Software for designing solar UV diffusers

Tomi Pulli[1], Petri Kärhä[1,2], Joop Mes[3] and Josef Schreder[4]

1. Metrology Research Institute, Aalto University School of Electrical Engineering, Espoo, Finland
2. Centre for Metrology and Accreditation MIKES, Espoo, Finland
3. Kipp & Zonen, Delft, The Netherlands
4. CMS - Ing. Dr. Schreder GmbH, Kirchbichl, Austria

Introduction

Large portion of the UV radiation that reaches the surface of the Earth is scattered. Therefore, an instrument that measures the total UV irradiance needs to collect radiation from the entire hemisphere. The angular response of such an instrument should be proportional to the cosine of the zenith angle and independent of the azimuth angle of the radiation. Deviations from the ideal cosine response can cause significant errors in the results. These errors can be corrected if the angular distribution of the radiation at the time of the measurement is well known, but only to an extent.

Diffusers are commonly used at the entrances of solar UV measuring instruments to reach near-ideal cosine response. As the angular response obtained with a flat sheet of diffusing material always deviates from the ideal cosine response, some form of diffuser shaping is required to reach low cosine errors. Optimizing the shape of the diffuser through trial-and-error can be time-consuming and expensive. Therefore, it would be beneficial to be able to model and optimize the structure of the diffuser before manufacturing. For this purpose a Monte Carlo software for simulating light transport inside the diffuser was developed. The software was validated by comparing the simulated results with measured angular responses. The software was then used to design an improved solar UV diffuser.

Diffuser structure

The diffuser geometry assumed by the simulation software is shown in Figure 1. The diffuser itself is characterized by the diameter d, the edge thickness t, the refractive index n_2, the scattering coefficient μ_s, the absorption coefficient μ_a, and the scattering anisotropy parameter g. The software can simulate flat diffusers as well as spherically shaped diffusers with a radius of curvature r_{sph}. The height of the diffuser is defined as $h = t - h_{sw}$, where h_{sw} is the height of the sidewall. The diameter of the visible area of the diffuser is d_{beam}. The distance between the detector and the back surface of the diffuser is z_{det}.

The diameter and the height of the shadow ring – whose purpose is to block radiation at incident angles equal to and larger than 90° – are d_{sr} and h_{sr}, respectively. The protective weather dome has an inner radius of r_{dom}, a thickness of t_{dom} and a refractive index of n_{dom}. The offset of the weather dome is determined by z_{dom} as shown in Figure 1.

Simulation algorithm

The diffuser simulation software is based on the Monte Carlo ray tracing. To ensure simulation efficiency, the particles are traced from the detector towards the sky – not vice versa – and are collected after they exit the detector head. Furthermore, instead of individual photon-like particles, large groups of particles are considered to allow for partial absorption and reflection of the weight of the particle. In the following discussion, these packages of particles traveling in the opposite direction to the photons are referred to simply as “particles.”

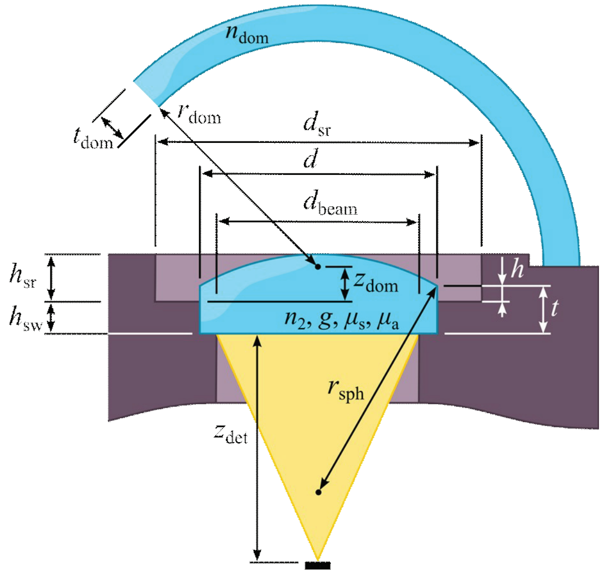

Figure 1. Structure of the diffuser. For explanation of the symbols, see text.

The workflow of the simulation software is presented in Figure 2. At the first stage of the tracing process, a particle emanating from the detector enters the diffuser. The particle is refracted at the material interface, and part of its weight is lost due to reflection. These processes are governed by Snell’s law and Fresnel equations, respectively.

Inside the diffuser, the propagation follows the framework laid out in an article [1] by Wang et al. concerning the light transport in tissues. At the beginning of each propagation step, the distance to travel is determined by a random number that follows the appropriate probability distribution. If the new coordinates are inside the diffuser, the position of the particle is updated. The weight of the particle decreases due to absorption. Scattering is governed by Henyey–Greenstein scattering phase function where scattering anisotropy parameter g determines the probability distribution of the deflection angles [2]. The propagation–absorption–scattering cycle is repeated until the weight of the particle has decreased below a user-set threshold value, or until the particle hits a material interface. Once the termination threshold is reached, the tracing process is repeated again for a new particle.

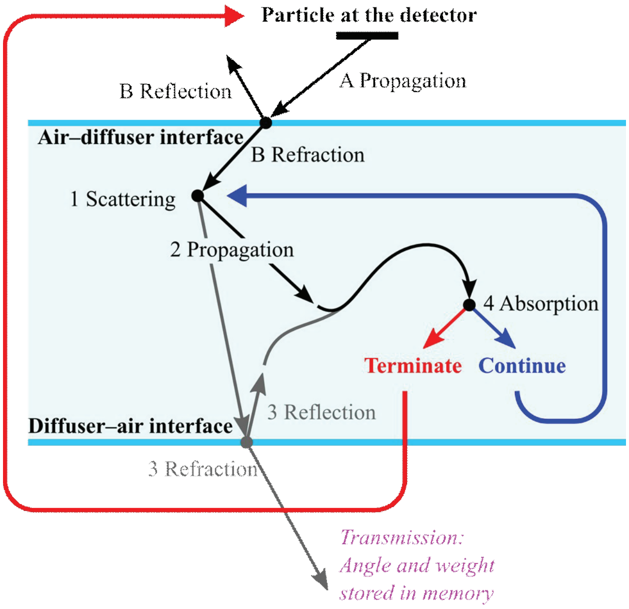

Figure 2. Workflow of the simulation software.

To find out the refraction and the reflection angles as well as the transmittance and the reflectance at the material interface, one must first calculate the coordinates at which the particle crosses the interface and the surface normal vector at those coordinates. When this is done, the law of reflection and the Snell's law of refraction in combination with the Fresnel equations can then be used to determine the necessary quantities. The reflected particle will continue to propagate inside the diffuser in a new direction and with diminished weight. If the transmitted particle hits either the shadow ring or the sidewall of the diffuser, it is absorbed. If not, then its weight will contribute to the overall angular response of the detector.

The protective weather dome, if present, has an effect on both the propagation direction and the weight of the transmitted particle. Furthermore, a particle that is reflected at the inner surface of the weather dome can still exit the detector structure at some other point of the weather dome, thus increasing the overall angular response at large zenith angles. However, this effect is only perceived when the diameter of the diffuser is close to that of the weather dome.

Simulation software

The graphical user interface of the simulation software, shown in Figure 3, was written in *Python* programming language. *Numpy* and *Matplotlib* libraries were used for array handling and data plotting, respectively. The Monte Carlo algorithm was implemented in *Cython* which is used to compile fast C extensions from a modified *Python* code.

During a typical simulation of one billion particles, roughly one hundred billion uniformly distributed pseudo-random numbers – equivalent to 800 GB of double precision floating point numbers – are generated. To draw these numbers, a fast, high quality pseudo-random number generator is required. An enhanced version of the popular Mersenne Twister algorithm, namely *SIMD-oriented Mersenne Twister* (*SFMT*) [3], was chosen for this purpose. A typical simulation of one billion particles takes roughly 10 to 15 minutes to complete with a new quad-core processor.

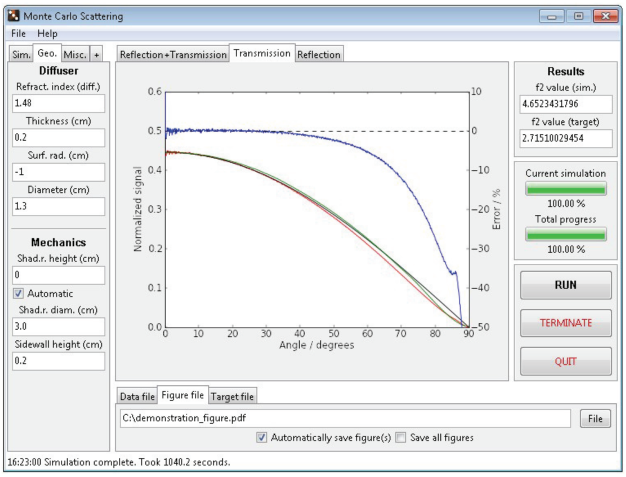

Figure 3. Graphical user interface of the simulation software.

All of the detector parameters shown in Figure 1 can be adjusted in the software, along with various simulation settings, such as the total number of particles and the termination threshold for tracing. The user can perform single simulations as well as parameters sweeps. New simulations can be queued while the current simulation is still running so that the user needs not be constantly present when a large number of simulations are performed sequentially.

Once the simulation has completed, the software plots the angular response as well as the cosine error and calculates the integrated cosine error f_2 as defined in [4]. To reduce the noise in the figures, the user can choose between different smoothing functions and adjust the span of averaging. The user can also load up a file with a target angular response that is plotted in the same figure with the simulation results. This feature is particularly useful when matching the simulated angular responses with the measurement results by adjusting one of the material parameters, e.g. the scattering coefficient μ_s.

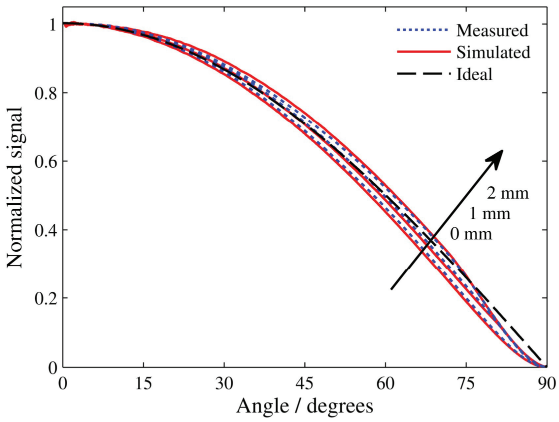

Figure 4. Comparison between the measured (dotted blue lines) and simulated (solid red lines) angular responses at three different diffuser heights h = 0 mm, 1 mm, and 2 mm. Also plotted is the ideal angular response (dashed black line).

The raw simulation data is automatically stored into a text file for further analysis along with the simulation parameters. The figures can be saved in various formats, including PDF, SVG, PNG and TIFF.

Software validation

The diffuser optimization software was validated by comparing the simulated and the measured angular responses of a prototype detector. The simulation results were first matched with the results of a measurement with a non-raised diffuser (h = 0 mm) by tuning the scattering coefficient μ_s. The assumption of uniform scattering (g = 0) turned out to yield the best results for the selected diffuser material. The height of the diffuser was then increased to 1 mm and 2 mm in both the simulations and the measurements, and the angular responses were compared again. Figure 2 shows good agreement between the measured and the simulated angular responses.

Conclusions

A software for optimizing diffusers was developed. The software utilizes Monte Carlo ray tracing to model light transport inside the diffuser. Apart from the shape of the diffuser, the software also takes into account the various surrounding structures that affect the overall angular response, such as the sidewalls of the detector, the shadow ring, and the protective weather dome. A comparison between the simulated and the measured angular responses confirmed that the software can be used to model light transport in the detector structure. Therefore, the software can be used to guide the diffuser optimization process.

Acknowledgement. This report was compiled within the EMRP ENV03 Project "Traceability for surface spectral solar ultraviolet radiation." The EMRP is jointly funded by the EMRP participating countries within EURAMET and the European Union.

References

[1] L. Wang, S. Jacques and L. Zheng, "MCML — Monte Carlo Modeling of Light Transport in Multi-layered Tissues," *Comput. Meth. Programs. Biomed.* **47**, 131–146 (1995).

[2] L. Henyey and J. Greenstein, "Diffuse Radiation in the Galaxy," *Astrophys. J.* **93**, 70-83 (1941).

[3] M. Saito, An *Application of Finite Field: Design and Implementation of 128-bit Instruction-Based Fast Pseudorandom Number Generator*, Master's thesis (Graduate School of Science, Hiroshima University, Japan, 2007) 20 p.

[4] CIE 53 - 1982, *Methods of Characterizing the Performance of Radiometers and Photometers* (International Commission on Illumination, Vienna, Austria, 1982) 27 p.