SIMULATION OF YEARLY ENERGY FOR SOLAR HEATING SYSTEM

Olivier Farges, Jean-Jacques Bézian, Mouna El-Hafi, Hélène Bru

SIMULATION OF YEARLY ENERGY FOR SOLAR HEATING SYSTEM

Olivier Farges[1] [3], **Jean-Jacques Bézian**[2], **Mouna El Hafi**[2], **and Hélène Bru**[3]

[1] Phd Student, Total New Energies and Centre RAPSODEE, UMR CNRS 5302,

Campus Jarlard, Route de Teillet, 81013 Albi Cedex 09, France

Tel : +33563493065, e-mail : olivier.farges@mines-albi.fr

[2] Centre RAPSODEE, UMR CNRS 5302, École des Mines d'Albi-Carmaux, 81013 Albi Cedex 09, France

[3] Total New Energies, R&D - Concentrated Solar Technologies, Immeuble Lafayette2,

place des Vosges - La Défense 5, 92078 Paris La Défense Cedex, France

Abstract

To perform ray tracing simulation during the design process of a central solar receiver, we have developed a code based on Monte Carlo algorithm suitable for computing the yearly average energy at the entrance of the receiver of a solar power tower. This code considers sun positions over the year. Based on EDStaR coding environment, it is fast and accurate and will be integrated inside an optimization scheme to design solar Central Receiver System (CRS) with enhanced annual performance.

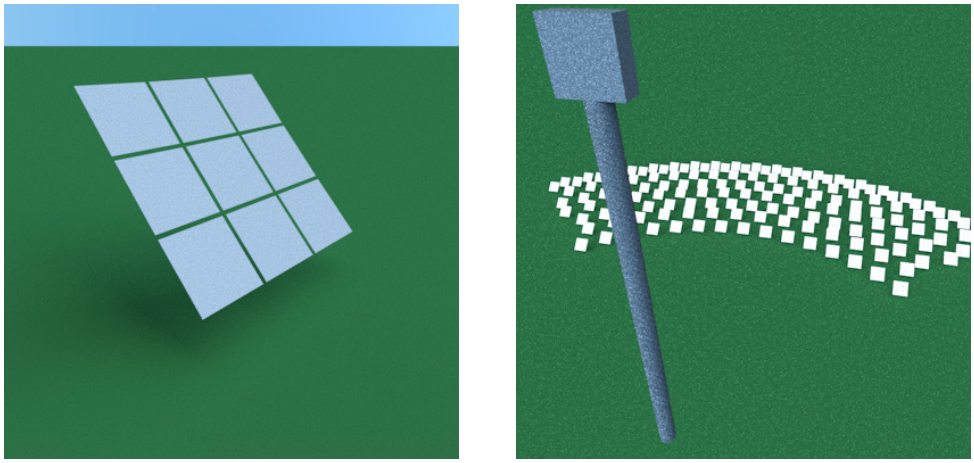
Keywords : Monte Carlo algorithm, Central solar receiver, Sun tracking, Yearly average energy

Nomenclature

Ω_S	Solar cone in sr	$\mathcal{H}$	Heliostats surface (the exponent + indicate the active side)
Φ	Latitude in rad	h	Hour angle in rad
δ	Declination in rad	$\boldsymbol{n_1}$	Ideal normal at $\boldsymbol{x_1}$
η	Hour in h:min:s	$\boldsymbol{n_h}$	Effective normal at $\boldsymbol{x_1}$ around the ideal normal $\boldsymbol{n_1}$
γ	Day of the year	Sh	Shadowing performance in %
$\boldsymbol{\omega_1}$	Direction after reflection in rad	Sp	Spillage performance in %
$\boldsymbol{\omega_S}$	Direction inside the solar cone in rad	$\mathcal{T}$	Target (the exponent + indicate the active side)
ϕ_S	Sun azimuth angle in rad	t	Time in s
θ_S	Sun elevation angle in rad	$\hat{w}_i$	Monte Carlo weight
A	Yearly average energy in J	$\boldsymbol{x_i}$	Point in the geometry
B	Blocking performance in %		
DNI	Direct normal irradiance in $\mathrm{W \cdot m^{-2}}$		
f_r	Bidirectional reflectance distribution function (BRDF)		

1. Introduction

Considering the investment needed to build a solar concentrating facility, the performance of such an installation has to be maximized. This is the reason why the preliminary design step is one of the most important stage of the project process. Nowadays, numerical simulation is widely used to perform this step and simulation tools have an important role in concentrated solar power (CSP) development. During the last decades, a significant number of well-known and widely-used tools have been developed. But those tools only simulate



(a) Heliostat **(b)** Central Receiver System

Fig. 1: Computer generated images with EDStaR

installation performance for one sun position so it's not possible to evaluate yearly performance. In addition, there are methods to design heliostat field according to yearly optical performance [1] but no raytracing tool allows designers to obtain the yearly performance of a solar central receiver in one calculation. We will propose in this paper a new approach based on Monte Carlo methods that permits to take into account the sun tracking annually. Thus, we can obtain a precise estimation of the yearly thermal energy received by the receiver with a reasonable computational time. This code, fast and accurate, permits to achieve optimization step with, for example, stochastic optimization methods (genetic algorithm, particle swarm optimization, ...).

2. Simulation tool

EDStaR (numerical Environment of Development for Statistical Radiative simulation) is a coding environment maintained by the StarWest group [2], a group of physicists specialized in radiative transfer and out-of-equilibrium statistical thermodynamics. EDStaR is mainly devoted to photon transport but it can also deal with micro-scale gaseous thermal flows, liquid-gas transitions or self-organization in biology. Using Monte-Carlo methods [3] [4], it takes advantages of advanced rendering techniques from computer graphics community : it can manage complex geometries with the use of the C++ object library designed in the frame of the Physically Based Rendering Techniques (PBRT) project [5]. PBRT is combined with the *mcm* C++ object library that handle with programming Monte Carlo algorithms, including sensitivity estimations. Another part core program constituting EDStaR is the *GNU Scientific Library (GSL)* [6] used for uniform random number sampling in the unit interval. The GSL contains many distinct random number generators to insure the same level of uniformity and independence. Those three libraries are combined into the Mcm3D development environment. Mcm3D takes the benefit of all modern possibilities of computing such as massive parallelization [7] and acceleration of the ray tracing in a complex geometry. We can obtain computer generated images of studied systems, as shown on fig. 1 illustrating the academic case studied in the validation part of this paper (4).

3. Sun tracking Monte Carlo algorithm

As mentioned in part 1 we have built a code that tracks sun positions during one year. EDStaR allows to build a specific Monte Carlo algorithm dealing with sun positions in the sky applied to central receiver system (CRS) simulation.

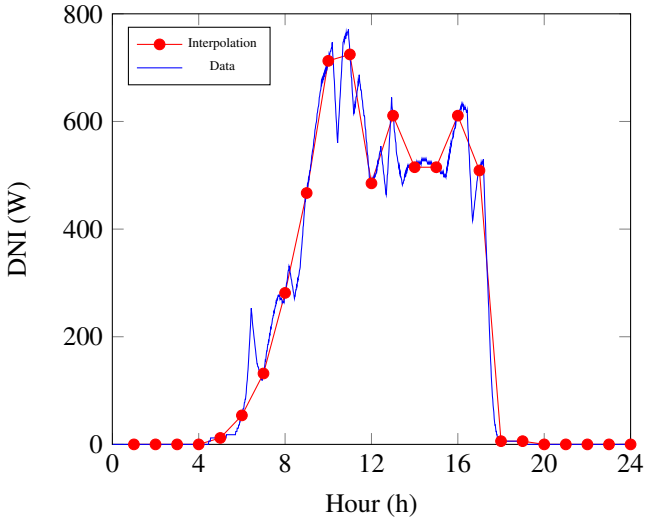


Fig. 2: Linear interpolation for DNI values

3.1. Sun position and DNI values

Taking into account the sun tracking inside the Monte-Carlo algorithm implies that the geometry of the solar installation is updated at each solar position. So, we have to find a convenient way to describe sun positions at each instant. For a given geographical position, *ie* a couple *longitude-latitude*, sun position can be described by angles (azimuth and elevation, azimuth and zenith angle, declination and hour angle, ...) or by dates (year, month, day , hour, minutes, seconds). Both representations are equivalent and could define sun positions with either of these descriptions. We make the choice to work with dates with a day γ between $[1;365]$ and a time of the day η between $[7:00:00;19:00:00]$[1]. We can define azimuth ϕ_S and elevation angle θ_S with Eqs.1 and 2 using the latitude Φ corresponding to the location of the central receiver system (CRS) to calculate declination δ and hour angle h with Eqs.3 and 4.

$$\cos\phi_S = \frac{\sin\delta\cos\Phi - \cos h\cos\delta\sin\Phi}{\cos\theta_S} \tag{1}$$

$$\sin\theta_S = \cos h\cos\delta\cos\Phi + \sin\delta\sin\Phi \tag{2}$$

$$\delta = 23.45\sin\left(360\times\frac{284+\gamma}{365}\right) \tag{3}$$

$$h = (\eta - 12)\times 15 \tag{4}$$

Each sun position corresponds to a DNI value according to weather pattern. As we need a DNI value for each instant but have only hourly radiation datasets stored in database, thereby we have to interpolate data. As represented on fig. 2 for the 21^{th} of June 2005 in Albi, France, linear interpolation between hour values gives a good approximation for a specific date when compared to values collected each minutes [8].

3.2. Monte-Carlo Sun Tracking algorithm

3.2.1. Mathematical formulation

The Monte Carlo Sun Tracking algorithm (MCST) implemented in this code takes into account sun positions. Doing this, we introduce an integration over time and we obtain an energy value (in J) instead of a power (in W).The aim of the computation is to evaluate the yearly energy received at the entrance of a CRS cavity receiver. The yearly energy A is expressed by the Eq.5.

[1] Sampling hours during all the day doesn't prove of interest so we focus on opening hours of a CRS, here for a location at latitude 0°. For location out of the equator, range of sampling would be extended to cover the longest day of the year, with the effect of increasing the proportion of sampled sun positions corresponding null DNI.

$$A = \int_{Year} \mathrm{d}t \int_{H^+} \mathrm{d}x \int_{\Omega_{sun}} \mathrm{d}\omega_S DNI \times [H(x_0 \notin (\mathcal{H} \cup \mathcal{T})) \times \int_{2\pi} \mathrm{d}\boldsymbol{\omega_1} f_r(\boldsymbol{\omega_1}|\boldsymbol{\omega}_S, \boldsymbol{x_1}, \nu) |\boldsymbol{n}(\boldsymbol{x_1}) \cdot \boldsymbol{\omega_1}| \times \left[H(\boldsymbol{x_2} \in \mathcal{T}^+)\right]\Big] \tag{5}$$

The integral over time is splitted into two integrals over day and hour as presented in Eq.6.

$$\int_{Year} \mathrm{d}t = \int_1^{365} \mathrm{d}\gamma \int_7^{19} \mathrm{d}\eta \tag{6}$$

Eq. 5 contains tests symbolized by *Heaviside functions* $H(x)$. Such a test consists in $H(x) = 1$ if x statement is true and $H(x) = 0$ if it is false. These tests in Eq.5 permit to identify shadowing effect between the reflective surface on which the first reflection appears and the sun ($H(x_0 \notin (\mathcal{H} \cup \mathcal{T}))$ and to know if the first intersection after the reflection on the heliostats happens on the target ($H(\boldsymbol{x_2} \in \mathcal{T}^+)$). In fact, the algorithm begins with a sampling of the $\mathcal{H}^+$ surface.Then, the incident direction ω_S is sampled in the solar cone. The bidirectional reflectance distribution function (BRDF) $f_r(\boldsymbol{\omega_1}|\boldsymbol{\omega}_S, \boldsymbol{x_1})$ is a four-dimensional function that defines how light is reflected at an opaque surface. It defines the behavior of photons during the reflection on heliostats, taking an incoming light direction, $\boldsymbol{\omega_S}$, and outgoing direction,$\boldsymbol{\omega_1}$, both defined with respect to the surface effective normal $\boldsymbol{n_h}$.
According to Monte Carlo methods, the quantity A is considered as the mean of a random variable W_A : $A = E[W_p]$. A is approximated by computing a finite number N of realizations $w_{A,k}$ of W_A as represented in Eq.7.

$$A \approx \tilde{A} = \frac{1}{N} \sum_{k=1}^{N} w_{A,k} \tag{7}$$

A standard deviation $\sigma_{\tilde{A}}$ associated to each estimation of $\tilde{A}$ can be obtained with a good approximation by a statistical uncertainty $\tilde{\sigma}_{\tilde{A}}$ (Eq.8)

$$\sigma_{\tilde{A}} \approx \tilde{\sigma}_{\tilde{A}} = \frac{1}{\sqrt{N}} \sqrt{\left(\frac{1}{N} \sum_{k=1}^{N} w_{P,k}^2\right) - \tilde{A}^2} \tag{8}$$

The MCST model also includes the reflection events involved in a CSP system. It's easy to compute additional values characterizing CSP facility optical performances such as :

- Shadowing (Eq.9);
- Blocking (Eq.10) ;
- Spillage (Eq.11) ;

$$Sh = \int_{Year} \mathrm{d}t \int_{H^+} \mathrm{d}x \int_{\Omega_{sun}} \mathrm{d}\omega_S DNI \times H(\boldsymbol{x_0} \in (\mathcal{H} \cup \mathcal{T})) \tag{9}$$

$$B = \int_{Year} \mathrm{d}t \int_{H^+} \mathrm{d}x \int_{\Omega_{sun}} \mathrm{d}\omega_S DNI \times [H(\boldsymbol{x_0} \notin (\mathcal{H} \cup \mathcal{T})) \times \int_{2\pi} \mathrm{d}\boldsymbol{\omega_1} f_r(\boldsymbol{\omega_1}|\boldsymbol{\omega}_S, \boldsymbol{x_1}, \nu) |\boldsymbol{n_1} \cdot \boldsymbol{\omega_1}| \times H(\boldsymbol{x_2} \in \mathcal{H})\Big] \tag{10}$$

$$Sp = \int_{Year} \mathrm{d}t \int_{H^+} \mathrm{d}x \int_{\Omega_{sun}} \mathrm{d}\omega_S DNI \times [H(\boldsymbol{x_0} \notin (\mathcal{H} \cup \mathcal{T})) \times \int_{2\pi} \mathrm{d}\boldsymbol{\omega_1} f_r(\boldsymbol{\omega_1}|\boldsymbol{\omega}_S, \boldsymbol{x_1}, \nu) |\boldsymbol{n}(\boldsymbol{x_1}) \cdot \boldsymbol{\omega_1}| \times H(\boldsymbol{x_2} \notin (\mathcal{H} \cup \mathcal{T}))\Big] \tag{11}$$

3.2.2. Algorithmic formulation

To evaluate those variables with Monte Carlo methods, we introduce probability density functions (PDF). PDF allow us to improve convergence and to reduce the variance. To illustrate our purpose, we assume that reflections are specular. With Monte Carlo weight and PDF, we obtain Eqs.12, 13, 14 and 15 from respectively Eqs.5, 9, 10 and 11

$$\begin{aligned} A = & \int_{\Delta} p_{\Delta}(\delta)\mathrm{d}\delta \int_{H} p_{H}(\eta)\mathrm{d}\eta \int_{\mathcal{H}^{+}} p_{X_1}(\boldsymbol{x_1})\mathrm{d}\boldsymbol{x} \int_{\Omega_S} p_{\Omega_S}(\boldsymbol{\omega_S})\mathrm{d}\boldsymbol{\omega_S} \times \Bigg\{ H(x_0 \in (\mathcal{H} \cup \mathcal{T}))\hat{w}_{out} \\ & + H(x_0 \notin (\mathcal{H} \cup \mathcal{T}) \times \int_{D_{N_h}} p_{N_h}(\boldsymbol{n_h}|\boldsymbol{\omega_S}; p)\mathrm{d}\boldsymbol{n_h} \{H(\boldsymbol{x_2} \notin \mathcal{T})\hat{w}_{out} + H(\boldsymbol{x_2} \in \mathcal{T})\hat{w}_A\} \Bigg\} \end{aligned} \tag{12}$$

$$\begin{aligned} Sh = & \int_{\Delta} p_{\Delta}(\delta)\mathrm{d}\delta \int_{H} p_{H}(\eta)\mathrm{d}\eta \int_{\mathcal{H}^{+}} p_{X_1}(\boldsymbol{x_1})\mathrm{d}\boldsymbol{x} \int_{\Omega_S} p_{\Omega_S}(\boldsymbol{\omega_S})\mathrm{d}\boldsymbol{\omega_S} \times \{H(\boldsymbol{x_0} \in (\mathcal{H} \cup \mathcal{T}))\hat{w}_{Sh} \\ & + H(\boldsymbol{x_0} \notin (\mathcal{H} \cup \mathcal{T})\hat{w}_{out}\} \end{aligned} \tag{13}$$

$$\begin{aligned} B = & \int_{\Delta} p_{\Delta}(\delta)\mathrm{d}\delta \int_{H} p_{H}(\eta)\mathrm{d}\eta \int_{\mathcal{H}^{+}} p_{X_1}(\boldsymbol{x_1})\mathrm{d}\boldsymbol{x} \int_{\Omega_S} p_{\Omega_S}(\boldsymbol{\omega_S})\mathrm{d}\boldsymbol{\omega_S} \times \{H(x_0 \in (\mathcal{H} \cup \mathcal{T}))\hat{w}_{out} \\ & + H(\boldsymbol{x_0} \notin (\mathcal{H} \cup \mathcal{T}) \times \big\{H(\boldsymbol{x_2} \in \mathcal{T}^{+})\hat{w}_{out} + H(\boldsymbol{x_2} \notin \mathcal{T}^{+}) \times [H(\boldsymbol{x_2} \notin (\mathcal{H} \cup \mathcal{T}))\hat{w}_{out} \\ & + H(\boldsymbol{x_2} \in (\mathcal{H} \cup \mathcal{T}))\hat{w}_B]\big\}\} \end{aligned} \tag{14}$$

$$\begin{aligned} Sp = & \int_{\Delta} p_{\Delta}(\delta)\mathrm{d}\delta \int_{H} p_{H}(\eta)\mathrm{d}\eta \int_{\mathcal{H}^{+}} p_{X_1}(\boldsymbol{x_1})\mathrm{d}\boldsymbol{x} \int_{\Omega_S} p_{\Omega_S}(\boldsymbol{\omega_S})\mathrm{d}\boldsymbol{\omega_S} \times \{H(x_0 \in (\mathcal{H} \cup \mathcal{T}))\hat{w}_{out} \\ & + H(\boldsymbol{x_0} \notin (\mathcal{H} \cup \mathcal{T}) \times \big\{H(\boldsymbol{x_2} \in \mathcal{T}^{+})\hat{w}_{out} + H(\boldsymbol{x_2} \notin \mathcal{T}^{+}) \times [H(\boldsymbol{x_2} \in (\mathcal{H} \cup \mathcal{T}))\hat{w}_{out} \\ & + H(\boldsymbol{x_2} \notin (\mathcal{H} \cup \mathcal{T}))\hat{w}_{Sp}]\big\}\} \end{aligned} \tag{15}$$

The PDF are defined in Eqs.16, 17, 18, 19, 20. For probability density function $p_{\boldsymbol{n_h}}$ in Eq.20, the Blinn's model is used, of parameter p, with a truncation of the distribution of the dot product $\boldsymbol{n_h} \cdot \boldsymbol{n_1}$ avoiding the occurrence of reflected directions toward the surface for quasi-tangent incidences.

$$p_{\Delta}(\delta) = \frac{1}{365} \tag{16}$$

$$p_{H}(\eta) = \frac{1}{12} \tag{17}$$

$$p_{X_1}(\boldsymbol{x_1}) = \frac{1}{S_{\mathcal{H}^{+}}} \tag{18}$$

$$p_{\Omega_S}(\boldsymbol{\omega_S}) = \frac{1}{2\pi(1-\cos\theta_S)} \tag{19}$$

$$p_{\boldsymbol{n}_h}(\boldsymbol{n}_h|\boldsymbol{\omega}_S; p) = \frac{2+\frac{1}{p}}{2\pi\left(1-\mu(\boldsymbol{\omega}_S)^{2+\frac{1}{p}}\right)} \tag{20}$$

The Monte Carlo weights are defined in Eqs.21, 22 and 23

$$\hat{w}_A = \frac{I_{sun}(\boldsymbol{\omega_S} \cdot \boldsymbol{n_h})}{p_{\Omega_S}(\boldsymbol{\omega_S}) p_{X_1}(\boldsymbol{x_1})} = DNI(\boldsymbol{\omega_S} \cdot \boldsymbol{n_h}) S_{\mathcal{H}^{+}} \tag{21}$$

$$\hat{w}_{Sh} = \hat{w}_B = \hat{w}_{Sp} = 1 \tag{22}$$

$$\hat{w}_{out} = 0 \tag{23}$$

The Monte Carlo Sun Tracking integral formulation can be directly interpreted as the algorithm 1. It is equivalent to Eqs. 12 - 23.

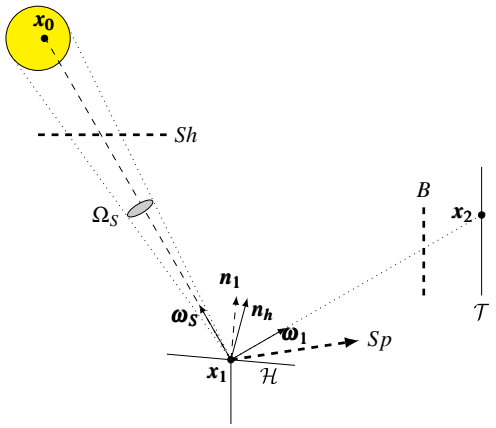
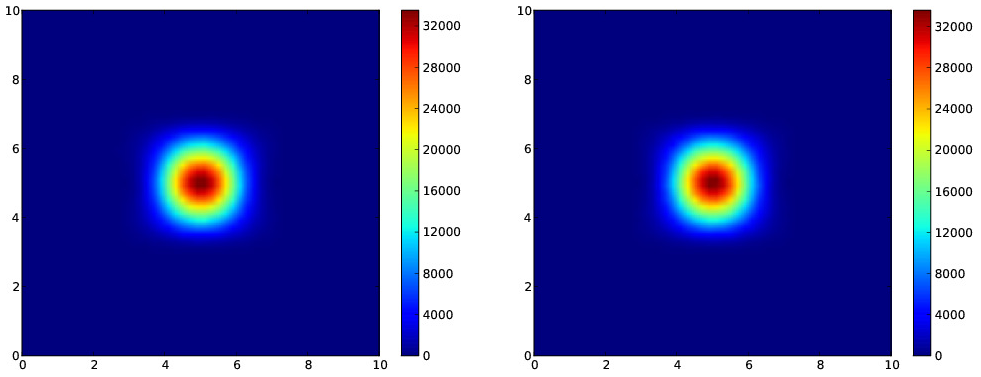


Fig. 3: Schematic representation of a CRS

$\tilde{A} = 0$;
foreach *event* **do**
  Uniform sampling of δ in $[0, 365]$;
  Uniform sampling of η in $[7A.M., 7P.M.]$;
  Uniform sampling of $\boldsymbol{x_0}$ on $\mathcal{H}$;
  Sampling of $/v\omega_0$ on solar disk;
  if *No shadowing between sun and* $\boldsymbol{x_0}$ **then**
    $\hat{w}_A = DNI|\boldsymbol{n}(\boldsymbol{x_1}) \cdot \boldsymbol{\omega_S}|\mathcal{H}^+$;
  else
    $\hat{w}_A = 0$;
    $\hat{w}_{Sh} = 1$;
    break;
  end
  $\hat{w}_A = \hat{w}_A \times \rho$;
  Generation of $\boldsymbol{n_h}$ according to Blinn's model ;
  Specular reflection $\boldsymbol{\omega_1} = \boldsymbol{\omega_S} + 2|\boldsymbol{n_h} \cdot \boldsymbol{\omega_S}|\boldsymbol{n_h}$;
  $\boldsymbol{x_2}$ = intersection of $Ray(\boldsymbol{x_1}, \boldsymbol{\omega_1})$ with geometry element ;
  if $\boldsymbol{x_1}$ *exists* **then**
    if $\boldsymbol{x_1} \in \mathcal{T}^+$ **then**
      $\tilde{A} = \tilde{A} + \hat{w}_A$ break;
    else
      $\hat{w}_A = 0$;
      $\hat{w}_B = 1$;
      break;
    end
  else
    $\hat{w}_A = 0$;
    $\hat{w}_{Sp} = 1$;
    break;
  end
end

Algorithm 1: Monte Carlo Sun Tracking algorithm (MCST)



(a) EDStaR fluxmap **(b)** Tonatiuh fluxmap

Fig. 4: Flux-map comparison for the spring equinox at noon in W

4. Validation

To validate our method, we have to do some comparisons. We choose to use Tonatiuh [9] which is a reference tool in the concentrated solar community to compute a testing case. This case is the central solar receiver represented on fig. 1b with a tower and 146 heliostats in a heliostats field. The field is designed with the MUEEN [10] method, following a radial staggered layout. Each heliostat is made of 9 squared mirrors of 1.6 meter sided. Fig. 1a represent the geometry of an heliostat. To be sure to have the same case with both codes, we describe the CRS in Tonatiuh with its script tool to suit to the heliostat shape used in EDStaR. Moreover, we make some general assumptions :

- Reflections are specular ;
- CRS is located at the junction of the Greenwich meridian and the equator ;
- the target is a square with sides of 10m long ;

4.1. Comparison with Tonatiuh at fixed date

To be sure that future simulations with sun tracking will be consistent, we firstly evaluate some simplified cases at fixed dates with both Tonatiuh and EDStaR. We choose to compute four dates with a DNI equal to $1000\mathrm{Wm}^{-2}$ with a number of rays equal to five million :

- Spring equinox : the 21^{st} of March at noon (solar time) ;
- Summer solstice : the 21^{st} of June at noon (solar time) ;
- Autumn equinox : the 22^{th} of September at noon (solar time) ;
- Winter solstice : the 21^{st} of December at noon (solar time) ;

Fig. 4 represents flux-maps obtained with the two codes for the spring equinox. They seem very similar so we compute the relative error for each pixel according to Eq. 24. It appears that the relative error for each pixel is almost insignificant (less than 2.25%, as illustrated on fig. 5. Numerical results for the four dates, presented in tab. 1, are equal to the second decimal place. We can conclude that the test case gives comparable results with both Tonatiuh and EDStaR.

$$\varepsilon_{ij} = \frac{Ft_{ij} - Fe_{ij}}{Ft_{ij}} \tag{24}$$

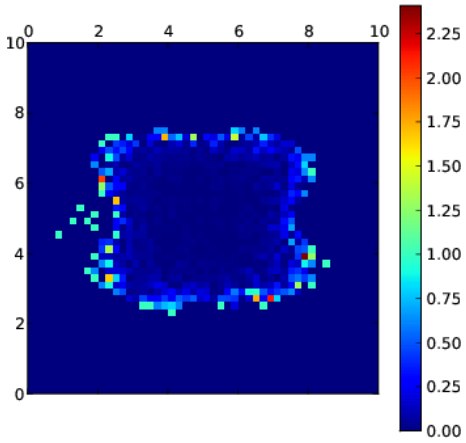


Fig. 5: Difference Between Tonatiuh fluxmap and EDStaR fluxmaps in %

Date	Tonatiuh	EDStar
Spring equinox	2.97MW	$2.97\text{MW} \pm 72\text{W}$
Summer solstice	3.19MW	$3.19\text{MW} \pm 98\text{W}$
Autumn equinox	2.98MW	$2.98\text{MW} \pm 73\text{W}$
Winter solstice	3.19MW	$3.19\text{MW} \pm 98\text{W}$

Table 1: Power at receiver

4.2. Comparison for 50 dates

Now, we randomly choose 50 dates (example : $\gamma = 15$ and $\eta = 14.745$ gives $15/01/2012\ 14:44:42$) and we compute each date with Tonatiuh (Number of rays = 1 000 000). We obtain a power at the receiver but we also can consider the result as the instantaneous energy arriving on the target. When we have the fifty results, assuming that the averaged power over the 50 dates is representative of the energy received at each second we obtain a thermal energy of 6.329GWh_{th}. We can now do the same simulation with EDStaR with the Monte Carlo Sun Tracking algorithm. So, we only do one simulation with the same 50 dates (with 1 photons for each date) and we obtain as a result $6,323 \pm 0.623\text{GWh}_{th}$. The tab. 2 represents those results. It appears that the value given by EDStaR is very close to the value obtained with Tonatiuh. We also could notice that the error bars of EDStaR are a bit large but with only 50 dates, it is logical to observe this trend. To conclude this part, EDStaR gives an estimation in accordance with Tonatiuh results for a yearly simulation done date by date even if error bars are significantly large due to the small number of dates computed. By increasing the number of dates we obtain a more precise value of the yearly energy.

Dates	Tonatiuh	MCST
50 dates	6.329GWh_{th}	$6,323 \pm 0.623\text{GWh}_{th}$

Table 2: Comparison Tonatiuh - EDStaR for 50 dates

4.3. Yearly simulation

To have a more precise result, *ie* to obtain narrower error bars than in part 4.2, we compute a simulation with a number of rays significantly larger. As the computational time is mainly devoted to the updating of the heliostats orientation, we develop an upgraded version of the code called Multi-Ray Monte Carlo Sun Tracking (MRMCST). It sends several rays for each date, before computing another date and updating orientation. To identify the best ration "*Accuracy / Computational time*", we made some tests considering different numbers of rays for the MRMCST algorithm. We compare results with the MCST algorithm, as represented in fig. 6. It appears that the number of dates needs to be sufficient to insure a representative sample over the year

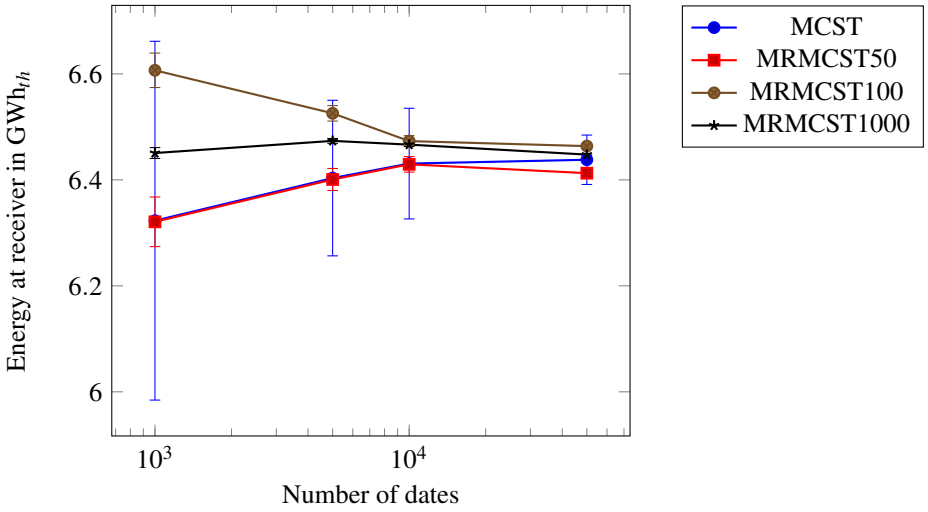


Fig. 6: Comparison of MCST and MRMCST codes

otherwise MRMCST create a bias in comparison with MCST results but, in the same time, error bars obtained with different MRMCST are widely narrower than MCST ones (with a confidence interval of 99.7% *ie* $\pm 3\sigma_{\bar{A}}$). We assume that 10^5 dates is a sufficient number to avoid the bias phenomena. Furthermore, computational times presented in tab. 3 dealing with number of dates computed, highlight the need to identify a compromise between accuracy and computational speed as the "multi rays" algorithm means more realizations for a date. Following results in part 4.4 handle with number of realizations instead of number of dates. Following results presented in part 4.4 are based on a MRMCST algorithm with 100 rays per date.

Number of dates	MCST	MRMCST 50r	MRMCST 100r	MRMCST 1000r
1000	0.62s	1.71s	3.84s	27.08s
5000	1.46s	9.96s	16.39s	127.57s
10000	1.97s	16.45s	26.77s	264.83s
50000	9.93s	69.54s	126.96s	1150s

Table 3: MCST versus MRMCST computational time

4.4. Simulation time comparison

As mentioned in part 1, we plan to use this code into an optimization scheme, so the computational time is of great interest. We compare computational time[2] with Tonatiuh on the basis of similar numbers of realisations, knowing that it runs date by date. We evaluate as reference a "Tonatiuh equivalent" to MRMCST (100 rays) considering the following step execution time : 4.003s for each date, taking into account script opening (2s), pre and post-processing (2s) and tracing 100 rays (3ms).

Realizations	Tonatiuh	Tonatiuh Eq.	MCST	MRMCST
$5 \cdot 10^4$	$\approx$ 3s	$2 \cdot 10^5$s	9s	2s
$5 \cdot 10^5$	$\approx$ 24s	$2 \cdot 10^6$s	59s	16.39s
$5 \cdot 10^6$	$\approx$ 156s	$2 \cdot 10^7$s	814s	126.96s

Table 4: Computational time comparison

It appears that MRMCST is faster than MCST as noticed in part 4.3 and very similar to Tonatiuh computing times with regards to computing time for a given number of realizations. The number of realizations for

[2]On a desktop PC with AMD Phenom II X6 1055T 2.8GHz and 4Mo RAM

Tonatiuh is for a single date as ray-tracing can be run only date by date whereas realizations for MRMCST cover the full year. So this computation speed comparison doesn't account for the time saving brought by MRMCST for yearly energy simulation.

5. Conclusion and outlook

A new code for simulation of central receiver systems has been developed with the coding environment EDStaR. It deals with sun positions during a year and thus computes the yearly energy at the entrance of the receiver. This code is fast and accurate as illustrated by the validation step with Tonatiuh. Then, it can be used in an optimization scheme during the preliminary design step of a solar power tower. Some improvements can be made with further investigation of probability density functions effects, particularly concerning date and hour variables.

References

[1] M. SÁNCHEZ and M. ROMERO, "Methodology for generation of heliostat field layout in central receiver systems based on yearly normalized energy surfaces," *Solar Energy*, vol. 80, no. 7, pp. 861 – 874, 2006.

[2] "Starwest (statistiques radiatives du sud-ouest)." http://www.starwest.ups-tlse.fr/.

[3] N. METROPOLIS and S. ULAM, "The monte carlo method," *Journal of the American Statistical Association*, vol. 44, no. 247, pp. 335–341, 1949.

[4] J. M. HAMMERSLEY and D. C. HANDSCOMB, *Monte Carlo Methods*. Chapman and Hall, 1969.

[5] M. PHARR and G. HUMPHREYS, *Physically Based Rendering, second edition : from theory to implementation*. Morgan Kaufmann Publishers, 2010.

[6] "Gsl - gnu scientific library." http://www.gnu.org/software/gsl.

[7] "The message passing interface (mpi) standard.." http://www.mcs.anl.gov/research/projects/mpi.

[8] "Soda : Solar energy services for professionals." http://www.soda-is.com/eng/index.html.

[9] M. J. BLANCO, J. M. AMIEVA, and A. MANCILLA, "The Tonatiuh Software Development Project: An open source approach to the simulation of solar concentrating systems," in *Proceedings of the ASME Computers and Information in Engineering Division*, pp. 157–164, AMER SOC MECHANICAL ENGINEERS, 2005. ASME International Mechanical Engineering Congress and Exposition, Orlando, FL, NOV 05-11, 2005.

[10] F. M. F. SIALA and M. E. ELAYEB, "Mathematical formulation of a graphical method for a no-blocking heliostat field layout," *Renewable energy*, vol. 23, no. 1, pp. 77–92, 2001.