Improved quantum supersampling for quantum ray tracing

Xi Lu · Hongwei Lin

Abstract
Ray tracing algorithm is a category of rendering algorithms that calculate pixel colors by simulating light rays in parallel. The quantum supersampling has achieved a quadratic speedup over classical Monte Carlo method, but its output image contains many detached abnormal noisy dots. In this paper, we improve quantum supersampling by replacing the QFT-based phase estimation in quantum supersampling with a robust quantum counting scheme. We do simulation experiments to show that the quantum ray tracing with improved quantum supersampling does perform better than classical path tracing algorithm as well as the original form of quantum supersampling.

Keywords Quantum algorithm · Computer graphics · Ray tracing

1 Introduction
Ray tracing [1–4] is a general term of rendering algorithms that calculate pixel colors by simulating all the physical interactions between light rays and the scene. Since a single ray scatters towards many directions when interacting with an object, and each of the scattered rays scatters towards more directions when interacting with other objects, the total number of rays grow exponentially in the number of interactions. A standard solution to this problem is Monte Carlo ray tracing, or path tracing [3], which randomly shoots only one ray at each bounce. To render an image with high quality, path tracing algorithm require many rays to reduce noise. In many situations people have to make a trade-off between time cost and quality. For real-time ray tracing applications where the rendering time is strictly limited, the state-of-the-art GPU can only handle sampling a small amount of rays per pixel, and the quality of result relies
Classical path tracing only traces one ray at a time, while quantum ray tracing can trace numerous rays as a superposition in one shot. Therefore, a major problem in ray tracing is to reduce the time cost while maintaining the quality.

Quantum computing is an emerging subject that studies how to perform computational tasks in quantum mechanical systems. By leveraging the superposition and entanglement of quantum computing, quantum computing has inherent advantages on parallel computational tasks. As a result, quantum computing shows its computational power by providing spectacular speedup over classical computing in some problems.

The idea of introducing quantum computing into computer graphics was early proposed in 2005, which raises many concepts such as quantum Z-buffer, ray tracing and radiosity algorithm. Later, Caraiman introduced quantum solutions for the polygon visibility and global illumination problems and developed the appropriate quantum algorithms. In both articles, quantum speedup comes from the superposition of all scene primitives. Johnson proposed quantum supersampling as the quantum variant of Monte Carlo integration in ray tracing, and did simulation experiments on binary image filtering to show that quantum supersampling can reduce mean pixel error faster. Shimada et al. also did experiments on binary image filtering using quantum coin (QCoin) method originally proposed in [15]. Recently Santos et al. investigated on using quantum computing for intersection searching subroutine in ray tracing, which provides a quadratic complexity in scene complexity.

There are generally two potential approaches for quantum speedup in ray tracing algorithm. One is to store the scene primitives in a superposition and apply quantum minimum search to find the nearest intersection between each ray and the scene, to obtain a quadratic speedup, as well studied in [16]. This approach requires to frequently read the scene primitives from classical memory and then superpose them in quantum memory, but the required qRAM is still not available. The other is to store the ray paths in a superposition, as illustrated in Fig. 1, such that an astronomical amount
of rays, at the cost of a logarithmic space and time, can be traced simultaneously to reduce error. Plus, we do not have the same worry as the first approach, since the rays are procedurally generated instead of reading from classical memory. This approach is used in quantum supersampling and quantum coin method.

As is shown in [13], the image produced by quantum supersampling (QSS) contains many detached noisy dots. In this paper, we improve quantum supersampling for quantum ray tracing, by replacing the standard quantum Fourier transformation (QFT) based phase estimation algorithm (QFT-PEA) with more robust quantum counting schemes. The quality of the output image relies dominantly on the quantum counting scheme. For example, the quantum counting scheme used in QSS is the standard QFT-PEA, which outputs a random variable that is concentrated in a vicinity of the ground truth, but has a long tail in the meantime. We quantitatively analyze the performances of different quantum counting schemes [14, 19, 20], and choose the QFT-based maximum likelihood unbiased phase estimation (UPEA) [20] as a substitute for QFT-PEA in QSS. We also propose an adaptive UPEA to improve its robustness.

Finally, we build a 3D scene, use a large enough number of samples with the Blender cycles renderer to generate the ground truth, and simulate the quantum noise by sampling random numbers from the theoretical distribution, to simulate the quantum ray tracing result. We also use a limited number of samples with Blender cycles as a representation of classical path tracing result, and show that quantum ray tracing does perform better than classical ray tracing, conditioned on similar computational cost.

Our contributions are as follows.

- Analyze the performances of some existing quantum counting schemes that are potential substitutes of QFT-based phase estimation algorithm in QSS.
- Propose an adaptive version of unbiased phase estimation algorithm.
- Apply improved QSS to ray tracing algorithm. Do experiments that simulate the workflow of ray tracing, and simulate the images rendered by classical and quantum ray tracing to prove that quantum ray tracing does have better visual performance over classical ray tracing, if appropriate quantum counting scheme is used.

2 Preliminary

2.1 Fundamental concepts of quantum computing

All stories began in the 1980s when Feynman suggested that quantum mechanics might be more computationally powerful than classical computers in some problems like simulating the physical world [21, 22]. By substituting classical bits for quantum bits, or qubits, which can be not only in the states |0⟩ and |1⟩ but also their superposition \(a|0⟩ + b|1⟩\) where \(a, b \in \mathbb{C}\) and \(|a|^2 + |b|^2 = 1\), quantum computing obtains many interesting features like entanglement, reversibility, parallelism, no-cloning and non-orthogonal indistinguishability [8].

Quantum computing follows the gate model with three general steps: initializing all qubits into zero state, performing unitary transformations, and finally measuring them to turn quantum information into classical one. Operations on qubits are implemented
by quantum gates, the quantum variants of classical logic gates. As we know, any classical computing circuit can be constructed with NOT gates, AND gates and COPY gates. Thus, a quantum computer can simulate a classical computer, by restricting the qubit states to \( \{ |0\rangle, |1\rangle \} \), and using the Toffoli gate, X gate and CNOT gate to replace the AND gate, NOT gate and the COPY gate in classical computers, respectively.

Toffoli: \( |a\rangle|b\rangle|0\rangle \mapsto |a\rangle|b\rangle|a \text{ and } b\rangle \)

X: \( |a\rangle \mapsto \text{not } a\rangle \)

CNOT: \( |a\rangle|0\rangle \mapsto |a\rangle|a\rangle \)

Furthermore, due to the reversibility of the three gates above, the quantum implementation of a classical function \( j \mapsto f(j) \) should be of the following form,

\[
|j\rangle|0\rangle \mapsto |j\rangle|f(j)\rangle,
\]

which we abbreviate as \( |j\rangle \mapsto |j\rangle|f(j)\rangle \) throughout this paper. Here \( |j\rangle \) and \( |f(j)\rangle \) are quantum registers that use several qubits to store various data structures like integers and real numbers. It follows immediately that when a superposition state \( \sum j x_j |j\rangle \) is inputted, where \( x_j \)s are arbitrary complex coefficients, the same quantum circuit performs the following linear transformation,

\[
\sum j x_j |j\rangle \mapsto \sum j x_j |j\rangle|f(j)\rangle,
\]

due to the linear property. We call such circuits quantum linear circuits. Any classical circuit can be turned into quantum linear circuit in the brute-force way above, though there are works studying more efficient approaches, like quantum circuit for addition and comparison [23].

Though several evaluations of the function \( f \) are computed in one query, we cannot read them all out directly. Once we measure the state on the computational basis and get access to a specific \( f(j_0) \), the whole state must collapse to the basis state \( |j_0\rangle|f(j_0)\rangle \), and the information of other evaluations is lost forever. Anyway, we have to design clever algorithms to make the best use of quantum parallelism. Some of such examples are Grover’s search [9], minimum finding [17], quantum counting [24], and quantum numerical integrals [15].

2.2 Quantum phase estimation

2.2.1 QFT-based phase estimation

Given a quantum circuit that performs unitary transformation \( U \), and an eigenstate \( |\psi\rangle \) of \( U \) such that

\[
U|\psi\rangle = e^{2\pi i \varphi}|\psi\rangle,
\]

where \( \varphi \) is the phase that needs to be estimated.
Fig. 2 The quantum circuit of QFT-PEA, in which the dashed box is the circuit for the inversed QFT.

Fig. 3 The quantum circuit of IPEA, in which $R_k$ implements the single-qubit operation 
\[
\begin{pmatrix}
1 & 0 \\
0 & \exp(-2\pi i/2^k)
\end{pmatrix}
\]
The double lines stand for classical bits instead of qubits, and the double-line controlled gates mean that if the control bit is 0 then remove the target gate, else reserve the target gate.

The **QFT-based phase estimation algorithm** (QFT-PEA) \([10, 25]\) provides an efficient way to estimate $\varphi$. For general state $|\psi\rangle$ that is not necessarily an eigenstate, let

\[
|\psi\rangle = \sum_j c_j |\psi_j\rangle,
\]

be the orthogonal decomposition onto the eigenspaces of $U$, where $|\psi_j\rangle$ is an eigenstate of $U$ with respect to eigenvalue $e^{2\pi i \varphi_j}$, then PEA returns an estimation of $\varphi_j$ with probability $|c_j|^2$. The PEA is considered the source of quantum speedup of the celebrated Shor’s integer factorization algorithm \([10]\).

The well-known Quantum Fourier transformation (QFT)-based form of PEA uses the circuit shown in Fig. 2, where $H^\otimes t$ gate means applying Hadamard gate to each of the $t$ qubits, $QFT^\dagger$ is the inversed QFT circuit. The number of queries to $U$ is $T - 1$. Let the measurement results be $s_1, s_2, \cdots, s_t$ from top to down. Define $s$ to be the integer whose binary representation is $s_1 \cdots s_2 s_1$, then the estimation of the phase is given by $\tilde{\varphi} = s/T$. From the textbook \([8]\) we can find that the output $\tilde{\varphi}$ of PEA obeys the following distribution,

\[
P(\tilde{\varphi} | \varphi) = \left( \frac{\sin(T \pi (\tilde{\varphi} - \varphi))}{T \sin(\pi (\tilde{\varphi} - \varphi))} \right)^2, \quad \tilde{\varphi} \in \{0, \frac{1}{T}, \frac{2}{T}, \cdots, 1 - \frac{1}{T}\}, \quad (5)
\]

where $T = 2^t$. From Eq. (5) we know that QFT-PEA is accurate when $\varphi$ is an integer multiplication of $T^{-1}$, and shows the biggest noise when $\varphi$ is a half integer multiplication of $T^{-1}$. The probability of estimating within accuracy $1/T$ is at least $8/\pi^2$ \([24]\).
2.2.2 Iterative phase estimation

Anyway, current hardware development allows only a limited amount of quantum memory and gate operations before decoherence. Therefore, compared to pure quantum algorithms that run entirely on quantum realm, the hybrid quantum-classical algorithms that run with less quantum memory and time in one shot and relies on post-processing on classical computers are more likely to be implemented on NISQ (Noise intermediate-scale quantum computers) in the near future.

The iterative phase estimation algorithm (IPEA) \cite{26–28} is a hybrid quantum-classical variant of QFT-PEA. From the textbook \cite{8} we know that the measurement gates and the controlled gates are exchangeable, so the circuit in Fig. 2 is identical to the circuit in Fig. 3. Furthermore, the circuit can be separated into \( t \) parts, as shown in Fig. 4. First we let \( M = 2^{t-1} \) and \( \phi_1 = 0 \), and obtain a measurement result \( s_1 \). Then we let \( M = 2^{t-2} \) and \( \phi_2 = 0.0s_1 \) in binary representation, and obtain a measurement result \( s_2 \). Continue the process until \( M = 1 \) and \( \phi_t = 0.0s_{t-1} \cdots s_2 s_1 \), we obtain the set \( \{ s_1, s_2, \cdots, s_t \} \), and the following steps are identical to QFT-PEA. Compared to PEA, IPEA uses less memory and time in each individual run, thus is believed to be more error-prone on NISQ. Since QFT-PEA and IPEA are identical, we only talk about IPEA in the rest of this paper.

2.3 Quantum counting schemes

2.3.1 Phase estimation based algorithms

Given any Boolean function \( f: \{0, 1, \cdots, N-1\} \rightarrow \{0, 1\} \), as well as the quantum circuit that performs the transformation,

\[
O_f : |j \rangle \mapsto |j \rangle |f(j)\rangle, \quad (j = 0, 1, \cdots, N-1)
\]

the quantum counting algorithm \cite{24} can estimate the sum \( S = \sum_{j=0}^{N-1} f(j) \) with a quadratic faster convergence rate over classical Monte Carlo counting.

The key idea of quantum counting is to construct a unitary transformation whose eigenvalue contains information about \( S \). The celebrated Grover’s iteration \cite{9} is exactly such a unitary transformation. Let

\[
|\alpha\rangle = \frac{1}{\sqrt{N-S}} \sum_{f(j)=0} |j\rangle,
\]
As illustrated in Fig. 5, a Grover’s iteration [9] $G$ consists of two reflections, one about $|\alpha\rangle$ and the other about $|u\rangle$, that is,

$$G = (I - 2|u\rangle\langle u|) (I - 2|\alpha\rangle\langle \alpha|) = H^\otimes n O_0 H^\otimes n P_f,$$

where $O_0 = I - 2|0\rangle\langle 0|$, and $H^\otimes n$ is to apply Hadamard transformation to each of the $n$ qubits, and $P_f$ is called the phase oracle of $O_f$ which performs the transformation,

$$P_f : \sum_j x_j |j\rangle \mapsto \sum_j (-1)^{f(j)} x_j |j\rangle.$$

Therefore, $U$ acts as a rotation by twice the angle between $|u\rangle$ and $|\alpha\rangle$ on the plane spanned by $\{|\alpha\rangle, |\beta\rangle\}$. When restricted in this plane, the eigenvalues of such a plane rotation is $e^\pm 2\pi i \varphi$, where

$$\varphi = \frac{1}{\pi} \arcsin \sqrt{\frac{S}{N}},$$

is the rotation angle. By applying phase estimation, we can estimate $\varphi$ and $S$.

As is mentioned in [20], the original form of PEA and quantum counting is biased. The solution is to use unbiased phase estimation algorithm (UPEA). We randomly generate $\theta \sim U(0, 1/T)$ for each sequence of $t$ runs in IPEA. The circuit of UPEA is shown in Fig. 6. Moreover, we can run UPEA for $R$ times, and use maximum
likelihood estimation to improve the robustness. A direct application of UPEA to quantum counting will break the unbiasedness, since the relationship between $S$ and $\varphi$ is nonlinear in Eq. (13). Therefore, a following correction step is required to keep the unbiasedness, with a small extra cost of mean absolute error.

### 2.3.2 Amplitude amplification based algorithms

Another category of quantum counting solutions is the Amplitude Amplification (AA) [29]. It is still a research hotspot in recent years [19, 30–35].

In 2019, Suzuki et al. [19] proposed an AA-based Maximum Likelihood Amplitude Estimation (MLAE) algorithm. By applying $O_f$ directly to the uniform superposition state $|u\rangle$ we obtain $1/\sqrt{N} \sum_j |j\rangle |f(j)\rangle$, so the measurement on the second register will output 1 with probability $\sin^2(\pi \varphi) = S/N$. If $M$ times of Grover’s iteration is applied to $|u\rangle$ before measurement, the probability becomes $\sin^2((2M + 1)\pi \varphi)$. The maximum likelihood amplitude estimation (MLAE) [19] choose $M = 0, 1, 2, \ldots, 2^t - 1$, and repeat the measurement for $R$ times for each $M$. Let $h_M$ be the number of measurements with result 1. Then the final estimation of $\tilde{\varphi}$ is obtained by maximizing the likelihood function,

$$L(\tilde{\varphi}; \{h_M\}) = \prod_M \sin^{2h_M} [(2M + 1)\pi \varphi] \cos^{2(R-h_M)} [(2M + 1)\pi \varphi]. \quad (14)$$

Experiments show that MLAE also has a quadratic faster error convergence over Monte Carlo method.

In 1999, Abrams et al. [15] proposed an AA-based algorithm for quantum amplitude estimation, and the algorithm is later called quantum coin method (QCoin) and numerical tested in [14]. To estimate $\varphi \in [0, 1]$ with high accuracy, they first estimate with Monte Carlo method to a smaller interval $[\varphi - \delta/2, \varphi + \delta/2]$, then remap the interval to $[0, 1 - \varepsilon]$ by amplitude amplification, and use Monte Carlo method to an even smaller interval. By using different number $M$ of AA in each iteration, where $M = 2^m$ for $m \in \{0, 1, \ldots, t - 1\}$, and each iteration using $R$ repetitions, the information about $\varphi$ is extracted also with quadratic faster error convergence than Monte Carlo method.

### 3 Algorithm

When dealing with a numerical integration problem, classical Monte Carlo method randomly evaluates $N$ samples to get an estimation with an error convergence of $1/\sqrt{N}$. In quantum computers, a large range of samples can be made into a superposition state, and thus can be computed in one shot. Based on this idea, Johnson [13]
proposed quantum supersampling (QSS), and experimentally proved it to have a faster convergence than Monte Carlo method. However, their result images contain many detached noisy dots that severely affect the quality of image, as they use a non-robust QFT-based phase estimation (QFT-PEA) as the quantum counting scheme. In this section, we present a framework of quantum ray tracing, then propose improved QSS by replacing the QFT-based phase estimation with more robust quantum counting schemes.

3.1 Framework of quantum ray tracing

In classical path tracing, many ray paths are required to calculate the color of a single pixel as accurately as possible. In quantum computing, the information of all those rays can be stored in a superposition. All we need is an extra register that stores the ID of each superposed ray, in the form $\sum |ID\rangle|\text{ray ID}\rangle$. Moreover, the quantum memory used for storing those IDs uses only a logarithmic space, while the memory used for storing path information like origins and directions is shared in superposition, so the biggest advantage of quantum ray tracing is that we can reduce the sampling error to a negligible level by using an astronomical number of rays.

As is introduced, we can assume that we are given a ray tracing oracle implementing the following transformation,

$$O_f(\text{pixel, channel}) : \sum_{j=0}^{N-1} x_j |j\rangle \mapsto \sum_{j=0}^{N-1} x_j |j\rangle |f(j)\rangle,$$

(15)

where pixel and channel (R, G or B) are classical parameters, $j$ plays the role of ray ID, and $f(j)$ is a real number that stands for the ray energy. In the rest of this paper $f$ is specified as the function that maps ray ID to ray energy. The oracle can trace $N = 2^n$ paths simultaneously, and the final color we hope to write to the corresponding pixel and channel is the average of those energies,

$$S = \frac{1}{N} \sum_{j=0}^{N-1} f(j).$$

(16)

Suppose those real numbers $f(j)$ are stored in a fixed-point format with integer bit length $b_0$ and total bit length $b$, we can transfer the estimation problem of Eq. (16) into quantum counting by constructing a Boolean function,

$$g(j, k) = \begin{cases} 
1, & f(j) \geq 2^{b_0-b}k; \\
0, & f(j) < 2^{b_0-b}k. 
\end{cases} \quad (k = 0, 1, \cdots, 2^b - 1)$$

(17)

The phase oracle $O_g$ for $g$ in Grover’s search is,

$$O_g : \sum_{j, k} |j\rangle|k\rangle \mapsto \sum_{j, k} (-1)^{g(j, k)} |j\rangle|k\rangle.$$  

(18)
Fig. 7  The construction of controlled-$O_g$, where the first qubit is the control qubit, and $Z: |x⟩ → (-1)^x |x⟩$.

Fig. 8  The construction of controlled-$G_g$ gate, where $|j, k⟩$ is written as an entity of the two registers $|j⟩$ and $|k⟩$.

To construct $O_g$, we need a comparison gate that performs the comparison operation $COMP$ on the two integers $2^{b-b_0} f(j)$ and $k$,

$$COMP: \sum_{j,k} |f(j)||k⟩ \mapsto \sum_{j,k} |f(j)||k⟩|g(j, k)⟩,$$  \hspace{1cm} (19)

which is already constructed by [23]. With this in hand, the $O_g$ gate can be easily constructed, as shown in Fig. 7.

It is easy to verify that,

$$S = \frac{1}{2^n} \sum_{j=0}^{N-1} f(j) = \frac{1}{2^{n+b-b_0}} \sum_{j=0}^{N-1} \sum_{k=0}^{2^{b-1}} g(j, k).$$ \hspace{1cm} (20)

Also, the conversion to fixed-point format brings a truncation error of $O(2^{-b} N)$.

Finally, the quantity $\sum_{j,k} g(j, k)$ can be estimated by quantum counting algorithm. The construction of controlled-$G_g$ required in quantum counting is shown in Fig. 8. This is also where the dominant error of the whole procedure is brought.

In the paper we assume one call to $O_f$ in quantum realm takes the same time as tracing one path in classical realm. We evaluate the cost of classical path tracing by the number of ray paths $N_c$, as the noise comes mostly from the Monte Carlo integration. And in quantum ray tracing, the time cost is evaluated by the number of queries $N_q$ to the ray tracing oracle $O_f$, and the noise comes mostly from the random distribution of the output of PEA. The QFT-based quantum counting has an error convergence rate of $O(1/N_q)^{[24]}$, hence has a quadratic speedup over classical Monte Carlo integration with convergence rate of $O(1/\sqrt{N_c})$.

Finally, the averaging step of ray energies takes place in the linear high-dynamic range (HDR) color space. To obtain a color between range $[0, 1]$ in the standard-dynamic range (SDR) space, a tone mapping$^[36]$ step should be applied to the averaged color.
In many cases, QFT-based phase estimation algorithm (QFT-PEA) is already enough to use. However, from Eq. (5) we know \( P(\tilde{\varphi} \mid \varphi) \) decays with an order of \( \sin^{-2}(\pi(\tilde{\varphi} - \varphi)) \) as \( |\tilde{\varphi} - \varphi| \) grows big. That is, though the output \( \tilde{\varphi} \) of PEA is randomly distributed around the ground truth \( \varphi \), it also has a long tail which shows as many detached noisy dots on the image. That is exactly why the result of the original form of QSS contains many distinct noisy dots.

To have a visual impression on this proposition, we do the experiments on the gray disk with the gray scale varying linearly from black at the center to white on the border. The gray scale of each pixel stands for \( S/N \) in quantum counting. By sampling from the theoretical random distribution of each algorithm, we can simulate the output \( \tilde{S}/N \) and draw the gray scale to the corresponding pixel. In Fig. 9, (a) shows the result of Monte Carlo sampling with \( N \) samples drawn from the binary distribution as a comparison, (b) shows the result of QFT-PEA, which contains some distinct dots. By using UPEA with maximum likelihood estimation, the result is shown in Fig. 9c, which is smooth in most area and the visual noise level is lower.

Based on the observation that about \( 8/\pi^2 \) estimations of PEA are within range \( \pm 1/T \), we can dynamically adjust \( R \) until most of the samples are within an interval of length \( 2/T \). We propose the adaptive UPEA in Algorithm 1.
Algorithm 1 Adaptive UPEA.

Require: $T, \alpha, N_{\text{min}}, N_{\text{max}}$;
Ensure: $\tilde{\phi}$: the estimation of $\phi$.
1: Initialize $S = \{\tilde{\phi}_1, \cdots, \tilde{\phi}_{N_{\text{min}}}\}$ with $N_{\text{min}}$ samples using QFT-PEA with parameter $T$;
2: Sort $S$ from small to big;
3: while $\# S < N_{\text{max}}$ and There is no subset in $S$ of length $\lfloor \alpha \# S \rfloor$ and interval at most $2/T$ do
4: Insert one more sample into $S$ while keeping $S$ sorted;
5: end while
6: Remove samples from $S$ that is not in the interval;
7: $\tilde{\phi} =$ the Bayesian estimation from dataset $S$.

In Fig. 9, we further do gray disk experiments for (d) adaptive UPEA, (e) MLAE and (f) QCoin. We can see that adaptive UPEA and QCoin has a lower noise level than UPEA, and MLAE shows some visual fake rings.

3.3 Comparisons between these quantum counting schemes

To quantitatively analyze how noise scales with the number of queries in those schemes, we randomly generate $10^6$ numbers uniformly between 0 and 1 as ground truths of $S/N$, simulate the quantum counting algorithm with different schemes, and use the following statistical quantity to evaluate each scheme:

- Mean absolute error: to evaluate the overall noise level.
- Percentage of samples beyond the accuracy of 0.1, 0.01 and 0.001: to evaluate the level of distinct noisy dots.

The results are shown in Fig. 10. In (a), each quantum algorithm shows a faster error convergence than classical Monte Carlo algorithm. In (b–d) we evaluate the concentration of estimations, where QCoin and adaptive UPEA perform the best. It should be mentioned that, adaptive UPEA behaves similarly to UPEA in mean absolute error, but much better in concentration.

Next, since the distribution of $\tilde{\phi}$ of each quantum counting scheme is relevant to $\phi$, we do another experiment to illustrate the error pattern of each scheme. To be specific, for fixed $\phi$s, we test each scheme for $N_{\text{test}}$ times, then analyze the bias

$$\frac{\sum_{i=1}^{N_{\text{test}}} \tilde{\phi}_i}{N_{\text{test}}} - \phi,$$

and the mean absolute error (MAE)

$$\frac{1}{N_{\text{test}}} \sum_{i=1}^{N_{\text{test}}} |\tilde{\phi}_i - \phi|.$$

We choose $\phi \in \{0, 0.001, 0.002, \cdots, 1\}$ and $N_{\text{test}} = 10000$. The results are shown in Fig. 11) The Monte Carlo algorithm is unbiased, but has the highest MAE. The MLAE has a periodical bias, which can also cause fake rings like QFT-based family. And QCoin tends to give an underestimation around 0 or 1. Both them have a moderate
Fig. 10 Quantitative evaluations on how the error scales with the number of queries to $G_g$ (the x-axis) for each scheme

(a) Monte Carlo with $R = 256$.
(b) MLAE with $T = 8$ and $R = 32$.
(c) QCoin with $T = 8$ and $R = 32$.
(d) QFT-PEA with $T = 256$.
(e) UPEA with $T = 64$ and $R = 4$.
(f) Adaptive UPEA with $T = 64$, $R = 4$, $N_{\text{min}} = 3$, $N_{\text{max}} = 8$ and $\alpha = 0.8$, which reports an average of 265 oracle-calls.

Fig. 11 The error patterns of different schemes
level of MAE. The original QFT-PEA has an intensively fluctuating bias. And the UPEA and adaptive UPEA behaves well in both bias and MAE.

4 Experiment

In this section, we make comparisons on real images rendered by classical path tracing and simulated quantum ray tracing with different schemes.

We use the Blender cycles renderer as a representation of the state-of-the-art classical ray tracing renderer. As for the quantum ray tracing result, both actual quantum computers and classical simulators available now cannot provide enough quantum
memory as well as coherence time for quantum ray tracing to show its power of superposing an astronomical number of rays. Instead, we first render with a huge number of samples in Blender as the ground truth, output the colors in HDR space, and scale them into $[0, 1]$. Then we simulate the quantum noise with the same method in the pre-experiments in the scaled HDR space. Finally, we scale them back, apply tone mapping and gamma correction to obtain the SDR color, and write these colors to the image.

We use the number of intersection searching sub-procedures as the measurement of the cost. Though we can control the number of samples per pixel and the max tracing depth $D$, the actual number of intersection searching can only be computed by

---

**Fig. 15** Quantum ray tracing using Adaptive UPEA with $T = 256$, $\alpha = 0.8$, $N_{\text{min}} = 3$ and $N_{\text{max}} = 8$, which reports $N_q = 1075$

**Fig. 16** Quantum ray tracing using MLAE. $T = 16$, $R = 16$. $N_q = 1120$

**Fig. 17** Quantum ray tracing using QCoin. $T = 64$, $R = 16$. $N_q = 992$
modifying the source code of Blender, since each ray may experience different tracing depth or direct light samples. In comparison, the number of intersection searching of quantum ray tracing is completely definite, as all rays are traced as a superposition and thus share the same depth. Suppose each $O_f$ traces the superposed rays to depth $D$, then one Grover’s iteration $G_g$ contains $2D$ number of intersection searching, as one $G_g$ consists of one $O_f$ and one $O_f^{-1}$. Suppose the quantum counting scheme requires $N_q$ queries to $G_g$, and notice that for each pixel there are three channels (RGB), then the calculation of the color of a single pixel requires $6DN_q$ number of intersection searching.

Denote $N_c$ to be the number of intersection searching divided by the number of pixels and $D$, as a measurement of the cost of classical ray tracing. Then our experiment compares the rendering result of them conditioned on

$$N_c \approx 6N_q,$$  \hspace{1cm} (23)

as it is hard to force $N_c = 6N_q$.

We use Blender cycles to render a scene with $2^{20}$ samples per pixel to depth $D = 4$ as the ground truth, as illustrated in Fig. 12. It should be mentioned that those ray IDs only require 20 extra qubits than storing a single ray in real quantum computers. In this scene we use $b_0 = 4$ in Eq. (20), that is, each HDR color is scaled to $1/16$ before simulating the quantum random distribution. In addition, we use Blender cycles with $2^{20}$ samples per pixel as the ground truth, as a representation of classical ray tracing result, as shown in Fig. 13. We simulate the quantum ray tracing result, with QFT-PEA in Fig. 14, adaptive UPEA in Fig. 15, MLA E in Fig. 16 and QCoin in Fig. 17.

In this experiment, while the classical ray tracing result is still noisy, the quantum ray tracing results can be already clean enough. The QFT-PEA result with $T = 1024$ shows many distinct noisy dots, but shows good smoothness in other area. The adaptive UPEA result contains much slighter distinct dots. The MLA E result shows a moderate level of fake rings and smoothness. Finally, QCoin does not behave well in this case where the HDR color of most pixels are not exposed and becomes close to zero when divided by $2^{b_0} = 16$, so QCoin tends to give an underestimation, as previously shown in Fig. 11c.

5 Discussion

Here we discuss the limitations of the application of improved QSS in quantum ray tracing.

First, we claim that we are comparing quantum ray tracing and classical ray tracing by comparing the visual noise level conditioned on similar numbers of queries. In other words, we assume that the costs of performing an intersection searching in both quantum ray tracing and classical ray tracing are the same. However, classical ray tracing uses data structures like BSP tree [37], KD-tree [38, 39] and BVH [40] to avoid traversing all scene primitives. Meanwhile, in quantum ray tracing where an astronomical number of rays are superposed, we have to traverse all scene primitives. In
short, we may overestimate the time cost of classical ray tracing, and thus overestimate the speedup of quantum ray tracing.

Second, since the rendering equation is infinitely recursive and any cutoff on the ray tree will bring numerical truncation error, classical ray tracing utilizes Russian Roulette to ensure the unbiasedness, but in quantum ray tracing all superposed rays share the same ray tree depth, and we have no access to the intermediate information to decide the halt condition dynamically and have to pre-determine a fixed ray tree, which may cause visual artifacts and must be compensated for. But in the experiments of this paper we do not simulate anything about that.

Third, quantum ray tracing only shows its power when the scene complexity is large enough. As we have addressed, the noise level of quantum ray tracing is irrelevant to the scene complexity. If a moderate level of ray samples in classical ray tracing can reduce the noise to an acceptable level, then the quantum ray tracing result with the same number of queries just brings extra noise. In real-time ray tracing where computational resources are strictly limited, quantum ray tracing may not perform better.

6 Conclusion

In the original form of QSS [13], the image produced by quantum supersampling contains many detached noisy dots, as illustrated in Fig. 14. We improve quantum supersampling for quantum ray tracing, by replacing the standard QFT-based phase with more robust quantum counting schemes. We compare different schemes like UPEA, MLAE and QCoin, and propose an adaptive UPEA that behaves best among these schemes as it is unbiased and shows the least mean absolute error.

Finally, we build a 3D scene, use \(2^{20}\) samples per pixel with Blender cycles renderer to generate the ground truth, and then simulate the quantum noise in the HDR linear space by sampling random numbers with respect to the theoretical distribution, to simulate the quantum ray tracing result. We show that by choosing adaptive UPEA as the quantum counting schemes, the image quality can be much better than classical ray tracing and original QSS.

Acknowledgements This work is supported by the National Natural Science Foundation of China under Grant nos. 62272406, 61872316, and the National Key Research and Development Plan of China under Grant no. 2020YFB1708900.

Data availability statement The data that support the findings of this study available from the corresponding author upon reasonable request.

Declarations

Conflict of interest The authors report no conflict of interest.
References

1. Whitted, T.: An improved illumination model for shaded display. Commun. ACM 23, 343–349 (1980). https://doi.org/10.1145/358876.358882
2. Cook, R.L., Porter, T., Carpenter, L.: Distributed ray tracing. Comput. Graph. 18, 137–145 (1984). https://doi.org/10.1145/800301.808590
3. Kajiya, J.T.: The rendering equation. In: Proceedings of the 13th Annual Conference on Computer Graphics and Interactive Techniques, SIGGRAPH vol. 20, 143–150 (1986). https://doi.org/10.1145/15922.15902
4. Haines, E., Akenine-Möller, T.: Ray Tracing Gems: High-Quality and Real-Time Rendering with DXR and Other APIs. Apress, New York (2019).
5. Akenine-Möller, T., Haines, E., Hoffmann, N.: Real-Time Rendering, 4th edn. A. K. Peters Ltd, Natick (2018).
6. Zeng, Z., Liu, S., Yang, J., Wang, L., Yan, L.-Q.: Temporally reliable motion vectors for real-time ray tracing. Comput. Graph. Forum 40(2), 79–90 (2021). https://doi.org/10.1111/cgf.142616
7. Schied, C., Kaplanyan, A., Wyman, C., Patney, A., Chaitanya, C.R.A., Burgess, J., Liu, S., Dachsbacher, C., Lefohn, A., Salvi, M.: Spatiotemporal variance-guided filtering: Real-time reconstruction for path-traced global illumination. In: Proceedings of High Performance Graphics. HPG ’17. Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3105762.3105770
8. Nielsen, M.A., Chuang, I.: Quantum computation and quantum information. Am. Assoc. Phys. Teach. 70, 558–559 (2002)
9. Grover, L.K.: Quantum mechanics helps in searching for a needle in a haystack. Phys. Rev. Lett. 79, 325 (1997). https://doi.org/10.1103/PhysRevLett.79.325
10. Shor, P.W.: Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer. SIAM J. Comput. 26, 1484–1509 (1997). https://doi.org/10.1137/S0097539795293172
11. Lanzagorta, M., Uhrmann, J.K.: Hybrid quantum-classical computing with applications to computer graphics. ACM SIGGRAPH 2005 Courses, SIGGRAPH 2005 (2005). https://doi.org/10.1145/1198555.1198723
12. Caraiman, S.: Quantum computer graphics algorithms. Buletinul Institutului Politehnic din Iasi, Sectia Automatica si Calculatoare 62(4), 21–38 (2012).
13. Johnston, E.R.: Quantum supersampling. ACM SIGGRAPH 2016 Talks (2016). https://doi.org/10.1145/2897839
14. Shimada, N.H., Hachisuka, T.: Quantum coin method for numerical integration. Comput. Graph. Forum 39, 243–257 (2020). https://doi.org/10.1111/cgf.14015
15. Abrams, D.S., Williams, C.P.: Fast quantum algorithms for numerical integrals and stochastic processes. arXiv (1999)
16. Santos, L.P., Bashford-Rogers, T., Barbosa, J., Navrátil, P.: Towards quantum ray tracing (2022). https://doi.org/10.48550/arxiv.2204.12797
17. Durr, C., Hoyer, P.: A quantum algorithm for finding the minimum. arXiv (1996)
18. Giovannetti, V., Lloyd, S., Maccone, L.: Quantum random access memory. Phys. Rev. Lett. 100(16), 160501 (2008)
19. Suzuki, Y., Uno, S., Raymond, R., Tanaka, T., Onodera, T., Yamamoto, N.: Amplitude estimation without phase estimation. Quantum Inf. Process. (2019). https://doi.org/10.1007/s11128-019-2565-2
20. Lu, X., Lin, H.: Unbiased quantum phase estimation. arXiv preprint arXiv:2210.00231 (2022)
21. Feynman, R.P.: Simulating physics with computers. Int. J. Theor. Phys. 21(6), 467–488 (1982). https://doi.org/10.1007/BF02650179
22. Feynman, R.P.: Quantum mechanical computers. Found. Phys. 16(6), 507–531 (1986). https://doi.org/10.1007/BF01886518
23. Takahashi, Y., Tani, S., Kunihiro, N.: Quantum addition circuits and unbounded fan-out. Quantum Inf. Comput. 10, 872–890 (2009). https://doi.org/10.26421/qic10.9-10-12
24. Brassard, G., Hoyer, P., Tapp, A.: Quantum counting. Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 1443 LNCS, pp. 820–831 (1998). https://doi.org/10.1007/BfB0055105
25. Kitaev, A.Y.: Quantum measurements and the abelian stabilizer problem. arXiv: https://arxiv.org/abs/quant-ph/9511026v1 (1995)
26. Mosca, M., Ekert, A.: The hidden subgroup problem and eigenvalue estimation on a quantum computer. In: NASA International Conference on Quantum Computing and Quantum Communications, pp. 174–188 (1998). Springer
27. Dobšíček, M., Johansson, G., Shumeiko, V., Wendin, G.: Arbitrary accuracy iterative quantum phase estimation algorithm using a single ancillary qubit: A two-qubit benchmark. Phys. Rev. A 76(3), 030306 (2007)
28. Dobšíček, M.: Quantum computing, phase estimation and applications. arXiv (2008). https://doi.org/10.48550/arxiv.0803.0909
29. Brassard, G., Hoyer, P., Mosca, M., Tapp, A.: Quantum amplitude amplification and estimation. Contemp. Math. 305, 53–74 (2002)
30. Svore, K.M., Hastings, M.B., Freedman, M.: Faster phase estimation. Quantum Inf. Comput. 14, 306–328 (2013). https://doi.org/10.26421/qic14.3-4-7
31. Wiebe, N., Granade, C.: Efficient Bayesian phase estimation. Phys. Rev. Lett. 117, 010503 (2016). https://doi.org/10.1103/PHYSREVLETT.117.010503/FIGURES/5/MEDIUM
32. Wie, C.R.: Simpler quantum counting. Quantum Inf. Comput. (2019). https://doi.org/10.26421/QIC19.11-12 https://doi.org/10.26421/QIC19.11-12 https://doi.org/10.26421/QIC19.11-12
33. Aaronson, S., Rall, P.: Quantum approximate counting, simplified. In: Symposium on simplicity in algorithms, pp. 24–32 (2020). SIAM
34. Grinko, D., Gacon, J., Zoufal, C., Woerner, S.: Iterative quantum amplitude estimation. NPJ Quantum Inf. 7, 1–6 (2021). https://doi.org/10.1038/s41534-021-00379-1
35. Plekhanov, K., Rosenkranz, M., Fiorentini, M., Lubasch, M.: Variational quantum amplitude estimation. Quantum 6, 670 (2022). https://doi.org/10.22331/q-2022-03-17-670
36. Arrighetti, W.: The academy color encoding system (aces): a professional color-management framework for production, post-production and archival of still and motion pictures. J. Imaging 3(4), 40 (2017)
37. Fuchs, H., Kedem, Z.M., Naylor, B.F.: On visible surface generation by a priori tree structures. ACM SIGGRAPH Comput. Graph. 14, 124–133 (1980). https://doi.org/10.1145/965105.807481
38. Bentley, J.L.: Multidimensional binary search trees used for associative searching. Commun. ACM 18, 509–517 (1975). https://doi.org/10.1145/361002.361007
39. Hapala, M., Havran, V.: Review: Kd-tree traversal algorithms for ray tracing. Comput. Graph. Forum 30, 199–213 (2011). https://doi.org/10.1111/j.1467-8659.2010.01844.x
40. Clark, J.H.: Hierarchical geometric models for visible surface algorithms. Commun. ACM 19, 547–554 (1976). https://doi.org/10.1145/360349.360354

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.