table 2) which is trained with globally rendered data. This can be explained by the following tree reasons. Firstly, the synthetic training

Method	Ball	Bear	Buddha	Cat	Cow	Goblet	Harvest	Pot1	Pot2	Reading	AVG
Baseline [1]	4.1	8.4	14.9	8.4	25.6	18.5	30.6	8.9	14.7	19.8	15.39
SPLINE-Net[30]	1.74	4.65	9.14	5.48	9.55	9.38	24.44	5.91	7.9	12.77	9.1
ICML [27]	1.47	5.79	10.36	5.44	6.32	11.47	22.59	6.09	7.76	11.03	8.83
Exemplars [37]	1.33	5.58	8.48	4.88	8.23	7.57	15.81	5.16	6.41	12.08	7.55
CNN-PS[14], K=1	2.7	4.5	8.6	5	8.2	7.1	14.2	5.9	6.3	13	7.55
CNN-PS[14], K=10	2.2	4.1	7.9	4.6	8	7.3	14	5.4	6	12.6	7.21
PX-CNN-PS, K=1	3.13	4.8	7.7	4.76	5.82	7.49	14.68	5.62	6.25	12.08	7.23
PX-CNN-PS, K=10	2.52	4.06	7.63	4.4	5.86	7.8	14.49	5.2	6.43	11.76	7.01
PX-NET, K=1	2.50	3.80	7.29	4.54	5.21	7.12	14.75	5.11	5.18	11.40	6.69
PX-NET, K=10	2.15	3.64	7.13	4.30	4.83	7.13	14.31	4.94	4.99	11.02	6.45

Table 2. Quantitative comparison of the proposed method (both simplified PX-CNN-PS and full PX-NET) on the DiLiGenT real benchmark [15]. For our networks as well as [14] results using K=10 (see section 4.3) are also presented for completeness.

data of CNN-PS do not include some of the effect we are modeling here namely light source brightness variation, noise and surface discontinuity (which is highly depend on the geometry of the training meshes).Secondly, CNN-PS was trained on a limited subset of Disney material parameters as global image rendering is slow. Finally, it is likely, that CNN-PS did overfit on the specific distribution of global effect of its training data which was made with only 15 meshes.

The superiority of our training data compared the ones of CNN-PS's is also confirmed on the synthetic, globally rendered, objects of Figure 6. PX-CNN-PS outperforms CNN-PS in 3 out of these 4 objects and on average MAE. The most significant difference is on the PUMPKIN object (6.08^o vs 9.89^o) and can be explained by the fact that CNN-PS does not include the discontinuity augmentation which is important as this object's surface is rough and detailed.

Evaluation on rendered MERL images. The next synthetic experiment (Figure 7) compares both of our networks PX-CNN-PS (with all augmentations) and PX-NET with CNN-PS on the synthetic images rendered with MERL materials [32]. The aim of this evaluation is to demonstrate that our networks can deal with various real world reflectances. It is noted that PX-CNN-PS outperforms CNN-PS (6.7^o vs 5.8^o MAE) and this can be attributed to the fact that we sampled the whole set of parameters of Disney instead of a limited subset. Various materials are quite a big amount of error and this signifies that the Disney BRDF may not be adequate for representing the entirety of MERL materials. In fact, this observation motivated us to include a material augmentation using MERL data to our final network version PX-NET. It is interesting that although the MAE is reduced to 4.8^o, there are still materials which are not very well modelled and thus this gives motivation for future research.

Real dataset - DiLiGenT. Finally, we compare our 2 networks with other state of the art method in the full DiLiGenT [15] dataset in Table 2. For completeness, we also include the results after applying the test time rotation pseudo-invariance augmentation (K=10) (for us and CNN-PS). Also three sample error maps are shown in Figure 8 (for the K=1 network evaluation). Both of our networks significantly outperform the competition in the average error as well in almost all

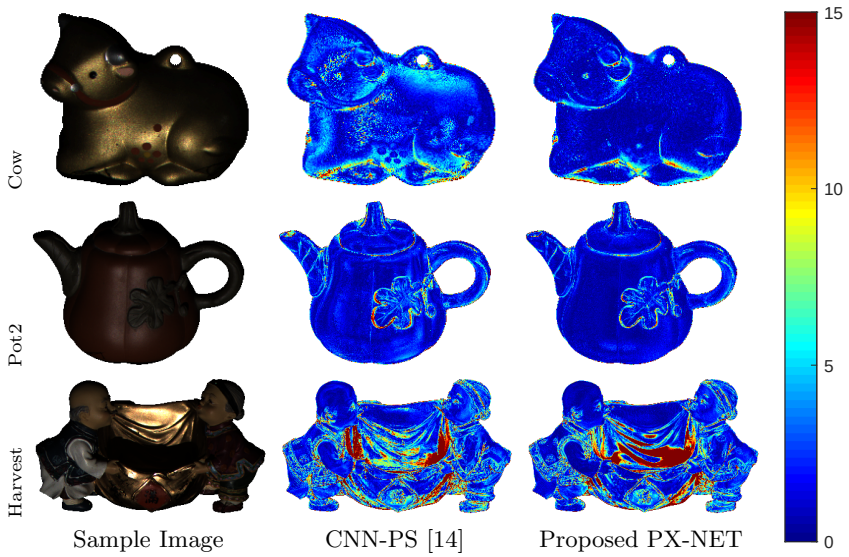


Fig. 8. Some sample error maps from Table 2 (for k=1) comparing our PX-NET to CNN-PS [14]. It is noted that we significantly outperform the competition in convex region due to using more broad set of materials at train time (this is evident on the COW which coated with metallic paint that is quite different that the metallic materials considered by CNN-PS). The Pot2 error map demonstrates the strength of our discontinuity augmentation on the leaf boundary. Finally, the Harvest error map shows that we can outperform in some concave regions (bellow left head); however regions with complicated self-reflection patterns (middle of the image) are a potential limitation to our approach (due to the piece-wise constant , self reflection model assumed).

objects individually. The success of our method can be attributed on the ability of the network to deal with real world materials with complex reflectance (we exhibit very minimal error in convex regions where the PS problem reduced to BRDF inversion) as well as simultaneously being very robust to global illumination effects due to our augmentation strategy.

6 Conclusion

In this work we presented a new, simple and more efficient concept for generating in-line training data for solving the PS problem, using simple pixel rendering. We deal with global effects like shadows, self-reflections, etc. by adopting an augmentation strategy based on real and synthetic data observations.

We analysed the performance of our approach while progressively augmenting the training data and we quantitatively showed the actual benefits in adopting such an augmentation strategy. We proved that the results are improved in regions with limited global physical effects. The reason for this is that we provide a first order augmentation of global effects by adding physically consistent offset. A future interesting work is to find a better way to augment pixel-wise rendered data with the aim to include higher order approximation of global effects.

Also, beside outperforming state-of-the-art results, our approach can be easily customised to perform better on more specific setups. This can be easily achieved by modifying the range of parameters used to generate in-line training data. For example, for a very specific lighting setup, it would make sense to directly train our network using the actual light directions as this would increase the accuracy of the predicted normals.

Finally, the proposed approach could easily be extended to deal with more challenging scenarios such as near field illumination [38]. Although the global physical effects remain the same, the spacial positioning of the light sources still remain a challenging feature to include in the feature space.

References

1. Woodham, R.J.: Photometric method for determining surface orientation from multiple images. Optical Engineering (1980)
2. Lambert, J.H.: Photometrie. (1760)
3. Horn, B.K.P.: Obtaining shape from shading information. The Psychology of Computer Vision, Winston, P. H. (Ed.) (1975)
4. Phong, B.T.: Illumination for computer generated pictures. Communications of the ACM **18** (1975) 311–317
5. Blinn, J.F.: Models of light reflection for computer synthesized pictures. In: SIGGRAPH. (1977)
6. Cook, R.L., Torrance, K.E.: A reflectance model for computer graphics. ACM Transactions on Graphics (1982)
7. Lafortune, E.P., Foo, S.C., Torrance, K.E., Greenberg, D.P.: Non-linear approximation of reflectance functions. In: SIGGRAPH. (1997)
8. Torrance, K.E., Sparrow, E.M.: Theory for off-specular reflection from roughened surfaces. Journal of the Optical Society of America (1967)
9. Ward, G.J.: Measuring and modeling anisotropic reflection. SIGGRAPH (1992)
10. Watson, R.M.J., Raven, P.N.: Comparison of measured BRDF data with parameterized reflectance models, International Society for Optics and Photonics, SPIE (2001)
11. Havran, V., Filip, J., Myszkowski, K.: Perceptually motivated BRDF comparison using single image. Comput. Graph. Forum (2016)
12. Ngan, A., Durand, F., Matusik, W.: Experimental analysis of BRDF models. In: EUROGRAPHICS. (2005)
13. Ngan, A., Durand, F., Matusik, W.: Experimental validation of analytical BRDF models. In: SIGGRAPH. (2004)
14. Ikehata, S.: Cnn-ps: Cnn-based photometric stereo for general non-convex surfaces. In: ECCV. (2018)
15. Shi, B., Wu, Z., Mo, Z., Duan, D., Yeung, S.K., Tan, P.: A benchmark dataset and evaluation for non-lambertian and uncalibrated photometric stereo. In: CVPR. (2016)
16. Burley, B.: Physically-based shading at disney. In: SIGGRAPH Course Notes. (2012)
17. Hill, S., McAuley, S., Burley, B., Chan, D., Fascione, L., Iwanicki, M., Hoffman, N., Jakob, W., Neubelt, D., Pesce, A., Pettineo, M.: Physically based shading in theory and practice. In: SIGGRAPH. (2015)

18. Ikehata, S., Aizawa, K.: Photometric stereo using constrained bivariate regression for general isotropic surfaces. In: CVPR. (2014)
19. Quéau, Y., Mecca, R., Durou, J.D.: Unbiased photometric stereo for colored surfaces: A variational approach. In: CVPR. (2016)
20. Blender-Online-Community: Blender - a 3D modelling and rendering package. Blender Foundation. (2018)
21. Ackermann, J., Goesele, M.: A survey of photometric stereo techniques. Foundations and Trends in Computer Graphics and Vision (2015)
22. Tang, Y., Salakhutdinov, R., Hinton, G.E.: Deep lambertian networks. In: ICML. (2012)
23. Hinton, G.E.: Deep belief networks. Scholarpedia (2009)
24. Yu, Y., Smith, W.A.P.: Pvnn: A neural network library for photometric vision. In: ICCV Workshop. (2017)
25. Santo, H., Samejima, M., Sugano, Y., Shi, B., Matsushita, Y.: Deep photometric stereo network. In: ICCV Workshops. (2017)
26. Ju, Y., Qi, L., Zhou, H., Dong, J., Lu, L.: Demultiplexing colored images for multispectral photometric stereo via deep neural networks. IEEE Access (2018)
27. Taniai, T., Maehara, T.: Neural inverse rendering for general reflectance photometric stereo. In: ICML. (2018)
28. Chen, G., Han, K., Wong, K.K.: PS-FCN: A flexible learning framework for photometric stereo. In: ECCV. (2018)
29. Chen, G., Han, K., Shi, B., Matsushita, Y., Wong, K.Y.K.: Self-calibrating deep photometric stereo networks. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 8739–8747
30. Zheng, Q., Jia, Y., Shi, B., Jiang, X., Duan, L.Y., Kot, A.C.: Spline-net: Sparse photometric stereo through lighting interpolation and normal estimation networks. In: CVPR. (2019)
31. Huang, G., Liu, Z., Van Der Maaten, L., Weinberger, K.Q.: Densely connected convolutional networks. In: CVPR. (2017)
32. Matusik, W., Pfister, H., Brand, M., McMillan, L.: A data-driven reflectance model. ACM Transactions on Graphics (2003)
33. Li, J., Robles-Kelly, A., You, S., Matsushita, Y.: Learning to minify photometric stereo. In: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition. (2019) 7568–7576
34. Nayar, S.K., Krishnan, G., Grossberg, M.D., Raskar, R.: Fast separation of direct and global components of a scene using high frequency illumination. In: ACM SIGGRAPH 2006 Papers. (2006) 935–944
35. Mecca, R., Quéau, Y., Logothetis, F., Cipolla, R.: A single lobe photometric stereo approach for heterogeneous material. SIAM Journal on Imaging Sciences **9** (2016) 1858–1888
36. Logothetis, F., Mecca, R., Quéau, Y., Cipolla, R.: Near-field photometric stereo in ambient light. In: BMVC. (2016)
37. Hui, Z., Sankaranarayanan, A.C.: Shape and spatially-varying reflectance estimation from virtual exemplars. IEEE Transactions on Pattern Analysis and Machine Intelligence (2016)
38. Mecca, R., Wetzler, A., Bruckstein, A., Kimmel, R.: Near Field Photometric Stereo with Point Light Sources. SIAM Journal on Imaging Sciences (2014)

A Appendix

This appendix provides supplementary material for the main submission. Section A.1 examines the evolution of PX-NET accuracy over number of epochs trained. Section A.2 provides an in depth explanation of the augmentations used. Finally, Section A.3 contains additional visualisations of the qualitative results on the DiLiGent dataset.

A.1 Network Convergence

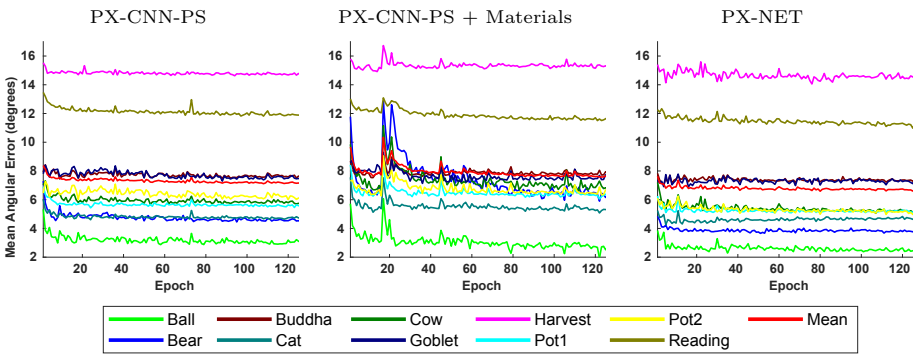
This section analyses the convergence of PX-CNN-PS after the introduction of the material augmentation (see Section A.2). This was one of the main reasons why we introduced PX-NET which could benefit from the increased material data.This is shown in Figure 9 with the different MAE trends for the DiLiGent objects and makes clear how PX-CNN-PS with the additional material augmentations performs worse than the proposed PX-NET. All three networks were also trained for 130 epoch (about 2 days) in contrast to the 65 epoch that we present in the main paper and the mean and standard deviation of the MAE is examined to have an unbiased comparison.

A.2 Augmentations

This section provides a more in depth explanation of our augmentation protocol which aims to give a level of realism to the pixelwise-render data. To simplify the notation, we assume that the image value i is a real number $i \in [0, 1]$ with 0 being completely black and 1 being the saturation level. We also denote a uniform real distribution in the interval $[a, b]$ as $\mathcal{U}_{\mathcal{R}}(a, b)$, a uniform integer one in the interval $[k, l]$ as $\mathcal{U}_{\mathcal{I}}(k, l)$ and a normal distribution with mean μ and standard deviation σ as $\mathcal{N}(\mu, \sigma)$.

Materials: 75% of the data are rendered using a random Disney material. All 9 parameters (excluding *anisotropy*) namely *metallic, subsurface, specular, roughness, specularTint, sheen, sheenTint, clearcoat* and *clearcoatGloss* are sampled uniformly and independently. 25% of the training data are rendered using the following material augmentation strategy: firstly a random MERL material $m \sim \mathcal{U}_{\mathcal{I}}(1, 100)$ is selected. Then a random weight $w \sim \mathcal{U}_{\mathcal{R}}(0, 1)$ is sampled. Finally, the material's BRDF B_m is mixed with a Lambertian component $\mathbf{N} \cdot \mathbf{L}$ to get overall reflectance $b = \rho\Big(wB_m(\mathbf{N}, \mathbf{L}) + (1 - w)\mathbf{N} \cdot \mathbf{L}\Big)$ where $\rho \sim \mathcal{U}_{\mathcal{R}}(0, 1)$ is the pixels albedo.

Light Source Brightness: We sample light sources brightness uniformly and independently inside the range found in the Diligent real dataset, hence $\phi \sim \mathcal{U}_{\mathcal{R}}(0.28, 3.2)$.



Network	Ep. 65 MAE	MAE Mean±STD	Best MAE	Best Ep.
PX-CNN-PS	7.22^o	7.21 ± 0.07^o	7.11^o	110
PX-CNN-PS+Mats	7.78^o	7.71 ± 0.10^o	7.58^o	122
PX-NET	6.69^o	6.72 ± 0.05^o	6.59^o	80

Fig. 9. Demonstration of how the material augmentation deteriorates the performance of PX-CNN-PS (exact architecture of [14]) , thus motivating the use of our proposed PX-NET. This is shown graphically on the top as well as in the tabular data of bottom row. The drop of performance of PX-CNN-PS due to the material augmentation is especially notable at the relatively simple 'Bear' and 'Cow' objects. The mean and standard deviation metrics in the bottom row are for the mean angular error (MAE) in the 65-130 epoch range. PX-NET achieve a significantly lower MAE as well as better convergence (smaller MAE standard deviation) than PX-CNN-PS. We note that PX-NET achieves slightly better performance at epoch 80 than the result reported in the main paper (epoch 65) but this difference is marginal and does not change any of our conclusions.

Cast Shadows: Cast shadows are a highly structured phenomenon with high correlation between nearby light sources. Therefore, we model their spacial distribution as spheres placed on the upper hemisphere of the unit circle which is the domain of the light directions. In addition, we note that for datasets consisting of continues objects, if a shadow occurs in one direction, it will likely occur in all other directions with the same azimuth and more oblique elevation angles. Finally, we note that light directions close to the north pole ($[0, 0, 1]$) cannot be blocked by shadow as this equals the viewing direction to that surface point. Hence, the shading spheres are placed only on the equator of the unit sphere ($[\cos(\theta), \sin(\theta), 0]$). More specifically, our hyper-parameters are the following: the shadow augmentation was applied to 75% of maps, containing $\mathcal{U}_{\mathcal{I}}(1, 10)$ shadows. The radius of the shading spheres was set to a truncated Gaussian with parameters $\mathcal{N}(30^o, 15^o)$.

Self reflections/Ambient light: Both self reflection and ambient light are jointly addressed with the two additive constants (r_H, r_L). r_H is aimed at modeling the high self reflection values and was applied to 50% of the training data in the non-shaded regions (setting $r_H = r_L$ in Equation 4 in the main text just

Augmentation	Probability	Magnitude
Materials	0.25	N/A
Light Brightness	1	$\mathcal{U}_{\mathcal{R}}(0.28, 3.2)$
Shadow	0.75	$\mathcal{U}_{\mathcal{I}}(1, 10)$, $\mathcal{N}(30^o, 15^o)$
Self reflections/Ambient r_H	0.5	$\mathcal{U}_{\mathcal{R}}(0, 0.1)$
Self reflections/Ambient r_L	1	$\mathcal{U}_{\mathcal{R}}(0, 0.01)$
Surface Discontinuity	0.15	$\mathcal{U}_{\mathcal{I}}(2, 3)$
Noise Multiplicative	1	$\mathcal{U}_{\mathcal{R}}(0.95, 1.05)\mathcal{N}(1, 10^{-3})$
Noise Additive	1	$\mathcal{U}_{\mathcal{R}}(-10^{-4}, 10^{-4}) + \mathcal{N}(0, 10^{-4})$
Quantisation	1	16bits

Table 3. Summary of all the augmentations used.

leaves a single constant r_L in the whole map). r_L takes into account the rest of the indirect illumination including ambient light thus we chose to apply it to all of the data points. In addition, these additive constant are made proportional to $\mathbf{N} \cdot \mathbf{V}$ in order to avoid diminishing the signal to noise ration in oblique normals. Thus, we have $r_H = \mathbf{N} \cdot \mathbf{V}\mathcal{U}_{\mathcal{R}}(0, 0.1)$ and $r_L = \mathbf{N} \cdot \mathbf{V}\mathcal{U}_{\mathcal{R}}(0, 0.01)$. Note that if $r_L > r_H$, we re-sample the values until this is not the case.

Surface Discontinuity: For this step, we allow 15% of the training data to be a combination of the rendering of 2 or 3 'subpixels' as indicated by the sum at Equation 4 in the main text. We allow for the different mixed subpixels to have different shadow and reflection distributions, which are completely independently sampled (hence the r_H^k, r_L^k s_j^k terms in the summation for the different subpixles k). In contrast, the material parameters and albedo are kept the same. We note that the overall normal (used for training the network) is simply the average of the subpixel's normals.

Noise: We apply four different types of noise namely additive and multiplicative, uniform and Gaussian. We empirically found that the most important component was the uniform multiplicative and thus was set to 5%, i.e. $\mathcal{U}_{\mathcal{R}}(0.95, 1.05)$. The rest of the hyper-parameters used were: multiplicative noise $\mathcal{N}(1, 10^{-3})$, Gaussian additive noise $\mathcal{N}(0, 10^{-4})$ and uniform additive noise $\mathcal{U}_{\mathcal{R}}(-10^{-4}, 10^{-4})$. Thus, the components n_M and n_A in Equation 4 are calculated as $n_M = \mathcal{U}_{\mathcal{R}}(0.95, 1.05)\mathcal{N}(1, 10^{-3})$ and $n_A = \mathcal{N}(0, 10^{-4}) + \mathcal{N}(1, 10^{-3})$

Discretisation: Finally, at the end of the rendering process, we also apply 16 bit discretisation (to be in line with most real data in the form of 16bit png images) and saturation to the overall pixel intensity, before proceeding to the map generation step. The discretisation and saturation function $D(x)$ has the following formulation:

$$D(x) = \begin{cases} 2^{16} - 1, & \text{if } x \geq 1 \\ 0, & \text{if } x \leq 0 \\ \text{uint16}\Big((2^{16} - 1)x\Big), & \text{otherwise} \end{cases}$$

All of the augmentations used are summarised in Table 3 and their application order is shown in Equation 4 in the main document .

A.3 Diligent Results

This section contains full visual comparison with CNN-PS [14] for all Diligent objects at Figures 10 and 11.

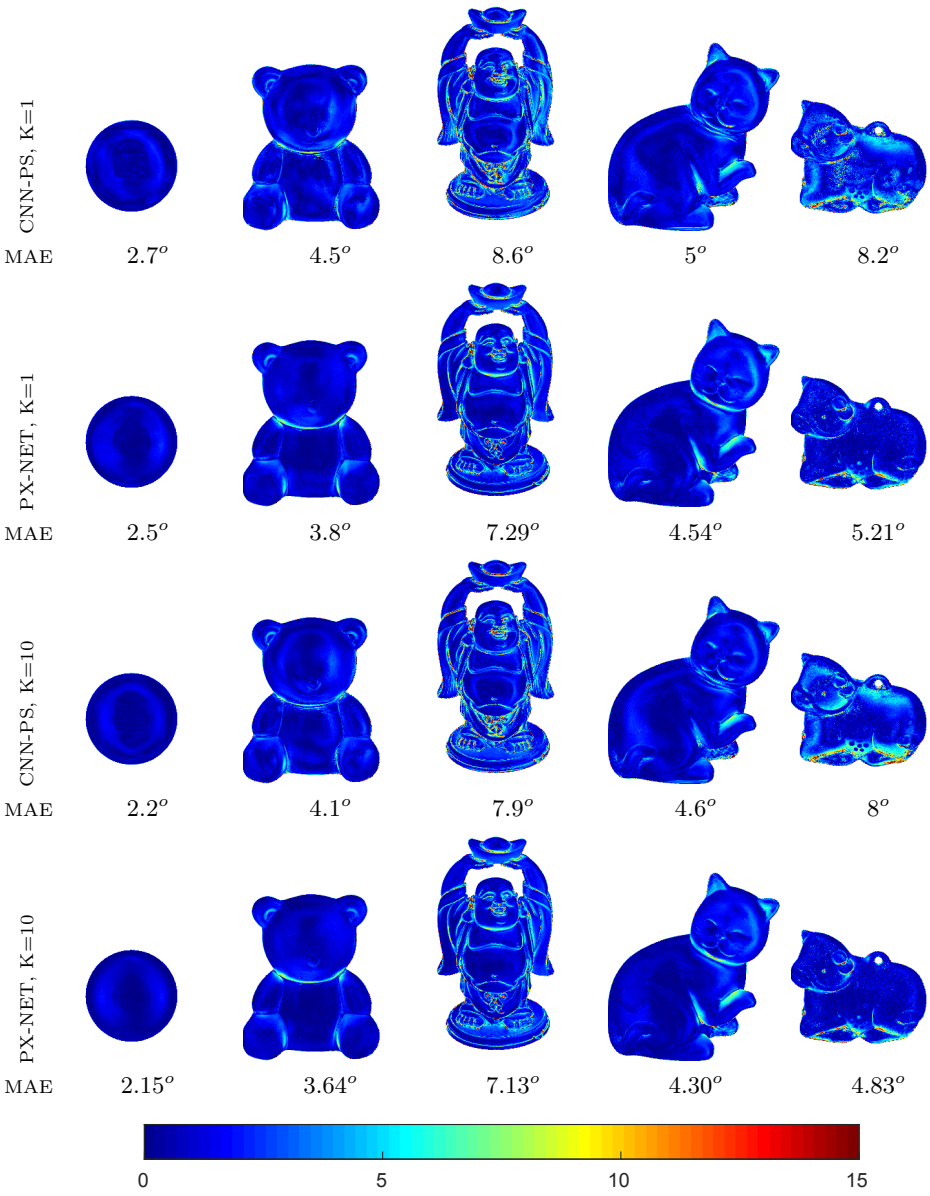


Fig. 10. Visual comparison [1/2] of CNN-PS [14] with the proposed PX-NET (Table 2 of the main paper) for both K=1 and K=10 of the on the Diligent dataset.

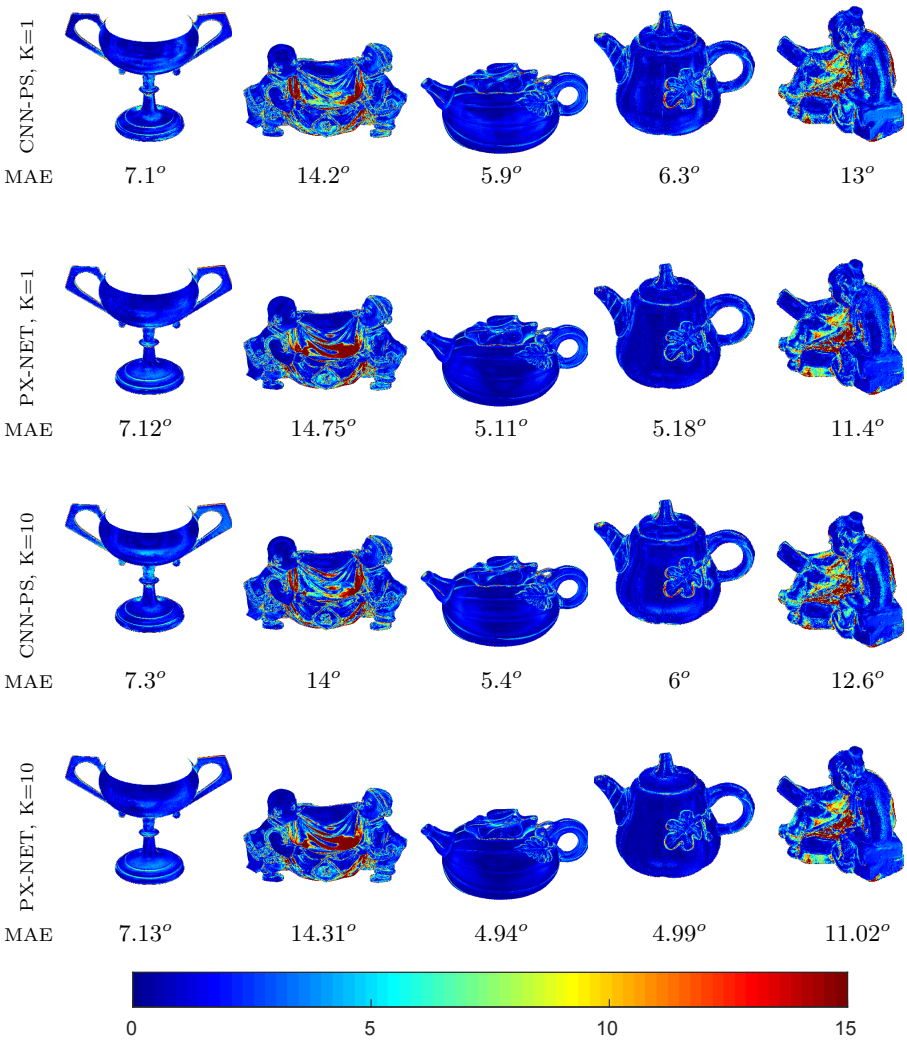


Fig. 11. Visual comparison [2/2] of CNN-PS [14] with the proposed PX-NET (Table 2 of the main paper) for both K=1 and K=10 of the on the Diligent dataset.