

Tone-Mapped Mean-shift Based Environment Map Sampling

Wei Feng, *Member, IEEE*, Ying Yang, Liang Wan[†], *Member, IEEE*, Changguo Yu

Abstract—In this paper, we present a novel approach for environment map sampling, which is an effective and pragmatic technique to reduce the computational cost of realistic rendering and get plausible rendering images. The proposed approach exploits the advantage of adaptive mean-shift image clustering with aid of tone-mapping, yielding oversegmented strata that have uniform intensities and capture shapes of light regions. The resulted strata, however, have unbalanced importance metric values for rendering, and the strata number is not user-controlled. To handle these issues, we develop an adaptive split-and-merge scheme that refines the strata and obtains a better balanced strata distribution. Compared to the state-of-the-art methods, our approach achieves comparable and even better rendering quality in terms of SSIM, RMSE and HDRVDP2 image quality metrics. Experimental results further show that our approach is more robust to the variation of viewpoint, environment rotation, and sample number.

Index Terms—Environment map sampling, adaptive mean-shift clustering, tone-mapping, adaptive split-and-merge

1 INTRODUCTION

THE rendering of 3D virtual scenes can have the rendering realism significantly enhanced by applying real-world illumination, i.e. taking HDR environment maps as lighting sources [1], yet it faces the challenge of huge computational burden since one environment map can have thousands of directional light sources corresponding to image pixels. As an effective technique to address this challenge, environment map sampling reduces the computational scale greatly by means of approximating environment maps with a finite number of directional lights. Consequently, the practical computing time of 3D scene rendering as well as the storage requirement can be largely reduced. Since environment map sampling for images is taken in a preprocessing stage, we may rely on sophisticated algorithms to get high-quality light samples.

In the literature, different environment map sampling methods have been developed [2] [3] [4] [5]. A stream of existing methods rely on regular decomposition of environment maps into rectangular strata with similar importance metric values, such as spherical q^2 -tree [3] and median cut [6]. Since light regions may have arbitrary irregular shapes, approximating them with a set of rectangular segments may lead to a waste of light samples near boundary regions of strong lights. One typical example using irregular decomposition is the structured importance sampling [2], which thresholds an environment map into several levels, and evenly splits each level into small irregular strata.

This thresholding-based scheme does not account for the non-uniform intensity distribution of environment maps. Warping-based sampling techniques [4] [7] were reported to cast more samples in more important regions.

In this paper, we propose a novel environment sampling method, aiming at better capturing strong light regions' shapes as well as considering the non-uniform range of high-dynamic-range pixel intensities. Our method takes advantage of mean-shift image clustering, which is able to over-segment an image into small uniform regions that respect object boundaries. We then explore the impact of high dynamic range on the clustering, and achieve adaptive mean-shift clustering by adopting tone mapping techniques. The irregular shapes of light regions can be respected in clustered strata. Since the resulted strata have unbalanced importance metric values for the rendering quality and the strata number is not user-controlled, we develop an adaptive split-and-merge scheme to obtain a strata distribution with better balanced importance metric values for a given sample number. The light samples are finally generated by casting one directional light in each stratum respectively.

The effectiveness of the proposed method is validated through the comparison with several state-of-the-art methods. Experiments show that our method achieves comparable and even better rendering quality in terms of three typical image quality metrics, namely SSIM, HDRVDP2, and RMSE. In addition, our method is more robust to the variation of viewpoint, environment orientation, and sample number, as demonstrated in the extensive experiments.

• W. Feng and Y. Yang are with the school of Computer Science & Technology, Tianjin University; L. Wan (corresponding author, E-mail: lwan@tju.edu.cn) and C. Yu are with the school of Computer Software, Tianjin University, P. R. China. L. Wan is also with Tianjin Key Lab for Advanced Signal Processing, Civil Aviation University of China.

2 RELATED WORK

2.1 Environment Map Sampling

The illumination computation for one point on a 3D object is an integration of visibility, incident lighting and the

BRDF function of the object's surface defined as

$$I(x, \vec{s}) = \int_{\Omega} L_{in}(\vec{\omega}) \rho(x, \vec{\omega}, \vec{s}) v(x, \vec{\omega}) (\vec{\omega} \cdot \vec{n}) d\vec{\omega}, \quad (1)$$

where L_{in} , $\vec{\omega}$, \vec{s} and \vec{n} are incident radiance, incident (integration) direction, viewing direction and surface normal respectively; ρ refers to surface BRDF and v is the visibility. For a high-resolution environment map that contains thousands of directional lights, one of which corresponds to a pixel, the brute force computation is quite time consuming. Environment map sampling [9] [10] [11] [12] simplifies the computation by approximating an environment map with a limited number of directional light sources, with more lights distributed in more important image regions.

A major category of existing methods basically decompose an environment map into small strata, with one directional light casted in one stratum. The decomposition usually takes place more densely in more important regions (e.g. highlights), and less in other regions (e.g. dark portions). Some works adopt regular decomposition, i.e. decomposing an environment map into non-overlapped rectangular regions [6] [3] [5], or overlapped rectangular regions [13]. One typical example for irregular decomposition is proposed in [2], which segments an environment map into several levels by simple thresholding and stratifies each level into strata of a different number of samples. However, the simple thresholding may not sufficiently account for the non-uniform intensity distribution of environment maps. Our method, on the other hand, over-segments an environment map by adaptive mean-shift clustering, with larger kernel sizes for higher intensities.

There exist methods based on warping algorithms. For example, hierarchical warping algorithms are proposed to warp a random point set to match the distribution of environment lighting in the wavelet domain [4] [14], or using spherical harmonics [7]. The work in [15] generates uniformly distributed samples on the hemisphere, and warps these samples by using the hemispherical environment map as a PDF function. In addition, Ostromoukhov et al. created the sampling pattern by constructing a Penrose tiling over the environment map [11].

2.2 Mean-shift Clustering

Mean-shift clustering is a popular non-parametric feature clustering technique, and has been widely used in a variety of applications, such as image segmentation [16], noise removal [17], object tracking [18], etc. It was pioneered by Fukunaga and Hostetler [19], and later expanded to make it converge rapidly and applicable to high-dimensional feature space or large data set [20] [21] [22].

Roughly speaking, mean-shift clustering automatically estimates the modes (maximum) of the multivariate distribution underlying feature space. In an iterative process, the modes are computed as successive averages of data points weighted by a Parzen window kernel function, which is centered at each feature point. The obtained results are largely affected by the kernel function as well as the kernel

bandwidth or window size assigned. One classical choice for the kernel function is a Gaussian function [21]. The kernel bandwidth, on the other hand, can be a fixed value or adaptively computed at each feature point, which has been shown to produce better results at the cost of more computations [22] [23] [24] [25].

The most expensive operation in mean-shift clustering is to find nearest neighbors in feature space. To increase the speed performance, acceleration techniques have been proposed. As an example, Yang et al. [26] used the fast Gauss transform to speed up the summation in each iteration. Paris and Durand [16] interpreted mean shift as a topological decomposition of feature space, based on which most pixels are classified without iteration. Freedman and Kisilev [27] simplified the kernel density estimate based on random sampling.

3 OUR ALGORITHM

Inspired by previous works, our goal is to divide an environment map into strata of close importance metric values, with an emphasis on respecting irregular shapes of light regions as well as non-uniform distribution of intensities. Intuitively speaking, an arbitrary image region may be approximated by one irregular-shaped segment; however, it has to be approximated by a set of rectangular-shaped segments. Therefore, using irregular-shaped segments may help us to assign less samples in less important regions, and hence allocate more samples in more important regions.

Following this idea, we first over-segment the environment map into small regions with uniform intensities via adaptive mean-shift clustering, and next construct strata with balanced importance metric values via an adaptive split-and-merge scheme. In the following, we start by introducing the classical mean-shift clustering on environment maps.

3.1 Classical Mean-shift Clustering on Environment Maps

To apply mean-shift clustering, each pixel in an environment map (stored in the 2D longitude-latitude format) is represented as a 5D feature point $\mathbf{x}_i = \{x_i, y_i, r_i, g_i, b_i\}$ in the joint spatial-range domain [21]. Here, x_i and y_i are spatial coordinates of one pixel, and $\{r_i, g_i, b_i\}$ are its color range components. Denote one initial seed point to be \mathbf{y}_0 , the seed point corresponding to the mode's center is updated according to

$$\mathbf{y}_{j+1} = \frac{\sum_{i=1}^n K(\mathbf{y}_j - \mathbf{x}_i) \mathbf{x}_i}{\sum_{i=1}^n K(\mathbf{y}_j - \mathbf{x}_i)}, \quad (2)$$

where n is the amount of feature points, and $K(\cdot)$ is the kernel function depicting local feature distribution. Considering the spatial domain and the color range domain, the kernel function $K(\cdot)$ is defined as the product of two radially symmetric kernels, given by

$$K(h_s, h_r, \mathbf{x}) = \frac{C}{h_s^2 h_r^2} k\left(\left\|\frac{\mathbf{x}^s}{h_s}\right\|^2\right) k\left(\left\|\frac{\mathbf{x}^r}{h_r}\right\|^2\right), \quad (3)$$

where \mathbf{x}^s and \mathbf{x}^r are the spatial and color range parts of the feature vector, respectively. The function $k(\cdot)$ is selected as the Gaussian function, and C is the normalization factor. The parameters h_s and h_r are the bandwidths of the spatial and range kernels.

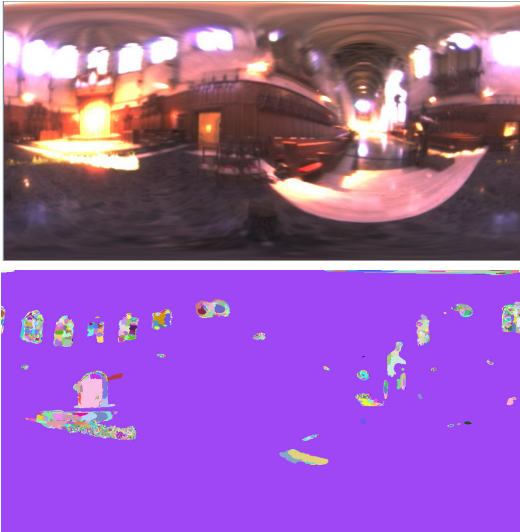


Fig. 1. Classical mean-shift clustering on an environment map. The low-intensity image portions are grouped into one purple segment.

Although being successful for LDR 2D images, the classical mean-shift clustering may cause low-quality segmentation results when applied for HDR environment maps. For the example shown in Figure 1, the visible image portions have intensities ranging between 0 and 1, while the highlights have quite large intensity values, e.g. above 100. Then, using a small bandwidth for the range kernel will generate numerous small segments; on the other hand, using a large bandwidth may create very big segments ($(h_s, h_r) = (7, 6)$ for Figure 1), which not only raises complexity for the succedent processing, but also degrades the rendering quality.

3.2 Adaptive Mean-shift Clustering by Tone-Mapping

To solve the problem mentioned above, we consider adaptive mean-shift clustering. Intuitively, we want to define a larger bandwidth \hat{h}_r in the high-intensity range, and a smaller bandwidth in the low-intensity range. The existing adaptive mean-shift clustering techniques usually determine the kernel bandwidth according to the k -nearest neighbors [22] [23] [24] [25], which have both small spatial distances and high color similarities. For instance, Georgescu et al. computed the adaptive bandwidth as the distance between the feature point and its k -nearest neighbor [22]. However, it is not an easy task to search for k -nearest neighbors, especially in a large dataset.

In our work, we adopt a much simpler scheme to achieve adaptive mean-shift clustering. We think that using adaptive bandwidth is equivalent to adjusting intensity values of the

environment map. Following this idea, we compress the intensity range adaptively, with more compression on the high intensity range. After non-linear compression, we still adopt a fixed bandwidth, which eventually corresponds to a varying bandwidth in the original intensity range.

Here, we perform the non-linear range compression by employing a global tone-mapping operation [28], which is quite fast and has decent performance. For each color value I , it is compressed adaptively according to the Weber-Fechner law and Naka-Rushton law, which is formulated as,

$$\hat{I} = \begin{cases} \eta \log(I + m) + s_0, & I \leq I_M, \\ \frac{I^n}{I^n + I_s^n}, & I > I_M, \end{cases} \quad (4)$$

where I_s is the image's semisaturation, computed from the median and mean of the radiance values (see the detailed definition in [28]); $I_M = 10^2 I_s$, $m = 10^{-1.2} I_s$, s_0 is computed to keep the above piece-wise function continuous; $n = 0.74$. The coefficient η is different for three channels: $\eta = 100/1.85$ for red and green; $\eta = 100/8.7$ for blue [28]. After Eq. (4), the compressed intensity values are finally normalized to $[0,1]$.

Figure 2 plots the tone mapping function for red and green channels. Obviously, when we apply a fixed kernel bandwidth in the compressed range domain, the actual bandwidth becomes smaller in visible light portions, and becomes larger in highlight portions. The adaptively segmented results are shown in Figure 3.

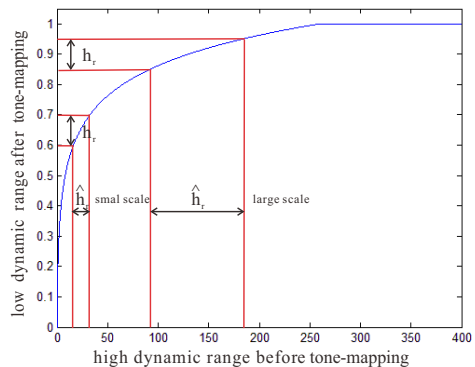


Fig. 2. Tone-mapping function for red and green channels. The same kernel bandwidth in the compressed range domain corresponds to a varying bandwidth in the original range domain.

3.3 Strata Construction via Adaptive Split-and-Merge

After the adaptive mean-shift clustering, the irregular shape of light regions is well maintained in the clustering segments, each of which has nearly uniform intensity. However, it is noted that the number of segments is not user-controlled. What's more, some visible light (highlight) regions may occupy large (or relatively large) area, leading to high importance metric values, and thus they will need more light samples. To handle these issues, we refine the



Fig. 3. Our adaptive mean-shift clustering on the environment map generates more reasonable segments for the input image in Fig. 1.

strata by using an adaptive split-and-merge scheme. Our basic idea is to first split the regions with high importance metric values and then combine adjacent regions to form a given number of strata.

For the importance measure of light regions, we adopt the metric proposed in [2], which unifies area-based stratification sampling and illumination-based importance sampling. It is a combination of light intensity and region area, given by,

$$\Gamma(L, \Delta\omega) = L^a \Delta\omega^b, \quad (5)$$

where $\Delta\omega$ is the solid angle of light region, and L the integrated illumination (i.e. pixel intensities) of light region. The parameters are set as $a = 1, b = 1/4$ according to a visibility-based variance analysis [2]. Note that L is computed using the original pixel intensity.

3.3.1 Adaptive Split Scheme

We detect those regions whose energy is much larger than the average importance Γ_a , i.e.

$$\{k \mid \Gamma(L_k, \Delta\omega_k) > \alpha \cdot \Gamma_a\}, \quad (6)$$

where k is region index, and parameter α is empirically set as $\alpha = 2$. The average importance metric Γ_a of all the segments is given by

$$\Gamma_a = \frac{\sum_{i=1}^N \Gamma(L_i, \Delta\omega_i)}{N}, \quad (7)$$

where N is the segment amount. For k -th region, we then set the number of newly generated small segments as:

$$N_k = \min\left(\left\lceil \frac{\Gamma(L_k, \Delta\omega_k)}{\Gamma_a} \right\rceil, M_k\right), \quad (8)$$

where M_k is the amount of pixels in k -th region. As the pixels in one region have similar intensities, we split the region evenly with Hochbaum-Shmoys' algorithm [29]. This detection and splitting processes iterate until the average importance metric Γ_a changes only slightly (Algorithm 1).

In practice, we found that the mean-shift clustering generated tiny fragments in the transition regions where intensity changes greatly, and their existence largely affects the splitting performance. This is because the inclusion of those fragments decreases the average importance significantly. To attenuate their effects, we perform a pruning operation that detects tiny fragments according to their sizes

Algorithm 1 Adaptive Splitting Scheme

```

1: Input: Region set  $R$  after the clustering.
2: do
3:   compute  $\Gamma_a$  of all regions
4:   determine the split region set  $\{k\}$ 
5:   for  $i \in \{k\}$ 
6:     split region  $i$ 
7:     update  $R$ 
8:   compute  $\Gamma_{an}$  of all regions
9:   while  $|\Gamma_a - \Gamma_{an}| > \varepsilon$ 
10: return  $R$ 

```

Algorithm 2 Adaptive Merging Scheme

```

1: Input: Region set  $R$ , adjacency matrix  $\mathcal{A}$ .
2: do
3:   find region  $i^*$  due to Eqn (8)
4:   Determine neighbor set  $N_{i^*}$  from  $\mathcal{A}$ 
5:   for  $j \in N_{i^*}$ 
6:     find neighbor  $j^*$  due to Eqn (9)
7:     merge region  $i^*$  and region  $j^*$ 
8:     update  $R$  and  $\mathcal{A}$ 
9:   while  $|R| > N_{sample}$ 
10: return  $R$ 

```

(with their solid angles less than a threshold ζ), and merge them into their nearby regions. In our experiments, we set the threshold $\zeta = 0.0004$. The segments after the adaptive splitting are shown in Figure 5(a).

3.3.2 Adaptive Merging Scheme

Up to now, we have decomposed the environment map into a set of disjoint regions with balanced importance metric values. The merging stage hierarchically combines the neighboring regions while maintaining importance balance. To be more specific, we always select one region with the minimum importance metric value, i.e.

$$i^* = \arg \min_i \Gamma(L_i, \Delta\omega_i), \quad (9)$$

where i^* is the index of the selected region. We then choose one of its neighbors to merge so that the importance metric after region combination is the minimum among all possible choices, as given by,

$$j^* = \arg \min_{j \in \eta(i^*)} \Gamma(L_{i^*} + L_j, \Delta\omega_{i^*} + \Delta\omega_j), \quad (10)$$

where $\eta(i^*)$ denotes the neighbors of region i^* . This merging process runs iteratively until the user-set sample number is reached (Algorithm 2).

To accomplish the merging process rapidly, we construct an adjacency matrix \mathcal{A} (illustrated in Figure 4(a)) to store the neighborhood information and the merging status of each region. A diagonal element $\mathcal{A}(i, i)$ denotes whether region i is merged into another region. The non-diagonal elements indicate if two regions are adjacent or not. All diagonal elements are initialized as -1 , and $\mathcal{A}(j^*, j^*) = i^*$ when region j^* is merged into region i^* . Non-diagonal

elements $\mathcal{A}(i, j)$ are assigned as 1, if region i and j are adjacent, otherwise as 0. We can see that the adjacency matrix is a diagonally symmetric matrix. With the adjacency matrix, we can rapidly find region i^* and determine the to-be-merged region j^* . Then we need to update the adjacency matrix. Besides setting $\mathcal{A}(j^*, j^*) = i^*$, we transfer the neighborhood information of region j^* to region i^* . In detail, if $\mathcal{A}(j^*, q) = 1$, where $q \neq i^*$ and $q \neq j^*$, we will set $\mathcal{A}(i^*, q) = 1$. A similar treatment is taken for i^* -th column. In the example shown in Figure 4(b), region 5 is merged into region 2 (shown in light blue).

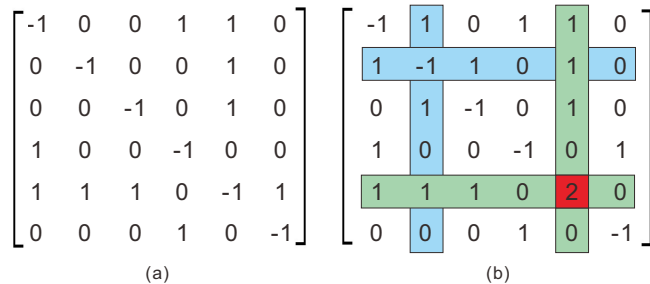
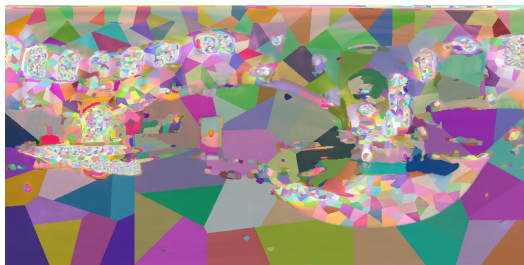
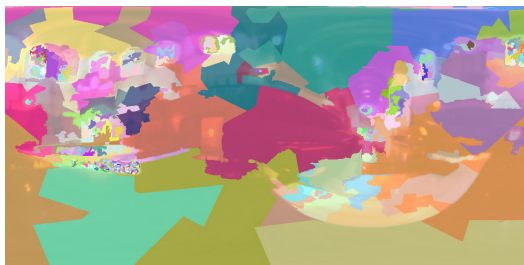


Fig. 4. Adaptive merging using adjacency matrix. (a) Initialized adjacency matrix. (b) Supposing region 5 is merged into region 2, the adjacency matrix is updated accordingly.



(a)



(b)

Fig. 5. The adaptive split-and-merge scheme for the result in Fig. 3. (a) Segments after the adaptive splitting. (b) Segments after the adaptive merging. The results are blended with the original image.

Figure 5(b) shows the final strata. In this example, we set the sample number to be 300. Although the final strata may not keep the original segment boundaries from the initial mean-shift clustering, we observe that such discrepancy often takes place in low-intensity regions, while highlight region boundaries are better maintained.

Once the adaptive merging is completed, we collapse all the pixels within a stratum to a directional light source located at the center, by integrating pixel intensities. This preserves the overall illumination contribution of the distant environment.

4 EXPERIMENTS AND DISCUSSION

In this section, we make comparison with five state-of-the-art methods, including structured importance sampling (structured for short) [2], hierarchical sample warping (HSW for short) [4], spherical q^2 -tree [3], Lightcuts [12] and Ostromoukhov et al's method (FHIS for short) [11]. Among those methods, the structured method [2], spherical q^2 -tree [3] and FHIS [11] sample environment map only. For the HSW method [4], which samples the product of environment map and BRDF function, we set the BRDF function to be a constant. Lightcuts [12] establishes a binary light tree to cluster a group of lights from many directional lights. According to the experiment settings in [12], we generated 3000 initial light samples via structured sampling method. We then omit material and geometry-related terms and use a horizontal cut to get clustered light samples.

To quantitatively measure the rendering quality, we prepare the ground truth results by taking each pixel in environment maps (with a size of 1024×512) as a directional light. Since recent evaluation of image quality metrics shows that no image quality metric is significantly better than others for different rendering distortions [30] [31], we adopt typical metrics of three different types, i.e. SSIM metric [32] that accounts for structure and contrast changes, HDR-VDP2 metric [33] that predicts visual differences, as well as the popular RMSE metric. The mean-opinion score (MOS) from HDR-VDP2 information is used to estimate the rendering quality [33]. Rendering results similar to the ground truth should have high SSIM values, high HDRVDP2 MOS values and low RMSE values.

4.1 Rendering a Ball with Different BRDFs and Environment Maps

In the first experiment, we employ seven HDR environment maps [34] [5] to render a ball with a fixed sample number (i.e. 300). We choose different BRDF settings, including diffuse and phong of increasing specularity. The detailed settings are listed in Table 1, where ka , kd , ks are ambient reflection coefficient, diffuse reflection coefficient and specular reflection coefficient respectively, and ns is the shininess index. For phong model, each combination (ks , ns) is used together with the diffuse settings.

TABLE 1
The settings of BRDF models

diffuse		phong	
ka	kd	ks	ns
0.05	0.5	0.3, 0.6, 0.9	1, 10, 100, 1000

We observed that the rendering results under different BRDF settings look generally similar to the ground truth.

TABLE 2
Mean and variance values of SSIM, RMSE, HDRVDP2 MOS for using different environment maps. The best values are marked in bold.

SSIM(mean, var)	Structured [2]	HSW [4]	Q ² -tree [3]	Lightcuts [12]	FHIS [11]	Our
campus	0.9957, 3.94E-05	0.9967, 2.10E-05	0.9959, 3.09E-05	0.9958, 3.00E-05	0.9963, 2.63E-05	0.9965, 2.22E-05
galileo	0.9968, 2.08E-05	0.9963, 2.61E-05	0.9967, 2.24E-05	0.9966, 2.04E-05	0.9935, 7.81E-05	0.9971, 1.71E-05
grace_flame	0.9996, 3.00E-07	0.9995, 6.95E-07	0.9997, 2.74E-07	0.9989, 2.97E-06	0.9987, 3.76E-06	0.9998, 1.75E-07
kitchen	0.9972, 1.67E-05	0.9971, 1.61E-05	0.9968, 2.06E-05	0.9970, 1.72E-05	0.9971, 1.57E-05	0.9972, 1.46E-05
readingroom	0.9972, 1.66E-05	0.9987, 3.03E-06	0.9986, 4.54E-06	0.9981, 7.65E-06	0.9974, 1.00E-05	0.9986, 3.96E-06
rnl	0.9963, 3.21E-05	0.9971, 1.93E-05	0.9968, 2.45E-05	0.9967, 2.32E-05	0.9970, 2.05E-05	0.9970, 2.01E-05
stpeters	0.9962, 3.62E-05	0.9964, 2.80E-05	0.9964, 3.08E-05	0.9960, 3.48E-05	0.9921, 1.12E-04	0.9967, 2.45E-05
RMSE(mean, var)	Structured [2]	HSW [4]	Q ² -tree [3]	Lightcuts [12]	FHIS [11]	Our
campus	0.0113, 7.19E-05	0.0155, 6.75E-05	0.0091, 6.38E-05	0.0183, 7.09E-05	0.0193, 5.61E-05	0.0114, 5.54E-05
galileo	0.0087, 4.41E-05	0.0150, 7.42E-05	0.0090, 5.49E-05	0.0153, 4.60E-05	0.0165, 1.21E-04	0.0114, 4.78E-05
grace_flame	0.0054, 1.23E-05	0.0248, 6.59E-05	0.0043, 1.44E-05	0.0319, 6.24E-05	0.0334, 1.28E-04	0.0062, 1.02E-05
kitchen	0.0101, 6.84E-05	0.0221, 1.25E-04	0.0101, 8.87E-05	0.0168, 8.19E-05	0.0144, 8.61E-05	0.0124, 6.84E-05
readingroom	0.0132, 8.43E-05	0.0170, 4.99E-05	0.0076, 4.25E-05	0.0169, 7.53E-05	0.0296, 1.13E-04	0.0119, 4.59E-05
rnl	0.0103, 7.17E-05	0.0188, 8.80E-05	0.0094, 7.10E-05	0.0139, 7.45E-05	0.0144, 6.91E-05	0.0155, 5.78E-05
stpeters	0.0089, 5.86E-05	0.0170, 7.19E-05	0.0091, 5.40E-05	0.0152, 6.27E-05	0.0339, 2.20E-04	0.0122, 4.62E-05
MOS (mean, var)	Structured [2]	HSW [4]	Q ² -tree [3]	Lightcuts [12]	FHIS [11]	Our
campus	83.4186, 1.07E-04	83.4192, 2.22E-05	83.4222, 1.49E-05	83.4061, 9.70E-04	83.4155, 1.17E-04	83.4192, 3.43E-05
galileo	83.4172, 1.39E-04	83.3856, 2.00E-03	83.4053, 8.69E-04	83.3501, 1.85E-02	83.2711, 3.13E-02	83.4056, 3.87E-04
grace_flame	83.4238, 6.18E-07	83.4193, 3.38E-05	83.4241, 7.38E-07	83.3775, 3.60E-03	83.3988, 1.90E-03	83.4242, 1.22E-07
kitchen	83.4098, 5.81E-04	83.3847, 1.50E-03	83.3926, 2.30E-03	83.3429, 9.80E-03	83.3858, 1.50E-03	83.3922, 1.80E-03
readingroom	83.4164, 6.00E-05	83.3668, 6.73E-04	83.4218, 1.71E-05	83.4154, 9.19E-05	83.3307, 1.04E-02	83.4014, 4.16E-04
rnl	83.4225, 1.19E-05	83.4203, 1.92E-05	83.4231, 3.73E-06	83.4148, 2.14E-04	83.4205, 1.90E-05	83.4204, 1.80E-05
stpeters	83.4214, 3.29E-05	83.4163, 9.25E-05	83.4222, 8.50E-06	83.4128, 2.62E-04	83.1569, 1.77E-01	83.4205, 2.24E-05

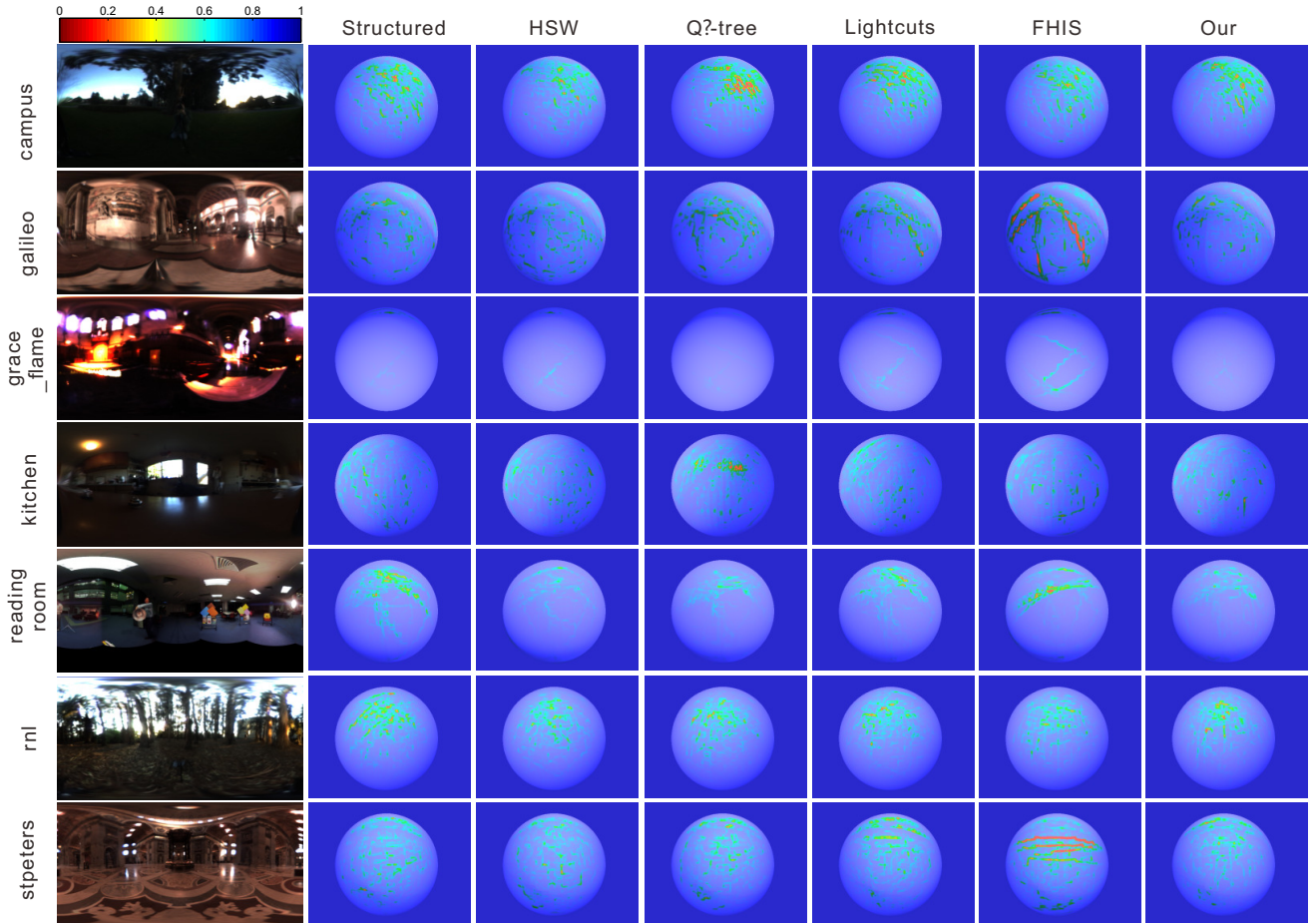


Fig. 6. SSIM maps for the phong model with $k_s = 0.6$, $n_s = 10$. The rows correspond to seven environment maps, and each column denotes one sampling method. In the color map from blue to red, the red color represents large error (low SSIM value) between the rendering and the ground truth.

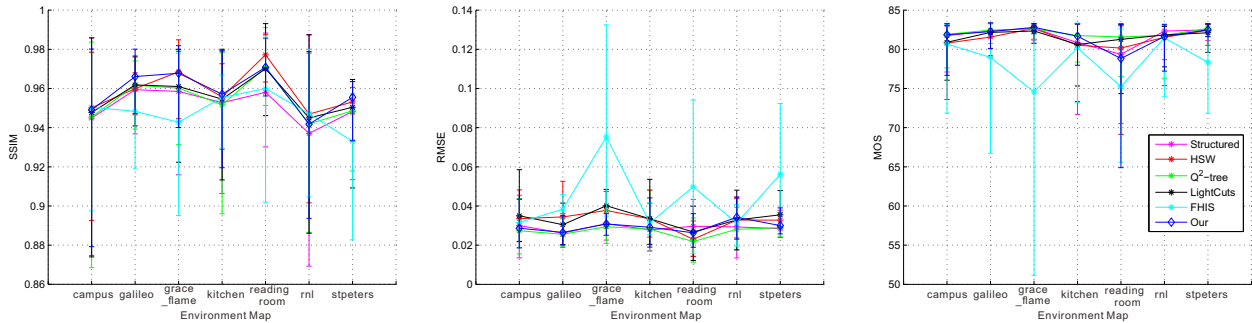


Fig. 7. SSIM, RMSE and HDRVDP2 MOS values for rotating environment maps when rendering the “Girl” model.

TABLE 3
Ranking of SSIM, RMSE and HDRVDP2 MOS in Fig. 7.

SSIM/RMSE/MOS	Structured [2]			HSW [4]			Q ² -tree [3]			Lightcuts [12]			FHIS [11]			Our		
campus	6	3	3	2	5	5	5	1	1	4	6	4	1	4	6	3	2	2
galileo	5	2	2	4	5	5	3	1	1	2	4	4	6	6	6	1	3	3
grace_flame	5	3	2	1	4	3	4	1	4	3	5	5	6	6	6	2	2	1
kitchen	5	1	3	2	5	5	6	2	1	4	6	4	3	4	6	1	3	2
readingroom	6	5	4	1	2	3	3	1	1	4	4	2	5	6	6	2	3	5
rnl	6	2	1	2	4	5	4	1	3	3	5	2	1	3	6	5	6	4
stpeters	5	1	3	2	4	4	4	2	1	3	5	5	6	6	6	1	3	2

Towards different environment maps, the mean and variance values of the three image quality metrics among different BRDF settings are summarized in Table 2. Our method gets maximum SSIM values and also minimum SSIM variance values for four environment maps, while HSW [4] yields three champions. For RMSE and HDRVDP2 MOS values, spherical q²-tree [3] and the structured method [2] report better statistics, yet our method has the third rank among the six comparative approaches.

We select a set of rendering results with obvious discrepancies on image quality metrics for visualization. Figure 6 shows the SSIM error maps using the phong model ($ks = 0.6$, $ns = 10$). The SSIM error maps are color-coded from red to blue. For “campus”, “galileo”, “kitchen” and “stpeters” maps, all comparative methods have obvious differences to the ground truth. For “grace_flame”, spherical q²-tree [3] and our results show almost no error, and thus the images appear almost completely blue. For “readingroom” and “rnl”, HSW [4] has best results. FHIS [11] gets worst results in “galileo” and “rnl”. In general, our method have comparative or better performance for seven scenes. It is worth noting that in this experiment, no shadow occurs. As will be demonstrated in the following experiments, our method in a general sense can produce better and more robust rendering results.

4.2 Rendering by Rotating Environment Maps

In the second experiment, we investigate the impact of rotating environment maps, using 36 rotation angles evenly distributed in $[0^\circ, 350^\circ]$ along y axis. Here, we render the “Girl” model, which is shown at the last second row in Figure 9.

Figure 7 plots mean and variance values of the three image quality metrics in rotating each environment map. Our method has close mean values to HSW [4] in terms of SSIM metric, and both of them are relatively larger than other methods for most maps. On the other hand, spherical q²-tree [3] and the structured [2] report lower RMSE and higher HDRVDP2 MOS mean values. We further summarize the ranking of different sampling methods in Table 3. The five comparative methods always have lower ranks for either one or more metrics. Particularly, FHIS [11] has maximum lowest ranks and Lightcuts [12] have minimum highest and/or second highest ranks. It can be clearly seen that except for some cases, our method has higher ranks for all the three metrics.

Figure 8 shows the image quality metrics at different rotating angles by using “galileo” environment map. We can see that all the methods have varying values for different rotating angles. In this example, although our method is not always the best, it in a general sense outperforms other sampling methods, since it achieves rather high SSIM, HDRVDP2 MOS values, and rather low RMSE values.

4.3 Rendering a Scene with More Environment Maps

In the third experiment, we evaluate the effect of using 20 different environment maps given one 3D scene. Besides the first 7 environment maps used in previous experiments, we randomly choose 13 environment maps from the Internet [35], which cover both indoor scenes and outdoor scenes. A 3D scene different from that used in the previous experiment is selected, i.e. the “Tableset” model shown at the third row in Figure 9. The three metric values for

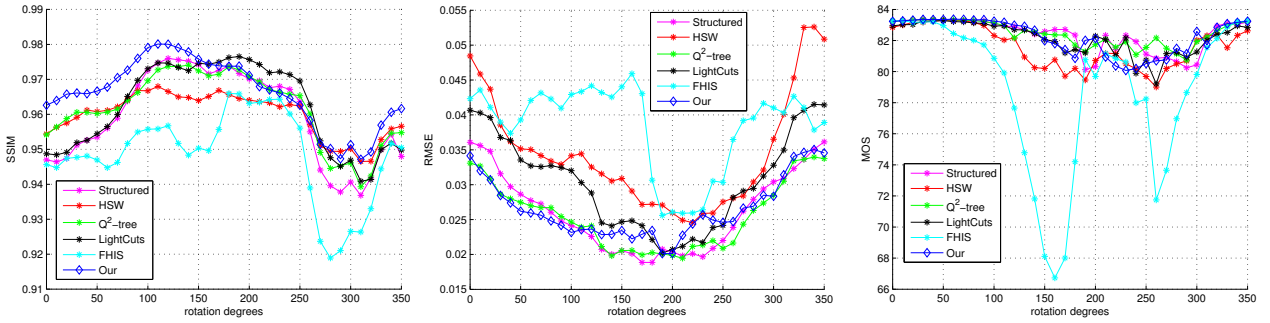


Fig. 8. SSIM, RMSE, and HDRVP2 values for rotating “galileo” environment map when rendering the “Girl” model.

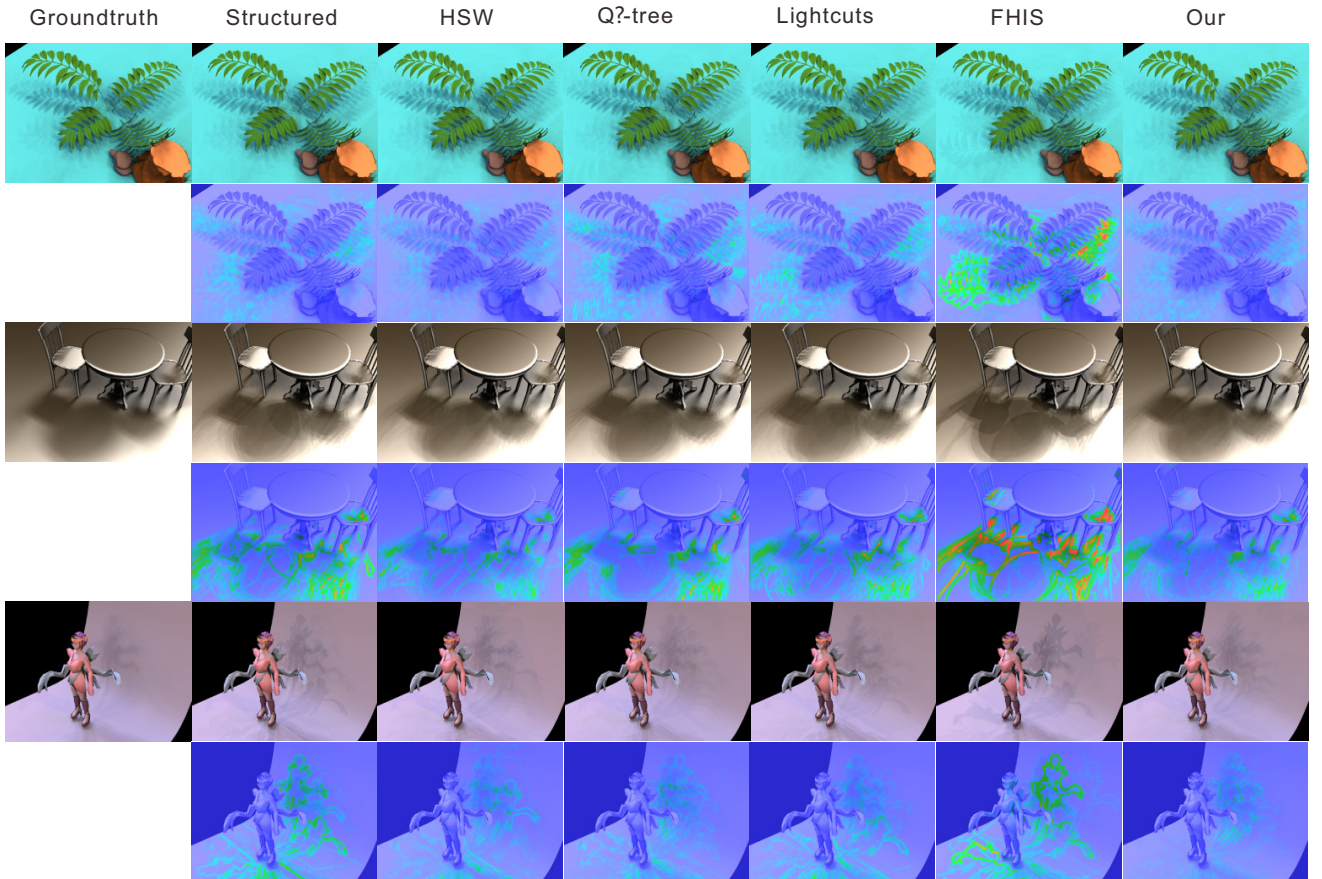


Fig. 9. Rendering results and their SSIM maps in different scenes by different methods.

different sampling methods are listed in Table 4. At the last row, the average ranks of different methods considering all the environment maps are calculated for each image quality metric respectively.

We can see that each sampling method has best performance for certain image quality metric and some environment maps. Among all the methods, spherical q^2 -tree [3] reports best values only for RMSE and MOS metrics, yet none for SSIM metric. By counting the ranks, we find HSW [4] has the highest rank for SSIM metric, spherical q^2 -tree [3] has the highest ranks for RMSE and MOS metrics, while our method yields the second highest ranks

for all the three image quality metrics.






4.4 Rendering by Varying Viewpoints

Next, we render three 3D scenes illuminated with “grace_flame”, “readingroom” and “stpeters” environment maps respectively, and change the view points randomly by 20 times for each scene.

Figure 9 visualizes the rendering results for one view point and their corresponding SSIM maps. For the “Tree” model and “Tableset” model, all the sampling methods except FHIS [11] generate similar rendering results, while our method and HSW [4] produce a bit better SSIM maps.

TABLE 4

SSIM, RMSE, HDRVDP2 MOS for 20 different environment maps. The best values are marked in bold, and the second best ranks are marked in bold with *.

	SSIM/RMSE/MOS	Structured [2]	HSW [4]	Q ² -tree [3]	Lightcuts [12]	FHIS [11]	Our
campus		0.9475 0.0274 80.8371	0.9491 0.0314 79.6775	0.9456 0.0255 80.8204	0.9416 0.0386 76.4454	0.9530 0.0380 79.0787	0.9506 0.0285 79.7815
galileo		0.9418 0.0278 79.9375	0.9526 0.0314 78.5506	0.9482 0.0256 80.2434	0.9462 0.0312 78.8113	0.9111 0.0465 70.5254	0.9533 0.0270 79.0340
grace_flame		0.9804 0.0156 80.4121	0.9803 0.0200 77.7017	0.9862 0.0135 79.8199	0.9502 0.0439 56.1932	0.8533 0.1124 41.1819	0.9867 0.0174 81.0675
kitchen		0.9601 0.0273 81.1698	0.9605 0.0598 78.6639	0.9413 0.0316 81.4471	0.9557 0.0381 80.9802	0.9622 0.0362 80.0806	0.9576 0.0305 81.8723
readingroom		0.9143 0.0470 79.1897	0.9484 0.0400 81.3506	0.9352 0.0363 81.1201	0.9214 0.0559 77.1054	0.8449 0.1629 48.2580	0.9436 0.0378 81.3268
rnl		0.8857 0.0480 78.0470	0.9161 0.0421 80.5293	0.9205 0.0349 81.0023	0.9089 0.0668 76.3495	0.9219 0.0425 80.6057	0.9197 0.0395 79.4642
stpeters		0.9297 0.0408 79.6333	0.9606 0.0459 82.3815	0.9422 0.0332 82.0503	0.9402 0.0453 80.3768	0.9018 0.1069 62.0952	0.9543 0.0337 81.6971
alexs		0.8941 0.0424 77.2774	0.9205 0.0399 77.6036	0.9015 0.0380 80.2461	0.9064 0.0584 75.4999	0.9228 0.0358 79.7375	0.9174 0.0367 79.2291
arches		0.9534 0.0315 78.7726	0.9441 0.0479 68.3988	0.9375 0.0392 73.3989	0.9468 0.0468 72.1862	0.9494 0.0396 71.9081	0.9360 0.0484 68.7692
caveroom		0.9770 0.0266 82.7876	0.9861 0.0391 83.0431	0.9725 0.0457 81.7287	0.9803 0.0357 82.4610	0.9395 0.1667 64.5904	0.9806 0.0337 82.9630
circus		0.9910 0.0167 83.2252	0.9843 0.0428 82.1184	0.9873 0.0203 82.6211	0.9909 0.0205 82.8552	0.9855 0.0403 78.6087	0.9877 0.0290 82.2584
factory		0.9519 0.0342 80.4581	0.9624 0.0315 81.7278	0.9606 0.0268 82.3228	0.9572 0.0715 80.0781	0.9572 0.0474 72.9679	0.9533 0.0400 78.9181
footprint		0.9202 0.0408 80.8934	0.9291 0.0474 79.1887	0.9123 0.0435 79.4961	0.9196 0.0545 76.9411	0.9310 0.0392 80.9511	0.9267 0.0511 79.7722
goldroom		0.9388 0.0304 78.8877	0.9484 0.0339 78.4738	0.9468 0.0258 81.1057	0.9435 0.0336 78.0721	0.9214 0.0440 72.6391	0.9549 0.0323 78.3232
hdrvfx		0.9493 0.0247 79.9815	0.9426 0.0279 78.7617	0.9425 0.0244 80.3093	0.9484 0.0369 75.1249	0.9113 0.0381 69.7812	0.9420 0.0299 76.3575
lobby		0.9274 0.0427 81.4583	0.9291 0.0549 81.5585	0.9136 0.0471 81.7331	0.9279 0.0665 79.2021	0.9411 0.0549 79.5008	0.9308 0.0509 79.7781
mans		0.9026 0.0508 68.7024	0.9260 0.0451 74.3374	0.9099 0.0437 74.2457	0.9220 0.0559 68.6139	0.9242 0.0403 74.6750	0.9328 0.0389 73.1137
monvalley		0.8969 0.0526 70.7949	0.9183 0.0522 70.3072	0.8929 0.0534 72.1918	0.8970 0.0849 62.2418	0.9201 0.0504 68.5107	0.9179 0.0467 73.0573
naturelab		0.9239 0.0393 81.4367	0.9460 0.0427 81.3075	0.9262 0.0375 82.0922	0.9165 0.0630 77.0433	0.8890 0.0875 64.8859	0.9458 0.0325 82.5327
provwash		0.9361 0.0422 82.4951	0.9553 0.0418 82.8147	0.9535 0.0303 83.0431	0.9436 0.0522 82.1719	0.8905 0.1141 68.7964	0.9688 0.0281 83.2402
Average Rank		4.25 2.7 2.95	2.45 4.1 3.35	4.1 2.05 2.15	4.05 5 4.85	3.7 4.5 4.8	2.55* 2.65* 2.9*

For the “Girl” model, the shadows behind and beneath the girl from different methods vary greatly. We can clearly see that the structured method [2], Lightcuts [12], and FHS [11] suffer from strong shadows. HSW [4] and spherical q^2 -tree [3] are more similar to our result, yet their SSIM maps reveal that both methods have much more differences around girl feet.

We compute the three image quality metrics for each rendering result and plot their curves in Figure 10. Due to occlusion changes, the metric values vary against the viewpoint. For the “Girl” model, our method has highest SSIM and HDRVDP2 MOS values, and almost smallest RMSE values. Although HSW [4] is ranked the best for the “Tableset” and “Tree” models in terms of SSIM, it reports relatively poor RMSE and HDRVDP2 values. We can clearly see the robust and better performance of our method over other methods in comparison.

4.5 Rendering by Varying Sample Number

We now evaluate the robustness of different methods for varying sample numbers. In this experiment, we render the three scenes by varying the sample number from 80 to 400, with an increment of 20, under the viewpoint and lighting environment orientation in Figure 9. Given a

sample number, we average 10 runs of the renderings for each method.

Figure 11 plots the curves of image quality metrics against the sample number. Generally speaking, the RMSE error decreases when the sample number becomes large for the six methods, and the SSIM and HDRVDP2 MOS values are in the reversal. In comparison, our method only reports relatively poor HDRVDP2 MOS values for the “Tableset” model, and achieves comparable or even better performance for the other eight cases.

4.6 Rendering by Using Environment Sequence

Although our method aims for single environment map sampling, we also tested its performance on an environment sequence. Specifically, we adopt the “grace_flame” sequence [5], and sample 300 points per frame to render the “Girl” model. As shown in Figure 12, our method has higher SSIM and HDRVDP2 MOS values in all frames than other methods. Besides, several methods including ours, spherical q^2 -tree [3], structured [2], and HSW [4], have relatively close RMSE values, among which ours is a bit better. However, we have to point out that our method does not consider the temporal coherence between environment

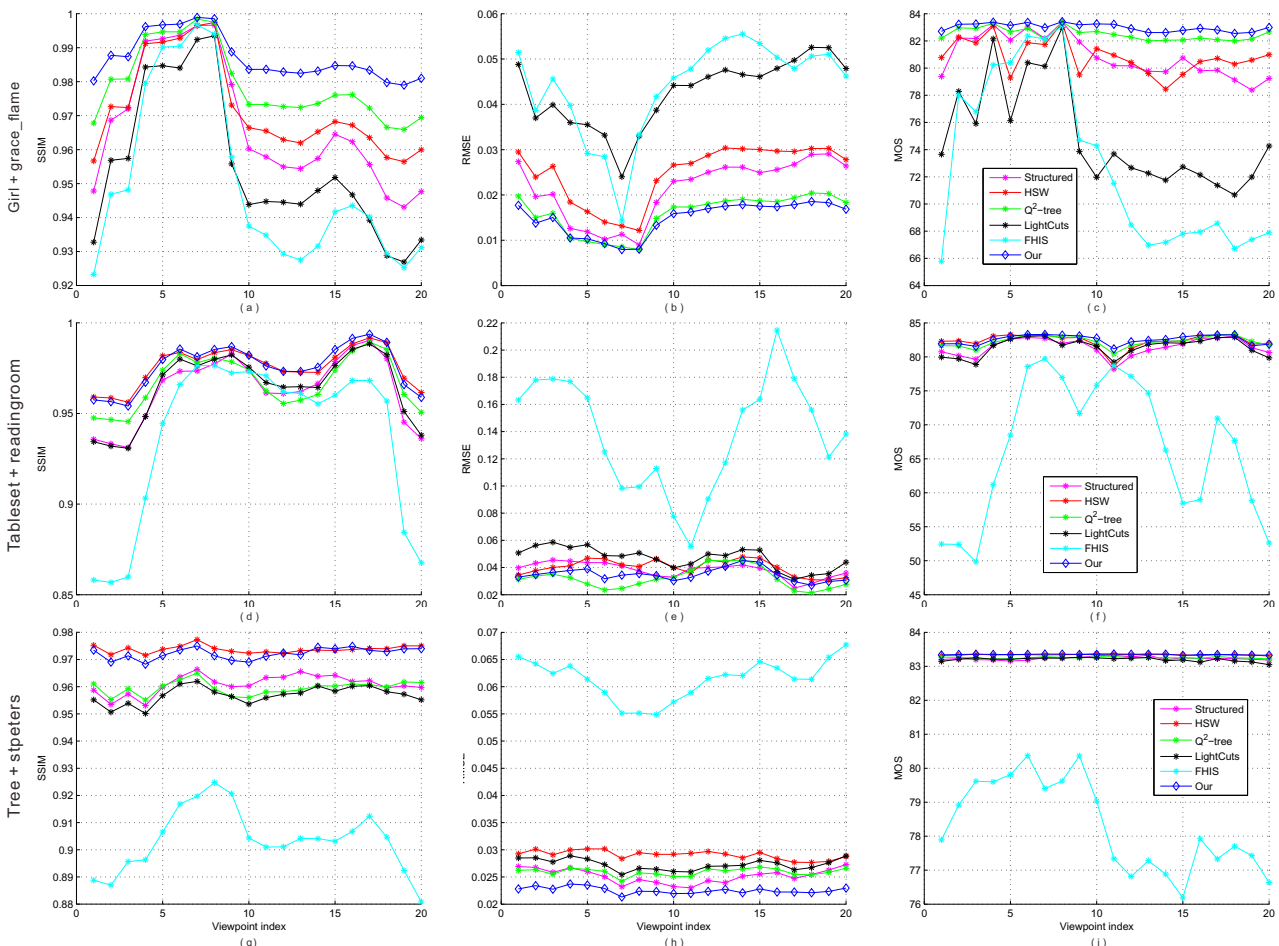


Fig. 10. SSIM, RMSE, HDRVDP2 MOS values for rendering at different view points in three scenes.

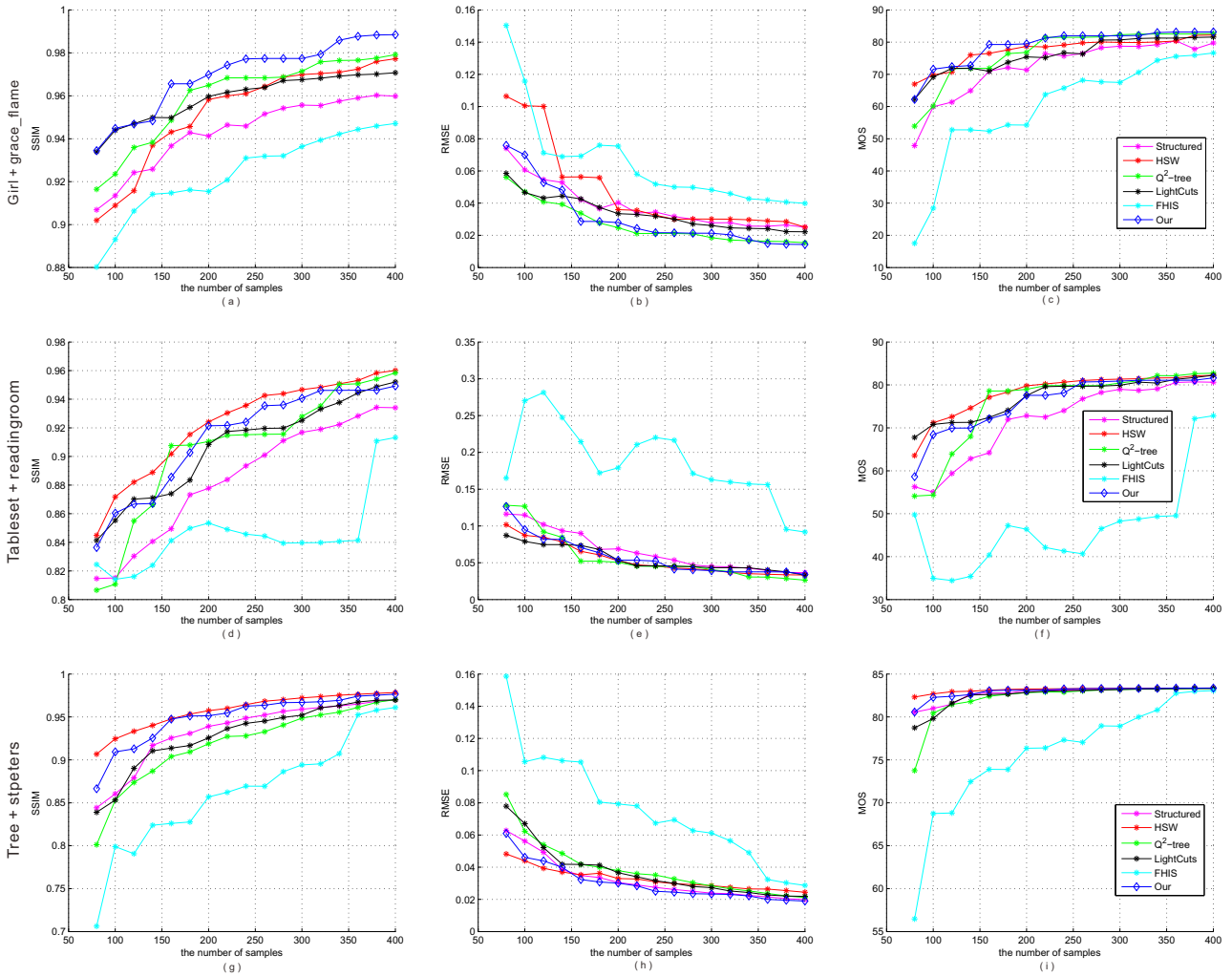


Fig. 11. SSIM, RMSE, HDRVDP2 MOS values for using different sample numbers in three scenes.

frames, and hence our results still suffer from flickering artifacts, as demonstrated in the supplementary video.

Here we visualize rendering results and SSIM maps from 20th, 60th and 100th frames in Figure 13. We can clearly see that our results contain less difference regions than those from other methods.

4.7 Timing Performance

Now we report the time cost for different methods. All experiments are conducted on a 3.0GHz Intel Core i5 computer with four cores and 8G memory. Table 5 shows the timing for different methods to generate 300 samples from a 1024×512 environment map. Structured [2] and HSW [4] methods are implemented using Matlab codes (M for short), others methods using C/C++ code (C for short). Spherical q^2 -tree costs only 0.387s for sampling one environment map, which seems to be the fastest method. Our method uses Edison codes [21] for mean-shift clustering, which consumes 10.36s; the rest steps cost 2.32s on average.

TABLE 5
Processing time for different methods.

Method	Time(s)
Structured [2] (M)	5.303
HSW [4] (M)	10.287
Q^2 -tree [3] (C)	0.387
Lightcuts [12] (C)	52.916
FHIS [11] (C)	1.997
Our (C)	10.359+2.324

5 CONCLUSION

We have presented a new environment sampling method by relying on the well-known mean-shift clustering. We explore the impact of high dynamic range on the clustering and achieve adaptive mean-shift clustering by adopting a global tone mapping technique. After generating segments capturing light region boundaries, an adaptive split-and-merge scheme is developed to generate strata with better balanced importance metric values, and the strata number can be user-controlled. The effectiveness of the proposed method is validated through both visual comparison and

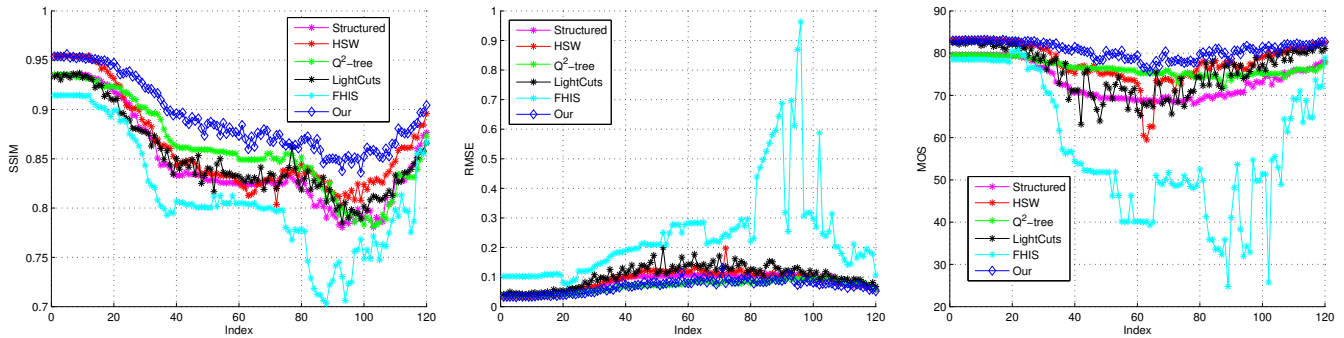


Fig. 12. SSIM, RMSE, HDRVDP2 MOS values for rendering by using “grace_flame” environment sequence.

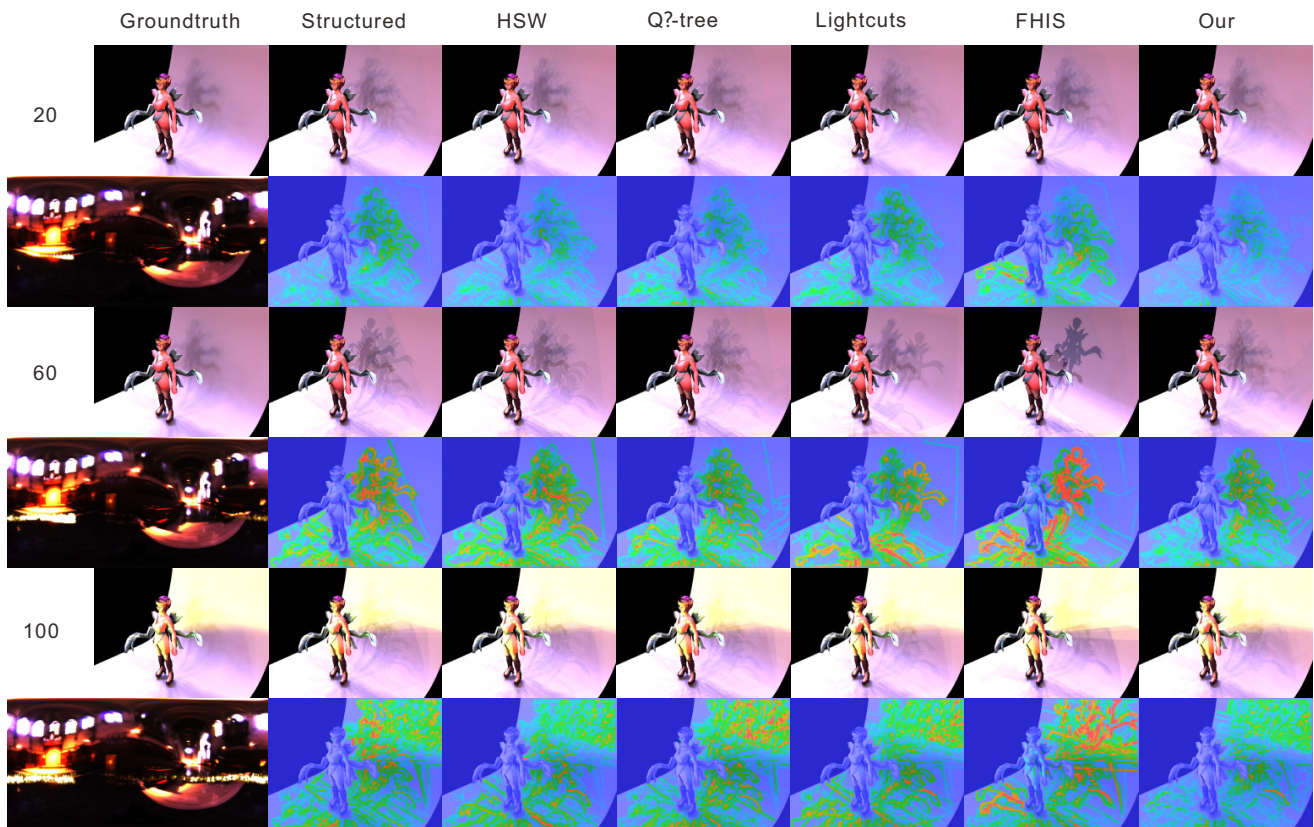


Fig. 13. Rendering results and SSIM maps by using three frames of “grace_flame” environment sequence.

quantitative evaluation. We found that our method achieves comparable and even better rendering quality, and it is more robust to the variation of viewpoint, environment rotation and sample number. Our method is focused on the sampling of single environment map. As a future work, we would like to extend it to handle a sequence of dynamic environment maps by considering temporal coherence.

ACKNOWLEDGMENTS

The authors thank all reviewers and the associate editor for their valuable comments. The authors also thank Rui Li for assisting the experiments. The work was supported by the National Natural Science Foundation of China (61100122, 61572354, 61272449), the National Science

and Technology Support Project (2013BAK01B01), Tianjin Science Foundation for Youth (12JCQNJC00100), and New Century Excellent Talents in University (NCET-11-0365).

REFERENCES

- [1] P. E. Debevec and J. Malik, “Recovering high dynamic range radiance maps from photographs,” in *Siggraph*, 1997, pp. 369–378.
- [2] S. Agarwal, R. Ramamoorthi, S. Belongie, and H. Jensen, “Structured importance sampling of environment maps,” *ACM Transactions on Graphics*, vol. 22, no. 3, pp. 605–612, 2003.
- [3] L. Wan, T.-T. Wong, and C.-S. Leung, “Spherical q^2 -tree for sampling dynamic environment sequences,” in *Proceedings of the Sixteenth Eurographics conference on Rendering Techniques*. Eurographics Association, 2005, pp. 21–30.
- [4] P. Clarberg, W. Jarosz, T. Akenine-Möller, and H. W. Jensen, “Wavelet importance sampling: efficiently evaluating products of complex functions,” *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 1166–1175, 2005.

- [5] L. Wan, S.-K. Mak, T.-T. Wong, and C.-S. Leung, "Spatiotemporal sampling of dynamic environment sequences," *IEEE Transactions on Visualization and Computer Graphics*, vol. 17, no. 10, pp. 1499–1509, 2011.
- [6] P. Debevec, "A median cut algorithm for light probe sampling," in *ACM SIGGRAPH 2006 Courses*, 2006, p. 6.
- [7] W. Jarosz, N. A. Carr, and H. W. Jensen, "Importance sampling spherical harmonics," *Computer Graphics Forum*, vol. 28, no. 2, pp. 577–586, 2009.
- [8] R. Y. Rubinstein, *Simulation and the Monte Carlo method*. Wiley-Interscience, 2009, vol. 190.
- [9] S. Gibson and A. Murta, "Interactive rendering with real-world illumination," in *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*. Springer-Verlag, 2000, pp. 365–376.
- [10] T. Kollig and A. Keller, "Efficient illumination by high dynamic range images," in *Proceedings of the 14th Eurographics workshop on Rendering*. Eurographics Association, 2003, pp. 45–50.
- [11] V. Ostromoukhov, C. Donohue, and P.-M. Jodoin, "Fast hierarchical importance sampling with blue noise properties," *ACM Transactions on Graphics (Proc. of SIGGRAPH)*, vol. 23, no. 3, pp. 488–495, 2004.
- [12] B. Walter, S. Fernandez, A. Arbre, K. Bala, M. Donikian, and D. P. Greenberg, "Lightcuts: a scalable approach to illumination," *ACM Transactions on Graphics*, vol. 24, no. 3, pp. 1098–1107, 2005.
- [13] T. Annen, Z. Dong, T. Mertens, P. Bekaert, H.-P. Seidel, and J. Kautz, "Real-time, all-frequency shadows in dynamic scenes," *ACM Transactions on Graphics*, vol. 27, no. 3, p. 34, 2008.
- [14] P. Clarberg and T. Akenine-Möller, "Practical product importance sampling for direct illumination," *Computer Graphics Forum (Proc. of Eurographics 2008)*, vol. 27, no. 2, pp. 681–690, 2008.
- [15] V. Havran, M. Smyk, G. Krawczyk, K. Myszkowski, and H.-P. Seidel, "Interactive system for dynamic scene lighting using captured video environment maps," in *Proceedings of the Sixteenth Eurographics conference on Rendering Techniques*. Eurographics Association, 2005, pp. 31–42.
- [16] S. Paris and F. Durand, "A topological approach to hierarchical segmentation using mean shift," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2007, pp. 1–8.
- [17] D. Barash and D. Comaniciu, "A common framework for nonlinear diffusion, adaptive smoothing, bilateral filtering and mean shift," *Image and Vision Computing*, vol. 22, no. 1, pp. 73–81, 2004.
- [18] D. Comaniciu, V. Ramesh, and P. Meer, "Kernel-based object tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 5, pp. 564–577, 2003.
- [19] K. Fukunaga and L. Hostetler, "The estimation of the gradient of a density function, with applications in pattern recognition," *IEEE Transactions on Information Theory*, vol. 21, no. 1, pp. 32–40, 1975.
- [20] Y. Cheng, "Mean shift, mode seeking, and clustering," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 8, pp. 790–799, 1995.
- [21] D. Comaniciu and P. Meer, "Mean shift: A robust approach toward feature space analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 5, pp. 603–619, 2002.
- [22] B. Georgescu, I. Shimshoni, and P. Meer, "Mean shift based clustering in high dimensions: a texture classification example," in *Proceedings of the 9th IEEE International Conference on Computer Vision*, vol. 2, 2003, pp. 456–463.
- [23] J. Wang, B. Thiesson, Y. Xu, and M. F. Cohen, "Image and video segmentation by anisotropic kernel mean shift," in *Proceedings of the European Conference on Computer Vision*, vol. 3022, 2004, pp. 238–249.
- [24] F. Li and R. Klette, "Adaptive mean shift-based clustering," Computer Science Department, The University of Auckland, New Zealand, Tech. Rep., 2008.
- [25] A. Mayer and H. Greenspan, "An adaptive mean-shift framework for mri brain segmentation," *IEEE Transactions on Medical Imaging*, vol. 28, no. 8, pp. 1238–1250, 2009.
- [26] C. Yang, R. Duraiswami, N. Gumerov, and L. Davis, "Improved fast gauss transform and efficient kernel density estimation," in *Proceedings of the 9th IEEE International Conference on Computer Vision*, 2003, pp. 664–671.
- [27] D. Freedman and P. Kisilev, "Fast mean shift by compact density representation," in *IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 1818–1825.
- [28] S. Ferradans, M. Bertalmio, E. Provenzi, and V. Caselles, "An analysis of visual adaptation and contrast perception for tone mapping," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 33, no. 10, pp. 2002–2012, 2011.
- [29] D. S. Hochbaum and D. B. Shmoys, "A best possible heuristic for the k-center problem," *Mathematics of operations research*, vol. 10, no. 2, pp. 180–184, 1985.
- [30] R. Herzog, M. Čadík, T. O. Aydın, K. I. Kim, K. Myszkowski, and H.-P. Seidel, "NoRM: no-reference image quality metric for realistic image synthesis," *Computer Graphics Forum*, vol. 31, no. 2, pp. 545–554, 2012.
- [31] M. Čadík, R. Herzog, R. Mantiuk, K. Myszkowski, and H.-P. Seidel, "New measurements reveal weaknesses of image quality metrics in evaluating graphics artifacts," *ACM Transactions on Graphics (Proc. of SIGGRAPH Asia)*, vol. 31, pp. 1–10, 2012.
- [32] Z. WANG, A. BOVIK, H. SHEIKH, and E. SIMONCELLI, "Image quality assessment: From error visibility to structural similarity," *IEEE Transactions on Image Processing*, vol. 13, no. 4, pp. 600–612, 2004.
- [33] R. Mantiuk, K. J. Kim, A. G. Rempel, and W. Heidrich, "Hdr-vdp-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions," vol. 30, no. 4, pp. 40:1–40:14, 2011.
- [34] P. Debevec, "Rendering synthetic objects into real scenes: Bridging traditional and image-based graphics with global illumination and high dynamic range photography," in *Siggraph*, Orlando, Florida, USA, July 1998, pp. 189–198.
- [35] C. Bloch, "sIBL archive: Free hdri sets for smart image-based lighting." [Online]. Available: <http://www.hdrlabs.com/sibl/archive.html>

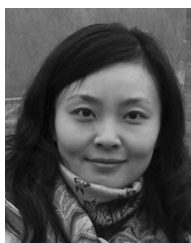


Wei Feng (M'10) received the B.S. and M.Phil. degrees in Computer Science from Northwestern Polytechnical University, China, in 2000 and 2003 respectively, and the Ph.D. degree in Computer Science from City University of Hong Kong in 2008. From 2008 to 2010, he worked as research fellow at the Chinese University of Hong Kong and City University of Hong Kong, respectively. He is currently an associate professor in school of computer science and technology,

Tianjin University. His major research interest is media computing, specifically including general Markov Random Fields modeling, discrete/continuous energy minimization, image segmentation, semi-supervised clustering, structural authentication, and generic pattern recognition. He got the support of the Program for New Century Excellent Talents in University, China, in 2011.



Ying Yang received the B.Eng and M.Eng degrees in computer science and engineering from Tianjin University, P.R. China, in 2012 and 2014, respectively. This work was conducted when she took her Master study.



Liang Wan (M'08) received the B.Eng and M.Sci degrees in computer science and engineering from Northwestern Polytechnical University, P.R. China, in 2000 and 2003, respectively. She obtained a Ph.D. degree in computer science and engineering from The Chinese University of Hong Kong in 2007. She is currently an Associate Professor in the School of Computer Software, Tianjin University, P. R. China. Her research interest is mainly on computer graphics, including

image-based rendering, non-photorealistic rendering, pre-computed lighting, and image processing.



Changguo Yu received the M.Sci degree in computer software engineering from Tianjin University, P.R. China, in 2014. This work was conducted when he took his Master study.