

Weakly-Supervised Contrastive Learning in Path Manifold for Monte Carlo Image Reconstruction

IN-YOUNG CHO, YUCHI HUO, and SUNG-EUI YOON, KAIST, Republic of Korea

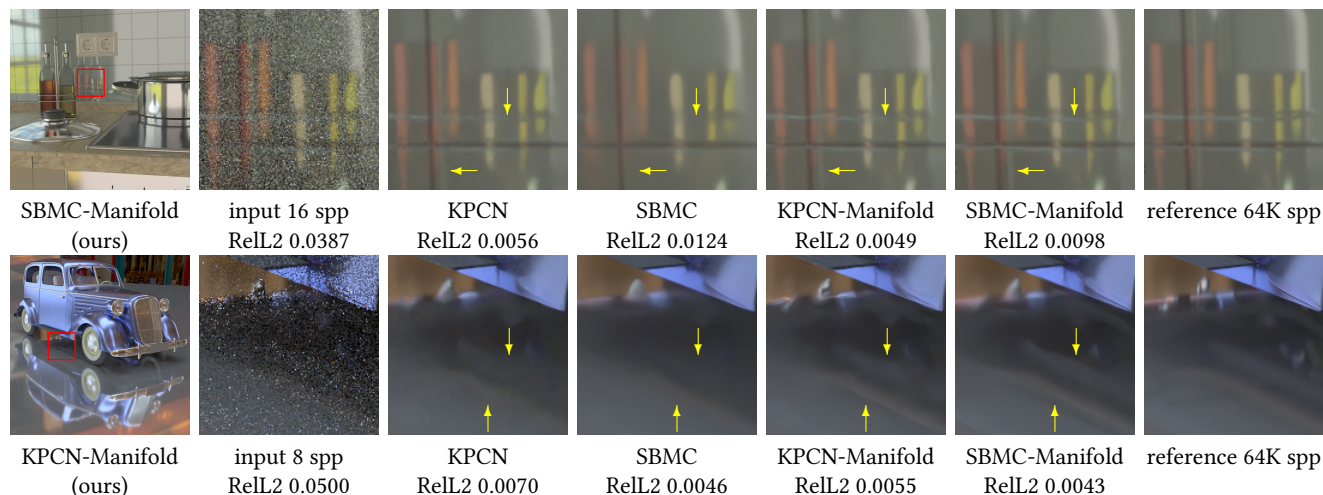


Fig. 1. We propose a path-space manifold learning framework to enhance Monte Carlo reconstruction networks. In this figure, KPCN [Bako et al. 2017] and SBMC [Gharbi et al. 2019] are respectively extended to KPCN-Manifold and SBMC-Manifold by our framework. The manifold models reconstruct an oil bottle holder reflected on the kitchen tile or seen through the oil bottle (first row) and tailpipes dimly reflected on the floor (second row) better than their vanilla counterparts, showing lower numerical errors. "My Kitchen" by tokabilitor under CC0. "Old vintage car" by piopis under CC0.

Image-space auxiliary features such as surface normal have significantly contributed to the recent success of Monte Carlo (MC) reconstruction networks. However, path-space features, another essential piece of light propagation, have not yet been sufficiently explored. Due to the curse of dimensionality, information flow between a regression loss and high-dimensional path-space features is sparse, leading to difficult training and inefficient usage of path-space features in a typical reconstruction framework. This paper introduces a contrastive manifold learning framework to utilize path-space features effectively. The proposed framework employs weakly-supervised learning that converts reference pixel colors to dense pseudo labels for light paths. A convolutional path-embedding network then induces a low-dimensional manifold of paths by iteratively clustering intra-class embeddings, while discriminating inter-class embeddings using gradient descent. The proposed framework facilitates path-space exploration of reconstruction networks by extracting low-dimensional yet meaningful embeddings within the features. We apply our framework to the recent image- and sample-space models and demonstrate considerable improvements, especially on the sample space. The source code is available at <https://github.com/Mephisto405/WCMC>.

Authors' address: In-Young Cho, ciy405x@gmail.com; Yuchi Huo, huo.yuchi.sc@gmail.com; Sung-Eui Yoon, sungeui@kaist.edu, KAIST, Daejeon, Republic of Korea.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than ACM must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.

© 2021 Association for Computing Machinery.

0730-0301/2021/8-ART38 \$15.00

<https://doi.org/10.1145/3450626.3459876>

CCS Concepts: • **Computing methodologies** → **Neural networks; Dimensionality reduction and manifold learning; Ray tracing.**

Additional Key Words and Phrases: Monte Carlo image reconstruction, contrastive learning, weakly-supervised learning

ACM Reference Format:

In-Young Cho, Yuchi Huo, and Sung-Eui Yoon. 2021. Weakly-Supervised Contrastive Learning in Path Manifold for Monte Carlo Image Reconstruction. *ACM Trans. Graph.* 40, 4, Article 38 (August 2021), 14 pages. <https://doi.org/10.1145/3450626.3459876>

1 INTRODUCTION

Image reconstruction such as denoising is an ill-posed problem in computer science. Fortunately, predefined 3D scenes provide fruitful guidance on Monte Carlo (MC) reconstruction to recover details while reducing artifacts. Although MC rendering suffers from severe noise at low sample counts due to its stochastic nature, reconstruction models often yield visually impressive results thanks to strong constraints such as normal, depth, and texture [Rousselle et al. 2013]. The geometry buffer (G-buffer) has shown a high correlation with the reference image, especially where diffuse reflections are dominant. As a result, regression-based reconstruction can effectively reduce random noise from MC integration [Moon et al. 2014, 2016]. Moreover, the image-space auxiliary features have led to significant improvements in the recent deep reconstruction [Bako et al. 2017; Chaitanya et al. 2017; Kettunen et al. 2019; Xu et al. 2019].

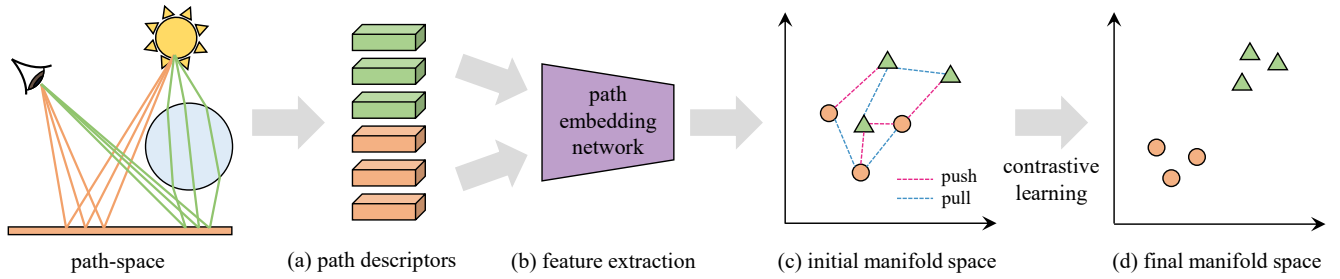


Fig. 2. Illustration of our weakly-supervised path-space contrastive learning. (a) For each path, we extract a path descriptor, a sequence of the path’s radiometric quantities at each vertex. Pseudo colors highlight the similarity between paths. (b) We use a sample-based convolutional network to transform path descriptor vectors into a low-dimensional space. (c) The network initially produces a poorly-structured manifold space. (d) As training goes on, our contrastive manifold learning framework refines the manifold space by the proposed sample-to-sample optimization; paths cluster together if they are sampled from pixels with similar reference colors, while pushing each other away in the opposite case. That is, we use reference pixel radiance as pseudo labels for path clustering. Our path-space contrastive learning separates the overlapped path distributions, providing distinguishable yet compact features to MC reconstruction networks. The embeddings are fed to a reconstruction network along with G-buffer to help MC reconstruction.

Most prior methods exploit the first bounce features from high-dimensional paths, and some of them utilize path-space decomposition [Zimmer et al. 2015] to obtain indirect features manually when the first few interactions are specular or near-specular [Bako et al. 2017; Gharbi et al. 2019]. We argue that these features do not provide a sufficient representation of various light phenomena for reconstruction networks. Since path tracing, for example, involves a sequence of scattering as shown in Fig. 1, a representation of light propagation is inherently high-dimensional. However, learning meaningful patterns between high-dimensional paths and reference images is still challenging due to the low correlation and high sparsity of path samples. Recent studies report that deep neural networks often struggle to explore the sparse space [Huo et al. 2020; Müller et al. 2019; Zheng and Zwicker 2019].

Main contributions. This study proposes a manifold learning framework that allows MC reconstruction models to fully utilize *paths*, containing not only first bounce features but also multi-bounce features (Fig. 2). Moreover, our framework aims to extract compact and useful embeddings of high-dimensional path features to remedy the sparsity of path-space. To achieve this goal, we leverage the recent deep manifold learning studies and their contrastive approaches, which cluster input data for downstream tasks such as classification and regression [Chen et al. 2017, 2020; Sun et al. 2014; Wu et al. 2018]. In the end, we successfully identify dense clusters in path manifold (Fig. 4) and exploit the information in the image- and sample-space MC reconstruction. Our contributions are as follows:

- We propose weakly-supervised contrastive learning, an orthogonal design to vanilla regression framework, to leverage path-space features for improving MC reconstruction (Sec. 4).
- We present path disentangling loss to directly learn the correlation between paths and alleviate the sparsity of path space (Sec. 4.3).
- We demonstrate that our training framework supplements vanilla image- and sample-space models, producing numerically and visually improved results (Sec. 6, Fig. 6, and Fig. 7).

We hope that our work takes a meaningful step toward utilizing path-space and triggers fruitful future work (Sec. 7).

2 RELATED WORK

This section discusses the recent trend in deep learning-based reconstruction methods for MC path tracing, followed by manifold learning approaches in computer vision and graphics applications.

2.1 MC Reconstruction with Deep Learning

Kalantari et al. [2015] first adopt a machine learning technique in MC rendering to reconstruct smooth images from noisy inputs. Following the pioneering work, Chaitanya et al. [2017] propose a recurrent convolutional neural network (RCNN), which processes MC images with extra features to predict the final denoised output directly. The work exploits temporal coherence of sequential images, producing temporally stable results in restricted lighting conditions.

Bako et al. [2017] and Vogels et al. [2018] both present convolutional neural network (CNN) approaches to predict per-pixel filtering kernels. Gharbi et al. [2019] extend these approaches and describe a CNN model that predicts per-sample filtering kernels. This sample-based method shows substantial results in suppressing high-energy outliers compared to prior pixel-based methods, which only take low-order statistics (e.g., mean and variance) of radiance samples. The main drawback of the sample-based approach was the high computational cost and memory consumption, but Munkberg and Hasselgren [2020] recently propose a cost-efficient reconstruction method distributing samples into multiple image layers. Also, Lin et al. [2021] separate auxiliary features into image- and sample-space and feed them into separate feature extractors to predict detail-preserved images.

Aside from the studies exploring the network structures, Kettunen et al. [2019] use a novel image gradient buffer produced by gradient-domain path tracing [Kettunen et al. 2015] to guide a reconstruction network. They demonstrated that the frequency information in image gradients helps the deep network infer image smoothness at the cost of producing the auxiliary inputs. Also, Chaitanya et al.

[2017] show that each G-buffer channel affects their network’s convergence to different extents.

Despite the large body of MC reconstruction research, most CNN-based denoisers are trained on a single task: *regression*. Although several reconstruction losses have been proposed [Chaitanya et al. 2017; Kettunen et al. 2019; Vogels et al. 2018], previous techniques essentially optimize the distance between output images and reference images, except a recent adversarial approach [Xu et al. 2019]. Nevertheless, when training deep networks with a single regression-based objective, complexity and sparsity of path-space degrade the utilization of path features. As such, we propose a novel task, *path-space manifold learning*, to offer additional guidance to CNN-based reconstruction models (Sec. 4).

2.2 Supervised Manifold Learning

Manifold learning, also known as metric learning, analyzes the similarity between data to remove redundant dimensions while preserving useful information. Since manifold learning can handle high-dimensional data cost-efficiently, it has been applied in various domains, such as image recognition [Hadsell et al. 2006] and image retrieval [Wang et al. 2017]. Its detailed discussions are available at a recent survey by Kaya and Bilge [2019]. We focus here on the most relevant supervised manifold learning techniques employing CNNs.

One of the intuitive applications of manifold learning is age estimation. Intuitively, even high-dimensional face images can be mapped onto a one-dimensional number line in order of age. Im et al. [2018] utilize the ordinal relation between ages, to provide valuable supervision to age estimation. They successfully observe a near-one-dimensional distribution in the embedded face image space.

Recently, a series of contrastive approaches has been proposed in the computer vision domain [Chen et al. 2020; Deng et al. 2019; Khosla et al. 2020]. These methods exploit contrastive losses, which manipulate the distance between embedding pairs according to their label similarities, to learn the affinity of inputs. These approaches induce useful embedding spaces and provide improved results when the embeddings are used as intermediate features for their target tasks (e.g., face recognition, image classification, and shape correspondence). Also, contrastive learning has been proven useful, especially when a dataset is sparse (i.e., a limited number of data for each label). For example, a popular face recognition benchmark LFW [Huang et al. 2007] contains 13,000 images, including photos of 1,680 different people.

Our work also designs a contrastive loss in addition to a regression loss to remedy the sparsity of path-space and extract more expressive path embeddings for improving MC reconstruction. Since we cannot clearly define path labels to distinguish between intra-class pairs and inter-class pairs, path-space contrastive learning creates new challenges, distinctive from typical image or point cloud processing. Hence, this paper adopts a weakly-supervised approach [Zhou 2018] to lessen the strict constraint on contrastive labeling (Sec. 4.3).

2.3 Manifold Techniques in Graphics

Due to the inherent complexity of path-space, efficient handling of light paths has been a central question in MC rendering. Researchers have exploited analytic constraints of path-space, reducing its dimensionality. *Manifold exploration*, proposed in the seminal paper of Jakob and Marschner [2012], is a path sampling method based on Fermat’s principle on specular surfaces. This approach scales down the dimensionality of path integration by analyzing the manifold of specular transport at the cost of computing the Jacobian.

Half vector space light transport (HSLT) [Kaplanyan et al. 2014] perturbs half vectors stochastically along a sequence of path vertices. Unlike manifold exploration, HSLT performs importance sampling on those half vectors from BSDFs and geometries, facilitating path-space exploration. Hanika et al. [2015b] present *breakup mutation strategy*, which further extends HSLT to displaced geometries.

Hanika et al. [2015a] apply the specular manifold analysis to complement next event estimation (NEE), combined with MC path tracing. In the spirit of manifold NEE, Zeltner et al. [2020] present a method for rendering specular microgeometry and complicated surfaces induced by normal and displacement mappings. Nonetheless, these techniques are not based on data-driven learning approaches.

Beside the MC rendering, various data-driven methods have been discussed, thanks to recent advances in machine learning. Font synthesis is one of the intuitive and valuable applications of manifold learning. Balashova et al. [2019] learn the manifold of typefaces by applying a machine learning-based manifold method. Zsolnai-Fehér et al. [2018] also leverage the same algorithm to explore the two-dimensional manifold of material appearances and synthesize unseen materials.

As far as we know, data-driven manifold techniques have not yet been studied for MC reconstruction, even though they have begun to be applied in other graphics sub-fields. Hence, we explore its potential by employing contrastive learning to MC reconstruction.

3 OVERVIEW

We now present the main intuition of path-space manifold learning for MC reconstruction. We then give an overview of our framework to train deep reconstruction models.

Path-space deep neural network. While the auxiliary features have proven valuable to high-quality MC reconstruction, high-dimensional data processing techniques are inevitable to efficiently utilize the information on light paths, another important piece of MC rendering. Due to the sparsity and complexity of paths, efficient training in path-space is difficult for deep neural networks. One brute force solution is to collect training samples as dense as possible. However, it often requires a large amount of data and long training time, exponentially growing with data dimensions for stable and reliable results.

Why manifold learning? The classic solutions of MC reconstruction, such as cross bilateral filtering [Eisemann and Durand 2004], non-local means filtering [Buades et al. 2005], ray histogram fusion [Delbracio et al. 2014], and adaptive regression [Moon et al. 2014], share the same philosophy; they define similarity metrics of pixels and take average colors of surrounding *similar* pixels. As

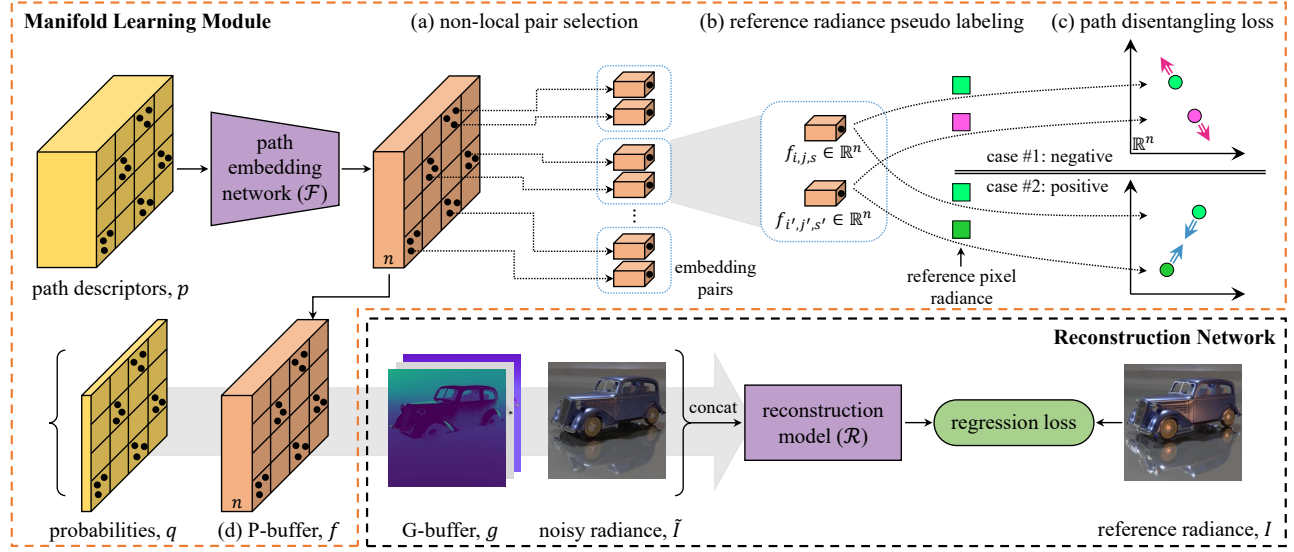


Fig. 3. Schematic of our joint manifold-regression training framework. The suggested manifold learning module (orange enclosure) is attached to an existing reconstruction network (black enclosure) to utilize path-space features better. The path embedding network in our framework is jointly optimized on two tasks: contrastive manifold learning and regression. (a) We pair not only two adjacent embeddings but also two distant embeddings in image space. (b) We label embeddings with the reference radiance at the pixel where each path is sampled. (c) Finally, path disentangling loss aims to adjust the distance between two embeddings according to the labels. (d) Meanwhile, the path embeddings (i.e., P-buffer) proceed to the reconstruction model along with auxiliary features and optimize the given regression loss.

an example of the metrics, non-local means filtering uses patch-wise mean luminance, and ray histogram fusion exploits Chi-Square distance between two radiance histograms. When extending the pioneering intuition to path-space, it is troublesome to define the affinity metric analytically due to the complexity of path-space. Some previous deep learning-based MC reconstruction methods [Gharbi et al. 2019; Lin et al. 2021] have tried to leverage of path-space information via image-space regression, but gains limited effects. Therefore, we propose a data-driven manifold learning method so that a reconstruction model can learn the affinity between paths and exploits the information in inferring its optimal parameters.

Manifold v.s. regression learning. The path-space contrastive learning aims to learn direct *sample-to-sample* correlation to discriminate overlapped path distributions (Fig. 2), which leads to more crisp embeddings. On the other hand, image-space regression learns correlation between input *pixels* and target *pixels*, and a sample-space model learns that between input *samples* and target *pixels*. These approaches may fail to reconstruct images in pathological cases where two pixels with similar color distributions converge to a different color, or pixels with different distributions converge to the same color.

Joint manifold-regression training. To achieve our goal, we present a joint manifold-regression training scheme for MC reconstruction networks. As shown in Fig. 3, we attach the proposed manifold learning module to a target MC reconstruction network. First, MC path tracing produces high-dimensional *path descriptors*, which represent the radiometric properties of individual paths. Then, we feed path descriptors into *path embedding network* serving the feature

extractor. The feature extractor learns the affinity between paths and a low-dimensional structure of path-space by our manifold loss, *path disentangling loss*, built on top of contrastive losses. Since a contrastive loss requires labels to distinguish inter-class paths, we use reference pixel radiance as our weak pseudo labels (Fig. 3 (b)). We call the low-dimensional output *P-buffer* as analogous to G-buffer. P-buffer is fed into the reconstruction network together with G-buffer. Finally, the feature extractor and the reconstruction model are trained simultaneously on the manifold loss and an ordinary regression loss (e.g., relative L2).

4 MANIFOLD LEARNING FOR MC RECONSTRUCTION

We propose a novel manifold learning framework to improve MC reconstruction models via weakly-supervised contrastive learning. The proposed framework can be adopted to both image- and sample-space deep learning-based reconstruction models. We illustrate our framework in Fig. 3.

4.1 Path Descriptor

Configuring the path descriptor is an essential prerequisite for effective path manifold learning. The proper path descriptor should provide enough information to the manifold learning module to distinguish each path, and the information should help image reconstruction as well.

The following information has been known as some useful path descriptors. First, as shown in Fig. 2, light transports on diffuse paths (orange) and caustic paths (green) should have vastly distinct radiance variances and intensities. Therefore, they need to

be processed separately throughout network layers. In fact, paths can be classified by material properties at each vertex, according to the Heckbert’s regular expression [Heckbert 1990]. Second, recent sample-based MC reconstruction methods also utilize some of the per-vertex material properties [Gharbi et al. 2019; Lin et al. 2021].

Referring to these studies, we hypothesize that the features, such as a bidirectional scattering distribution function (BSDF) at each path vertex and photon energy propagated through the path, play an important role in path classification and image noise removal as well. Consequently, we construct our path descriptor with five channels of per-vertex features representing a BSDF of each path vertex and six channels of per-path features representing each path’s lighting condition. Additionally, one channel of path sampling probability is also collected.

A radiance field at equilibrium can be defined by an integral equation, called rendering equation [Kajiya 1986]:

$$L_r(x, \omega_o) = \int_{\Omega} L(x, \omega) f_s(x, \omega_o, \omega) |\cos(\theta)| d\omega, \text{ and} \quad (1)$$

$$L_o(x, \omega_o) = L_e(x, \omega_o) + L_r(x, \omega_o), \quad (2)$$

where L_r, L, L_o , and L_e are respectively the reflected, incident, outgoing, and emitted radiances, f_s is the BSDF, ω_o and ω are respectively the outgoing and incident directions at a surface point x , and θ is the angle between the incident direction and the surface normal. Using MC estimation, Eq. 1 is approximated with a sampling density $q(\omega|x, \omega_o)$ (i.e., backward path tracing) as follows:

$$\bar{L}_r(x, \omega_o) = \frac{1}{N} \sum_{i=1}^N \frac{L(x, \omega_i) f_s(x, \omega_o, \omega_i) |\cos(\theta_i)|}{q(\omega_i|x, \omega_o)}. \quad (3)$$

Suppose we have a path that is bounced k times from the eye $x^{(-1)}$ to a point on a light $x^{(k)}$ and let $\bar{x} = x^{(-1)}x^{(0)} \dots x^{(k)}$ denotes such path. Let $L_e(x^{(l)}, \omega_o^{(l)})$ is the emitted radiance from $x^{(l)}$ to $x^{(l-1)}$, where $\omega_o^{(l)} = \omega_{x^{(l)} \rightarrow x^{(l-1)}}$ and $0 \leq l \leq k$. Then, our path descriptor is $(5k + 6)$ -dimensional and constructed as follows:

- three channels per vertex for the attenuation;
 $f_s(x^{(l)}, \omega_o^{(l)}, \omega^{(l)}) |\cos(\theta^{(l)})|, \forall 0 \leq l \leq k - 1$,
- one channel per vertex for the material-light interaction tag (reflection, transmission, diffuse, glossy, specular),
- one channel per vertex for the roughness parameter of BSDF,
- three channels per path for the radiance undivided by the sampling probability;
 $L_e(x^{(k)}, \omega_o^{(k)}) \prod_{0 \leq l \leq k-1} f_s(x^{(l)}, \omega_o^{(l)}, \omega^{(l)}) |\cos(\theta^{(l)})|$,
- three channels per path for the photon energy propagated through the path;
 $L_e(x^{(k)}, \omega_o^{(k)})$.

Besides the path descriptor, we collect a path sampling probability from the path, which is directly fed into a reconstruction network rather than the path embedding network:

- one channel per path for the path sampling probability,
 $\prod_{0 \leq l \leq k-1} q(\omega^{(l)}|x^{(l)}, \omega_o^{(l)})$.

In our framework, the reconstruction network uses the probability buffer to determine whether a path is an outlier. Note that we do not input the probability buffer to the manifold module since the it is

greatly influenced by the underlying importance sampling method, potentially hindering the learning of direct sample-to-sample correlations while training the path embedding network.

The number of path bounces is limited to six for fair comparisons with one of our baselines, SBMC, in Sec. 6.1. That is, we use $k = 6$ in constructing path descriptors, and the number of channels of path descriptors is 36. We put zero pads to handle paths shorter than six bounces as SBMC’s implementation.

4.2 Path Embedding Network

We adapt the sample-based feature extractor block proposed by Gharbi et al. [2019] to construct our path embedding network. The feature extractor block uses stacks of fully-connected layers and an UNet to embed path descriptor vectors, considering neighbor paths; the details can be seen in Fig. 13. Note that SBMC uses the structure to infer splatting kernel parameters for radiance samples, while we utilize it to infer path embeddings.

P-buffer, the output of the path embedding network, flows in two branches simultaneously afterward. One is to evaluate our contrastive loss described in the following section, and the other is to feed P-buffer into the reconstruction network with one channel of the sample variance of P-buffer and the path probability buffer (Fig. 3). Since P-buffer still has the sample dimension, we average it along the sample dimension and feed it to pixel-based reconstruction models. For sample-based models, we use P-buffer as it is.

4.3 Path Disentangling Loss and Weak Supervision

Our goal is to learn relevant embeddings from path descriptors. Though contrastive losses have been widely adopted to solve manifold learning problems, one limitation is that the losses require ground-truth labels that decide whether two data points are either a positive or negative pair. Unfortunately, it remains unclear how to define exact labels for light paths.

Hence, we unify our contrastive approach with weakly-supervised learning. As a pseudo label for a path sampled at a pixel, we use reference radiance at the pixel (Fig. 3). Suppose we have a pair of path descriptors, $\{x, y\}$, where $x, y \in \mathbb{R}^{36}$, and the reference radiances $\{I_x, I_y\}$, where $I_x, I_y \in \mathbb{R}^3$ at the pixels where respective paths are sampled. Let \mathcal{F} be the path embedding network in our manifold learning module (Fig. 3). $f_x = \mathcal{F}(x)$ and $f_y = \mathcal{F}(y)$ denote the respective path embeddings. Then, our path disentangling loss is defined as follows:

$$\mathcal{L}_m(f_x, f_y, I_x, I_y) = \left(\|f_x - f_y\|_2^2 - \|\tau(I_x) - \tau(I_y)\|_2^2 \right)^2, \quad (4)$$

where $\tau(I) = \left(\frac{I}{1+I} \right)^{\gamma}$ is the tone-mapping function [Reinhard et al. 2002], required to reduce the range of images and P-buffers and lead stable training [Bako et al. 2017; Gharbi et al. 2019].

The loss indicates the discrepancy between the path embedding space and the reference color space. Two paths similar in the reference color space attract each other in the path manifold, while repelling each other in the opposite case. Intuitively, two paths embedded as neighbors on the same manifold have a high-correlation on contributing to correlated pixels, i.e., pixels of similar colors once converged with high sample counts. Also, the strong correlation among the input of a reconstruction model and the reference image

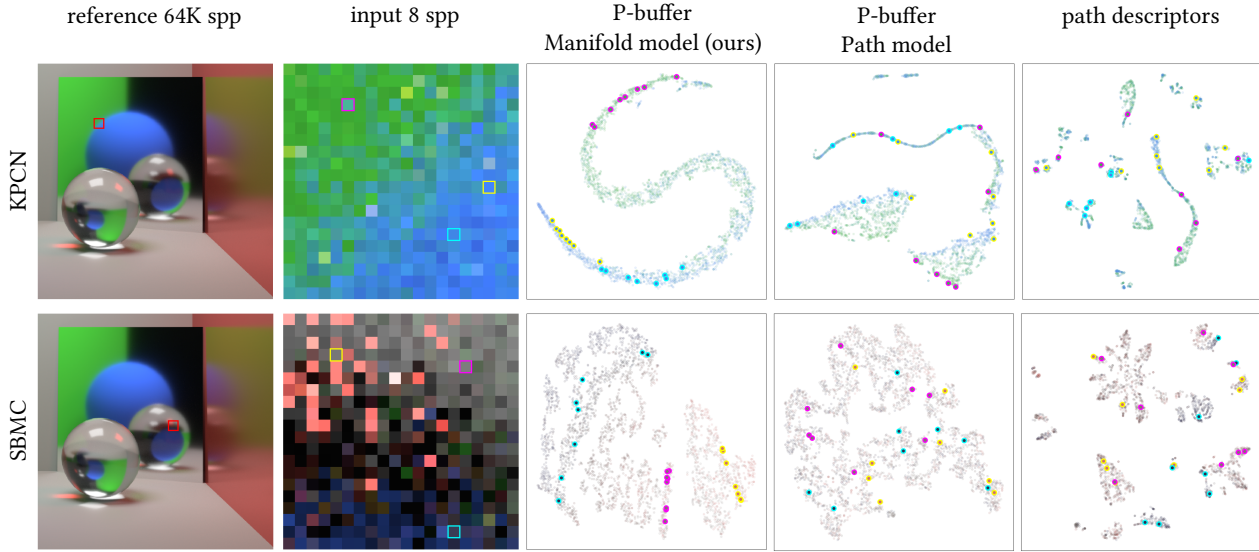


Fig. 4. Visualization of various spaces. Each row shows three different pixels enclosed by either yellow, cyan, or magenta boxes, followed by relevant path samples; the input image is rendered at 8 spp, thus each pixel consists of 8 paths. From the third to fifth columns, each plot presents a distribution of either P-buffers or raw path descriptors (i.e., inputs of the proposed framework), where the data points are color-coded by reference pixel colors. All samples are projected onto the 2D spaces using t-SNE dimensionality reduction method [Maaten and Hinton 2008]. In the first row, though the magenta pixel shows similar color with others, the samples are clearly separable in our manifold space (third column). In contrast, without explicit manifold supervision, three pixels' path distributions are highly tangled (fourth and fifth columns). In the second row, our method shows individual clusters caused by at least three different light-material interactions (i.e., red caustics, white floor, and a blue ball), which is not the case in the rest columns.

indeed improves MC reconstruction according to the recent discussions on *ideal features for denoising* [Back et al. 2018; Delbracio et al. 2014; Kettunen et al. 2019; Moon et al. 2013]. In Fig.4, we empirically discover that our weakly-supervised contrastive learning works well to induce discriminate and well-clustered P-buffer.

4.4 Joint Manifold-Regression Training

In Eq. 4, a pair selection scheme for x and y is important to achieve robust and fast training [Wu et al. 2017]. The network cannot learn meaningful weights if only easy pairs, whose path disentangling error is relatively smaller than other pairs, are selected. We propose non-local pair selection to alleviate this issue.

Given a large image of 1280×1280 , we extract 256 patches of size 128×128 in the training phase. We then construct mini-batches of 8 image patches. Path descriptors are located at the corresponding pixels at each sample count. Since we construct training batches from high-resolution images, paths belonging to different patches are likely to have considerable distinctions in path manifold and vice versa, just as their respective reference radiances. Thus, two paths from different patches, a non-local pair, have a high chance to supply a hard case to path disentangling loss, leading to robust contrastive training.

We construct a set of non-local pairs by randomly shuffling path descriptors within a batch and by comparing the original batch and shuffled one. We also construct a set of local pairs to enforce the balance between similar and dissimilar cases; we shuffle samples within patches and comparing the original and shuffled ones. Non-local pair selection offers not only embeddings at adjacent pixels

Algorithm 1 Joint Manifold-Regression Training Algorithm

notations

\tilde{I} and I	noisy input and reference image
g	auxiliary features
p and q	path descriptors and sampling probabilities
$\Theta_{\mathcal{F}}$	weights of the path embedding network
$\Theta_{\mathcal{R}}$	weights of the given reconstruction network
λ	manifold-regression balancing parameter

while total loss is decreasing **do**

$f = \mathcal{F}(p|\Theta_{\mathcal{F}})$ // path embedding

$f', I' = \text{SHUFFLEWITHINBATCH}(f, I)$ // non-local pairs

$f'', I'' = \text{SHUFFLEWITHINPATCH}(f, I)$ // local pairs

$\hat{I} = \mathcal{R}(\tilde{I}, g, f, q|\Theta_{\mathcal{R}})$ // image reconstruction

$\mathcal{L}_{total} = \lambda(\mathcal{L}_m(f, f', I, I') + \mathcal{L}_m(f, f'', I, I'')) + \mathcal{L}_r(\hat{I}, I)$

$\Theta_{\mathcal{F}}, \Theta_{\mathcal{R}} \leftarrow \text{ADAM}(\mathcal{L}_{total})$

end while

return $\Theta_{\mathcal{F}}, \Theta_{\mathcal{R}}$

but also embeddings far apart in image-space to our manifold loss, aiming to help contrastive learning to mitigate the sparsity of path samples.

The overall training algorithm is summarized in Alg. 1. Here, \mathcal{L}_r denotes an arbitrary regression loss that the reconstruction network aims to minimize. Note that we average P-buffer along the sample dimension for image-space models. Also, we find that $\lambda = 0.1$ yields empirically good results. See Table 4 for the impact of λ on reconstruction performance.

5 EXPERIMENTAL SETUP

5.1 Reconstruction and Training Details

Reconstruction models. We choose three reconstruction models to test our joint manifold-regression approach: KPCN [Bako et al. 2017], SBMC [Gharbi et al. 2019], and LBMC [Munkberg and Hasselgren 2020]. Three models employ different approaches to denoising: SBMC and LBMC are sample-based and KPCN is pixel-based. Since our manifold method utilizes the affinity among path *samples*, SBMC and LBMC are better suited for our approach. Nonetheless, we would like to analyze benefits of utilizing manifold learning and its outcome, P-buffer, even on the pixel-based approach, since image- and sample-space features can be used together [Lin et al. 2021].

For all models, we use the publicly available implementations that Gharbi et al. [2019] and Munkberg and Hasselgren [2020] provided^{1 2}. KPCN was initially written in Tensorflow [Abadi and et al. 2016]³, and then Gharbi et al. [2019] re-implemented it in PyTorch [Paszke et al. 2017]. Munkberg and Hasselgren [2020] is also written in the same framework. Our experiments are conducted in PyTorch environment as we exploit a building block of SBMC to implement our path embedding network.

Training details. We used regression losses adopted by KPCN, SBMC, and LBMC to train all models. The specular and diffuse branches of KPCN were trained to minimize L1 error between outputs and reference images in linear space. In fine-tuning of KPCN, the entire model was tuned using the same L1 metric [Bako et al. 2017]. SBMC was trained to minimize relative L2 error between tone-mapped outputs and references [Gharbi et al. 2019]. LBMC optimized *symmetric mean absolute percentage error* (SMAPE), which shows numerically stable results [Munkberg and Hasselgren 2020].

We emphasize that we trained all networks from scratch, i.e., training from randomly initialized weights with the same dataset. Therefore, we can exclude the quality of pretrained weights and focus on evaluating the impact of our manifold learning framework on different architectures. Also, we used the same initialization methods, optimizers, and schedulers as suggested in each paper. The convolutional layers of KPCN and LBMC were initialized using Xavier method [Glorot and Bengio 2010], and those of SBMC were initialized using He et al. [2015]’s approach. Both KPCN and LBMC were trained by ADAM optimizer [Kingma and Ba 2014] with a learning rate of 10^{-4} , and SBMC was also trained by the same optimizer with a learning rate of 5×10^{-4} . The batch size is 8. Note that we adjust the learning rates to use the same batch size for all models.

We choose relative L2 (RelL2), relative L1 (RelL1), and structural dissimilarity [Wang et al. 2004] ($DSSIM = 1 - SSIM$) as error metrics to evaluate all models. RelL2 is based on the mean-squared error.

5.2 Dataset

Data acquisition. We use three different datasets; training, validation, and test sets. The training and validation sets are rendered from 18 scenes and used in the model training phase. Especially, we



Fig. 5. Example reference images from our training set. Our scenes vary in light phenomena, including glossy reflections, rough transmissions, specular highlights, and color bleeding.

stop training when the average error on the validation set starts increasing. Our 12 test scenes are entirely separated from the training and validation scenes. Thus, the test set is unseen by models in training, and we can expect test errors show general performance in practice. We assembled all scenes from Blend Swap⁴ and a publicly available repository [Bitterli 2016].

For training and validation, we randomized camera parameters, material parameters, and environment maps to simulate diverse light transport phenomena in a restricted number of scenes. We rendered 26 different images of resolution 1280×1280 for each of 18 scenes. We randomly select one out of 26 images to build a hold-out validation set. In total, our training set consists of 450 images (Fig. 5), and validation set consists of 18 images. We trained all models on 2 to 8 spp inputs to make our method compatible with SBMC and LBMC; both are trained at the range due to I/O bottlenecks. Note that any spp inputs can be used at the test time thanks to their sample-based architectures.

We converted the rendering results to the format that each reconstruction model expects (e.g., image-space features for KPCN). The reference images are rendered at 8,000 spp for training and validation. Each rendering used a different random seed to avoid correlation between an input and a reference. We did not use the test set in the model exploration phase, preventing design choices and hyperparameters from being optimized only in the test set. The final model was selected based on the validation results. All datasets are rendered by OptiX engine [Parker et al. 2013].

Preprocessing. We used the original implementations for input preprocessing as well. Specially, we transformed the specular component of the linear radiance into the log-domain to prevent artifacts around high energy highlights. We decomposed the albedo from the diffuse color buffer for KPCN. For our path descriptors and path sampling probabilities, we used log transformations to handle the

¹<https://github.com/adobe/sbmc>

²<https://github.com/NVlabs/layerdenoise>

³http://civc.ucsb.edu/graphics/Papers/SIGGRAPH2017_KPCN

⁴<https://www.blendswap.com>

high-dynamic-ranges of BSDFs and probability density functions (PDF), respectively.

Data augmentation. We sampled patches on-demand from high-resolution images in disks. We sampled 256 patches of size 128×128 from each image. Therefore, models are exposed to different sets of patches at every training epoch, increasing their generalities. Also, we trained models on 2 to 8 spp inputs to achieve generality on different noise levels as Gharbi et al. [2019]; Munkberg and Hasselgren [2020].

To reduce the number of trivial patches during the patch sampling process, we utilize a hard patch mining strategy, inspired by Bako et al. [2017]; Gharbi et al. [2016]. Patches involving low color deviation, background, or lights are trivial to denoise. At a high level, our sampling strategy improves the learning efficiency by allowing reconstruction models learn more glossy surfaces, surfaces with shadows and textures, and surfaces with complex geometries.

In summary, we used 806,400 (i.e., $450 \times 256 \times 7$) on-demand patches with resolution 128×128 in a training epoch, and used 12,600 fixed patches in validation. We trained each model until its validation loss stopped improving. Training KPCN and its corresponding manifold model took 2 to 7 days (i.e. see 6.5 million patches) on a NVIDIA RTX Titan GPU, including the fine-tuning of diffuse and specular branches. Training SBMC and its corresponding manifold model took 10 days (i.e. 5 million patches) on a NVIDIA Quadro RTX 8000 GPU. It took 7 days (i.e. 8 million patches) to train LBMC and its corresponding manifold model using the same GPU.

6 RESULTS

Throughout this section, we evaluate our framework both numerically and visually. We also extensively analyze the effectiveness and benefits of the proposed path-space manifold learning and P-buffer.

We use the following terms throughout this section.

- *Vanilla* models denote the original implementations of KPCN, SBMC, and LBMC.
- *Manifold* models denote the models trained by our manifold learning framework as shown in Fig. 3.
- *Path* models denote the *ablated* Manifold models that only minimize \mathcal{L}_r instead of minimizing \mathcal{L}_m as well. They are used to demonstrate the effectiveness of manifold learning.

Note that we remove the original path features of SBMC when training SBMC-Manifold to remedy I/O bottlenecks. Thus, SBMC-Manifold exploits radiance, G-buffer, and our path descriptors. SBMC uses their original path features for fairness. Also, we respectively use 12, 3, and 6 channels of P-buffer (i.e., hyperparameter) for KPCN-, SBMC-, and LBMC-Manifold, which yields empirically good results. See Table 4 for the impact of P-buffer dimension on reconstruction performance.

6.1 Comparisons

We provide quantitative summaries and convergence comparisons in Fig. 6 and qualitative comparisons between Vanilla models and their manifold opponents in Fig. 7.

All Manifold models numerically outperform their Vanilla opponents consistently across all the tested sample counts up to 64 spp (Fig. 6). Impressively, KPCN-Manifold shows a convergence rate

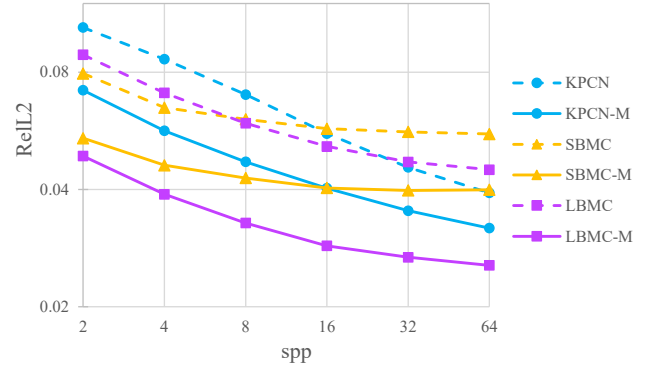


Fig. 6. Error comparisons between Vanilla models and their manifold opponents up to 64 sample counts on 12 test scenes. The errors, vary significantly across test scenes, are normalized to be relative to those of the noisy inputs of 2 spp and then averaged [Bako et al. 2017; Vogels et al. 2018]. Manifold models consistently outperform their Vanilla counterparts, while maintaining the convergence rates comparable to their opponents. Note that KPCN beats its latest successors with the help of our manifold framework.

comparable to that of KPCN while producing lower errors than any of Vanilla models, even at low sample counts.

The visual comparisons show that our framework improves on the drawbacks of KPCN, SBMC, and LBMC. We find that KPCN and two sample-based models show different characteristics. Sample-based models suppress high energy outliers thanks to its discriminative sample-space features and the kernel-splatting architecture, producing smoother images. In contrast, it consistently struggles to reconstruct high-frequency textures and geometries, as shown by the fourth and fifth rows in Fig. 7; this can be seen in the Balls scene in Gharbi et al. [2019] as well. On the other hand, KPCN reconstructs texture details well thanks to the dense image-space features. Still, the image-space statistics collapse modes of a radiance distribution and cannot discriminate noises and features, resulting in noticeable artifacts, as shown by the second row in Fig. 7. We observe that our method improves on these shortcomings of the three methods while preserving their strengths.

6.2 Discussions

In this section, we compare Path and Manifold models and analyze the effectiveness of manifold learning. We also study the impact of various design choices on reconstruction performance.

Importance of manifold learning. In Sec. 3, we presented our intuition behind manifold learning for handling high-dimensional path-space in MC reconstruction. Fig. 8 supports the claim that the utilization of path-space features is non-trivial and restricted without proper guidance. Also, Fig. 4, Fig. 14, and Fig. 15 show that manifold learning induces a fruitful embedding space, whereas the raw path space (i.e., path descriptors) and the embeddings learned solely by regression is highly unstructured. Thus, utilization of path-space features is limited in Path models.

The effectiveness of manifold supervision is more evident in the training process and test results. In Fig. 9, the KPCN-Path

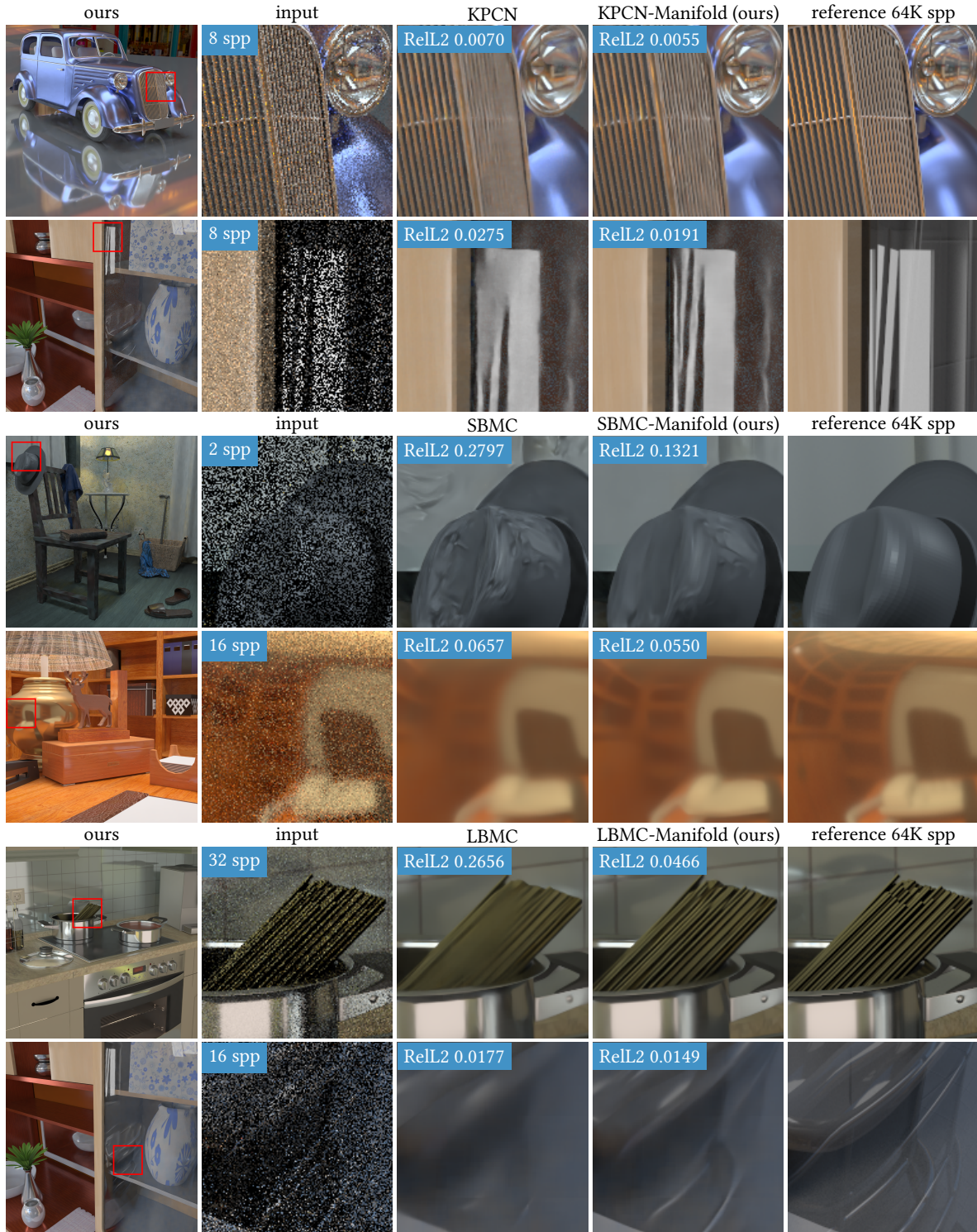


Fig. 7. Visual comparisons between Vanilla models and their manifold counterparts. It is challenging for both image-space and sample-space methods to reconstruct fine details caused by high-frequency textures or complex geometries, especially on reflective/refractive objects. Manifold models alleviate these issues by providing reconstruction networks with discriminative path cluster information as auxiliary inputs. Path-space contrastive learning, which uses dense reference labels, leverages rich sample features to distinguish fine details from noise while remedying the sparsity. "BATH" by Ndakasha under CC0. "Room Scene" by oldtimer under CC BY-SA 3.0. "Library-Home Office" by ThePefDispenser under CC BY 3.0.

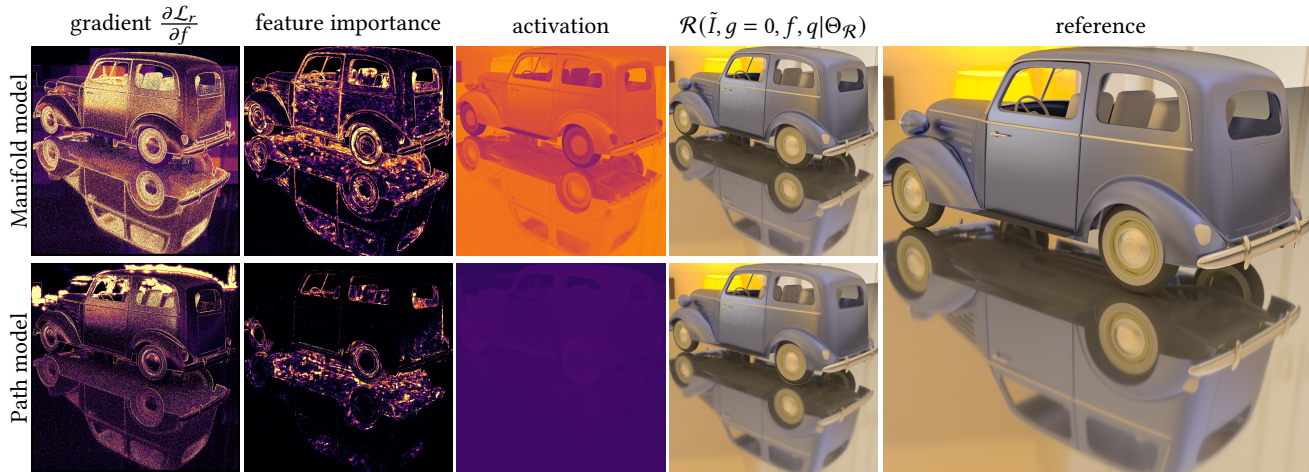


Fig. 8. Evidence that our manifold learning framework increases the utilization of path-space features in the MC reconstruction problem. The **gradient** illustrates the norm of the back-propagation signal with respect to the P-buffer during the first training epoch, representing the P-buffer’s contribution to the reduction of regression loss in the training stage. We obtain the second and third columns using learned model weights. The **feature importance** shows the importance of path descriptors in reducing MC noises in the inference stage by permuting the descriptor vectors in the training data and examining the rise in the error of the output [Breiman 2001]. The **activation** indicates the output of the first ReLU activation layer of KPCN; we obtain the activation map by zeroing out the noisy 8 spp input and G-buffer of KPCN, to visualize the P-buffer’s impact on kernel parameters exclusively. In the fourth column, we zero out the noisy image and G-buffer when predicting kernels of each model, then apply the kernels to produce the reconstructed images. That is, only path-space features are responsible for the kernel prediction. Surprisingly, KPCN-Manifold produces the sufficiently crisp image, unlike the Path model. These results imply that path descriptors hardly affect kernel prediction without manifold supervision. The manifold framework is shown to exploit high-dimensional paths successfully throughout all columns, while KPCN-Path, trained with a single regression loss, struggles to utilize path descriptors.

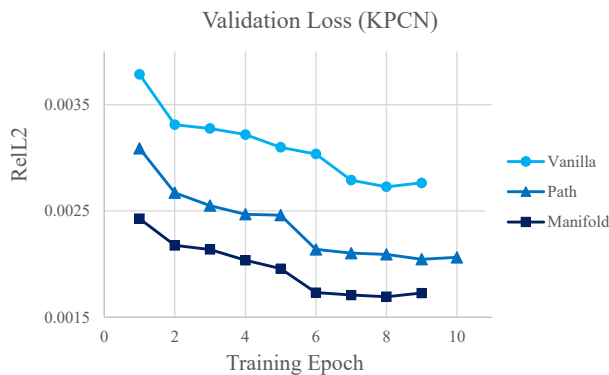


Fig. 9. Impact of manifold learning and path-space features on training convergence and the final validation quality. We stop training when the validation error starts increasing.

result shows that path descriptors and the path embedding network already help to improve KPCN to some extent. Furthermore, KPCN-Manifold improves this result in another large margin. KPCN-Manifold stabilizes the initial error, resulting in better convergence and the validation error on the last epoch.

Table 1 offers numerical comparisons among Vanilla, Path, and Manifold models in our test set. All metrics are relative to the noisy inputs. Since our framework provides more representative feature spaces to reconstruction models as shown in Fig. 4, we achieve substantial performance improvement across different error metrics (i.e., RelL2, RelL1, DSSIM). We also provide I.C. that measure a model’s

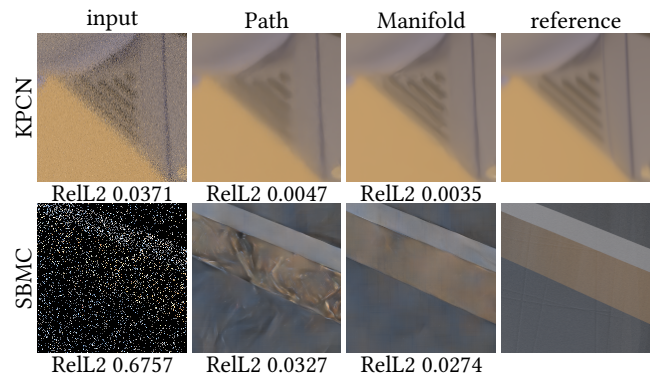


Fig. 10. Comparisons between Path models and Manifold models. The input of the first row uses 8 spp, and that of the second row uses 2 spp.

improvement consistency across whole test scenes, in addition to the mean error across the test scenes. I.C. denotes the number of cases where Path/Manifold model improves the error metric over its opponent Vanilla reconstruction model, divided by the total number of comparisons. A similar metric is also used by Xu et al. [2019]. Our framework boosts the improvement consistency in large margins compared to Path models. Finally, we offer visual comparisons between Manifold and Path models (Fig. 10). Path models show their limitations in suppressing artifacts and reconstructing crisp geometries.

Table 1. Ablation study on the proposed manifold learning framework. I.C. denotes the number of cases where a Path or Manifold model improves an error metric over its opponent Vanilla reconstruction model, divided by the total number of comparisons. I.C. indicates how consistently the performance improvement against the Vanilla model is observed, across different test scenes. Errors of different scenes are normalized to be relative to the noisy inputs for computing the average errors, as mentioned in Fig. 6. The same normalization is adopted for computing average errors across test scenes.

KPCN	RelL2 / I.C.	RelL1 / I.C.	DSSIM / I.C.
Vanilla	0.0869 / -	0.4090 / -	0.0579 / -
Path	0.0735 / 72.22%	0.3703 / 68.06%	0.0604 / 68.08%
Manifold	0.0585 / 93.06%	0.3084 / 93.06%	0.0500 / 94.44%
SBMC	RelL2 / I.C.	RelL1 / I.C.	DSSIM / I.C.
Vanilla	0.0682 / -	0.3656 / -	0.0538 / -
Path	0.0505 / 88.89%	0.2995 / 94.44%	0.0499 / 88.89%
Manifold	0.0477 / 94.44%	0.2926 / 98.61%	0.0459 / 97.22%
LBMC	RelL2 / I.C.	RelL1 / I.C.	DSSIM / I.C.
Vanilla	0.0729 / -	0.3441 / -	0.0489 / -
Path	0.0576 / 59.72%	0.2607 / 66.67%	0.0494 / 44.44%
Manifold	0.0401 / 83.33%	0.2102 / 90.28%	0.0458 / 90.28%

Table 2. Ablation study on joint manifold-regression training. The result shows that it is essential to provide two supervisions simultaneously for the performance of reconstruction models. Path embedding network of KPCN-Pre-training model is first optimized on path disentangling loss, and the network is attached to KPCN during the original KPCN training process afterward.

KPCN	RelL2 / I.C.	RelL1 / I.C.	DSSIM / I.C.
Path	0.0735 / 72.22%	0.3703 / 68.06%	0.0604 / 68.08%
Pre-training	0.0759 / 68.1%	0.3703 / 62.5%	0.0600 / 72.2%
Manifold	0.0585 / 93.06%	0.3084 / 93.06%	0.0500 / 94.44%

Alternative designs. We now verify benefits of the joint training scheme, which provides two different supervisory signals simultaneously to a reconstruction network. Although we demonstrated that path-space manifold learning provides MC reconstruction with fruitful feature spaces, path-space manifold learning and regression are conceptually different. Thus, pre-trained path embedding network might result in sub-optimal P-buffers without proper regularization of MC reconstruction. The result of Table 2 reinforces this concern. The table shows that the path embedding network, which is pre-trained by path disentangling loss (KPCN-Pre-training, second row) followed by the original KPCN training, provides performance improvements at most comparable to KPCN-Path.

Failure cases. Since path-space features provide more benefits in complex lighting conditions, Manifold models give rather limited improvements in simple scenes. The first row of Fig. 11 shows a sitting room, where the most materials are diffuse except glossy red chairs and metallic balls. In such configuration, we may not observe noticeable visual changes despite numerical improvements.

At extremely low sample counts, it is still challenging for sample-based methods as well as pixel-based methods. The second row of Fig. 11 shows a scene of bathroom cabinet shown in the third row of Fig. 7. A transmissive glass on the bathroom cabinet and

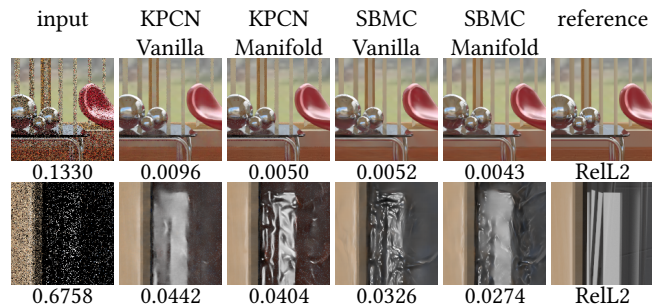


Fig. 11. Failure cases. The first row, where the input uses 4 spp, shows a simple scene where all methods reconstruct well. Our manifold learning framework gives visual improvements in a small margin in a simple scene. The second row, where the input uses 2 spp, shows that all models suffer from the severely under-sampled input, yielding noticeable artifacts for all models. "The Chillout Room" by Wig42 under CC BY 3.0.

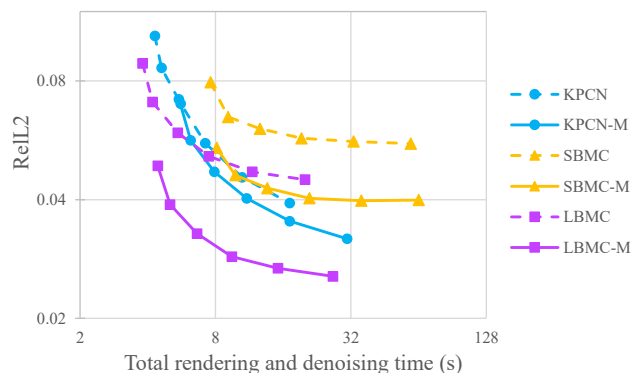


Fig. 12. Numerical convergences over time on 12 test scenes.

sunlight involves intensive fireflies. Since it is difficult to sample paths connected to the light source, especially with a small sample count, neither G-buffer nor path descriptors can adequately capture these light transport phenomena. Hence, our framework may not remedy this issue unless leveraging a different rendering algorithm, such as path guiding [Bako et al. 2019; Müller et al. 2017; Vorba et al. 2014].

Performance. Overall time-error comparisons are summarized in Fig. 12. We use a sample-based, path embedding network to process path descriptors. Hence, the performance of the embedding network scales linearly on the sample counts. A manifold module carries about 16% overhead compared to SBMC during inference (Table 3). However, SBMC requires more than four times sample counts to achieve the same error of SBMC-Manifold, according to Fig. 6. Similarly, KPCN requires more than four times sample counts to achieve the same error of KPCN-Manifold though the runtime cost of KPCN is constant. The overhead of manifold learning, however, exceeds its benefit for some simple cases where KPCN converges with very low sample counts. Note that we parallelly execute the specular and diffuse branches of KPCN models in a large GPU to reduce idle period.

Table 3. Runtime cost breakdown of the Vanilla models and our manifold learning module (in seconds), using test scenes of size 1280×1280 . Due to the sample-based path embedding network in our framework, the cost of our module increases linearly with the number of samples. RoC denotes the rate of change of runtime with respect to sample counts. Note that the cost of path tracing also increases linearly with the number of samples, and the benefit of embedding exceeds that of tracing more paths with an equal time budget.

spp	2	4	8	16	32	64	RoC
OptiX rendering	2.7	3	4	5.6	8.9	15.5	0.21
KPCN	1.6	1.6	1.6	1.6	1.6	1.6	0
SBMC	4.9	6.1	8.6	13.7	24	43.5	0.62
LBMC	1.09	1.2	1.43	1.86	2.74	4.5	0.05
path embed. net.	0.64	0.82	1.19	1.99	3.53	6.6	0.1

Nevertheless, a cost-efficient sample-based neural network would likely be achieved by image-sample hybrid approaches. For example, LBMC divides samples into mutually exclusive image-space buffers by sample binning. The further optimized LBMC achieves comparable results to SBMC at the cost of few tens of microseconds [Munkberg and Hasselgren 2020].

We also note that the implementation of a deep model greatly influences the inference performance. TensorRT can accelerate the inference of neural networks by orders of magnitude compared to PyTorch [Ulker et al. 2020], thanks to Tensor Cores, half-precision execution, and parallelism of CUDA multi-streams on NVIDIA GPUs. Thus, efficient implementation of sample-based models would be a promising future direction for production-ready performance.

Temporal extension. Sample-based methods can be applied to video reconstruction as described in Gharbi et al. [2019]. A path-space manifold learning framework needs to learn features invariant over time in paths to achieve temporal consistency. For example, the parameters of a moving camera would be useful to cluster similar path embeddings from different frames. Our pseudo labeling method can still be applied to this direction. Also, a self-supervised approach is recently proposed to extract features from video frames, considering visual invariants in different frames [Tschannen et al. 2020]. Similar approaches can improve the temporal stability of path-space contrastive learning and MC reconstruction.

Unbounded path length. In this work, we follow the known setting of existing path-space methods [Gharbi et al. 2019; Munkberg and Hasselgren 2020], which numerically and visually demonstrate that six bounces are sufficient to capture most visual effects within storage and memory limits. Handling infinite path length is still a huge open problem in the deep MC reconstruction domain. Perhaps, splitting a path into a finite number of sub-paths might be a solution. Sequential models, such as a recurrent neural network [Rumelhart et al. 1986], that map infinite-dimensional paths into a finite-dimensional feature space are also promising. Yet, such solutions introduce extra overheads and require orthogonal effort to balance the benefits. For conciseness, we focus on weakly-supervised contrastive learning and regard the infinite length problem as future work.

7 CONCLUSION

In this paper, we have proposed a novel manifold learning method that can be easily integrated into existing reconstruction models. We have also demonstrated the benefits of our weakly-supervised contrastive learning in path manifold across realistic test scenes.

Many interesting research directions lie ahead. Although we have shown benefits of path disentangling loss, a type of contrastive loss, numerous other forms of manifold losses have been explored in deep learning and computer vision [Kaya and Bilge 2019]. For example, it is well-known that a group of triplet losses shows better robustness than contrastive losses since it samples both intra-class and inter-class pairs and manipulates the distances simultaneously. Exploring different types of manifold losses would give us further improvement. In this paper, we have shown separate network structures between the path embedding network and reconstruction model. A shared model between them can result in a more compact and efficient model. Also, the layer-based approach [Munkberg and Hasselgren 2020] recently mitigates the overhead of sample-based networks using multiple image buffers rather than samples, which can also be helpful to optimize our path-space embedding network.

ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their valuable comments and insightful suggestions. We are also grateful to Prof. Bochang Moon, who read drafts of our paper and gave thorough and constructive feedback. Finally, thanks to our colleagues at SGVR Lab for their effort to revise drafts. Sung-Eui Yoon and Yuchi Huo are co-corresponding authors of the paper. This work was supported by the MSIT/NRF (No. 2019R1A2C3002833) and ITRC (IITP-2021-2020-0-01460).

REFERENCES

- Martin Abadi and et al. 2016. Tensorflow: Large-scale machine learning on heterogeneous distributed systems. *arXiv preprint arXiv:1603.04467* (2016).
- Jonghee Back, Sung-Eui Yoon, and Bochang Moon. 2018. Feature Generation for Adaptive Gradient-Domain Path Tracing. *Computer Graphics Forum* 37, 7 (2018), 65–74.
- Steve Bako, Mark Meyer, Tony DeRose, and Pradeep Sen. 2019. Offline deep importance sampling for Monte Carlo path tracing. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 527–542.
- Steve Bako, Thijs Vogels, Brian McWilliams, Mark Meyer, Jan Novák, Alex Harvill, Pradeep Sen, Tony DeRose, and Fabrice Rousselle. 2017. Kernel-predicting convolutional networks for denoising Monte Carlo renderings. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 97.
- Elena Balashova, Amit H. Bermano, Vladimir G. Kim, Stephen DiVerdi, Aaron Hertzmann, and Thomas Funkhouser. 2019. Learning A Stroke-Based Representation for Fonts. *Computer Graphics Forum* 38, 1 (2019), 429–442.
- Benedikt Bitterli. 2016. Rendering resources. <https://benedikt-bitterli.me/resources/>.
- Leo Breiman. 2001. Random forests. *Machine learning* 45, 1 (2001), 5–32.
- Antoni Buades, Bartomeu Coll, and Jean-Michel Morel. 2005. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation* 4, 2 (2005), 490–530.
- Chakravarty R Alla Chaitanya, Anton S Kaplanyan, Christoph Schied, Marco Salvi, Aaron Lefohn, Derek Nowrouzezahrai, and Timo Aila. 2017. Interactive reconstruction of Monte Carlo image sequences using a recurrent denoising autoencoder. *ACM Transactions on Graphics (TOG)* 36, 4 (2017), 98.
- Shixing Chen, Caojin Zhang, Ming Dong, Jialiang Le, and Mike Rao. 2017. Using Ranking-CNN for Age Estimation. In *CVPR*.
- Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. 2020. A simple framework for contrastive learning of visual representations. In *International conference on machine learning*. PMLR, 1597–1607.
- Mauricio Delbracio, Pablo Musé, Antoni Buades, Julien Chauvier, Nicholas Phelps, and Jean-Michel Morel. 2014. Boosting monte carlo rendering by ray histogram fusion. *ACM Transactions on Graphics (TOG)* 33, 1 (2014), 1–15.

- Jiakang Deng, Jia Guo, Niannan Xue, and Stefanos Zafeiriou. 2019. ArcFace: Additive Angular Margin Loss for Deep Face Recognition. In *CVPR*.
- Elmar Eisemann and Frédo Durand. 2004. Flash photography enhancement via intrinsic relighting. *ACM transactions on Graphics (TOG)* 23, 3 (2004), 673–678.
- Michaël Gharbi, Gaurav Chaurasia, Sylvain Paris, and Frédo Durand. 2016. Deep joint demosaicking and denoising. *ACM Transactions on Graphics (TOG)* 35, 6 (2016), 1–12.
- Michaël Gharbi, Tzu-Mao Li, Miika Aittala, Jaakko Lehtinen, and Frédo Durand. 2019. Sample-based Monte Carlo denoising using a kernel-splatting network. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.
- Xavier Glorot and Yoshua Bengio. 2010. Understanding the difficulty of training deep feedforward neural networks. In *Proceedings of the thirteenth international conference on artificial intelligence and statistics*. 249–256.
- Raia Hadsell, Sumit Chopra, and Yann LeCun. 2006. Dimensionality Reduction by Learning an Invariant Mapping. In *CVPR*.
- Johannes Hanika, Marc Droske, and Luca Fascione. 2015a. Manifold next event estimation. *Computer Graphics Forum* 34, 4 (2015), 87–97.
- Johannes Hanika, Anton Kaplanyan, and Carsten Dachsbacher. 2015b. Improved half vector space light transport. *Computer Graphics Forum* 34, 4 (2015), 65–74.
- Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *CVPR*.
- Paul S Heckbert. 1990. Adaptive radiosity textures for bidirectional ray tracing. In *Computer graphics and interactive techniques*.
- Gary B. Huang, Manu Ramesh, Tamara Berg, and Erik Learned-Miller. 2007. *Labeled Faces in the Wild: A Database for Studying Face Recognition in Unconstrained Environments*. Technical Report 07-49. University of Massachusetts, Amherst.
- Yuchi Huo, Rui Wang, Ruzahng Zheng, Hualin Xu, Hujun Bao, and Sung-Eui Yoon. 2020. Adaptive Incident Radiance Field Sampling and Reconstruction Using Deep Reinforcement Learning. *ACM Transactions on Graphics (TOG)* 39, 1 (2020), 1–17.
- Woobin Im, Sungeun Hong, Sung-Eui Yoon, and Hyun S Yang. 2018. Scale-Varying Triplet Ranking with Classification Loss for Facial Age Estimation. In *ACCV*. 247–259.
- Wenzel Jakob and Steve Marschner. 2012. Manifold exploration: a Markov Chain Monte Carlo technique for rendering scenes with difficult specular transport. *ACM Transactions on Graphics (TOG)* 31, 4 (2012), 1–13.
- James T Kajiya. 1986. The rendering equation. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*. 143–150.
- Nima Khademi Kalantari, Steve Bako, and Pradeep Sen. 2015. A machine learning approach for filtering Monte Carlo noise. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 122.
- Anton S Kaplanyan, Johannes Hanika, and Carsten Dachsbacher. 2014. The natural-constraint representation of the path space for efficient light transport simulation. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–13.
- Mahmut Kaya and Hasan Şakir Bilge. 2019. Deep metric learning: A survey. *Symmetry* 11, 9 (2019), 1066.
- Markus Kettunen, Erik Härkönen, and Jaakko Lehtinen. 2019. Deep convolutional reconstruction for gradient-domain rendering. *ACM Transactions on Graphics (TOG)* 38, 4 (2019), 1–12.
- Markus Kettunen, Marco Manzi, Miika Aittala, Jaakko Lehtinen, Frédo Durand, and Matthias Zwicker. 2015. Gradient-domain path tracing. *ACM Transactions on Graphics (TOG)* 34, 4 (2015), 1–13.
- Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. 2020. Supervised contrastive learning. *arXiv preprint arXiv:2004.11362* (2020).
- Diederik P Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980* (2014).
- Weiheng Lin, Beibei Wang, Jian Yang, Lu Wang, and Ling-Qi Yan. 2021. Path-based Monte Carlo Denoising Using a Three-Scale Neural Network. *Computer Graphics Forum* (2021).
- Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-SNE. *Journal of machine learning research* 9, Nov (2008), 2579–2605.
- Bochang Moon, Nathan Carr, and Sung-Eui Yoon. 2014. Adaptive rendering based on weighted local regression. *ACM Transactions on Graphics (TOG)* 33, 5 (2014), 1–14.
- Bochang Moon, Jong Yun Jun, JongHyeob Lee, Kunho Kim, Toshiya Hachisuka, and Sung-Eui Yoon. 2013. Robust Image Denoising Using a Virtual Flash Image for Monte Carlo Ray Tracing. *Computer Graphics Forum* 32, 1 (2013), 139–151.
- Bochang Moon, Steven McDonagh, Kenny Mitchell, and Markus Gross. 2016. Adaptive polynomial rendering. *ACM Transactions on Graphics (TOG)* 35, 4 (2016), 40.
- Thomas Müller, Markus Gross, and Jan Novák. 2017. Practical path guiding for efficient light-transport simulation. In *Computer Graphics Forum*, Vol. 36. Wiley Online Library, 91–100.
- Thomas Müller, Brian McWilliams, Fabrice Rousselle, Markus Gross, and Jan Novák. 2019. Neural importance sampling. *ACM Transactions on Graphics (TOG)* 38, 5 (2019), 1–19.
- Jacob Munkberg and Jon Hasselgren. 2020. Neural Denoising with Layer Embeddings. *Computer Graphics Forum* 39, 4 (2020), 1–12.
- Steven G. Parker, Heiko Friedrich, David Luebke, Keith Morley, James Bigler, Jared Hoberock, David McAllister, Austin Robison, Andreas Dietrich, Greg Humphreys, Morgan McGuire, and Martin Stich. 2013. GPU Ray Tracing. *Commun. ACM* 56, 5 (May 2013), 93–101. <https://doi.org/10.1145/2447976.2447997>
- Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. 2017. Automatic differentiation in pytorch. (2017).
- Erik Reinhard, Michael Stark, Peter Shirley, and James Ferwerda. 2002. Photographic tone reproduction for digital images. In *Proceedings of the 29th annual conference on Computer graphics and interactive techniques*. 267–276.
- Fabrice Rousselle, Marco Manzi, and Matthias Zwicker. 2013. Robust denoising using feature and color information. In *Computer Graphics Forum*, Vol. 32. Wiley Online Library, 121–130.
- David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature* 323, 6088 (1986), 533–536.
- Yi Sun, Yuheng Chen, Xiaogang Wang, and Xiaoou Tang. 2014. Deep Learning Face Representation by Joint Identification-Verification. In *Annual Conference on Neural Information Processing Systems 2014*. Montreal, Quebec, Canada, 1988–1996.
- Michael Tschann, Josip Djolonga, Marvin Ritter, Aravindh Mahendran, Neil Houlsby, Sylvain Gelly, and Mario Lucic. 2020. Self-supervised learning of video-induced visual invariances. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 13806–13815.
- Berk Ulker, Sander Stuijk, Henk Corporaal, and Rob Wijnhoven. 2020. Reviewing inference performance of state-of-the-art deep learning frameworks. In *Proceedings of the 23th International Workshop on Software and Compilers for Embedded Systems*. 48–53.
- Thijs Vogels, Fabrice Rousselle, Brian McWilliams, Gerhard Röhlin, Alex Harvill, David Adler, Mark Meyer, and Jan Novák. 2018. Denoising with kernel prediction and asymmetric loss functions. *ACM Transactions on Graphics (TOG)* 37, 4 (2018), 124.
- Jiří Vorba, Ondřej Karlík, Martin Šik, Tobias Ritschel, and Jaroslav Krivánek. 2014. On-line learning of parametric mixture models for light transport simulation. *ACM Transactions on Graphics (TOG)* 33, 4 (2014), 1–11.
- Jian Wang, Feng Zhou, Shilei Wen, Xiao Liu, and Yuanqing Lin. 2017. Deep Metric Learning with Angular Loss. In *IEEE International Conference on Computer Vision, ICCV 2017*. IEEE Computer Society, Venice, Italy, 2612–2620.
- Zhou Wang, Alan C Bovik, Hamid R Sheikh, and Eero P Simoncelli. 2004. Image quality assessment: from error visibility to structural similarity. *IEEE transactions on image processing* 13, 4 (2004), 600–612.
- Chao-Yuan Wu, R Manmatha, Alexander J Smola, and Philipp Krahenbuhl. 2017. Sampling matters in deep embedding learning. In *Proceedings of the IEEE International Conference on Computer Vision*. 2840–2848.
- Zhirong Wu, Yuanjun Xiong, Stella X Yu, and Dahua Lin. 2018. Unsupervised feature learning via non-parametric instance discrimination. In *CVPR*.
- Bing Xu, Junfei Zhang, Rui Wang, Kun Xu, Yong-Liang Yang, Chuan Li, and Rui Tang. 2019. Adversarial Monte Carlo denoising with conditioned auxiliary feature modulation. *ACM Transactions on Graphics (TOG)* 38, 6 (2019), 224–1.
- Tizian Zeltner, Iliyan Georgiev, and Wenzel Jakob. 2020. Specular manifold sampling for rendering high-frequency caustics and glints. *ACM Transactions on Graphics (TOG)* 39, 4 (2020), 149–1.
- Quan Zheng and Matthias Zwicker. 2019. Learning to importance sample in primary sample space. In *Computer Graphics Forum*, Vol. 38. Wiley Online Library, 169–179.
- Zhi-Hua Zhou. 2018. A brief introduction to weakly supervised learning. *National science review* 5, 1 (2018), 44–53.
- Henning Zimmer, Fabrice Rousselle, Wenzel Jakob, Oliver Wang, David Adler, Wojciech Jarosz, Olga Sorkine-Hornung, and Alexander Sorkine-Hornung. 2015. Path-space motion estimation and decomposition for robust animation filtering. In *Computer Graphics Forum*, Vol. 34. Wiley Online Library, 131–142.
- Károly Zsolnai-Fehér, Peter Wonka, and Michael Wimmer. 2018. Gaussian Material Synthesis. *ACM Transactions on Graphics (TOG)* 37, 4, Article 76 (July 2018), 14 pages.

A PATH EMBEDDING NETWORK

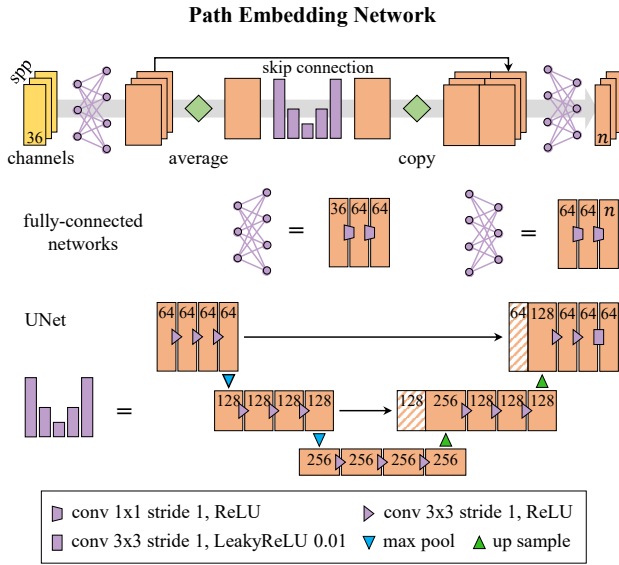


Fig. 13. Architecture of our sample-based path embedding network that adapts the sample-based feature extractor block. FC layers, implemented by 1-by-1 convolution, process a variable number of samples per pixel. We generate per-sample embeddings of path descriptor vectors using the first FC layer. Then, we average them along the sample axis to leverage spatial correlation of path embeddings by the image-space UNet. The UNet efficiently extracts spatially correlated features from neighbor path embeddings by 3-by-3 convolution, pooling, and skip-connections. Finally, we repeat the output of the UNet along the sample dimension and proceed it to another FC layer to produce n -dimensional P-buffer.

B HYPERPARAMETERS

Table 4. Impact of hyperparameters on validation errors. We select the final hyperparameters based solely on validation results but not test results to prevent models from being optimized in the test set that should be unseen by models. We empirically found that the performance strikes a balance when the scales of manifold loss and regression loss are roughly the same ($\lambda = 0.1$). Compared to the balancing parameter, the number of channels of path embeddings does not significantly affect validation errors.

balancing parameter (λ)	0.01	0.1	0.5
RelL2 ($\times 10^{-3}$)	1.837	1.693	1.731
# of channels of P-buffer	3	6	12
RelL2 ($\times 10^{-3}$)	1.693	1.681	1.644

C P-BUFFER VISUALIZATION

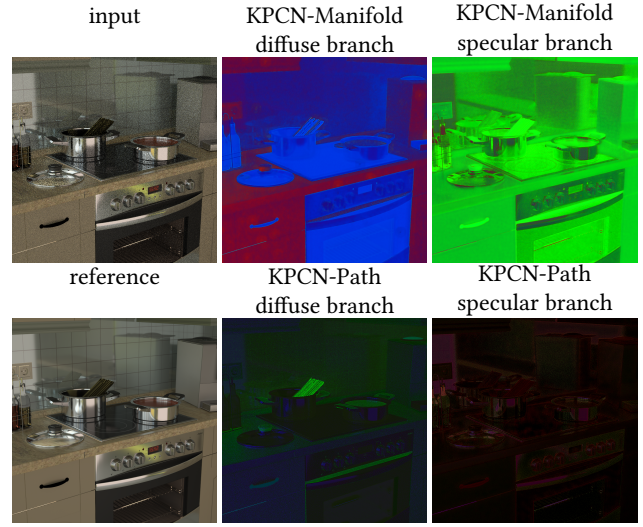


Fig. 14. Visualization of P-buffers of Manifold and Path models. We train KPCN-Manifold and -Path models with three channels of P-buffers. We train two independent path embedding networks for diffuse and specular branches of the KPCN framework, respectively, and visualize both diffuse and specular P-buffers for each KPCN model. We clamped the P-buffers into the range $[0, 1]$ to plot them in image space. The results demonstrate that P-buffers of the Manifold model capture complex geometries and reflections in the scene. Also, the P-buffers are noticeably cleaner than the noisy input and P-buffers of KPCN-Path. Best seen in zoom.

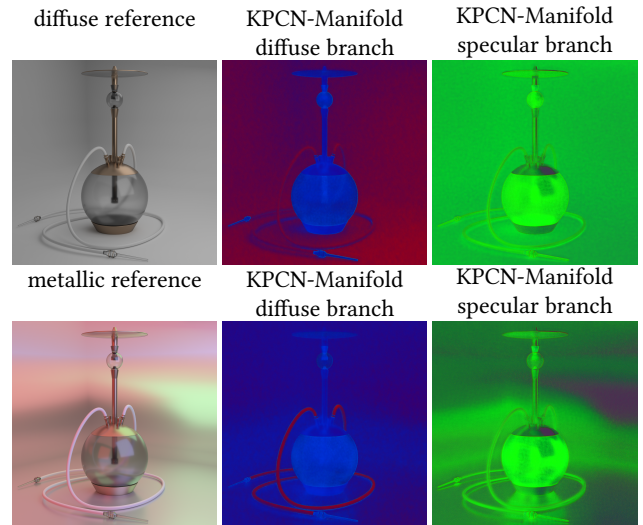


Fig. 15. Visualization of P-buffers in the same scene with different background materials. Comparing the two diffuse P-buffers, it can be seen that P-buffer is greatly affected by surface materials, intuitively. Also, we can observe that P-buffer captures light propagation effects, such as colorful lights reflecting off the wall or hookah reflecting off the floor, by the specular P-buffers. "Modern Hookah" by kexsz under CC BY 3.0.