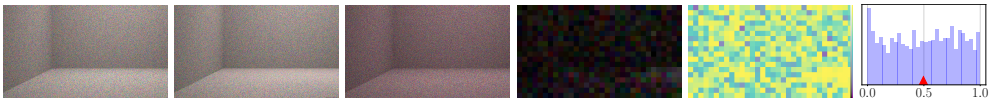
Detecting Bias in Monte Carlo Renderers using Welch's *t*-test

Alisa Jung Johannes Hanika Carsten Dachsbacher

Karlsruhe Institute of Technology



(a) PT (b) BDPT (c) Diff × 8 (d) Block diff (e) Color map (f) p-values

Figure 1. Welch's t-test in an empty, gray box: (a) an unbiased forward path tracer reference implementation compared to (b) a biased bidirectional path tracer, each limited to paths with three vertices, and 10 samples per pixel. The two center images show the difference (c) and tile-wise difference (d), revealing hardly any bias. Welch's t-test outputs a color map (e) revealing bias to the right and a non-uniform histogram of p-values (f) as strong evidence that (a) and (b) will not converge to the same image with more samples.

Abstract

When checking the implementation of a new renderer, one usually compares the output to that of a reference implementation. However, such tests require a large number of samples to be reliable, and sometimes they are unable to reveal very subtle differences that are caused by bias, but overshadowed by random noise. We propose using Welch's t-test, a statistical test that reliably finds small bias even at low sample counts. Welch's t-test is an established method in statistics to determine if two sample sets have the same underlying mean, based on sample statistics. We adapt it to test whether two renderers converge to the same image, i.e., the same mean per pixel or pixel region. We also present two strategies for visualizing and analyzing the test's results, assisting us in localizing especially problematic image regions and detecting biased implementations with high confidence at low sample counts both for the reference and tested implementation.

1. Introduction

The underlying integrals for generating photorealistic imagery are only solvable using statistical methods, the most common being Monte Carlo integration. In this work, we focus on verifying that an implementation of a Monte Carlo estimator does what

 ISSN 2331-7418

it is supposed to do. We only consider unbiased estimators, i.e., estimators whose expected value at any final sample count is equal to the true solution of the integral that needs to be solved—at least, if the estimator was designed and implemented correctly.

Conventional tests for verifying the correctness of a new renderer implementation are typically based on comparisons to a reference implementation. For example, we can compute difference images or analyze the behavior of the root-mean-square error (RMSE) at increasing sample counts with respect to a converged reference image. However, strong noise at lower sample counts can make those tests unreliable at low computation times. Even with increased computation times, subtle bias may still be hidden by weak remaining noise.

Visual metrics are an alternative approach to evaluate correctness with respect to ground-truth photographs. These techniques focus on the visual similarity in the context of the human visual system instead of on a purely mathematical error analysis. Their goal is to find images that look as plausible as possible, but they do not provide any measure of the mathematical accuracy of a renderer's implementation.

We propose using Welch's t-test, a statistical test, to determine whether an implementation of an unbiased renderer does indeed produce an unbiased result when tested against an unbiased reference implementation. Welch's t-test is a two-sided hypothesis test to decide whether the underlying means of two normal-distributed sample sets are equal. In the context of rendering, a correctly implemented unbiased Monte Carlo estimator converges to the underlying mean of the samples drawn during rendering. The test's results help detect bias and thus faulty implementations by considering statistics of the samples generated during rendering. We can thus use Welch's t-test to test whether the underlying mean of the distribution sampled by the Monte Carlo estimator is equal to the mean of a reference implementation. The test can only be used to check whether a new implementation is correct; to compare convergence speed, one still has to fall back on other methods such as RMSE over time or sample count.

Our contributions include

- a detailed description of the steps necessary for applying Welch's t-test to test implementations of supposedly unbiased renderers for bias;
- a method for converting samples generated by a Monte Carlo renderer to samples that can be used as input to Welch's t-test;
- two visualization schemes for analyzing the test's result;
- an evaluation of the test's behavior under various settings.

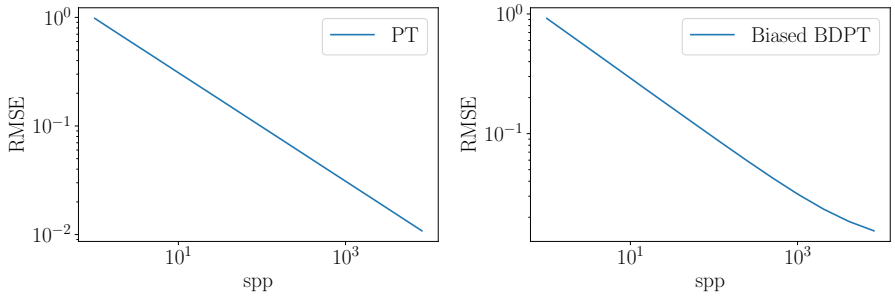


Figure 2. RMSE of an unbiased PT render and an intentionally slightly biased BDPT render of the corner scene (Figure 1(a)). The BDPT's bias only becomes noticeable with over 1000 samples per pixel (spp) which required two hours of render time, not including the time spent rendering the reference.

2. Related Work

Quantitative Image Metrics. Typical quantitative metrics for comparing two images are the root-mean-square error (RMSE) or the absolute-difference image. The RMSE between an image X and a reference R is the root of the normalized sum of squared pixel differences over the entire image with N pixels:

$$\mathrm{RMSE} = \sqrt{\frac{1}{N}\sum_{N}(x_i - r_i)^2}.$$

The log-log plot of an unbiased renderer's RMSE over samples is a straight line, due to the convergence behavior of Monte Carlo integration. However, testing in the form of checking whether the RMSE behaves as a straight line requires both a converged reference image and running the tested renderer for a large enough sample count to get a reliable plot. An example is shown in Figure 2.

The RMSE can also be adapted, e.g., by giving less weight to noise in especially bright regions using a term such as

$$\frac{1}{N}\sum_{N}\frac{(x_i - r_i)^2}{r_i^2 + \mathrm{offset}}.$$

Instead of waiting for the tested renderer to somewhat converge before computing the RMSE, [Celarek et al. 2019] compute many low-sample renderings and estimate the MSE's expected value and variance. This allows them to detect renderers with occasional outliers and measure the distribution of errors over frequencies. However they still rely on a converged reference, and use the same budget for rendering many unconverged renders as would typically be used to generate one converged render.

So while RMSE plots are a good tool for comparing convergence of different renderers, they are time consuming to use for debugging purposes.

Pixel-wise or tile-wise signed or absolute-different images are a subjective tool for detecting problematic image regions; however, they may be unreliable due to remaining or unevenly distributed noise in unconverged images.

Predictive Rendering. In the context of predictive rendering it is especially important that a renderer is able to match the appearance of real-world objects. Therefore renderers are validated with respect to real data, such as measurements of real materials, light sources or photographs. One example is the Cornell Box [Cornell University Program of Computer Graphics 1998], a real-world model of a box, photographs of which can then be compared to the output of a renderer [Goral et al. 1984] [Meyer et al. 1986]. [Ulbricht et al. 2006] and [Drago and Myszkowski 2001] give an overview of techniques used to verify light transport compared to physical measurements for predictive rendering.

Visual Metrics. Other metrics such as [Mantiuk et al. 2011] focus on humanly perceived image quality instead of mathematical difference to a reference render or photograph. For instance, structural similarity (SSIM) [Wang et al. 2004] captures the effect on a human observer of image artifacts, such as overall bias, additional noise, or blurring, better than purely quantitative errors such as RMSE.

Statistical Methods. To our knowledge, Subr and Arvo [2007] are the only ones to employ classical statistical methods for testing renderers, aside from the MSE estimation done by [Celarek et al. 2019]. They propose using various statistical tests to analyze variances and means of renderers, including Welch's t-test to test for equal means. Their work covers the basics of statistical hypothesis-testing and also tests isolated components such as the BRDF, but Welch's t-test and its application are only briefly discussed. We provide a more in-depth explanation of the test and the steps necessary for applying it to Monte Carlo renderers, as well as a more extensive analysis, including visualization of the test results.

3. Welch's t-test

Given two sets $\{X_{1,1}, \dots, X_{1,N_1}\}$, $\{X_{2,1}, \dots, X_{2,N_2}\}$ with N_1 and N_2 individual independent random samples, where each sample set was drawn from a normal distribution with unknown individual means μ_1, μ_2 and variances σ_1, σ_2, Welch's t-test can be used to test either of the hypotheses $\mu_1 = \mu_2$ or $\mu_1 \geq \mu_2$. Unlike similar tests, it does not require the variances σ_1, σ_2 of the underlying distributions to be equal.

We use the two-tailed Welch's t-test for testing $\mu_1 = \mu_2$. First, we compute a statistic of the sample sets. If $\mu_1 = \mu_2$, that statistic is distributed according to a known two-tailed distribution. This allows us to evaluate the probability of observing the measured sample sets under the tested hypothesis $\mu_1 = \mu_2$, which then lets us draw conclusions about the hypothesis' plausibility.

Welch's t-test is based on the sample sets' properties, i.e., the *sample mean*,

$$\bar{X}_k = \frac{1}{N}\sum_N X_{k,i},$$

and the *unbiased sample variance*,

$$S_k^2 = \frac{1}{N_k - 1}\sum_{N_k}(X_{k,i} - \bar{X}_k)^2 = \frac{1}{N_k - 1}\left(\sum_{N_k}(X_{k,i})^2 - \frac{(\sum_{N_k} X_{k,i})^2}{N_k}\right),$$

for each sample set $k = 1, 2$. These properties can be combined into the t-statistic,

$$t = \frac{\bar{X}_1 - \bar{X}_2}{\sqrt{\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}}}, \tag{1}$$

which is a random variable itself and, if $\mu_1 = \mu_2$, the statistic is distributed according to the t-distribution,

$$f_\nu(t) = \frac{\Gamma\left(\frac{\nu+1}{2}\right)}{\sqrt{\nu\pi}\,\Gamma\left(\frac{\nu}{2}\right)}\left(1 + \frac{t^2}{\nu}\right)^{-\frac{\nu+1}{2}}, \tag{2}$$

with the appropriate *degrees of freedom* $\nu \in \mathbb{N}$. The Gamma function, Γ, is a generalization of the factorial. The degrees of freedom ν resulting in the t-distribution of the measured t-statistic can be approximated by the Welch-Satterthwaite equation [Satterthwaite 1946], based on the measured sample variances and sizes, as

$$\nu = \frac{\left(\frac{S_1^2}{N_1} + \frac{S_2^2}{N_2}\right)^2}{\frac{S_1^4}{N_1^2 \cdot (N_1 - 1)} + \frac{S_2^4}{N_2^2 \cdot (N_2 - 1)}}. \tag{3}$$

Figures 3 and 4 show the t-distribution for several values of ν.

3.1. Testing for Equal Means

Now we can test whether the means of the underlying distributions (not the sample means) are equal. We call this the *null hypothesis*

$$H_0 : \mu_1 = \mu_2$$

and test it against the *two-sided alternative hypothesis*

$$H_a : \mu_1 \neq \mu_2.$$

where two-sided simply means that if $\mu_1 \neq \mu_2$ we do not care which one is bigger.

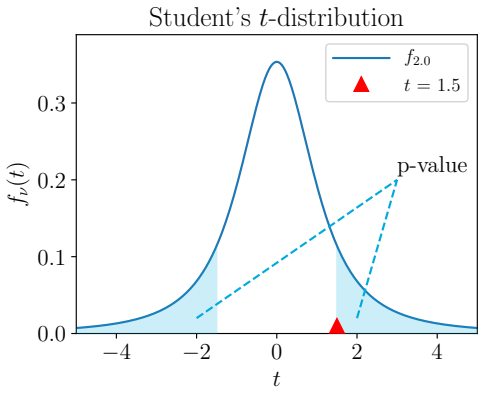


Figure 3. Two-tailed Welch's t-test where $\nu = 2$ and the t-statistic of the observed sample sets is 1.5. The p-value is the area of both tails of the t-distribution beyond t, which is ≈ 0.27.

The essential insight of Welch's t-test is that if the null hypothesis H_0 is true, the t-statistic computed from the observed sample sets (Equation (1)) approximately follows the t-distribution (Equation (2)) with degrees of freedom approximated using Equation (3). We can use this knowledge to compute the probability $p_\nu(t)$, called the p-value, of measuring the observed t-statistic or something more extreme (i.e., any t' with $|t'| \geq |t|$) for the measured t-statistic t computed from the available samples:

$$p_\nu(t) = 2 \cdot \int_{|t|}^{\infty} f_\nu(\bar{t}) d\bar{t}. \tag{4}$$

A p-value below a certain threshold (e.g., $a = 0.01$) suggests it is rather unlikely that two sample sets, drawn from normal distributions with equal means, result in the t-statistic computed from the observed sample sets, and as such, it is unlikely that the two sample sets were indeed drawn from normal distributions with equal means. In turn, this indicates H_0 is likely false, and we can reject it with a confidence of $(1 - a) \cdot 100\%$. In other words, the probability of falsely rejecting H_0 is $a \cdot 100\%$.

Visually speaking, the p-value equals two times the area of the tail of the t-distribution beyond the computed t-statistic, as illustrated in Figure 3.
To conclude, in order to conduct Welch's t-test, we need to

1. measure two normal-distributed sample sets;
2. compute their respective sample means and sample variances;
3. compute the t-statistic;
4. compute the degrees of freedom ν;
5. compute the p-value based on t and ν.

For evaluating the integral in Equation (4) we use the algorithm presented by Cooper [Cooper 1968]. If conducting a single test, a p-value below a user-defined threshold can then be used to reject H_0. However, note that Welch's t-test can only be used to detect a false hypothesis; a p-value above some threshold does not prove H_0 to be true.

In the context of rendering, we will in fact conduct separate tests per image region and color channel, and combine their results to gain more conclusive insights in Section 4.

3.2. Properties of the p-value

Since the p-value depends on t and ν, which in turn depend on the samples, the p-value is a random variable itself. As p describes the probability of observing the measured t or something more extreme given H_0, if H_0 is indeed true, p should take any value in $[0, 1]$ with equal probability. It turns out that if the null hypothesis is true (i.e., the two distributions have the same underlying mean), the p-value is actually uniformly distributed. This also implies that p is smaller than some $a \in [0, 1]$ with a probability of $a \cdot 100\%$, or $P(p < a) = a$.

Therefore we can reject H_0 if $p < a$ for some threshold a with confidence $(1 - a) \cdot 100\%$: The probability of falsely rejecting a true null hypothesis due to $p > a$ is $a \cdot 100\%$, since the probability of measuring a $p < a$ is $\int_0^a 1dx = a$.

We will use this criterion indirectly, not by conducting multiple Welch tests with different samples, but by comparing the p-values resulting from applying Welch's t-test to multiple image regions. Each image region corresponds to a different distribution. If the estimator (i.e., the renderer) is unbiased in each region, i.e., both the reference and the tested estimator have the same expected value, each region's p-value is uniformly distributed, and therefore the p-values taken from the different image regions are uniformly distributed as well. This allows us to detect a biased estimator through a non-uniform distribution of p-values taken from different image regions. Note that this line of reasoning does not work the other way around: A uniform distribution of p-values taken from different image regions does not imply that the individual p-values are uniformly distributed for their respective image regions. So while non-uniformly distributed p-values from different image regions imply bias, uniformly distributed p-values are no guarantee that the estimators are unbiased.

3.3. Gaussian Approximation

For increasing degrees of freedom ν, the t-distribution converges to the Gaussian

$$f_\infty(t) := \lim_{\nu \to \infty} f_\nu(t) = \frac{1}{\sqrt{2\pi}} e^{-\frac{t^2}{2}}.$$

Figure 4 illustrates that the difference between f_ν and f_∞ vanishes quickly, even for "smaller" values of ν (e.g., 20).

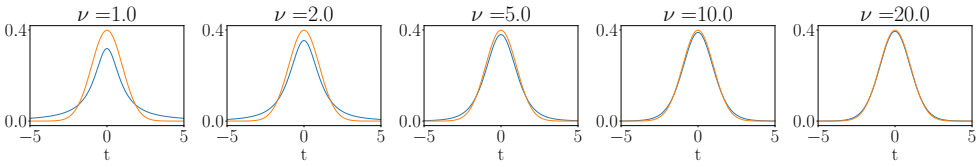


Figure 4. The t-distribution for $\nu = 1, 2, 5, 10, 20$ (blue) and the standard normal distribution f_∞ ($\mu = 0, \sigma = 1$) (orange). As $\nu \to \infty$ the t-distribution approaches f_∞.

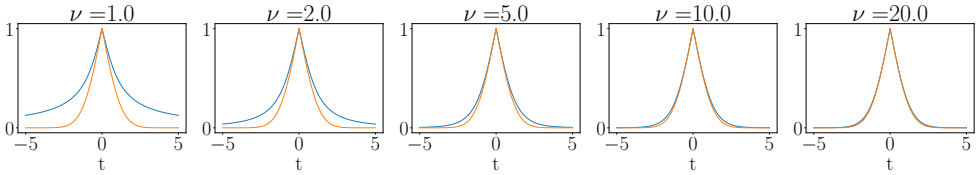


Figure 5. The true p-value p_ν for $\nu = 1, 2, 5, 10, 20$ (blue) and the approximation p_∞ based on the standard normal distribution (orange). As $\nu \to \infty$ the true p-value approaches p_∞.

As the evaluation complexity of p_ν increases with ν, we replace f_ν with f_∞ once ν is above a certain threshold and instead compute

$$p_\infty(t) := 2 \cdot \int_\infty^{-|t|} f_\infty(u)du = 1 - \text{erf}\left(\frac{t}{\sqrt{2}}\right). \tag{5}$$

Figure 5 shows that the difference between p_∞ and p_ν also vanishes quickly, motivating this replacement. Since there exists no analytical solution for the above integral we use an approximation by Abramowitz and Stegun [Abramowitz and Stegun 1964] with an error $\leq 2.5 \cdot 10^{-5}$:

$$\text{erf}(t) \approx 1 - (0.3480242x - 0.0958798x^2 + 0.7478556x^3)e^{-t^2}, \tag{6}$$
$$x := \frac{1}{1 + 0.47047t}.$$

We did not detect any improvement when using one of their more complex approximations of erf. In our experiments (Section 5.7) approximating p_ν with p_∞ whenever $\nu \geq 20$ has no noticeable impact on the test results.

4. Application to Monte Carlo Rendering

In this section we discuss how Welch's t-test can be applied to test a new renderer for unintended bias by comparing it to an unbiased reference implementation. We consider Monte Carlo renderers that solve the *path integral*,

$$I = \int_\mathcal{P} f(X)dX, \tag{7}$$

for each pixel numerically by sampling paths X_i from some probability distribution p over path space $\mathcal{P}$ and averaging their contributions $f(X_i)$ to the pixel I in the estimator

$$F_N = \frac{1}{N} \sum_N \frac{f(X_i)}{p(X_i)} \approx I. \tag{8}$$

As long as $p = 0 \Rightarrow f = 0$, this estimator is *unbiased*: The expected value of its error $\mathbb{E}[F_N - I]$ is zero for all N. In other words, the expected value of the estimator $\mathbb{E}[F] = I$. In contrast, biased estimators are functions $F'_N(X_1, \ldots, X_N)$ of a number of random samples with an expected error $\mathbb{E}[F'_N - I] \neq 0$ for some N.

Note that $\mathbb{E}[F_N]$ is the mean μ of the underlying distribution of $\frac{f(X)}{p(X)}$. So given two implementations of unbiased estimators F^1_N, F^2_N, we would like to use Welch's t-test to test whether the underlying means of the implementations are equal, which would imply the implementations do indeed represent unbiased estimators (assuming one of the implementations is a reference implementation known to be unbiased).

4.1. Generating Normal Distributed Samples

In order to conduct Welch's t-test we require two sets of normal distributed samples. However, the underlying distribution of a Monte Carlo estimator for light transport (i.e., the distribution of $\frac{f}{p}$) is typically not a normal distribution, as shown in Figure 6.

The *central limit theorem* states that the normalized sum of an increasing number of samples drawn from some distribution tends to be normal distributed. We employ this theorem by adding up multiple samples generated by the Monte Carlo estimator and using this sum as one sample for Welch's t-test.

We use the term *Monte Carlo (MC) sample* to refer to one sample generated by the renderer for one pixel. For example, considering a path tracer with next event estimation (NEE), by "one MC sample," we mean the sum of MIS-weighted contributions of one camera path and the NEE connections started from its inner vertices. In order to create one *Welch sample* X_i that can be used as input to Welch's t-test, we sum up a certain number of MC samples. If that number is large enough, the sum is

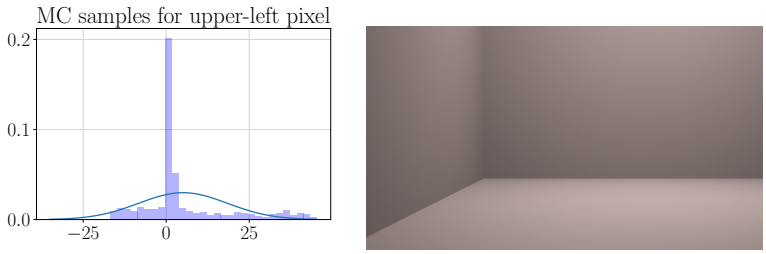


Figure 6. Histogram of individual path-traced Monte Carlo samples for the red channel of the upper-left pixel in the image on the right. The blue line represents a fitted normal distribution.

roughly normal distributed.

More specifically, we conduct Welch's t-test separately for disjoint tiles of 32×32 pixels. One Welch sample is formed by summing up one MC sample from each pixel within that tile. This corresponds to stratified sampling of the underlying distribution of that pixel tile (by stratifying MC samples over pixels), and summing them up. For most test scenes used in our evaluation, summing up 1024 MC samples to get one Welch sample does indeed result in a normal distribution of Welch samples within each pixel tile, as we evaluate empirically in Section 5.1. Since we always sum up the same number of samples (1024) to get one Welch sample, we can skip the normalization (dividing the MC sample sum by 1024). If the image resolution is not a multiple of 32×32, we discard incomplete tiles. If a light tracer is used, one needs to explicitly count the number of samples splatted into each tile per iteration and perform the normalization accordingly.

It should be noted that the central limit theorem does not make any guarantees about a finite sum of samples approaching a normal distribution. So for each scene and algorithm, before applying Welch's t-test, one should evaluate the number of MC samples needed to (roughly) achieve a normal distribution of Welch samples. In Section 5.1 we present one example where this is not the case.

4.2. Sample Preparation During Rendering

Since the renderer continues generating more samples per pixel, we can prepare sample statistics during rendering to use for Welch's t-test. For computing the t-statistic (Equation (1)) and the appropriate degrees of freedom ν (Equation (3)), we need the sample mean and sample variance of the Welch samples. Therefore, the renderer needs to be able to create individual Welch samples by summing up one MC sample from each pixel within a pixel tile, in order to compute the sum of squares of Welch samples $\sum_N X_i^2$. We require two additional frame buffers with $1/32^2$ the resolution of the actual image: one for storing the sum of Welch samples for each pixel tile and one for storing the sum of squared Welch samples. We compute and store the sum of Welch samples and their squares for each color channel separately.

4.3. Considerations When Using Halton Points

In our implementation, when using random numbers, the renderer iterates over all pixels, places one random sample in each, and repeats. This means that for an image with N pixels, after placing N samples each pixel receives exactly one sample, and therefore each pixel tile receives 32^2 samples. We also support rendering with Halton points: in that case, pixel samples are not placed one for each pixel, but instead where the first two dimensions of the Halton points fall in the image plane: $(\Phi_2(i), \Phi_3(i))$, i.e., the radical inverse Φ with basis 2 and 3. We know that every 2×3 block will have received six samples after drawing N samples; however some pixels will have received more than one while others received none. This also means that in one

iteration a 32×32 pixel tile may not always receive 32^2 samples.

The central limit theorem works on the normalized sum of samples, which means we would have to normalize the sum by the actual sample count, which is cumbersome. To arrive at a well-defined sample-count per pixel, we make use of the elementary interval property of Halton points, which guarantees that after placing $3 \cdot N$ samples, each pixel received three samples and therefore each tile received $3 \cdot 32^2$ samples. We can then simply sum up those samples and skip normalizing. This issue can also be avoided by stamping repeated replications of the sample points for each tile.

Using quasi-Monte Carlo point sets introduces correlation which may yield inconclusive results when comparing two renders. To avoid this, we only compare renders with Halton points to ones with independent random points. Section 5.8 analyzes this.

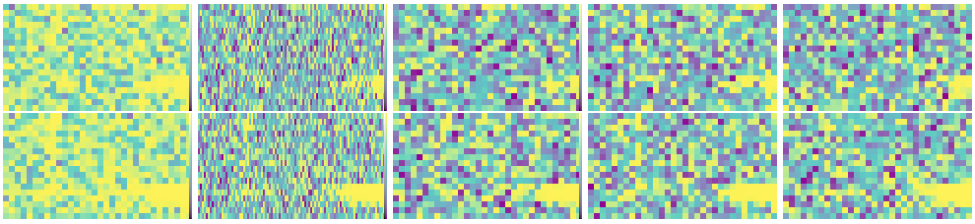
4.4. Visualization

We propose two visualizations for interpreting the results. In both cases we visualize the p-value, as it is bounded in $[0, 1]$, as opposed to the t-statistic which can take on arbitrary values.

The p-value color map. Since we conduct Welch's t-test separately for many image regions (i.e., square tiles of pixels), we can visualize the p-value obtained for each region in a color map. This scheme allows us to localize problematic regions in the image that have especially high bias. While this could also be achieved using difference images, Welch's t-test is both statistically meaningful and more expressive at low sample counts, as shown in Section 5.3. Since we get three p-values for each image region (one per color channel), we try visualizing them combined and separately in Figure 7. A color-mapping scheme maps large p-values to violet and small ones to yellow, as indicated to the right of each image and enlarged in Figure 8.

Visualizing only the smallest p-value per square pixel tile (Figure 7(a)), while skewing the image towards colors representing small values, seems easier to parse visually thanks to the square shape of individual p-value color tiles and combines all values in a single image. Visualizing all three p-values requires us to use oblong blocks to squeeze three p-values into one square pixel tile (Figure 7(b)), or to output three separate images (Figure 7(c)–7(e)). The former results in a cluttered image while the latter requires us to look at three separate images to evaluate the results. One advantage of the last visualization is that it reveals whether clusters of small p-values just happened at random, or if they are symptomatic of some intrinsic problem, i.e., present in all three color channels. In general, we find the visualization of the smallest p-value easiest to use. In the example in Figure 7, it best reveals the bias along the left corners of the scene.

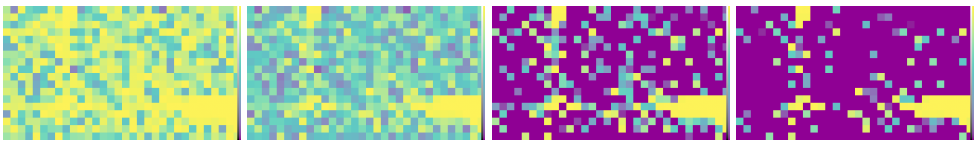
For focusing a higher color resolution on small p-values corresponding to poten-



(a) smallest (b) all values (c) red channel (d) green channel (e) blue channel

Figure 7. Color map for Welch's t-test applied to 32×32 pixel tiles of an unbiased PT render and a biased BDPT (as in Section 5.3) render of the corner scene (Figure 11 left) with 10 spp (top) and 60 spp (bottom), visualizing the smallest (a), all three (b) and individual p-values (c,d,e) per color channel.

Figure 8. Color scale for $p = 0$ (left) to $p = 1$ (right).



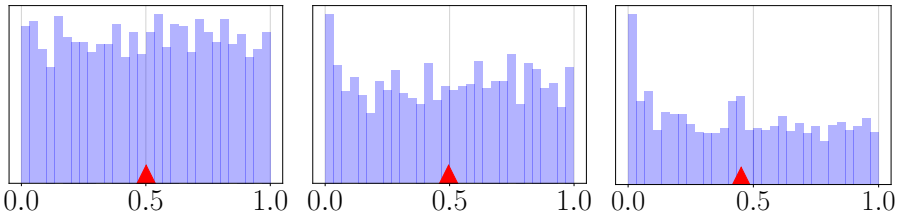
(a) original (b) square root (c) multiplied by 10 (d) multiplied by 20

Figure 9. Different p-value mappings (left: same as Figure 7).

tially biased regions, Figure 9 visualizes the square root or a multiple of the p-value.

Histogram of p-values. Whenever the null hypothesis (two compared unbiased renderers) is true, the p-value, which is a random variable itself, is uniformly distributed (see Section 3.2). Since we conduct only one test per color channel and image region, each corresponding to its own distribution, we cannot directly check whether the p-value obtained for one image region is uniformly distributed. However, we can at least check whether the p-values taken from different regions are uniformly distributed, based on a histogram of all p-values, as well as their mean. Note that a uniform histogram is no proof that all individual p-values stem from uniform distributions. However, a non-uniform histogram is a strong indicator that the individual p-values cannot all be uniformly distributed and thus the tested renderer is likely biased. Three examples are shown in Figure 10.

In the supplemental materials, available at `jcgt.org/published/0009/02/01/code.zip`, we provide code for computing the color map (*welch.c*) and histogram (*pvalhist.py*), given two images and the relevant statistics that need to have been computed at render time.



(a) unbiased PT (b) biased BDPT, 10 spp (c) biased BDPT, 60 spp

Figure 10. Histogram of p-values for an unbiased render (a) and two biased renders (biased BDPT as in Section 5.3 with 10 (b) and 60 (c) samples per pixel (spp), as in Figure 7) tested against an unbiased reference implementation. At 10 spp the p-value mean (red triangle) is indistinguishable from 0.5, but the $[0, 1/30]$-bin contains way more p-values than would be expected if all p-values were uniformly distributed, indicating bias.

5. Results

We evaluate Welch's test for two scenes, a simple scene showing the lower-left corner of an empty Cornell box with a constant square light source (Figure 11 (left)), and the dining scene (Figure 11 (right)). The first scene is helpful in detecting errors in the transport algorithm itself, e.g., problems with the camera or missing cosines. The second scene illustrates the test's behavior under more complex conditions. We use a path tracer with next-event estimation (PT) and a bidirectional path tracer (BDPT) of our custom spectral renderer.

Unless stated otherwise, the renderers draw their random numbers from a pseudo-random number generator, and Welch samples are constructed within 32×32 pixel tiles by summing up one sample from each pixel within the tile.

Figure 11. Left: Corner scene. Right: Dining scene.

5.1. Welch Sample Generation

Before applying Welch's t-test, we need to make sure that the number of MC samples contributing to one Welch sample is sufficient for (approximating) a normal distribution of Welch samples, as described in Section 4.1. We therefore generate histograms of the Welch samples created for individual tiles within the two images in Figure 12.

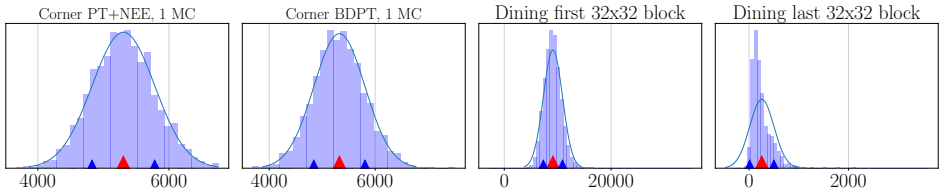


Figure 12. Histogram of Welch samples generated by summing up one MC sample per pixel for the red channel of a 32×32 pixel tile. (a), (b): Upper-left pixel tile in Figure 11; (c) upper-left pixel tile in the dining scene rendered with PT; (d) lower-right pixel tile in the dining scene (PT). MC samples are based on Mersenne random numbers. The red and blue markers represent the sample mean and variance, respectively; the blue line is a fitted normal distribution using those as mean and variance.

For simple setups, such as the corner scene or the directly lit upper wall in the dining scene, summing up 1024 MC samples per pixel tile creates normal distributed samples for both the path tracer and the bidirectional path tracer. However, in more difficult setups, such as the caustic on the table or the shadowed lower-right part of the wall, Welch samples created that way are not as close to a normal distribution. It turns out that Welch's t-test still produces reasonable results for the dining scene with Welch samples created that way, but in general one should be cautious about the number of MC samples contributing to one Welch sample.

Failure case. Figure 13 shows the corner scene with a glass sphere added in the center. The resulting caustic noise causes the Welch samples in the respective image regions to not be normal distributed anymore, even for larger tiles such as 64×64 pixels as shown in Figure 18(c). Therefore Welch's t-test should not be used with this scene, as it produces a far from uniform histogram of p-values even when testing the reference implementation against itself with different seeds for the random number generator. As a side note, if a renderer produces a Welch sample distribution such

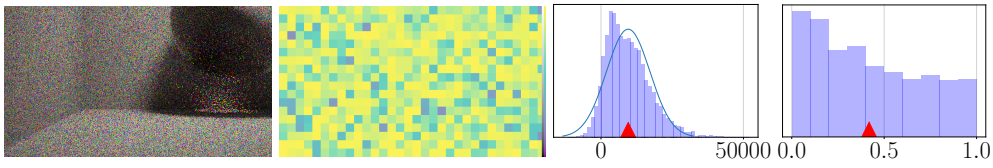


Figure 13. The corner scene with a glass sphere no longer has normal distributed Welch samples when rendered by a path tracer, not even when summing up 64×64 pixel samples as in this example (c). Thus, when testing the unbiased reference against itself (shown here for 10 spp) Welch's t-test does not produce a uniform histogram of p-values (d) anymore.

as this one, one might reconsider using that algorithm for the given scene, since its convergence will be rather slow anyway.

5.2. Unbiased Path Tracer Tested Against Itself

We start with the test results for a correct path tracer with next event estimation tested against itself in the corner scene (Figure 11 (left)) with different random seeds. This serves as a reference for what the results should look like for unbiased renders. Figure 14 shows the color map and p-value histogram for testing 10 and 1000 spp against 10 and 1000 spp renders.

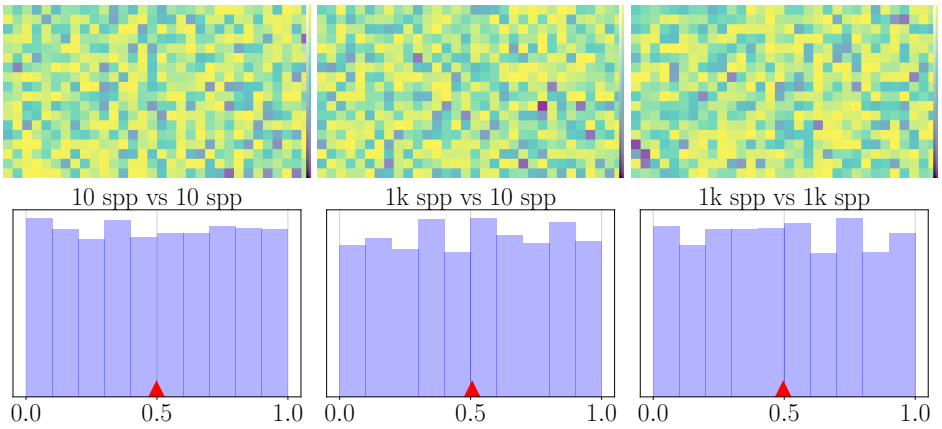
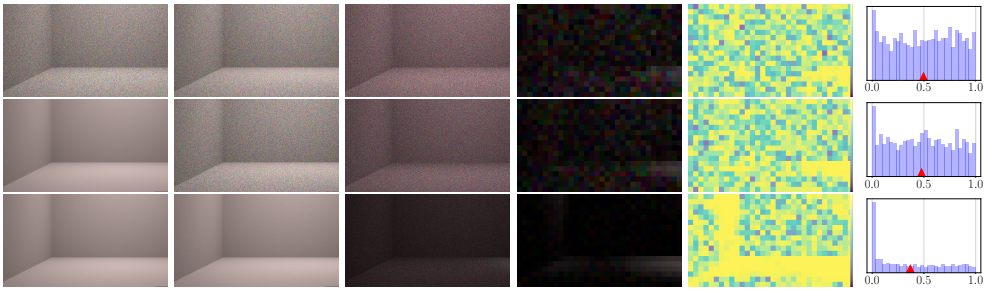


Figure 14. Test results of an unbiased path tracer against itself for the corner scene (Figure 11 (left)). From left to right: 10 spp vs 10 spp, 10 spp vs 1000 spp, 1000 spp vs 1000 spp. Top: Color map for smallest p-value. Bottom: Histogram (blue) and mean (red) of p-values from all color channels and pixel tiles.

5.3. Finding Bias in a Bidirectional Path Tracer

We used Welch's t-test to trace down a bug in our bidirectional path tracer (BDPT). In this version, the light tracer occasionally replaced the camera normal with the connecting segment's direction, leading to an incorrect cosine evaluation between the segment's direction and camera normal. This affected the light tracer's contribution as well as the MIS-weight computation. The resulting error was greater towards the borders of the image, where the correct cosine is much smaller than the cosine that was computed (which was always one).

Figure 15 shows the PT reference and the biased BDPT renders, both with 3-vertex paths only, as well as the results of Welch's t-test and difference images. The clumped yellow regions in the color maps are strong evidence that the two renderers do not converge to the same mean, i.e., pixel color, at least not in those image regions. Similarly, the non-uniform histogram indicates that not all, maybe even none, of the p-values stem from a uniform distribution, and thus one of the renderers is biased.



(a) PT (b) BDPT (c) Difference (d) Block diff (e) Color map (f) p-values

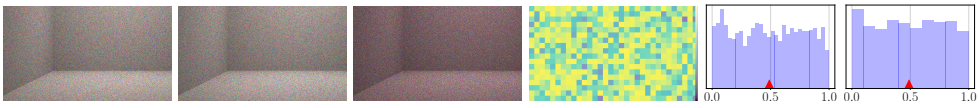
Figure 15. Top: PT (10 spp) (a) vs biased BDPT (10 spp) (b). Differences (c, d) are absolute. Center: PT (1k spp) vs biased BDPT (10 spp) shows that a more converged reference produces a visually similar color map, but a more conclusive p-value histogram. Bottom: PT (1k spp) vs biased BDPT (500 spp)—only now do we start to get a meaningful difference image (c, d), revealing a subtle difference in brightness at the right part of the floor.

In this case, using more samples in the biased renderer results in even more significant evidence (more low p-values, manifesting as clumped yellow regions and a non-uniform histogram) against the hypothesis that the renderer is unbiased.

For a fair comparison to what insight can be gained from difference images, we also include a tile-wise difference image, such that one difference value can benefit from the same number of MC samples as one t-test.

5.4. Detecting a Biased Scene

In Figure 16 we show two renders of the box scene created by the unbiased PT with 10 spp. In Figure 16(a), the walls and floor have the diffuse color (0.2, 0.2, 0.2) while in Figure 16(b), they have color (0.21, 0.21, 0.21). In this case, the color map and difference image are useless, the latter being dominated by residual noise. Yet, the p-value histogram reveals a tendency towards small values, indicating that the two renders will not converge to the same result. With fewer bins (Figure 16(f)), this is even more noticeable.



(a) Gray=0.2 (b) Gray=0.21 (c) Difference (d) Color map (e) p-values (f) p-values

Figure 16. Corner scene with gray $= 0.2$ (a) and gray $= 0.21$ (b), PT, 10 spp. The absolute difference (c) is dominated by noise and at this sample count is indistinguishable from a difference image of two correct renders with different seeds. In this case the histogram of p-values with fewer bins (f) is most helpful at detecting bias.

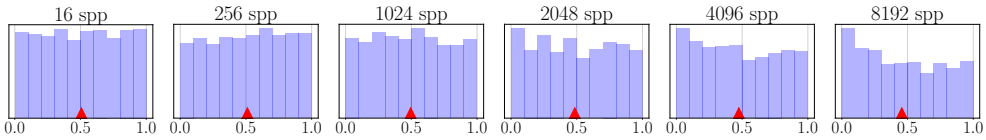
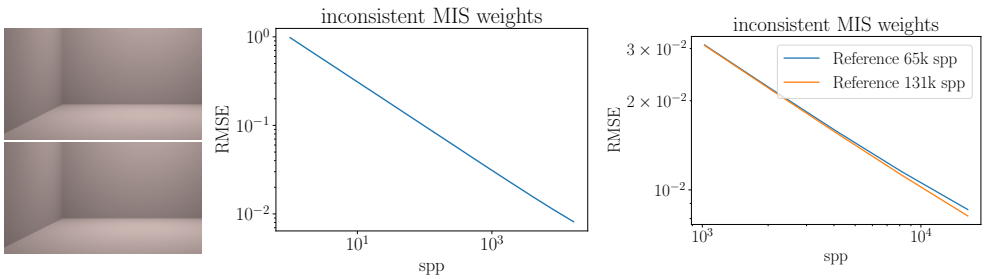


Figure 17. p-values for the path tracer with biased MIS weights. Welch's t-test only detects the bias reliably at several thousand samples per pixel, where it produces the non-uniform histograms.

5.5. Biased MIS Weights in a Path Tracer

Figure 17 shows Welch's t-test for a path tracer with incorrect multiple-importance-sampling (MIS) weights, where, for paths finding the light source by accident, the probability density function of the last scattering event is ignored in the MIS weight computation. This results in a slightly brighter image, which is hardly noticeable by eye (see Figure 18(a)). In this case, Welch's t-test does not pick up on the error at 10 spp, but is able to detect it at much higher sample counts. Since the difference image and color map look similarly unhelpful, as in Figure 16, we only show the p-value histograms at different sample counts in Figure 17.

Since in this case Welch's t-test requires large sample counts, we also compare to the RMSE plot in Figure 18. We only plot the RMSE up to 8k samples, since our reference has 131k samples and plotting for larger sample counts would result in a curve caused by the unconverged reference. For instance, when using a 65k reference instead, this effect is already noticeable at a few thousand samples per pixel (Figure 18(c)). With the 131k spp reference, the RMSE plot is a straight line and does not reveal any error. Computing a reference with even more samples to eventually detect a curving RMSE line would be possible, but highly impractical.



(a) PT (b) RMSE (c) RMSE (zoomed)

Figure 18. The corner scene rendered with a path tracer at 131072 (2^{17}) spp. (a) reference PT (top) and biased PT due to inconsistent MIS weights (bottom). The RMSE plot (b) looks like a straight line and thus does not reveal any bias. With this reference, any curve occurring at higher sample counts might as well be caused by the unconverged reference, as is already the case when plotting the RMSE with a 65k spp reference ((c), zoomed in).

5.6. Influence of the Pixel Tile Size

Figures 19 and 20 show the test's result if we use individual MC samples as Welch samples and conduct Welch's t-test per pixel instead of per pixel tile. Figure 19 tests the unbiased PT reference against itself, where the test produces a false reject (a non-uniform histogram of p-values) for the dining scene. Figure 20 tests the unbiased PT against the biased BDPT described in Section 5.3 for the corner scene. Here, the test does not detect bias at 60 spp, and instead outputs a uniform color map and histogram of p-values, producing a false positive. We conclude that Welch's t-test is unreliable both in detecting bias and in not-rejecting unbiased renderers when using MC samples directly as Welch samples.

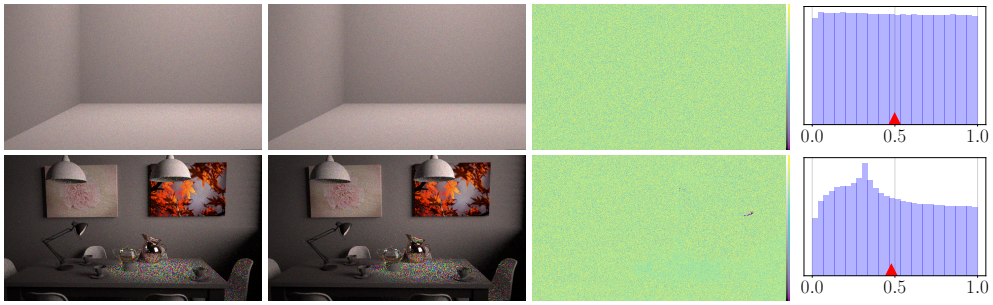


Figure 19. Testing the unbiased PT against itself with different random seeds and 100 spp each for two different scenes. Welch's t-test is conducted per pixel, and individual MC samples are used as input samples to the test.

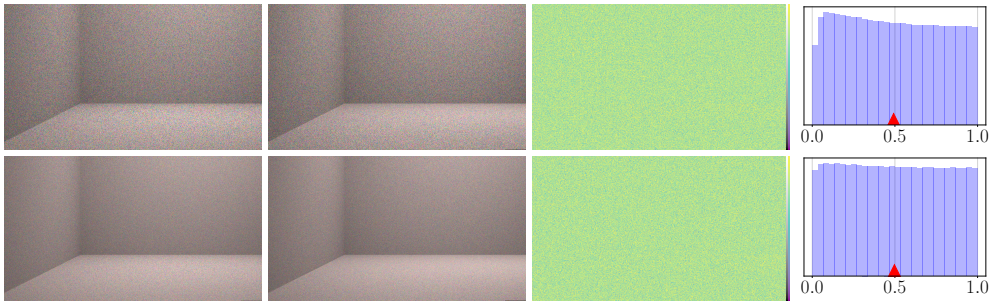


Figure 20. Testing the unbiased PT (left) against a biased (Section 5.3) BDPT (center left), resulting color map for smallest p-value (center right) and respective p-value distributions (right). Welch's t-test is conducted per pixel, and individual MC samples are used as input samples to the test. Top: 20 spp. Bottom: 60 spp.

5.7. Gauss Approximation

Figure 21 shows the range of appropriate degrees of freedom ν (Equation (3)) over all image regions when comparing different sample counts for the corner and dining

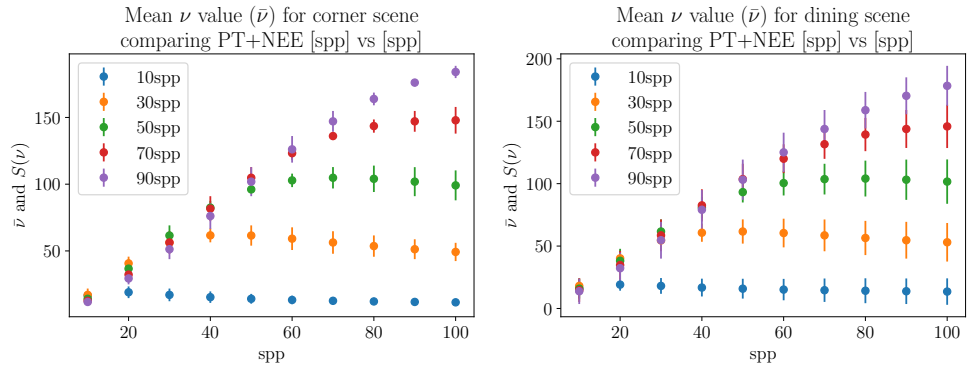


Figure 21. Mean ($\bar{\nu}$) and standard deviation ($S(\nu)$) of ν-values over all pixel tiles when comparing our PT against itself for different sample counts for the corner (left) and dining scene (right), with 1 sample per pixel per 32×32 tile per Welch sample. The ν-value seems to depend mostly on the lower sample count and less on the scene.

scene. Starting from a few dozen Welch samples, we consistently get ν values such that the normal distribution approximates the actual t-distribution quite well over the entire image. With fewer Welch samples, this is only the case in some regions, while other regions have low ν-values, as each region corresponds to its own distribution.

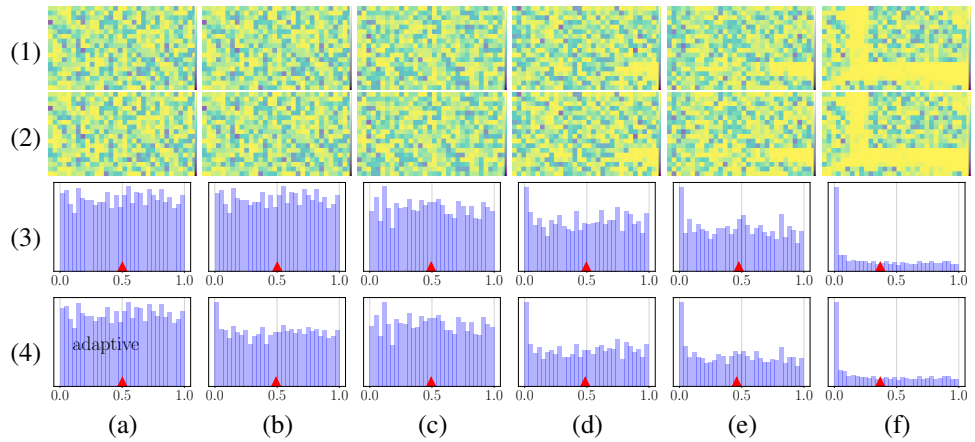


Figure 22. Welch's t-test for the corner scene with the correct t-distribution ((1),(3)), compared to the Gaussian approximation as in Equation (5) ((2),(4)). In (a) the Gauss approximation is applied wherever $\nu \geq 20$, in (b)–(f) it is applied to all image regions and color channels. (a), (b) PT (10 spp) vs PT (10 spp) ($\bar{\nu} \approx 20$); (c) PT (10 spp) vs PT (1k spp) ($\bar{\nu} \approx 2000$); (d) PT (10 spp) vs biased BDPT (10 spp) ($\bar{\nu} \approx 20$); (e) PT (1k spp) vs biased BDPT (10 spp) ($\bar{\nu} \approx 10$); (f) PT (1k spp) vs biased BDPT (500 spp) ($\bar{\nu} \approx 1000$). While the color maps seem indistinguishable, the histogram of Gauss-approximated p-values for (b) ($\bar{\nu} \approx 20$) falsely rejects the hypothesis of the renderers being unbiased. Selectively applying the Gauss approximation in tiles and color channels with $\nu \geq 20$ fixes this (a).

Figure 22 shows p-values p_ν computed from f_ν with the correct degrees of freedom ν, as well as p_∞ based on the approximated (see Equations (5) and (7)) normal distribution f_∞. If the number of samples is not sufficient (e.g., $N_1 = 10$, second column) and the appropriate ν is too small, the approximated p-values are not uniformly distributed, leading to false negatives (Welch's t-test "finding" bias when comparing two unbiased renderers). This is to be expected, as the true p-value is larger than the Gauss-approximated p-value (see Figure 5 left), and thus the Gauss approximation results in more p-values close to zero than a uniform distribution should have. This also explains why the Gauss approximation only creates more false negatives, but not more false positives than p-values based on the true f_ν.

Selectively applying the Gauss approximation only in image regions and color channels where $\nu \geq 20$ fixes this problem, as shown in the left column in Figure 22. Here ($N_1 = 10$), the value of ν still varies over image regions and channels. For many the approximation is valid, but the few where it is not, due to low ν-values, it results in a non-uniform histogram when using the approximation.

5.8. Halton Points

In Section 4.3, we noted that to assure an equal number of MC samples are summed up for each Welch sample, we need to sum up a multiple of three samples per pixel if using Halton random numbers to place MC samples in the pixel plane. Figure 23 shows a histogram of Welch samples constructed this way on the right, and a histogram of Welch samples created from *on average* 32^2 MC samples per tile on the left. At least in this example, both approaches yield similarly distributed Welch samples.

We also show the test's results based on such samples, compared against a reference based on random numbers from a Mersenne twister, of the unbiased PT in Figure 24 and of the biased BDPT from Section 5.3 in Figure 25. Note that we did not properly normalize the Welch samples according to the number of MC samples that were actually used to create them, and we assume a constant normalization instead. Yet, Welch's t-test seems to work regardless of the Welch sample distribution

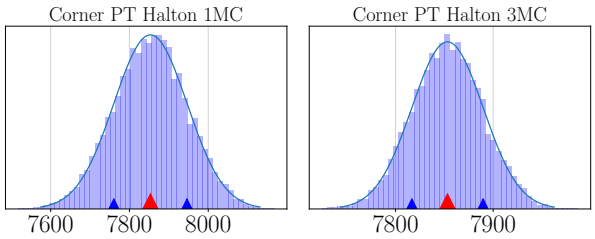


Figure 23. 10k Welch samples for the lower right 32×32 pixel tile in the corner scene for a path tracer based on Halton points. Left: on average one MC sample per pixel summed up for one Welch sample. Right: three MC samples per pixel per Welch sample.

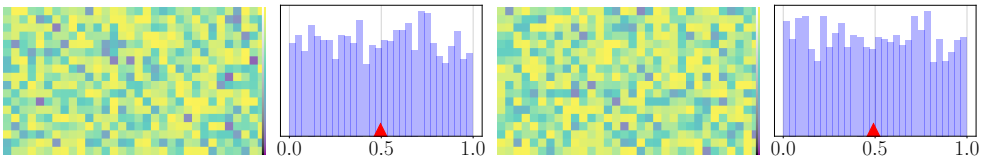


Figure 24. Testing the unbiased PT based on Halton points in the corner scene against a PT based on Mersenne random numbers. Left: on average one MC sample per pixel summed up for one Welch sample, 10 spp. Right: three MC samples per pixel per Welch sample, 30 spp. Both yield fairly uniform histograms.

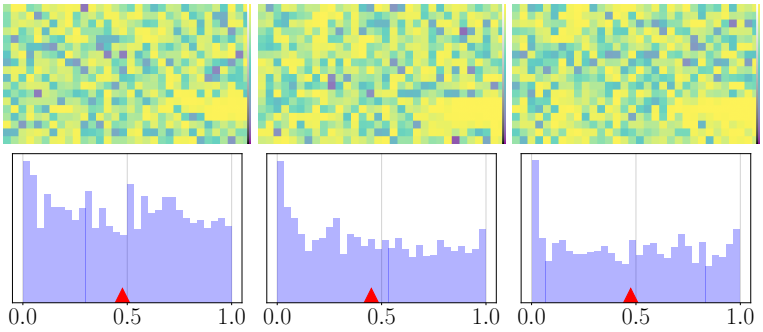


Figure 25. Testing the biased BDPT based on Halton points in the corner scene. Left: on average one MC sample per pixel summed up for one Welch sample, 10 spp. Center: three MC samples per pixel per Welch sample, 30 spp. Both cases have 10 Welch samples available per test; if we sum up on average one MC sample per pixel but use 30 spp (right), the test achieves similar, if not slightly better, quality.

less resembling a normal distribution than when using three MC samples per pixel per Welch sample, and the tests yield qualitatively similar results.

Figure 26 reveals another issue: Due to the correlations in the Halton dimensions, i.e., those used to place MC samples in the pixel plane, testing two renders, both of which are based on Halton points, yields useless results. When testing a Halton-based unbiased BDPT against either unbiased Halton based PT or BDPT, the corner regions look at least a bit as expected, likely due to the larger influence of the light tracer which does not depend on an explicit sample in the pixel plane.

5.9. Run Time and Memory Consumption

It took approximately three seconds on an Intel i7-6700 to run 110 Welch tests with $N_1, N_2 \in [10, 100]$, half of which used the Gaussian approximation, with an image resolution of 1024×576 and a tile size of 32×32. Therefore, we did not consider further optimizations or faster approximations, especially, since in a typical use case, one would conduct only one Welch test at a time.

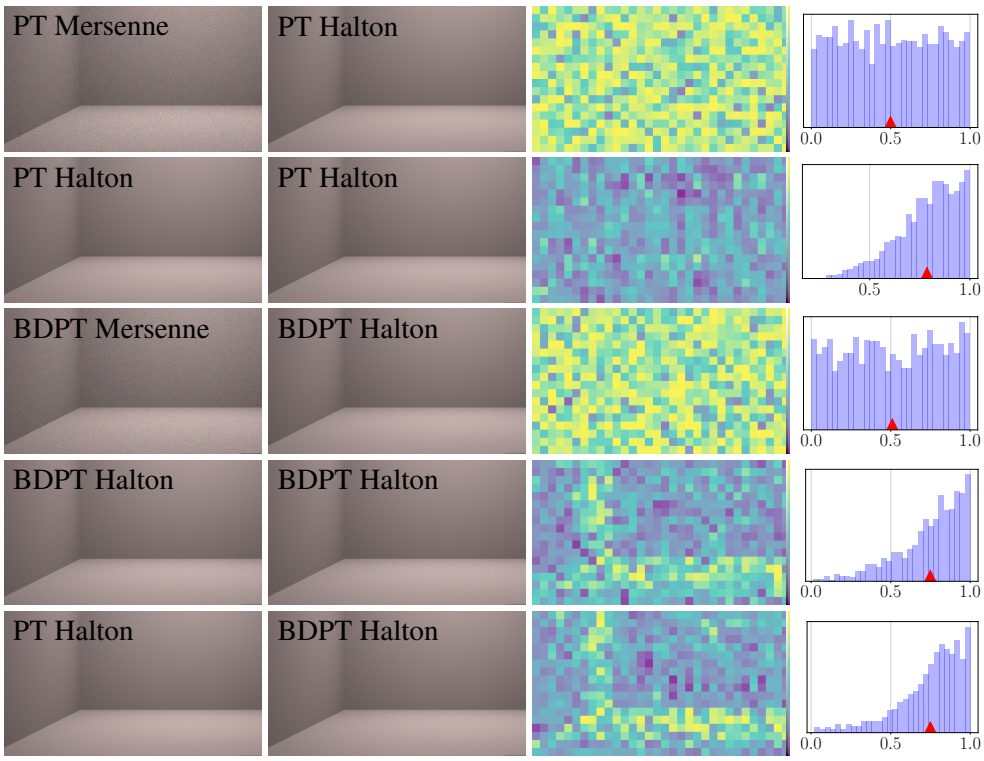


Figure 26. Unbiased PT and unbiased BDPT in the corner scene. Each image has 60 MC samples per pixel and different random seeds, the test works on 20 Welch samples with 3 MC samples per pixel per 32×32 tile. Whenever both renders are based on Halton points, Welch's t-test produces misleading results.

As for memory consumption, we store two additional frame buffers during rendering with $1/32^2$ the resolution of the original frame buffer. With sample set sizes $N_{1,2}$ up to the order of 10^3, we found 32-bit floating point numbers to be sufficiently precise.

6. Conclusion

We showed how to apply Welch's t-test to test supposedly unbiased renderers for bias compared to a reference implementation. Welch's t-test can help locate faulty image regions even at low sample counts and detects bias much earlier than RMSE or difference-image tests. Our color visualization proved useful at identifying particularly biased image regions, while at very low sample counts, the p-value histogram revealed errors that were not yet obvious in the color map, a difference image or RMSE plot.

Our results suggest that Welch's t-test is more reliable if it is applied to (approximately) normal distributed samples. Therefore, before using Welch's t-test, de-

pending on the scene, one should make sure to use an appropriate pixel tile size and sample-per-pixel count within tiles to ensure at least roughly normal distributed Welch samples. When in doubt, it may help to first test the reference against itself, and if the test fails to produce a uniform histogram, fall back on some other method.

Limitations. The Welch test only tells us whether two distributions might have the same mean. Therefore, we can only use it to detect incorrect rendering algorithms when compared to a reference implementation which is known to work. We cannot use it to assess the asymptotic error behavior, i.e., we cannot use it to tell which of two algorithms converges faster.

Large p-values are no proof that two renderers converge to the same image, neither is a uniform histogram of p-values; we can only use a large number of pixel tiles with small p-values (e.g., < 0.01) as strong evidence that they do not. Since we apply Welch's t-test to many image tiles at once, we can use all the individual results at once and thus, at least, decrease the chance of Welch's t-test not finding existing bias.

Although it did work in most of our test cases, due to the numerical nature of the problem, there is no guarantee that Welch's t-test always detects existing bias at a low sample count. In one example of particularly small bias, the test only detected bias at a few thousand samples per pixel, which however was still less than what was required to detect the bias using the RMSE plot.

References

ABRAMOWITZ, M., AND STEGUN, I. A. 1964. *Handbook of Mathematical Functions with Formulas, Graphs, and Mathematical Tables*. Dover, New York. Ninth printing, tenth GPO printing. 8

CELAREK, A., JAKOB, W., WIMMER, M., AND LEHTINEN, J. 2019. Quantifying the error of light transport algorithms. *Computer Graphics Forum 38*, 4 (July), 111–121. URL: https://www.cg.tuwien.ac.at/research/publications/2019/celarek_adam-2019-qelta/, doi:10.1111/cgf.13775. 3, 4

COOPER, B. E. 1968. Algorithm as 3: The integral of student's t-distribution. *Journal of the Royal Statistical Society. Series C (Applied Statistics) 17*, 2, 189–190. URL: http://www.jstor.org/stable/2985684. 7

CORNELL UNIVERSITY PROGRAM OF COMPUTER GRAPHICS, 1998. Cornell box. URL: http://www.graphics.cornell.edu/online/box/. 4

DRAGO, F., AND MYSZKOWSKI, K. 2001. Validation proposal for global illumination and rendering techniques. *Computers & Graphics 25*, 3, 511 – 518. URL: http://www.sciencedirect.com/science/article/pii/S0097849301000723. 4

GORAL, C. M., TORRANCE, K. E., GREENBERG, D. P., AND BATTAILE, B. 1984. Modeling the interaction of light between diffuse surfaces. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, ACM, New York,

NY, USA, SIGGRAPH '84, 213–222. URL: http://doi.acm.org/10.1145/800031.808601. 4

MANTIUK, R., KIM, K. J., REMPEL, A. G., AND HEIDRICH, W. 2011. Hdr-vdp-2: A calibrated visual metric for visibility and quality predictions in all luminance conditions. *ACM Trans. Graph. 30*, 4 (July), 40:1–40:14. URL: http://doi.acm.org/10.1145/2010324.1964935. 4

MEYER, G. W., RUSHMEIER, H. E., COHEN, M. F., GREENBERG, D. P., AND TORRANCE, K. E. 1986. An experimental evaluation of computer graphics imagery. *ACM Trans. Graph. 5*, 1 (Jan.), 30–50. URL: http://doi.acm.org/10.1145/7529.7920. 4

SATTERTHWAITE, F. E. 1946. An approximate distribution of estimates of variance components. *Biometrics Bulletin 2*, 6, 110–114. URL: http://www.jstor.org/stable/3002019. 5

SUBR, K., AND ARVO, J. 2007. Statistical hypothesis testing for assessing Monte Carlo estimators: Applications to image synthesis. In *15th Pacific Conference on Computer Graphics and Applications (PG'07)*, IEEE, Los Alamitos, CA, 106–115. URL: https://ieeexplore.ieee.org/document/4392721, doi:10.1109/PG.2007.55. 4

ULBRICHT, C., WILKIE, A., AND PURGATHOFER, W. 2006. Verification of physically based rendering algorithms. *Computer Graphics Forum 25*, 2, 237–255. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/j.1467-8659.2006.00938.x. 4

WANG, Z., BOVIK, A., RAHIM SHEIKH, H., AND SIMONCELLI, E. 2004. Image quality assessment: From error visibility to structural similarity. *IEEE Transactions on Image Processing 13*, 4 (05), 600–612. URL: https://ieeexplore.ieee.org/document/1284395. 4

Author Contact Information

Alisa Jung
Karlsruhe Institute of Technology
Am Fasanengarten 5
76131 Karlsruhe
alisa.jung@kit.edu

Johannes Hanika
Karlsruhe Institute of Technology
Am Fasanengarten 5
76131 Karlsruhe
hanika@kit.edu

Carsten Dachsbacher
Karlsruhe Institute of Technology
Am Fasanengarten 5
76131 Karlsruhe
dachsbacher@kit.edu

A. Jung, J. Hanika, C. Dachsbacher, Detecting Bias in Monte Carlo Renderers Using Welch's t-test, *Journal of Computer Graphics Techniques (JCGT)*, vol. 9, no. 2, 1–25, 2020
http://jcgt.org/published/0009/02/01/

Received:	2019-10-29		
Recommended:	2020-01-14	Corresponding Editor:	Jarosz Wojciech
Published:	2020-06-13	Editor-in-Chief:	Marc Olano