MSc thesis in Geomatics

High-resolution, large-scale, and fast calculation of solar irradiance with 3D City Models

Longxiang Xu

2024

TU Delft

MSc thesis in Geomatics

High-resolution, large-scale, and fast calculation of solar irradiance with 3D City Models

Longxiang Xu

October 2024

A thesis submitted to the Delft University of Technology in partial fulfillment of the requirements for the degree of Master of Science in Geomatics

The work in this thesis was carried out in the:

3D geoinformation group
Delft University of Technology

Supervisors:	Camilo Alexander León Sánchez
	Dr. Giorgio Agugiaro
Co-reader:	Prof. dr. Olindo Isabella and Dr. Jérôme Henri Kämpf

Abstract

Solar energy is essential for the transition to a net-zero energy system. It serves not only as a renewable energy source but also influences local climate, thereby affecting human energy consumption patterns. However, solar energy is inherently unstable, particularly in urban environments, where shading and sky visibility can change rapidly, even within short distances and timeframes. These characteristics lead to the underutilization of solar energy. As a result, accurately simulating solar irradiance in urban environments and at city-scale is crucial for reducing the gap between predicted and actual energy demand and supply in urban energy systems, helping to minimize energy waste.

This thesis develops, implements, and tests a scalable, high-resolution, and realistic solar irradiance simulation tool using Semantic 3D City Model (3DCM). The methodology specifically accounts for the complexities of urban environments while maintaining manageable computational overhead. It includes shadow calculation based on Bounding Volume Hierarchy (BVH) for direct beam irradiance, viewshed calculation for sky diffuse irradiance, and an iterative global reflective irradiance calculation using semantic viewshed and scene voxelization. The results demonstrate that the method achieves higher accuracy, particularly under clear-sky conditions, while maintaining computational efficiency. For a city model covering 35 km^2 with over 4 million surfaces, 16 million sample points, the simulation for 54 timesteps can be completed in approximately 27 hours on a consumer-level laptop.

Acknowledgements

First and foremost, I want to express my deepest gratitude to my supervisor, Camilo Alexander León Sánchez. His unwavering support throughout my master's journey, both academically and personally, has been an endless source of inspiration and encouragement.

My sincere thanks also go to Dr. Giorgio Agugiaro for his invaluable feedback and guidance, which continually kept me on the right path.

I am especially grateful to my co-reader, Prof. dr. Olindo Isabella, who valuable data and insights has greatly contributed to my work. I would also like to thank Dr. Jérôme Henri Kämpf for his review of my thesis.

A heartfelt thank you to my fellow students for the time we shared. I will always treasure the memories of our journey of growth together.

My dear friends deserve my deepest appreciation. Their love, spanning thousands of miles, has kept me feeling forever connected.

Lastly, I owe my family endless thanks. They have always been my strongest source of strength, and their support has filled me with courage.

Contents

List of Figures

List of Tables

List of Algorithms

Acronyms

1. Introduction

Modelling solar energy resources in urban environments has received much attention in recent years. The WorldBank states that in 2023, 56% of the global population lives in urban areas, and the number is expected to increase to 70% by 2050 World Bank [2023]. The growing urban population density has challenged the urban energy and climate systems. For example, urban heat island Oke [1982] brought by dense built environment in urban areas can increase up to 40% of cooling demand Mauree et al. [2019]. To tackle the issues brought by dense population and buildings in the built environment, whether through urban planning or market adjustment (such as the energy market), quantification of the demand and supply of energy is crucial. Determining incoming solar irradiance is critical as it can not only be the source of the distributed solar power grid but also affect the environment and the occupants' behaviour within the building Diao et al. [2017]. The more accurate the estimation of the solar power available, the better cities will be able to be powered by green and sustainable energy and reach the goal of net zero.

Accurate estimation of the clean energy, represented by solar energy, is essential for the energy system to schedule its usage distribution of energy resources. Incorrect quantification of the current and expected energy demand Allegrini et al. [2012]; Mauree et al. [2019]; Boccalatte et al. [2020]; Wang et al. [2020] and supply Song et al. [2021]; Shaik et al. [2023]; Tercha et al. [2024]; Kwok and Hu [2023] of buildings can lead to erroneous decisions and misguided planning for energy systems Erell and Zhou [2022]; Staffell et al. [2023]. Conversely, accurate estimations can significantly enhance energy efficiency Staffell et al. [2023]. Considering the situation where the predicted output of clean energy is lower than actual values, more fossil fuels that are not necessary will be prepared and used, wasting clean energy. Research indicates that lowering the thermostat by 1°C could reduce Europe's gas consumption by 240 TWh per year, equivalent to one-sixth of historical imports from Russia, highlighting the substantial potential for improving energy efficiency through precise energy demand predictions Staffell et al. [2023].

Despite the hope brought by solar energy for energy transitions, solar energy in urban environments has high variability, and modelling the solar radiation hitting surfaces of urban objects (e.g. a building, a shed, or the solar panels placed on top of them) is rather challenging. The complexity arises from the unique geometrical characteristics of these objects (such as surface tilt and inclination), the position of the sun, and varying weather conditions. Consequently, solar irradiance on surfaces in urban areas can exhibit significant variability in both time and space, even at tiny scales. These abrupt changes in solar energy resources of the solar panels can put excessive pressure on the power grid's consumption of these resources Litjens et al. [2018]. Research suggests that any surface in an urban environment will be able to host solar panels with technology advancement Calcabrini [2023], which indicates there could be more challenges for more accurate prediction of solar energy resources to aid the power grid scheduling. In addition, solar energy also affects the building occupants' behaviour, such as cooling when there is excessive solar energy reaching the building surfaces and heating the building, and heating when there is less solar energy reaching the

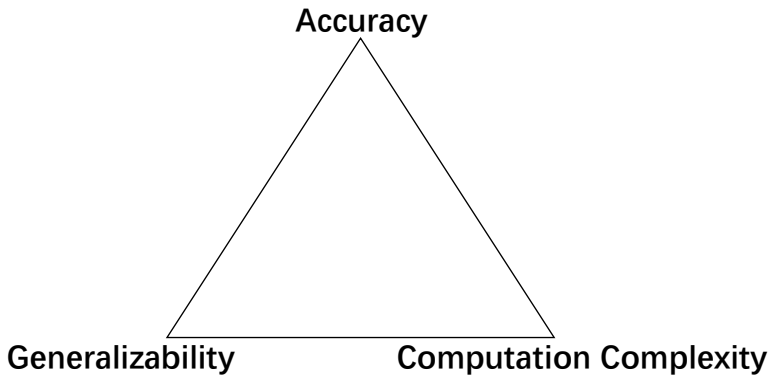


Figure 1.1.: The impossible triangle of current methods for solar irradiance simulation. The three aspects are essential for city-scale simulations

building surfaces. Therefore, accurate solar radiation estimation in urban environments is highly beneficial.

There are several significant methods to predict solar irradiance. RT models simulate light behaviour (travelling with fixed direction) and interactions like reflections and scattering, providing detailed and realistic irradiance simulations Ward [1994]; Erdélyi et al. [2014]; Liang et al. [2014, 2020]; Jaillot et al. [2017]; Wang et al. [2023]; Xu et al. [2023]. Empirical methods are data-driven, typically using empirical equations or techniques like Recurrent Neural Networks (RNN) and Long Short-Term Memory (LSTM) networks to predict solar irradiance Loutzenhiser et al. [2007]; Nielsen et al. [2021]; Gneiting et al. [2023]; Tercha et al. [2024]. Viewshed methods merge vector-tracing and empirical methods techniques to calculate Sky View Factor (SVF) from viewshed maps, usually estimating irradiance over longer durations Fu and Rich [1999]; Tabik et al. [2013]; Liang and Gong [2017]; Oh and Park [2018]; Stendardo et al. [2020]. Pixel-counting methods utilize game engines to relate RGB values in rendered images to solar irradiance through mapping functions Ward [1994]; Hegazy et al. [2021]; Meines [2023]; Epic Games [2023]; Unity Technologies [2023].

However, for RT methods, computational demand grows exponentially with increased calculation depth, limiting scalability for city-wide applications. For Time-series methods, the short-term prediction by these methods can be accurate, but the data collection and training have to be repeated for different surfaces. The viewshed methods often oversimplify the 3D urban environment. Moreover, for pixel-counting methods, simplifications for visual effects in game engines can create inaccuracies like false shadows Piórkowski and Mantiuk [2015]; Piórkowski et al. [2017]. Recalibration and recomputation are also needed for changed solar positions.

These methods prioritize different aspects of solar irradiance computation, focusing on either accuracy, computational efficiency, or generalizability. However, as illustrated in figure 1.1, none of the existing approaches can achieve all three simultaneously, which is critical for city-scale simulations.

This thesis introduces a method that balances these three essential characteristics: accuracy, scalability, and long-term prediction capabilities. The proposed framework offers a scalable and realistic approach to solar irradiance simulation by combining Ray Tracing and Viewshed techniques. This approach holds significant potential for advancing urban climate modelling and energy systems research.

1.1. Research Objectives

1.1.1. Research questions

The main research question of the thesis is:

How can a solar irradiance simulation tool be developed to balance accuracy and computational efficiency for city-scale simulations while utilizing semantic information from 3D city models?

This question is extended further by the following sub-questions:

- In what ways can semantic data derived from 3D city models refine the precision of solar irradiance simulations by considering the direct, diffuse and reflected solar components?
- What are the potential trade-offs between accuracy and computation simplification when utilizing 3D city models for estimating solar irradiance at an urban scale?

1.1.2. Scope of research

The thesis will focus on developing and testing a method to compute large-scale, high-resolution solar irradiance with 3D city models to support energy transition, architecture design, and urban planning. A model with case study values will be the final product.

2. Theories and Background

2.1. Semantic 3D City Model

To account for the variations brought by the complexity of the built environment for solar potential analysis, it is essential to consider the geographical location and surroundings. The required information for such consideration can be obtained from semantic 3DCM Agugiaro et al. [2020]. With 3DCM, we can derive the required parameters to accurately determine the local solar irradiance value on any surface, such as surface orientation and tilt, and shadowing effect.

A 3DCM is a digital twin of the urban environment with three-dimensional geometries of urban objects and structures, with buildings as the most prominent feature Ohori et al. [2024]. The structure, format, and characteristics vary significantly as typical 3DCM are reconstructed from various acquisition techniques, such as photogrammetry, laser scanning, extrusion, conversion from architectural and drawings, procedural modelling, and volunteered geoinformation Biljecki et al. [2015].

3DCM allow the representation of relevant city objects (and their sub-parts) to be organized in a more structured manner, with geometry and semantic attributes. More specifically, the urban environment is modelled and decomposed hierarchically Ohori et al. [2024]. For example, a city can be decomposed into classes such as 'building', 'road', or 'vegetation'. Moreover, a building can be further decomposed into a 'building part', which can be further decomposed into several surfaces; examples include wall, roof, or ground.

The Open Geospatial Consortium (OGC) CityGML standard has been developed to represent 3DCM to improve interoperability while preventing the redefinition of urban environment decomposition Gröger and Plümer [2012]. The data model has provided a unified framework for representing semantic 3d models. For the thesis, several key relevant concepts of the CityGML data model will be introduced.

Boundary representation of geometry Rather than "modelling a 3D object through a volumetric representation, " it implicitly models the object by representing the 3D surfaces that bound it. This modelling technique provides a flexible way to represent arbitrary 3D volumes. In the thesis, the geometries of the urban objects are, by default, considered with boundary representation.

Geometry template Other than explicitly modelling the geometry of urban objects, an alternative way is to use geometry templates for urban objects that generally do not vary significantly in shape. Examples include road lamps, where each road lamp is identical in geometry but differs in geographic location. In the thesis, the trees that are being modelled are stored as implicit geometries in the KNMI weather station dataset.

Level of Detail LoD in CityGML standard are formalised definitions of the granularity of representing the urban objects Biljecki et al. [2016]. There are five main LoDs, LoD0, LoD1, LoD2, LoD3, LoD4. LoD0 is a 3D footprint representation of urban objects. LoD1 is a box model, where 3D models are normally derived by extruding the 2D footprint to a given height. In LoD2, the generalised roof shape and larger roof superstructures are present. Lod3 enable more detailed modelling of building openings such as windows and doors, chimneys, and other facade details. LoD4 models the interior structure within buildings. In the thesis, LoD2 will be the most common LoD adopted as it is the highest LoD available from open data. Meanwhile, it provides sufficient detail for solar irradiance modelling of building exterior surfaces.

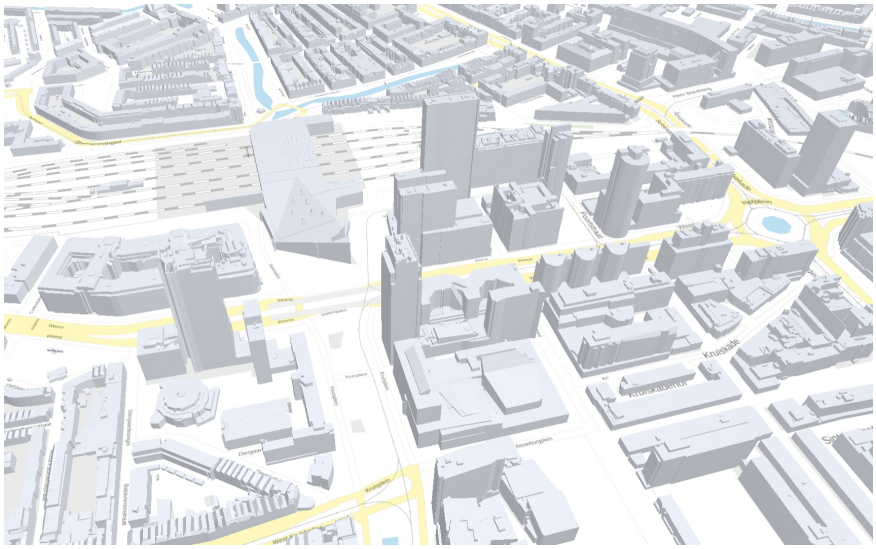


Figure 2.1.: Semantic 3D City model. Screenshot of the the semantic 3D city model in Rotterdam. Screenshot retrieved from 3DBAG online viewer.

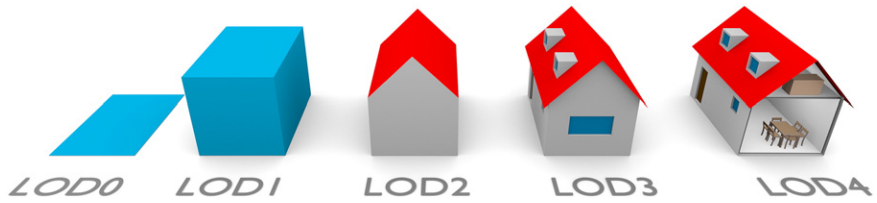


Figure 2.2.: Illustration of a single building in different LoD. Figure retrieved from Biljecki et al. [2016]

2.2. Ray Tracing and Global Illumination

The general RT technique refers to the simulation of light travelling, which could reflect, diffuse, and travel along a straight line in space. In the more specific region of rendering, RT refers to "A rendering technique in 3D computer graphics that calculates the color of

pixels by tracing the path that light would take if it were to travel from the eye of the viewer through the virtual 3D scene."

The RT that refers to rendering can be described in Kajiya's rendering equation:

$$L_o(x, \omega_o) = L_e(x, \omega_o) + \int_{H^2} f_r(x, \omega_o, \omega_i) L_i(x, \omega_i) \cos\theta_i d\omega_i \tag{2.1}$$

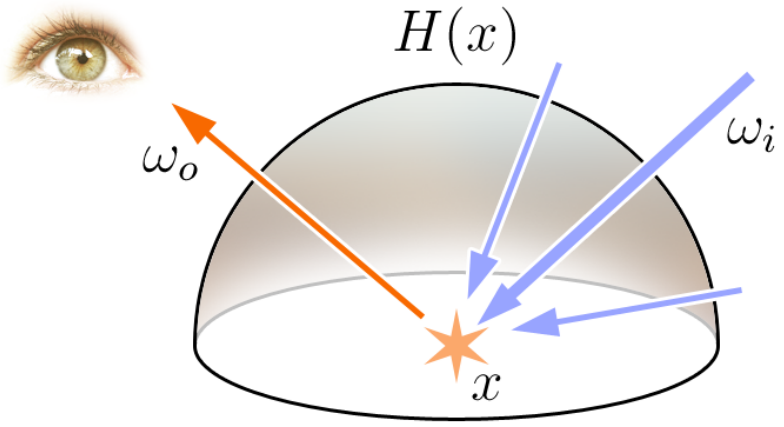


Figure 2.3.: Illustration of the rendering equation. Figure retrieved from Contributors [2024]

where $L_o(x, \omega_o)$ refers to the outgoing irradiance from position x to direction ω_o. In the context of rendering, this is the observed irradiance value or color when looking at the surface at point x from direction ω_o. $L_e(x, \omega_o)$ represent the emitted irradiance. In most case where x is not a light source, the black body radiation can be ignored. The latter term represent the reflected irradiance. Where $\int_{H^2}$ represent the hemisphere at point x. $f_r(x, \omega_o, \omega_i)$ is the Bidirectional Reflectance Distribution Function (BRDF), which describes how incoming light from direction ω_i is reflected at the surface point x to direction ω_o. $L_i(x, \omega_i)$ is the incoming irradiance from direction ω_i. $\cos\theta_i$ is the cosine of the incident angle of light from direction ω_i at point x. An illustration of the equation is shown in 2.3.

In the case of solar irradiance simulation, what is of interest will be the specific term: $\int_{H^2} L_i(x, \omega_i) \cos\theta_i d\omega_i$, the solar irradiance received at the point of interest x. However, the deriviation of this term would further require a recursive calculation as each of $L_i(x, \omega_i)$ is also derived from $L_o(x_i, \omega_i)$. The multibounce of light would indicate computation that grows exponentially as seen in figure 2.4.

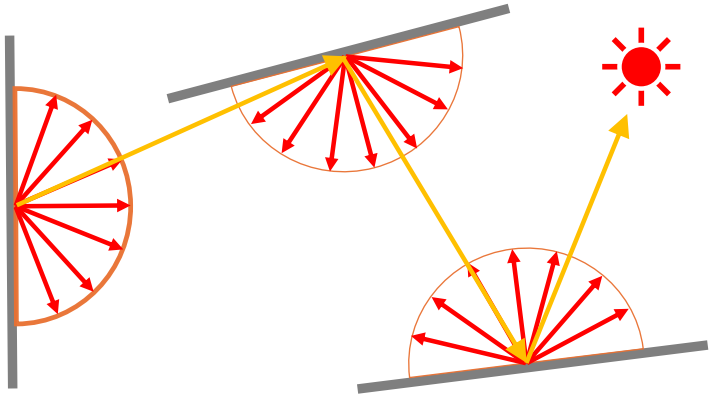


Figure 2.4.: The illustration of multi bouncing of light in the backward RT. Notice for each intersection, a new set of rays need to be sampled to simulate the light reflection.

Instead of tracing rays that could grow exponentially to consider the indirect reflected or diffused irradiance in the scene, there are other techniques to consider the global indirect irradiance.

Photon mapping with irradiance caching Jensen et al. [2002] is a commonly used solution for global irradiance computation. Photon mapping generally works as follows: First, rays are cast from the light sources, the path of each ray will be traced. Once a ray hit a surface, a hit record will be generated, representing a new light source. New rays will be shot from the new light source. The iteration will continue, until the energy is absorbed or a maximum depth is reached. With this photon mapping being constructed, when considering the indirect irradiance received at the point of interest x, it is only necessary to do one hemisphere sampling, for each ray in the hemisphere, the cloeset photon hit records will be found and interpolated to obtain the corresponding irradiance. Photon mapping method avoids the exponentialy growing rays that need to be traced when evaluating the irradiance of a point of interest. However, in the scenario of solar irradiance modelling, the photon mapping needs to be reconstructed each time the solar position changes, which makes this method unsuitable for large-scale, long-term simulation.

Radiosity is another technique that is able to preserve the simulation of light bounces while maintaining a manageable computation complexity. The method works according to following equation:

$$L(x) = L_e(x) + \gamma_{refl} \int_{H^2} F_{x,y_i} L(y_i) d\omega_i \tag{2.2}$$

where $L(x)$ represent the radiosity of point x, $L_e(x)$ represent the self-emission, γ_{refl} represent the albedo, and F_{x,y_i} represent the form factor between points x and y_i, and $L(y_i)$ represent the radiosity of point y_i. The function describes the outgoing irradiance from point x is dependent on the outgoing irradiance of x's visible patches and their geometric relationship.

The form factor F_{x,y_i} is defined as:

$$F_{x,y_i} = \frac{\cos(x, y_i)cos(y_i)}{\pi r^2} \tag{2.3}$$

where $\cos(x, y_i)$ represents the angle away from the normal of x and the incoming direction from y_i, and $cos(y_i)$ represents the angle away from the normal of y_i and the outgoing direction from y_i to x.

Discritizing equation (2.2), we can obtain:

$$L(x) = L_e(x) + \gamma_{refl} \sum^{n} F_{x,y_i} L(y_i) \tag{2.4}$$

L_e is generally ignored or known, γ_{refl} is known, F_{x,y_i} is computable. For all the sample points in the scene, we can obtain such equation. These equations forms a linear system that can be solved.

Radiosity methods provide good inspirations as it also eliminates the need to trace rays that will grow exponentially. But this method has not been adopted in large scale due to its complexity and high-memory requirement.

The thesis proposed a method that has a similar spirit of Photon mapping and Radiosity, but with more efficient computation techniques including aggregation, caching, lookup etc.

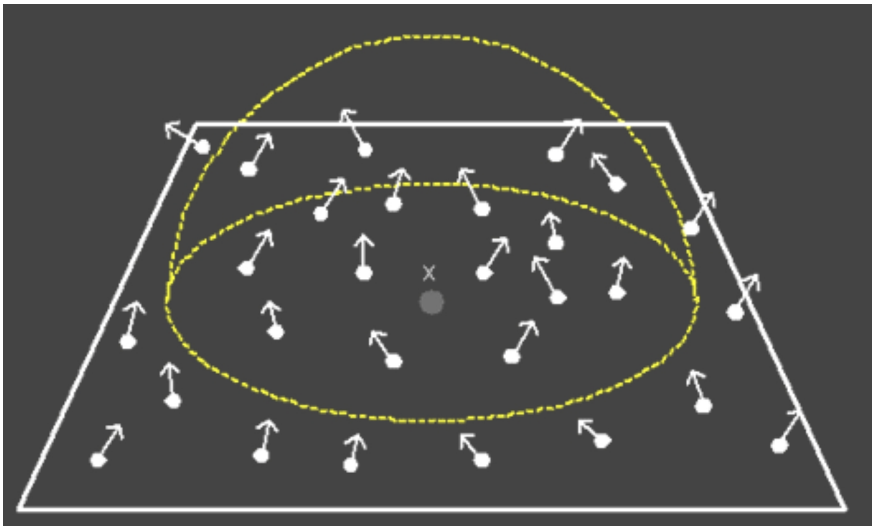


Figure 2.5.: Illustration of photon mapping technique. Figure retrieved from Komura [2013]

2.3. Solar Irradiance Simulation

It is essential to utilise a transition model based on meteorological data to analyse the solar irradiance impacting tilted surfaces. This data provides measurements of solar irradiance that reach the ground, which are expressed by equation (2.5) Loutzenhiser et al. [2007].

$$GHI = DHI + DNI \cdot \cos(\theta_Z) \tag{2.5}$$

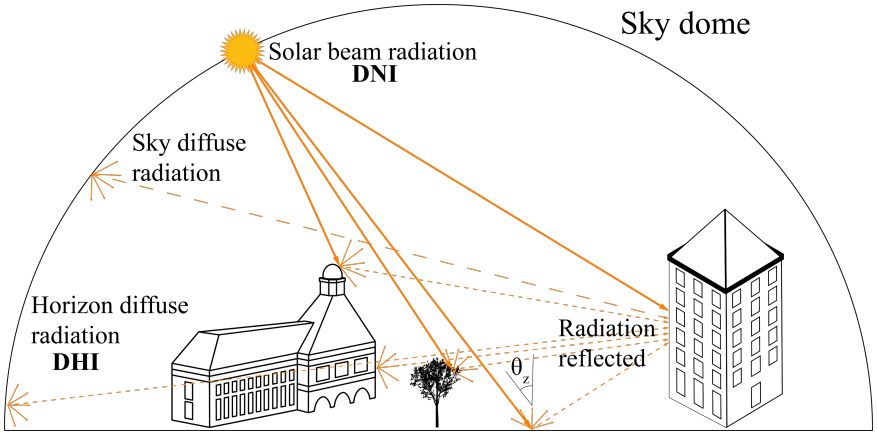


Figure 2.6.: Solar irradiance components. Figure retrieved from Xu et al. [2024]

Global Horizontal Irradiance (GHI), Diffuse Horizontal Irradiance (DHI), Direct Normal Irradiance (DNI) are key metrics recorded in weather data, essential for understanding solar energy potential and distribution. GHI refers to the total solar irradiance received on a horizontal surface, which includes both direct sunlight and scattered light from the sky. DHI measures the irradiance from the sky, excluding direct sunlight, and is crucial for assessing the potential for diffuse sky radiation to contribute to solar energy generation. DNI is the irradiance received on the surface that is perpendicular to the sun's rays and is a measure of the direct beam solar energy. θ_Z represents the solar zenith angle. The diffuse components are divided into sky and horizon figure 2.6. The general form of the transition model is formulated as equation (2.6).

$$I_{S,\beta} = I_{S,dir,\beta} + I_{S,diff,\beta} + I_{S,refl,\beta} \tag{2.6}$$

Where $I_{S,dir,\beta}$, $I_{S,diff,\beta}$, $I_{S,refl,\beta}$ represent the direct beam irradiance, sky diffuse irradiance, and ground reflected irradiance. $I_{S,\beta}$ represent the total solar irradiance and β stands for surface inclination. The determination of the direct beam solar irradiance is articulated as equation (2.7).

$$I_{S,dir,\beta} = M_{shadow} \cdot DNI \cdot \cos\delta \tag{2.7}$$

Here, M_{shadow} represents the binary shadow mask and δ represent the angle between surface normal and the solar vector.

Over the past several decades, there has been extensive research on estimating the diffuse sky component of total solar irradiance Loutzenhiser et al. [2007]. Typically, these methods involve analyzing the visible part of the sky. The level of detail considered in these analyses necessitates different kinds of input data. In the thesis, *Isotropic model* Kamphuis et al. [2020] will be adopted for sky diffuse solar irradiance. The isotropic model considers the sky as a uniform source of diffuse radiation. The decision to adopt this model in the thesis is primarily driven by its implementation simplicity, which is a critical factor given the study's emphasis on simulating reflective solar irradiance.

Moreover, the thesis acknowledges the dominant role of direct solar irradiance in contributing to the overall solar energy received under clear-sky conditions. Given these considerations, the isotropic model has been selected to estimate sky-diffuse solar irradiance. The

isotropic model is chosen for its ability to provide a straightforward approach to accounting for the diffuse components of solar irradiance that originate from the sky.

The diffuse components of solar irradiance in the isotropic model can be formulated by equation (2.8).

$$I_{S,diff,\beta} = DHI \cdot SVF \tag{2.8}$$

In most cases, SVF is simplified Heim and Knera [2021] as:

$$SVF = \frac{1 + \cos\beta}{2} \tag{2.9}$$

The reflective irradiance calculation methods can be categorised into two primary approaches: simplified Ground View Factor (GVF) methods and RT methods. GVF-based methods assess the reflective solar irradiance from surrounding objects by a straightforward calculation. That is achieved by multiplying the global horizontal irradiance by the GVF and a predetermined albedo value. The GVF plays a pivotal role in various solar irradiance estimation models, including the Perez Perez et al. [1990] and isotropic models. These models are typically represented as follows:

$$GVF = \frac{1 - \cos\beta}{2} \tag{2.10}$$

Where θ is the surface inclination. This estimation oversimplifies the surrounding environment, ignoring shading behavior and surface types. In this case, the resulting ground reflected solar irradiance will be:

$$I_{S,refl,\beta} = GHI \cdot \gamma_{refl} \cdot GVF \tag{2.11}$$

Where γ_{refl} represents the albedo.

For RT based methods, the reflected solar irradiance is determined by tracing rays in the hemisphere as illustrated in Equation equation (2.1):

$$I_{S,refl,\beta} = \int_{H^2} L_i(x, \omega_i) \cos\theta_i d\omega_i \tag{2.12}$$

Although importance sampling Bako et al. [2019] can reduce the number of rays need to be traced will still grow exponentially with the number of bounces the calculation will consider.

3. Related work

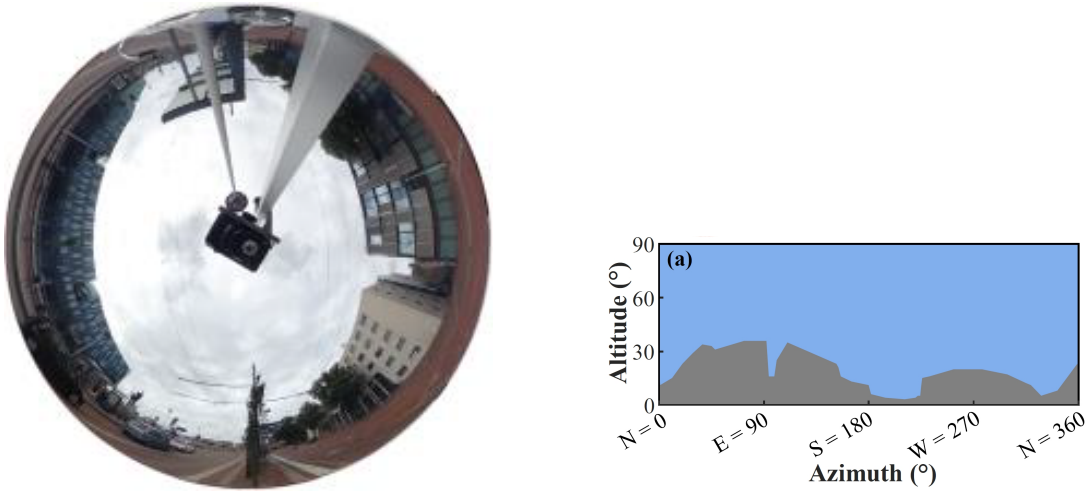
Over the past decades, researchers have investigated methods for calculating solar irradiance while considering the surrounding environment. These methods can be categorized as view-shed based methods, pixel counting methods, ray-tracing methods, and empirical methods.

3.1. Viewshed-based methods

The viewshed method simulates the view from the perspective of a shadow-receiving position. The general workflow involves sampling rays from a hemisphere originating at the test position. Each ray undergoes an intersection test, producing a viewshed map representing the test point's surroundings which support the solar irradiance calculations. Direct solar irradiance is determined by projecting the sun's location onto the viewshed map; if the solar position overlaps with surrounding objects, the direct beam solar irradiation is masked out. Diffuse solar irradiance is calculated based on the visible sky patch in the viewshed.

Fu and Rich Fu and Rich [1999] developed one of the first viewshed-based methods. Their approach involves generating a viewshed map and a sun map that plots the sun's locations in the same hemispheric coordinate system. The overlay of the sunmap and the viewshed map supports solar irradiance calculations. However, this method only supports 2.5D digital elevation model (DEM) with limited resolution. Subsequent research has focused on optimizing computation through GPU acceleration Tabik et al. [2013]; Stendardo et al. [2020], voxel octree data structures Liang and Gong [2017], and data pyramids Oh and Park [2018].

Novel methods for deriving viewsheds with 3D data, such as fisheye image rendering Liang et al. [2020], have also been proposed. While most models can consider diffuse and reflective components due to the embedded surroundings and sky patches in the viewshed map, they do so in a simplified manner. Furthermore, most methods cannot handle 3D data with high levels of detail (LoD2 or above), and computational requirements and the availability of 3D environmental data often limit the resolution of viewsheds.



(a) The scene

(b) The resulting viewshed

Figure 3.1.: Illustration of viewshed-based method. Figure retrieved from Calcabrini [2023]

3.2. Pixel counting methods

Pixel counting methods leverage modern rendering engines that utilize GPU acceleration. These methods involve rendering images of areas of interest and recording the pixel values of the planes of interest. By validating with ground measurements, a mapping function between pixel values and solar irradiation can be derived, enabling accurate predictions of solar irradiation. Modern rendering engines, designed with physically based rendering principles, can simulate the physical behavior of light, including diffusion and reflection, providing accurate results Hegazy et al. [2021]; Ward [1994]. However, these methods can be computationally intensive due to the exponential growth of ray path tracing from diffusion, and the need to render new images each time lighting conditions change Andres et al. [2023].

Recent rendering engines like Unreal Engine Epic Games [2023] and Unity Unity Technologies [2023] are designed for photo-realistic real-time rendering, showing great potential for simulating solar irradiance at a city scale. However, since they are optimized for visualization, sample values on surfaces must be manually extracted after rendering Meines [2023]. Additionally, these sample values are deeply coupled with the defined surface material and light source properties. For urban-scale simulations with high temporal resolution, the manual effort required to account for extraterrestrial irradiance changes, camera position settings, and the lack of surface material information make rendering engines unsuitable for most users.

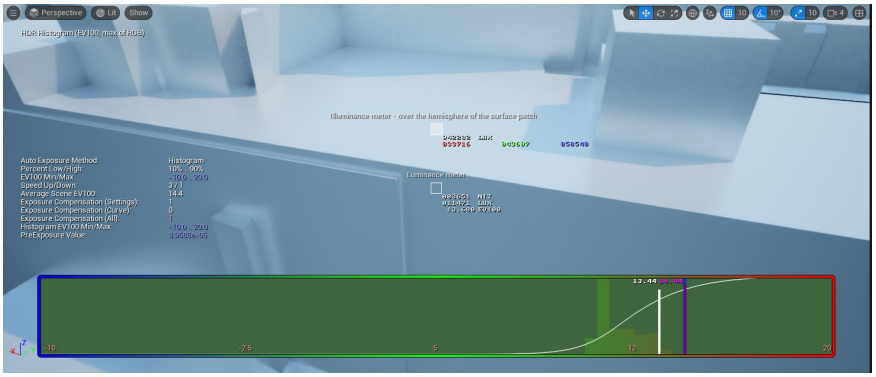


Figure 3.2.: Illustration of pixel counting method. Figure retrieved from Meines [2023]

3.3. Ray Tracing based methods

RT methods determine solar irradiation by casting ray vectors, and they can be categorized into two types: forward RT and backward RT. Forward RT involves casting rays directly from the solar position, but this approach is often considered inefficient because many rays do not intersect with the plane of interest, leading to redundant computations Arias-Rosales and LeDuc [2022]. Backward RT is more commonly used as it casts rays originating from the plane of interest toward the sun followed by tests for intersections with objects. The intersection results directly affect the calculation of direct beam solar irradiance.

The SORAM model is a RT-based solar irradiance calculation method that utilizes 3D city models. For a given point of interest, hemispheric sampling is applied, and energy is accumulated for each sampled ray. However, the 3D models are limited to box representations, and the diffuse and reflected solar irradiance from the surroundings are excluded from the calculations Erdélyi et al. [2014]. Wieland et al. developed one of the first models to use semantic 3D city models, employing RT methods to account for more complex urban environments Wieland et al. [2015]. In this model, a single ray is cast from the point of interest toward the sun, ignoring the reflected component and simplifying the calculation of the diffuse component without considering the surrounding environment. SURFSUN3D is another model that applies RT to account for shadowing effects in solar irradiance computation. To reduce computation, it employs view-frustum culling and radius culling methods to decrease the number of ray-object intersection tests. However, the model also disregards the reflected and diffuse components from surrounding objects, directly computing these components using the r.sun model Liang et al. [2014, 2015]. To minimize redundant ray-object intersections, various techniques such as solar azimuth filtering, nightside filtering Wang et al. [2023], semantic bounding volume hierarchy Jaillot et al. [2017], and bounding volume hierarchy Xu et al. [2023] have been employed. These methods enable fast and accurate determination of direct beam irradiance but make little consideration for reflected and diffuse irradiance.

Essentially, viewshed-based methods and pixel-counting methods are also RT-based methods. The difference is that viewshed methods precompute a viewshed map while ray-tracing methods determine shadowing effects on the fly. Pixel-counting methods are indirect, sampling values from rendered images, whereas RT methods provide direct results.

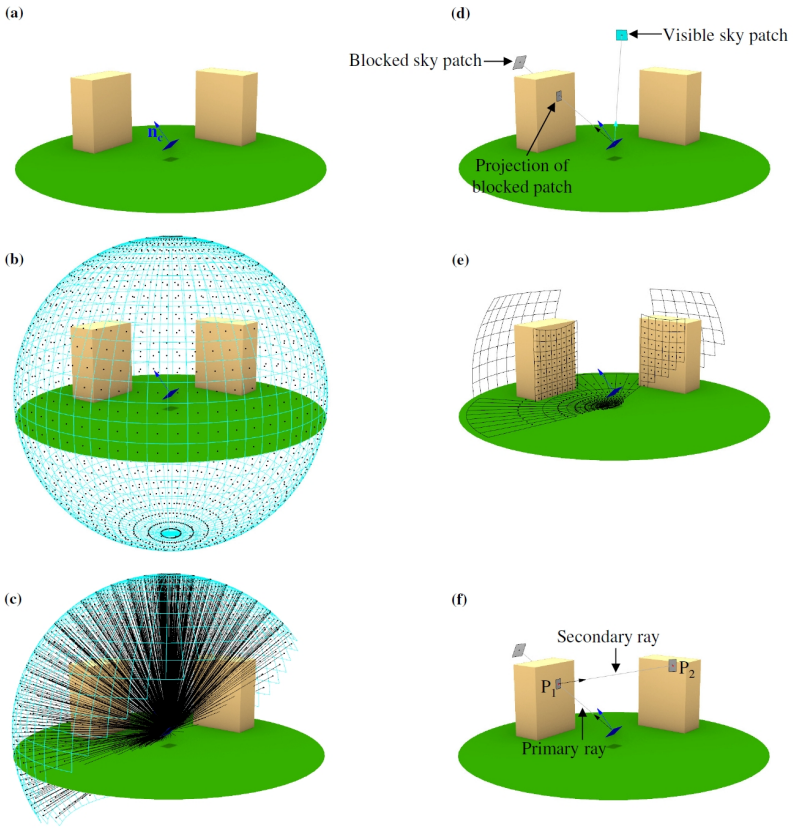


Figure 3.3.: Illustration of RT method. Figure retrieved from Andres et al. [2023]

3.4. Empirical Methods

Empirical methods predict solar irradiance based on historical data, often supplemented by external parameters. One of the earliest models is the Perez model Perez et al. [1987], which uses coefficients derived from historical data, combined with surface inclination and orientation, to generate predictions. Other models in this category include the Hay and Davies model Hay [1993], the Klucher model Klucher [1979], and the Reindl model Reindl et al. [1990a,b]. These methods rely on empirical equations to convert weather data into solar irradiance estimates for specific surfaces. While they offer good generalizability, their simplifications and broad applicability can lead to significant deviations from reality in specific locations.

In recent years, advancements in machine learning techniques, particularly RNN and LSTM networks, have significantly improved solar irradiance prediction Nielsen et al. [2021]; Gneiting et al. [2023]; Tercha et al. [2024]. These deep learning models eliminate the need for manual feature engineering or empirical equation derivation, relying primarily on historical data for training, and sometimes incorporating satellite imagery. While these models can achieve high accuracy for short-term forecasts, they lack the ability to sense or account for

3D environmental factors. Consequently, for new surfaces or varying climate conditions, additional historical data must be collected, and the models need to be retrained. As a result, existing empirical methods either lack precision for specific surfaces or lose generalizability across different scenarios.

3.5. Solar Irradiance Simulation with 3D City Models

In the methods discussed above, the input data formats are not standardized, ranging from DEM, meshes, and fisheye images to environmental representations with undefined data models. This heterogeneity in data types makes the methods and their results difficult to reproduce, especially when data models lack detailed specifications.

An increasing number of studies are turning to standardized 3DCM models, such as CityGML, for input data. The required information for accurate solar irradiance analysis can be derived from semantic 3DCM models, as highlighted by Agugiaro et al. [2020]. These models enable the extraction of detailed information about urban environments, including buildings, terrain, and vegetation, which is essential for accurately determining solar irradiance on any surface.

Several studies have developed solar irradiance simulation tools based on 3DCM. One of the earliest tools is SimStat Rodríguez et al. [2017], which provide functionalities to evaluate solar energy potential using 3DCM. However, SimStat does not explicitly model the shading effects from surrounding objects, the visible sky, or reflected solar irradiance. Instead, it applies a reduction coefficient to simulate shading effects. Similarly, CitySim is a software that offers a solar potential simulation tool Walter and Kämpf [2015]; Giannelli [2021]; Giannelli et al. [2022].

Wieland et al. introduced one of the first methodologies that utilize 3DCMç while accounting for precise shading effects Wieland et al. [2015]. Their method uses the PostGIS function "ST_3DIntersects" PostGIS [2024] to calculate shadowing. However, this function can only test intersections between a single sun ray and a single geometry, requiring the model to exhaustively traverse through all geometries for each ray, resulting in computational complexity that scales quadratically with the size of the scene. Additionally, their method does not model the sky diffuse component in detail.

Hurkmans improved the ray-object intersection tests by incorporating an R-tree data structure, which accelerates the process Hurkmans [2022]. However, the R-tree is two-dimensional and only filters neighboring buildings. This limitation introduces errors in environments with significant elevation changes, such as urban areas with high-rise buildings or mountainous regions. In addition, Hurkmans' study does not consider the sky's diffuse and reflective components.

It is important to note that, although existing models are integrated with 3DCM, they do not fully exploit the detailed geometry and semantics available for comprehensive solar irradiance computation. Simplifications are often made, not due to a lack of geometric or semantic data, but to avoid the excessive computational demands imposed by current simulation methods, which are not optimized for city-scale, high-resolution simulations. This highlights the need for new methods that can fully leverage 3DCM datasets and modern computational power to enable realistic, city-scale solar irradiance simulations at a high resolution.

4. Methodology

The methodology consists of four key components: Point Grid Generation, Viewshed Calculation, Shadow Calculation, and Solar Irradiance Calculation, as illustrated in Figure 4.1.

First, in Point Grid Generation, points are sampled on the surfaces of the 3D city model to form a grid of samples representing the points of interest.

Next, Viewshed Calculation generates a viewshed for each sample point, representing the 3D environment surrounding it. This step captures the visibility of the surroundings from each point.

In the Shadow Calculation step, the shadow effect is simulated for each sample point at different time steps, creating a binary mask to indicate whether a point is in shadow or sunlight at any given time.

Finally, the Solar Irradiance Calculation be applied to transport the weather data to actual irradiance received at each sample points, taking into account the viewshed and shading effects determined in the previous steps.

The details of these steps will be discussed in the following subsections.

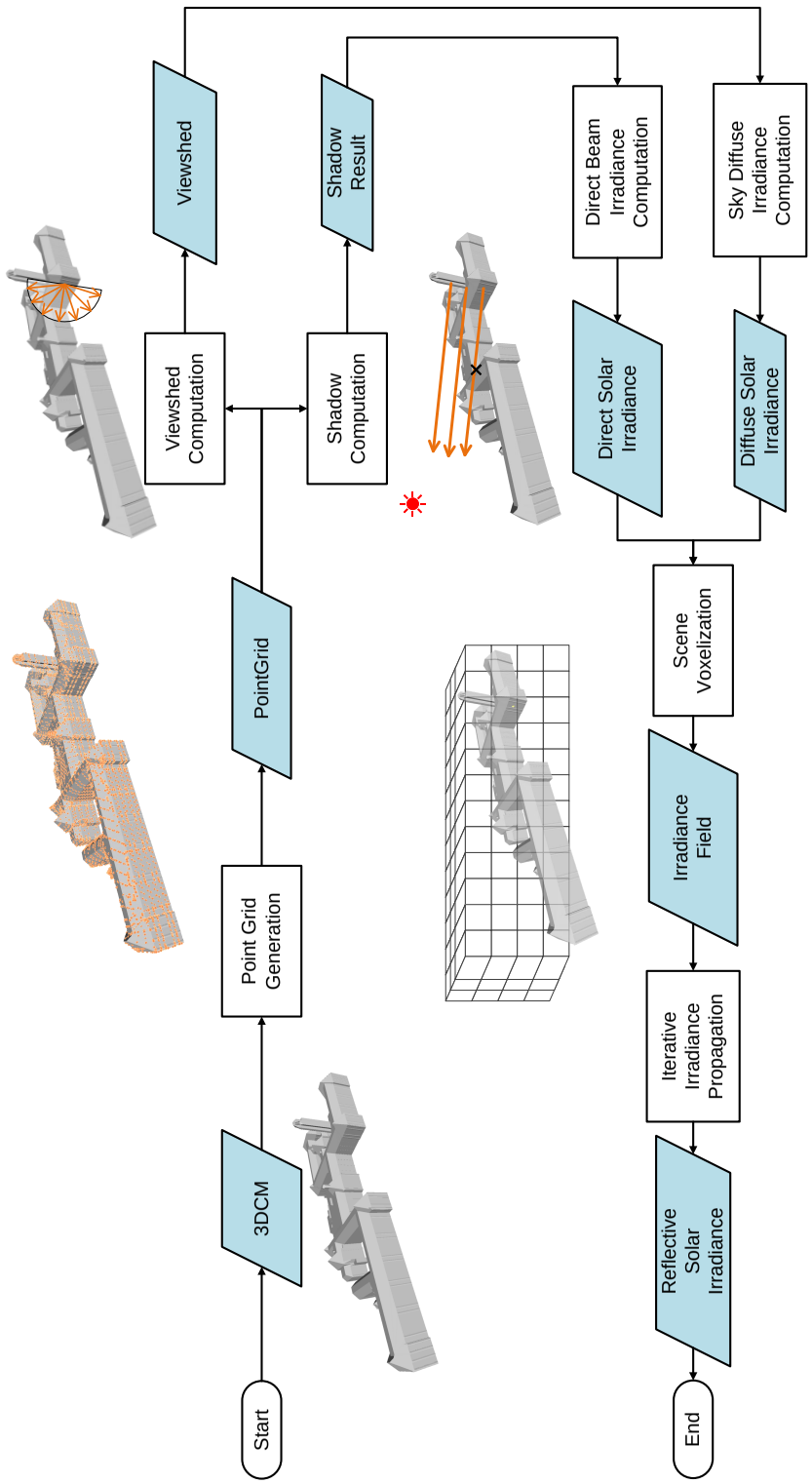


Figure 4.1.: The workflow diagram of the methodology

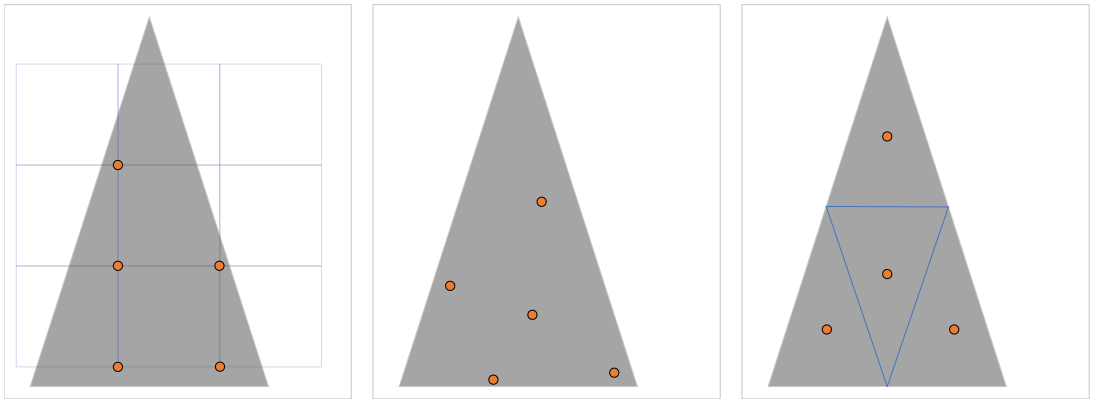
4.1. Point Grid Generation

This step aims to create a set of samples that can describe the scene's solar access. Therefore, it is necessary to sample points on the scene surfaces. In the case of 3DCM, their triangulated geometry is used for simplicity. Three main strategies for uniformly sampling points on triangle meshes are probability distribution function Probability Distribution Function (PDF) based, projection-based, and recursive splitting methods.

In this work, we adopt the recursive splitting method. In city-scale simulation, the number of samples must be manageable. In the solar irradiance simulation, the number of points generated using a PDF-based approach may be too small to ensure uniform distribution across the triangle mesh. For the projection-based method, each triangle in 3D space must be projected onto its underlying plane; then a uniform grid needs to be constructed on the plane, followed by constructing a uniform grid on the plane and checking whether each point lies within the triangle. However, this method may not provide holistic coverage, particularly near acute angles of the triangle.

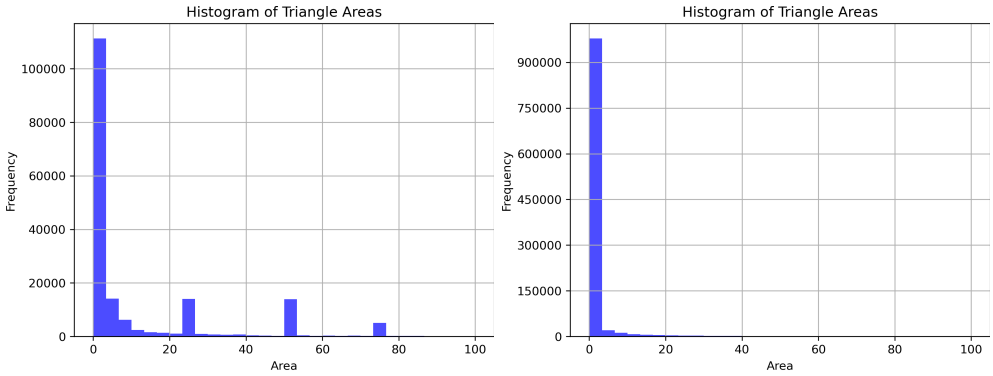
We use the recursive splitting method to address these limitations, which is better suited for city-scale solar irradiance simulations. This method calculates the number of splits required to achieve the desired sampling density. For each split level, we split the triangle by connecting the three midpoints on the three edges. The centroid of each new triangle is then computed and used as a sample point. The algorithm is described in algorithm 4.1 and figure 4.4. It is important to note that due to the splitting process, the density of sampled points is not perfectly uniform across the study area. For example, if a triangle has an area of 16 m^2 and the sampling density is 4 m^2, one split is required. However, for a triangle with an area of 18 m^2, two splits are necessary. This can result in triangles with similar areas having significantly different numbers of sampled points. Despite this, the sampling density generally ensures even coverage across arbitrary triangles.

A comparison of sampling methods is shown in figure 4.2. In the experiment of the thesis, a typical scenario involves a triangle with an area smaller than 20 m^2, as indicated in figure 4.3, where a sampling density of 4 m^2 per point is required, leading to around five sample points. It can be observed that the recursive splitting method provides better stability and coverage of the entire triangle. In contrast, grid sampling places points near the edges, and the randomness of PDF sampling can lead to poor coverage.



(a) Grid sampling (b) PDF sampling (c) Split sampling

Figure 4.2.: Comparison of the three sampling methods



(a) Heino weather station dataset (b) TU Delft dataset

Figure 4.3.: The surface area distribution of the study areas in the thesis

Algorithm 4.1: Algorithm to Calculate Grid Points and Mass Centers in a Delaunay Tetrahedralization

Input: Triangle meshes $(meshes)$, maximum recursion depth (max_depth), the area represented by each sample point (s_τ)
Output: A list of grid points (P)

1 **foreach** $triangle \in meshes$ **do**
2 $depth \leftarrow 0$;
3 $p \leftarrow (triangle.A + triangle.B + triangle.C)/3$;
4 `append`(P, p);
5 $s \leftarrow$ `calculate_area`$(triangle)$;
6 $num\_split \leftarrow \log_3\left(\frac{2\times s}{\text{density}} + 1\right)$;
7 $depth \leftarrow depth + 1$;
8 **if** $num\_split > 0$ ***and*** $depth < max\_depth$ **then**
9 split the $triangle$ into three new triangles $new\_triangles$ by connecting p and the three vertices;
10 **foreach** $new\_triangle \in new\_triangles$ **do**
11 loop back to step 3
12 **else**
13 terminate the recursion;
14 **return** P

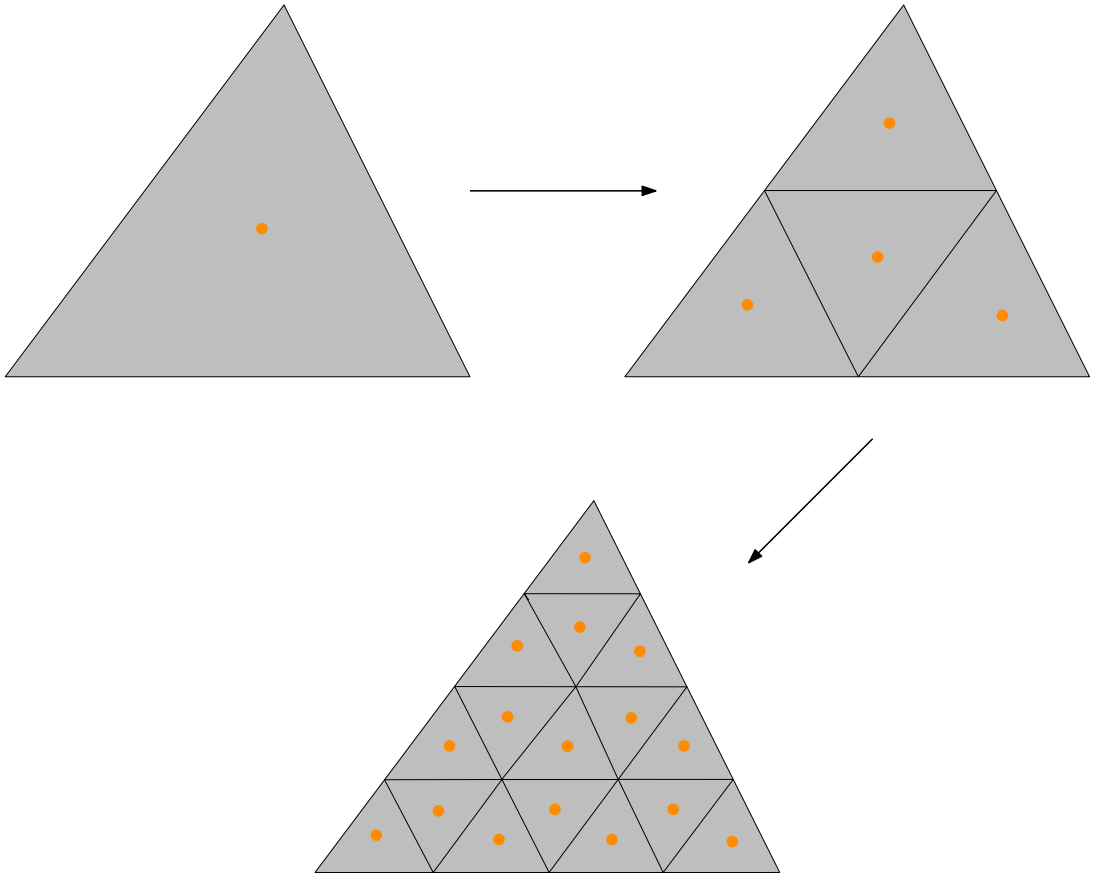


Figure 4.4.: The triangle splitting process for grid point sampling

4.2. Shadow Calculation

This step will calculate the shadowing effect essential for direct beam solar irradiance. The solar positions of the desired calculation times are first obtained. For each sample point, a ray that is originating from the point, pointing towards the sun will be traced as illustrated in figure 4.5. The result for each point and at each timestep will be stored as binary values, where 1 indicates the point is not shadowed, and 0 indicates it is in shadow. This binary mask will be applied to the direct beam solar irradiance to account for the shading effect of the surroundings.

The ray-object intersection test is at the core of shadow calculation, similar to the one used in viewshed calculation. Previous methods, such as radius culling, view-frustum culling, and nightside culling, have been employed to reduce the number of intersection tests required. However, even with these optimizations, the number of mesh surfaces to test remains significant, and these simplifications may introduce errors in complex environments, such as areas with high-rise buildings or mountainous terrain. Here we adopt the BVH strategy in Xu et al. [2023] reduces the time complexity of the intersection tests from $\mathcal{O}(NM)$ to $\mathcal{O}(Nlog(M))$, making the shadow calculation process more efficient without sacrificing accuracy.

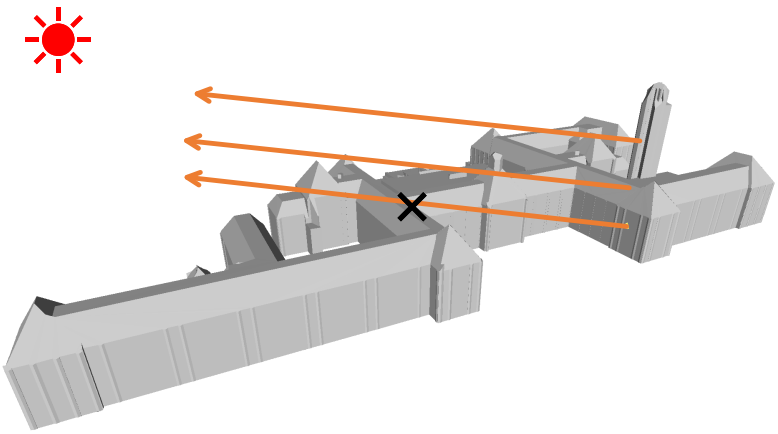


Figure 4.5.: Illustration of shadow calculation based on ray-object intersection test

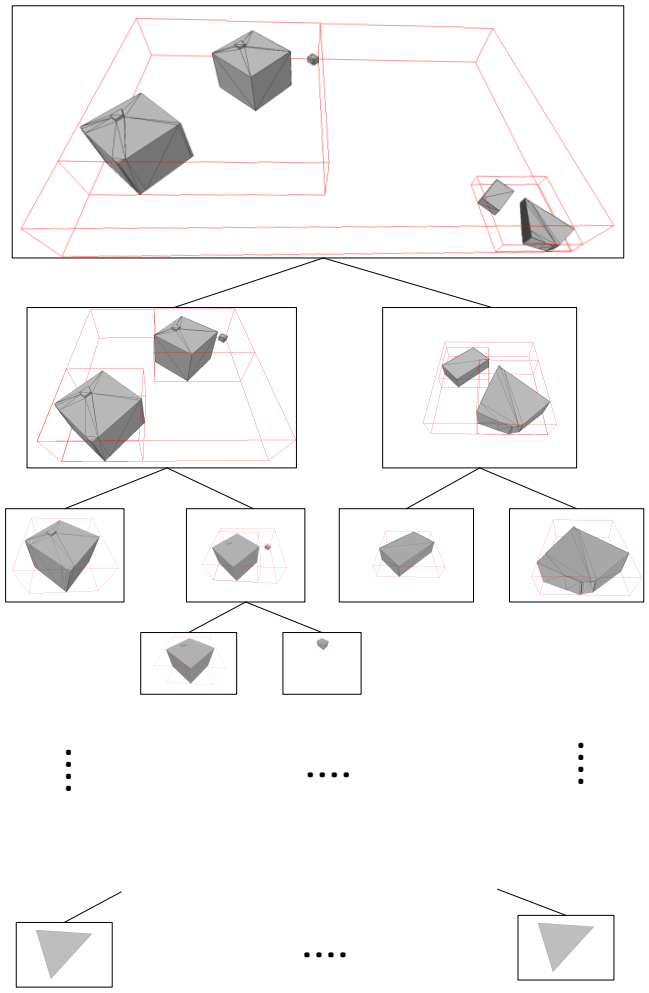


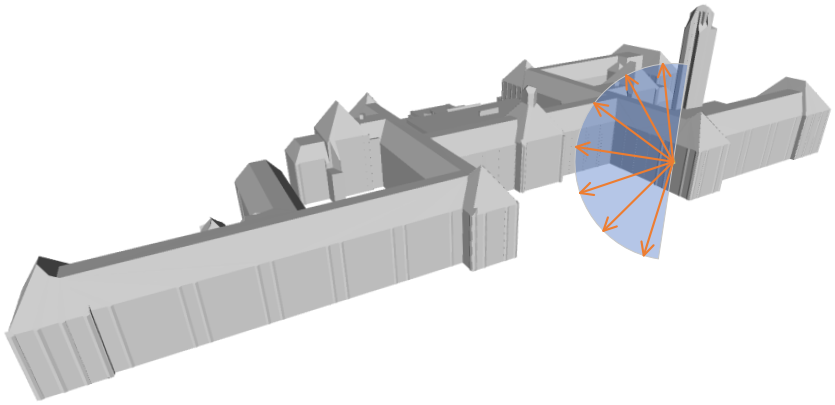
Figure 4.6.: Illustration of the BVH accleration structure

4.3. Viewshed Calculation

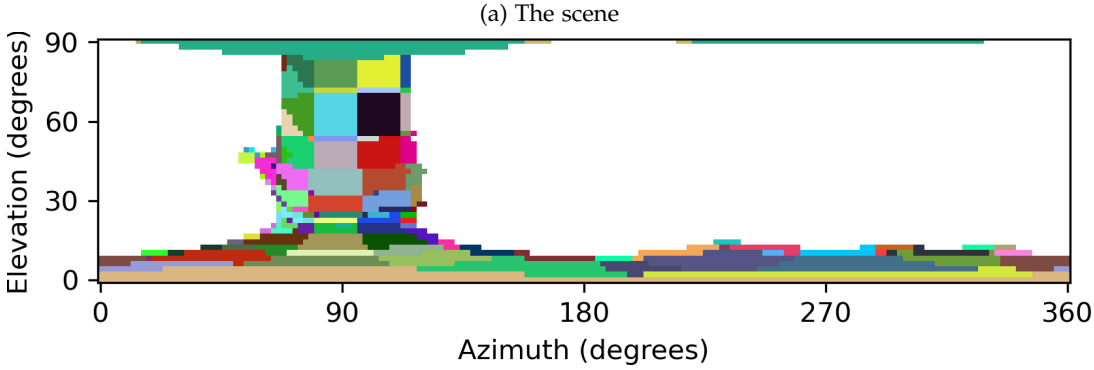
Viewshed calculation plays a crucial role in determining both sky diffuse irradiance and reflective irradiance. In this step, a hemisphere of rays is sampled from each grid point, with the direction of the hemisphere aligned with the orientation of the underlying mesh. Each ray is then traced, and the results are stored in a vector. If a ray hits an object, the closest hit object and its position are recorded. Based on this hit position, the voxel ID that encloses the location is stored. Additionally, the incident angle of the ray relative to the voxel is also saved.

This process is detailed in algorithm 4.2 and figure 4.7a. With the ray tracing result stored as an vector with shape (N, M), where N represent N azimuth steps and M represent M

elevation steps, the SVF can be computed. The resulting vector is also used for reflective irradiance calculations and can be reused for different solar positions.



(a) The scene



(b) The resulting vector

Figure 4.7.: Hemisphere Sampling

4.4. Direct and Sky Diffuse Irradiance calculation

The direct beam component of the solar irradiance are calculated according to equation (2.7). The sky diffuse component of solar irradiance are calculated according to the isotropic model as illustrated in equation (2.8). But instead of using the simplified SVF and GVF calculation method described in equation (2.9) and equation (2.10), we calculate them while tracing the hemiphere sampled rays.

Algorithm 4.2: Hemisphere Viewshed Rendering

Input: Grid point p, azimuth resolution r_a, elevation resolution r_e, 3D City model *Mesh*
Output: Semantic Viewshed map M

number of azimuth steps $n_{azimuth} = 360/r_a$;
number of elevation steps $n_{elevation} = 90/r_e$;
foreach $i, j \in n_{azimuth}, n_{elevation}$ **do**
    $Ray_{dir} = p.normal + i \cdot r_a + j \cdot elevation$;
    hit = Trace(Ray_{dir});
    if *hit* **then**
        find closest hit location X;
        find enclosing voxel *Voxel* of X;
        $M[i][j] = Voxel.id$
    else
        $M[i][j] = UINT32\_MAX$
return M

$$SVF = \frac{1}{\pi} \int_{H^2} f(\omega_i) d\omega_i \approx \frac{1}{\pi} \sum_{i=1}^{N} f(\omega_i) \cos\theta_i \Delta\omega_i \tag{4.1}$$

$$GVF = 1 - SVF \tag{4.2}$$

Where:

- N is the number of samples in the hemisphere
- ω_i is the *ith* sampling direction
- $f(\omega_i)$ is an indicator function, determining whether direction ω_i is visible to the sky. $f(\omega_i) = 1$ if direction ω_i is visible to the sky. $f(\omega_i) = 0$ if direction ω_i is not visible to the sky.
- $\cos\theta_i$ represent the cosine value of the angle between ω_i and the surface normal.
- $\Delta\omega_i$ is the direction weights. In the case, the weight is $2\pi/N$

4.5. Iterative Global Reflective Irradiance Calculation based on Semantic Viewshed and Scene Voxelization

In this step, a method is proposed for simulating the reflective solar irradiance of the surroundings. The proposed approach, Iterative Global Reflective Irradiance Calculation Based on Semantic Viewshed and Scene Voxelization, optimizes irradiance calculations using cached semantic viewsheds and a voxelized irradiance field. The core of this method is to use cached semantic viewshed and voxelized scene irradiance field to eliminate the

need to trace rays that grow exponentially with the number of bounces. In this method, the computation complexity will be linear to the number of bounces rather than exponentially.

The methodology is outlined as follows. To prevent the exponential growth of calculations, this method does not trace the bounces of each ray within the sampling hemisphere. First, for each grid point sampled, the shading effect and SVF are computed to determine the initial direct beam irradiance and sky diffuse irradiance.

Next, the scene is voxelized to aggregate the irradiance values from the grid points into voxels. Reflective solar irradiance is treated as a low-frequency signal Marques and Santos [2010], meaning using voxels to represent the 3D irradiance distribution of the scene is an efficient approach, particularly for reflective irradiance from urban objects.

For each grid point, rays are sampled within a hemisphere to gather surrounding information, storing the voxel identifier of the hit surface for each ray to form a viewshed. The solar irradiance from the identified voxels is then propagated to the grid point from each hemisphere's rays, completing the first bounce of solar rays. In subsequent iterations, representing additional bounces, voxel irradiance values will be first updated (voxelization in the previous step) based on the grid point irradiance, followed by another propagation step.

This iterative process ensures that reflective irradiance is calculated efficiently without the need for complex ray-tracing algorithms, maintaining manageable computational loads even with the number of bounces increases.

4.5.1. Semantic Scene Voxelization

This step involves aggregating the previously calculated direct beam and sky diffuse solar irradiance through voxelization to create a light field within the scene, as illustrated in figure 4.8. Therefore, instead of tracing the bounces of rays, the reflective irradiance from urban objects is treated as irradiance emitted from the voxels.

Here we consider the voxel as an anisotropic emissive object. And the emissive intensity for each direction is dependent on the actual irradiance sources i.e., the grid points within the voxel. The emissivity of the voxel can be descibed in:

$$I_{out,\omega_i} = \frac{\sum max(0, I_p \cdot \cos(p.normal, \omega_i)) \cdot \gamma_{refl}}{\sum \mathbb{I}(\cos(p.normal, \omega_i) > 0)} \tag{4.3}$$

Where:

- I_{out,ω_i} represents the outoging irradiance of the voxel to direction ω_i
- I_p represents the irradiance value of the grid point that is within the voxel
- $p.normal$ is the normal of the grid point
- γ_{refl} is the albedo value of the grid point's underlying surface material
- $\mathbb{I}$ is an indicator function. If $x > 0$, then $\mathbb{I}(x) = 1$, $\mathbb{I}(x) = 0$ otherwise

In equation (4.3), A irradiance source could have infinite number of emissive directions (ω_i), but for computation efficiency and implementation simplicity, here we choose to use only six directions: up, down, left, right, front, back, which is coherent to the voxel representation of the scene (as shown in figure 4.8 and figure 4.9). The method is also described in algorithm 4.3.

To better illustrate the process. The example in figure 4.8 is given here: Consider there are two grid points inside the voxel, and they have normal vector (not normalized) $(1,1,1)$ and $(-1,-1,1)$, and intensity values 50 and 30. Then the intensity values for the voxel faces:

$$\begin{aligned} Face\_up &= (cosine(normalize((1,1,1)),(0,0,1)) \cdot 50 + cosine(normalize((-1,-1,1)),(0,0,1)) \cdot 30)/2 \\ &= (0.577 \cdot 50 + 0.577 \cdot 30)/2 \\ &= 23.094 \end{aligned}$$

$$Face\_down = 0$$

$$\begin{aligned} Face\_left &= cosine(normalize((-1,-1,1)),(-1,0,0)) \cdot 30 \\ &= 0.577 \cdot 30 \\ &= 17.32 \end{aligned}$$

$$\begin{aligned} Face\_right &= cosine(normalize((1,1,1)),(1,0,0)) \cdot 50 \\ &= 0.577 \cdot 50 \\ &= 28.868 \end{aligned}$$

$$\begin{aligned} Face\_front &= cosine(normalize((-1,-1,1)),(0,-1,0)) \cdot 30 \\ &= 0.577 \cdot 30 \\ &= 17.32 \end{aligned}$$

$$\begin{aligned} Face\_back &= cosine(normalize((1,1,1)),(0,1,0)) \cdot 30 \\ &= 0.577 \cdot 50 \\ &= 28.868 \end{aligned}$$

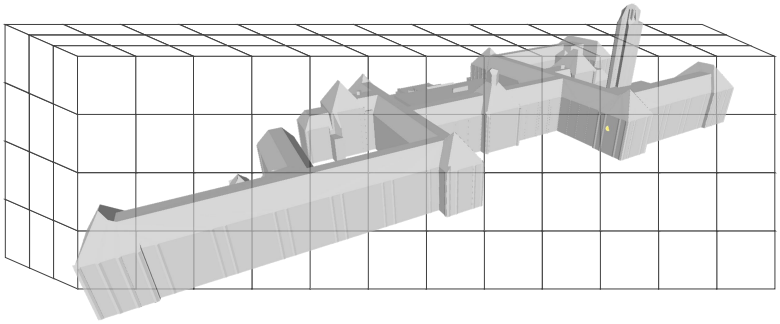


Figure 4.8.: Voxelization of the scene

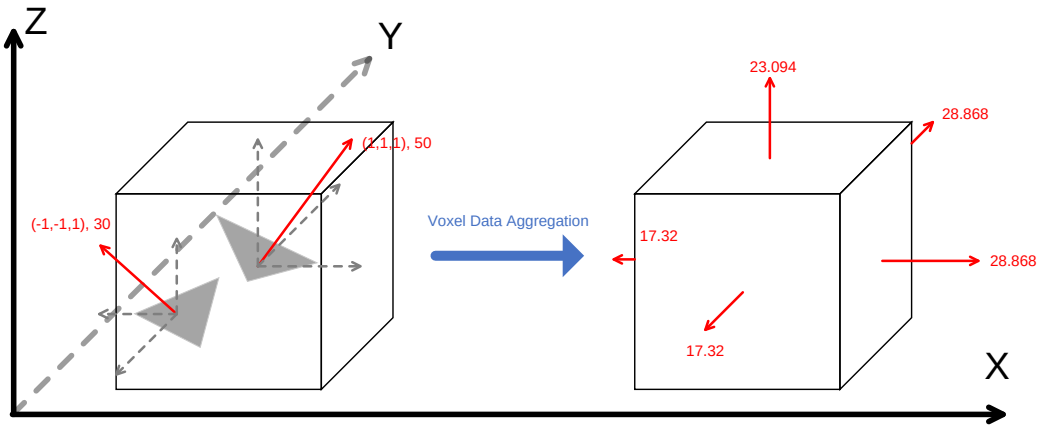


Figure 4.9.: Semantic Voxel

4.5.2. Iterative Irradiance propagation

We can begin simulating reflective solar irradiance with the voxelized irradiance field and the viewshed map for each grid point being constructed. We reference all the ”pixels” in the stored viewshed map for each grid point, where the voxel ID and the incident angle towards the voxel are recorded. Using this information, we identify the three faces of the voxel visible to the pixel.

By calculating the cosine of the incident angle between the ray and the surface normals of the three visible faces, the cosine of the ray direction and the surface normal of the grid point, and multiplying them by the irradiance intensities of these faces, we can determine the

Algorithm 4.3: Voxelization

Input: Grid Points with initial irradiance values $(Grid)$, a predefined space of voxels of the scene (V)
Output: Voxels with aggregated Irradiance (V_I)

foreach $Voxel \in V_I$ **do**
  Find grid points $P_{insideVoxel}$ inside $Voxel$;
  foreach $face \in Voxel.faces$ **do**
    Find grid points $P_{visibleFromFace}$ where $cosine(d_p, d_{face} > 0)$;
    $V_I[face] = \frac{\sum_{p \in P_{visibleFromFace}} (p.albedo \cdot p.irradiance \cdot cosine(\theta_{p,face}))}{|P_{visibleFromFace}|}$;
return V_I

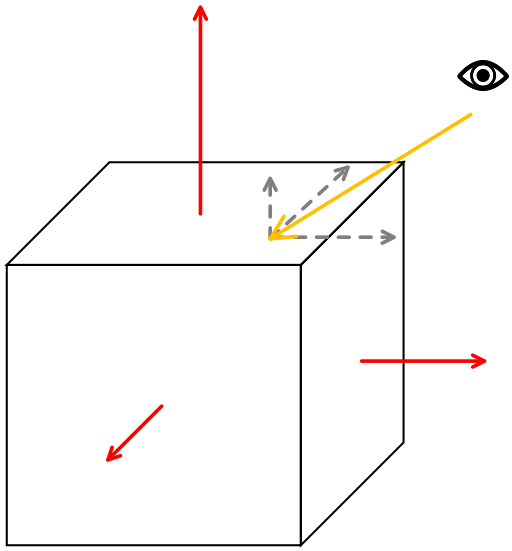
amount of irradiance transferred from the voxel to the grid point (as shown in figure 4.10). The computation follows:

$$I_p = \frac{2\pi \cdot \sum_{pixel}^{n_{pixels}} \sum_{face}^{n_{faces}} max(0, I_{out,face} \cdot \cos(ray_{dir}, face.normal) \cdot \cos(ray_{dir}, p.normal))}{n_{pixels}} \quad (4.4)$$

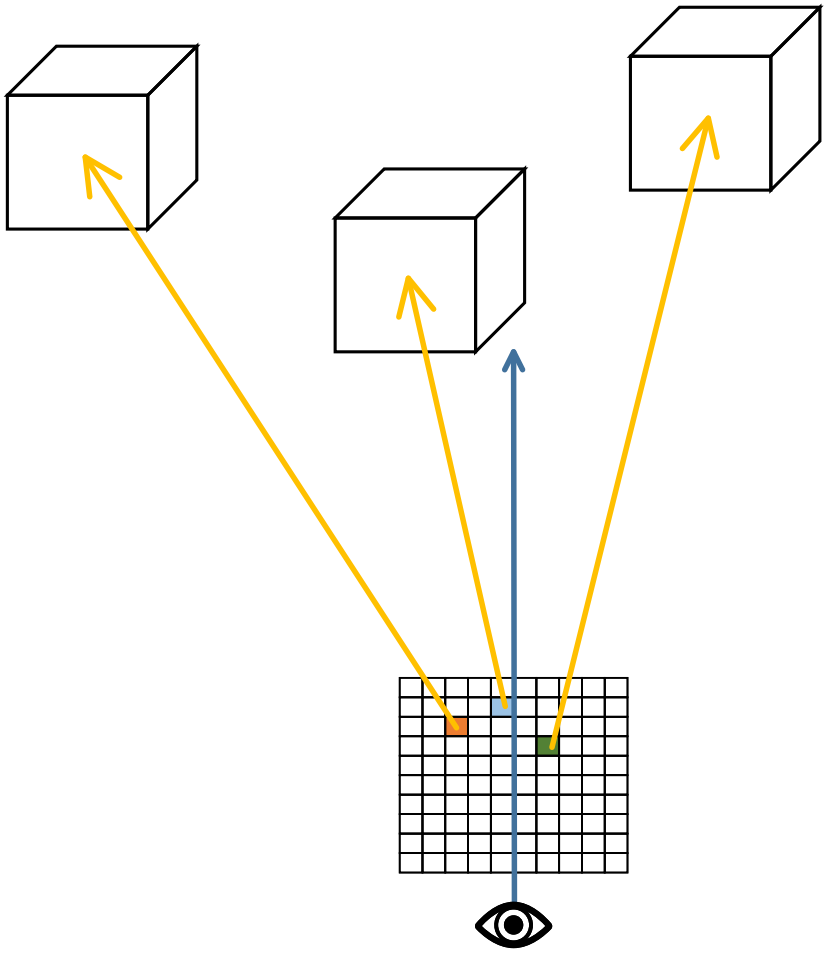
Where

- n_{pixels} represent the number of pixels in the viewshed
- $n_{faces} = 6$
- $I_{out,face}$ represent the outgoing irradiance of the corresponding voxel face for current pixel
- ray_{dir} represent the corresponding ray direction of the pixel in the viewshed (Viewshed is generated by sampling a hemisphere or rays).

The detailed process is described in algorithm 4.4



(a) The ray incident to the voxel



(b) Updating the grid point irradiance by finding visible voxels according to the viewshed

Figure 4.10.: Illustration of the irradiance propagation process

Algorithm 4.4: voxel propagation

Input: Grid point with irradiance value p, index map storing the hemisphere sampling result for the grid point (M), Voxels with aggregated Irradiance (V_I)

Output: Grid point with updated irradiance values $(p_{updated})$

foreach $index \in M$ **do**
  Relevant voxel $Voxel = V_I[index]$;
  foreach $face \in Voxel.faces$ **do**
    if $cosine(\theta_{face,ray} > 0)$ **then**
      $p.irradiance \mathrel{+}= Voxel[face] \cdot cosine(\theta_{face,ray}) * \cdot cosine(p.normal, ray)$
    else
      continue
    ;
$p_{updated} = 2\pi \cdot p/M.length$ **return** $p_{updated}$

The above computation accounts for a single bounce of light. If multiple bounces are required, further iterations of semantic scene voxelization (section 4.5.1), and iterative irradiance propagation (section 4.5.2) can be applied to simulate additional light interactions.

5. Implementation

The method has been implemented using both C++ and Python. This chapter is divided into three sections that cover the following components: section 5.1 Input and Data Processing, section 5.2 Shadow Calculation and Viewshed Calculation with Nvidia Optix API Nvidia [2024], and section 5.3 Semantic Scene Voxelization and Iterative radiance Propagation. C++ and Nvidia Optix are primarily used for the first part, while Python is employed for the subsequent sections.

The schematic representation of the method's implementation is depicted in Figure 5.1. This figure delineates the physical configuration of the deployed functions, elucidating the inter-device data interactions and the flow dynamics. It can be noted that data is either directly accessed from the memory or indirectly, via memory mapping from the hard drive. The computational tasks and resource loading are strategically distributed across the Central Processing Unit (CPU) and Graphics Processing Unit (GPU), respectively. This judicious distribution, which leverages the computational prowess of both the CPU and GPU, has facilitated a harmonious balance among implementation simplicity, operational efficiency, and code legibility.

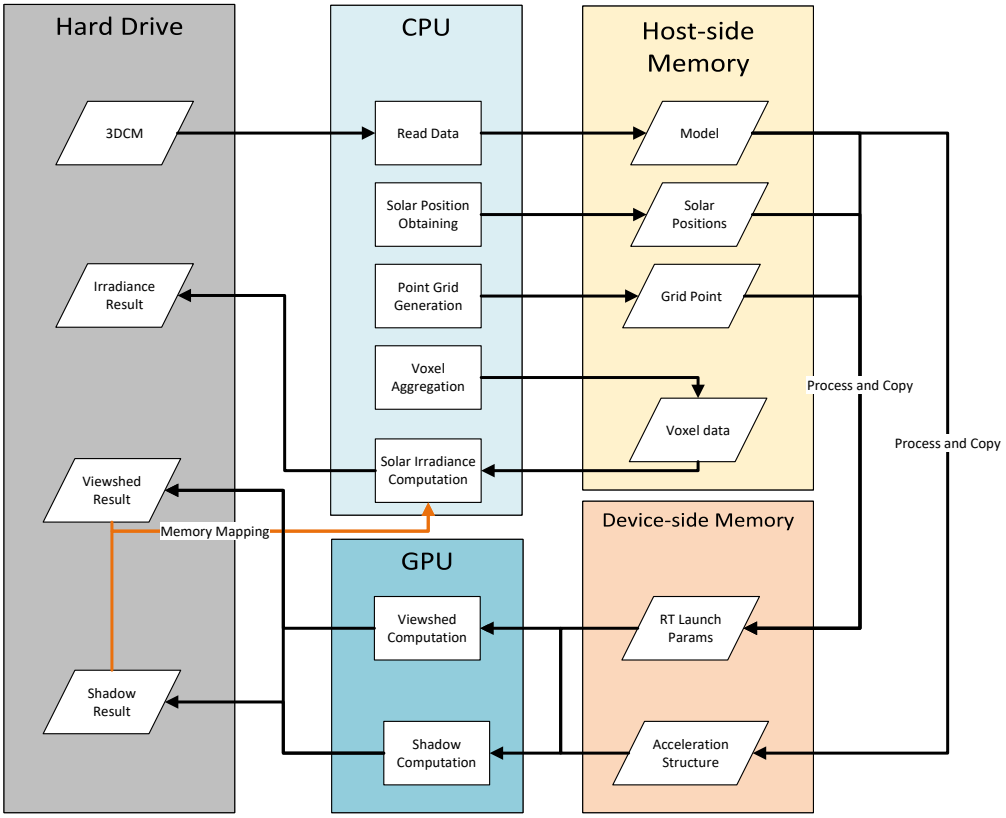


Figure 5.1.: The implementation overview

5.1. Input and Data Processing

The input to the program is a triangulated 3D City Model that adheres to the OGC CityGML standards. Since the triangle mesh representation of the geometry is optimal for ray tracing, the input 3DCM must be triangulated. This triangulation is performed using FME.

The simulation considers three types of city objects: Buildings, Terrain, and Vegetation. The geometry for buildings and terrain is stored explicitly, while vegetation is represented as implicit geometry. Each vegetation object is instantiated according to its geometry template and transformation matrix.

To maintain a link between the sampled grid points and the input 3DCM, each triangle surface in the scene is assigned a unique identifier, which is inherited by the sampled grid points from that surface.

5.2. Shadow Calculation and Viewshed Calculation with Nvidia Optix Ray Tracing Engine

The Nvidia Optix Ray Tracing Engine leverages RT cores on Nvidia GPUs, alongside Compute Unified Device Architecture (CUDA) programming, to enable hardware-accelerated RT. Although ray tracing is typically associated with rendering, it is well-suited for simulating light behaviour for radiance calculations in solar iradiance simulations. Nvidia Optix offers several key advantages:

GPU Optimization The Optix Ray Tracing Engine is optimized for GPU usage, utilizing dedicated RT cores to achieve hardware-level acceleration.

Programmability The ray tracing pipeline is customizable, allowing flexible program design. The API offers a straightforward programming model using C++.

Built-in Geometry Acceleration Structures Optix provides built-in acceleration structures like BVH to optimize ray-object intersection searches within the scene's geometric data.

The general structure of Optix Ray Tracing programs are illustrated in figure 5.2. The program will start with Ray Generation. Next, for generated rays, the scene objects will be traversed for ray-object intersection test to be applied. And based on the intersection result, there will be user defined Miss programs, Closest Hit programs, and Any Hit programs.

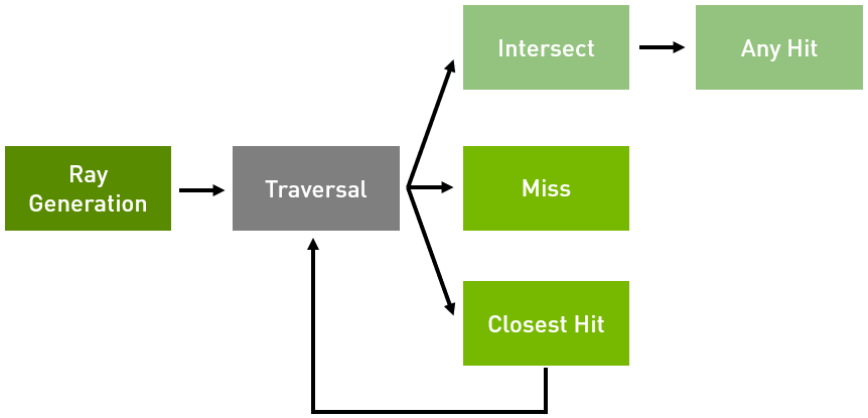


Figure 5.2.: The optix program structure. Figure retrieved from Parker et al. [2018]

For shadow calculation, rays are generated at each time step, with the sampled points on the triangle meshes serving as the ray origins. The rays are directed towards the sun for each time step. In this specific application, we are only interested in a binary result—whether the ray hits an object or not—so the program uses Miss and Any Hit programs.

For semantic viewshed calculation, a hemisphere of rays is generated from each sample point for viewshed rendering. In this application, both the Miss and Closest Hit programs are implemented as described in algorithm 4.2.

The semantic viewshed for each grid point is stored on the hard drive, allowing the reuse of ray tracing results for different solar positions. The data is stored as a flattened multi-dimensional array, where the dimensions represent grid points, azimuth, elevation, and voxel face indices. To avoid redundant computations, the cosine of the angle between the ray and the voxel's faces is $cosine(\theta_{face,ray})$ precomputed. The structure of the array is as follows:

$$Array[Grid\_Point\_Index][Azimuth\_Index][Elevation\_Index][Face\_Index] = cosine(\theta_{face,ray})$$

The flattened array is indexed as:

$$Index\_1d = Face\_Index + 6 \cdot (Elevation\_Index + n\_elev \cdot (Azimuth\_Index + n\_azi \cdot Grid\_Point\_Index))$$

$$Array[Index\_1d] = cosine(\theta_{face,ray})$$

The resulting file can be huge (approximately 20-200 GB depending on the resolution and the scale of the study area) in binary format, making loading entirely into memory on most computers impractical. Although compression can reduce file size, it introduces extra overhead for decompression. Instead, memory-mapped files are used to read the data in a streaming manner, which minimizes memory usage while maintaining Input/Output efficiency.

5.3. Irradiance calculation

For implementation simplicity, this step is mainly accomplished with Python programming language, with open-source libraries such as Numpy, Pandas.

First, the grid point and shadowing information is processed to calculate the initial direct solar and sky diffuse irradiance. The scene is then voxelized based on the grid points with irradiance values. Following this, iterative irradiance propagation, as described in section 4.5.2, is applied using Numpy functions. Numpy's array operations and broadcasting features enable efficient parallel computing, which improved both the readability and performance of the code.

6. Experiment and Results

Experiments were conducted to validate the accuracy of the developed methods and assess their scalability for city-scale simulations. All experiments were performed on a laptop equipped with a 12th Gen Intel(R) Core(TM) i7-12700H CPU 2.70 GHz, 32GB of RAM, and an NVIDIA GeForce RTX 3070 Laptop GPU.

6.1. Validation

To substantiate the effectiveness of the proposed method for solar irradiance computation, an ideal validation strategy involves a direct comparison of the estimated solar irradiance values with those obtained from ground-based measurements. While validation against ground measurements is optimal, it necessitates the availability of both a comprehensive 3D model of the experimental setting and corresponding sensor recordings captured by pyranometers. The majority of existing open-source datasets for solar irradiance simulation primarily provide tabular data for sensor readings, which do not encompass the detailed spatial information required for thorough validation.

An alternative validation approach involves correlating the estimated results with outputs from other established simulation software, as exemplified in Xu et al. [2024]; Giannelli [2021]. In scenarios where such measurements are lacking, the existing solar irradiance computation methods can provide a reference. However, in the case when their model fails to predict the solar irradiance in the specific settings of our experiment, this evaluation can be erroneous.

In this thesis, the primary validation method is selected due to the availability of two datasets that include ground truth measurements, namely the Koninklijk Nederlands Meteorologisch Instituut (KNMI) Weather Station dataset and the TUDelft dataset. These datasets, which feature both 3DCM and pyranometer recordings, are instrumental in ensuring the robustness of the validation process. The datasets are further elaborated upon in the subsequent sections.

6.1.1. Heino KNMI Weather Station Dataset

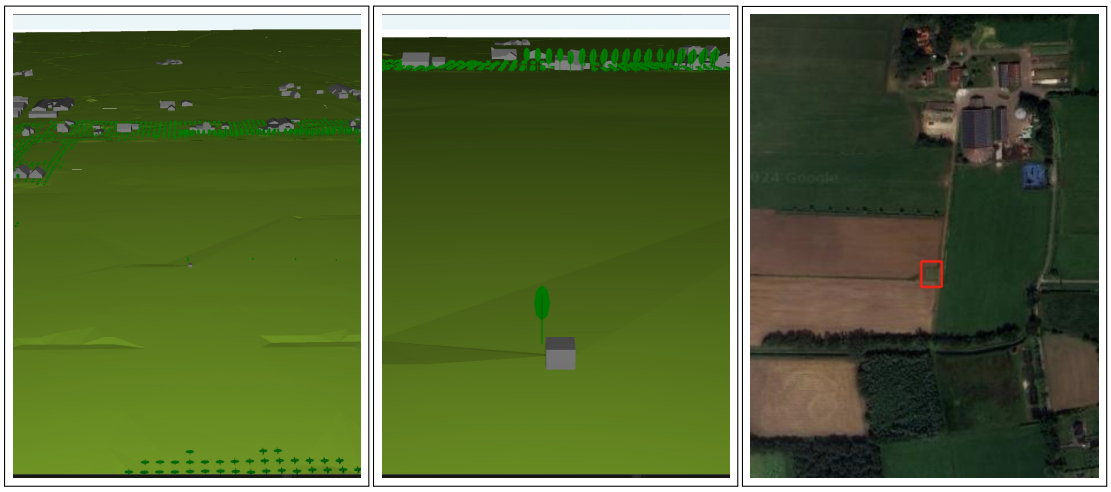
The KNMI is the Royal Dutch Meteorological Institute, with nationwide weather stations. KNMI [2024]. It provides historical weather recordings from 1951 to now at hourly intervals. The dataset contains recordings of the solar irradiation, including GHI, DNI, DHI. This climate data is available in text files, and it does not include a 3DCM of the weather station and its surroundings. Therefore, the 3D Geoinformation Group prepared a CityGML v2.0-compliant dataset.

Dataset Overview

The details of the dataset are presented in table 6.1. The study area is in Heino. The geographic coordinates of the weather station are 52.43°and 6.26°. Weather data are available as epw files, which provide typical year values for solar irradiance simulation. In addition, there are also historical values available for download from 1991 up to now. The screenshot of the study area is presented in figure 6.1. It can be observed that the area that surrounds the weather station is open space. Moreover, the only potential occluding object is a tree. It is expected that the reflective solar irradiance will not contribute a significant portion to the total solar irradiance. However, this dataset is still valuable for validating the direct and diffuse solar irradiance computation.

Attribute	**Value**
Geographic center	52.43, 6.26
Spatial extent	2665 * 2665 * 17
#Buildings/#Faces	272/29409
#Trees/#Faces	3950/94800
#Terrain/#Faces	121/86020
#Total faces	210229
#Sample points	1539022

Table 6.1.: KNMI dataset details



(a) Overview of Heino dataset

(b) Zoomed in view of Heino dataset

(c) Satellite image of Heino. The weather station is labelled with a red square

Figure 6.1.: Heino Weather Station

Experiment Settings

A total of four days were simulated and compared for the Heino weather station dataset: March 21st, June 21st, September 22nd, and December 22nd. Simulations were conducted at hourly intervals, which is the highest time resolution available from the EPW file. For each simulation, scenarios with 0, 1, and 2 light bounces were considered. Additionally, two simplified scenarios were used as baselines for comparison.

In Setting 1, both the sky diffuse and ground reflected components of solar irradiance were calculated without accounting for the occluding effects of the surroundings, as described in equation (2.9) and equation (2.10). In Setting 2, the ground reflected component was calculated without tracing ray-object interactions in the scene, using a global albedo value as shown in equation (2.11). However, in this scenario, the SVF and GVF are still calculated with consideration of the actual surrounding environment, as indicated in equation (4.1) and equation (4.2).

For surface albedo, referring to Google Maps satellite images and the ECOSTRESS Spectral Library, the following albedo values were used for each type of surface:

Surface Type	Materials	Albedo
WallSurface	White paint; Grey Paint; Reddish Brown Paint; Brick	0.4
RoofSurface	Solar Panels; Red Tiles	0.1
Terrain	Asphalt; Bare Soil; Grass; Water	0.2
Tree	Tree Crowns	0.3

Table 6.2.: Surface types, materials, and albedo values for Heino dataset

6.1.2. TU Delft Dataset

Dataset Overview

The second dataset was created using data collected by the Photovoltaic Materials and Devices (PVMD) group at Delft University of Technology Group [2024]. In their previous work, the group collected solar irradiance measurements on top of a building on the TU Delft campus, along with a 3D model of the surrounding environment Andres et al. [2023]; Calcabrini [2023]. A total of 82805 records, at minute intervals, from 2020 August 19 to 2021 February 4, were collected. The 3D city model created from their raw data is illustrated in Figure 6.2.

Four sensors were deployed: A Kipp & Zonen SOLYS2 sun tracker equipped with a SMP 21 pyranometer and a SHP1 pyrheliometer was used to measure the DHI and the DNI, respectively. A monocrystalline ISET sensor (IKS Photovoltaik) facing south and tilted 30 degrees (S1); a SMP10 thermopile pyranometer (Kipp & Zonen) facing 65 degrees east of north and tilted 90 degrees (S2); and a MS-700 spectroradiometer (EKO Instruments) mounted horizontally (S3). Since the thesis does not require spectral data, the measurements from sensor S3 are not used.

Attribute	Value
Geographic center	52.6, 4.23
Spatial extent	1174 * 1178 * 99
#Buildings/#Faces	1114/99738
#Trees/#Faces	3/153
#Terrain/#Faces	20/964403
#Total faces	1064294
#Sample points	889500

Table 6.3.: TUD dataset details

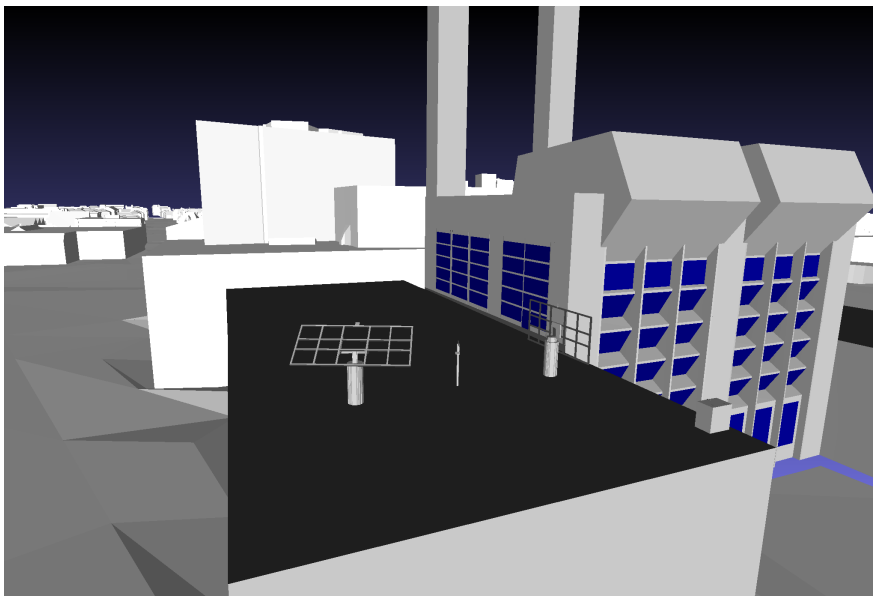


Figure 6.2.: screenshot of the TU Delft weather station dataset generated from PVMD group data Andres et al. [2023]

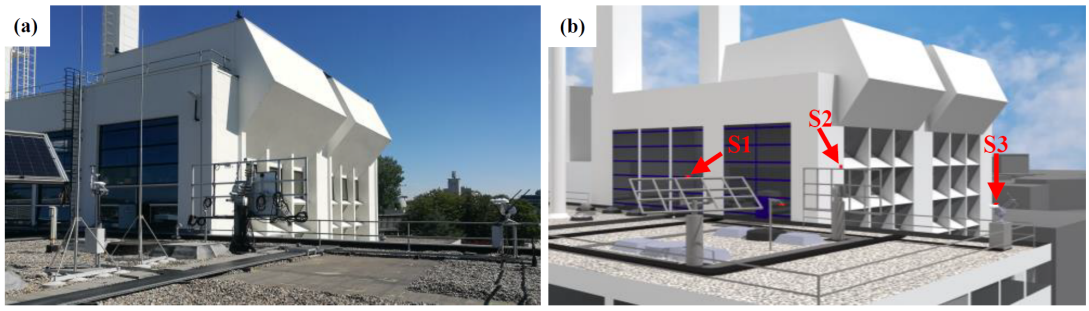


Figure 6.3.: TU Delft weather station sensors, screenshot PVMD

Dataset Preparation

The 3D model provided by the PVMD group was in 3dm format, and it included the geometries of the target building, sensors, surrounding buildings, and trees. Upon manual inspection, the target building model was found to be at LoD3, while the surrounding buildings were mostly at LoD1. Additionally, the dataset included only a plain terrain representation of the area.

To prepare the dataset for use, the 3dm file is first converted to wavefront OBJ format, which is more convenient for reading and editing without relying on closed-source software. Next, the geometry of the sensors and the LoD3 building adjacent to them was extracted. This also included an adjacent building that was not present in the 3D BAG dataset to ensure that the 3D city model aligned with the scene as it existed during data collection. The extracted geometries were then manually reoriented to correct any reversed orientations in the triangle meshes that could affect the computation.

Afterwards, any overlapping buildings from the 3D BAG dataset were removed. The final step involved merging the extracted triangle mesh, the 3D BAG dataset, and the terrain to create the complete 3DCM of the scene. This process resulted in a model containing one LoD3 building adjacent to the sensors and other surrounding LoD2 buildings, trees, and terrain.

The screenshot of the PVMD group data, the 3D BAG data, and the final merged data are illustrated in Figure 6.13b and 6.14. From Figure 6.13b, it can be observed that the 3D model of the PVMD group data features a high level of detail (LoD3) near the sensors, with the detail decreasing to LoD1 for the other buildings. The PVMD dataset does not include realistic terrain data. In addition, there are no buildings on the south side of the sensors included. These characteristics impede the consideration of global reflective irradiance in the scene.

Experiment Settings

A total of seven days are spent simulating and comparing the TUDelft dataset. They are 2020 Aug 20, 2020 Sept 1st, 2020 Oct 1st, 2020 Nov 1st, 2020 Dec 1st, 2021 Jan 1st, and 2021 Feb 1st. Simulations are completed at 10-minute time intervals. For each simulation, we consider the situation with 0, 1, and 2 times the bounces of the light—the two additional simplified baseline settings used in section 6.1.1. The setting 1 scenario refers to calculating both the sky diffuse component and ground reflected component of solar irradiance without considering the occluding effect of surroundings, as indicated in equation (2.9) and equation (2.10). The setting 2 scenario refers to calculating the ground reflected component without tracing the ray object interaction in the scene but simply using a global albedo value as indicated in equation (2.11). In this scenario, the SVF and GVF are still calculated with considerations of the actual surrounding environment, as indicated in equation (4.1) and equation (4.2).

For surface albedo, referring to the albedo values presented in PVMD, google map satellite images, and the ECOSTRESS Spectral Library, the following albedos were used for each kind of surface:

Surface Type	Materials	Albedo
WallSurface	White paint; Grey Paint; Reddish Brown Paint; Brick	0.4
RoofSurface	Pebbles; Ethylene Propylene Diene Monomer; Glass	0.1
Terrain	Asphalt; Brick; Bare Soil; Grass; Water	0.05
Tree	Tree Crowns	0.3
Window	Glass	0.3

Table 6.4.: Surface types, materials, and albedo values for TU Delft dataset

Figure 6.4.: TU Delft Google map

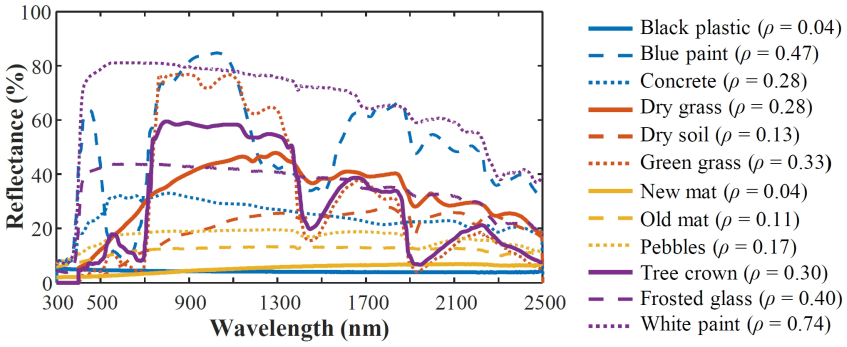


Figure 6.5.: TU Delft dataset material from PVMD

6.1.3. Result

In previous studies on solar irradiance simulation, there has often been a lack of comparison between estimated results and ground-truth measurements. While some research has explored solar irradiance modelling in 3D environments, few have tested these models in

complex urban settings. In this thesis, the solar irradiance simulation model was evaluated in two scenarios representing varying levels of complexity: one using a weather station dataset for a simple, unoccluded scene and the other using the TU Delft dataset to reflect more intricate urban conditions.

The method proposed in this thesis improves upon previous approaches that simplify the estimation of sky diffuse and ground-reflected components. This model takes into account the detailed 3D geometry and material properties of the urban environment. Moreover, it is optimized for city-scale simulations, avoiding the exponential growth in computational requirements typical of earlier methods.

Viewshed generation

The viewshed results for the Heino weather station dataset, as well as for sensors S1 and S2 from the TU Delft dataset, are shown in figure 6.6, figure 6.7c, and figure 6.7d.

The tree adjacent to the Heino weather station can be clearly identified in the viewshed from figure 6.6. However, the viewsheds for sensor S1 and sensor S2 are more challenging to interpret for two reasons. First, the viewing direction is not horizontal, making the resulting viewshed less intuitive to analyze. Second, the colours in the viewsheds are randomly assigned based on the voxel ID, which complicates mapping the pixels to the actual objects in the corresponding directions. Despite these challenges, the surroundings of the Heino weather station and sensor S1 remain recognizable from their respective viewshed results.

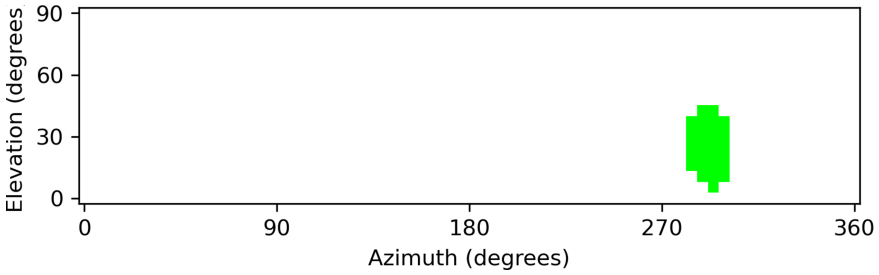
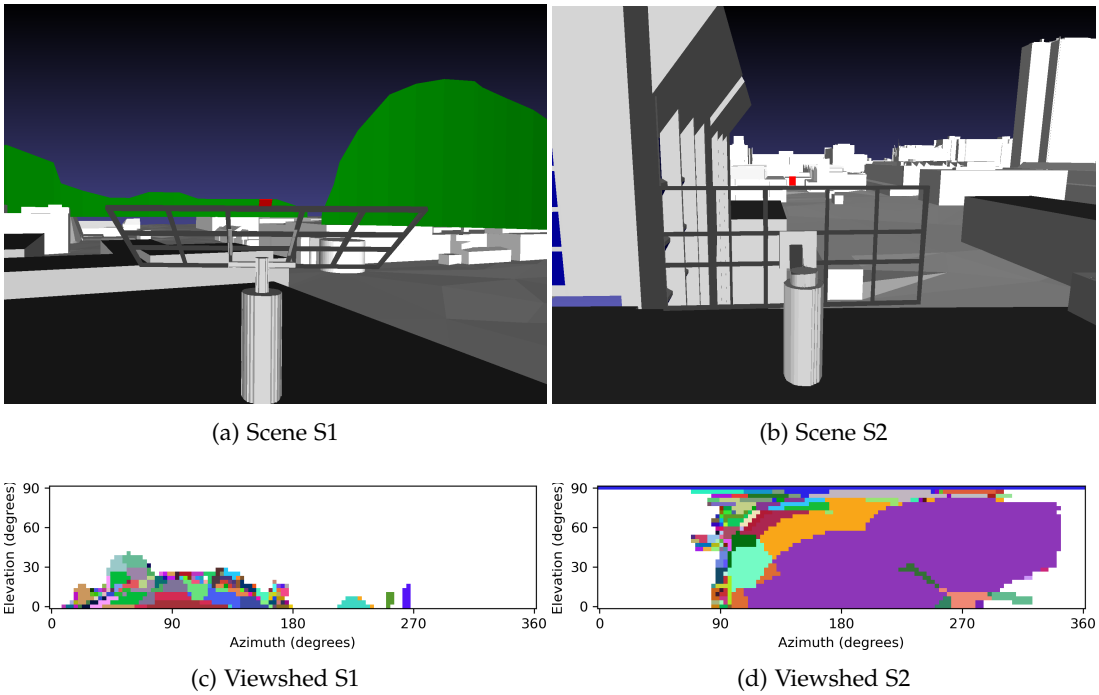


Figure 6.6.: Viewshed heino



(a) Scene S1 (b) Scene S2

(c) Viewshed S1 (d) Viewshed S2

Figure 6.7.: Scene and corresponding viewshed

Solar Irradiance Prediction

As shown in figure 6.9, figure 6.10, figure 6.11, figure 6.12, the results for the weather station dataset closely align with ground-truth measurements. As expected from the irradiance propagation mechanism, increasing the number of bounces in the simulation leads to higher estimated solar irradiance values. In this particular scenario, the reflective component contributed minimally to the overall solar irradiance, as there was little difference between the zero-bounce and one-bounce estimates.

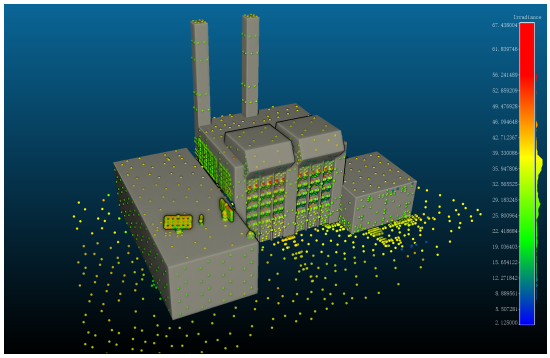
Figures figure 6.15 to figure 6.21 present the results for the TU Delft dataset across various settings and dates. For sensor S1, the estimated solar irradiance matches the ground measurements well. However, for sensor S2, the degree of alignment varied across different dates. For instance, on August 21, 2020, which coincides with data from the PVMD research, our results (especially for bounce 1) follow a similar trend to the PVMD estimates. The bounce 1 setting shows the closest alignment with ground truth, particularly between 9:30 AM and 1:00 PM, when solar irradiance was at its peak. After 1:00 PM, however, our method overestimated the irradiance values.

A similar pattern emerged on September 1, 2020, though the discrepancies were more minor in the afternoon when irradiance levels were lower. On October 1, 2020, when irradiance was consistently below 100, the misalignment was notable, especially with the global reflective setting. In contrast, the "no global reflective" setting performed better. This trend was also observed on February 1, 2021. On November 1, December 1, 2020, and January 1, 2021, the method with one or two bounces showed the tiniest discrepancies. Notably, on December 1,

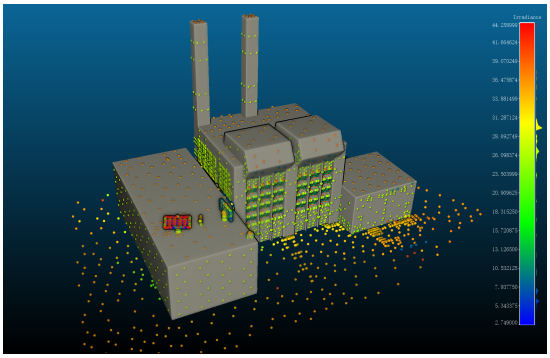
2020, our bounce 1 method closely followed the rapid changes in solar irradiance after 11:00 AM.

Additional visualizations of the results are presented in figure 6.8. In these figures, solar irradiance values are represented by the colour of each grid point. It is evident that for the building where the sensors are located, surface inclination and orientation are the dominant factors influencing solar irradiance. Vertical surfaces receive lower irradiance due to self-shading effects. Shading is also prominent for grid points on the ground and those near adjacent buildings.

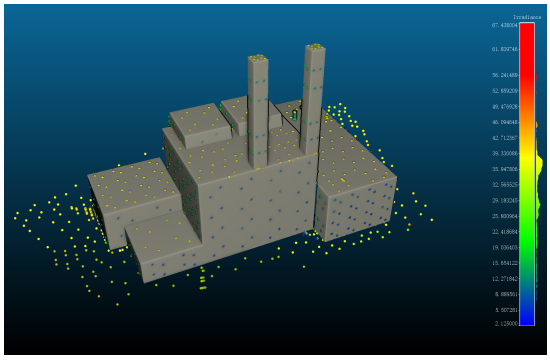
In figure 6.8e and figure 6.8f, we can observe that sample points on the window ledges across different floors show more significant variation in irradiance values when using our method, which models reflective solar irradiance more detailedly. In contrast, the simplified method for estimating reflective irradiance results in much more minor variations.



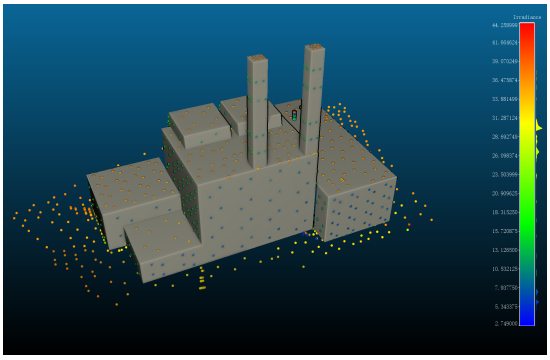
(a) Predicted daily solar irradiance. Looking at north direction



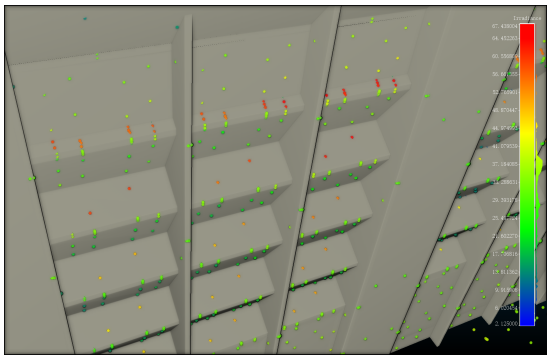
(b) Predicted daily solar irradiance with simplified reflective solar irradiance. Looking in the north direction



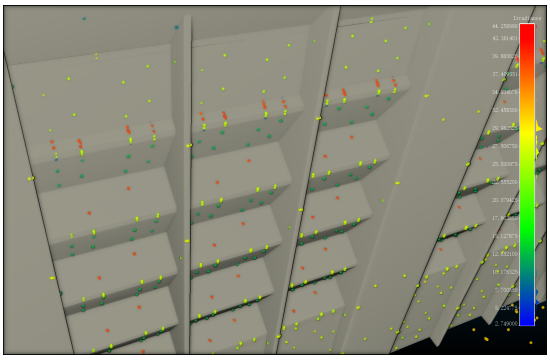
(c) Predicted daily solar irradiance. Looking at south direction



(d) Predicted daily solar irradiance with simplified reflective solar irradiance. Looking in the south direction



(e) Daily solar irradiance of points on the window structure



(f) Daily solar irradiance of points on the window structure with simplified reflective solar irradiance

Figure 6.8.: Visualization of the irradiance calculation result of TU Delft dataset. Value unit is KWh/m^2

Quantitative Evaluation

table 6.5 provides more quantitative evaluation than the line plots discussed in the above section. The table presented four evaluation metrics of the estimated solar irradiance. They are Pearson Correlation Coefficient (PCC), Normalized Mean Bias Error (nMBE), Normalized Mean Absolute Error (nMAE), Normalized Root Mean Square Error (nRMSE).

PCC is calculated as:

$$PCC = \frac{\sum(P_i - \bar{P})(O_i - \bar{O})}{\sqrt{\sum(P_i - \bar{P})^2 \sum(O_i - \bar{O})^2}} \tag{6.1}$$

Where:

- P_i is the predicted solar irradiance value at time step i,
- O_i is the observed or actual solar irradiance value at time step i,
- $\bar{P}$ is the mean of predicted values, and
- $\bar{O}$ is the mean of observed values.

PCC measures the strength of the linear relationship between predicted and observed values. A PCC value of positive and negative values indicate positive and negative linear correlation, respectively, and 0 indicates no linear relationship. In the context of solar irradiance predictions, PCC evaluates how well the predicted values follow the trend of actual irradiance values over time, making it an essential metric for evaluating time-series predictions.

The nMBE is computed using the following equation:

$$MBE = \frac{\sum(P_i - O_i)}{\sum O_i} \tag{6.2}$$

Where:

- P_i is the predicted value,
- O_i is the observed value.

nMBE quantifies the average deviation of the predicted values from the observed values. A positive nMBE indicates that the model tends to overestimate, whereas a negative nMBE indicates underestimation. This metric is handy in solar irradiance forecasting as it helps in identifying systematic bias in predictions, which is critical when forecasting energy yields for solar power systems.

nMAE is calculated as:

$$nMAE = \frac{\sum|P_i - O_i|}{\sum O_i} \tag{6.3}$$

Where:

- $|P_i - O_i|$ is the absolute difference between the predicted and observed values.

nMAE measures the average magnitude of errors between the predicted and observed values without considering their direction. It provides an intuitive measure of the accuracy of the predictions by quantifying how far, on average, the predictions deviate from the actual values. This metric is robust to outliers and is particularly useful for understanding the overall error distribution in solar irradiance forecasts.

nRMSE is given by the following equation:

$$nRMSE = \frac{\sqrt{\frac{1}{n}\sum(P_i - O_i)^2}}{\bar{O}} \tag{6.4}$$

Where:

- n is the number of time steps,
- P_i is the predicted value,
- O_i is the observed value,
- $\bar{O}$ is the mean observed value.

nRMSE represents the square root of the average of the squared differences between the predicted and observed values, normalized by the mean observed value. nRMSE is particularly sensitive to significant errors due to the squaring of differences, making it more responsive to outliers than nMAE. This metric is commonly used to assess the accuracy of solar irradiance models, as it provides insight into how well the model captures both the magnitude and variability of the irradiance data.

These metrics collectively provide a robust framework for evaluating solar irradiance prediction models, capturing aspects such as correlation, bias, error magnitude, and variability. This multidimensional evaluation is essential for accurately assessing the performance of models used in solar energy forecasting and ensuring that energy generation predictions are reliable and efficient.

The evaluation metrics for sensor S1 and sensor S2 from the TU Delft dataset are shown in table 6.5 and table 6.6. Notably, the correlation values, represented by PCC, are consistently high across all prediction settings and time steps, with most exceeding 0.95. For sensor S1, as seen in table 6.5, settings b=2 and setting 1 show a clear advantage in terms of nMBE, while no clear winner emerges for nMAE and nRMSE. For sensor S2, table 6.6 indicates that, on average, settings b=1 and setting 1 outperform other methods. Meanwhile, the results suggest that setting b=2 tends to overestimate values, and setting 1, which oversimplifies the 3D urban environment, does not deliver results as realistic as the proposed method. Furthermore, table 6.6 highlights how estimation accuracy is influenced by weather conditions, as setting 1's approach to reflective solar irradiance modelling fails to produce accurate results on overcast days.

The computation times for solar irradiance estimation in the study areas are presented in table 6.7 and table 6.8. Depending on the number of simulation time steps, the total computation time, which includes two iterations of global reflective solar irradiance, ranges from

one to four hours. This applies to a hemisphere sampling resolution of 5 degrees and approximately one million sample points. It is important to note that the computation was conducted for all sample points in the scene.

As explained in the methodology section, the computation time should scale linearly with both the number of time steps and the number of sample points, and the results have confirmed this. Additionally, around 99% of the total computation time is spent on reflective solar irradiance calculations. The time required for shadow computation and the subsequent direct solar irradiance estimation is negligible, taking less than one second. Even if the number of time steps were increased by a factor of 100, the overhead for direct beam component computation would remain minimal.

It is worth noting that the majority of the time required for diffuse solar irradiance simulation is spent on viewshed rendering, which is necessary for accurately modelling the occlusion effects of surrounding objects (obtaining SVF in this case). Since the viewshed rendering is decoupled from the actual solar irradiance computation, the resulting SVF can be reused multiple times after it has been computed once. The fact that the combined computation time for direct and diffuse components takes no more than two seconds for the tested scenarios further supports this observation.

Overall, the results indicate that our method accurately predicts solar irradiance in complex urban environments, particularly under conditions of high irradiance. In overcast conditions, with lower irradiance, our method did not show a clear advantage over other settings. Nonetheless, the data suggests that the global solar irradiance computation model holds promise for more precise predictions in urban environments, especially on days with high irradiance, where the alignment between the model and ground measurements is significantly better.

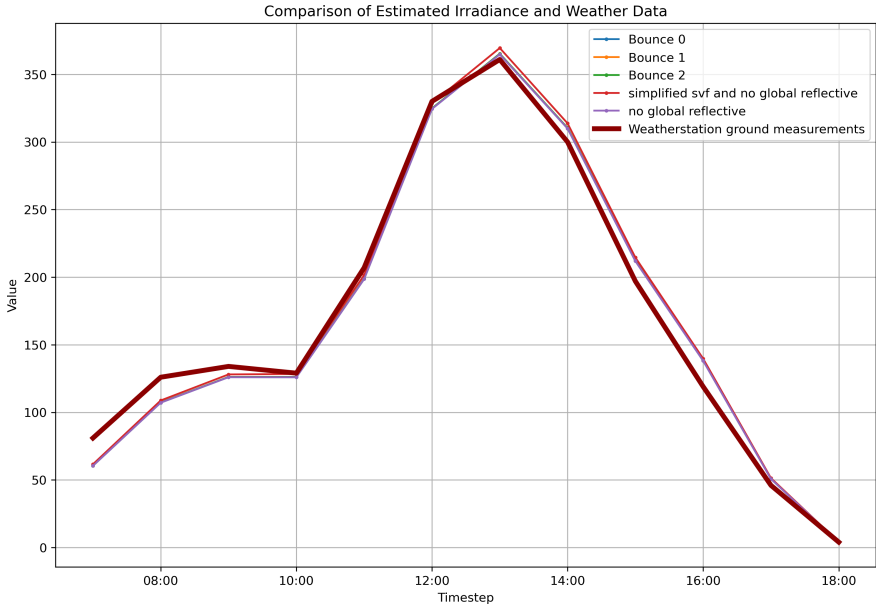


Figure 6.9.: Result of March 21. Heino dataset. Value unit is W/m^2

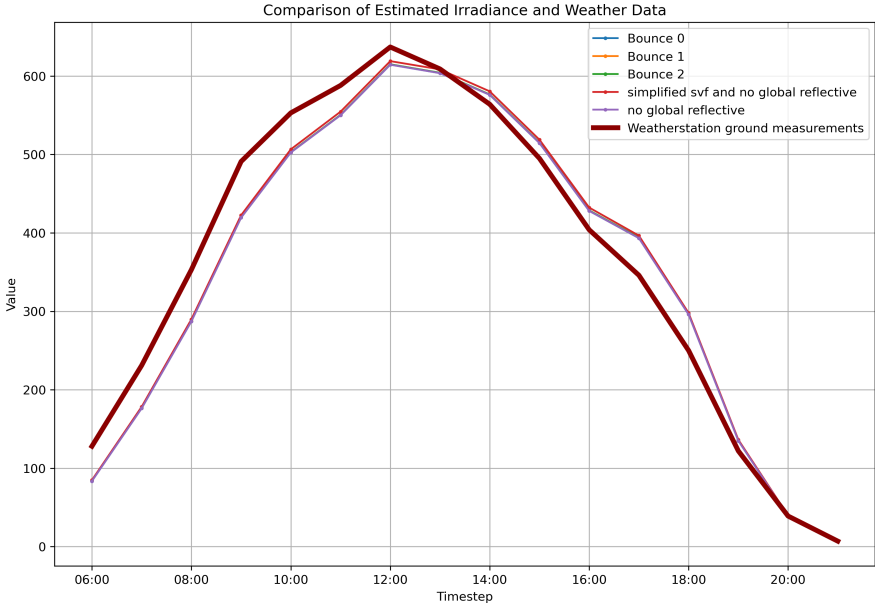


Figure 6.10.: Result of June 21. Heino dataset. Value unit is W/m^2

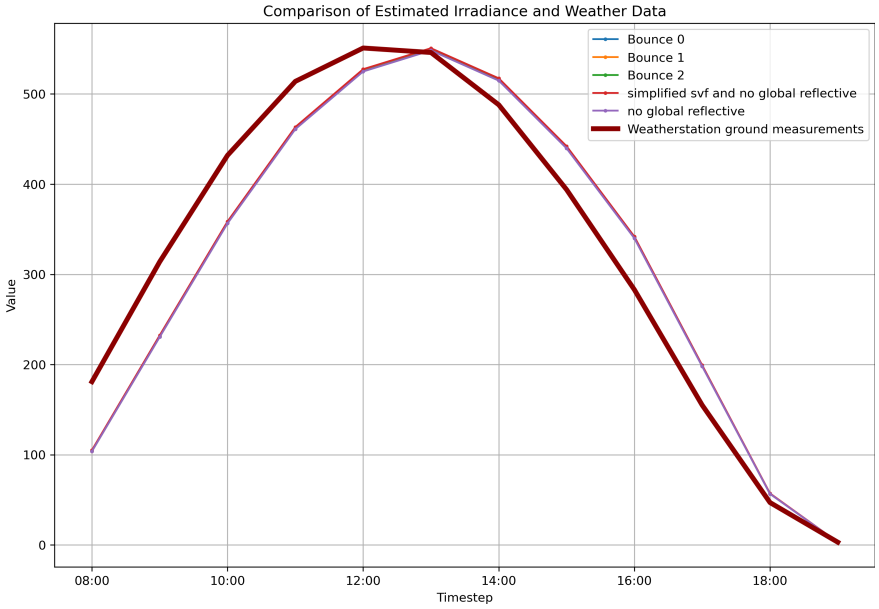


Figure 6.11.: Result of September 22. Heino dataset. Value unit is W/m^2

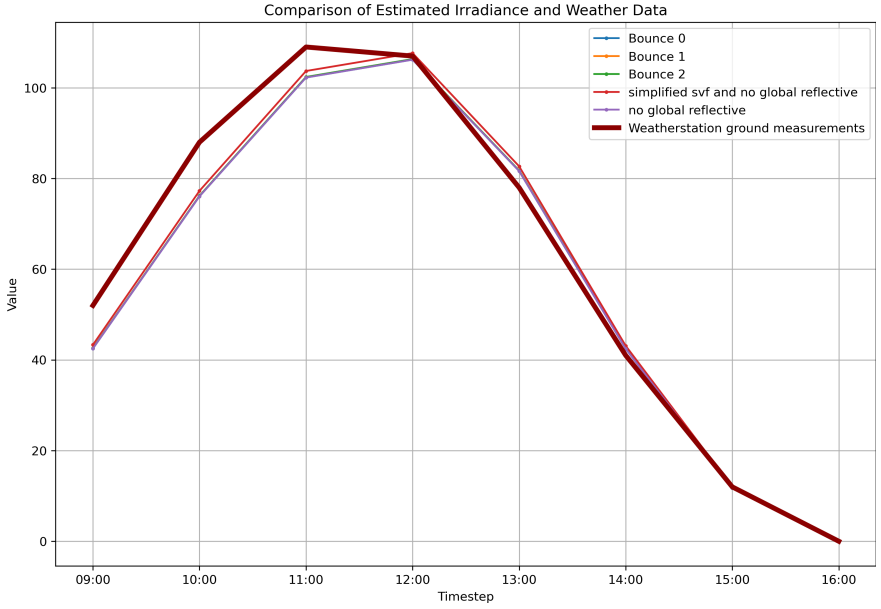
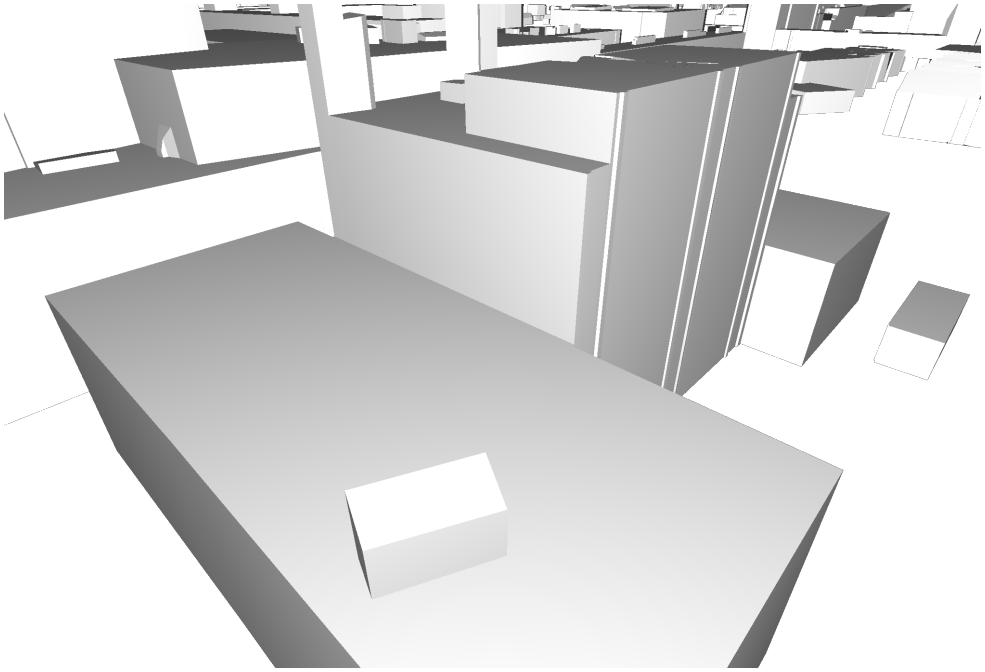
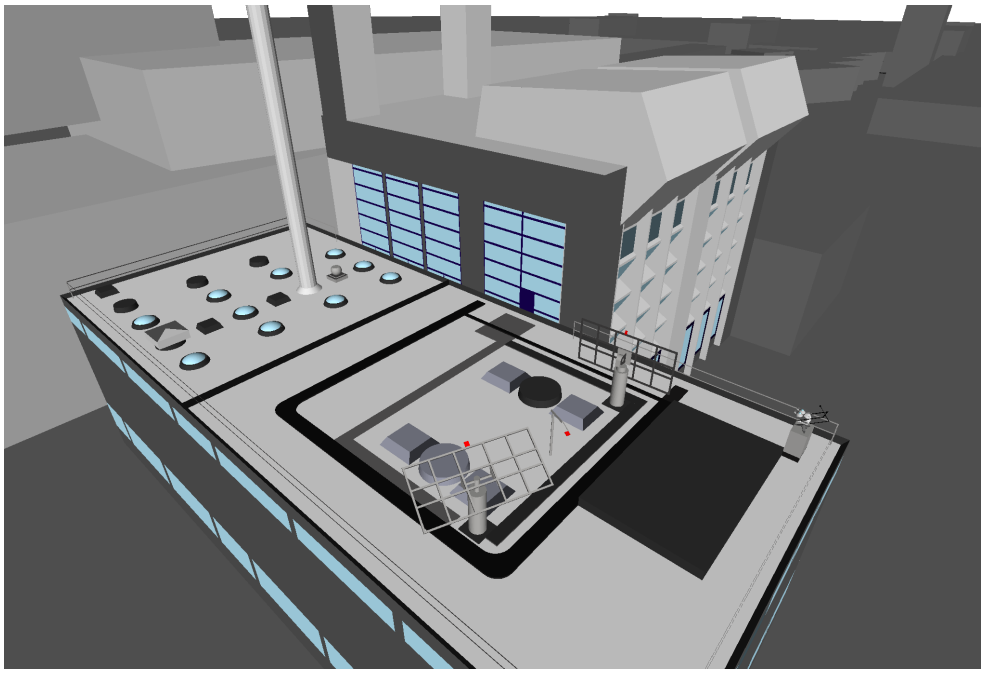


Figure 6.12.: Result of December 22. Heino dataset. Value unit is W/m^2

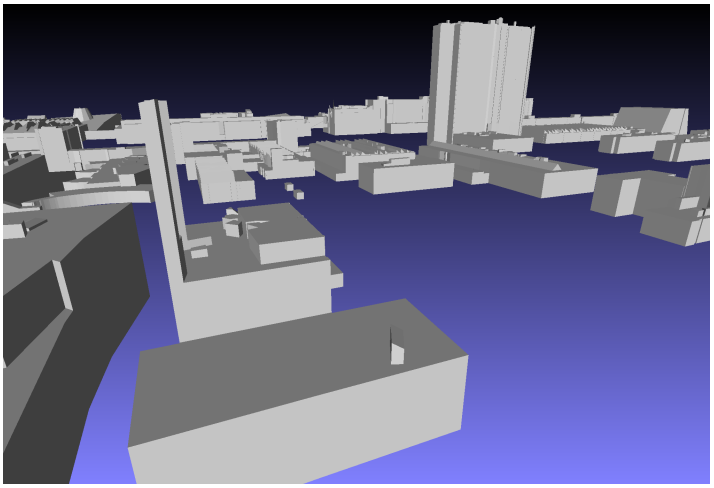


(a) Screenshot of the target sensors and building in 3DBAG Peters et al. [2022]

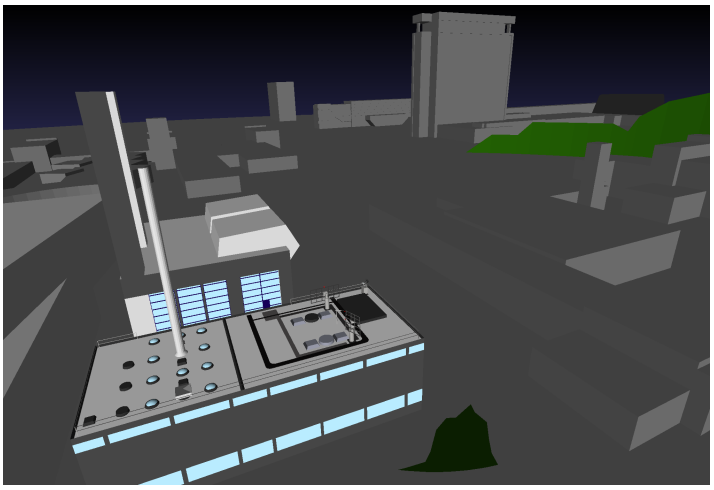


(b) Screenshot of the target sensors and building from Andres et al. [2023]

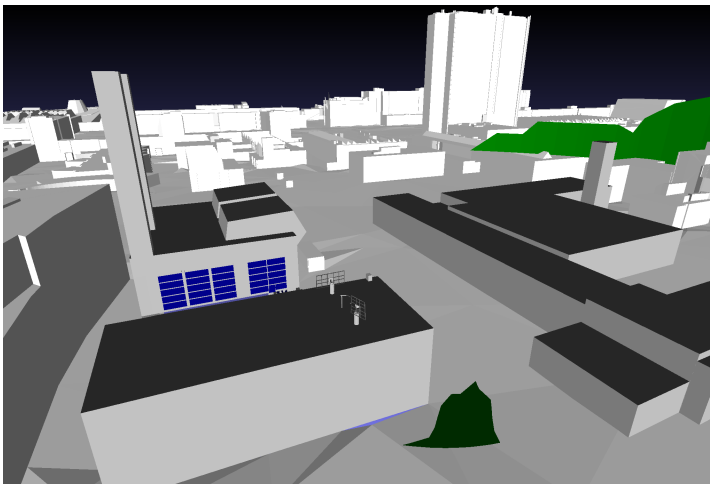
Figure 6.13.: screenshot of the sensors and the underlying building



(a) Screenshot of the 3DCM from Peters et al. [2022]

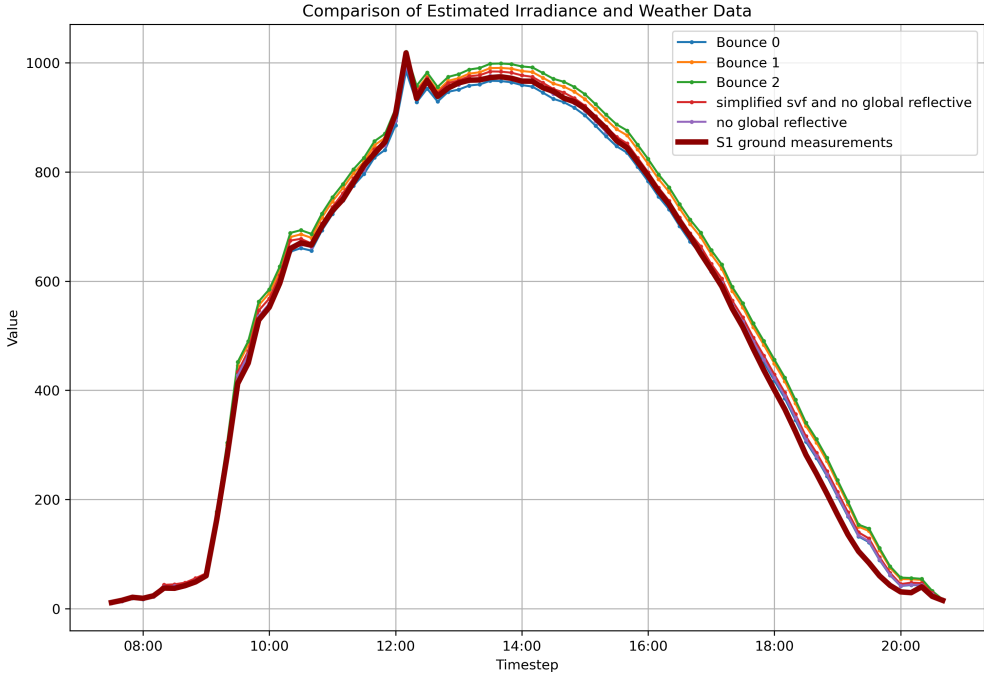


(b) Screenshot of the 3DCM from Andres et al. [2023]

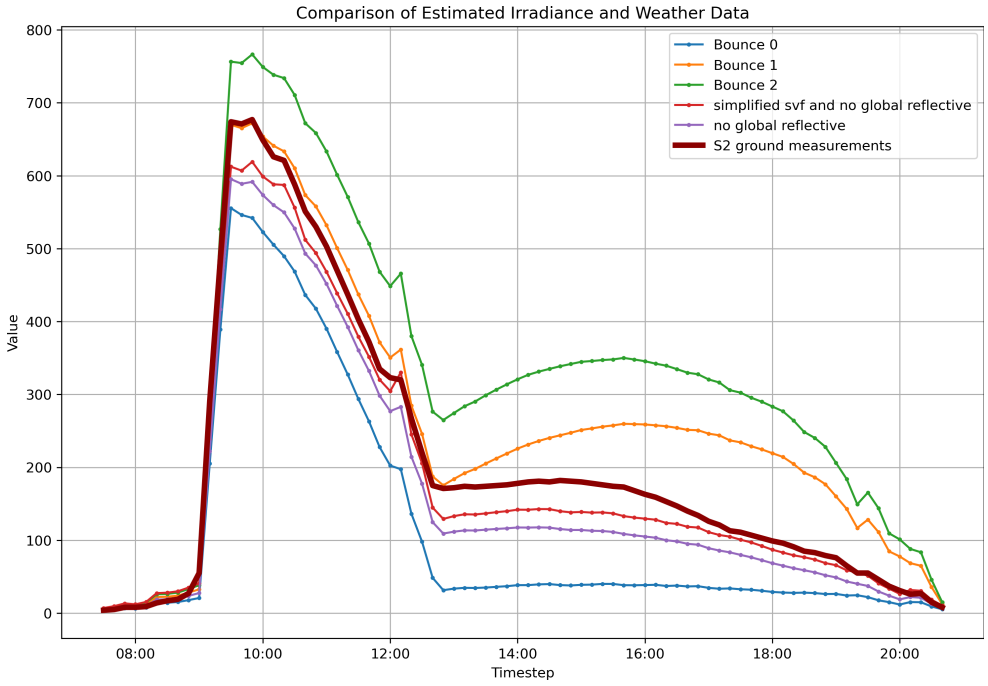


(c) Screenshot of the merged 3DCM

Figure 6.14.: Comparison of the 3DCM of TU Delft

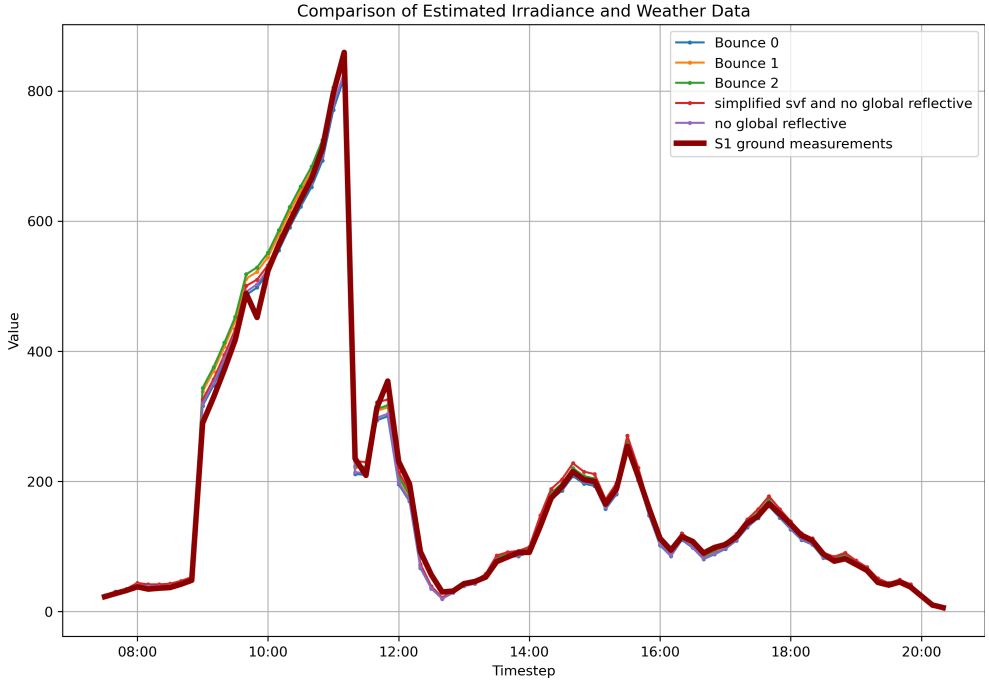


(a) Sensor S1

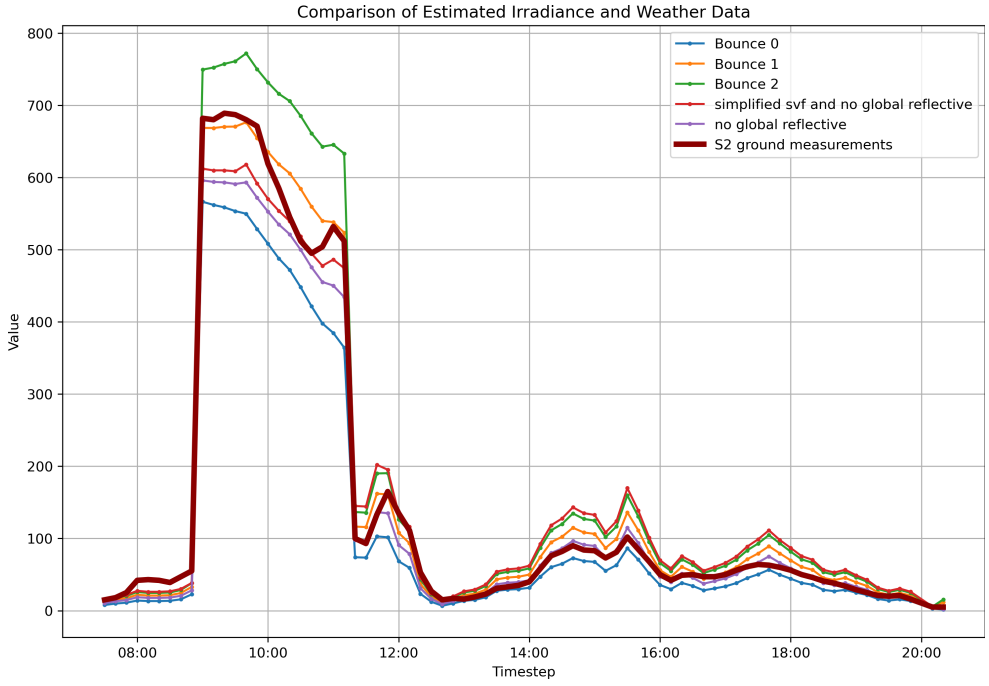


(b) Sensor S2

Figure 6.15.: Result of 2020 August 21. Value unit is W/m^2

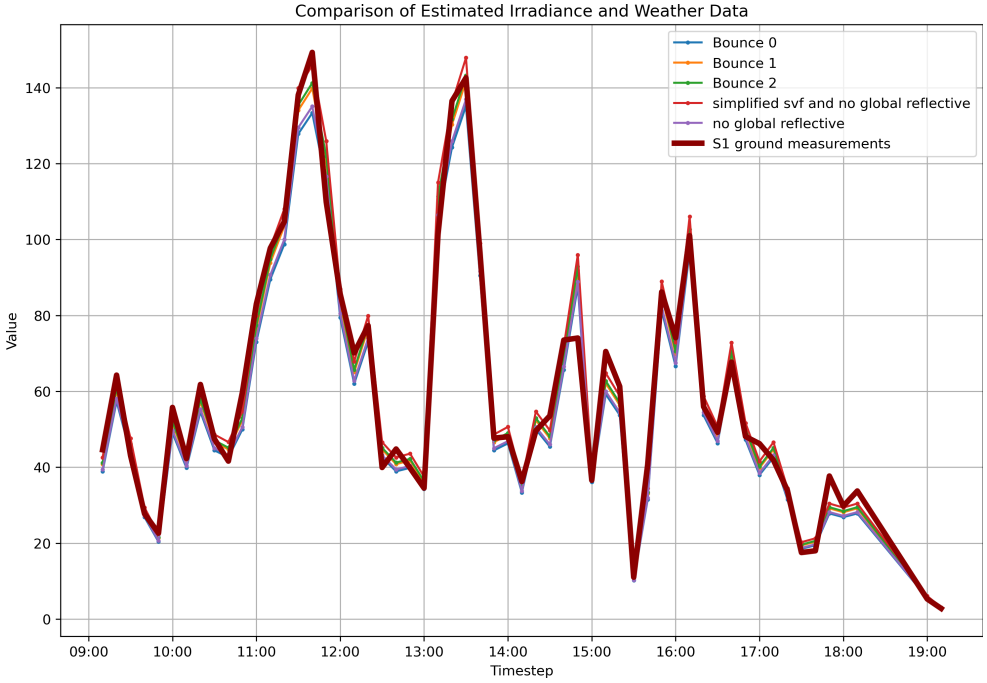


(a) Sensor S1

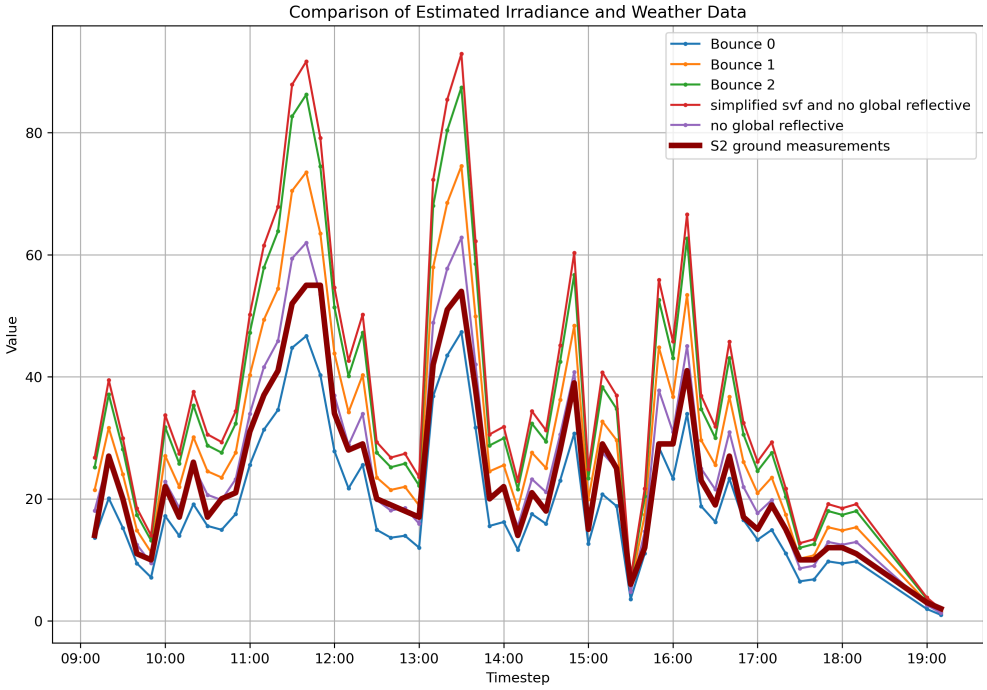


(b) Sensor S2

Figure 6.16.: Result of 2020 September 1. Value unit is W/m^2

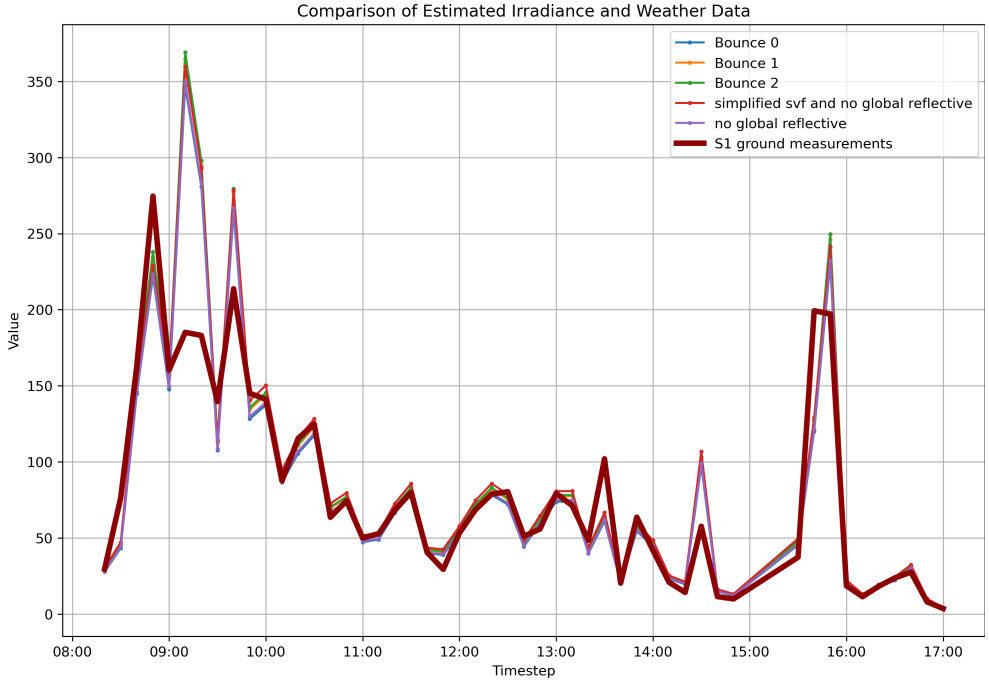


(a) Sensor S1

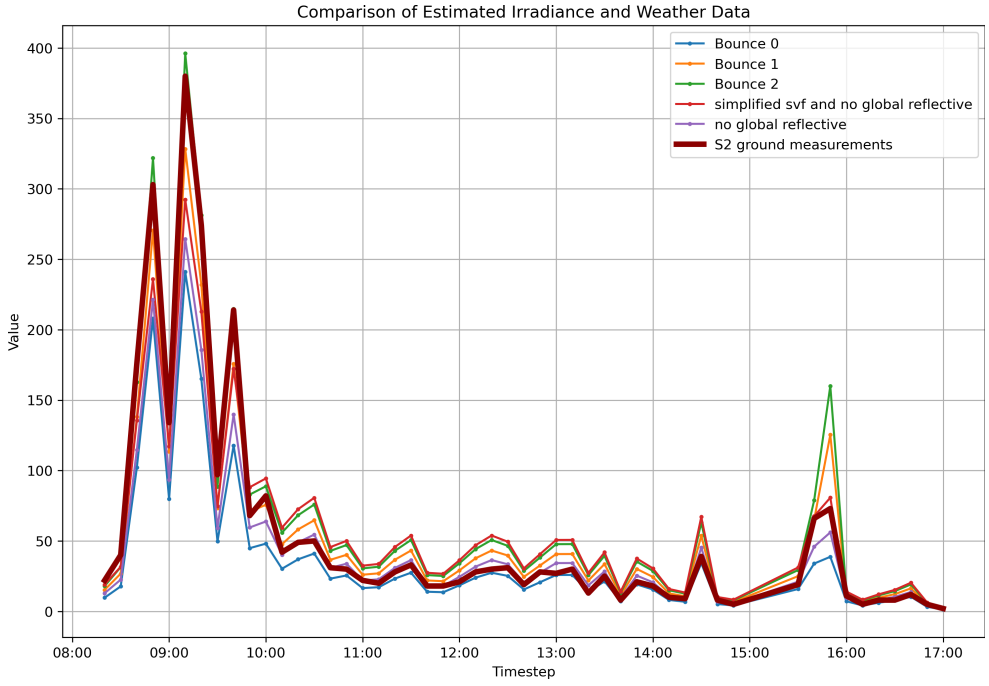


(b) Sensor S2

Figure 6.17.: Result of 2020 October 1. Value unit is W/m^2

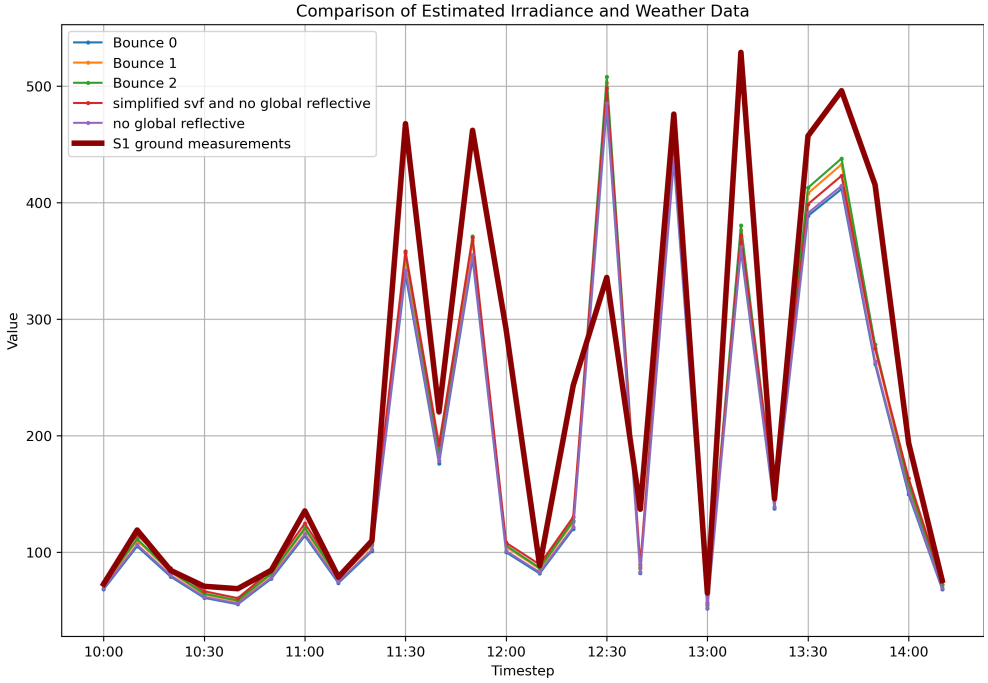


(a) Sensor S1

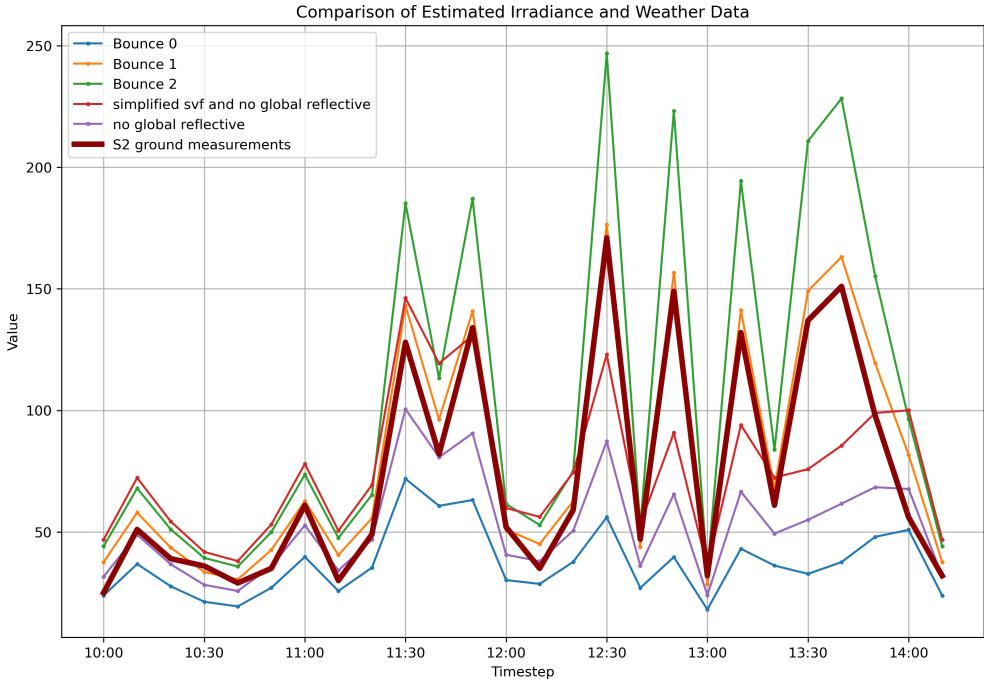


(b) Sensor S2

Figure 6.18.: Result of 2020 November 1. Value unit is W/m^2

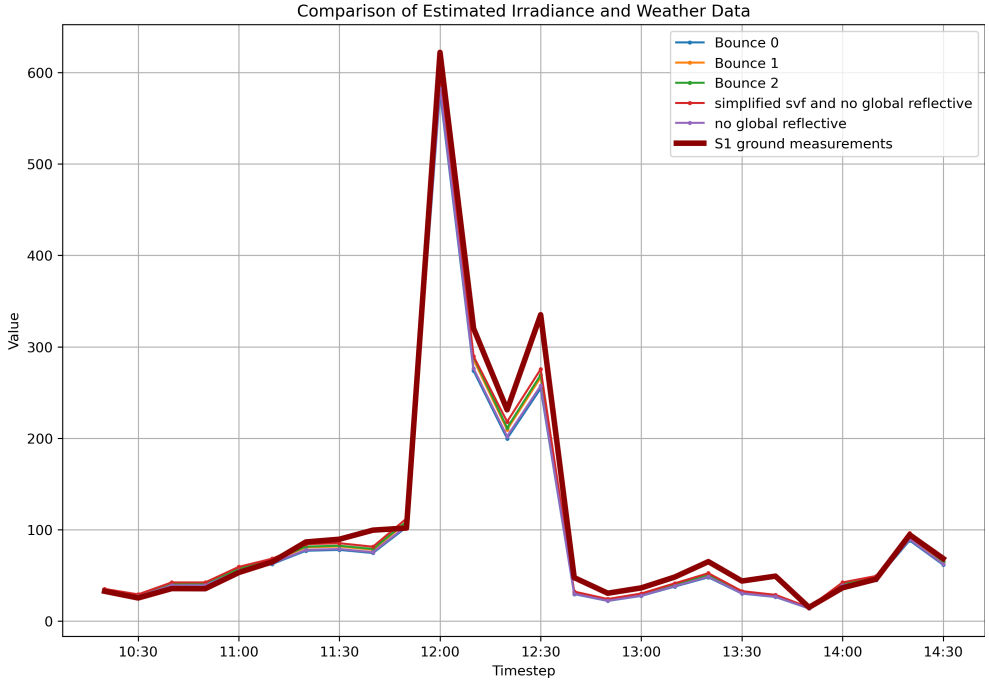


(a) Sensor S1

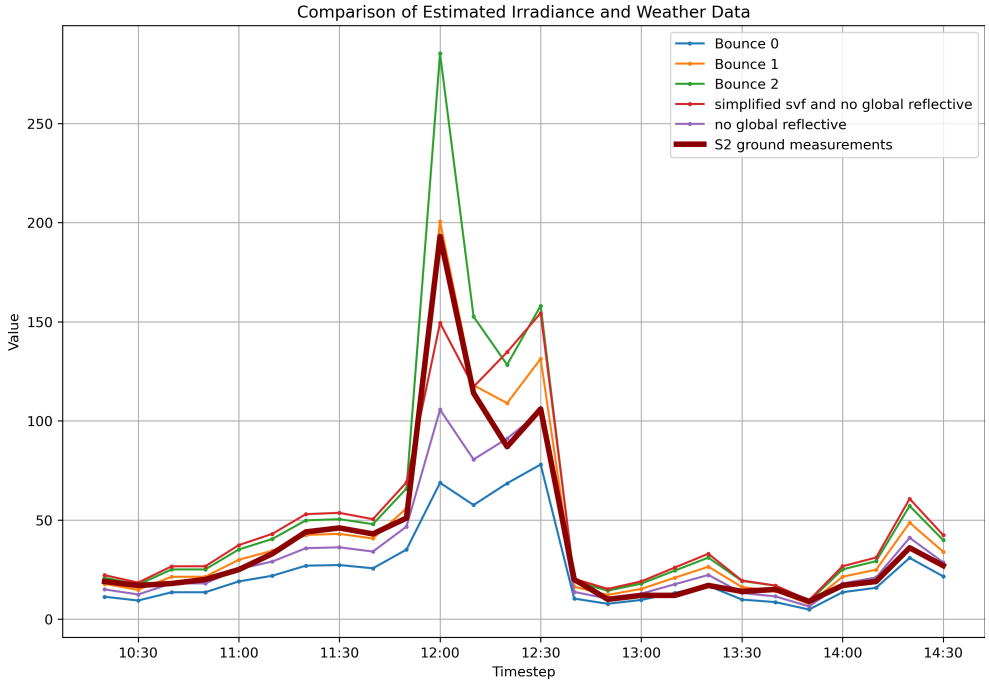


(b) Sensor S2

Figure 6.19.: Result of 2020 December 1. Value unit is W/m^2

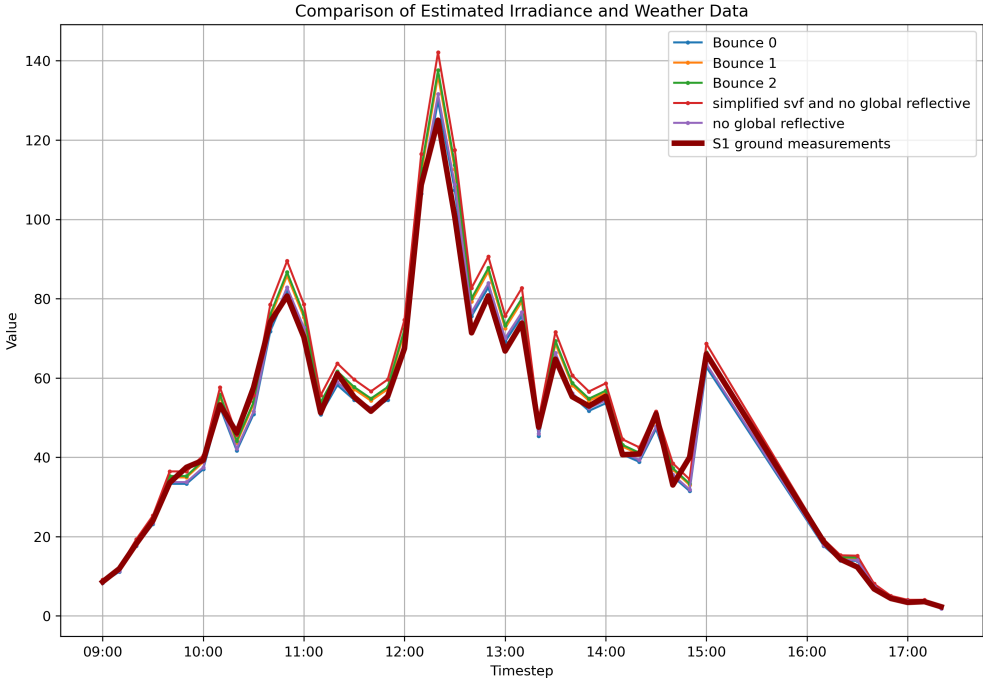


(a) Sensor S1

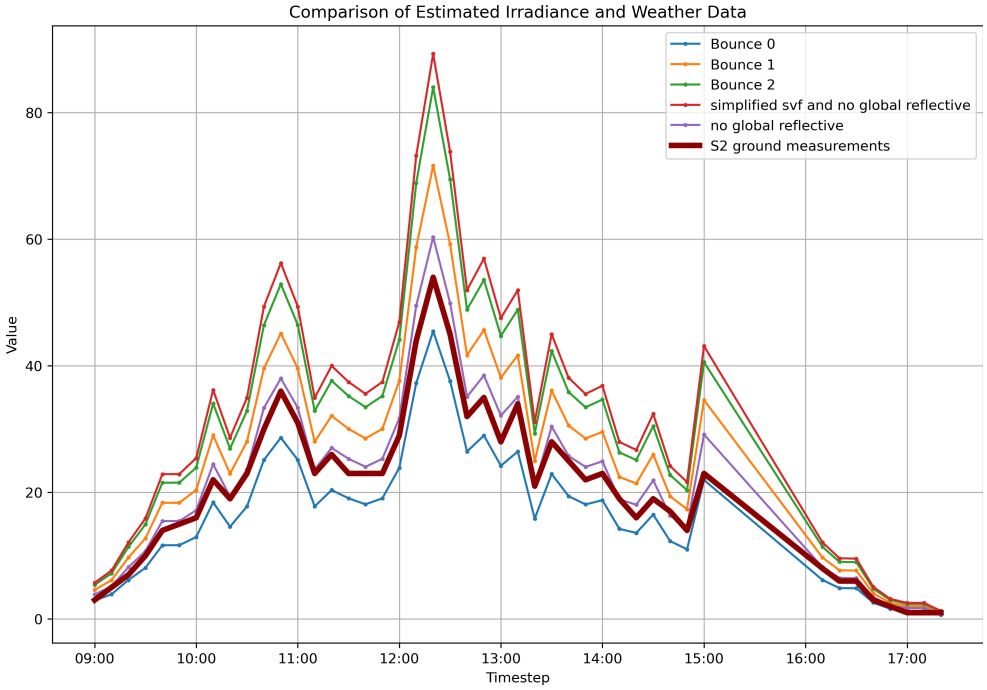


(b) Sensor S2

Figure 6.20.: Result of 2021 January 1. Value unit is W/m^2



(a) Sensor S1



(b) Sensor S2

Figure 6.21.: Result of 2021 February 1. Value unit is W/m^2

Metrics	Setting	2020/8/20	2020/9/1	2020/10/1	2020/11/1	2020/12/1	2021/1/1	2021/2/1	Average
Correlation	b=0	0.9996	0.9979	0.9900	0.8938	0.9112	0.9946	0.9960	0.9690
	b=1	0.9990	0.9976	0.9900	0.8946	0.9118	0.9947	0.996	0.9691
	b=2	0.9989	0.9974	0.9900	0.8947	0.9 118	0.9947	0.9960	0.9691
	setting 1	0.9995	0.9983	0.9900	0.8971	0.9125	0.9956	0.9960	0.9699
	setting 2	0.9996	0.9980	0.9900	0.8941	0.9113	0.9947	0.9960	0.9691
nMBE	b=0	**0.0000***	-0.0282	-0.0647	**0.0169***	-0.2015	-0.1302	**-0.0110**†	-0.0598
	b=1	0.0401	**0.0167***	**-0.0194**†	0.0684	-0.1637	-0.0900	0.0370	-0.0158
	b=2	0.0513	0.0281	**-0.0093***	0.0803	**-0.1548***	**-0.0807**†	0.0477	**-0.0053**†
	setting 1	0.0227	0.0264	0.0232	0.0915	**-0.1564**†	**-0.0661***	0.0821	**0.0033***
	setting 2	**0.0099**†	**-0.0173**†	-0.0526	**0.0281**†	-0.1940	-0.1206	**0.0019***	-0.0492
nMAE	b=0	**0.0194**†	0.0462	0.0826	**0.2002**†	0.2508	0.1390	**0.0406***	0.1113
	b=1	0.0406	**0.0453**†	0.0595	0.2038	0.2201	0.1119	0.0553	**0.1052***
	b=2	0.0514	0.0517	**0.0563***	0.2061	**0.2129***	**0.1061**†	0.0628	0.1068
	setting 1	0.0232	**0.0453**†	**0.0592**†	0.2101	**0.2142**†	**0.1041***	0.0909	**0.1067**†
	setting 2	**0.0152***	**0.0407***	0.0753	**0.1983***	0.2445	0.1323	**0.0409**†	**0.1067**†
nRMSE	b=0	**0.0255**†	0.0726	0.1046	**0.4191***	0.3613	0.2229	**0.0576***	0.1805
	b=1	0.0499	0.0789	**0.0811**†	0.4502	0.3382	0.1753	0.0811	0.1792
	b=2	0.0602	0.0885	**0.0796***	0.4593	**0.3336***	**0.1656**†	0.0904	0.1825
	setting 1	0.0310	**0.0655***	0.0862	0.4441	**0.3349***	**0.1562***	0.1244	**0.1775**†
	setting 2	**0.0243***	**0.0677**†	0.0961	**0.4231**†	0.3566	0.2116	**0.0597**†	**0.1770***

Table 6.5.: Sensor S1 Table showing various metrics across different settings and dates. Numbers marked with * represent the best entry, while the numbers marked with † represent the worst entry

Metrics	Setting	2020/8/20	2020/9/1	2020/10/1	2020/11/1	2020/12/1	2021/1/1	2021/2/1	Average
Correlation	b=0	0.9695	0.9972	0.9897	0.9929	0.6786	0.9036	0.9966	0.9326
	b=1	0.9735	0.9964	0.9897	0.9872	0.9907	0.9903	0.9966	0.9892
	b=2	0.9558	0.9949	0.9897	0.9856	0.9932	0.9941	0.9966	0.9871
	setting 1	0.9968	0.9928	0.9897	0.9825	0.7749	0.9256	0.9966	0.9513
	setting 2	0.9949	0.9972	0.9897	0.9929	0.8111	0.9346	0.9966	0.9596
nMBE	b=0	-0.4327	-0.2205	**-0.1831**†	-0.3358	-0.4966	-0.3751	**-0.1822**†	-0.3180
	b=1	0.2435	**0.0487***	0.2865	**0.0050***	**0.1035**†	**0.1123***	0.2880	**0.1554**†
	b=2	0.5738	0.2183	0.5091	0.1988	0.4224	0.3729	0.5108	0.4009
	setting 1	**-0.1065***	**0.0513**†	0.6040	**0.0649**†	**0.0293***	0.2467	0.6059	0.2137
	setting 2	**-0.2009**†	-0.0995	**0.0842***	-0.1906	-0.2910	**-0.1509**†	**0.0854***	**-0.1090***
nMAE	b=0	0.4327	0.2205	**0.1831**†	0.3365	0.4966	0.3776	**0.1835**†	0.3186
	b=1	0.2554	**0.0999***	0.2878	**0.2071***	**0.1142***	**0.1436***	0.2880	**0.1994**†
	b=2	0.5770	0.2410	0.5094	**0.2241**†	0.4224	0.3746	0.5108	0.4085
	setting 1	**0.1180***	0.1787	0.6041	0.3220	**0.3162**†	0.3318	0.6059	0.3538
	setting 2	**0.2024**†	**0.1221**†	**0.1024***	0.2491	0.3186	**0.2021**†	**0.0874***	**0.1834***
nRMSE	b=0	0.4968	0.3493	**0.2119**†	0.6674	0.7139	0.7306	**0.2150**†	0.4836
	b=1	0.3234	**0.1330***	0.3535	**0.3141***	**0.1395***	**0.2083***	0.3498	**0.2602***
	b=2	0.6471	0.3613	0.6051	**0.3212**†	0.5451	0.6336	0.6109	0.5320
	setting 1	**0.1427***	0.2271	0.7133	0.4467	**0.4023**†	**0.4745**†	0.7227	0.4470
	setting 2	**0.2340**†	**0.2163**†	**0.1382***	0.5347	0.5158	0.4785	**0.1190***	**0.3195**†

Table 6.6.: Sensor S2. The table shows various metrics across different settings and dates. Numbers marked with * represent the best entry, while the numbers marked with † represent the worst entry

Date	#Timestep	Shadow computation	SVF computation	Direct+Diffuse	Reflective	Total	Reflective ratio
2024/3/21	12	0.13	37.37	0.29	3232.19	3269.98	98.84
2024/6/21	16	0.20	46.61	0.37	5271.96	5319.14	99.11
2024/9/22	12	0.22	44.67	0.29	4675.88	4721.06	99.04
2024/12/22	8	0.20	41.60	0.26	4022.49	4064.55	98.97

Table 6.7.: Computation time Heino with the proposed method. Unit is second

Date	#Timestep	Shadow computation	SVF computation	Direct+Diffuse	Reflective	Total	Reflective ratio
2024/8/21	80	0.62	51.03	1.47	12319.02	12372.14	99.57
2024/9/1	77	0.58	49.63	1.39	15331.52	15383.12	99.66
2024/10/1	57	0.51	41.64	1.00	9093.19	9136.34	99.53
2024/11/1	50	0.44	32.25	0.99	8118.75	8152.43	99.59
2024/12/1	26	0.23	31.56	0.59	4073.14	4105.52	99.21
2024/1/1	26	0.12	23.44	0.46	3765.48	3789.50	99.37
2024/2/1	45	0.22	26.76	0.68	8017.02	8044.68	99.66

Table 6.8.: Computation time TU Delft with the proposed method. Unit is second

6.2. Testing on larger areas

6.2.1. Study areas

In addition to the Heino weather station dataset and the TU Delft dataset, we conducted further testing of our model on larger areas in the cities of Amsterdam, Delft, Rijssen-Holten, and Rotterdam to evaluate its scalability and robustness. These cities, distributed across the Netherlands as shown in figure 6.23, are representative of large-scale urban settings with dense 3D structures. For each city, we generated a CityJSON file containing buildings, terrain, and vegetation, covering areas exceeding 35 km^2. The H3 tiling approach was used to generate the test datasets, as hexagonal tiles minimize the boundary effects Uber Technologies [2024]. Details of the study areas are provided in table 6.9, where it can be noted that these areas are significantly larger compared to the Heino weather station dataset (table 6.1) and the TU Delft dataset (table 6.3). Each study area spans more than 35 km^2 of land, contains between 15,000 and 46,000 buildings, and includes a total of 1.4 to 4.1 million surfaces, with sampled grid points ranging from 6 to 16 million.

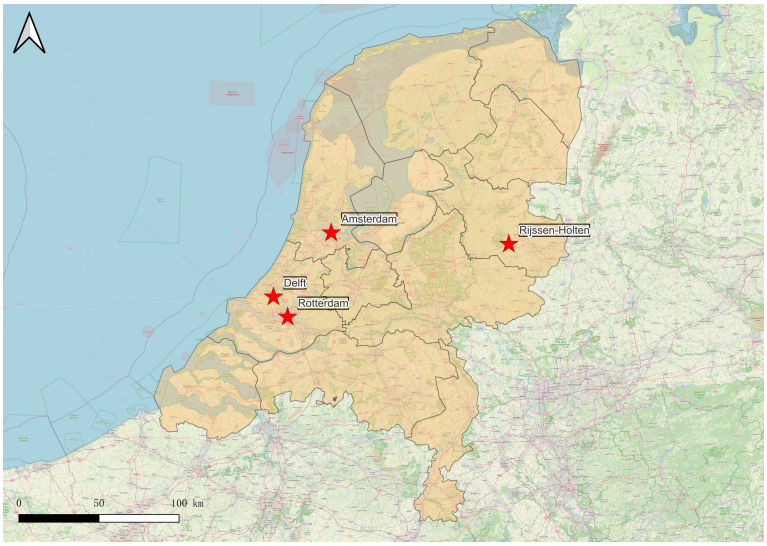
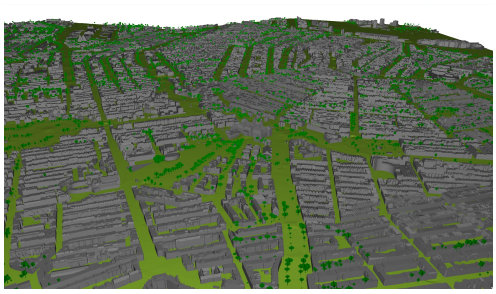
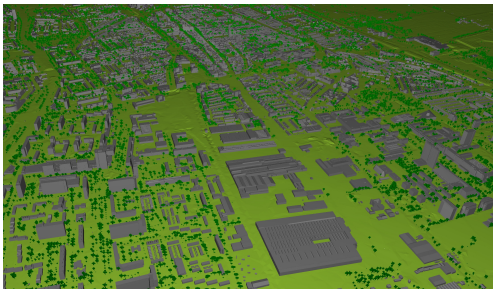


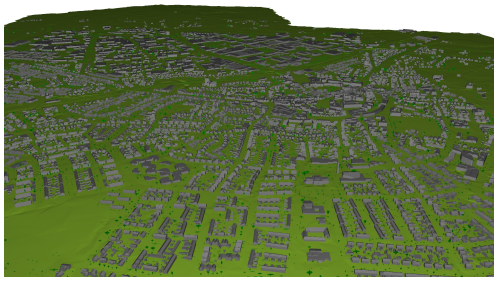
Figure 6.22.: Distribution of the study areas



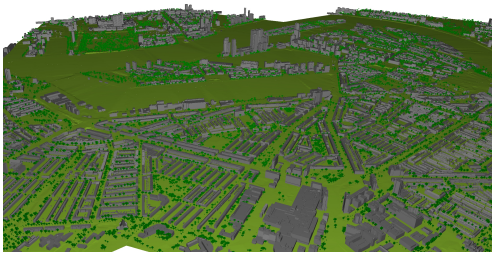
(a) Amsterdam



(b) Delft



(c) Rijssen-Holten



(d) Rotterdam

Figure 6.23.: Overview of the study areas in the scalability test

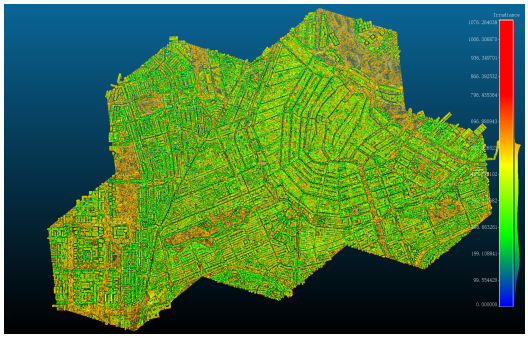
Attribute	**Amsterdam**	**Delft**	**Rijssen-Holten**	**Rotterdam**
Geographic center	4.88E, 52.37N	4.37E, 52.01N	6.52E, 52.31N	4.50E, 51.90N
H3 indexes	871969c9bffffff 871969526ffffff 871969534ffffff 871969535ffffff 871969530ffffff	86196bb17ffffff	871f16cf1ffffff 871f16cf5ffffff 871f16cc4ffffff 871f16ce6ffffff 871f16ce2ffffff	87196bb52ffffff 87196bb51ffffff 87196bb50ffffff 87196bb53ffffff 87196ba2cffffff
Spatial extent	7091 * 5070 * 98	6680 * 6689 * 119	6100 * 6200 * 50	6998 * 5109 * 190
#Buildings/#Faces	46273/3499950	40210/1744663	15744/717747	31004/1322841
#Roof surfaces	214927	119189	49827	28847
#Wall surfaces	780241	428357	174436	317257
#Trees/#Faces	21833/523992	28512/684288	25358/608592	28847/692328
#Terrain/#Faces	435/130241	535/173378	398/100091	435/130006
#Total faces	4154183	2602329	1426430	2145175
#Sample points	16214787	12000715	6457806	10279549

Table 6.9.: Dataset details for four different cities

6.2.2. Results

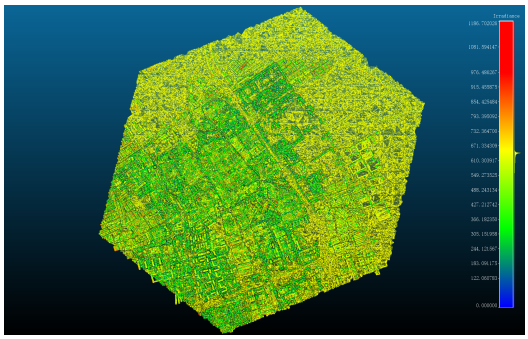
The simulation for February 2, 2021, was successfully run on the same laptop described at the beginning of this chapter. The computation times are recorded in table 6.10, where the most extended duration was observed for the city of Amsterdam. For more than 16 million sample points and 54 timesteps, the total computation time was 98,738 seconds or approximately 27 hours. A similar trend, as seen during validation, was also observed here, where the reflective solar irradiance calculation accounted for over 99% of the total computation time. Without simulating reflective solar irradiance, the entire simulation would take less than 7 minutes. These results demonstrate the model's capability to handle city-scale simulations.

Screenshots of the results, presented as coloured point clouds, are shown in figure 6.24, providing an overview of the computation outcomes.

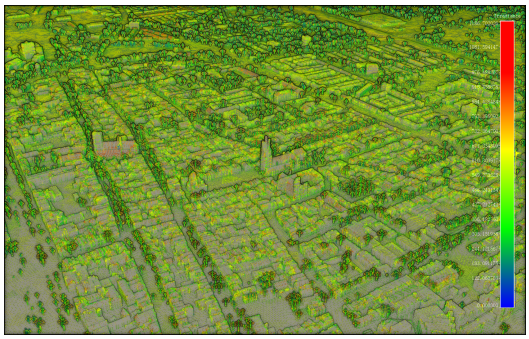


(a) Amsterdam - overview

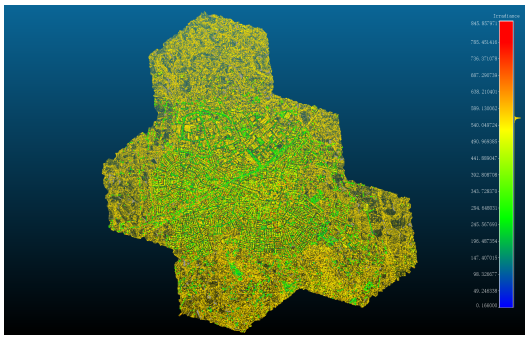
(b) Amsterdam - zoomed in



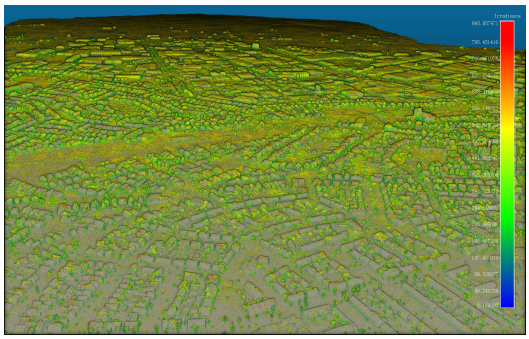
(c) Delft - overview



(d) Delft - zoomed in

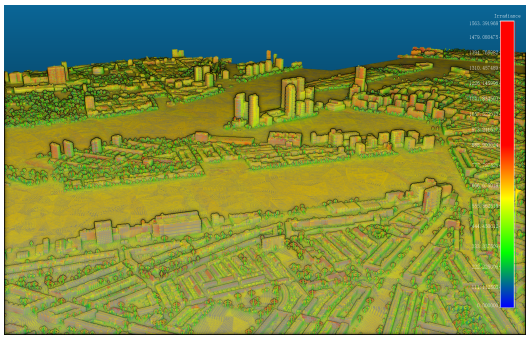


(e) Rijssen-Holten - overview



(f) Rijssen-Holten - zoomed in

(g) Rotterdam - overview



(h) Rotterdam - zoomed in

Figure 6.24.: screenshot of cities results displayed as a coloured point cloud. Unit is Wh/m^2

City	#Timestep	Shadow computation	SVF computation	Direct+Diffuse	Reflective	Total	Reflective ratio
Amsterdam	54	2.33	406.48	27.06	98302.20	98738.07	99.56
Delft	54	1.57	301.10	16.21	79782.55	80101.43	99.60
Rijssen-Holten	54	1.02	151.44	5.97	42941.12	43093.58	99.65
Rotterdam	54	1.58	240.70	14.39	80424.76	80681.43	99.68

Table 6.10.: Computation time of the four cities with the proposed method. Unit is second

Additional analysis was conducted to explore potential applications for high-resolution, city-scale solar irradiance simulations.

As shown in table 6.11, the orientation with the lowest average solar irradiance for Delft, Rijssen-Holten, and Rotterdam is east, while for Amsterdam, surfaces facing north have the lowest values. Conversely, surfaces oriented south receive the highest average solar irradiance in Amsterdam, Delft, and Rotterdam. For Rijssen-Holten, surfaces facing southwest have the highest solar irradiance.

Results in table 6.12 present the average solar irradiance values grouped by surface inclination. Surfaces with an inclination of -0°(facing down) have the lowest average irradiance for all four cities. The surfaces receiving the most solar irradiance have inclinations of 30°for Delft and Rijssen-Holten and 45°for Amsterdam and Rotterdam.

Orientation	Amsterdam	Delft	Rijssen-Holten	Rotterdam
east	301.90	**390.17**$^{-}$	**361.43**$^{-}$	**349.95**$^{-}$
north	**267.16**$^{-}$	435.10	424.53	400.49
northeast	293.52	414.16	451.07	405.99
northwest	303.21	408.25	446.68	398.53
south	**457.67***	**615.00***	484.40	**611.62***
southeast	440.36	549.69	492.35	516.33
southwest	379.14	587.55	**494.79***	592.71
west	302.90	546.36	443.82	499.84

Table 6.11.: Average solar irradiance for each orientation. Unit is Wh/m^2. Numbers marked with * represent the best value, while the numbers marked with $^{-}$ represent the lowest value

Inclination	Amsterdam	Delft	Rijssen-Holten	Rotterdam
-0°	**135.98**$^{-}$	**168.86**$^{-}$	**116.81**$^{-}$	**195.08**$^{-}$
-75°	304.13	453.66	329.67	442.60
-60°	265.97	368.49	220.01	328.79
-45°	214.57	276.47	209.53	237.93
-30°	163.63	277.59	135.32	165.47
-15°	131.12	144.04	128.71	155.22
0°	494.71	575.38	502.79	581.44
15°	440.42	553.81	506.38	540.40
30°	502.76	**597.69***	**518.69***	581.93
45°	**504.07***	594.85	503.19	**595.61***
60°	457.81	575.24	480.62	580.60
75°	428.53	543.52	426.92	545.44
90°	273.96	411.45	300.83	405.19

Table 6.12.: Average solar irradiance for each inclination. Unit is Wh/m^2. Numbers marked with * represent the best value, while the numbers marked with $^{-}$ represent the lowest value. -0°indicates surfaces that point towards the ground

We also calculated the average solar irradiance for all combinations of surface orientation and inclination. table 6.13 and table 6.14 document the combinations with the highest and

lowest average values. In all four cities, surfaces facing south with an inclination of 30°or 45°have the highest average solar irradiance. The lowest values are found for north-facing surfaces with a -15°inclination in Amsterdam and Delft. In Rijssen-Holten, the lowest irradiance occurs for surfaces facing southwest with an inclination of -30°, while in Rotterdam, it is for surfaces oriented northwest and inclined at -15°.

Aggregation of solar irradiance values by surface type was also applied, as shown in table 6.15. As expected, roof surfaces have the highest solar potential, while wall surfaces show the lowest potential.

These macro-level statistics for the entire study area provide valuable insights for urban planning. For example, in Delft, a general recommendation would be to install solar panels on south-facing roof surfaces with an inclination of 30 degrees. Since the simulation was conducted in winter, households with east-facing openings may require from additional heating solutions, as these surfaces receive less solar heating.

City	Orientation	Inclination	Irradiance
Amsterdam	south	45°	634.20
Delft	south	30°	791.49
Rijssen-Holten	south	45°	636.74
Rotterdam	south	30°	812.10

Table 6.13.: The combinations of orientations and inclinations that have the highest average solar irradiance. Unit is Wh/m^2

City	Orientation	Inclination	Irradiance
Amsterdam	north	-15°	39.08
Delft	north	-15°	56.83
Rijssen-Holten	southwest	-30°	62.83
Rotterdam	northwest	-15°	78.83

Table 6.14.: The combinations of orientations and inclinations that have the lowest average solar irradiance. Unit is Wh/m^2

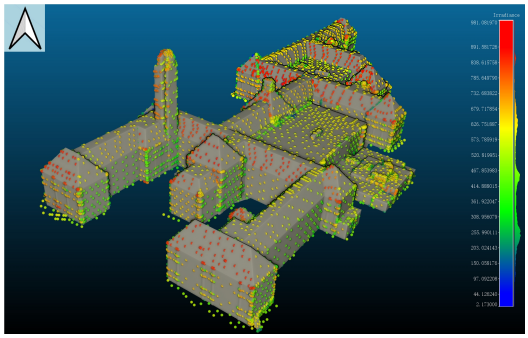
Surface type	Amsterdam	Delft	Rijssen-Holten	Rotterdam
Wall surface	**260.65⁻**	**393.17⁻**	**288.28⁻**	**390.05⁻**
Roof surface	**497.54***	**592.64***	**509.72***	**587.60***
Terrain surface	406.78	547.52	504.82	532.83
Vegetation surface	363.20	456.67	325.88	441.15

Table 6.15.: Average solar irradiance for each surface type. Unit is Wh/m^2. Numbers marked with * represent the best value, while the numbers marked with ⁻ represent the lowest value

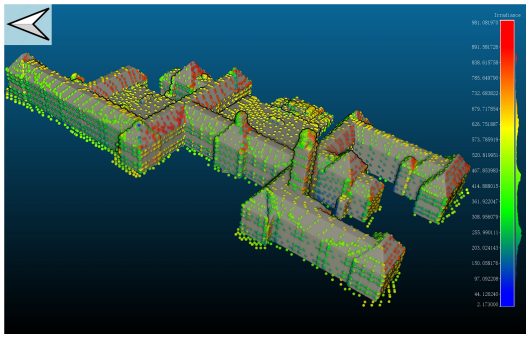
Analysis for specific buildings was also applied. The first chosen location is the BK Bouwkunde building, the main building of the Faculty of Architecture at Delft University of Technology.

Screenshots in figure 6.26 show the simulation results for the BK building (figure 6.25). The slanted rooftops facing southeast received higher total solar irradiance on the simulation day, suggesting they may be optimal locations for solar panel installation. Additionally, the marked polygons in figure 6.26e and figure 6.26f highlight the significant impact of shading from surrounding objects on the solar irradiance, even for surfaces with identical orientation and inclination. The points in polygon B in figure 6.26f further illustrate how surface orientation and shading influence total irradiance values, suggesting that during renovations, materials with varying thermal performance could be applied to these surfaces to optimize energy efficiency.

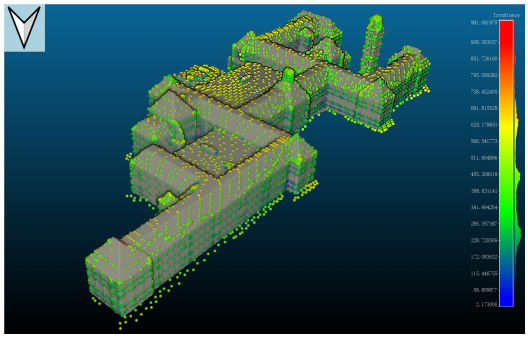
Figure 6.25.: Google Maps view of BK



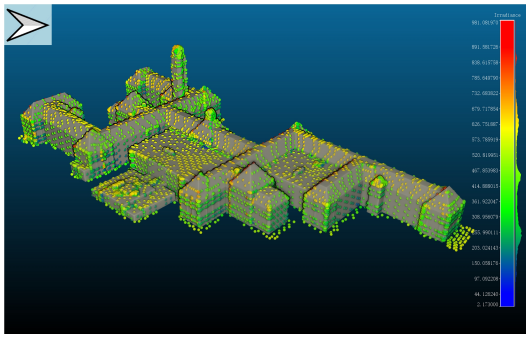
(a) Looking at the north direction



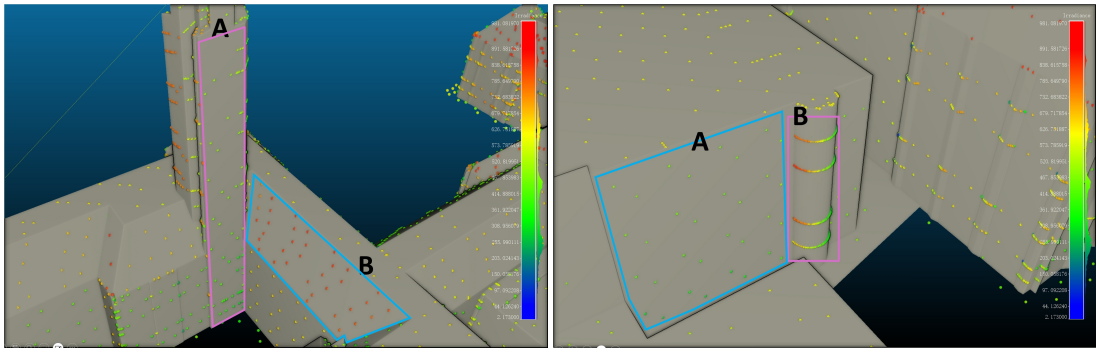
(b) Looking at the east direction



(c) Looking at the south direction



(d) Looking at the west direction

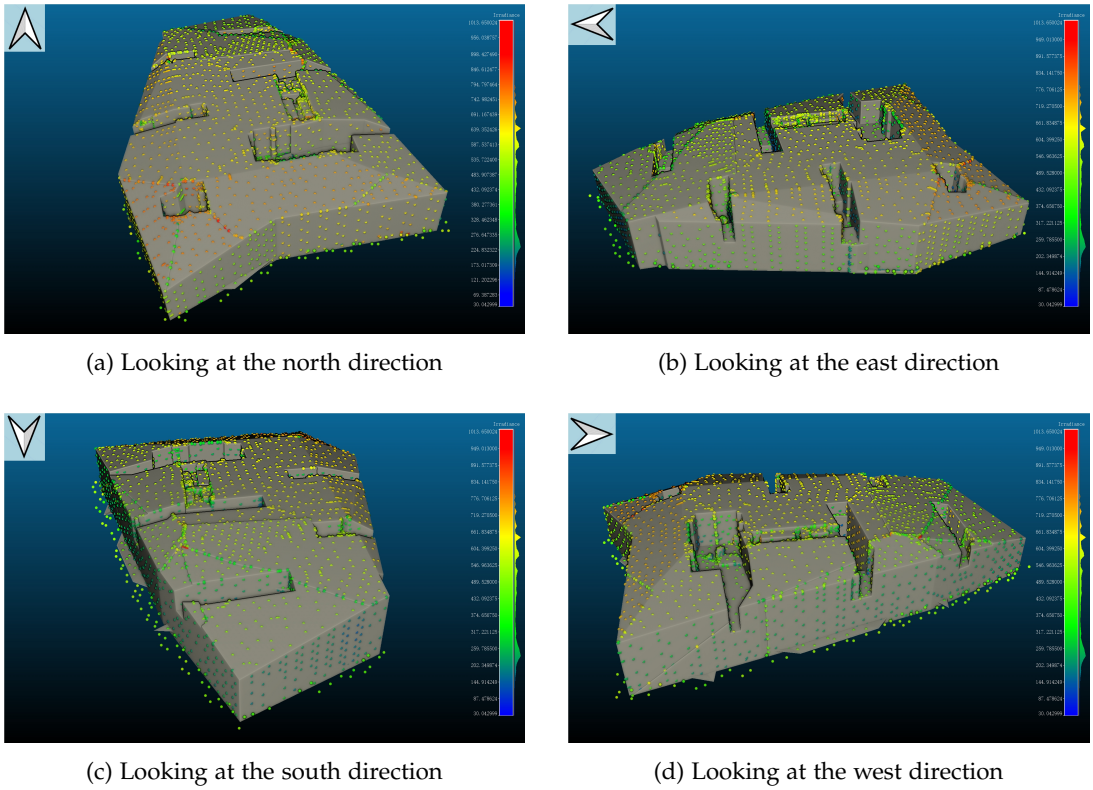


(e) screenshot of the tower (f) screenshot of the wall

Figure 6.26.: Screenshot of the BK Bouwkunde, the main building of the Faculty of Architecture at the Delft University of Technology. Unit is Wh/m^2

The screenshot in figure 6.28 displays the simulation results for Delft station, the central train station in the city. A comparison between our simulation and the actual scene on Google Maps (figure 6.27) reveals that the existing solar panels are not located on the rooftops with the highest solar potential. This finding highlights the potential of the proposed method to guide solar panel placement more effectively.

Figure 6.27.: Google Maps view of Delft Station



(a) Looking at the north direction

(b) Looking at the east direction

(c) Looking at the south direction

(d) Looking at the west direction

Figure 6.28.: Screenshot of the Delft Station, the train station of Delft. Unit is Wh/m^2

The aggregation of solar irradiance data based on orientation, inclination, combinations, and surface types was also applied to the BK building and Delft Station. Similar trends observed in the whole study area of the cities were evident in both buildings. For instance, roof surfaces exhibited higher average solar irradiance than wall surfaces, and south-facing (south or southwest) surfaces with inclinations directed towards the sky tended to receive more solar irradiance than others. Another observation can be found in the average solar irradiance intensity for the BK building was 416.36 Wh/m^2, slightly lower than the 439.06 Wh/m^2 recorded for Delft Station. This difference is likely due to the concave structure of the BK building and shading from nearby vegetation. This pattern can be seen in table 6.16, table 6.17, and table 6.20, where surfaces with the same inclination, orientation, or surface type tend to receive higher irradiance at Delft Station compared to the BK building.

It is also notable that within the same building, surfaces with different orientations and inclinations can show up to an 87% difference in solar irradiance intensity, as shown in table 6.18 and table 6.19. This underscores the importance of high-resolution solar irradiance simulations, as lower-resolution methods struggle to capture such detailed variations, leading to reduced accuracy in building-level energy demand and supply analyses.

We also calculated the total solar irradiance received by both buildings on the simulation day, February 2, 2021, as shown in figure 6.29. Despite the winter conditions and low sky clearness, the potential hourly solar energy reached as high as 26 kWh, with the BK building receiving a total of 143.70 kWh in a day, equivalent to the electricity consumption of an average Dutch household for 20 days Statistics Netherlands [2023].

These simulation results can also be extended to further applications, such as building thermal performance simulations, allowing for room-level or even surface-level energy performance evaluations.

Orientation	BK	Delft Station
east	343.83	380.12
north	**268.48**$^{-}$	**340.37**$^{-}$
northeast	298.51	350.99
northwest	293.43	340.57
south	526.48	**632.45***
southeast	465.82	545.03
southwest	**676.35***	504.08
west	427.11	434.49

Table 6.16.: Average solar irradiance for each orientation. Unit is Wh/m^2. Numbers marked with * represent the best value, while the numbers marked with $^{-}$ represent the lowest value

Inclination	BK	Delft Station
-75°	368.65	394.72
-30°	**274.00**$^{-}$	Nan
0°	409.29	**383.80**$^{-}$
15°	424.01	582.82
30°	**640.90***	**670.40***
45°	609.10	602.24
60°	570.48	Nan
75°	471.85	Nan
90°	380.09	428.60

Table 6.17.: Average solar irradiance for each inclination. Unit is Wh/m^2. Numbers marked with * represent the best value, while the numbers marked with $^{-}$ represent the lowest value

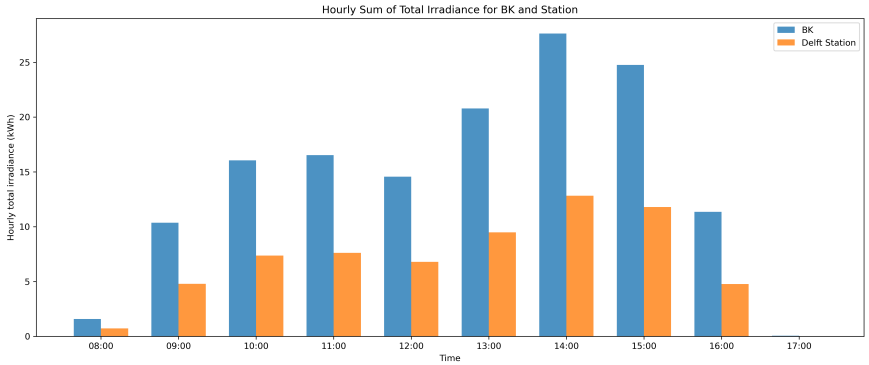


Figure 6.29.: Total solar irradiance received for BK building and Delft Station

Building	Orientation	Inclination	Irradiance
BK	southwest	75°	867.58
Delft Station	southwest	30°	779.85

Table 6.18.: The combinations of orientations and inclinations that have the highest average solar irradiance. Unit is Wh/m^2

Building	Orientation	Inclination	Irradiance
BK	northeast	90°	227.48
Delft Station	north	45°	103.19

Table 6.19.: The combinations of orientations and inclinations that have the lowest average solar irradiance. Unit is Wh/m^2

Surface type	BK	Delft Station
Wall surface	**379.01**$^{-}$	**428.13**$^{-}$
Roof surface	**576.70***	**563.80***

Table 6.20.: Average solar irradiance for each surface type. Unit is Wh/m^2. Numbers marked with * represent the best value, while the numbers marked with $^{-}$ represent the lowest value

7. Discussion

7.1. Answers to Research Questions

How can semantic data from 3D city models improve the accuracy of solar irradiance simulations by accounting for direct, diffuse, and reflected solar components? The comparison between the proposed method and two baseline approaches, demonstrates the significance of incorporating semantic and geometric data from semantic 3DCM for enhanced precision. Setting 1 uses simplified SVF and GVF to estimate both the sky diffuse and ground reflected components. Setting 2 calculates SVF and GVF in detail but applies a global albedo value for the reflective component. This comparison highlights the importance of incorporating semantic data to achieve greater precision.

In the simplified reflective irradiance calculation described in equation (2.11), a universal albedo value is applied to all occluding objects. The reflective irradiance is computed by multiplying this fixed albedo by GVF and GHI. In Setting 1, SVF and GVF are calculated based on surface inclination alone, ignoring the complex urban geometry, as articulated in equation (2.10). Although Setting 2 uses detailed hemisphere sampling for GVF, the reflective irradiance still follows the simplified computation in equation (2.11).

The proposed method refines this approach by using hemisphere scanning for both SVF and reflective irradiance, accounting for reflections from all incident angles, as described in algorithm 4.4.

For the Heino weather station dataset and TU Delft's sensor 1, improvements are subtle. However, for sensor 2 (S2), the advantage of incorporating semantic and geometric details from the 3D city model becomes evident. On December 1, 2020, from 13:00 to 14:00, and January 1, 2021, from 12:10 to 12:30, the simplified models not only deviate significantly from ground measurements but also fail to capture the trend in irradiance changes, as shown in figure 6.19b and figure 6.20b. Although the evaluation metrics in table 6.6 do not reveal significant differences, it is clear that simplified settings are less reliable.

What are the trade-offs between accuracy and computational efficiency when using 3D city models for urban-scale solar irradiance estimation? The computational feasibility of city-scale reflective irradiance modeling has been confirmed. For the entire study area, all three solar irradiance components were computed with logarithmic or linear complexity. Direct solar irradiance, based on shadow calculations, has a time complexity of $\mathcal{O}(log(M))$, where M represents the number of sample points in the city. Sky diffuse irradiance at each time step has a complexity of $\mathcal{O}(M)$, while reflected irradiance has a complexity of $\mathcal{O}(NM)$, where N is the number of light bounces considered.

In terms of implementation and experiment, GPU acceleration has significantly reduced the computation time for shadowing and SVF calculations. However, reflective irradiance is inherently computationally intensive, as each sample point requires processing every pixel in

the viewshed, representing the result of hemisphere-sampled rays. In addition, the computation of reflective component is not implemented with GPU parallel computing. The main bottleneck lies in data input/output and memory management. Despite this, the computation time remains acceptable for city-scale applications, with approximately 30 minutes required for 1 million sample points across 20 time steps. Further optimizations are possible to reduce this time.

In summary, the proposed method is scalable to city-scale simulations, effectively considering 3D urban environments and all solar irradiance components.

7.2. Research Implications

The methodology proposed in this thesis represents, to our knowledge, the first solar irradiance simulation model capable of balancing realism, computational complexity, and generalizability. Built on a combination of Ray Tracing and Viewshed methods, the model provides a detailed estimation of the three main components of solar irradiance: direct beam, sky diffuse, and ground reflected irradiance. Specifically, ground-reflected irradiance is estimated efficiently, avoiding the exponential growth of traced rays while maintaining the physical accuracy of light reflection.

The results demonstrate the model's ability to capture the reflectance of solar irradiance in complex urban environments. For instance, in the TU Delft dataset, sensor 2 results show that the method accurately captures reflective solar irradiance, particularly under clear-sky conditions with high overall irradiance. Additionally, the simplified version of the model, focusing on direct beam and sky-diffuse irradiance, performs well under overcast conditions, with predictions closely aligning with ground measurements.

These findings suggest that the proposed model holds promise as a suitable tool for realistic solar irradiance simulation in urban environments. By utilizing its predictions, urban energy systems could improve energy efficiency through proactive responses. Furthermore, researchers will be able to obtain realistic solar irradiance values for larger scales with reduced computational costs.

7.3. The influence of hyperparameters

A series of user-defined hyperparameters are needed for the methodology. The thesis did not run a strict grid search and ablation studies to evaluate the influence of these parameters. Here we provide some empirical evaluations regarding these evaluations.

- **Grid Sampling Density**. The grid sampling density defines the expected coverage area of per sample point. The unit of this parameter is m^2/point. It is worth noting that the actual coverage area of each sample point does not perfectly equal this density as the recursive splitting method described in algorithm 4.1 only sample points with number $(3^n - 1)/2$ where n represent the number of splits. In addition, addtional splitting of sliver triangles will also alter the coverage area of the point. However, the actual coverage area of each sample point will be recorded for subsequent computations to ensure consistency. This parameter mainly affects the spatial resolution of the simulation, and it will also affect the choosing of the parameter voxel resolution. The

computation overhead will grow linearly with the increase of grid sampling density. A sampling density between 9 to 36 can provide good coverage of while maintain a low computation overhead according to our test runs.

- **Hemisphere resolution**. The viewshed generation process involves tracing a hemisphere of rays to obtain the visibility information of the sample point. Each ray in the hemisphere represents a solid angle. The seperation of each ray is determined by the hemisphere resolution. The computation overhead also grows linearly with the resolution increase. Theoratically, the higher the resolution, the more detailed and accurate the modelling. In our test runs, the we have testd seperation angle with 2, 3, 5 degrees. And the resolution of 5 degree seperation angle has shown almost equivalent predictions but with faster computation speed.
- **Voxel size**. The voxel representation determines the resolution of the light field for reflective solar irradiance computation. Similarly, the smaller the voxel size, the more detailed the modelling. However, the voxel size should be larger than than the grid point sampling density. Otherwise the aggregation function of the voxel light filed will fail, reducing computation efficiency. The voxel size should also not too large to matain accurate modelling. The computation overhead increase related to this parameter is negletable. The test runs indicate a voxel resoltuion between 9 and 15 meters is optimal.

7.4. The influence of the scale of the study area

The relationship between city scale and computation time is illustrated in figure 7.1, figure 7.2, and figure 7.3, which show the computation time required per million points per timestep. The results reveal a logarithmic relationship between computation time and city scale, measured by the number of surfaces. For shadow computation, as depicted in figure 7.1, the computation time decreases as city scale increases, eventually stabilizing at a nearly constant level for cities with over 1.5 million surfaces. This behavior may be attributed to varying ray-object intersection hit rates caused by different urban morphologies.

It is also important to note that city scale cannot increase indefinitely for simulations to be completed on a consumer-level laptop without modifying the code. We tested a significantly larger tile in Rijssen-Holten with a spatial extent exceeding 20km by 20km, resulting in a point grid size of 60 million, compared to the 10 million points in the Delft tile. This increase in scale requires approximately six times more storage for intermediate files (around 1.5TB), as well as six times more computation time and memory. Such resource demands indicate that the current implementation may not be feasible for these larger scales. However, this limitation can be addressed by refactoring the code. Since code optimization is beyond the scope of this thesis, these improvements were not implemented.

In summary, these findings indicate that the method is scalable for larger areas, making it suitable for urban-scale analyses.

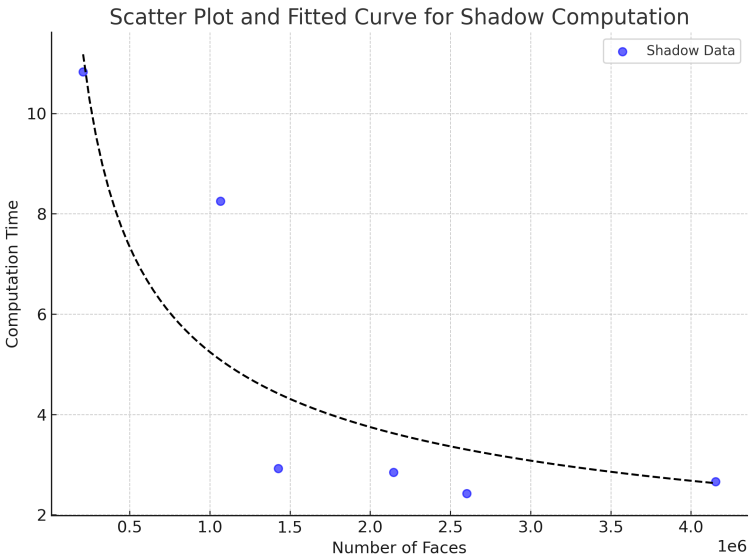


Figure 7.1.: Scatter plot and fitted curve for shadow computation. The unit of computation time is **millisecond** per million points per timestep

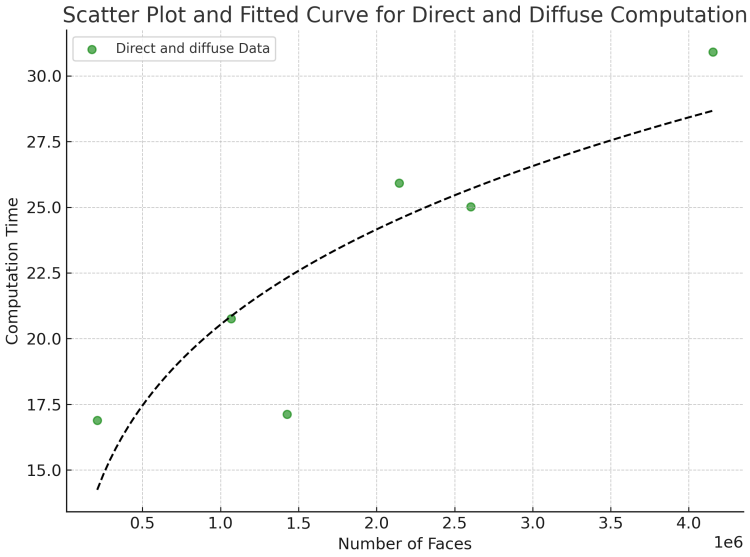


Figure 7.2.: Scatter plot and fitted curve for direct and diffuse irradiance computation. The unit of computation time is **millisecond** per million points per timestep

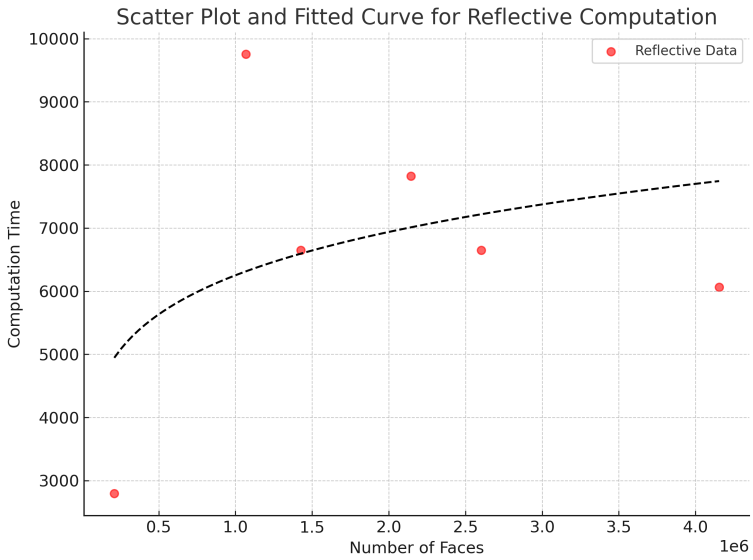


Figure 7.3.: Scatter plot and fitted curve for reflective solar irradiance computation. The unit of computation time is **second** per million points per timestep

8. Conclusion

8.1. Research Overview

This thesis presents a methodology for scalable and realistic solar irradiance simulation using 3D city models.

Solar irradiance plays a critical role in current and future energy systems, as it serves as a clean energy source, influences weather conditions, and impacts human energy consumption behaviors. Accurate modeling and prediction of solar irradiance are essential to reduce the gap between predicted energy demand and supply. However, existing methods face significant limitations. Some rely on data-driven predictions that are difficult to generalize, while others use simplified simulations that overlook the complex urban landscape. Additionally, ray-tracing methods, though accurate, are computationally expensive and time-consuming. As a result, achieving accuracy, scalability, and efficiency in solar irradiance modeling at the city scale, with high spatial and temporal resolution, remains a challenge.

This thesis develops a novel approach that specifically models the direct beam, sky diffuse, and reflected components of solar irradiance on urban surfaces. Dedicated data structures and algorithms ensure both computational speed and realism. The key innovation is the semantic scene voxelization algorithm, which reduces the complexity of modeling reflected solar irradiance from exponential to linear, making the method scalable for large-scale urban simulations. The implementation, carried out using C++ and Nvidia Optix Ray Tracing Engine, ensures high computational efficiency. The implementation will be made open source, and the details are described in the reproducibility self-assessment in appendix A. Validation against ground truth measurements shows that this method outperforms others with simplified assumptions, achieving significantly higher accuracy under clear sky conditions. Specifically, the proposed method reduces the average nRMSE by 18% compared to simplified models for sensor S2. Additionally, experiments demonstrate the scalability of the method, with simulations for a large urban area spanning over 35 km^2 with approximately 16 million sample points, accounting for two reflections over 54 timesteps, completing in 27 hours.

The experimental results suggest that leveraging the detailed semantic and geometric information provided by 3DCM can significantly improve the accuracy of solar irradiance modeling. In particular, the precise computation of reflected solar irradiance is more accurate under clear sky conditions, when solar power is abundant.

8.2. Limitations

Several limitations persist:

- **Simplifications and assumptions**. The 3DCM inherently involves simplifications in geometry and semantics. Additionally, this thesis uses the isotropic model to approximate sky diffuse radiation, which is a simplification. For reflective solar irradiance estimation, all surfaces are assumed to behave as perfect Lambertian reflectors. However, in reality, reflection patterns vary by material and include specular and glossy reflections, among others. These assumptions inevitably limit prediction accuracy.
- **Limited validation data**. The scarcity of ground measurements and surface material information constrains the validation process. In essense, only Sensor S2 is representative in complex urban environments, capturing a wide variety of urban objects in its viewshed. Consequently, additional validation remains necessary. Accurate reflective irradiance computation requires specific surface material information, such as albedo and reflection patterns, which was unavailable for most surfaces in the study area.
- **High computational overhead for reflective irradiance**. Reflective solar irradiance computation consumes around 99% of the total processing time, yet the resulting accuracy improvements are inconsistent.

8.3. Future Work

Potential improvements based on the identified limitations include:

- **More detailed input data**. Using 3DCM with higher LoD and comprehensive surface material data could enhance prediction accuracy.
- **Enhanced modeling of sky diffuse irradiance**. This thesis adopts an isotropic model, assuming uniform irradiance from the sky dome. Alternative models, such as the Perez model, offer more realistic representations by dividing the sky into circumsolar, horizon, and other zones. The semantic viewshed already includes the data necessary to integrate advanced sky diffuse models.
- **Improved light field representation**. Currently, reflective solar irradiance is computed with voxels that has limited directional resolution. Spherical harmonics could replace this voxel representation to more accurately simulate irradiance distribution, while supporting varied reflection patterns.
- **Model calibration**. Deriving empirical calibration factors from additional validation data could further refine model accuracy.
- **Database input support**. Enhancing the method to accept input via database connections would improve interoperability, offering a more flexible alternative to the current file-based input system.
- **Integration with remote sensing data**. Current methods simulate solar irradiance for the entire area using the same values recorded by a single weather station. However, these recorded values may not accurately reflect solar conditions across the study area due to variations in cloud cover. Future work could incorporate remote sensing data to provide more accurate GHI, DNI, and DHI values, thereby improving the accuracy of the simulations.

8.4. Applications

Despite these limitations, the proposed method has considerable potential across various applications:

- **Building energy performance evaluation**. Detailed solar exposure modeling can assist architects and energy planners in assessing building energy performance with greater accuracy.
- **Power grid management**. The methodology can estimate the solar energy output of distributed solar power grids, aiding in grid scheduling and management.
- **Household solar potential assessment**. This tool can help individuals assess the solar potential of their homes, supporting informed decisions on solar panel installations.
- **Visibility analysis**. The semantic viewshed generated during simulations can also serve as a substitute for Street View Imagery (SVI) in various applications.

A. Reproducibility self-assessment

A.1. Marks for each of the criteria

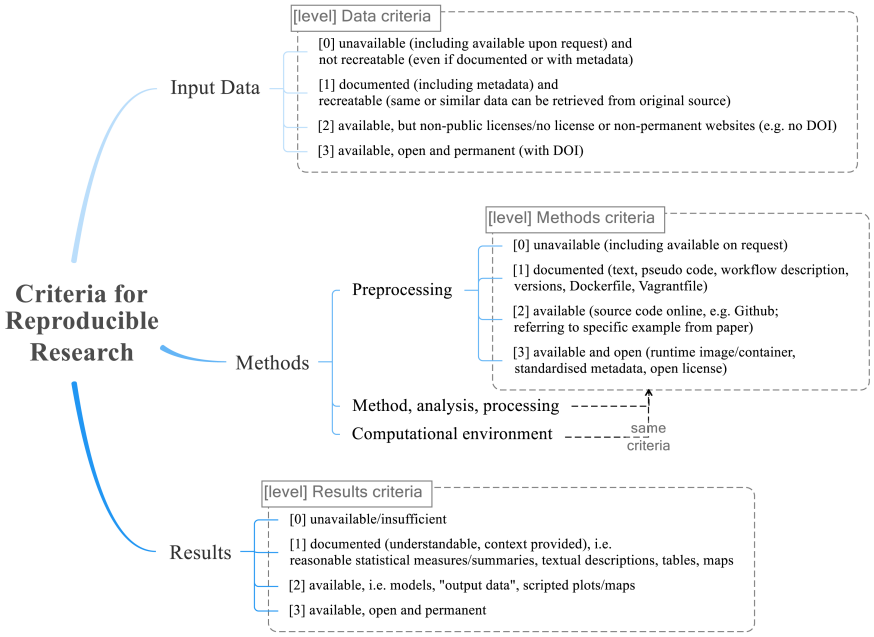


Figure A.1.: Reproducibility criteria to be assessed.

1. Input Data [2]
2. Preprocessing [3]
3. Methods [3]
4. Computational Environment [3]
5. Results [2]

All datasets and code will be freely available on the GitHub repository: `https://github.com/longxiangxu321/ProjectSolar` before the end of 2025. However, due to file size limitations, the experiment results are not suitable for online sharing.

A.2. Self-reflection

Reflecting on the process of completing this thesis, I have gained significant knowledge and skills in both research and engineering.

Throughout the thesis work, I developed experience in handling various types of data, such as 3DCM, point clouds, and arrays, across different formats and programming languages, including C++ and Python. This experience made me realize that completing a project or research requires not only a comprehensive set of skills and tools but also the ability to continuously learn and adapt to new ones.

Moreover, I learned the importance of maintaining focus on the overall research objectives. It is easy to become too focused in specific stages or implementations, losing sight of the main goals. I am grateful to my supervisors for regularly reminding me of the research purpose, which helped me stay on track.

Finally, resilience proved to be a crucial quality. Consistent effort is sometimes more important than shot-term efficiency. Despite the challenges I faced, I continued to work through them, solving each problem step by step.

I am thankful for choosing this topic and for the invaluable experiences I gained throughout the thesis process.

Bibliography

Agugiaro, G., González, F. G. G., and Cavallo, R. (2020). The city of tomorrow from . . . the data of today. *ISPRS International Journal of Geo-Information*, 9(9).

Allegrini, J., Dorer, V., and Carmeliet, J. (2012). Influence of the urban microclimate in street canyons on the energy demand for space cooling and heating of buildings. *Energy and Buildings*, 55:823–832.

Andres, C., Ruben, C., David, G., Pavel, B., Patrizio, M., Miro, Z., and Olindo, I. (2023). Time-varying, ray tracing irradiance simulation approach for photovoltaic systems in complex scenarios with decoupled geometry, optical properties and illumination conditions. *Progress in Photovoltaics: research and applications*, 31(2):134–148.

Arias-Rosales, A. and LeDuc, P. R. (2022). Shadow modeling in urban environments for solar harvesting devices with freely defined positions and orientations. *Renewable and Sustainable Energy Reviews*, 164:112522.

Bako, S., Meyer, M., DeRose, T., and Sen, P. (2019). Offline deep importance sampling for monte carlo path tracing. In *Computer Graphics Forum*, volume 38, pages 527–542. Wiley Online Library.

Biljecki, F., Ledoux, H., and Stoter, J. (2016). An improved lod specification for 3d building models. *Computers, environment and urban systems*, 59:25–37.

Biljecki, F., Stoter, J., Ledoux, H., Zlatanova, S., and Çöltekin, A. (2015). Applications of 3d city models: State of the art review. *ISPRS International Journal of Geo-Information*, 4(4):2842–2889.

Boccalatte, A., Fossa, M., Gaillard, L., and Menezo, C. (2020). Microclimate and urban morphology effects on building energy demand in different european cities. *Energy and Buildings*, 224:110129.

Calcabrini, A. (2023). Solar resource modelling and shading tolerant modules for the urban environment.

Contributors, W. (2024). Rendering equation — wikipedia, the free encyclopedia. Accessed: 2024-09-28.

Diao, L., Sun, Y., Chen, Z., and Chen, J. (2017). Modeling energy consumption in residential buildings: A bottom-up analysis based on occupant behavior pattern clustering and stochastic simulation. *Energy and Buildings*, 147:47–66.

Epic Games (2023). Global illumination in unreal engine. Accessed: 2024-05-31. https://dev.epicgames.com/documentation/en-us/unreal-engine/global-illumination-in-unreal-engine.

Erdélyi, R., Wang, Y., Guo, W., Hanna, E., and Colantuono, G. (2014). Three-dimensional solar radiation model (soram) and its application to 3-d urban planning. *Solar Energy*, 101:63–73.

Erell, E. and Zhou, B. (2022). The effect of increasing surface cover vegetation on urban microclimate and energy demand for building heating and cooling. *Building and Environment*, 213:108867.

Fu, P. and Rich, P. M. (1999). Design and implementation of the solar analyst: an arcview extension for modeling solar radiation at landscape scales. In *Proceedings of the nineteenth annual ESRI user conference*, volume 1, pages 1–31. San Diego USA.

Giannelli, D. (2021). Solar analysis on buildings of favelas in sao paulo to estimate pv potential.

Giannelli, D., León-Sánchez, C., and Agugiaro, G. (2022). Comparison and evaluation of different gis software tools to estimate solar irradiation. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4:275–282.

Gneiting, T., Lerch, S., and Schulz, B. (2023). Probabilistic solar forecasting: Benchmarks, post-processing, verification. *Solar Energy*, 252:72–80.

Gröger, G. and Plümer, L. (2012). Citygml–interoperable semantic 3d city models. *ISPRS Journal of Photogrammetry and Remote Sensing*, 71:12–33.

Group, P. (2024). Photovoltaic materials and devices. Accessed: 2024-09-19. https://www.tudelft.nl/en/ewi/over-de-faculteit/afdelingen/electrical-sustainable-energy/photovoltaic-materials-and-devices.

Hay, J. E. (1993). Calculating solar radiation for inclined surfaces: Practical approaches. *Renewable Energy*, 3(4):373–380. Solar radiation, environment and climate change.

Hegazy, M., Yasufuku, K., and Abe, H. (2021). Validating game engines as a quantitative daylighting simulation tool.

Heim, D. and Knera, D. (2021). A novel photometric method for the determination of reflected solar irradiance in the built environment. *Renewable and Sustainable Energy Reviews*, 137:110451.

Hurkmans, R. (2022). Efficient solar potential estimation of 3d buildings: 3d bag as use case.

Jaillot, V., Pedrinis, F., Servigne, S., and Gesquière, G. (2017). A generic approach for sunlight and shadow impact computation on large city models. In *25th International Conference on Computer Graphics, Visualization and Computer Vision 2017*, pages 10–pages.

Jensen, H. W., Christensen, P. H., Kato, T., and Suykens, F. (2002). A practical guide to global illumination using photon mapping. *SIGGRAPH 2002 Course Notes CD-ROM*.

Kamphuis, N., Gueymard, C., Holtzapple, M., Duggleby, A., and Annamalai, K. (2020). Perspectives on the origin, derivation, meaning, and significance of the isotropic sky model. *Solar Energy*, 201:8–12.

Klucher, T. (1979). Evaluation of models to predict insolation on tilted surfaces. *Solar Energy*, 23(2):111–114.

KNMI (2024). Knmi - about knmi. Accessed: 2024-09-18.

Komura, T. (2013). Global illumination (2): Monte-carlo ray tracing and photon mapping. Lecture slides, University of Edinburgh. Lecture 15, Computer Graphics course.

Kwok, K. and Hu, G. (2023). Wind energy system for buildings in an urban environment. *Journal of Wind Engineering and Industrial Aerodynamics*, 234:105349.

Liang, J. and Gong, J. (2017). A sparse voxel octree-based framework for computing solar radiation using 3d city models. *ISPRS international journal of geo-information*, 6(4):106.

Liang, J., Gong, J., Li, W., and Ibrahim, A. N. (2014). A visualization-oriented 3d method for efficient computation of urban solar radiation based on 3d–2d surface mapping. *International Journal of Geographical Information Science*, 28(4):780–798.

Liang, J., Gong, J., Xie, X., and Sun, J. (2020). Solar3d: An open-source tool for estimating solar radiation in urban environments. *ISPRS international journal of geo-information*, 9(9):524.

Liang, J., Gong, J., Zhou, J., Ibrahim, A. N., and Li, M. (2015). An open-source 3d solar radiation model integrated with a 3d geographic information system. *Environmental Modelling & Software*, 64:94–101.

Litjens, G. B., Kausika, B. B., Worrell, E., and van Sark, W. (2018). A spatio-temporal city-scale assessment of residential photovoltaic power integration scenarios. *Solar Energy*, 174:1185–1197.

Loutzenhiser, P. G., Manz, H., Felsmann, C., Strachan, P., Frank, T., and Maxwell, G. (2007). Empirical validation of models to compute solar irradiance on inclined surfaces for building energy simulation. *Solar Energy*, 81(2):254–267.

Marques, R. and Santos, L. P. (2010). Instant global illumination on the gpu using optix. In *Proc. INForum*, volume 10.

Mauree, D., Naboni, E., Coccolo, S., Perera, A. T. D., Nik, V. M., and Scartezzini, J.-L. (2019). A review of assessment methods for the urban environment and its energy sustainability to guarantee climate adaptation of future cities. *Renewable and Sustainable Energy Reviews*, 112:733–746.

Meines, S. (2023). Luminacity: a real-time daylight analysis tool for architectural and urban development using unreal engine.

Nielsen, A. H., Iosifidis, A., and Karstoft, H. (2021). Irradiancenet: Spatiotemporal deep learning model for satellite-derived solar irradiance short-term forecasting. *Solar Energy*, 228:659–669.

Nvidia (2024). Nvidia optix™ ray tracing engine. Accessed on 2024-09-18. https://developer.nvidia.com/rtx/ray-tracing/optix.

Oh, M. and Park, H.-D. (2018). A new algorithm using a pyramid dataset for calculating shadowing in solar potential mapping. *Renewable energy*, 126:465–474.

Ohori, K. A., Ledoux, H., and Peters, R. (2024). *3D Modelling of the Built Environment*. Ken Arroyo Ohori, Hugo Ledoux, and Ravi Peters, v0.9 edition. Available under a Creative Commons Attribution 4.0 International License. Downloadable from https://github.com/tudelft3d/3dbook/releases.

Oke, T. R. (1982). The energetic basis of the urban heat island. *Quarterly journal of the royal meteorological society*, 108(455):1–24.

Parker, S. et al. (2018). Nvidia optix™: Ray tracing powered by rtx. Accessed: 2024-09-29. https://developer.nvidia.com/blog/nvidia-optix-ray-tracing-powered-rtx/.

Perez, R., Ineichen, P., Seals, R., Michalsky, J., and Stewart, R. (1990). Modeling daylight availability and irradiance components from direct and global irradiance. *Solar energy*, 44(5):271–289.

Perez, R., Seals, R., Ineichen, P., Stewart, R., and Menicucci, D. (1987). A new simplified version of the perez diffuse irradiance model for tilted surfaces. *Solar energy*, 39(3):221–231.

Peters, R., Dukai, B., Vitalis, S., van Liempt, J., and Stoter, J. (2022). Automated 3d reconstruction of lod2 and lod1 models for all 10 million buildings of the netherlands.

Piórkowski, R. and Mantiuk, R. (2015). Using full reference image quality metrics to detect game engine artefacts. In *Proceedings of the ACM SIGGRAPH Symposium on Applied Perception*, pages 83–90.

Piórkowski, R., Mantiuk, R., and Siekawa, A. (2017). Automatic detection of game engine artifacts using full reference image quality metrics. *ACM Transactions on Applied Perception (TAP)*, 14(3):1–17.

PostGIS (2024). St_3dintersects. Accessed: 2024-09-19. https://postgis.net/docs/ST_3DIntersects.html.

Reindl, D., Beckman, W., and Duffie, J. (1990a). Diffuse fraction correlations. *Solar Energy*, 45(1):1–7.

Reindl, D., Beckman, W., and Duffie, J. (1990b). Evaluation of hourly tilted surface radiation models. *Solar Energy*, 45(1):9–17.

Rodríguez, L. R., Duminil, E., Ramos, J. S., and Eicker, U. (2017). Assessment of the photovoltaic potential at urban level based on 3d city models: A case study and new methodological approach. *Solar Energy*, 146:264–275.

Shaik, F., Lingala, S. S., and Veeraboina, P. (2023). Effect of various parameters on the performance of solar pv power plant: a review and the experimental study. *Sustainable Energy Research*, 10(1):6.

Song, Z., Liu, J., and Yang, H. (2021). Air pollution and soiling implications for solar photovoltaic power generation: A comprehensive review. *Applied Energy*, 298:117247.

Staffell, I., Pfenninger, S., and Johnson, N. (2023). A global model of hourly space heating and cooling demand at multiple spatial scales. *Nature Energy*, 8(12):1328–1344.

Statistics Netherlands (2023). Population; key figures. https://www.cbs.nl/en-gb/figures/detail/81528ENG#shortTableDescription. Accessed: 2024-10-04.

Stendardo, N., Desthieux, G., Abdennadher, N., and Gallinelli, P. (2020). Gpu-enabled shadow casting for solar potential estimation in large urban areas. application to the solar cadaster of greater geneva. *Applied Sciences*, 10(15):5361.

Tabik, S., Villegas, A., Zapata, E. L., and Romero, L. F. (2013). Optimal tilt and orientation maps: a multi-algorithm approach for heterogeneous multicore-gpu systems. *The Journal of Supercomputing*, 66:135–147.

Tercha, W., Tadjer, S. A., Chekired, F., and Canale, L. (2024). Machine learning-based forecasting of temperature and solar irradiance for photovoltaic systems. *Energies*, 17(5):1124.

Uber Technologies, I. (2024). H3: A hexagonal hierarchical geospatial indexing system. https://h3geo.org/. Accessed: 2024-10-21.

Unity Technologies (2023). Global illumination introduction. Accessed: 2024-05-31.

Walter, E. and Kämpf, J. H. (2015). A verification of citysim results using the bestest and monitored consumption values. In *Proceedings of the 2nd Building Simulation Applications conference*, pages 215–222. Bozen-Bolzano University Press.

Wang, C.-K., Tindemans, S., Miller, C., Agugiaro, G., and Stoter, J. (2020). Bayesian calibration at the urban scale: a case study on a large residential heating demand application in amsterdam. *Journal of Building Performance Simulation*, 13(3):347–361.

Wang, X., Zhang, X., Zhu, S., Ren, J., Causone, F., Ye, Y., Jin, X., Zhou, X., and Shi, X. (2023). A novel and efficient method for calculating beam shadows on exterior surfaces of buildings in dense urban contexts. *Building and Environment*, 229:109937.

Ward, G. J. (1994). The radiance lighting simulation and rendering system. In *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, pages 459–472.

Wieland, M., Nichersu, A., Murshed, S. M., and Wendel, J. (2015). Computing solar radiation on citygml building data. In *18th AGILE international conference on geographic informaton science*.

World Bank (2023). World development indicators. Accessed: 2024-08-22.

Xu, L., León-Sánchez, C., Agugiaro, G., and Stoter, J. (2023). Shadowing calculation on urban areas from semantic 3d city models. In *International 3D GeoInfo Conference*, pages 31–47. Springer.

Xu, L., León-Sánchez, C., Agugiaro, G., and Stoter, J. (2024). High resolution solar potential computation in large scale urban areas by means of semantic 3d city models. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, XLVIII-4/W11-2024:167–174.

Colophon

This document was typeset using LaTeX, using the KOMA-Script class `scrbook`. The main font is Palatino.