Quasi-Monte Carlo ray tracing algorithm for radiative flux distribution simulation

Xiaoyue Duan[a], Caitou He[a,c], Xiaoxia Lin[a], Yuhong Zhao[b,*], Jieqing Feng[a,*]

[a] *State Key Laboratory of CAD & CG, Zhejiang University, Hangzhou, 310058, China*
[b] *Institute of Industrial Process Control, College of Control Science and Engineering, Zhejiang University, Hangzhou, 310027, China*
[c] *College of Computer Science and Artificial Intelligence, Wenzhou University, Wenzhou, 325006, China*

Abstract

Monte Carlo ray tracing (MCRT) is a fundamental simulation method for central receiver systems(CRSs). MCRT is an effective method to describe the radiative flux distribution on the receiver surface reflected by either a single heliostat or all heliostats in a heliostat field. In this paper, a GPU-based ray-tracing simulation method, namely, quasi-Monte Carlo ray tracing (QMCRT), is proposed to address problems of both efficiency and accuracy. First, QMCRT, as a bidirectional approach, can avoid unnecessary intersection calculations. This method also facilitates sunshape sampling and heliostat surface slope error sampling, which can achieve memory and run-time efficiency. Second, in the traditional approaches, the simulated maximum radiative flux (MaxRF) is randomly higher than the reference value, even if tens of millions of rays are traced. In QMCRT, the problem is solved by applying a trimmed mean smoothing operation to the generated radiative flux distribution. As a result, a stable MaxRF value approaching the reference value is obtained, while the total power remains almost unchanged. The results obtained for both synthetic and real heliostats obtained using QMCRT are substantially in keeping with the results obtained using established computational tools. QMCRT is two orders of magnitude faster than the traditional MCRT method when addressing traditional one-reflection CRS case. Compared with the state-of-the-art GPU-based grid ray tracing (GRT) approach, QMCRT is equally fast but generates a more accurate result. QMCRT also has an advantage in terms of efficiency for CRS compared with two well-known simulation software tools, i.e., SolTrace and Tonatiuh.

Keywords: Central receiver system, Monte Carlo ray tracing, Graphics processing unit, Radiative flux distribution, Maximum radiative flux

1. Introduction

Efforts regarding the development and utilization of solar energy are attracting increasing attention because of its clean and renewable nature. The most common central receiver systems (CRSs) (Behar et al., 2013; Li et al., 2016; Levêque et al., 2017) are power facilities for converting solar energy into electrical energy (Lovegrove and Stein, 2012; Duffie et al., 2013). In this type of CRS, thousands of highly reflective mirrors, known as heliostats, are deployed to form a heliostat field. The heliostats track the movement of the sun and concentrate the lights they reflect onto the surface of a central receiver, which is usually mounted on top of a tower (Behar et al., 2013). The concentrated radiation energy heats the transfer fluid in the central receiver, such as water or molten salt, for subsequent electricity generation (Conroy et al., 2018). There are many concerns regarding the

*Joint corresponding author
Email addresses: xiaoyueduan@zju.edu.cn (Xiaoyue Duan), hctou89@zju.edu.cn (Caitou He), linxxcad@zju.edu.cn (Xiaoxia Lin), yhzhao@iipc.zju.edu.cn (Yuhong Zhao), jqfeng@cad.zju.edu.cn (Jieqing Feng)

design and deployment of a heliostat field (Imenes et al., 2006), the aiming strategy for the heliostats (Wang et al., 2017), and the estimation of the yearly received energy (Islam et al., 2018). Issues as such these ones could threaten the construction and operations of a CRS, since they could greatly affect the economy, efficiency and safety of solar energy utilization. The tasks mentioned above are related to the radiative flux distribution that is reflected onto the receiver surface by either a single heliostat or all heliostats in the heliostat field. The reflected radiative flux can be influenced by many optical losses, which are elaborated by Li et al. (2016) and Levêque et al. (2017), accompanied by available simulation tools. Therefore, an efficient and accurate algorithm for simulating this radiative flux distribution is a fundamental requirement for CRS research.

At present, there are two approaches for simulating the radiative flux distribution on the receiver surface: one is Monte Carlo ray tracing (MCRT) (Veach and Guibas, 1995; Jensen and Christensen, 1995; Modest, 2013), and the other is the analytical approach. MCRT traces millions or even billions of rays that are emitted from the sun. These rays are then reflected by the heliostats and projected to the receiver surface. In the analytical approach, the radiative flux distribution utilizes an analytical function in the form of a Gaussian function or convolution. In general, MCRT is more accurate, and the analytical approach is more efficient. Since the mechanism of energy transmission from the sun to the receiver surface via a heliostat is very complicated, the result of MCRT is commonly regarded as the reference for the radiative flux distribution simulation (Garcia et al., 2008; Izygon et al., 2011; Lovegrove and Stein, 2012). In turn, the results can also be used to derive an approximate analytical distribution model and determine the model's parameters. Thus, improving the efficiency and accuracy of ray-tracing simulations is worthwhile.

First, due to the inherent characteristics of Monte Carlo method, obtaining an accurate result with MCRT is time–consuming. The intermediate data structure of this method is, more often than not, designed elaborately. According to the experiments conducted, obtaining an accurate simulation result using a typical MCRT method for a single heliostat with dimensions of $1.25\,\mathrm{m}\times1.6\,\mathrm{m}$ would require tracing approximately 20 million rays. To accelerate the MCRT process, one feasible solution is to perform MCRT on a contemporary graphics processing unit (GPU). A GPU can utilize its tremendous parallel computing power and high input and output capacity. However, due to the enormous number of rays to be traced, there is a dilemma regarding whether to generate all arguments on the fly or load them from a precomputed data set. It requires a large amount of computations when tracing reflections from thousands or tens of thousands of heliostats in a field, even for a GPU.

Second, it is challenging to efficiently and randomly sample the angular distribution of the incident sun rays, the sunshape (Wang et al., 2020). In general, Gaussian (Bendt and Rabl, 1981), PillBox (Biggs and Vittitoe, 1979), and Buie (Buie et al., 2003) sunshapes are widely employed. Among these sunshapes, the Buie sunshape is more accurate than the others since the spatial radial distribution is defined based on the observed data; however, it is difficult to sample faithfully because of its complicated probability distribution. The most commonly used sampling method, Monte Carlo Markov chain (MCMC) sampling (Henrik et al., 2003), is sequential and requires approximately 3.5 seconds (s) to sample 20 million random variables on a CPU. It may be a feasible solution to precompute all of the samples, load them into GPU memory, and process them in parellel. However, the total arguments will occupy approximately 468 megabytes (MB) of memory. Thus, an efficient and accurate sampling method for the Buie sunshape is yet another challenge for an MCRT algorithm.

Third, the maximum radiative flux (MaxRF) value is important for parameter fitting of analytical models, such as HFLCAL (Schwarzbözl et al., 2009). However, the MaxRF value is numerically unstable due to the various random samplings performed in MCRT, and sometimes it is 10% higher than the reference value. As a result, the numerical instability of MaxRF is the third drawback for MCRT. Consequently, an accurate ray-tracing method with a stable MaxRF value that is efficient in terms of run time and memory is still worth exploring.

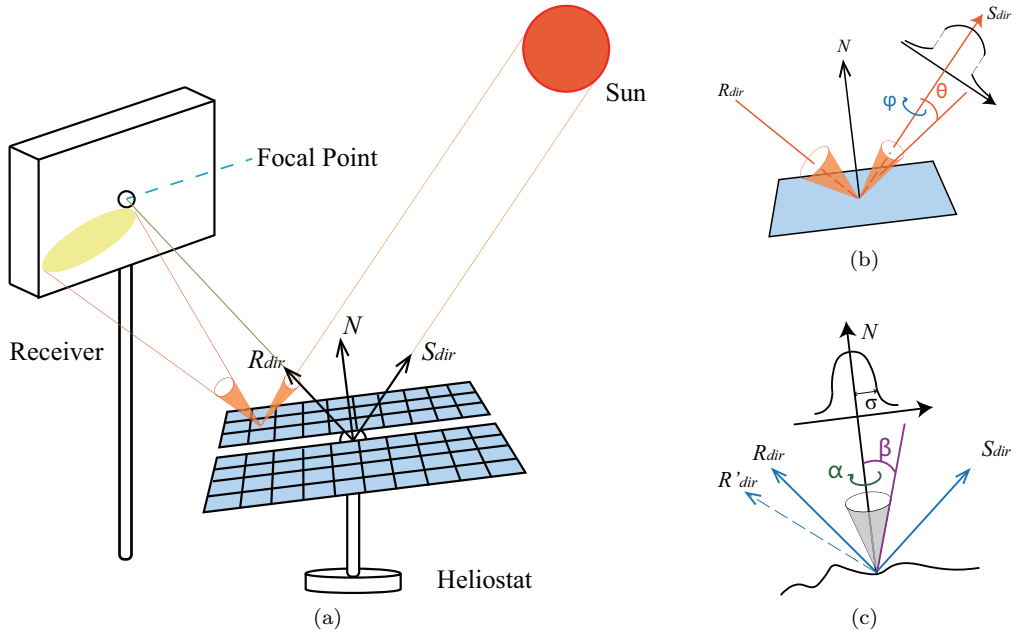


Figure 1: Illustration of the bidirectional ray tracing strategy used in QMCRT. (a) An example of a heliostat aimed at the center of the receiver at a given moment, where S_{dir} is the direction opposite to the direction of incidence from the sun, R_{dir} is the main reflection direction (as determined from the center of the heliostat and the focal point), and N is the normal to the heliostat. (b) Illustration of the parameters φ and θ, which determine the direction of an incident ray. (c) Illustration of the parameters α, β and σ, which determine the direction of the normal to the local microheliostat. R'_{dir} is the real reflection direction considering the surface slope error of the microheliostat.

In this paper, a novel quasi-Monte Carlo ray tracing (QMCRT) method is proposed to address the above problems. QMCRT is fully designed and implemented on a GPU. This method uses a bidirectional ray-tracing strategy to accelerate the computations, as illustrated in Figure 1a. In this instance,"quasi" means that the algorithm does not generate all of the random variables. Instead, the algorithm uses two precomputed random variables sets: the SunShape Pool (SSP), whose elements are drawn from the Buie sunshape distribution controlled by a parameter called the CircumSolar Ratio (CSR); and the Microheliostat Normal Pool (MHNP), whose elements are drawn from a Gaussian distribution with a given standard deviation (Box and Muller, 1958). Although the sizes of the two pools are much smaller than the number of rays to be traced, flexible combinations of their elements can yield simulation results similar to those of MCRT. This approach significantly reduces the memory consumption and execution time for on-the-fly random variable generation. In addition, a GPU-friendly sampling approach called inverse transform sampling (ITS) (Inverse Transform Sampling) is adopted in QMCRT to accelerate the sampling of the Buie sunshape. Thus, QMCRT solves the problems of massive sampling workloads and heavy memory consumption that are embedded in MCRT. In QMCRT, each heliostat is uniformly tessellated into small tiles called microheliostats. For each microheliostat, a bundle of incident rays originating from the center of the microheliostat is generated via the SSP. The reflection directions of the rays are determined by selecting elements from the MHNP. Therefore, QMCRT is applicable for heliostats of any shape, even those characterized by measured geometric data. In QMCRT, shadowing and blocking effects are efficiently processed via a spatial grid index technique called the 3-Dimensional Digital Differential Analyzer (3D-DDA) (Amanatides and Woo, 1987). After ray tracing, a trimmed mean smoothing (TMS) algorithm is applied to alleviate the radiative flux noise on the receiver. As a result, a stable MaxRF value approaching the reference value is obtained, while the total power is barely changed. The results for both synthetic and real heliostats obtained using the proposed QMCRT approach substantially agree with the results obtained

using established computational tools. Furthermore, QMCRT also has an advantage in terms of efficiency for one-reflection CRS when compared with two well-known simulation software tools, i.e., SolTrace and Tonatiuh.

The main contributions of this paper are summarized as follows:

(1) An efficient and accurate QMCRT algorithm is proposed. Regarding efficiency, the algorithm is fully designed and implemented on a GPU and saves memory and execution time by using a pregenerated SSP and MHNP. For accuracy, the algorithm can produce results as accurate as those of MCRT by flexibly combining the elements in the SSP and MHNP. QMCRT is also applicable for heliostats of any shape.

(2) A sampling approach, ITS, is adopted to sample the complex probability distribution of the Buie sunshape (Buie et al., 2003).

(3) The MaxRF instability problem is addressed in QMCRT via a TMS operation. As a result, an accurate MaxRF value approaching the reference value is obtained, while the total power remains almost unchanged.

The nomenclature that is frequently used in this paper is defined as follows:

Nomenclature

- CRS: central receiver system
- MCRT: Monte Carlo ray tracing
- GPU: graphics processing unit
- MCMC: Monte Carlo Markov chain
- MaxRF: maximum radiative flux
- QMCRT: quasi-Monte Carlo ray tracing
- SSP: SunShape Pool
- CSR: CircumSolar Ratio
- MHNP: Microheliostat Normal Pool
- ITS: inverse transform sampling
- 3D-DDA: 3-Dimensional Digital Differential Analyzer
- TMS: trimmed mean smoothing
- **xyz**: global coordinate system
- **N**: heliostat surface normal
- S_{dir}: direction opposite to the major direction of incidence from the sun
- φ, θ: parameters determining the direction of an incident solar ray
- σ: standard deviation of the heliostat slope error
- χ: circumsolar ratio specifying the Buie sunshape
- α, β: parameters determining the direction of a microheliostat normal
- HIA: heliostat index array
- HSIA: heliostat starting index array
- RSIA: random starting index array
- N_{c}: number of rays in a ray cone
- I_{D}: direct normal solar irradiance
- S_{hsub}, S_{rsub}: areas of a microheliostat and a pixel
- ρ: heliostat reflectivity
- p: trimming ratio in TMS
- e_{RMS}: root mean square error

The remainder of the paper is organized as follows: Section 2 briefly reviews the related work on ray-tracing methods. Section 3 introduces the prerequisite knowledge for QMCRT. Section 4 describes the steps of QMCRT

in detail. In Section 5, the results generated via QMCRT are compared against captured photographs of the light reflected by a heliostat characterized by measured geometric data as well as the results of two well-known software tools and two GPU-based bidirectional ray-tracing methods on synthetic data. Section 6 presents the conclusion and discusses future work.

2. Related work

MCRT is a fundamental rendering technique to synthesize photorealistic images in computer graphics (Glassner, 1989; Henrik et al., 2003). This method involves tracing the paths of an enormous number of rays and computing the intersections between primary, reflective and refractive objects in a scene. Due to its effectiveness for simulating the propagation of rays, many studies have investigated the utilization of MCRT to simulate the radiative flux distribution on the receiver surface in a CRS. Currently, three types of MCRT methods are used in solar energy simulations: forward ray-tracing methods, reverse ray-tracing methods, and bidirectional ray-tracing methods.

2.1. Forward ray-tracing methods

Forward ray-tracing simulation methods constitute a straightforward and relatively mature technology used in solar energy simulation applications. In a typical forward ray-tracing method, an enormous number of rays are generated and emitted from the sun. Then, the rays hit the heliostats in the heliostat field, changing the direction under the laws of reflection, and reach the receiver surface. Although forward ray-tracing is intuitive and easy to implement, it is inefficient, since many rays either miss a heliostat or fail to reach the receiver.

Forward ray-tracing methods are predominantly adopted in simulation software tools such as SolTrace (Wendelin, 2003), Tonatiuh (Blanco et al., 2005; Mutuberria et al., 2011; Cardoso et al., 2018; Bonanos et al., 2019), and TieSOL (Izygon et al., 2011). SolTrace is a free software that can significantly accelerate the simulation process utilizing multithread processing. It supports various commonly used object geometries, such as flat or focused heliostats and flat or curved receiver surfaces. SolTrace only includes two built-in sunshape options, i.e., PillBox and Gaussian. The results of SolTrace are somewhat coarse since it does not support the Buie sunshape. Furthermore, no atmospheric attenuation models are considered in SolTrace. Tonatiuh is a powerful open-source ray tracer with multiplatform support. Tonatiuh has a user-friendly interface (Cardoso et al., 2018) and supports attenuation models, sunshapes, and various heliostat surfaces. The software also includes a heliostat surface with a measured geometry (Bonanos et al., 2019). Tonatiuh also possesses the advantage of multithread processing tailored for a multicore CPU. Since the system is designed to be highly multipurpose and capable of simulating not only CRS systems but also many other solar concentrating systems, both reflective and refractive, it is revealing that it is not particularly optimized for speed, as has been pointed out by others (Roccia et al., 2012). Finally, TieSOL is a commercial software package that can accelerate forward ray-tracing using a GPU implementation,and it can run under 1 s to trace 50–80 million rays on 3 GTX 570 GPUs.

2.2. Backward ray-tracing methods

To avoid tracing invalid rays, backward ray tracing methods were proposed. In a standard backward ray-tracing method, rays are generated from the receiver surface and traced in reverse to the heliostats and then to the sun or solar disk. Backward ray-tracing methods can be basically classified into discrete or continuous.

In a discrete approach, the radiative fluxes on each individual receiver surface pixel, i.e., a small tile on the receiver surface, are computed by accumulating the energies of all discrete rays irradiating it. Daly (Daly, 1979) used a discrete approach to simulate the radiative flux distribution on a circular cylindrical receiver. Guo

et al. (Guo et al., 2017) improved the method of computing the radiative flux value of each pixel by summing the energies of all rays connecting the center of that pixel to the center of a microheliostat. The energy of each ray is determined by the angle between the reflection direction and S_{dir}. However, the instability of the MaxRF value still arises in such methods.

In continuous approaches, the energy on each pixel is determined by integrating the solar energy intensity in the corresponding area of the sun disk. The flat heliostat (Pancotti, 2007) became crucial to such an approach since it can simplify the reflection procedure by regulating symmety between the computational virtual image of the sunshape and the heliostat. The radiative flux distribution can therefore be computed based on the irradiation distribution of the virtual sunshape. By exploiting the parallel computing power of a heterogeneous multi-GPU system, Chiesi et al. (Chiesi et al., 2013) proposed a backward method that could speed up the process by a factor of 52. Although continuous methods can generate smooth results and stable MaxRF values, they are not suitable for addressing nonflat heliostats.

The backward ray-tracing method saves computing resources by avoiding the unneccessary calculation of the rays' interception with the receiver. However, in a modern CRS, the size of the receiver surface is much larger than the projected area of a heliostat to ensure that the reflected solar energy will be absorbed as much as possible. As a result, when calculating the radiative flux distribution on the receiver surface reflected by a heliostat via backward ray-tracing, only a small portion of the receiver surface is involved, which means most calculations are wasted.

2.3. Bidirectional ray-tracing methods

In a bidirectional ray-tracing method, rays are projected directly on the primary reflection surface or surfaces. The incident directions of the rays are subject to the given sunshape distribution around S_{dir}. If the rays are not shadowed by other heliostats, their reflection directions will obtain slight deviations from the ideal normal. The main advantage of the bidirectional approach lies in its computational efficiency. The bidirectional approach not only eliminates the calculation of the rays' intersection with primary reflection surfaces but also avoids the calculation of invalid rays impinging on the ground. In Belhomme's method (Belhomme et al., 2009), the centers of the microheliostats are the origins of the rays. Thus, their approach is called grid ray tracing, or GRT. GRT can achieve a simulation speed of one million rays traced per second on a 2-core PC. Ulmer et al. Ulmer et al. (2011) improved the simulation precision by introducing an optical method of measuring heliostat slope errors. They further enhanced the approach by utilizing the input of geometric data from real heliostats. However, even on a GPU, the on-the-fly generation of an enormous number of random numbers is expensive. As a compromise, bidirectional ray-tracing methods typically generate a small number of random rays for all microheliostats. Consequently, the GRT algorithm often generates results that show distinct patterns.

Among all the existing discrete ray-tracing methods described above, one common problem arises from all of the simulation results: the MaxRF value varies among simulations. The reason lies in the discreteness of the MCRT method and the fact that the sampled random arguments vary each time. Because the MaxRF value is important for parameter fitting in analytical methods (Collado, 2010), an efficient and accurate simulation method that generates a reliable MaxRF value is required.

3. Preliminary knowledge and terminology

Several terms and concepts that are frequently used in describing the proposed QMCRT method. For clarity and convenience, the key terms and concepts will be discussed briefly.

3.1. Root mean square error

The metric used to evaluate the similarity between the reference distribution and simulated distributions is the root mean square error (RMSE). The equation for calculating the RMSE is as follows:

$$e_{\mathrm{RMS}} = \sqrt{\frac{1}{WH}\sum_{w=1}^{W}\sum_{h=1}^{H}[\hat{g}(w,h) - g(w,h)]^2} \tag{1}$$

where W and H represent the resolution of the receiver surface in terms of the width and height, respectively, and $\hat{g}(w,h)$ and $g(w,h)$ are the radiative flux in the reference result and the simulation result, respectively, at position (w,h). Generally, the smaller the RMSE is, the more indistinguishable the simulated radiative flux distribution is to the reference distribution. In this paper, when a simulation result is said to be accurate, precise, or of high quality, it suggests a low RMSE value.

3.2. Coordinate systems

There are two types of right-handed coordinate systems involved in the QMCRT process: the global coordinate system (referred to as the **xyz** system in the following) and the local coordinate systems. The heliostats and receiver surface are defined in the global coordinate system, and the precomputed directions saved in the SSP and MHNP are defined in the local coordinate system. Conversion between different coordinate systems can be accomplished via affine transformations. In addition, a uniform 3D space partition, i.e., a 3D grid, in the **xyz** system is generated for the heliostat field in the QMCRT algorithm. Each element in this 3D grid is called a voxel.

3.3. Heliostat orientation adjustment and rotation

To make full use of the available solar energy, the orientation of each heliostat should be adjusted to track the movement of the sun. To maximize the reflected solar energy concentrated on the receiver, the normal to each heliostat is adjusted to align with N, the vector bisecting the angle between S_{dir} and R_{dir}, as shown in Figure 1a. In this paper, the azimuth-elevation heliostat tracking model Chen et al. (2004) will be used. Other tracking models are feasible, as well, since the proposed flux simulation method is independent of the heliostat adjustment style.

3.4. Sampled variables in QMCRT

In the proposed QMCRT algorithm, four random parameters are sampled. Two of them describe the direction of the incident light from the sun, i.e., φ and θ, as illustrated in Figure 1b. The Buie sunshape model is adopted to describe the solar intensity distribution $S(\theta)$. Other models can also be applied and are elaborated by Wang et al. (2018, 2020), such as pillbox model, Gaussian model, calibrated Buie sunshape model, etc. φ follows a uniform distribution in the range of $[0, 2\pi)$. φ and $S(\theta)$ are fromulated in Eqs. (2) and (3), respectively:

$$\varphi \sim U(0, 2\pi) \tag{2}$$

$$S(\theta) = \begin{cases} \dfrac{\cos(0.326\theta)}{\cos(0.308\theta)}, & \{\theta \in R | 0 \le \theta \le 4.65 \text{ mrad}\} \\ e^{\kappa}\theta^{\gamma}, & \{\theta \in R | \theta > 4.65 \text{ mrad}\} \end{cases} \tag{3}$$

$$\kappa = 0.9\ln(13.5\chi)\chi^{-0.3}, \quad \gamma = 2.2\ln(0.52\chi)\chi^{0.43} - 0.1$$

where χ is CSR specifying the Buie sunshape. The other two parameters, α and β, represent the distribution of the microheliostat normals, as illustrated in Figure 1c. α is identical to φ, and β is the arctangent of β', which follows a Gaussian distribution with a given σ, as shown in Eqs. (4) and (5), respectively:

$$\alpha \sim U(0, 2\pi) \tag{4}$$

$$\beta' \sim N(0, \sigma^2), \quad \beta = \arctan(\beta') \tag{5}$$

The value of σ depends on the flatness of the heliostat surface.

Although the origins of the rays are uniformly sampled on the heliostat in QMCRT, i.e., each ray is assumed to be directly projected to the center of its corresponding microheliostat, the experiments reported in Sections 5.1.1 and 5.2.2 prove that QMCRT can still generate accurate results efficiently.

3.5. Sampling method: ITS

In addition to the ray-tracing algorithm, the sampling density and the chosen sampling method also play important roles in determining the simulation quality. The higher and better the sampling density and the method are, the more accurate the results are. Current CPUs and GPUs support both uniform and Gaussian random generators. These two built-in random generators are equipped to deal with simple random variables, such as φ, α and β, but they become inadequate in regard to more complex variables such as probability density functions (PDFs), in this case, θ. Thus, the question of how to efficiently and effectively sample an arbitrary PDF using the simple random generators on a CPU or a GPU is a key problem to QMCRT.

One commonly used approach is MCMC sampling. The MCMC sampling method can be used to achieve complex PDFs from simple PDFs through rejection laws. For a given PDF, these rejection laws are continuously applied until the number of sampling data reaches a specified quota. This method is straightforward and easy to implement but time- consuming. Furthermore, its sequential nature prevents it from GPU implementation.

To compensate the flaws, the ITS (Inverse Transform Sampling) method is adopted in QMCRT. This method is more efficient and scalable for GPU implementation than the MCMC method. There are four steps in the ITS method: For each given PDF, its cumulative density function (CDF) is calculated. Next, the inverse function of the CDF is computed. Next, the CDF is uniformly sampled on the interval [0,1], where u is the sampling variable generated by this simple generator, as shown in Eq. (6):

$$u \sim U(0, 1), \quad u = \mathrm{CDF}(x) = \int_{-\infty}^{x} \mathrm{PDF}(x')dx' \tag{6}$$

where u is the sampling variable generated by this simple generator. Finally, the value of the random variable x corresponding to u is calculated via Eq. (7):

$$x = \mathrm{CDF}^{-1}(u) \tag{7}$$

As a result, x becomes the given distribution of PDF.

Sometimes, the PDF might be nonintegrable. In such a case, the CDF will be calculated numerically. K histograms are thus computed from the PDF and are combined to form cumulative histograms. When K is sufficiently large, the cumulative histograms will approach the corresponding CDF. Then, two consecutive cumulative histograms are found such that u falls between them, and x is obtained via linear interpolation between these two cumulative histograms. In QMCRT, θ is sampled using the numerical ITS method.

4. Quasi-Monte Carlo ray tracing

Current MCRT methods require tracing a large number of rays. Thus, these methods are generally time-consuming when implemented on the CPU. Nevertheless, the resulting MaxRF value is unstable and can even be 10% higher than the reference value due to the low sampling density. In the proposed QMCRT algorithm, a compromise between efficiency and accuracy is reached elaboratly by pregenerating the random variables in the SSP and MHNP and reusing them in a combined way. Then, a postprocessing smoothing step is applied to generate a stable MaxRF. In this way, precise simulation results with a stable MaxRF value can be efficiently obtained.

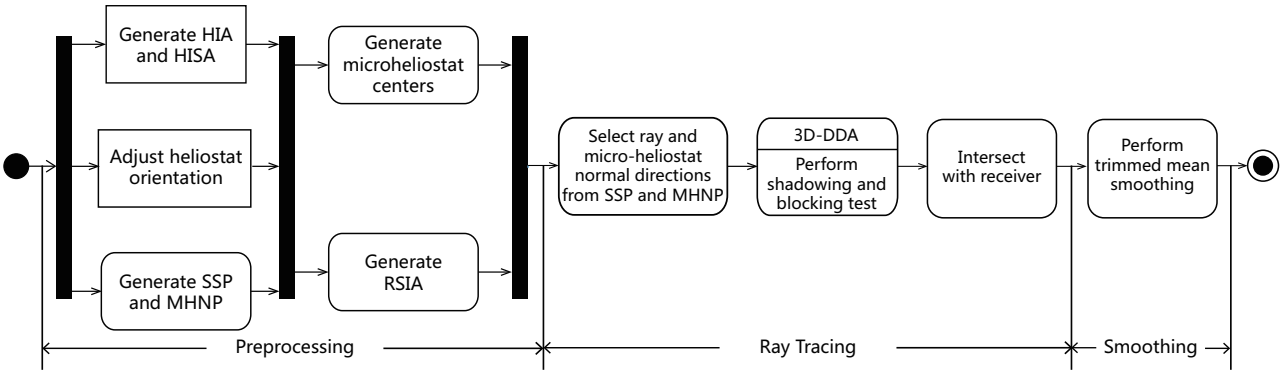


Figure 2: Flowchart of QMCRT for a given moment of time, where the operations presented in square-cornered boxes are implemented on the CPU, while the operations presented in boxes with rounded corners are implemented on the GPU.

The flowchart of the QMCRT algorithm for a given moment of time is illustrated in Figure 2. QMCRT consists of three stages: preprocessing, ray tracing, and smoothing. In preprocessing phase, the data to be used in the subsequent ray-tracing stage are calculated. Then, in the ray tracing stage, the traces of the incident solar rays in the heliostat field are simulated, and the preliminary radiative flux distribution is generated. Finally, a smoothing operation is applied to the radiative flux distribution. The details of the above operations will be disclosed in the following sections.

4.1. Preprocessing

The preprocessing phase of QMCRT consists of two steps. The first preprocessing step consists of three operations, each of which can be executed in parallel sequences:

(1) The heliostat index array (HIA) and the heliostat starting index array (HSIA) are set up to save information on the correspondence between the voxels and the heliostats.
(2) Adjust the heliostat orientations so that the reflected light will be focused on the receiver surface.
(3) Precompute the random variables α, β, φ and θ and store them in the MHNP and SSP.

The second step of preprocessing contains two parallel operations:

(1) Tessellating each heliostat into microheliostats, the centers are then chosen as the origins of the incident rays.
(2) Generate the random starting index array (RSIA) to assist in selecting directions from the SSP and MHNP.

The data processed in the first preprocessing step apply universally to all heliostats in the heliostat field, while the data processed in the second step vary among the heliostats. The method of heliostat orientation adjustment is discussed in Section 3.3, and the generation of the RSIA is introduced in Section 4.2.1. Further discussion for these two steps is omitted.

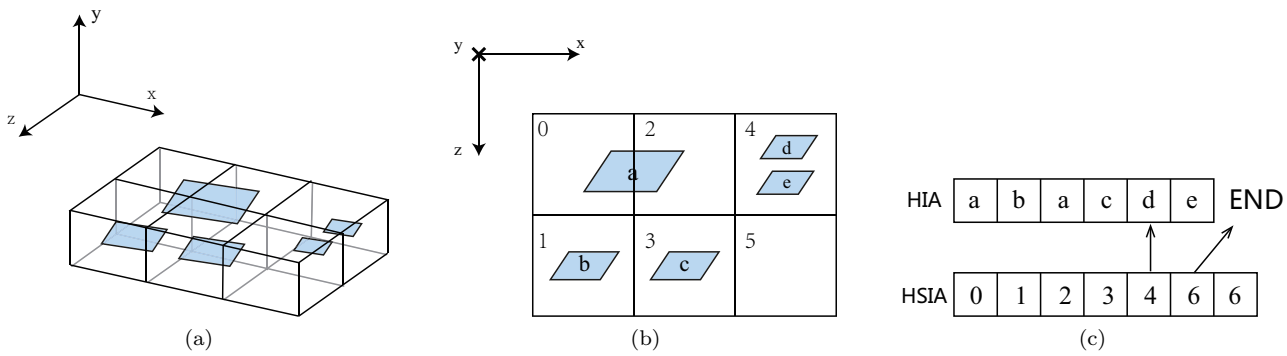


Figure 3: Illustration of the correspondence information between the voxel indices and heliostat indices for 3D-DDA traversal. An illustrative heliostat field is subdivided into 6 voxels (a). Its 2D projection in the **xz** plane is shown in (b), and its HIA and HSIA are shown in (c). For example, the 4th voxel contains heliostats d and e. Therefore, the values of HSIA[4] and HSIA[5] are 4 and 6, respectively.

4.1.1. 3D-DDA for ray traversal

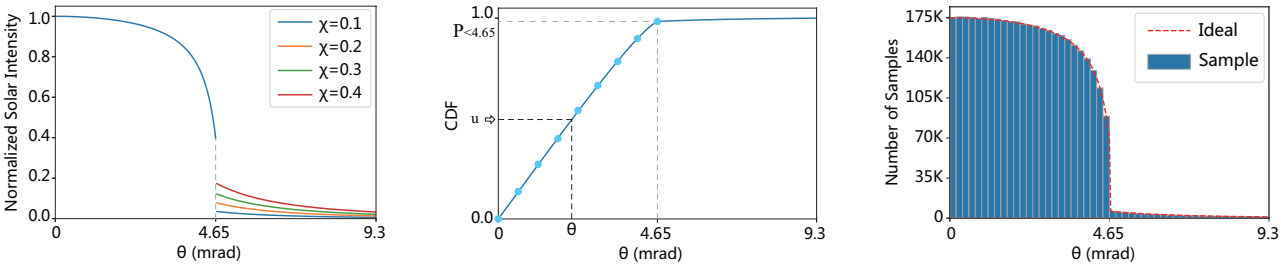
The 3D-DDA ray-tracing algorithm (Amanatides and Woo, 1987) is used for ray-heliostat collision detection. First, the algorithm constructs some auxiliary data as a premise (He et al., 2017). The space of heliostat field is uniformly divided into 3D voxels thereafter. Second, an axis-aligned cubical bounding box is first computed for each heliostat; each center lies at the heliostat center and each edge length is the diagonal length of the heliostat surface. The heliostat lies within the cubical bounding box regardless of the heliostat's rotation. If a cubical bounding box intersects with a voxel, the associated heliostat is assigned to the voxel. In this way, the heliostat-voxel relationship remains unchanged with respect to the heliostat rotation at any moment.

For illustration, the 6-voxel division of a heliostat field that contains 5 heliostats is shown in Figure 3a. Each voxel can be linked to an arbitrary number of heliostats. One heliostat may cross several voxels, as shown for *heliostat a*. In QMCRT, the information on the correspondence between the voxel indices and the heliostat indices is saved into two 1D arrays in QMCRT, namely, HIA and HSIA (He et al., 2017). This adaptation is suitable for GPU implementation.

3D-DDA is perfectly suited for ray tracing in scenes where the objects are distributed relatively uniformly because the uniformly spaced voxel structure can greatly accelerate the ray and object intersection calculations. A heliostat field is one such scene. This is why 3D-DDA is adopted in the proposed QMCRT algorithm. When performing shadowing and blocking tests using 3D-DDA (computing the intersections between rays and heliostats), the QMCRT algorithm first tests whether a ray intersects a neighboring voxel of the voxel in which the origin of the ray lies. If so, then the algorithm tests whether the ray intersects with any heliostat linked to the intersected voxel. This procedure is iteratively applied until the ray exits the boundary of the 3D grid. In this way, QMCRT performs shadowing and blocking tests efficiently.

4.1.2. SSP and MHNP generation

The SSP and MHNP are two random variable sets of the same size, which is considerably smaller than the number of random variables used in a typical MCRT algorithm. The elements in the SSP (θ and φ) and the MHNP (α and β) are factors that determine the directions of incident solar rays and microheliostat surface normals. Both should be accurately sampled to obtain precise results. While φ, α and β can be simply generated by the built-in engine, θ is generated via the ITS approach, as described in Section 3.5. As shown in Figure 4a, since few sampled θ values fall above 4.65 mrad for various χ values, QMCRT mainly samples the angle θ from the range of [0 mrad, 9.3 mrad]. The θ distribution below 4.65 mrad is nonintegrable and can be calculated numerically, as shown in Figure 4b. The distribution between 4.65 mrad and 9.3 mrad is integrable



(a) (b) (c)

Figure 4: (a) Buie sunshape distributions with various χ values. (b) CDF of the Buie sunshape with χ equal to 0.1. During the ITS process, θ is calculated discretely when u is less than $P_{<4.65}$; otherwise, it is calculated via Eq. 9. (c) Comparison of the ideal results with the ITS results for approximately 4 million variables. The close match between the sample distribution and the ideal curve proves the correctness of the sampling method.

and can be calculated numerically as sampled directly based on Eq. (8) and Eq. (9):

$$u = CDF(\theta) = \int_{\theta}^{0.00465} e^{\kappa}\theta'\gamma d\theta' + P_{<4.65} \tag{8}$$

$$\theta = \sqrt[\gamma+1]{\frac{(u-C)(\gamma+1)}{e^{\kappa}}} \tag{9}$$

where $P_{<4.65}$ is the CDF when θ is equal to 4.65 mrad and C is an integration constant. Figure 4c shows the results of sampling via the ITS method compared with the theoretical distribution. The similarity proves that the sampling results are accurate.

Table 1 shows the execution times in units of milliseconds (ms) for two sampling methods, i.e., MCMC sampling on a CPU and ITS on a GPU, for the same number of θ values. As shown in the table, the ITS method runs 191 times faster than the MCMC approach.

Table 1: Run-time comparison between MCMC sampling on a CPU and ITS on a GPU.

Number of Sampled θ Values	65,536	419,430	20,480,000
MCMC Sampling on a CPU (ms)	12.172	743.056	3573.033
ITS on a GPU (ms)	0.063	3.885	18.958

4.1.3. Origins of incident rays

In QMCRT, all the incident rays originate from the centers of microheliostats. The same number of rays are generated for each microheliostat, and all rays from the same microheliostat center form a ray cone. Thus, the number of ray origins depends only on the heliostat tessellation resolution. This approach can dramatically reduce the memory cost for the storage of ray data. For example, a 1.25 m×1.6 m heliostat with tessellation units of 0.01 m× 0.01 m only requires 20,000 origins. It is a significant reduction of more than 20 million origins. Subsequently, the reported experiments will show that similar simulation results can be obtained by either adopting the microheliostat centers as the ray origins or generating the ray origins fully randomly.

4.2. Quasi-Monte Carlo ray tracing

The ray-tracing stage consists of three steps: generating the incident solar ray directions and microheliostat surface normals via the SSP and MHNP; performing shadowing and blocking tests; and computing the

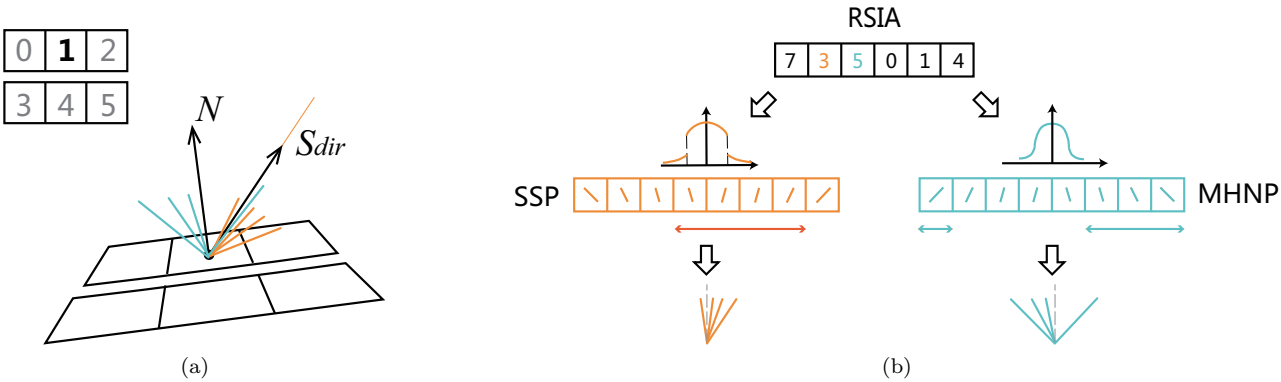


Figure 5: Illustration of how to generate incident rays and microheliostat surface normals from the SSP and MHNP in QMCRT. The example heliostat is divided into 6 microheliostats with the indices given in the upper left corner of (a). The size of both the SSP and the MHNP is 8, and N_c is equal to 4. Taking the 1st microheliostat as an example, the QMCRT algorithm will consecutively choose 4 rays and normals from the SSP and MHNP with starting indices of 3 and 5, respectively, in accordance with the RSIA, as shown in (b). Then, the selected elements are transformed into global coordinate values around S_{dir} or N.

intersections between the reflected rays and the receiver surface. The workflow of this stage is described as follows:

(1) For each microheliostat, randomly select a bundle of incident rays and a bundle of heliostat surface normals consecutively from the SSP and MHNP, respectively.

(2) Transform the sampled rays and heliostat surface normals from their local coordinate system into the global coordinate system.

(3) Perform shadowing and blocking tests via the 3D-DDA algorithm:

  (3.1) If a ray is shadowed, stop.

  (3.2) Otherwise, calculate its reflection direction with respect to the corresponding microheliostat normal.

  (3.3) If the reflected ray is blocked, stop.

  (3.4) Else, compute the intersected pixel and add the reflected energy to that pixel.

Since the shadowing and blocking tests are described in the section on 3D-DDA (Section 4.1.1), only the ray and surface normal direction generation and energy calculation steps will be introduced.

4.2.1. Generating incident ray directions and surface normals via the SSP and MHNP

After a microheliostat center is selected to serve as the current ray origin, a bundle of rays with the same origin is generated consecutively from the SSP with a random starting index. If a ray is shadowed, tracing stops; otherwise, the corresponding microheliostat surface normals are generated consecutively from the MHNP using another random starting position. The starting positions in the SSP and MHNP are all precomputed and saved in the RSIA by the built-in uniform distribution sampling engine of the GPU. The size of the RSIA is equal to the number of microheliostats. Figure 5 shows an example of how to generate ray directions and microheliostat surface normals for the initial microheliostat. Notably, since the elements in the RSIA are independently and identically distributed, they can be reused by setting the next element in the RSIA as the microheliostat surface normal starting index.

4.2.2. Ray energy density calculation

When a reflected ray intersects with the receiver surface, the corresponding pixel in which the intersection point lies is determined. The energy density associated with the reflected ray is computed using Eq. (10):

$$E_{\text{ray}} = \frac{I_{\text{D}} \cdot S_{\text{hsub}} \cdot \cos\phi \cdot \rho}{N_{\text{c}} \cdot S_{\text{rsub}}} \cdot \eta_{\text{aa}} \ (\text{W/m}^2) \tag{10}$$

where I_{D} is the direct normal solar irradiance in units of W/m^2; S_{hsub} and S_{rsub} are the areas of a microheliostat and a pixel, respectively; ϕ is the angle between S_{dir} and the microheliostat normal; ρ is the reflection ratio; N_{c} is the number of rays in the ray cone; η_{aa} is an atmospheric attenuation factor calculated as shown in Eq. (11), based on an empirical atmospheric attenuation model (Leary and Hankins, 1979).

$$\eta_{\text{aa}} = \begin{cases} 0.99321 - 0.0001176 \cdot d + 1.97 \cdot 10^{-8} \cdot d^2 & d \leq 1000 \text{ m} \\ e^{-0.0001106 \cdot d} & d > 1000 \text{ m} \end{cases} \tag{11}$$

where d is the distance. The radiative flux of each pixel is contributed by the total energy density of all rays hitting the corresponding pixel region.

4.3. Simulation result smoothing

As mentioned in Section 1, even if more than 20 million rays are traced from a single heliostat, the simulated MaxRF value can still vary across different runs. In QMCRT, a smoothing operation is applied to address this problem. This approach is inspired by the denoising techniques that are frequently used in image processing (Lee, 1983). Essentially, the radiative flux distribution is treated as an image, and the underlying principle of the smoothing process is to perform a low-pass filter operation on this noisy image. Among low-pass filters, the TMS approach is selected as the postprocessing filter because of its excellent performance in processing the simulation results. This approach calculates the mean of the neighboring pixels that form the kernel, by which the minimal and maximal $p\%$ of the pixels are discarded. Thus, mean smoothing is a special case of TMS in which the trimming ratio p is equal to zero.

To implement TMS, it is necessary to eliminate the $p\%$ minimal and maximal values in each kernel. Two heaps are used to record the minimal and maximal values during iteration in the form of a max-heap and a min-heap of the same size, respectively, over all elements in the kernel. The max-heap is a complete binary tree in which the value of each node is no less than the value of its two subnodes. The min-heap is similar to the max-heap, except that the value of each of its nodes is no greater than the values of the two corresponding subnodes. In QMCRT, the max-heap is used to save the minimal values in the kernel, while the min-heap is used for the maximal values. The two heaps are both registered in the shared GPU memory for computational acceleration.

As a result of the TMS operation, the MaxRF value will approach the reference value; however, the total power will change slightly due the rejection of some elements. Nevertheless, the change in the energy is very small, i.e., less than 0.5% in general. If the user is concerned about the total power exceeding the MaxRF value in the simulation result, then p can be set to zero. In this case, the TMS operation degenerates into the mean smoothing operation that preserves a constant total power during processing.

5. Validation and comparison

The proposed QMCRT algorithm has been implemented on a desktop PC equipped with an Intel(R) Core(TM) i7-6700K CPU running at 4.00 GHz and an NVIDIA GTX 1070 GPU. All code was written in

Table 2: Details of the sun, receiver and experimental heliostat configurations for single-heliostat experiments.

	Parameter	*Heliostat 1*	*Heliostat 2*
Sun	Altitude and Azimuth	53.87°, 258.02°	41.67°, 263.29°
	I_{D} (W/m^2)	1000	
	χ	0.1	
	Kernel Size	11×11	
	p	3	
	Size of SSP or MHNP	32768	
Receiver	Size (m)	4.6 × 4.6	
	Pixel Size (m)	0.01 × 0.01	
	Location (m)	0.0, 55.0, 0.0	
	Normal	0.7071, 0.0, 0.7071	0.7071, 0.0, -0.7071
Heliostat	Location (m)	90.972, 3.911, 20.967	84.046, 6.292, -35.129
	Distance from Receiver (m)	96.368	99.4580
	Size (m)	1.25×1.6	
	ρ	0.88	
	Microheliostat Unit Size (m)	0.01	
	σ (mrad)	1	
	N_{c}	1024	

C++ and CUDA 8.0 and compiled in Visual Studio 2017 under the 64-bit Windows 7 operating system. Similar algorithms, i.e., MCRT and GRT, were also implemented for comparison. The simulation software packages considered for comparison, i.e., SolTrace and Tonatiuh, were also evaluated on the same desktop PC.

To evaluate the correctness and efficiency of QMCRT, scenarios with both a single heliostat and multiple heliostats in a heliostat field were simulated. For the single-heliostat simulations, QMCRT, two other GPU-based bidirectional ray-tracing algorithms (MCRT and GRT) and two widely used simulation software tools (SolTrace and Tonatiuh) were first implemented based on synthetic data for a flat rectangular heliostat. Then, the measured surface data and captured radiative flux distribution for an experimental heliostat were used to further validate the correctness of the QMCRT method. To demonstrate the performance of the proposed algorithm, QMCRT, SolTrace and Tonatiuh were also applied to the case of a large heliostat field with 6282 heliostats. Detailed comparisons among these methods are given in terms of the MaxRF value, RMSE, total power, contour map and execution time.

5.1. Performance and comparison for single-heliostat simulations

In this section, two heliostat configurations will be used to compare QMCRT with other algorithms, simulation software tools and captured photographs. The details of the heliostat configurations, receiver and sun are listed in Table 2.

5.1.1. Synthesized heliostat results

First, the synthetic flat *Heliostat 1* will be used for performance comparisons between QMCRT and the other state-of-the-art methods. Considering that SolTrace does not support the Buie sunshape or atmospheric attenuation, SolTrace and Tonatiuh were implemented with the PillBox sunshape with no transmissivity decay.

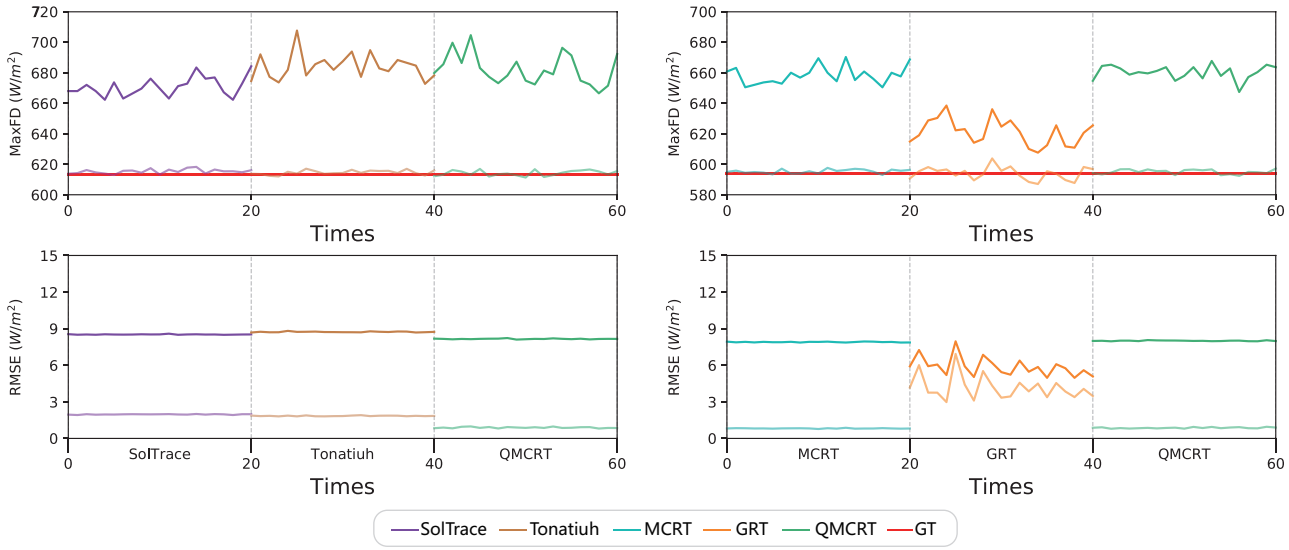


Figure 6: MaxRF and RMSE values for the simulation results obtained with the five methods for *Heliostat 1*. The first column shows the MaxRFs and RMSEs from 21 simulations each using SolTrace, Tonatiuh and QMCRT with the PillBox sunshape and no transmissivity decay; the second column shows the results of MCRT, GRT and QMCRT with the Buie sunshape. The reference value of MaxRF is shown in red. The MaxRFs and RMSEs after the smoothing operation are shown in corresponding lighter colors. Before the smoothing operation, the MaxRF values obtained by these algorithms are always greater than the reference value. After the TMS operation, the MaxRFs approach the reference value, and the RMSEs greatly decrease, except for GRT because of its distinctly patterned results due to the low sampling density.

The PillBox sunshape distribution for β is given in Eq. (12):

$$\theta = arctan(0.00465\sqrt{u}) \tag{12}$$

Both GRT and bidirectional MCRT use the Buie sunshape with the MCMC sampling algorithm implemented on the CPU. QMCRT simulations were performed under both conditions. The reference distributions were also computed for both sunshapes by averaging the results of 1000 MCRT simulations.

MaxRF, RMSE and total power. Table 3 lists the statistical results of the five methods, including the total power, the MaxRFs and RMSEs before and after the TMS operation, and the execution time. With the TMS operation implemented, the data in Table 3 show that the proposed QMCRT method can generate simulation results with more accurate MaxRF and truncated RMSE compared to the previous methods.

In addition to QMCRT, the TMS operation also increases the accuracy of the MaxRFs and reduces the RMSEs for other algorithms. The data in Table 3 show that before the TMS operation, no matter which ray-tracing algorithms, the total power concentrated on the receiver is almost the same; however, the MaxRF value is greater than the reference value, and the RMSE value is large. This is caused by the discreteness and low sampling density inherent to these algorithms. The energy density of each ray is relatively large in sparse sampling density. Thus, the accumulated energy density error will be higher when a single ray incorrectly intersects with other pixels. However, since the rays are mainly distributed in a range of $0 \sim 4.65$ $mrad$ around S_{dir}, any incorrectly intersected pixel should be close to the correct position. As a result, a smoothing operation can reduce the uncertainty by taking the average of the neighboring pixel values. With that being said, not all neighboring pixels should be included in this operation; only those pixels with similar values should be considered. This is the reason for the satisfying performance of TMS in reducing the RMSE and increasing the accuracy of the MaxRF value.

Table 3: Means of total power, MaxRF, RMSE and execution time for five state-of-the-art algorithms.

Sun Shape Type	Algorithm	Before TMS			After TMS			Execution Time (ms)
		Total Power (W)	MaxRF (W/m^2)	RMSE (W/m^2)	Total Power (W)	MaxRF (W/m^2)	RMSE (W/m^2)	
PillBox	GT	1695.82	613.36	—	1695.82	613.36	—	—
	SolTrace	1694.54	682.28	8.35	1694.10	615.45	1.96	24254.12
	Tonatiuh	1695.66	684.61	8.73	1695.19	614.32	1.84	25912.52
	QMCRT	1695.83	681.94	8.15	1695.37	613.89	0.89	11.22
Buie	GT	1663.33	594.01	—	1663.33	594.01	—	—
	MCRT	1663.33	660.81	7.93	1662.86	594.98	0.81	4176.61
	GRT	1663.18	624.94	5.91	1663.04	593.82	4.56	9.98
	QMCRT	1663.32	658.74	7.99	1662.86	595.23	0.88	12.24

Figure 6 also shows the curves for the other algorithms before and after the TMS operation in 21 runs. Before TMS, the MaxRFs of all five algorithms are approximately $5\% \sim 11\%$ higher than the reference value. After TMS, all the MaxRFs are readjusted to the reference value. As a result, the RMSEs decreased significantly, except for GRT. In GRT, all bundles of rays emitted from a microheliostat share the same ray distribution; thus, the lack of a large number of samples leads to coarse simulation results. These results also illustrate that the TMS operation can reduce the noise in the simulation results provided that the sampling density should be reasonably high for the sunshape and heliostat surface slope error distributions.

In summary, we can draw the following conclusions from Table 3 and Figure 6

- Without the trimmed mean smoothing operation, the MaxRF values of the test algorithms are $5\% \sim 11\%$ greater than the reference MaxRF value, which is the average of 1000 MCRT simulation results. The total power values are almost identical to the reference value. The RMSE values are $5.91 \sim 8.73 \ \ W/m^2$.
- After the trimmed mean smoothing operation, the deviated MaxRF values decrease from $5\% \sim 11\%$ to $0.032\% \sim 0.34\%$. The RMSE values decrease from $5.91 \sim 8.73 \ \ W/m^2$ to $0.81 \sim 4.56 \ \ W/m^2$. However, the total power values are only $0.017\% \sim 0.1\%$ lower than the reference total value.
- Since the smoothing operation is performed in parallel on the GPU, the execution time is less than 2 ms. The execution time is related to the discretization resolution of receiver surface, the window size of the trimmed mean smoothing filter, and number of trimmed values.

Execution Time. The execution time statistics for all five methods are shown in the last column of Table 3. Compared with SolTrace and Tonatiuh, QMCRT runs more efficiently due to the tremendous parallel computing power of the GPU. MCRT runs significantly slower than QMCRT because both bidirectional MCRT and GRT algorithms sample the θ values via the MCMC method on the CPU. Although rays are generated on the CPU in GRT, it has a shorter execution time because GRT samples only a very small number of θ values (1024) compared to the number of values in the SSP (32768) in QMCRT, and it has fewer algorithmic steps than QMCRT.

Contour Map. Figure 7 and Figure 8 illustrate the contour maps (Elsayed et al., 1995; Huang and Sun, 2016) of the radiative flux distributions simulated with the five methods in comparison with the reference contour map. All of the contour maps are noisy before the smoothing operation. After TMS, the contour maps become smoother and approach the reference contour map, except for GRT. Due to its low sampling density, after

smoothing operation reduces the noise in all the simulation results,and GRT is still less accurate than the other algorithms. The consistency between the QMCRT and the reference contour map proves the superiority of the QMCRT algorithm. These results again indicate that the sampling densities for various distributions during ray tracing affect the quality of the simulation results.

Sampling Density. The sampling density in MCRT has a great impact on the simulation results, as mentioned in Section 3.4. The sampling densities for the various distributions in QMCRT are high due to the adoption of the RSIA, even though the sizes of the SSP and MHNP are considered smaller than the number of rays required to trace a single heliostat. Randomly and consecutively selecting elements from the SSP and MHNP is a feasible strategy that saves considerable GPU memory without sacrificing the accuracy of the results. The low RMSE of QMCRT further proves its dominance. Consequently, the greater the SSP and MHNP are, the better the results.

Another possible method of increasing the result accuracy is to adopt random ray origins in QMCRT. However, considering the time expense and the limited performance gains for such an approach, adopting fixed origins is more reasonable. If taking the example of a rectangle into account, the simplest way to generate random origin points within the rectangle is via random number generation engine. The sampling results are shown in Figure 9a. These result are not uniformly distributed because the built-in engine is a white noise sampler. Even an advanced sampling technique, e.g., a blue noise sampling method, could not produce satisfactory results. On the one hand, the efficiency of current built-in uniform sampling algorithms is not sufficiently high. They can be implemented either on a CPU (White et al., 2007) or with explicit sequential loops on a GPU (Ying et al., 2013; Wei, 2008). Furthermore, even with generated relatively uniform sampling results, there is still no guarantee for unbiased sampling (Yan et al., 2015). Figure 9b shows the results of a relatively uniform sampling algorithm (Wei, 2008). These results demonstrate that random ray origins have little impact on improving the accuracy of the simulation results even with a large quantity of traced rays, e.g., over 10 million. Therefore, in QMCRT, the microheliostat centers are still used as the origins of the ray cones.

Validation with Simple Setup Experiments. As indicated by Wang et al. (Wang et al., 2020), a single flat heliostat with a slope error that follows a Gaussian distribution, implements Buie sunshape and has a receiver collinear to the sun and heliostat will lead to an analytical distribution. These analytical distributions from simple setup experiments can be used for QMCRT validation. We also performed related experiments to investigate the effects of sunshape and surface slope error and verify the correctness of the QMCRT. In the sunshape related experiments, a heliostat with an ideal specular reflecting surface is assumed. The polynomial calibration for CSR regarding sunshape was also implemented. In the slope error verification, collimated rays from the sun are adopted. All of the implementation results are consistent with those from Tonatiuh, SolTrace, Tracer, Solstice, Heliosim and SolarPILOT in the paper (Wang et al., 2020). These analytical solution experiments verify the correctness of QMCRT from another aspect. Due to the limited space of the paper, readers can refer to the materials and results at GitHub (https://github.com/linxxcad/QMCRT).

5.1.2. Validation of measured heliostat data and corresponding radiative flux distributions

QMCRT was qualitatively validated against real heliostats and captured images. The experimental heliostat surface was measured, and a uniformly distributed point set was generated, as shown in Figure 10. The heliostat surface is not flat and has very small curvature.

The radiative flux distributions on the receiver surface for two heliostats, whose configurations are given in Table 2, were also captured as images by an industrial camera with all color channels. The first column of Figure 11 shows the captured images of the radiative flux distributions reflected by two experimental heliostats.

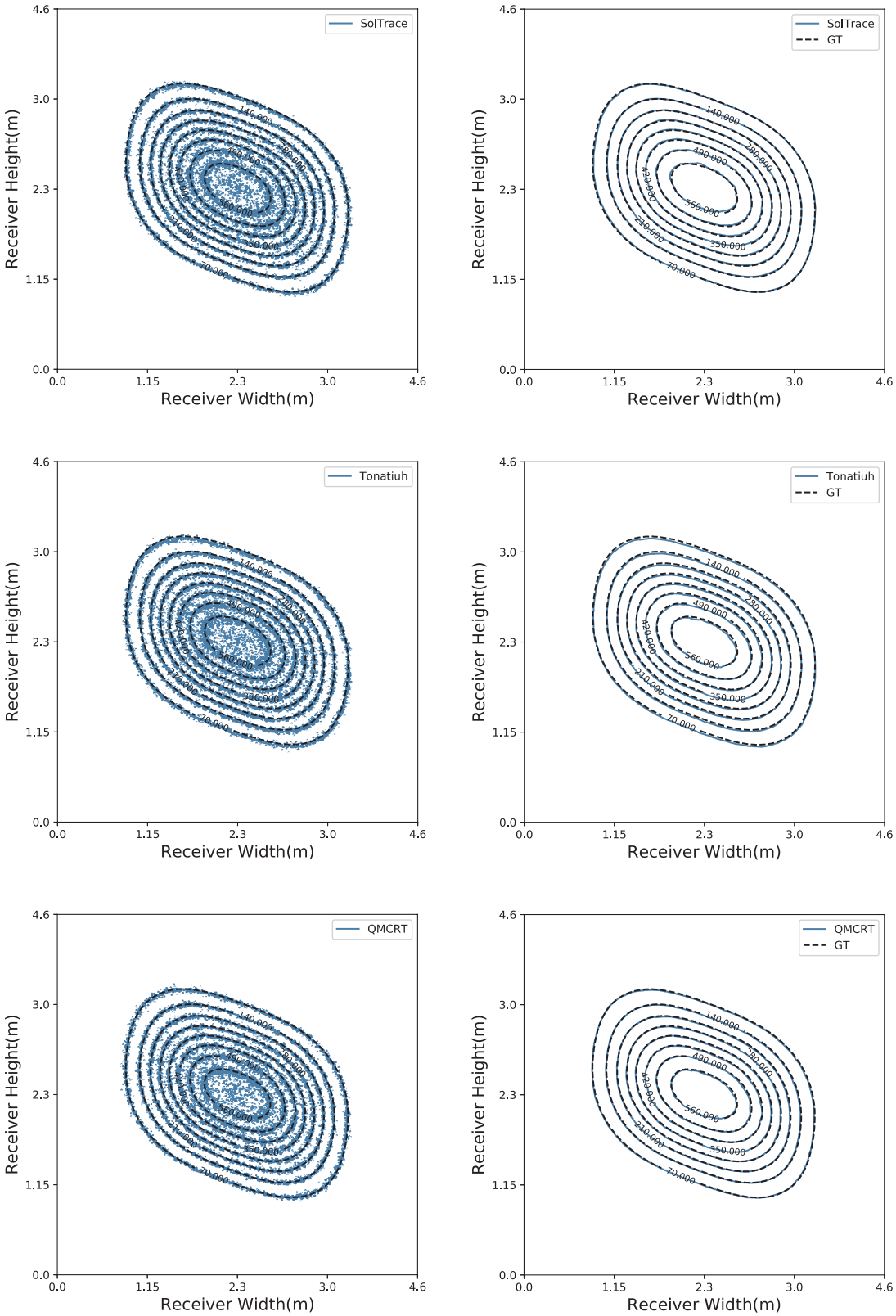


Figure 7: Contour maps of the radiative flux distribution reflected by *Heliostat 1*. The simulation results are shown as solid blue lines, while the GT map is presented as dashed black lines. The contour maps of SolTrace (first row), Tonatiuh (second row) and QMCRT (last row) with the PillBox sunshape and no transmissivity decay before (first column) and after (second column) TMS. The GT map is also shown in the second row for comparison.

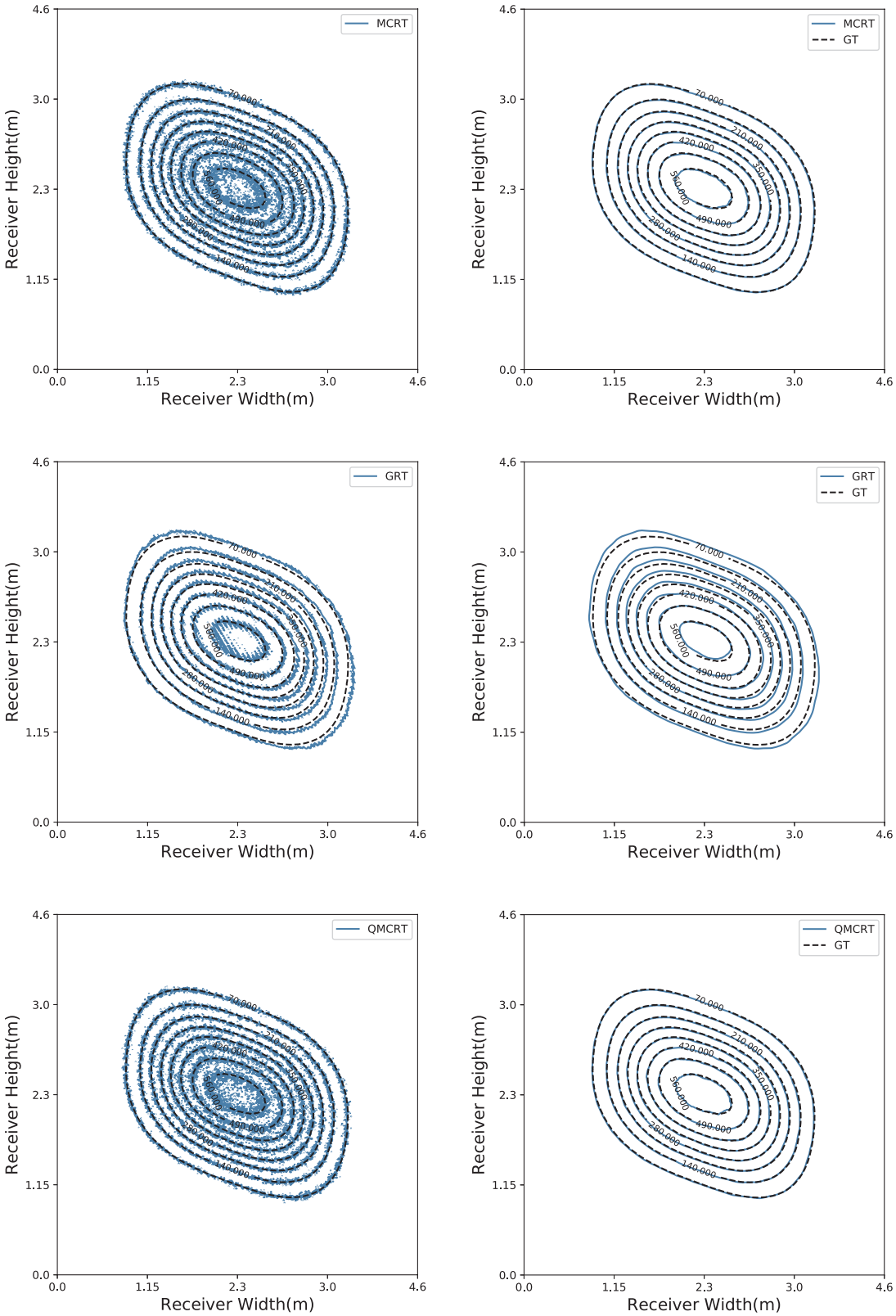
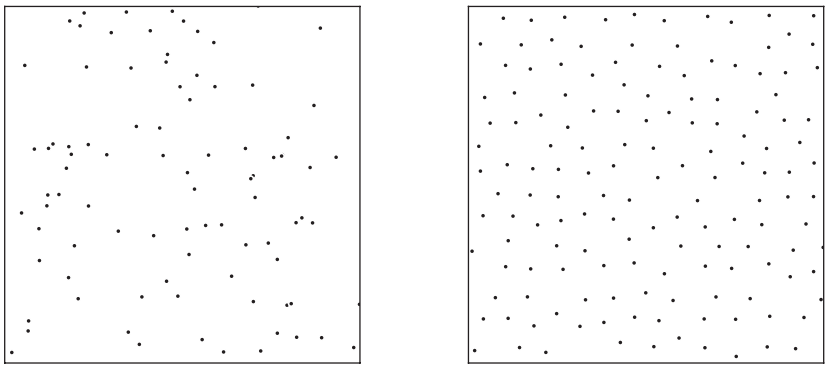


Figure 8: Contour maps of the radiative flux distribution reflected by *Heliostat 1*. The simulation results are shown as solid blue lines, while the GT map is presented as dashed black lines. The contour maps of MCRT (first row), GRT (second row) and QMCRT (last row) with the Buie sunshape before (first column) and after (second column) TMS. The GT map is also shown in the second row for comparison.



(a) Random sampling with the built-in sampler.

(b) Relatively uniform sampling via a blue noise algorithm (Wei, 2008).

Figure 9: Two sets of random sampling results in a 2D rectangle.

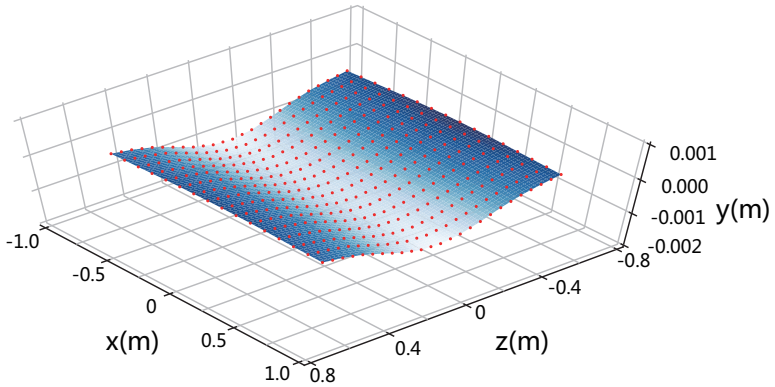


Figure 10: The raw measured points (red) on the experimental heliostat.

By removing the ambient light illumination and background color in the captured images and applying geometric calibration and color normalization, two processed images were obtained, as shown in the second column of Figure 11. The images in the third column of Figure 11 are the simulation results obtained via QMCRT based on the measured geometric data of the experimental heliostat. The radiative flux distributions in the simulation results are similar to those of the captured images. To enable further estimation of the simulation results, the contour maps of the captured images and simulation results are presented in the fourth column of Figure 11. Their distributions of contour maps are also similar.

5.2. Performance and comparison for simulations of a heliostat field

In the previous section, QMCRT was validated and assessed on the basis of a single heliostat. This section reports the results of applying QMCRT, SolTrace, and Tonatiuh to simulate a heliostat field. These results will be compared in terms of the radiative flux distribution, MaxRF, total power, and execution time.

5.2.1. Synthetic heliostat field

The distribution of the synthetic heliostat field Wendelin (2003) in the **xz** plane is shown in Figure 12. The **x**-axis and **z**-axis correspond to the south and east directions, respectively. The heliostat field contains 6282 flat heliostats, and the size of each heliostat is 4 m×3.2 m. A circular cylindrical receiver, with a height of 20 m and a radius of 10 m, is located at the origin of the **xz** plane, 180 m above the ground. No optimization of the

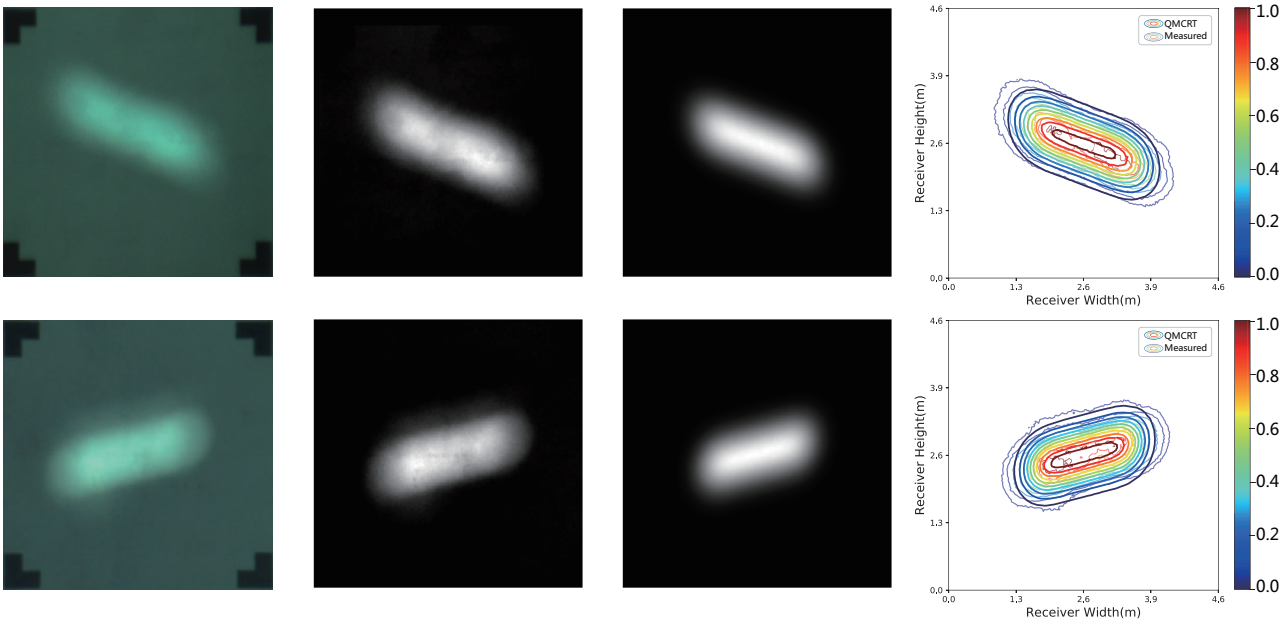


Figure 11: Comparison of the radiative flux distributions between captured images and QMCRT simulation results for two heliostat configurations, which are separately represented in the first and second rows. The first two columns show the captured photographs and the corresponding calibrated images with the ambient light and background color removed. The third column presents the QMCRT simulation results. The last column compares the contour maps of the QMCRT results and the measured data (in lighter colors). The similarity between the contour maps validates the correctness of QMCRT.

focusing strategy is applied here since it is beyond of the scope of this paper. The focal points of the heliostats are uniformly spread over the middle part of the receiver cylinder, i.e., a region of $[-5\,m, +5\,m]$in height. The simulation result is a rectangular image obtained by flattening the surface of the cylinder clockwise along the **z**-axis. More details of the heliostat field simulation are given in Table 4.

Table 4: Details of the sun, receiver and heliostat fields.

	Parameter	Value
Sun	Altitude and Azimuth	90°, 0°
	I_{D} (W/m^2)	1000
	χ	0.1
	Kernel Size	11×11
	p	3
Receiver	Height and Radius (m)	20, 10
	Pixel Size (m)	0.05
	Location (m)	0, 180, 0
Heliostat	Size (m)	4.0×3.2
	ρ	0.88
	Microheliostat Unit Size (m)	0.02
	σ (mrad)	1
	N_{c}	1

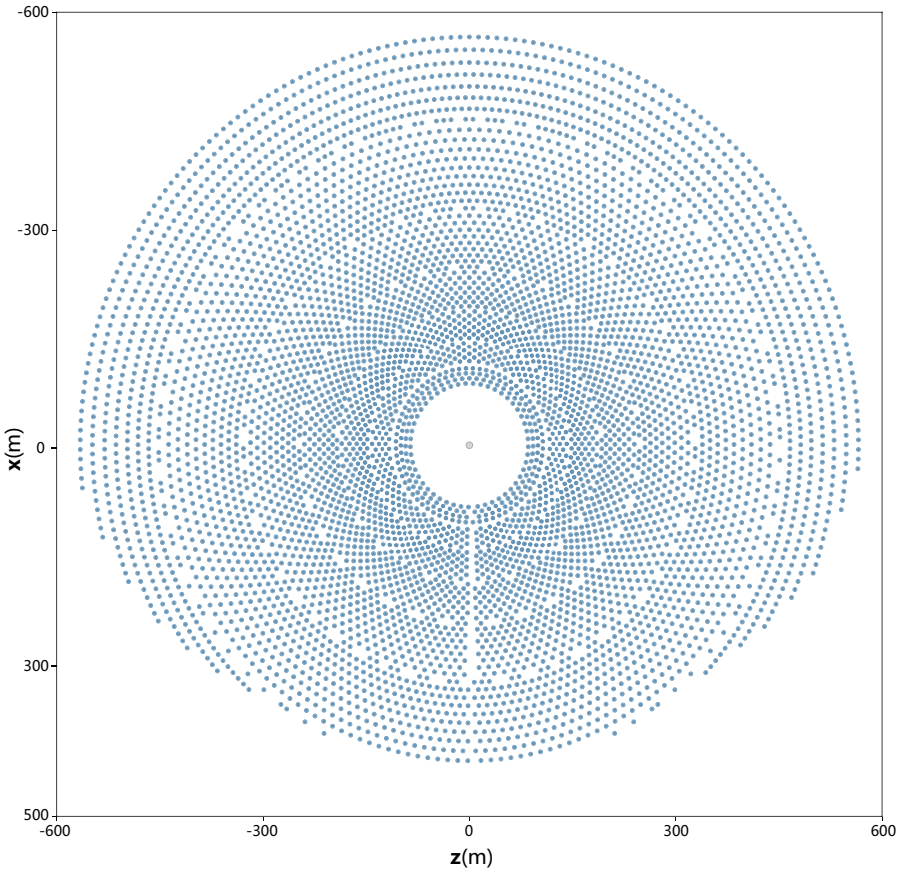


Figure 12: Distribution of the synthetic heliostat field Wendelin (2003) in the **xz** plane.

5.2.2. Heliostat field simulation results

Table 5 shows the statistical results of the three methods in terms of MaxRF, total power, and execution time. Because SolTrace does not support any atmospheric attenuation models, QMCRT was also implemented without transmissivity decay. The MaxRFs and total power values for the QMCRT before and after the TMS operation are also listed.

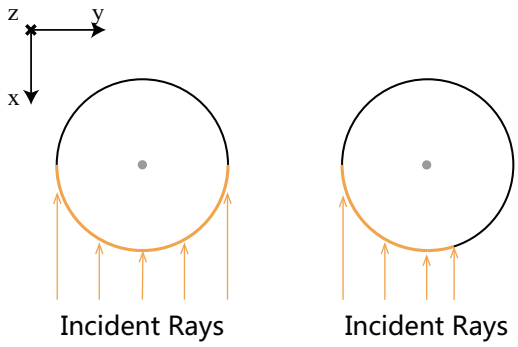
The MaxRFs of SolTrace and QMCRT without the smoothing operation are similar, but the total power of SolTrace is approximately 6% lower than that of QMCRT, even lower than in the case with atmospheric attenuation. Through careful inspection, a bug was found the intersections between light rays and the cylindrical receiver in SolTrace, as shown in Figure 13. In SolTrace, only some of the light rays along the **-x** direction can reach the receiver. Due to the fact that SolTrace can calculate 200 million rays in 32320 s(approximately 8.9 h) on a CPU, it seems infeasible to simulate a large-scale heliostat field using SolTrace on a CPU.

The MaxRF and total power values for Tonatiuh and QMCRT without smoothing have consistent results. Tonatiuh performs well on the CPU, taking only 178 s, since it uses the octree space acceleration algorithm. However, since Tonatiuh is a forward ray tracing method, approximately 5% of the generated rays from the sun can reach a heliostat. This software wastes considerable computational resources. For QMCRT, the total execution time is approximately 1.5 s, which can meet the requirements for practical applications.

Figure 14 shows the simulated radiative flux distributions of SolTrace, Tonatiuh and QMCRT on the surface of the cylindrical receiver. Although the same number of rays are used in each simulation, the results of SolTrace and Tonatiuh are coarse, while the QMCRT results are smooth due to TMS postprocessing.

Table 5: Statistical data on the results of SolTrace, Tonatiuh and QMCRT in terms of MaxRF, total power and execution time.

	Algorithm		MaxRF (kW/m^2)	Total Power (kW)	Execution Time (s)
No Transmissivity Decay	SolTrace		117.722	56860.91	32320
	QMCRT	Before TMS	121.587	60618.53	1.5
		After TMS	112.100	60615.81	
With Transmissivity Decay	Tonatiuh		116.657	57553.20	178
	QMCRT	Before TMS	116.484	57522.55	1.5
		After TMS	105.671	57520.29	



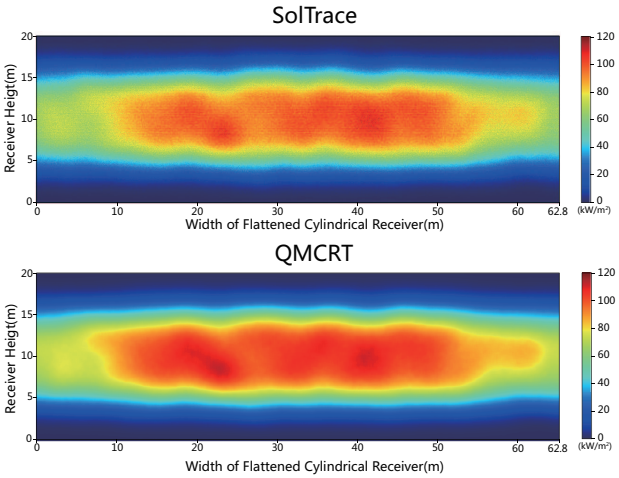
(a) (b)

Figure 13: Bug in SolTrace that arises when computing the intersections between rays and the cylindrical receiver. The expected lighted region when the rays are along the **-x** direction is colored in yellow in (a). However, only some of the rays can reach the expected region in SolTrace, as shown in (b).

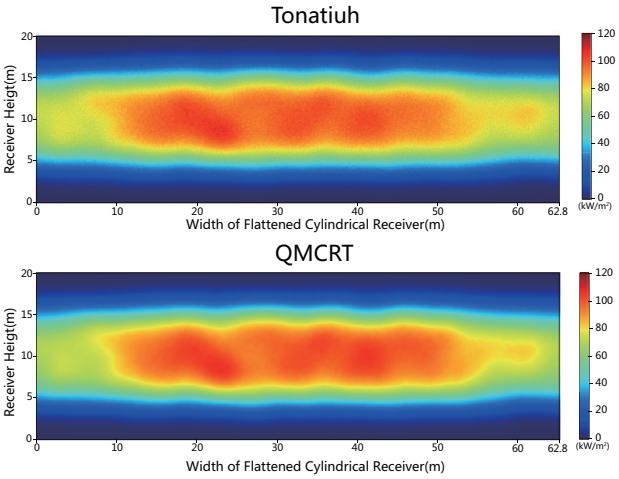
6. Conclusion

In this paper, a GPU-based Monte Carlo ray-tracing algorithm named QMCRT is proposed for simulating the radiative flux distribution on the surface of a receiver reflected by one or more heliostats. By computing and storing samples drawn from the sunshape and microheliostat normal distributions in advance, QMCRT achieves a higher simulation efficiency and sampling density by means of simple combinations of the precomputed values. By employing a suitable spatial structure, i.e., 3D-DDA, shadowing and blocking effects can also be checked efficiently. If a ray hits the receiver, its energy density(considering atmospheric attenuation) is added to the intersected pixel. To overcome the problem of unstable MaxRF values encountered in previous methods, a TMS operation is performed on the resultant radiative flux distribution. Notably, the discrete ITS and TMS methods adopted in QMCRT can also be applied to other MCRT simulation methods. Intensive experiments and comparisons for both single-heliostat and heliostat fields show that QMCRT is highly efficient (tracing 20-M rays within 13 ms) and accurate in terms of both the total power and MaxRF, achieving a low RMSE. In addition, QMCRT has been further validated using measured data.

In this paper, we only present the cases with the most classical CSR value ($\chi = 0.1$) for experiments due to limited space. In the future, more CSR values will be tested in the experiments. In addition, when the angle between the solar vector and the tower vector becomes larger or the distance between the heliostat and the receiver grows, the influence of the sunshape increases. In future work, researchers could take the relationship between these factors and simulation accuracy into consideration. Another avenue of future work may be to include a heliostat focusing strategy in QMCRT. In addition, QMCRT may be extended to the simulation of a very-large-scale heliostat field in parallel on a GPU cluster.



(a) Simulation results of SolTrace and QMCRT without transmissivity decay



(b) Simulation results of Tonatiuh and QMCRT with transmissivity decay

Figure 14: Simulation results of SolTrace, Tonatiuh and QMCRT for the synthetic heliostat field Wendelin (2003).

Acknowledgements

Special thanks are due to Jimmy Ding, who proofread the manuscript. This work was jointly supported by the National Key Research and Development Program of China (2017YFB0202203) and the National Natural Science Foundation of China (61772464).

References

Amanatides, J., Woo, A., 1987. A fast voxel traversal algorithm for ray tracing, in: Eurographics, pp. 3–10. doi:10.2312/egtp.19871000.

Behar, O., Khellaf, A., Kamal, M., 2013. A review of studies on central receiver solar thermal power plants. Renewable and Sustainable Energy Reviews 23, 12–39. doi:10.1016/j.rser.2013.02.017.

Behar, O., Khellaf, A., Mohammedi, K., 2013. A review of studies on central receiver solar thermal power plants. Renewable and sustainable energy reviews 23, 12–39. doi:10.1016/j.rser.2013.02.017.

Belhomme, B., Robert, P.P., Schwarzbözl, P., Ulmer, S., 2009. A new fast ray tracing tool for high-precision simulation of heliostat fields. Journal of Solar Energy Engineering 131, 031002:1–031002:8. doi:10.1115/1.3139139.

Bendt, P., Rabl, A., 1981. Optical analysis of point focus parabolic radiation concentrators. Applied Optics 20, 674–683. doi:10.1364/AO.20.000674.

Biggs, F., Vittitoe, C.N., 1979. Helios model for the optical behavior of reflecting solar concentrators. NASA Scientific and Technical Information 79.

Blanco, M.J., Amieva, J.M., Azael, M., 2005. The tonatiuh software development project: An open source approach to the simulation of solar concentrating systems, in: American Society of Mechanical Engineers 2005, pp. 157–164. doi:10.1115/IMECE2005-81859.

Bonanos, A.M., Faka, M., Abate, D., Hermon, S., Blanco, M.J., 2019. Heliostat surface shape characterization for accurate flux prediction. Renewable Energy 142, 30–40. doi:10.1016/j.renene.2019.04.051.

Box, G.E.P., Muller, M.E., 1958. A note on the generation of random normal deviates. Annals of Mathematical Statistics 29, 610–611. doi:10.1214/aoms/1177706645.

Buie, D., Monger, A.G., Dey, C.J., 2003. Sunshape distributions for terrestrial solar simulations. Solar Energy 74, 113–122. doi:10.1016/S0038-092X(03)00125-7.

Cardoso, J.P., Mutuberria, A., Marakkos, C., Schoettl, P., Osório, T., Iñigo, L., 2018. New functionalities for the tonatiuh raytracing software, in: AIP Conference Proceedings 2018, p. 210010. doi:10.1063/1.5067212.

Chen, Y.T., Kribus, A., Lim, B.H., Lim, C.S., Chong, K.K., Karni, J., Buck, R., Pfahl, A., Bligh, T.P., 2004. Comparison of two sun tracking methods in the application of a heliostat field. Journal of Solar Energy Engineering 126, 638–644. doi:10.1115/1.1634583.

Chiesi, M., Vanzolini, L., Scarselli, E.F., Guerrieri, R., 2013. Accurate optical model for design and analysis of solar fields based on heterogeneous multicore systems. Renewable Energy 55, 241–251. doi:10.1016/j.renene.2012.12.025.

Collado, F.J., 2010. One-point fitting of the flux density produced by a heliostat. Solar Energy 84, 673–684. doi:10.1016/j.solener.2010.01.019.

Conroy, T., Collins, M.N., Fisher, J., Grimes, R., 2018. Thermohydraulic analysis of single phase heat transfer fluids in csp solar receivers. Renewable Energy 129, 150–167. doi:10.1016/j.renene.2018.05.101.

CUDA 8.0, 2016. URL: https://devblogs.nvidia.com/cuda-8-features-revealed/. [Online; accessed 1-September-2017].

Daly, J.C., 1979. Solar concentrator flux distributions using backward ray tracing. Applied Optics 18, 2696–2699. doi:10.1364/AO.18.002696.

Duffie, J.A., Beckman, W.A., Mcgowan, J., 2013. Solar Engineering of Thermal Processes. John Wiley & Sons.

Elsayed, M.M., Fathalah, K.A., Al-Rabghi, O.M., 1995. Measurements of solar flux density distribution on a plane receiver due to a flat heliostat. Solar Energy 54, 403–411. doi:10.1016/0038-092X(95)00010-O.

Garcia, P., Ferriere, A., Bezian, J.J., 2008. Codes for solar flux calculation dedicated to central receiver system applications: A comparative review. Solar Energy 82, 189–197. doi:10.1016/j.solener.2007.08.004.

Glassner, A.S., 1989. An introduction to ray tracing. 1st ed., Academic Press.

Guo, M., Sun, F., Wang, Z., 2017. The backward ray tracing with effective solar brightness used to simulate the concentrated flux map of a solar tower concentrator, in: SolarPACES 2016, pp. 030023:1–030023:9. doi:10.1063/1.4984366.

He, C., Feng, J., Zhao, Y., 2017. Fast flux density distribution simulation of central receiver system on gpu. Solar Energy 144, 424–435. doi:10.1016/j.solener.2017.01.025.

Henrik, W.J., James, Arvo, P.D., Alexander, K., Art, O., Matt, P., Peter, S., 2003. Monte Carlo Ray Tracing. Technical Report. ACM SIGGRAPH Computer Graphics. URL: http://www.citidel.org/bitstream/10117/5376/1/86-mcrt-sg03c.pdf.

Huang, W., Sun, L., 2016. Solar flux density calculation for a heliostat with an elliptical gaussian distribution source. Applied Energy 182, 434–441. doi:10.1016/j.apenergy.2016.08.082.

Imenes, A.G., Stein, W., Hinkley, J., Benito, R., Bolling, R., Schramek, P., Ulmer, S., 2006. Ray tracing and flux mapping as a design and research tool at the national solar energy centre, in: Australia and New Zealand Solar Energy Society 2006. doi:10.13140/2.1.1309.0248.

Inverse Transform Sampling, . URL: https://en.wikipedia.org/wiki/Inverse_transform_sampling. [Online; accessed 6-June-2018].

Islam, M.T., Huda, N., Abdullah, A.B., Saidur, R., 2018. A comprehensive review of state-of-the-art concentrating solar power(csp) technologies: Current status and research trends. Renewable and Sustainable Energy Reviews 91, 987–1018. doi:10.1016/j.rser.2018.04.097.

Izygon, M., Armstrong, P., Nilsson, C., Vu, N., 2011. TieSOL–a GPU-based suite of software for central receiver solar power plants, in: Proceedings of SolarPACES, Granada, Spain.

Izygon, M., Armstrong, P., Nilsson, C., Vu, N., 2011. TieSOL–a GPU-based suite of software for central receiver solar power plants, in: Proceedings of SolarPACES, Granada, Spain.

Jensen, H.W., Christensen, N.J., 1995. Photon maps in bidirectional Monte Carlo ray tracing of complex objects. Computers & Graphics 19, 215–224. doi:10.1016/0097-8493(94)00145-O.

Leary, P.L., Hankins, J.D., 1979. A User's guide for MIRVAL: a computer code for comparing designs of heliostat-receiver optics for central receiver solar power plants. Technical Report Sandia Laboratories Report, Albuquerque, NM, Report No. SAND77-8280.

Lee, J., 1983. Digital image smoothing and the sigma filter. Computer Vision Graphics and Image Processing 24, 255–269.

Levêque, G., Bader, R., Lipiński, W., Haussener, S., 2017. High-flux optical systems for solar thermochemistry. Solar Energy 156, 133–148.

Li, L., Coventry, J., Bader, R., Pye, J., Lipiński, W., 2016. Optics of solar central receiver systems: a review. Optics Express 24. doi:10.1364/OE.24.00A985.

Lovegrove, K., Stein, W., 2012. Concentrating solar power technology: principles, developments and applications. Elsevier.

Modest, M.F., 2013. Radiative Heat Transfer (3rd edition). Academic Press.

Mutuberria, A., Monreal, A., Albert, A., Blanco, M., 2011. Results of the empirical validation of tonatiuh at mini-pegase cnrs-promes facility, in: SolarPACES, Seville.

Pancotti, L., 2007. Optical simulation model for flat mirror concentrators. Solar Energy Materials and Solar Cells 91, 551–559. doi:10.1016/j.solmat.2006.11.007.

Roccia, J., Piaud, B., Coustet, C., Caliot, C., Guillot, E., Flamant, G., Delatorre, J., 2012. Solfast, a ray-tracing monte-carlo software for solar concentrating facilities 369, 012029. doi:10.1088/1742-6596/369/1/012029.

Schwarzbözl, P., Robert, P.P., Mark, S., 2009. Visual HFLCAL - a software tool for layout and optimisation of heliostat fields, in: Proceedings of 15th International SolarPACES Symposium, pp. 15–18.

Ulmer, S., Marz, T., Prahl, C., Reinalter, W., Belhomme, B., 2011. Automated high resolution measurement of heliostat slope errors. Solar Energy 85, 681–687. doi:10.1016/j.solener.2010.01.010.

Veach, E., Guibas, L.J., 1995. Optimally combining sampling techniques for Monte Carlo rendering, in: Proceedings of ACM SIGGRAPH95, (September 1995), Addison-Wesley. pp. 419–428. doi:10.1145/218380.218498.

Wang, K., He, Y., Xue, X., Du, B., 2017. Multi-objective optimization of the aiming strategy for the solar power tower with a cavity receiver by using the non-dominated sorting genetic algorithm. Applied Energy 205, 399–416. doi:10.1016/j.apenergy.2017.07.096.

Wang, Y., Potter, D., Asselineau, C., Corsi, C., Wagner, M., Blanco, M., Kim, J., Pye, J., 2018. Comparison of optical modelling tools for sunshape and surface slope error, in: AIP Conference Proceedings 2018. doi:10.1063/1.5067222.

Wang, Y., Potter, D., Asselineau, C., Corsi, C., Wagner, M., Caliot, C., Piaud, B., Blanco, M., Kim, J., Pye, J., 2020. Verification of optical modelling of sunshape and surface slope error for concentrating solar power systems. Solar Energy 195, 461–474. doi:10.1016/j.solener.2019.11.035.

Wei, L., 2008. Parallel poisson disk sampling. ACM Transactions on Graphics 27, 20:1–20:9. doi:10.1145/1360612.1360619.

Wendelin, T., 2003. Soltrace: A new optical modeling tool for concentrating solar optics, in: International Solar Energy Conference, pp. 253–260. doi:10.1115/ISEC2003-44090.

White, K.B., Cline, D., Egbert, P.K., 2007. Poisson disk point sets by hierarchical dart throwing, in: Proceedings of the 2007 IEEE Symposium on Interactive Ray Tracing, Washington, DC, USA. pp. 129–132. doi:10.1109/RT.2007.4342600.

Yan, D., Guo, J., Wang, B., Zhang, X., Peter, W., 2015. A survey of blue-noise sampling and its applications. Journal of Computer Science and Technology 30, 439–452. doi:10.1007/s11390-015-1535-0.

Ying, X., Xin, S., Sun, Q., He, Y., 2013. An intrinsic algorithm for parallel poisson disk sampling on arbitrary surfaces. IEEE transactions on visualization and computer graphics 19, 1425–1437. doi:10.1109/TVCG.2013.63.