Vis Comput (2010) 26: 543–553
DOI 10.1007/s00371-010-0503-5

ORIGINAL ARTICLE

Animation rendering with Population Monte Carlo image-plane sampler

Yu-Chi Lai · Stephen Chenney · Feng Liu · Yuzhen Niu · Shaohua Fan

Published online: 17 April 2010

Abstract Except for the first frame, a population Monte Carlo image plane (PMC-IP) sampler renders with a start-up kernel function learned from previous results by using motion analysis techniques in the vision community to explore the temporal coherence existing among kernel functions. The predicted kernel function can shift part of the uniformly distributed samples from regions with low visual variance to regions with high visual variance at the start-up iteration and reduce the temporal noise by considering the temporal relation of sample distributions among frames. In the following iterations, the PMC-IP sampler adapts the kernel function to select pixels for refinement according to a perceptually-weighted variance criterion. Our results improve the rendering efficiency by a factor between 2 to 5 over existing techniques in single frame rendering. The rendered animations are perceptually more pleasant.

Keywords Ray-tracing · PMC · Monte Carlo · Global illumination

Y.-C. Lai funded by: NSC 99-2218-E-011-005-, Taiwan.

Y.-C. Lai
National Taiwan University of Science and Technology, Taipei, ROC
e-mail: yu-chi@mail.ntust.edu.tw

S. Chenney
Emergent Game Technology, Chapel Hill, USA
e-mail: schenney@gmail.com

F. Liu
University of Wisconsin-Madison, Madison, USA
e-mail: fliu@cs.wisc.edu

Y. Niu
Shandong University, Jinan, China
e-mail: yuzhen@cs.wisc.edu

S. Fan (✉)
Suzhou University Financial Engineering Research Center, Suzhou, China
e-mail: shaohua@cs.wisc.edu

S. Fan
NYU Courant Institute of Mathematical Sciences, New York, USA

1 Introduction

Global illumination based on Monte Carlo integration provides the most general solution for photorealistic rendering problems. To reduce image noise (variance) at practical computation times is the core of research in the global illumination community. To render an animation the temporal variance at each pixel among consecutive frames must also be considered because our eyes are good at noticing temporal inconsistency among consecutive frames for survival purposes; our algorithm is derived from the Population Monte Carlo (PMC) sampling framework, which is a technique that adapts sampling distributions over iterations, all with theoretical guarantees on error and little computational overhead. Our algorithm generates a start-up kernel function from previous rendered frames by considering the temporal correlation among the kernel functions. In addition, our algorithm evenly distributes the variance over the image plane in each frame to remove noisy spikes on the image and in turn, reduce the temporal inconsistency generally existing in a frame-by-frame ray-tracing-based algorithm.

PMC algorithms iterate on a population of samples. For our sampler, image-plane sampling (PMC-IP), the population is a set of image-plane locations. To render each frame, the population is initialized with a sample distribution whose predictions are based on the previous rendering results, and then PMC-IP generates an intermediate image to start the iteration. Any information available at this stage can then be used to adapt a *kernel function* that produces a new population. The initial prediction of the kernel function is based on the result of the current frame rendered with a few samples in each pixel, the result of previous frames, and the sample distribution of previous frames. We can explore the temporal coherence of sample distributions among consecutive frames by using computer vision techniques to generate a good start-up kernel function. This prediction prevents the redundant probed samples on smooth regions of the image. In addition, the prediction takes the temporal variance into account: perceptually high variance regions in previous frames have higher probability to be perceptually high variance regions in current frame, and as a result, the algorithm puts more samples at these regions to reduce the noisy spikes. In image-plane sampling, the perceptually-weighted variance in the intermediate images is used to construct the kernel function, resulting in more image-plane samples in regions of high variance. The procedure is then iterated: sample, adapt, sample, ... The result is an unbiased adaptive algorithm. This can achieve an *evenly-distributed* variance over the image plane.

Photon mapping and path tracing have been the industrial rendering algorithms for global illumination. Our sampler can be easily incorporated into ray-tracing-based global illumination with minimal modifications to improve the efficiency of these algorithms. We demonstrate the ease of incorporation into the current rendering framework by modifying the ray-starting pixel positions.

Our contribution is a specific tool for rendering that uses the Population Monte Carlo framework: An *image-plane sampler*, PMC-IP, that adapts to guide samples to perceptually high variance image regions, is cheap to compute, maintains stratification, and is unbiased. In addition, we incorporate motion analysis techniques from the vision community into global illumination algorithms to explore the temporal coherence of sample distributions among consecutive frames to *improve the sample usage and enhance the temporal consistency* among consecutive frames.

We include results comparing each algorithm to existing approaches. We find that PMC-based algorithms improve the efficiency in a factor of 2 to 5 over existing methods when rendering a single frame. In addition, our algorithm can generate a more perceptually pleasant animation than others.

2 Related work

Here we focus on the adaptive image-plane sampling and algorithms using sequential Monte Carlo algorithm. For an overview of Monte Carlo rendering in general, see Pharr and Humphreys [22].

Typically, adaptive image-plane algorithms perform a first pass with a small number of samples per pixel and use the resulting values to label pixels as adequately sampled or in need of further refinement. The algorithm then iterates on the pixels requiring more samples [3, 11, 20, 21, 23, 24].

A common property of these existing algorithms is that they stop sampling a given pixel when some image-derived metric is satisfied. As Kirk and Arvo [13] point out, the evaluation of the image metric relies on random samples, so there is some non-zero probability that the threshold is incorrectly detected and that sampling stops too soon. This introduces bias in the final image, which is a problem when physically accurate renderings are required. Our algorithm never uses a threshold to stop sampling a pixel and is statistically unbiased.

Many metrics have been proposed for the test to trigger additional sampling. Lee et al. [16] used a sample variance-based metric. Dippé and Wold [5] estimated the change in error as sample counts increase. Painter and Sloan [21] and Purgathofer [23] used a confidence interval test, which Tamstorf and Jensen [30] extended to account for the tone operator. Mitchell [20] proposed a contrast-based criterion because humans are more sensitive to contrast than to absolute brightness, and Schlick [27] included stratification into an algorithm that used contrast as its metric. Bolin and Meyer [3], Ramasubramanian et al. [24] and Farrugia and Péroche [7] used models for human visual perception, of which we use a variant. Rigau et al. [25, 26] introduced entropy-based metrics. References [1, 19] introduced metrics utilizing a model of the human visual system from the visible distortion based on the detection and classification of visible changes in the image structures.

Our algorithm views the image plane as a single sample space for the purposes of sampling. Dayal et al. [4] used a variance-based metric to control a kD-tree subdivision where samples are drawn uniformly within each adaptively sized cell of the subdivision. Stokes et al. [28] also took a global approach with their perceptual metric.

A Sequential Monte Carlo algorithm, similar in spirit to Population Monte Carlo, has recently been applied by Ghosh, Doucet and Heidrich [10] to the problem of sampling environment maps in animated sequences. Their work exploits another property of iterated importance sampling algorithms—the ability to re-use samples from one iteration to the next—and is complementary to our approach.

Lai et al. [14, 15] adapted the Population Monte Carlo algorithm into the energy redistribution framework to adapt the extent of energy redistribution. Their method focuses on the global rendering algorithm but our algorithm focuses on the first step of sample distributions. Our algorithm can be easily adapted to incorporate the algorithm by adjusting the energy of each path according to the kernel function of the pixel-position distribution.

3 PMC-IP: image-plane sampling

To render a frame in a sequence of animation is to compute the intensity, $I(i,j,t)$, of each pixel (i,j) for each frame t, by estimating the integrals:

$$I_{i,j,t} = \int_{\mathcal{I}} W_{i,j,t}(\mathbf{u}) L(\mathbf{x},\omega,t)\, d\mathbf{u}, \qquad (1)$$

where $\mathcal{I}$ is the image plane, $W_{i,j,t}(\mathbf{u})$ is the measurement function for pixel (i,j) of tth frame—non-zero if $\mathbf{u}$ is within the support of the reconstruction filter at (i,j) of tth frame—and $L(\mathbf{x},\omega,t)$ is the radiance leaving the point, $\mathbf{x}$, seen through $\mathbf{u}$ in the direction $-\omega$ at tth frame, determined by the projection function of the camera. We are ignoring, for discussion purposes, depth of field effects, which would necessitate integration over directions out of the pixel, and motion blur, which would require integration over time. In the following discussion, we will neglect the denotation of t for the simplification of description. And all the adaptation of the sample distribution happens in the process of rendering a single frame.

An image-plane sampler selects the image-plane locations, $\mathbf{x}$ in (1) for a specific frame. For simplicity, assume we are working with a ray-tracing-style algorithm that shoots from the eye out into the scene. Adaptive sampling aims to send more rays through image locations that have high noise, while avoiding bias in the final result.

Taking an importance sampling view, given a set of samples, $\{\mathbf{X}_1,\dots,\mathbf{X}_N\}$ from an importance function $p(\mathbf{x})$ for a single frame, each pixel is estimated using

$$\hat{I}_{i,j} = \frac{1}{n}\sum_{k=1}^{N} \frac{W_{i,j}(\mathbf{X}_k) L(\mathbf{X}_k,\omega)}{p(\mathbf{X}_k)}. \qquad (2)$$

The source of bias in most existing adaptive image-plane samplers is revealed here. Adaptive sampling without bias must avoid decisions to terminate sampling at an individual pixel, and instead look at the entire image plane to decide where a certain number of new samples will be cast. Every pixel with non-zero brightness must have non-zero probability of being chosen for a sample, regardless of its estimated error. This guarantee that all pixels will have chance to receive samples to achieve unbiasedness.

We also note the (2) can be broken into many integrals, one for the support of each pixel. Provided $p(\mathbf{x})$ is known in each sub-domain, the global nature of $p(\mathbf{x})$ is not important.

Figure 1 summarizes the final PMC-IP algorithm for a single frame:

```
1  Estimate the start-up kernel function $\alpha_k^{(0)}$ from previous frames
2  for $s = 0,\dots,S$
4      Use DMS to allocate samples according to $\alpha_k^{(s)}$
5      Generate samples from $K_{\mathrm{IP}}^{(s)}(\mathbf{x})$ and accumulate to image
6      Compute the perceptually-weighted variance image
7      Compute $\alpha_k^{(s+1)}$ for each pixel $k$
```

Fig. 1 The PMC-IP Algorithm for rendering a single frame

3.1 Integrating the sampler into a global rendering system

The PMC-IP is easily incorporated into a single rendering pipeline and allows us to improve the rendering efficiency for ray-tracing-based algorithms such as path tracing, photon mapping. Figure 2 shows a modern plug-in style Monte Carlo rendering framework. The only core framework modification required to support adaptive sampling is the addition of a feedback path from the output image generator back to the samplers, required to pass information from one sampling iteration back to the samplers for the next iteration. The PMC-IP sampler also contains a tracking algorithm, described in Sect. 3.2, to guess the sample distribution from the previous rendering frames and sample distributions.

The kernel function is the starting point in creating a PMC algorithm for adaptive image-plane sampling. We need a function that has adaptable parameters, is cheap to sample from, and supports stratification. This can be achieved with a *mixture model* of component distributions, $h_{\mathrm{IP}(i,j)}(\mathbf{x})$, one for each pixel:

$$K_{\mathrm{IP}}^{(s)}(\mathbf{x}) = \sum_{(i,j)\in\mathcal{P}} \alpha_{(i,j)}^{(s)} h_{\mathrm{IP}(i,j)}(\mathbf{x}), \qquad \sum_{(i,j)\in\mathcal{P}} \alpha_{(i,j)}^{(s)} = 1,$$

where (i,j) is the pixel coordinate and $\mathcal{P}$ is the set of all pixels in this image. Each component is uniform over the domain of a single pixel integral. The parameters to the distribution are all the $\alpha_{(i,j)}^{(s)}$ values, and these change for each iteration, s. We achieve an unbiased result if every $\alpha_{(i,j)}^{(s)} \geq \epsilon$, where ϵ is a small positive constant (we use 0.01). We enforce this through the adaptive process, and the use of ϵ, rather than 0, provides some assurance that we will not overlook important contributions (referred to as *defensive sampling* [12]).

The use of a mixture as the kernel results in a D-kernel PMC [6] algorithm. Sampling from such a distribution is achieved by choosing a pixel, (i,j) according to the $\alpha_{(i,j)}^{(s)}$, and then sampling from $h_{\mathrm{IP}(i,j)}(\mathbf{x})$. The latter can be done with a low-discrepancy sampler within each pixel, giving sub-pixel stratification. Stratification across the entire image plane can be achieved through deterministic mixture sampling, which we describe shortly. The importance function $p(\mathbf{x})$ in estimating contribution of a path for a given pixel must be modified as $p_{\mathrm{sample}}(\mathbf{x}) = p_{\mathrm{path}}(\mathbf{x}) p_{\mathrm{uniform}}/\alpha_{(i,j)}^{(s)}$ where p_{path} is the probability to generate the path starting from that pixel position and p_{uniform} is the probability to choose that pixel uniformly for guaranteeing unbiasedness. Notice that this kernel function is not conditional, in other words, $K_{\mathrm{IP}}(\mathbf{x}^{(s)}|\mathbf{x}^{(s-1)}) = K_{\mathrm{IP}}(\mathbf{x}^{(s)})$. Hence,

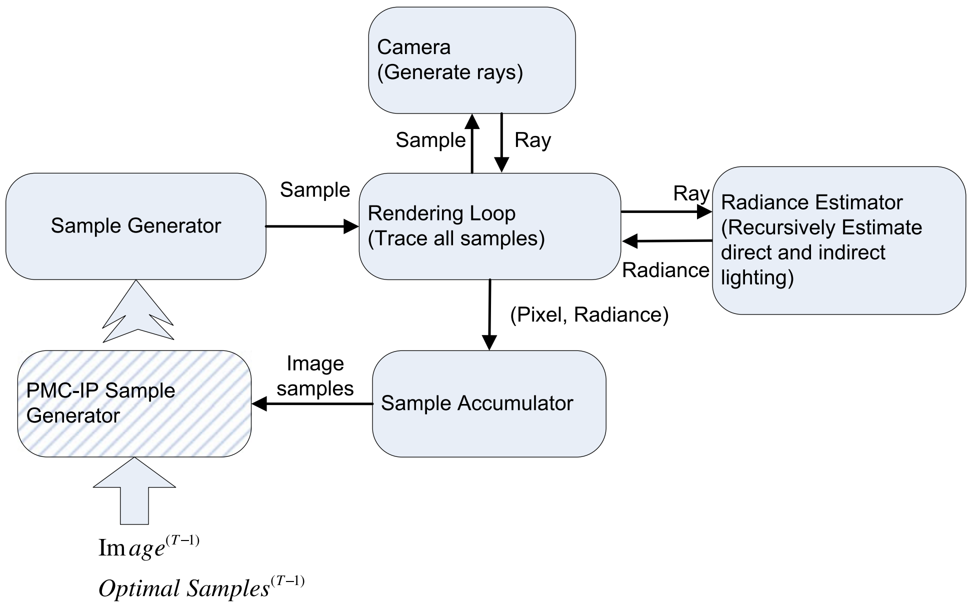


Fig. 2 A block diagram of a plug-in style Monte Carlo rendering system, following Pharr and Humphreys [22]. The PMC-IP sampler replaces a uniform sample generator with the addition of a feedback path from the sample accumulator in order to calculate the perceptual variance. The PMC-IP sampler also contains a predictor to estimate the start-up sample distribution for rendering a single frame

for image-plane sampling we do not include a resampling step in the PMC algorithm because no samples are re-used. The knowledge gained from prior samples is instead used to adapt the kernel function.

3.2 Predict a good initial start-up kernel

When observing the kernel function for each frame by using PMC-IP algorithm, we realize that the high-probability regions should be temporally correlated among consecutive frames. Thus, if we can use motion analysis techniques to predict the movement of these regions, we can save the extra cost of probing the entire image plane to estimate the $\alpha_{(i,j)}$. In addition, since the high-probability regions should also be regions with high variance in general Monte Carlo methods, to use the prediction can also reduce the temporal inconsistency between noise by putting more samples in these regions.

For the current frame t, we first obtain a coarse rendering result $\hat{I}^t$ using a small amount of initial samples per pixel such as 4 in our implementation. Then we predict the kernel function of the current frame $\alpha^t_{(i,j)}$ from that of the previous frame $\alpha^{t-1}_{(i,j)}$ by exploring the correspondence between $\hat{I}^t$ and I^{t-1} as follows:

$$\alpha^t_{(i,j)} = \alpha^{t-1}_{M^t_{t-1}(i,j)}, \tag{3}$$

where M^t_{t-1} describes the correspondence between α^t and α^{t-1}.

We approximate M^t_{t-1} by estimating the correspondence between images $\hat{I}^t$ and I^{t-1}. Optical flow [2, 18] algorithms in computer vision community provide natural solutions to estimate this correspondence. However, since the coarse rendering result $\hat{I}^t$ is noisy, optical flow estimation is not reliable. We regularize the optical flow using a homography between $\hat{I}^t$ and I^{t-1}. A homography is a 2D perspective matrix described by 8 parameters. It describes the correspondence between $\hat{I}^t$ and I^{t-1} as follows:

$$\begin{bmatrix} si^t \\ sj^t \\ s \end{bmatrix} = \begin{bmatrix} h_{11} & h_{12} & h_{13} \\ h_{21} & h_{22} & h_{23} \\ h_{31} & h_{32} & 1 \end{bmatrix} \begin{bmatrix} i^{t-1} \\ j^{t-1} \\ 1 \end{bmatrix} \tag{4}$$

Vision research has provided rich literatures for estimating the homography [29]. We adopt a feature-based method. SIFT features [17] are used due to their robustness to noise. Ideally, 4 pairs of correctly matched feature points are enough to estimate the homography. In practice, to get a robust estimation, we extract from each image a dense set of SIFT features. ($\hat{I}^t$ is pre-processed using a median filter to reduce the noise before estimating the homography.) We use a RANSAC [9] algorithm to obtain a robust estimation of the homography.

When the scene is a plane, or the camera only rotates along its optical center, the homography perfectly describes the correspondence between two images. In practice, when the difference between two consecutive frames are small, the homography can be a good approximation. However, when the camera motion or object motion is significant, the homography is not accurate enough. To relieve this problem, we first detect the high-density regions of the kernel α^{t-1}; then we estimate the homography within these regions between two images, and use this new homography to describe the correspondence between the high-density regions. The insight of this strategy is two-fold: first, a homography is likely to be successful when modeling the correspondence between small regions; second, the high-density region is

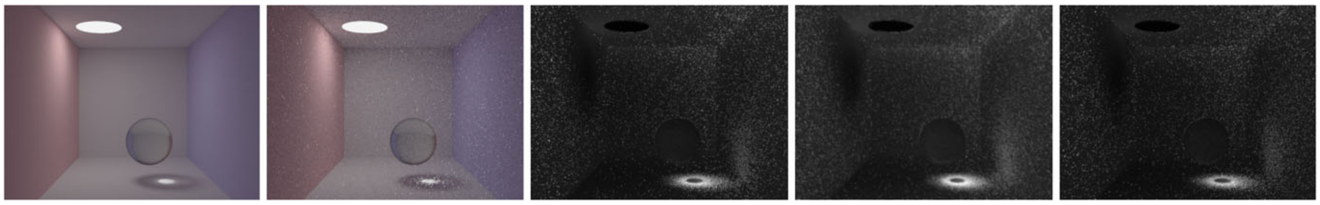


Fig. 3 This is the prediction of the kernel function at frame 4. The images from *left* are result at frame 3, rough result at frame 4, the final kernel function at frame 3, the final predicted kernel function at frame 4, and the final kernel function at frame 4. We analyze the deviation by using the root mean square error which is 9.02×10^{-07}

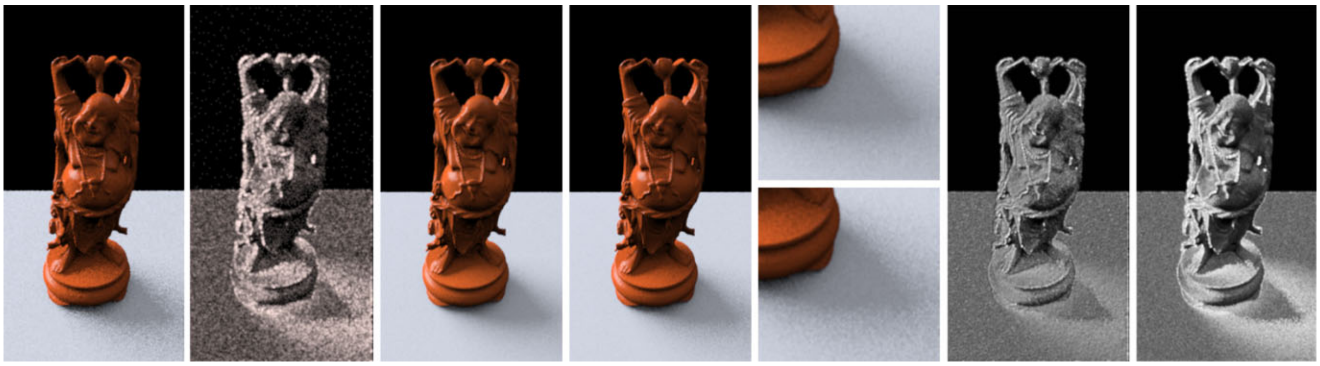


Fig. 4 A comparison between adaptive and uniform image-plane sampling on a direct-lighting example. *Leftmost* is the initial image for PMC-IP sampling, and the $\alpha_k^{(0)}$ image. The initial image used 2 samples per pixel. The *next* image is the result of PMC-IP sampling with two iterations at 4 SPPs on average. *Center* is a 10 SPPs image uniformly distributed. The zooms show the shadow near the Buddha's base (PMC-IP top, uniform bottom). To the *right* are the corresponding variance images. Note that the variance image for the PMC-IP sampler has few high variance regions, and has a lower contrast in general, representing a more even distribution of error

more important than the other region, thus deserving accurate prediction. An example of the kernel predication is illustrated in Fig. 3.

3.3 Adapting the PMC-IP kernel

The adaption method is responsible for determining the value of each $\alpha_{(i,j)}^{(s)}$ given the populations from previous iterations and any information available from them, such as the image computed so far. We need to define an $\alpha_{(i,j)}^{(s)}$ for every pixel, with pixels that require more samples having higher high $\alpha_{(i,j)}^{(s)}$ for the component that covers the pixel.

An appropriate criterion assigns $\alpha_{(i,j)}^{(s)}$ proportional to the perceptually-weighted variance at each pixel. The algorithm tracks the sample variance in power seen among samples that contribute to each pixel. To account for perception, the result is divided by the threshold-versus-intensity function $\mathrm{tvi}(L)$ introduced by Ferweda et al. [8]. Normalization also accounts for ϵ.

$$\alpha'_{i,j} = \frac{\overline{\sigma}^2_{(i,j)}}{\mathrm{tvi}(L_{(i,j)})},$$

$$\alpha^{(s)}_{i,j} = \epsilon + \frac{(1-\epsilon)\alpha'_{(i,j)}}{\sum_{(i',j')\in\mathcal{P}} \alpha'_{(i',j')}},$$

where σ^2 is the variance of radiance carried by samples dropped at that pixel.

The first iteration for the first frame of the algorithm samples uniformly over the image plane, so this criterion can always be computed. The first iteration for the following frames predict the kernel function according to the result of previous frame and the current rendering algorithm. The left images in Fig. 4 show an example of an $\alpha_{(i,j)}^{(0)}$ map for a given initial image. The perceptual term in the error image prevents very high errors in both bright regions (a problem with unweighted variance) and dark areas (a problem with luminance-weighted variance).

Note that $\alpha_{(i,j)}^{(s)} \geq \epsilon$, so there is a non-zero probability of generating a sample at any given image-plane location. This meets the requirement for importance sampling that the importance function is non-zero everywhere where the integrand is non-zero. Furthermore, as the total sample count approaches infinity, the count at any pixel also approaches infinity. Hence, with the correctly computed importance weights (2), the algorithm is unbiased.

3.4 Deterministic mixture sampling

Randomly sampling from the discrete distribution defined by the $\alpha_{(i,j)}^{(s)}$ produces excess noise—some pixels get far more or fewer samples than their expected value. This problem is solved with *deterministic mixture sampling*, DMS, which is designed to give each component (pixel) a number of samples roughly proportional to its $\alpha^{(s)}_{(i,j)}$. Deterministic mixture sampling is unbiased and always gives lower variance when compared to random mixture sampling, as proven by Hesterberg [12].

The number of samples per iteration, N (the population size) is fixed at a small multiple of the number of pixels. We typically use 4 samples per pixel, which balances between spending too much effort on any one iteration and the overhead of computing a new set of kernel parameters. For each pixel, the deterministic sampler computes $n'_{(i,j)} = N\alpha_{(i,j)}$, the target number of samples for that pixel. It takes $\lfloor n'_{(i,j)} \rfloor$ samples from each pixel (i, j)'s component. The remaining un-allocated samples are sampled from the *residual distribution* with probability $n'_{(i,j)} - \lfloor n'_{(i,j)} \rfloor$ at each pixel (suitably normalized).

4 Results

This section presents the rendering results when we apply our PMC-IP algorithm to render a single frame of scenes and animation scenes by plugging in our sampler into the modern global illumination framework to demonstrate the usefulness of our algorithm

4.1 Static image rendering

Adaptive image-plane sampling can be used in many situations where pixel samples are required and an iterative algorithm can be employed. We have implemented it in the contexts of direct lighting using a Multiple Importance Sampler (MIS) and global illumination with path tracing, and as part of a complete photon mapping system.

Figure 4 shows the Buddha direct-lighting example. The surface is diffuse with an area light source. Each pixel sample used 8 illumination samples, and the images were rendered at 256×512, with statistics presented in Table 1.We introduce the perceptually-based mean squared efficiency (P-Eff) metric for comparing algorithms, computed as:

$$\mathrm{Err} = \frac{\sum_{\mathrm{pixels}} e^2}{\mathrm{tvi}(L)}, \qquad \text{P-Eff} = \frac{1}{T \times \mathrm{Err}}$$

where e is the difference in intensity between a pixel and the ground truth value and T is the running time of the algorithm on that image. P-Eff is a measure of how much longer (or less) you would need to run one algorithm to reach the perceptual quality of another [22].

The final adaptive image shown is the unweighted average of three sub-images (initial and two iterations). While weighting each sub-image may be helpful, in this context it is not clear that the samples from one iteration are any better than those from another because they all used the same per-sample parameters. We obtained more samples in places that needed it, but not better samples.

Table 1 Measurements comparing PMC-IP and uniform image-plane sampling, for equal total sample counts. The P-Eff is perceptual efficiency used in [14]. The Buddha image computed direct lighting with the MIS method, with a total of 8 lighting samples for each pixel sample. PMC-IP sampling improves the perceptually-based RMS error by a factor of 5.4 over uniform sampling with only 7.5% more computation time. It corresponds to an improvement in efficiency of 5.01. The Cornell Box images use path tracing to compute global illumination including caustics. Comparing with images of 16 SPPs, PMC-IP improves the efficiency by a factor of 2.65

Image	Method	#SPP	T(s)	Err	P-Eff
Buddha	Uniform	10	58.1	0.625	0.027
	PMC-IP	$2+4+4$	62.4	0.116	0.138
Box	Uniform	16	163	0.545	0.011
	Uniform	32	328	0.255	0.012
	PMC-IP	$4+6+6$	169	0.182	0.033

The path tracing algorithm differs from a standard version only in how pixel locations are chosen. The improvement due to PMC-IP sampling is more pronounced in this situation because some areas of the image (the caustic, for instance) have a much higher variance than others due to the difficulty of sampling such paths. We compare the results in two aspects. First, we compare them visually. Working toward a target image quality, we would continue iterating the PMC-IP sampler until we were satisfied with the overall variance. In Fig. 5, we show the final result of the Cornell box and the comparison between a set of snapshots of the caustic region between the general PT algorithm and our adaptive algorithm. We can see that the result of 16th (equivalent to 64 SPPs) is even better than the result of 256 SPPs. We also notice that even at diffuse regions, our method converges more quickly than the general PT algorithm.

Second, we compare the efficiencies of our algorithm and the PT algorithm. In this Table 1, we see that PMC-IP sampling with a total of 16 SPPs improves the efficiency by a factor of 3 to the uniform sampling with 16 SPPs and 32 SPPs. In this result, we ran our examples for a fixed number of iterations (bounded by computation time). Note that because the PMC-IP sampler evenly spreads variance over the image, an overall image error bound is very unlikely to leave any high-error pixels.

Photon mapping is an industry standard method for global illumination, and we implemented the above method for the gather portion of a photon mapping implementation. Figure 6 shows a room scene computed with the system. Looking at the blown-up images of right wall by the lamp, in Fig. 6, we can see that our algorithm converges more rapidly

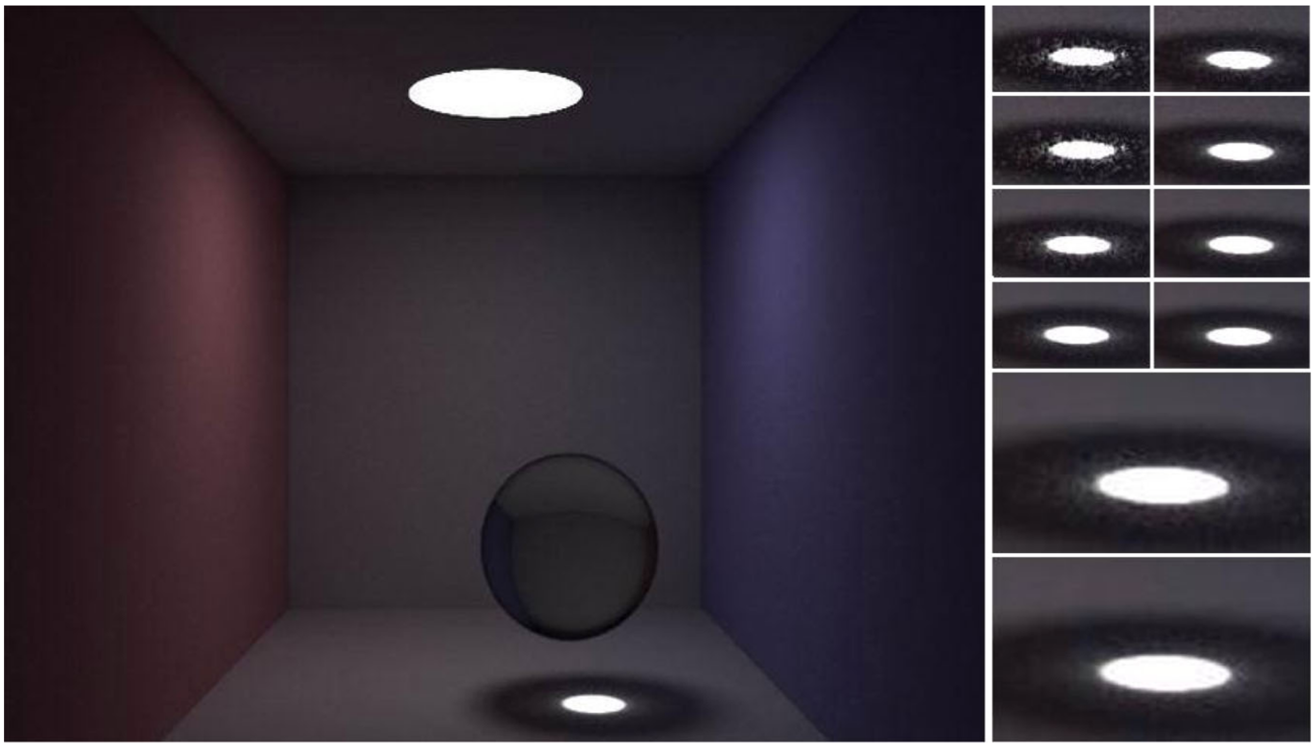


Fig. 5 A Cornell Box image computed using the PMC-IP algorithm. The *left* image is the final result using PMC-IP algorithm with 64 iterations, with each iteration averaging 4 SPPs. It is easier to find converged values in diffuse regions than in the caustic region. Thus, we compare the results by focusing on this region. The images in the *second row* on the *top*, from *top* to *down*, are the cropped images of the caustic region computed using non-adaptive path tracing with 16, 32, 64 and 128 SPPs. The images in the *third row* on the *top*, from *top* to *down*, are intermediate results from the adaptive algorithm at 4, 8, 16 and 32 iterations when computing the top image. The *right bottom* demonstrates that our adaptive sampler produces better visual results at lower sample counts: on the *top* is the result from 256 SPPs, unadapted, and on the *bottom* image is the result of 16 adapting iterations at an average of 4 SPPs per iteration

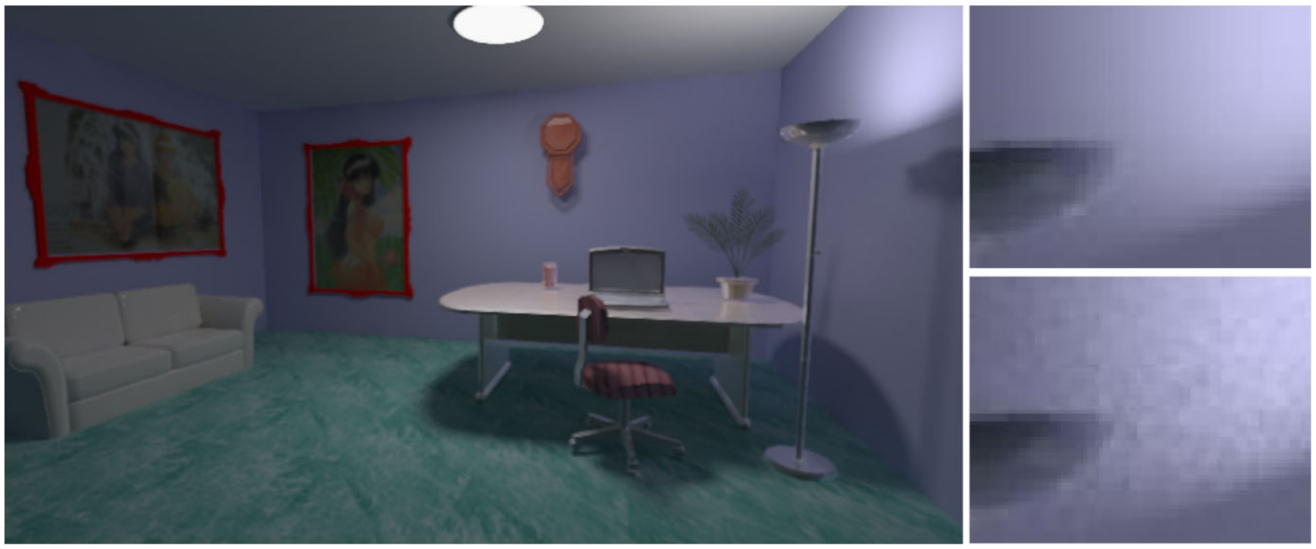


Fig. 6 The *top* is a room scene with complex models computed using photon mapping and the adaptive image plane sampler. The *bottom row* are blown up images of the upper right portion of the room scene. The image was generated with a standard photon shooting phase. On the *left* is the result of a final gather with 4 PMC-IP iterations, with each iteration averaging 4 samples per pixel. Right is the result of a standard photon mapping gather using 16 SPPs and using 16 shadow rays per light to estimate direct lighting. Note the significant reduction in noise with our methods

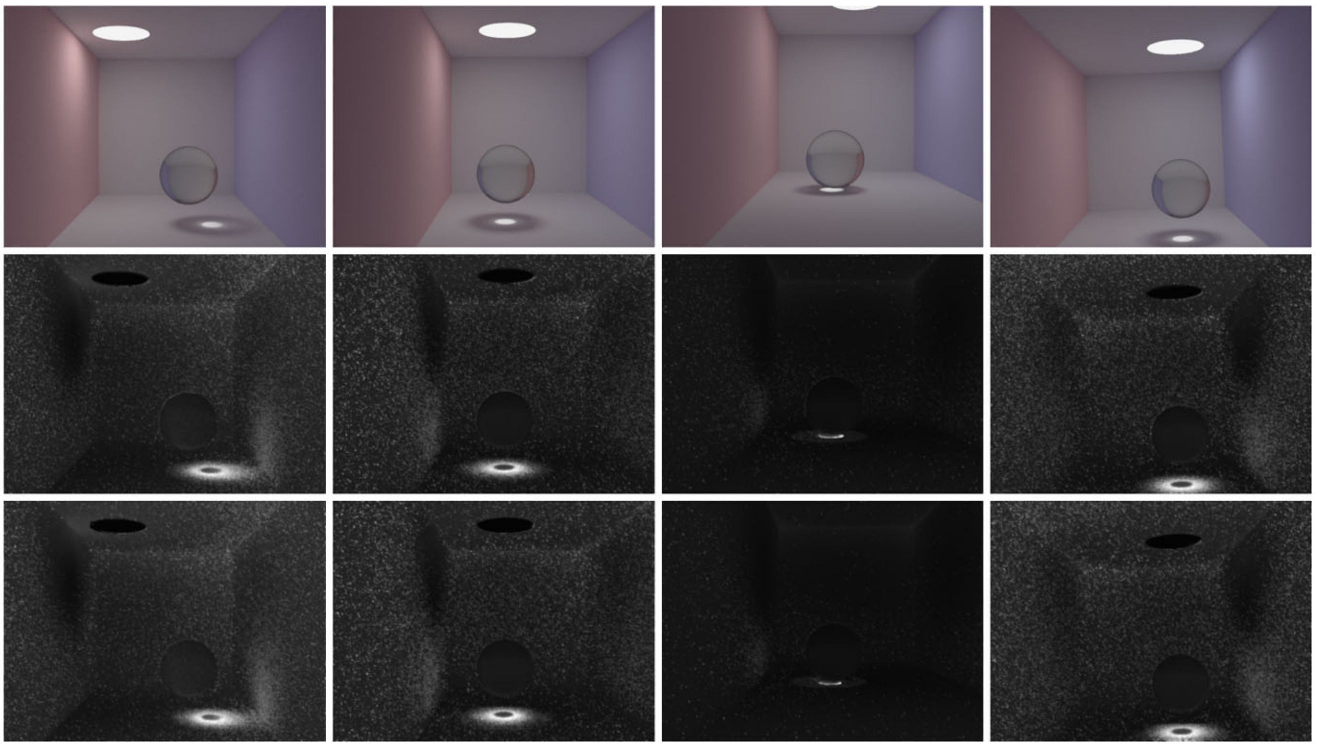


Fig. 7 A sequence of images animation of Cornell Box scene are rendered using the PMC-IP algorithm. The *top row* of images are the final results of the 1st, 31st, 61st, and 91st frame using PMC-IP algorithm with 16 iterations, with each iteration averaging 16 SPPs. The *second row* are the $\alpha_{(i,j)}$ estimated after rendering at the 1st, 31st, 61st, and 91st frame. The *third row* are the prediction of $\alpha_{(i,j)}$ using prediction algorithm in Sect. 3.2 for the 1st, 31st, 61st, and 91st frame. The corresponding root mean square error between the prediction and final value of $\alpha_{(i,j)}$ are 9.28×10^{-7}, 1.26×10^{-6}, 8.81×10^{-7}, and 1.16×10^{-6}

to a smooth image. This is because PMC-IP puts more samples in this region because of its high variance nature. Our algorithm improves the efficiency of the final result.

4.2 Animation rendered with the adaptive image plane sampler

We apply our frame-based PMC-IP animation rendering algorithm to two animation scenes: Cornell Box and Room. Each contains the movement of the camera, objects, and lights. For the Cornell Box scene, we render each frame with 16 iterations and with each iteration averaging 16 samples per pixel by plugging in PMC-IP into general path tracing algorithm. To do the comparison, we also render the animation with 256 samples per pixel by plugging in uniform sample distribution into the path tracing algorithm. For the room scene, we render each frame with 64 iterations and each iteration with averaging 16 samples per pixel and with uniform 1024 samples per pixel. The overhead of prediction of the kernel function is roughly 5 s. The overhead of adjusting the kernel function at each iteration is about 10 s. The cost of tracing a view ray through a scene is the same because the only difference is the start-up position of a view ray. Figure 7 shows the final results of 4 frames taken from a sequence of animation in a Cornell Box scene. When visually checking the prediction and final estimation, the position of the high-probability region and the strength are similar. When numerically analyzing the deviation between the prediction and final estimation of the kernel function, $\alpha_{(i,j)}$, with root mean square error for each frame, the values are small with a maximum of 1.26×10^{-6}. The predictor does a good job in tracking the high-sample region roughly corresponding to the caustics lighting on the ground. As a result, our algorithm consistently puts more effort at this region to get a smoother caustics lighting region. When comparing the animation result, we can see that the variance of the caustics region when using the uniform 256 SPPs is roughly the same as the variance of caustics regions when plugging in our algorithm with 4 iterations, with averaging 16 SPPs per iteration. Our algorithm with 16 iterations, with averaging 16 SPPs per iteration can generate a much smoother caustics region.

Figure 8 shows the final results of 4 frames taken from a sequence of animation in a room scene when plugging in PMC-IP and plugging in uniform sample distribution with the path tracing. We notice that our algorithm distributes more samples to high variance regions and therefore renders images with less noise spikes. In addition to even dis-

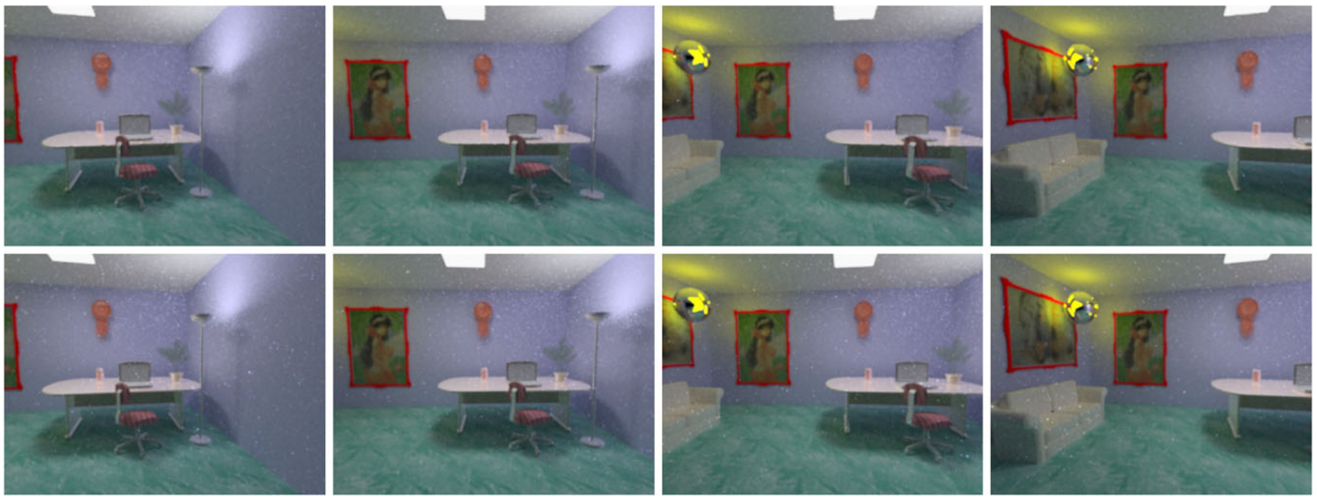


Fig. 8 A room scene is rendered using the PMC-IP algorithm. The *top row* of images are the final result using PMC-IP algorithm with 64 iterations, with each iteration averaging 16 SPPs at 1st, 31st, 61st, and 91st frame. The *second row* are the final results using general path tracing algorithm with 1024 sample per pixel at 1st, 31st, 61st, and 91st frame. There are obvious less artifacts existing in all four frames. When watching the animation, the noisy spike keep popping up in the room scene animation using path tracing algorithm

tribution of variance on the image plane, our algorithm uses the prediction of the kernel function to incorporate the temporal information of the sample distribution among consecutive frames. As a result, the inconsistency among consecutive frames is lower. We can observe this from how the chance of a noise spike popping up in the animation rendered by using general path tracing algorithm is higher than the chance in the animation rendered with our algorithm. The result is a more perceptually pleasant animation.

5 Conclusion

Adaptive image-plane sampling, derived from a PMC framework, learns to become an effective sampler based on the results from early iterations. Instead of starting from a uniform distribution, we use computer vision techniques to predict the start-up kernel function based on the previous results. This start-up kernel function shifts part of the uniformly distributed samples from regions with low visual variance to regions with high visual variance on the image plane and also provides us a base of considering temporal coherence of kernel functions among consecutive frames. The adaptive algorithm automatically adjusts the kernel function to approximate the ideal image-plane sample distribution. The prediction and adaptation of the kernel function can both improve the rendering efficiency and temporal consistency among frames. The pixel-position generator is a common component in many basic ray-tracing algorithms. PMC-IP sampler could be used as a plug-in component for essentially any algorithm that forms light paths through the eye, including the gather phase of photon mapping, bi-directional path tracing, irradiance caching, and so on. We have shown how photon mapping can use PMC samplers in the final gathering phase.

There are several future research directions. First, we apply the temporal prediction for the pixel-position distribution. In addition, direct lighting is another main factor for the quality of generated images. The importance of lights to a point in the scene should also be temporally coherent. If we can use similar prediction for the importance of lights, we can use more direct-lighting samples to estimate the intensity from an important light. Second, we only use perceptual metrics suitable for measuring the performance of rendering a single frame. However, it cannot measure human's sensitivity for the temporal inconsistency among consecutive frames. The sensitivity of the temporal inconsistency is very important cue for survival. A proper metrics to evaluate the performance among consecutive frames can allow us to further adjust our kernel functions to take the entire rendering sequence into account instead of a frame-based manner. Third, now we use a fixed number of iterations and a fixed number of samples per pixel when rendering a frame. However, some frames may need less samples than others. A convergence test on each frame can terminate the rendering process earlier. One step further, we may even want to have method that can predict the number of iterations and samples per pixel needed to generate a smooth image from the previous frames.

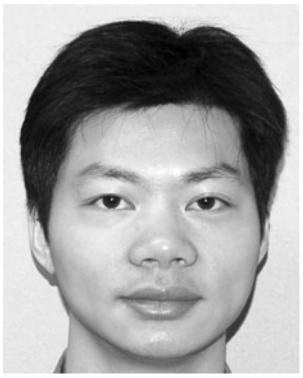
PMC algorithm is useful to take advantage of correlated information. The Kalman filter is a well-known example. We believe that it can provide proper solution to exploit the correlated and temporal-coherence information existing in integrals for animation rendering. In addition, motion analysis of a film in the vision community provides us with techniques to explore the temporal coherence which may be the key to render a consistent animation in an efficient way.

References

1. Aydin, T., Mantiuk, R., Myszkowski, K., Seidel, H.: (2008)
2. Baker, S., Black, M.J., Lewis, J., Roth, S., Scharstein, D., Szeliski, R.: A database and evaluation methodology for optical flow. In: IEEE ICCV (2007)
3. Bolin, M.R., Meyer, G.W.: A perceptually based adaptive sampling algorithm. In: SIGGRAPH '98, pp. 299–309 (1998)
4. Dayal, A., Woolley, C., Watson, B., Luebke, D.: Adaptive frameless rendering. In: Proc. of the 16th Eurographics Symposium on Rendering, pp. 265–275 (2005)
5. Dippé, M.A.Z., Wold, E.H.: Antialiasing through stochastic sampling. In: SIGGRAPH '85, pp. 69–78 (1985)
6. Douc, R., Guillin, A., Marin, J.M., Robert, C.P.: Convergence of adaptive sampling schemes. Technical Report 2005–2006, University Paris Dauphine (2005). http://www.cmap.polytechnique.fr/~douc/Page/Research/dgmr.pdf
7. Farrugia, J.P., Péroche, B.: A progressive rendering algorithm using an adaptive pereptually based image metric. Comput. Graph. Forum (Proc. Eurograph. 2004) **23**(3), 605–614 (2004)
8. Ferwerda, J.A., Pattanaik, S.N., Shirley, P., Greenberg, D.P.: A model of visual adaptation for realistic image synthesis. In: SIGGRAPH '96, pp. 249–258 (1996)
9. Fischler, M.A., Bolles, R.C.: Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. Commun. ACM **24**(6), 381–395 (1981). doi:10.1145/358669.358692
10. Ghosh, A., Doucet, A., Heidrich, W.: Sequential sampling for dynamic environment map illumination. In: Proc. Eurographics Symposium on Rendering, pp. 115–126 (2006)
11. Glassner, A.: Principles of Digital Image Synthesis. Morgan Kaufmann, San Mateo (1995)
12. Hesterberg, T.: Weighted average importance sampling and defensive mixture distributions. Technometrics **37**, 185–194 (1995)
13. Kirk, D., Arvo, J.: Unbiased sampling techniques for image synthesis. In: SIGGRAPH '91, pp. 153–156 (1991)
14. Lai, Y., Fan, S., Chenney, S., Dyer, C.: Photorealistic image rendering with population Monte Carlo energy redistribution. In: Eurographics Symposium on Rendering, pp. 287–296 (2007)
15. Lai, Y.C., Liu, F., Zhang, L., Dyer, C.: Efficient schemes for Monte Carlo Markov chain algorithms in global illumination. In: ISVC '08: Proceedings of the 4th International Symposium on Advances in Visual Computing, pp. 614–623 (2008)
16. Lee, M.E., Redner, R.A., Uselton, S.P.: Statistically optimized sampling for distributed ray tracing. In: SIGGRAPH '85, pp. 61–68 (1985)
17. Lowe, D.G.: Distinctive image features from scale-invariant keypoints. Int. J. Comput. Vis. **60**(2), 91–110 (2004)
18. Lucas, B., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proc. of International Joint Conference on Artificial Intelligence, pp. 674–679 (1981)
19. Mantiuk, R., Myszkowski, K., Seidel, H.P.: Visible difference predicator for high dynamic range images. In: Proceedings of IEEE International Conference on Systems, Man and Cybernetics, pp. 2763–2769 (2004)
20. Mitchell, D.P.: Generating antialiased images at low sampling densities. In: SIGGRAPH '87, pp. 65–72 (1987)
21. Painter, J., Sloan, K.: Antialiased ray tracing by adaptive progressive refinement. In: SIGGRAPH '89, pp. 281–288 (1989)
22. Pharr, M., Humphreys, G.: Physically Based Rendering from Theory to Implementation. Morgan Kaufmann, San Mateo (2004)
23. Purgathofer, W.: A statistical method for adaptive stochastic sampling. In: Proc. EUROGRAPHICS 86, pp. 145–152 (1986)
24. Ramasubramanian, M., Pattanaik, S.N., Greenberg, D.P.: A perceptually based physical error metric for realistic image synthesis. In: SIGGRAPH '99, pp. 73–82 (1999)
25. Rigau, J., Feixas, M., Sbert, M.: New contrast measures for pixel supersampling. In: Proc. of CGI'02, pp. 439–451. Springer, Berlin (2002)
26. Rigau, J., Feixas, M., Sbert, M.: Entropy-based adaptive sampling. In: Proc. of Graphics Interface 2003, pp. 149–157 (2003)
27. Schlick, C.: An adaptive sampling technique for multidimensional integration by ray-tracing. In: Proc. of the 2nd Eurographics Workshop on Rendering, pp. 21–29 (1991)
28. Stokes, W.A., Ferwerda, J.A., Walter, B., Greenberg, D.P.: Perceptual illumination components: a new approach to efficient, high quality global illumination rendering. In: SIGGRAPH '04, pp. 742–749 (2004)
29. Szeliski, R.: Image alignment and stitching: A tutorial. Tech. Rep. MSR-TR-2004-92, Microsoft Research (2006)
30. Tamstorf, R., Jensen, H.W.: Adaptive sampling and bias estimation in path tracing. In: Proc. of the 8th Eurographics Workshop on Rendering, pp. 285–296 (1997)

Yu-Chi Lai received the B.S. from National Taiwan University, Taipei, ROC, in 1996 in Electrical Engineering Department. He received his M.S. and Ph.D. degrees from University of Wisconsin—Madison in 2003 and 2009 respectively in Electrical and Computer Engineering and his M.S. and Ph.D. degrees in 2004 and 2010 respectively in Computer Science.
He is currently an assistant professor in NTUST and his Research interests are in the area of graphics, vision, and multimedia.

Stephen Chenney received his a Ph.D. in 2000 from the Computer Science division at the University of California at Berkeley. He is current a programmer in Emergent Game Technology.



Feng Liu received the B.S. and M.S. degrees from Zhejiang University, Hangzhou, China, in 2001 and 2004, respectively, both in computer science. He is currently a Ph.D. candidate in the Department of Computer Sciences at the University of Wisconsin-Madison, USA.
His research interests are in the areas of graphics, vision and multimedia.

Yuzhen Niu received the B.S. degree from Shandong University, Jinan, China, in 2005 in computer science. She is currently a Ph.D. candidate in the School of Computer Science and Technology at the Shandong University, China.
Her research interests are in the areas of graphics, vision and human-computer interaction.

Shaohua Fan received his B.S degree in Mathematics from Beijing Normal University, M.S. degree in Mathematical Finance from Courant Institute at NYU, and Ph.D. in computer science from University of Wisconsin-Madison. He is currently a professor at financial engineering research center at Suzhou University, China and a quantitative researcher at Ernst & Young in New York.
His research interests are statistical arbitrage, high-frequency trading, and Monte Carlo methods for financial derivatives pricing.