Opto-Geometric Modelling of Complex Urban Landscapes

Karthik Ganapathi Subramanian

OPTO-GEOMETRIC MODELLING OF COMPLEX URBAN LANDSCAPES

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

Master of Science in Sustainable Energy Technology

by

Karthik Ganapathi Subramanian

July 2021

Delft University of Technology

Karthik Ganapathi Subramanian: *Opto-geometric Modelling of Complex Urban Landscapes* (2021)

An electronic version of this thesis is available at https://repository.tudelft.nl.

The work in this thesis was made in the:

PhotoVoltaic Materials & Devices group
Department of Electrical Sustainable Energy
Faculty of Electrical Engineering, Mathematics and Computer Sciences
Delft University of Technology

Supervisors: Prof.dr. Olindo Isabella
Dr. ir. Rudi Santbergen
Dr. ir. Pirouz Nourian

ABSTRACT

The injection of renewable energy into the grid requires meticulous planning and execution due to the unpredictable nature of weather and in-turn the energy generated by these sources. Moreover, the inclusion of grid-independent energy generators requires accurate tools to predict energy yield in order to simplify matters such as grid-congestion management. One such tool useful to compute solar energy yield, which is under development, is the PVMD toolbox from the PhotoVoltaic Devices and Materials research group within the Delft University of Technology. This toolbox, which is completely based on MATLAB®, allows users to specify parameters ranging from the micro-scale (such as the chemistry and characteristics of a single cell) to the macro-scale (such as the electrical parameters of the whole system). However, in its present version, the toolbox can simulate the yield only for simple systems, such as a single PV panel without any obstructions present nearby. This makes it incompatible to solve complex problems such as models with solar panels installed in the built environment.

Therefore, the primary focus of this thesis is on the optical side of the PV simulation, by enabling the toolbox to precisely compute reflected component of irradiance and include shading effects on modules from nearby objects in the scene. This functionality will help consider the presence of features such as dormers, chimneys and highly reflective surfaces present in urban environments which contribute to solar power output. To achieve this, attention was paid towards the formats and sources of geometric data (3D objects in particular) which could be used as an input in the toolbox. Once the data format was selected, the 3D object is parsed into a MATLAB readable format and refined in order to preserve geometric information. Afterwards, the scene was completed by enumerating the geometric data of the solar cells and placing them beside the object or on an external surface of the object. Optical properties were then obtained in the form of spectral reflectance data from NASA' ECOSTRESS Library and converted into the appropriate format to assign these material properties to the geometries. This Opto-geometric data is then parsed back into a file which can be read by RADIANCE, a visual rendering software capable of backward ray-tracing. This ray-tracing technique is used to obtain the irradiance available for each solar cell at a given time instant.

The developed workflow is validated by comparing it with real data stored at the PV Monitoring Station within the TU Delft Campus. For the given data-set, the mean percentage error is computed to be 1.89% , with a standard deviation of 12.49%. This proves that the proposed workflow is suitable to visualise shading performance and the effect of reflected irradiance. This model can be further improved by considering parameters such as spectrally responsive albedo and inclusion of other geometry formats. The outputs from this project can also be used as inputs to compute the spectral absorptance of different layers within a solar cell, such as tandem solar cells or develop sensitivity maps for different modules present in the scene.

ACKNOWLEDGEMENTS

This thesis would not have been a reality without the help and support of so many people. Firstly, I would like to thank my supervisors Dr. Rudi Santbergen and Dr. Pirouz Nourian for giving me the chance to work on this project. I am very grateful to have such amazing professors guiding me throughout the duration of the thesis and keep me focused on the big picture.

I am indebted to my parents and my family, who were my pillars of strength during the entirety of my masters programme. I am also very blessed to have such amazing friends who supported me mentally and provided great inputs to progress in my thesis during the tough times. A special thanks to Dhanvin, Aditya, Swami and Rahul for being there for me when the going got tough.

A special shout-out goes to Andres Calcabrini from the PVMD group, who was willing to entertain even the most trivial of queries and provide as much help as possible. Without his help and feedback, I would never have obtained such a holistic view of the problem and I am very thankful for that. I would also like to thank Maarten Verkou, who helped me during the early stages of the project with data parsing and file conversions. Additionally, I am grateful to all the professors part of the PVMD group for helping me with data procurement, giving general feedback throughout the project and making me feel at home.

Finally, I would like to thank God for blessing me through this adventure and giving me the strength to overcome all the problems I have during the project.

Karthik
05th of July 2021

CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

dem	DEM Digital Elevation Model
cad	CAD Computer Aided Design
pv	PV Photovoltaics
pvmd	PVMD PhotoVoltaic Materials & Devices
lidar	LiDAR Light Detection and Ranging
gis	GIS Geographic Information System
dsm	DSM Digital Surface Model
esra	ESRA European Solar Radiation Atlas
dtm	DTM Digital Terrain Model
bim	BIM Building Information Model
bipv	BIPV Building Integrated Photovoltaics
lod	LoD Level of Detail
nrel	NREL National Renewable Energy Laboratory
insel	INSEL Integrated Simulation Environment Language
gps	GPS Global Positioning System
sebe	SEBE Solar Energy on Building Envelopes
cams	CAMS Copernicus Atmosphere Monitoring Service
brdf	BRDF Bidirectional Reflectance Distribution Function
ffa	FFA Far Field Albedo

nfr	NFR Near Field Reflectance
eqe	EQE External Quantum Efficiency
cnn	CNN Convolutional Neural Network
ogc	OGC Open Geospatial Consortium
nm	nm nanometer
uv	UV Ultraviolet
ir	IR Infrared
dhi	DHI Diffuse Horizontal Irradiance
dni	DNI Direct Normal Irradiance
poa	POA Plane of Array
csg	CSG Constructive Solid Geometry
pert	PERT Program Evaluation Review Technique

1 INTRODUCTION

1.1 MOTIVATION

PV panels have become a common sight in today's world, with a tremendous rise observed in the amount of PV panels installed in urban environments. Thanks to advances in computers and their computational capabilities, the modelling of PV panel performance has become a very trivial task today. However, this modelling has to be extremely precise to ensure that the solar panels operate within the nominal range on installation and not turn out to be a liability for the user.

The Plane of Array Irradiance on Solar panels can be divided into three components: direct, diffuse and reflected irradiance. Direct irradiance is the energy incident on the panels directly from the sun. Diffuse irradiance accounts for light scattered due to particles suspended in the air or objects present near the panels, while reflected irradiance accounts for light reflected from nearby objects. So far, previous work in determining the PV potential in urban environments, (as will be explained in detail in Chapter 2) neglect the contribution of the reflected component of irradiance towards the available amount of solar energy for electricity production. In most cases, this component of irradiance has been overlooked completely and in others, this effect has been included by assuming a constant value of reflectance for surrounding surfaces and objects. Although these assumptions might not affect the results for a simulation at the level of a city/district as observed in previous research, the contribution of the reflected component cannot be neglected for a PV (pv) panel installed in a small urban scene with just a few buildings and vegetation surrounding it. It has been found that the yield for bifacial modules can vary between 10% to 35% when considering the reflected component of irradiance accurately Vogt et al. [2018]. One such software package which can be used to model small scenes is the PVMD Toolbox, developed by the PVMD (pvmd) group, which allows users to predict the yield of a PV system with precision. However, in its present version, it is very cumbersome for users to model even simple geometries as users have to manually input the Cartesian coordinates of each vertex defining an object. Therefore, this research is aimed at incorporating a workflow in the toolbox which is user-friendly with complex geometries and can accurately model reflected irradiance.

1.2 SCOPE

As mentioned earlier, the scope of this research project is to devise a method through which reflected irradiance and shading performance can be modelled in the toolbox due to objects present beside solar panels, similar to the ones encountered in urban scenes. It should not be confused to solely be a computer science project where only a workflow is developed to model PV potential in urban environments. This project involves the following scientific disciplines as well: geo-information science, material science and concepts involving ray-optics. It can be depicted in the form of a Venn Diagram as shown in Figure 1.1 , with the Centre of the Venn - Diagram representing the "big picture" of this thesis.

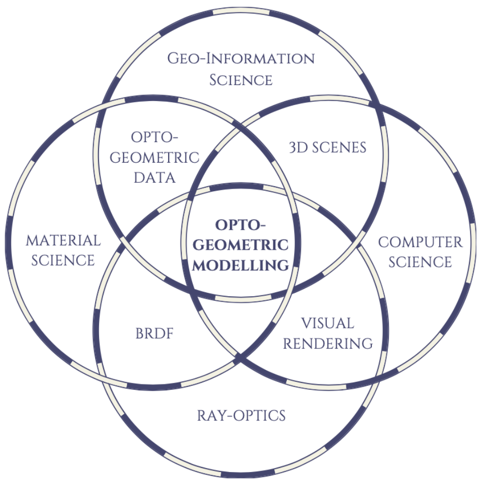


Figure 1.1: Venn Diagram representation of the Project.

In order to visualise reflected irradiance and shading, the help of a software called RADIANCE will be used. This is a visual rendering package which uses advanced techniques such as ray-tracing to accurately render objects. This is done with the help of a parameter named the Bi-directional Reflectance Distribution BRDF (brdf), which contains information on how light gets reflected from a surface when it is incident at a certain angle. The concept of BRDF has been explained in detail in chapter 2 of this report. Long story short, with the help of this toolbox, a user will be able to determine how objects surrounding a panel can affect its yield, and such examples have been provided in chapter 7 of this thesis.

1.3 LITERATURE

The sources of literature that were used as a starting point for this research were the ones which described the methods used to estimate PV potential in urban environments. This included the workflows to input geo-spatial information of urban landscapes and various software that were used to visualise this geo-spatial information to generate 3D models. The sources of obtaining geo-spatial data could be from LiDAR (lidar) point clouds, user-generated 3D geometries on CAD softwares or images obtained aerially and from satellites. There are also multiple open source data-sets for obtaining the spectral reflectance of materials such as NASA's ECOSTRESS library, but no data-set exists which have combined opto-geometric data for the modelling. Therefore, most workflows assumed constant values of reflectance in their simulations or neglected it entirely. These workflows which have been adopted in previous research have been explained in detail in Chapter 2.

Although the omission of reflectance behaviour does not affect the results on large-scale models at the level of neighbourhoods/cities, it is crucial in modelling smaller scenes where cutting-edge precision is desirable. Thus, the search was directed to determine the effect of reflected irradiance on the results of the solar modelling and how this effect has been introduced in solar simulations before. The reflected component is of major importance in bifacial PV panel modelling and thus previous work in this field was referred to for this purpose. This aided in formulating a research gap found through the literature review.

1.4 PROBLEM STATEMENT

The ”Big Picture” for this project is to address the following question:

What is the most user-friendly way of modelling PV potential in a small urban scene, without compromising on accuracy?

In order to do so, the following sub-problems will be addressed via this research:

- Which file format is the most suitable for inputting 3D geometries and optical properties of materials into the PVMD Toolbox?
- How can the PV panels be incorporated into the 3D scene?
- How is Opto-geometric data obtained from the geometric and optical data?
- How is Opto-geometric data used to calculate the irradiance available for each solar cell present in the simulation?
- How is the developed computational workflow validated?

1.5 OBJECTIVES

Based on the research questions elucidated in the previous section, the objectives of this thesis can be summarised as:

- Review existing methods and previous research to determine shortcomings
- Develop a computational workflow in MATLAB using which users can input 3D geometries of urban scenes available in open source softwares such as 3dwarehouse and parse this data into the MATLAB format
- Refine the loaded 3D geometry to eliminate errors. Complete the scene by generating and placing panel geometry in it.
- Create a material library and use this to assign the optical properties to the geometries. Run the ray-tracing simulation by converting the opto-geometric data into a compatible format and incorporate a brdf for the simulations.
- Use data available from the PVMD monitoring station present within TU Delft to validate the model.
- Present a Case Study to explore the new functions of the PVMD toolbox

1.6 METHODOLOGY AND PLANNING

The thesis project starts with the intention of gathering as much information as possible about 3D geometries, available sources and their file formats. This is used to select an appropriate file which can be included in the toolbox and develop a workflow around it, with the aim of maximising user ease. Once this file is procured, it is parsed into the MATLAB format and processed in order to preserve the quality of the model. Simultaneously, spectral reflectance data for various materials is obtained from NASA’s ECOSTRESS library and imported into the toolbox, which is then used to assign the optical properties to the various surfaces defining the scene.

Once the geometric and optical data is loaded into the toolbox, the scene is completed by adding the solar PV panels to the scene. The workflow allows the possibility of placing panels beside the 3D object or on a surface of the same. The optical properties are then assigned to the surfaces, with optical data for the solar cells obtained from the Toolbox's GenPRO simulations. GenPRO is an optical model which produces absorptance data for the different layers of a solar cell as a function of wavelength and angle of incidence of light. This, along with the geometric data and optical properties of the surfaces are converted into a file format suitable for carrying out the ray-tracing simulations. The Ray-tracing simulation is performed in RADIANCE, a visual rendering tool which is capable of using advanced concepts such as a BRDF to ray-trace and compute irradiance values available at the required points on a solar cell. The BRDF is a data-set which contains the reflectivity properties of a surface as a function of the wavelength and angle of incidence. The software in itself uses a backward-ray tracing algorithm, where the rays travel back from the sensor to the source of light to compute the irradiance on a solar cell.

The processes involved to achieve milestones within the project and the respective sub-outputs have been presented in the form of a PERT (pert) chart, as shown in Figure 1.2.

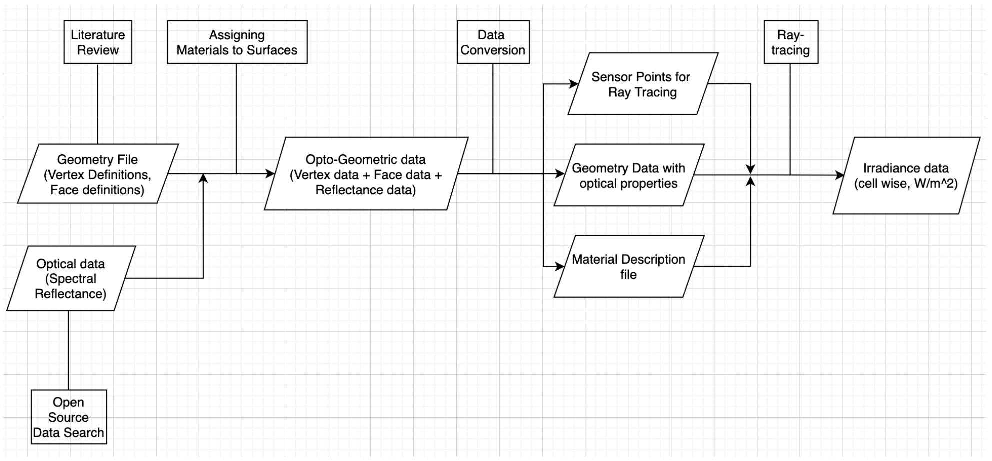

Literature Review
Assigning Materials to Surfaces
Data Conversion
Ray-tracing
Sensor Points for Ray Tracing
Geometry File (Vertex Definitions, Face definitions)
Opto-Geometric data (Vertex data + Face data + Reflectance data)
Geometry Data with optical properties
Irradiance data (cell wise, W/m^2)
Optical data (Spectral Reflectance)
Material Description file
Open Source Data Search

Figure 1.2: PERT Chart Representation of the Project Plan.

2 PREVIOUS WORK IN MODELLING PV POTENTIAL IN URBAN ENVIRONMENTS

The first objective of this project is to review existing methods and previous research to determine the shortcomings from those models. Predicting the yield of solar panels accurately is the need of the hour, as the results can be used in a variety of fields ranging from urban planning to future renewable energy system management. To achieve this, it is essential to model all the physical attributes which contribute to the yield such as shading effects and reflected irradiance. This in-turn requires combining different geometries and surfaces with their associated optical properties to precisely predict solar performance. This chapter will therefore explore the different workflows and softwares adopted in previous work used to model solar potentials in urban environments and the different file formats used for storing and interpreting geometric and optical data.

This chapter is divided as follows: section 2.1 will highlight the data sources, modelling methodologies and file formats used for storing geometries. Section 2.2 will discuss the approaches and the importance of the reflected component of irradiance and the chapter is concluded with the section 2.3

2.1 WORKFLOWS AND FILE FORMATS USED FOR GEOMETRIC DATA

For reproducing 3D models of localities and calculating solar potentials , there are multiple workflows and numerical methods which could be used. The data input or in other words, the type of data used as an input for the modelling procedure dictates the workflow to be followed. The three main inputs for generating 3D models are LiDAR point clouds, user generated or open source 3D models which can be directly processed, or photogrammetric techniques used to obtain images of the case study location, for example aerially or via satellites.

These geometric files are then used as input in different software packages for carrying out the solar simulations. Most LiDAR point clouds are processed in GIS based software to obtain the 3D model, while images are processed in multiple steps using various image processing software to obtain the required data. 3D models of objects, such as the ones available from Openstreetmap, Google Earth's 3DWarehouse can also be used to directly obtain the 3D models of an urban landscape, alongside the use of 3D CAD modelling software for defining user based landscapes.

Once the 3D model is obtained, the solar simulations can be modelled in GIS based softwares by including plugins to the tool, or custom tools can be designed for the same purpose as well. Other techniques, such as the use of a physically-based rendering software (eg: RADIANCE) can also be used for the simulations. Most workflows for solar potential involve the use of one or many of these softwares. The use of these multiple approaches was reviewed by Freitas et al. [2015] in detail, along with extensive descriptions of numerical models used in previous work and the various software packages available to perform detailed simulations. Most of the aforementioned packages can be classified in terms of the geometric modelling into web-based, GIS (gis) based or a 3D model based software. In these softwares, various plugins can be added to perform additional functions such as shadow or reflected irradiance modelling. Similarly, Jakica [2018] also reviewed existing solar design tools in detail, with the study conducted for over 200 design tools and combinations of these various tools used in modelling solar potential. In this section, an in-depth analysis of these various workflows classified based on the nature of input data is provided, along with the file formats used for storing these geometries. Subsection 2.1.1 will review previous research were LiDAR data was used as input, subsection 2.1.2 which discusses those works where images were used to obtain the 3D model. This is followed by subsection 2.1.3, where the use of 3D modelling softwares and open source 3D building libraries is highlighted. Subsection 2.1.4 reviews other approaches which have been adopted and the section is summarised with subsection 2.1.5.

2.1.1 LiDAR based data input

The use of LiDAR point clouds is one of the most common techniques used for generating 3D models. LiDAR data points are obtained in the .las format and triangulated using various algorithms such as the Delaunay algorithm to obtain a mesh model of the 3D objects, which is called a DEM (dem). This DEM can be later used to generate a DSM (dsm) and/or a DTM (dtm) based on modelling requirements, which are the 3D representations of the environment. The processing and simulation for solar potential can be achieved using custom algorithms, commercially available software or a combination of both.

LiDAR point clouds are visualised commonly using GIS software packages such as ArcGIS or GRASS GIS. The addition of plugins to these tools make it easier to visualise the results. M. Martín et al. [2015] assessed various LiDAR data and GIS based approaches for solar potential modelling that are available, and information on softwares such as ArcGIS, GRASS and their respective add-on tools were described in detail. In some instances however, the use of a GIS based tool was to perform very specific tasks such as obtaining building footprints or refining the LiDAR data that has been used for the simulations. Other tools such as RADIANCE were also used alongside these GIS based tools for the solar potential modelling.

Verso et al. [2015] came up with a GIS based PV potential model through ArcGIS, and used the software to generate a DSM from LiDAR data processed in *VRMesh* software. By classifying the roofs of buildings as sloped or flat, a geometric solar irradiance model was designed using meteostat measurements as inputs. The *hillshade* algorithm was used for modelling shadows of nearby objects on roofs. Similarly, Jochem et al. [2009] in their research used LiDAR data to produce a DSM to determine the solar potential of roofs in a small urban settlement in Austria. The novelty in the work was that they introduced shadowing effects from nearby vegetation by categorising the LiDAR data as vegetation, roofs and building surfaces. A comparable work was also carried out by Fogl and Moudrý [2016], where LiDAR data and building footprint data was used to generate a DSM on the ArcGIS platform, while an extra software called LAStools was used to distinguish between vegetation and buildings. Strzalka et al. [2012] in their work used LiDAR data as inputs for obtaining the 3D model, along with the map of the analysed location in the .dxf format. This data was then processed in a GIS software to obtain the roofs which can be used for installing PV panels. The PV simulations were carried out using the *INSEL8* software, and shadowing was used to only determine those roofs were the panels would work with least shading.

In the work done by Brito et al. [2012], a plugin available with ArcGIS called Solar Analyst tool was used for the solar simulations of the 3D model. The models were generated by using LiDAR point clouds and aerial photography to obtain a DTM and a DSM of a suburban site in Lisbon. Similarly, Santos et al. [2014] in their work for solar potential calculations in Lisbon used LiDAR data in conjunction with digital cartography (in the vector data format) to obtain the 3d model and took help of the Solar Analyst tool in ArcGIS to perform the simulations. On the contrary, a building typology approach was chosen by Xu et al. [2019] where buildings were classified as residential, industrial or commercial and roof types were defined on top of LiDAR data to determine possible roofs for solar analysis. Again, these definitions were used to model the solar potential in Wuhan, China by using the ArcGIS platform. This classification approach was also used by Jacques et al. [2014] for determining the solar potential over large areas using low resolution LiDAR data and aerial images. They classified the roofs into observed shapes based on the aerial images for easier roof detection.The point cloud generated by the LiDAR assisted in the detection of roofs, their tilts and orientations and was processed in a text based format.

Meanwhile, Nguyen and Pearce [2012] in their research used multiple softwares for modelling PV potential in Kingston, Ontario from LiDAR data. Their main innovation was in designing a new shadowing algorithm for use in the open source GRASS GIS software, with the .asc file format recommended for grid files on GRASS. The ArcGIS software was used for obtaining the DSM of the site, while GRASS was utilised for the irradiance modelling and Scilab was used instead of MATLAB for the data processing. The new shading algorithm works similar to the *Hillshade* function available on ArcGIS. The albedo components are not given importance in this model however, and a constant reflectance value of 0.2 was assumed for all surfaces. On the other hand, Jakubiec and Reinhart [2013] in their study wanted to create solar potential maps at the scale of cities and used sampled LiDAR data and available GIS data to generate 3D models. Mesh models of these 3D geometries was generated by applying the Delaunay algorithm and the solar irradiance was modelled using the RADIANCE software's backward ray tracer tool. The authors in their study assumed that all surfaces are Lambertian scatterers, with 35% reflectivity for building walls and 20% for the landscape. For roofs, building material data was used to assign the reflectance values.

Kodysh et al. [2013] in their work used the upward looking hemispherical viewshed algorithm, in which a upward looking camera was used to obtain the 360°picture of the site to determine obstructions in the skyline. This data was used alongside a LiDAR generated DEM in a raster-based file format on the ArcGIS platform to determine the effect of shading on solar potential calculations for multiple rooftops in Knox County, Tennessee. Salimzadeh and Hammad [2017] also used LiDAR data like the previous works explained so far, but also included additional information of the buildings surfaces while reconstructing the 3D model, such as a BIM (bim). The input data was taken in the .fbx format from CityGML and using an alternate software the dataset was converted to .sat format. This file format can be exported to the Revit Solar Analyst plugin in the 3D software Autodesk for modelling the solar potential. The advantage of using a BIM to reconstruct the 3D model is that it can improve the results of modelling vertical facades for BIPV (bipv) applications.

Another workflow which can be adopted without taking the help of GIS platforms is to create custom made algorithms. Redweik et al. [2013] in their work modelled the potential of facades in aiding PV generation by defining their own algorithm called SOL, where the algorithm itself is capable of generating a DSM from a LiDAR point cloud. The algorithm also included modelling of shadows from nearby buildings, objects and facades to model the PV potential accurately and included solar radiation data from the SOLTERM database. Another custom algorithm was designed by Lukač et al. [2013], where LiDAR data was referenced with GPS (gps) data for obtaining geo-spatial information. The inputs for the solar irradiation were taken from pyranometer measurements at the site and included shading from buildings and vegetation in their model. This work was a followup on the previous progress achieved by the authors in Lukač and Žalik [2013]'s work, where the CUDA technology in NVIDIA's GPU was used for a GPU based PV modelling program. The model incorporated the solar position calculator designed by NREL (nrel), along with the shading algorithm mentioned earlier for solar simulations. A LiDAR point cloud was again obtained as input and processed to classify buildings, terrain and vegetation.

Design of custom algorithms for pv potential analysis was also noticed in the model developed by Lindberg et al. [2015], where a DSM was used for solar simulations based on the Perez model. Called SEBE (sebe), it was completely executed on MATLAB and has the capability to also visualise vegetation to account for shadows. The use of SEBE was continued in the PV potential assessment for Dar es Salaam in Tanzania by Lau et al. [2017], where a DSM was produced using LiDAR data and building footprints obtained from satellite images taken from WorldView-2. A laser rangefinder from Nikon®was used for obtaining the roof's vertices in the model. The input for the solar radiation model was taken from Meteonorm-7. The presence of shadows was considered using a "shadow volumes" method described by Ratti et al. [1999].

A slightly different approach was chosen by Carneiro et al. [2009] however, where the researchers used raw LiDAR data and orthophotos to generate the 3D Urban model, with the help of the CIBERCITY enterprise. A DSM was thus generated with the help of a DTM and the height data provided by the LiDAR input, stored in a bitmap format. Image processing techniques were also used to eliminate the slopes observed in vertical surfaces to make them truly vertical. These slopes were a result of the interpolation technique used in the DTM. A segmentation technique was then employed to determine the slopes and orientations of roofs. With all this information, solar irradiance calculations were performed on the model in MATLAB with input data for the solar calculations obtained from Meteonorm. However, the albedo component was neglected in the entire study.

Up until this point, a review of papers where LiDAR was used as a starting point has been provided. It can be seen that LiDAR data points are alone not sufficient and to generate the entire 3D scene, other datasets such as building footprints, obtained in the .shp file format are essential to complete the model. Moreover, this requires the intervention of other software packages such as a GIS software or Blender to visualise this data and this is cumbersome to obtain a 3D model. Therefore the use of LiDAR, although a very interesting procedure has to be further simplified in order to improve user friendliness.

2.1.2 Photogrammetry Techniques

Another form of input data for obtaining 3D city models without relying on LiDAR data can be through the use of aerial imagery or photogrammetry techniques. The biggest advantage of using images is the fact that dependence on the availability of public geo-spatial data is greatly reduced. Compagnon [2004] in his work determined the solar potential in a small neighbourhood in Fribourg, Switzerland by using image processing techniques on aerial images to produce a DEM of the locality. This data was used to generate the 3D model, which was simulated in the RADIANCE lighting software for determining the solar potential. In this work, the reflectance of all surfaces was considered to be constant with a value of 0.2. All data files were stored and read in the ASCII format and transformed to the radiance format for the simulations . A very closely related work was carried out by Ronzino et al. [2015], where traditional photogrammetry methods were used as the first step in generating the 3D model by the use of a BIM. LiDAR data was used only to enrich the model generated from the image processing, but not as a direct input. The capturing of the images was done manually on-site in this study and therefore can be time consuming if conducted for large areas. The BIM data was transferred in the .ifc or gbxml formats.The 3D model was then used in the GRASS GIS environment to simulate the solar potential in the location of interest.

A fully automated JavaScript model for solar potential assessment was coded by Mainzer et al. [2017] which is intended to reduce the need for big data to conduct such analyses. The code only requires images for generating the urban scene. Using OpenStreetMap and Microsoft Bing Maps for generating the footprints and roof data, it relied on the assumption that human beings can examine available roof area to a certain extent based on images. Image processing techniques were used to enrich the images and obtain roof shapes and areas. The tilts and orientations were estimated based on a normal distribution function with a mean of 37°and a standard deviation of 15°. A contour detection and polygon approximation method was used to compute usable roof areas, and *Algorithm 3* provided from Grena [2012] was used for the solar simulations. This solar input data for the model was obtained from CAMS (cams). The image processing algorithm directly took into account the effects of shadows in the model. Likewise, Hofierka and Kaňuk [2009] used open source tools in their work for modelling solar potential in a urban areas, with the case study being the city Bardejov in Slovakia. The softwares ArcView GIS and GRASS GIS were used for obtaining the 3D city model, with the help of orthophoto maps and topographic maps to generate a DEM in a raster based format. The solar irradiance was modelled using the *r.sun* module of GRASS GIS and *r.shadow* module was used for modelling the effect of shadows on PV potential.

The use of image processing for obtaining urban models specific to PV potential analysis has not been found commonly in literature. However, the use of image processing for building detection has been found in literature. Miyazaki et al. [2016] in their work propose the use of a CNN (cnn) to process images for generating high resolution models. The input images were again obtained from Microsoft Bing maps. The use of high-resolution satellite images for building detection was also found in Roopa et al. [2018], where buildings are detected using a segmentation and grouping technique developed on MATLAB. On the contrary, the work of Aamir et al. [2019] uses low-quality and contrast satellite images to detect building contours. Similar to the previous work, segmentation techniques were used after enhancing the quality of the images.

It can observed from the review of previous work that the use of image processing techniques has tremendous potential in modelling PV potential, but involves various processing steps to obtain high quality models. Although it is useful when geo-spatial data is unavailable, the sheer effort required to acquire 2D or 3D geometric data outweighs its advantages and therefore should considered as a back-up option if other sources are not sufficient to achieve results.

2.1.3 3D Modelling Software and 3D Libraries

The easiest and the most practical way to obtain the 3D model of urban landscapes is to access open source libraries for existing models or generating the landscape directly from scratch This method wherein the models are user defined or user generated is called the "Bottom Up" approach. This method includes the use of 3D CAD software to produce the 3D model of the buildings and vegetation. Whenever modelling buildings using CAD softwares, the LoD (lod) is an important parameter that has to be decided for accurate modelling. These levels of detail have been defined by the OGC (ogc) and varying from LoD0 to LoD4, each LoD represents certain features of buildings in greater detail. For instance, LoD0 represents building footprints while LoD4 includes all features, including information about the interior surfaces. As the research reviewed thus far only model PV potentials of roofs in urban environments, no more than LoD2 is necessary for the models. However, while modelling facades or the influence of vertical surfaces, greater LoDs are required. According to Biljecki et al. [2016b] however, these LoDs are not sufficient and greater subdivisions are required for defining building features, as the current system can be vague for specific cases. Their work introduced additional classes within the current system, such as LoD2.2 where each class took into account geometric features such as chimneys or dormers.

A model with greater detail was designed by Catita et al. [2014], where the team modelled the buildings using 3D software SketchUP for computing the potential of vertical facades in PV generation. This model was refined to match real world dimensions by using data available from a CityGML plugin, and data was exported in the COLLADA format. They were also able to triangulate surfaces such that doors, windows and openings on facades could be modelled and achieved a LoD3. This work was a followup to Redweik's work explained in subsection 2.1.1. Melo et al. [2013] presented an extension to the Google SketchUP environment called Solar3DBR, which can be used to accurately model shading factors in solar irradiance calculations. This was implemented by generating a shading matrix, which takes into account the diffuse and direct shading factors caused due to objects in the horizon.

An economic potential study of installing PV panels in the city of Karlsruhe, Germany was executed by Fath et al. [2015] where 3D models of LoD2 were used for modelling the city. In their case, the 3D city model was directly provided to them by the municipal authority of the city, in the .dwg format. This model was processed to reduce the computational time on ArcGIS software and then transferred to RADIANCE in a text-file based format for irradiance computations. In this simulation, all facade surfaces were assumed to be Lambertian scatterers, with a constant reflectance of 30% and the roofs and ground were assumed to have a reflectance of 20%. Another research work were 3D models were directly obtained was by Robledo et al. [2019], where the open source tool 3D Warehouse was used to obtain the 3D scene, which is also used for obtaining 3D building models from Google Earth®in the COLLADA format. In Robledo's research, the PV panels were arranged at the appropriate location in the scene and simulated for PV potential using meteorological data provided by the Weather station in Denver, in TMY3 format. The new feature in their study was the use of GPU based program to model their shading algorithm, similar to Lukac's work explained earlier in subsection 2.1.1.

The biggest advantage of using 3D CAD software is that it allows users to not only plan urban environments with ease, but also work in creating existing landscapes with minimum information. For instance, if the clustering pattern of buildings or information on buildings layouts are available, it becomes relatively easy to recreate the scene virtually. A study on urban forms was conducted by Cheng et al. [2006], where groups of buildings were classified based on their clustering pattern and layouts. 18 such groups were generated based on different building forms and urban densities. The daylight simulations were carried out using the simulation tool *PPF*, which is a RADIANCE based tool which works on the principle of backward ray tracing. Although sky view factors were computed in this study using an elevation model to determine surface availability for solar installations.

It is very evident from this subsection that the use of the existing 3D geometry libraries allows users to include great levels of detail and model urban landscapes more accurately than ever before. However, the scale of the model selected is of great significance as this can drastically effect computational time and processing required and thus is not recommended for large scale PV potential modelling.

2.1.4 Other Sources

Apart from the previously discussed methodologies, other techniques can also be used for obtaining 3D models of urban landscapes. One such technique is use of digital cartography or digital cadastral maps for generating the DSM of the urban scene. The research carried out by Bergamasco and Asinari [2011] in Italy found the solar potential of available roof area in the Piedmont region of Italy by using cadastral data obtained from local municipal authorities used to generate the DEM in the e00 file format. This was used along with the solar radiation data provided by ESRA (esra) for modelling solar radiations and shadows. For determining the roofs, a building typology was established whereby roofs of commercial or residential buildings are easily identified. All these simulations were carried out in ArcGIS, while data processing was performed by MATLAB to convert them into shape file formats. This digital cartography approach was also taken up byHorváth et al. [2016] in their work. Buildings were classified based on application (commercial/residential) and also based on other parameters such as energy consumption. The solar radiation calculations were performed on the generated model based on the approach by Reindl et al. [1990]. Simple assumptions were made such that a shadow correction factor was introduced to take shadow losses into account, and the reflected component was completely neglected in this study.

Another approach in generating 3D models is using the CityGML software for transfer and visualisation of geo-spatial data. This has also been carried out in previous PV potential studies, and the data is input in the .xml format. The inputs for obtaining data in a CityGML format can be through laser scanning, using stereo digital photos of the site or using digital cadastre data with building and roof information. Eicker et al. [2014] used this in their solar resource study for analysing and processing the 3D model in the *SimStadtPreProc* software, which is a JavaScript based tool developed at the University of Stuttgart, Germany. The model had the LoD2 and for enhancing the quality of the model, a healing module called *CityDoctor* was used. A plugin called *Ruby* was then included in the software to simulate the solar irradiance, which was based on the INSEL (insel) environment. The Simstadt tool was also used by Romero Rodríguez et al. [2017], for simulating the PV potential in the German County of Ludwigsburg. As presented in this paper, the solar irradiation can be modelled using the Hay-Davies Sky Model or the Perez Model as desired by the user. The research by Willenborg et al. [2018] proposed a model with 3D mesh data and semantic data to improve the quality of a model. Most 3D models are viewed as meshes, mostly in the .obj file format which contain the geometric data. However, Semantic data includes additional information and thus enhances the 3D model. In their model, a CityGML LoD2 model was combined with a 3D mesh model to also include data of vegetation, dormers and chimneys which can improve shading analysis of the landscape. A solar simulation tool developed by the author in a previous research was used for the simulations.

Although open source, the use of CityGML has a lot of drawbacks as well. According to Biljecki et al. [2016a], the datasets available on CityGML are prone to errors and only models with low levels of detail are reliable for use in other applications. These reliability issues make CityGML datasets incompatible for small-scale studies wherein reflected irradiance and shading effects are to be precisely computed.

2.1.5 Summary of Methodologies and geometric file formats

In this sub-section, the various methodologies followed in previous research to estimate PV potential in urban environments were elucidated and classified based on the nature of input data used. The input data can be sorted as LiDAR point clouds, aerial images, 3D objects from open source libraries or user defined 3D geometries, digital cadastral data and lastly CityGML datasets.

The use of LiDAR point clouds has been found to be the most common in estimating PV potentials, as the data is easily available on public forums. However, it was evident that the use of LiDAR data also has drawbacks, such as the necessity for multiple software platforms and additional datasets for visualising the data. Therefore, it is extremely resource intensive. While the use of photogrammetry techniques is useful in regions where it is difficult to find public geo-spatial datasets, the images have to be processed multiple times to generate useful data and is thus time consuming. Moreover, the use of image processing techniques in PV yield computations have been hardly found in literature and has a lot of potential, but not a practical or user friendly approach to 3D modelling.

Another avenue which can be used for 3D modelling is the use of CityGML datasets, which also has the advantage of being open source. Although these datasets include other information such as semantics, the datasets are prone to lots of errors, especially for datasets with greater levels of detail. They are therefore not advised for generating 3D models while modelling small urban scenes. The use of cadastral maps was also observed, but hardly a few in literature. The most practical and user friendly option in modelling 3D geometries is through the use of existing 3D object libraries such as Google Earth, 3DWarehouse or Openstreetmap data. This summary has been presented in the form of a table in Table 2.1, where the pros and cons of each data input has been explained.

Table 2.1: Summary of Various Data Inputs in 3D Modelling.

Data inputs	Data formats	Advantages	Disadvantages
LiDAR Point Clouds	.las	Public data and therefore easily accessible	Requires Multiple softwares and additional data in other formats
Photogrammetry Techniques	.jpeg, .tiff, .giff	Extremely useful to model locations with no public data	Requires multiple processing steps and is time intensive
CityGML	.xml	Public Data Useful for city/province level modelling	Models with greater levels of detail have errors and can tamper results
3D Libraries	.skp,.kml	User Friendly, requires minimum processing to obtain geometric data	All models and scenes not available easily It is computationally expensive to model complex scenes

Using all these workflows and data formats, a 3D mesh model will be generated which only contains information about the geometry. However, as explained earlier at the start of this chapter, it is also a pre-requisite to include the optical behaviour of the different surfaces to examine the overall solar performance of the panels. The different workflows and softwares used for estimating solar yield, such as GIS based tools, custom algorithms or ray tracing methods have already been explained in this section. However, it is to be noted that these workflows have been developed for larger scales in space (city/district level) and therefore neglect the greater levels of detail required for modelling smaller urban scenes. These approaches for modelling greater optical properties and the data formats used for storing optical data will be explained in section 2.2.

2.2 OPTICAL MODELLING AND DATA FORMATS FOR STORING OPTICAL DATA

In the previous work which has been reviewed so far, it is very obvious that the reflected component of irradiance has been neglected in most studies and in those where considered, the reflectance has been assumed to be a constant value. In urban landscapes however, these assumptions are not valid as such landscapes include a variety of surfaces and materials which exhibit different reflectance behaviour and thus needs to be given greater attention. This effect is amplified further when considering smaller urban scenes with models involving greater levels of detail. In this section, the importance of the reflected component of irradiance illustrated in literature will be provided along with the workflows used for modelling optical properties with a greater level of detail. In addition to this, the file formats used for storing optical data will be highlighted.

The reflected component of irradiance is expressed using the term albedo, which represents the ”whiteness” of a surface. In scientific terms, the albedo is defined as the fraction of incoming solar radiation reflected by a surface. This term is not to be confused with the term reflectivity, which is a material property and is applicable for all wavelengths in the spectrum. The albedo is always a function of the material reflectivity. A review on albedo and reflectivity definitions was carried out by Gueymard et al. [2019], where the differences between the definitions used in previous studies was explained. A thorough study was presented on current state-of-the-art albedo databases for reflectivity studies such as the albedo database provided by NREL, and their usage limitations. The authors recommended that there is a necessity for combining FFA (ffa) and NFR (nfr) data for improving reflectivity studies. As explained earlier in this section, the albedo is a function of the reflectivity of a surface and the ASTER Spectral Library provided in Baldridge et al. [2009]'s is a database which contains the spectral properties for various natural, man-made and lunar materials which can be used to assign the spectral reflectivity properties of a surface based on the material it is made of. Most spectral library sources store the reflectivity data is stored in the ASCII file format and used to compute albedos.

The constant value of albedo which is commonly observed in research could be considered as the weighted average of the spectral albedo over the entire solar spectrum, derived from these reflectances. While this value of albedo can be made to save computational time, it is incorrect to make such an assumption when using an advanced method for modelling solar irradiance. One such advanced method with a greater level of detail is the use of ray tracing techniques, which involves physical equations determining the reflection from light of different surfaces. In a ray tracing approach, rays are sent from the source to the sensor or the other way around to determine how many rays are finally intercepted by the sensor. The first method is called forward ray tracing where light is sent from the source to the sensor, and the second approach is called backward ray tracing where the opposite happens. While forward ray tracing is very accurate for predicting solar irradiance, it is also very time consuming for simulations. Backward ray tracing on the other hand is not as resource intensive as the first method and is useful while modelling complex scene with many asymmetric objects. Nonetheless, it can also tamper the accuracy of the results and therefore a compromise is to be made on which method is to be considered based on the requirement.

There are different workflows which can be used for performing ray tracing simulations. One such workflow based on the forward ray tracing approach has been implemented in the PVMD toolbox. This uses a software called *LUX* which is a forward ray tracer used to determine the plane of array irradiance. This toolbox can so far only model simple arrangements of modules and therefore the reflectances are not modelled effectively. Another approach which can be used for yield calculations is the use of a RADIANCE based backward ray tracer. There are multiple softwares which work based on this workflow. Grasshopper is such a software, where a plugin named Rhino can be used for modelling 3D geometries and another plugin named ladybug/honeybee can be used for assigning optical properties to these surfaces. While the ladybug plugin uses .epw format for modelling the solar radiation, the grasshopper files are stored in the .ghx or .gx format which are XML based and binary based formats respectively. Using the honeybee framework, optical properties only at the visible range can be assigned to the surfaces as this workflow is mainly used for image rendering and not for solar simulations. It is also possible to use other softwares such as COMSOL for the optical modelling where the files are stored in the .mph format, but they are also used for image rendering or optics studies. Therefore, this is the workflow which is to be emulated in the PVMD toolbox to model the solar irradiation by also including the optical behaviour across the incident spectrum of light as this captures the overall picture of the performance. In this research, a backward ray tracer would be used for the simulations.

While using a ray tracing method for solar simulations, the best way to represent reflections is with the use of spectral albedo, which is the wavelength dependent solar reflectance behaviour. The spectral albedo is dependent not only on the wavelength of incident light, but also on the angle of incidence of the rays. The effect of these parameters on the albedo is quantified using the BRDF, which can be defined for all surfaces based on spectral reflectance. Some softwares such as RADIANCE generate their own BRDF based on optical data which was input, while in others this data has to be externally produced. To understand this quantity better, previous research in which BRDF has been studied and utilised in PV modelling will be provided here. In this section, spectral albedo and BRDF (which are the same) will be used interchangeably. It has been very evident from previous research that combining angular data along with spectral reflectivity data improves the results of PV simulations. One explanation of BRDF was provided by the MSc thesis of Gribnau [2020] from the Delft university of Technology, where two BRDF models, the Lambertian and Oren-Nayar reflectance model was explained. The BRDF functions for Lambertian and specular reflectors were also provided in the paper. The differences between reflectivity and albedo were also clearly explained, and it is to be noted that the albedo value is always lesser than equal to the reflectivity for a surface.

An overview of various BRDF models available was presented by Montes and Ureña [2012], and the concepts behind the models was explained in detail. The presented models were compared by classifying them into analytical, experimental and physical models. Based on the requirements for the reflection study, the appropriate reflectance model can be chosen for the research. Moreover, an explanation of various processing techniques used for the reflectance models were also provided. One BRDF model, called the *Ross-Li* model which is used in the MODIS BRDF/Albedo product was integrated by Yang et al. [2020] for a reflectance study to be used in remote sensing applications. As will be seen later in this section, this library is widely sought after while conducting reflectance studies. A geometric albedo model was developed by Ziar et al. [2019] which included the effect of all influential parameters which effect the albedo. These parameters also included the presence of shadows and geometries. Using this information, albedo models were presented for various surfaces, ranging from simple and smooth to complex and rough. To simulate the model, the Unity software package was used for generating the 3D objects to be used in the study.

The influence of spectral albedo in PV performance was modelled by Brennan et al. [2014] by taking into account the variation in EQE (eqe) and short circuit current of the cells. The spectral library for over 22 materials was considered in the study which was obtained from the ASTER Spectral Library. It was shown through the study that albedo makes a significant contribution to solar potential of rooftops and shouldn't be disregarded for BIPV studies. Another study by Andrews and Pearce [2013] considered the effect of spectral albedo on the performance of crystalline and amorphous silicon cells. A comparison was made between spectral-averaged albedo and spectral integrated albedo for both technologies, as the spectrum changes with time during different seasons. A correction factor was introduced to account for these variations, and the spectral library was obtained from the SBDART radiative transfer model developed by the University of California, Santa Barbara. The analysis showed that the power output depends on the ambient albedo, geometric index, module tilt and sky clearness index.

The effect of albedo is more prominent in bi-facial modules, where the reflected component from transmitted light is also a contributor to power output. The research by Russell et al. [2017] studied the effect of spectral albedo on bifacial silicon solar cells and showed that the albedo has a pronounced effect on the power output and affects the thermodynamic limits of the solar cell. The albedo was modelled using the data available from the ASTER Spectral Library and the authors note that the geometric factors, location and self-shading (specific to the bifacial panels) have to be taken into account. The thesis work completed by Sonmez [2017] also explains the importance of spectral albedo in modelling bifacial PV panel performance, and the fact that the multiple cell materials have different band gaps means that spectral albedo can have a big impact in PV yield calculations. Gostein et al. [2020] in their work also calculate the spectral albedo from a spectral library in the SMARTS software library to study its influence on bifacial panel yield. They also noted that the use of sensors to calculate albedo must take into account the geometric influence in albedo measurements, and therefore suitable correction factors have to be included in the calculations.

It can be seen that the reflected component of irradiance has a huge impact on the yield calculations and has to be included in the simulations. While using ray tracing methods, the finest method which can be used for the modelling is the use of the BRDF. Softwares such as RADIANCE and LUX generate their own BRDFs and thus are extremely useful for solar potential calculations. Moreover, it can be seen that the file formats used in these softwares are not compatible for use in the toolbox and a hence it is essential to generate files which are compatible with both the toolbox and the ray-tracing software. A summary of data formats, softwares and other relevant information has been tabulated in Table 2.2.

Table 2.2: Summary of Various Data Formats for Optical Data.

Data Input	Data Format	Data Type	Data Sources
Spectral Libraries	ASCII text files	Spectral reflectivity data	ASTER Spectral Library, SMARTS Spectral Library
Reflectance data	.ghx, .gh	RGB Reflectance data	Honeybee Framework within Grasshopper
Geometric data with Optical Properties	.mph	Optical properties assigned to objects	COMSOL file format for ray optics simulations

2.3 SUMMARY

In this Chapter, various methodologies adopted for visualising 3D geometries and performing solar calculations have been presented. Moreover, the file formats used for storing geometric and optical data has also been summarised in their respective sections.

The data sources for obtaining geometric data include LiDAR point clouds, Photogrammetry techniques, CityGML datasets or from 3D libraries and CAD software designs. On comparing the various formats, it was discovered that open source 3D Libraries are the most user friendly option to obtain geometric data and requires minimum processing and thus recommended for use in the toolbox. The file formats associated with the respective data sources have already been mentioned in Table 2.1.

While a lot of attention has been provided in obtaining geo-spatial data and workflows using different open source or custom softwares, there has been little heed to simulating the reflected component of irradiance. There are multiple methods which can be used for modelling reflected irradiance, such as using a BRDF or assuming a constant reflectance value. It was concluded that for smaller urban scenes, data-sets such as BRDF are required for accurate computation of the same. It was also observed that file formats containing opto-geometric data are very software dependent and thus such a file has to be designed for use in the toolbox. The available file formats containing optical data have been highlighted in Table 2.2.

In literature, most studies neglect the fineness of the model in-favour of computational time and thus neglect attributes such as reflected irradiance. Moreover, the workflows deployed require multiple softwares with constant file format conversions which are cumbersome to adopt. Therefore, a workflow is to be developed which not only user friendly, but also allows flawless modelling of irradiance. This will be presented in the remainder of this report starting with chapter 3, where the workflow of the toolbox, sources and formats of the 3D object will be clarified.

3 WORKFLOW AND THE 3D OBJECT

As mentioned in chapter 1, the second objective is to define a workflow which is simple but effective. Besides this, it is equally important to select the appropriate source for the data used in the toolbox as this will determine the computational limitations of the chosen workflow. This chapter focuses on these two aspects in the thesis, with more emphasis on the source of the geometric data. The source for optical data will be discussed in chapter 5 of this report.

Section 3.1 will discuss the proposed workflow in detail and briefly highlight the software requirements to perform the calculations, while Section 3.2 will extensively explain the source and format of the 3D object which is to be loaded in the toolbox. The chapter is finally summarised in section 3.3.

3.1 PROPOSED WORKFLOW

As is with any computational software, it is crucial to define the apt workflow to ensure ease of access and comfort for the users. The best-case scenario to achieve this is to completely automate the entire process which requires minimal user intervention. However, this cannot be attained at all times due to computational and time restrictions. Therefore, a workflow has been developed to reach an optimal solution, maximise user friendliness and minimise crossover issues. The workflow which will be deployed in the PVMD toolbox is illustrated in Figure 3.1.

As seen in Figure 3.1, the process begins with the input of the 3D geometries. The 3D geometry could be imported in a variety of file formats from numerous sources, and the significance of the sources and their respective formats will be explained once again in section 3.2 of this chapter. As the imported 3D scene is not designed by the user and obtained externally, it is essential to check for errors in the file during the import step. Thus, the 3D object has to be pre-processed to improve model quality and preserve geometric information. The pre-processing steps encompass the meshing of the object and dividing the mesh into discreet elements which best capture the overall 3D scene. It is to be noted that the discretisation of the surfaces only affects undefined objects or polygons within the scene and does not affect other elements within the model. This step could be carried out in an external 3D modelling software, or within the toolbox and the adopted methodology for use in the toolbox will be explained in this chapter.

Once the model has been successfully imported into the MATLAB framework, panels will be added to the scene by either placing it beside the loaded 3D geometry or on an available surface. This feature will be explored in chapter 4 of this report. This will be followed by the assigning of optical properties to the surfaces defining the model, which will be thoroughly elucidated in chapter 5. Finally, the opto-geometric data from MATLAB will then be parsed into an ASCII based text-file which can be input into a ray-tracing software, which will be the RADIANCE light simulation engine in this project. The use of RADIANCE and its capabilities will also be highlighted in chapter 5 of this thesis.

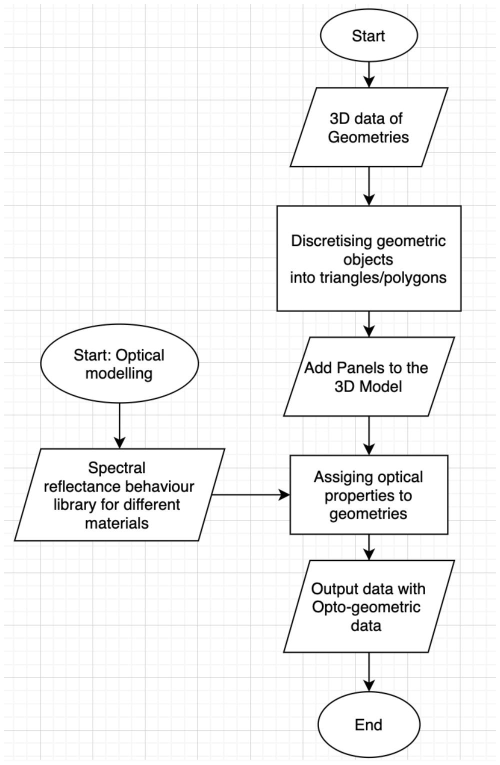

Start
3D data of Geometries
Discretising geometric objects into triangles/polygons
Add Panels to the 3D Model
Start: Optical modelling
Spectral reflectance behaviour library for different materials
Assiging optical properties to geometries
Output data with Opto-geometric data
End

Figure 3.1: Workflow Diagram for Modelling Complex Urban Scenes.

This workflow requires negligible user operations, while simultaneously reducing the need for other softwares such as Ladybug and Honeybee to assign optical properties. Although a 3D CAD software is used, this process can be automated by generating computer codes which can run external softwares from MATLAB. Thus, the proposed workflow satisfies the requirements of maximising ease of use and minimising computational needs without hampering the accuracy of the simulations.

3.2 SOURCES, FORMATS AND 3D OBJECT PRE-PROCESSING

The availability and the existence of a wide range of sources for 3D objects is both a boon and a bane, with each accompanied by its limitations and advantages. As mentioned in chapter 2, multiple approaches have been previously adopted to compute solar potential in urban environments. These approaches included various softwares, required several file conversions and processing steps to arrive at the final results. The selection of a suitable format is thus a balancing act, with the requirement being user-friendliness and low computational requirements as mentioned earlier in this chapter. As a refresher, a compilation of the various sources, their advantages and disadvantages are presented again in the form of a table in Table 3.1 .

As seen in Table 3.1, 3D libraries give the most practical solution to procure 3D objects with minimal editing and processing. Although it becomes computationally expensive to create a scene with multiple objects and features, the ease of use of this data means that they make it easier to model smaller urban scenes consisting of one or two buildings with greater levels of detail and include features such as dormers

Table 3.1: Advantages and Disadvantages of Various 3D File Sources.

Data inputs	Advantages	Disadvantages
LiDAR Point Clouds	Public data and therefore easily accessible	Requires Multiple softwares and additional data in other formats
Photogrammetry Techniques	Extremely useful to model locations with no public data	Requires multiple processing steps and is time intensive
CityGML	Public Data Useful for city/province level modelling	Models with greater levels of detail have errors and can tamper results
3D Libraries	User Friendly, requires minimum processing to obtain geometric data	All models and scenes not available easily It is computationally expensive to model complex scenes

and chimneys. Within 3D Libraries, there are numerous sources available such as Google Earth, OpenstreetMap and 3dwarehouse (a Google SketchUP library). In this project, objects from the 3dwarehouse library will be imported as they are open source, contain various real-world 3D objects & scenes and provides an option to design custom scenes with the help of Google SketchUP.

Files which are stored in 3dwarehouse are available in the *.skp* format, which is the file format SketchUP produces its files in. As MATLAB is not a good platform for directly importing geometric data, an external 3D CAD (cad) software is required to import the file into a MATLAB compatible format. In this project, Rhinoceros 6 (RHINO for short) was chosen as the platform to import files from 3dwarehouse and export it for use in the toolbox. Other softwares such as Blender or Autodesk could also be used. While selecting the CAD software, file compatibilities are to be considered as not all softwares accept geometry files in the .skp format. Similarly, the export capabilities of the software are important as MATLAB is only good at reading ASCII-based text-file data. In addition to file formats, file versions also need to be compatible with the CAD software, as older softwares cannot read new file formats or updated file extensions and vice-versa. For RHINO 6, only SketchUP files from versions up-to 2019 are compatible. Therefore, this project begins with obtaining a 3D object to be placed in the scene in the 2019 .skp format.

Let us consider a mountain cabin as shown in Figure 3.2, which has been taken from Google SketchUP's 3dwarehouse library. As the file was externally generated and imported, the file could have import errors and therefore will be pre-processed before exporting into the toolbox.The first step is to re-mesh the object, which can be carried out using the ***MESH*** command on RHINO. This ensures that surfaces which have been incorrectly meshed or not meshed at all will be rectified. Not all surfaces undergo the meshing, as seen in Figure 3.3, where only the vegetation present in the model of the house has been highlighted for performing the mesh function.

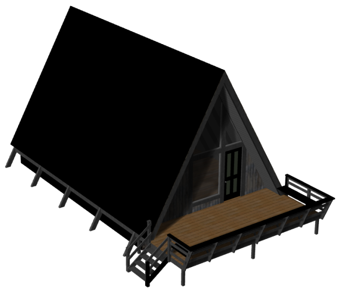


Figure 3.2: The Selected 3D Object for the Simulation.

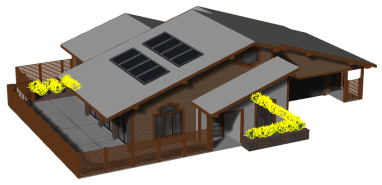


Figure 3.3: Meshing of un-meshed surfaces in RHINO for 3D objects.

The second step is to quadrangulate or triangulate the mesh. This is essential to ensure model quality by correcting errors in surface definitions. Moreover, it maintains vertex definitions by keeping the relationships between one vertex and its neighbours intact. As the name suggests, triangulation converts all newly meshed faces into triangles, while quadrangulation converts all faces into rectangles. Each method has its pros and cons: defining surfaces with triangles preserves model quality, but results in more surfaces/elements and therefore larger files and greater computational times. On the other hand, quadrangulating surfaces results in smaller files, but can include non-planar surfaces and therefore model quality. In-order to reduce the number of surfaces defining a model and simultaneously preserve model quality, a two-step procedure will be followed: firstly, the model will be quadrangulated in the 3D CAD software. After importing the file into MATLAB, the model will be refined and non-planar surfaces will be eliminated. The second step to be carried out will be explained in chapter 4 of this report. Quadrangulation ensures that a surface is defined by a minimum of four vertices and a snippet of the quadrangulation step for the selected 3D object (the mountain cabin) is shown in Figure 3.4 . This step is achieved in RHINO using the ***QUADRANGULATEMESH*** command. Although the command seems to selects all surfaces in the model, it only performs the function on newly meshed surfaces as will be elucidated later in this chapter.

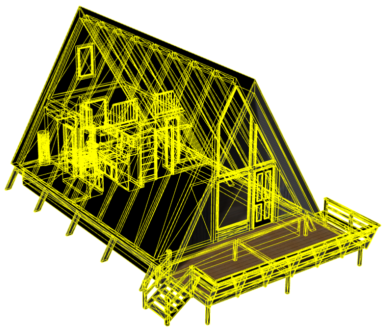
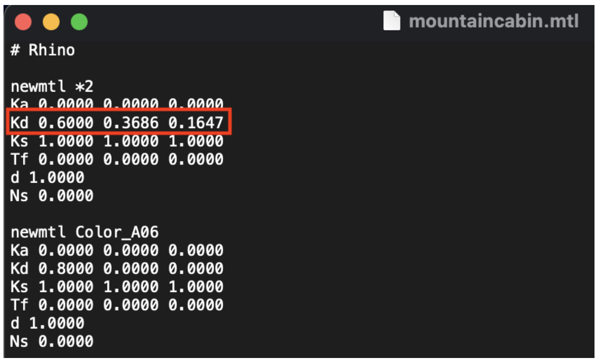


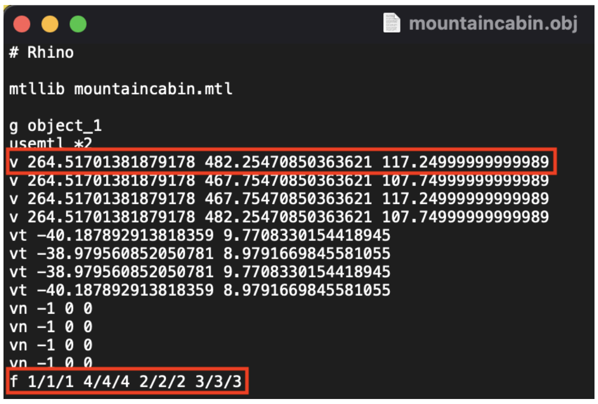
Figure 3.4: Generating Quad Elements in the mesh of a 3D object.

Once the model has been meshed and quadrangulated, the file can be imported for use in the toolbox. As brought up earlier in this chapter, MATLAB is not good in all reading geometric data formats and can read only text-file based information. There are many file formats which contain geometric data in text-files, but the one chosen in this project is the Wavefront Technologies' .obj format, where the **".obj"** is the abbreviation for **object**. This is a very simple file which contains only geometric information and entails data on the vertices in Cartesian coordinates, surface normals, surface texture values and face definitions. The faces are defined by the vertex numbers which constitute the face. Apart from the geometric data, .obj files are always accompanied by a file containing material information, in the format .mtl, where the **".mtl"** stands for **material**. The 3D object from RHINO is thus exported into these text-file based formats and Figures 3.5 and 3.6 show a preview of the 3D object stored in the .mtl and .obj file respectively.



```
# Rhino

newmtl *2
Ka 0.0000 0.0000 0.0000
Kd 0.6000 0.3686 0.1647
Ks 1.0000 1.0000 1.0000
Tf 0.0000 0.0000 0.0000
d 1.0000
Ns 0.0000

newmtl Color_A06
Ka 0.0000 0.0000 0.0000
Kd 0.8000 0.0000 0.0000
Ks 1.0000 1.0000 1.0000
Tf 0.0000 0.0000 0.0000
d 1.0000
Ns 0.0000
```

Figure 3.5: Preview of data stored in .mtl files, with the Value "Kd" highlighted.



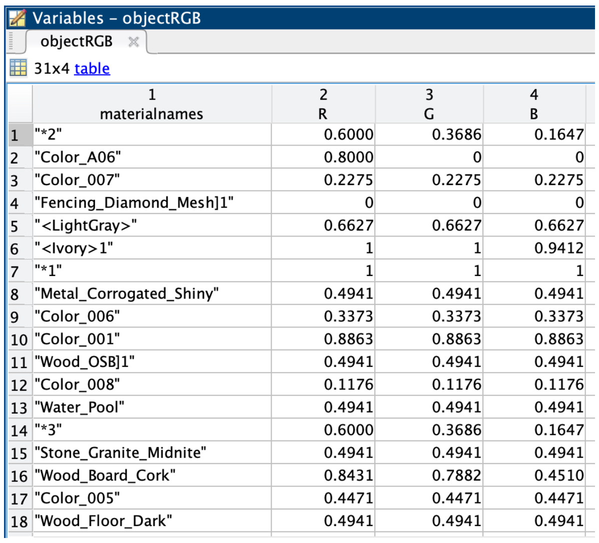
```
# Rhino

mtllib mountaincabin.mtl

g object_1
usemtl *2
v 264.51701381879178 482.25470850363621 117.24999999999989
v 264.51701381879178 467.75470850363621 107.74999999999989
v 264.51701381879178 467.75470850363621 117.24999999999989
v 264.51701381879178 482.25470850363621 107.74999999999989
vt -40.187892913818359 9.7708330154418945
vt -38.979560852050781 8.9791669845581055
vt -38.979560852050781 9.7708330154418945
vt -40.187892913818359 8.9791669845581055
vn -1 0 0
vn -1 0 0
vn -1 0 0
vn -1 0 0
f 1/1/1 4/4/4 2/2/2 3/3/3
```

Figure 3.6: Preview of data stored in a .obj file, with the values "v" and "f" of our interest.

In both files, it can be seen that comments are defined using the # symbol. The names of the materials used in the 3D object are defined with the command **newmtl**, as seen in Figure 3.5. These material names need not mean anything logical and are used to represent the R,G and B values of the textures used to visualise a given object in a 3D CAD software. The parameters used to define these textures Wavefront-Technologies [1995] are defined with reference to the Phong Reflection Model Phong [1975], which is used in computer graphics and image rendering:

- Ka: Ambient colour, to account for light scattered in the entire space. The values always lie between 0 and 1
- Kd: Diffuse colour, contributes to colour of the surface. The value of the parameters lie between 0 and 1
- Ks: Specular colour, to visualise shiny mirror like surfaces
- Tf: Transmission filter, to filter light passing through the surface
- d: Dissolve factor, to define how much the object dissolves into the background
- Ns: specular highlights, ranging from 0 to 1000

The parameter of importance for this project is **Kd**, as this defines the visual properties of the material assigned to the surfaces, as highlighted in Figure 3.5. To obtain this data, a material file is loaded and read line-by line in MATLAB, and only lines containing the value *Kd* are parsed and the values in this line are stored in the work space. Similarly, to obtain the texture names, lines consisting of the phrase *newmtl* are parsed and the names are stored in a string. These two variables are combined and the texture data, along with texture names in the form of a table array, as highlighted in Figure 3.7.


Variables - objectRGB

objectRGB

31x4 table

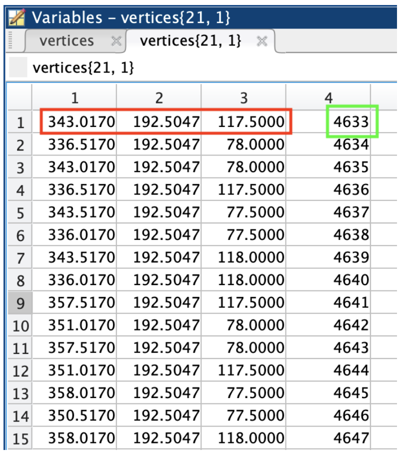
	1 materialnames	2 R	3 G	4 B
1	"*2"	0.6000	0.3686	0.1647
2	"Color_A06"	0.8000	0	0
3	"Color_007"	0.2275	0.2275	0.2275
4	"Fencing_Diamond_Mesh]1"	0	0	0
5	"<LightGray>"	0.6627	0.6627	0.6627
6	"<Ivory>1"	1	1	0.9412
7	"*1"	1	1	1
8	"Metal_Corrogated_Shiny"	0.4941	0.4941	0.4941
9	"Color_006"	0.3373	0.3373	0.3373
10	"Color_001"	0.8863	0.8863	0.8863
11	"Wood_OSB]1"	0.4941	0.4941	0.4941
12	"Color_008"	0.1176	0.1176	0.1176
13	"Water_Pool"	0.4941	0.4941	0.4941
14	"*3"	0.6000	0.3686	0.1647
15	"Stone_Granite_Midnite"	0.4941	0.4941	0.4941
16	"Wood_Board_Cork"	0.8431	0.7882	0.4510
17	"Color_005"	0.4471	0.4471	0.4471
18	"Wood_Floor_Dark"	0.4941	0.4941	0.4941

Figure 3.7: Material information stored in the form of a table in MATLAB.

Once the .mtl file data has been loaded in successfully, the .obj file is loaded into MATLAB. From Figure 3.6, the variables used to store data in such a file are defined as:

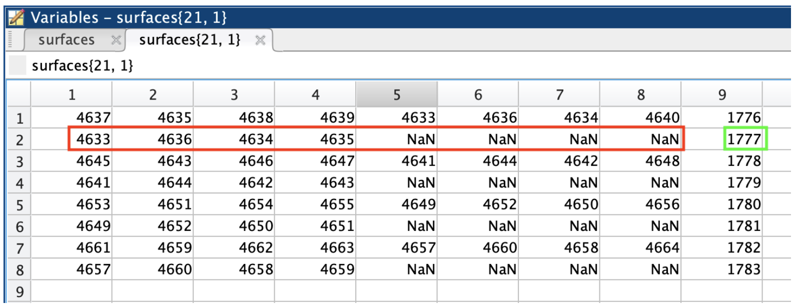
- v: Vertex definition, succeeded by the Cartesian coordinates of the point
- vt: Texture definition of the vertex
- vn: Normals of the vertices defining a surface
- f: Face definition, succeeded by the vertex numbers defining a surface in counter-clockwise order.

A .obj file references itself to its corresponding .mtl file with the command **mtllib**, which means **material library**. To access material information stored in a .mtl file, the command **usemtl** is used and each object or surface is defined by its name, preceded by the letter **g**. The representation of a face in the manner shown in Figure 3.6 means that the face is defined by vertices 1,4,2 and 3 and they are each repeated three times to represent the X,Y and Z coordinates of the vertices respectively. From the .obj file, only the parameters defined by the variable **v** and **f** are of significance to us, as 1) we are only interested in the geometric data and 2) the normals can be derived with the help of the vertices defining a surface. These lines have been marked in the red box in Figure 3.6. Therefore, when the file is loaded into MATLAB, lines from the file containing the value **v** are parsed and the Cartesian coordinates of the point are stored in the work space. To parse surface definitions, the lines with the variable **f** are read and unique values of the vertices are stored in the work space. Figures 3.8 and 3.9 show how geometric data is successfully read and loaded in MATLAB. The red box in both images represents the vertex definition (with X, Y and Z coordinates) and face definition (with vertex numbers) respectively.



	1	2	3	4
1	343.0170	192.5047	117.5000	4633
2	336.5170	192.5047	78.0000	4634
3	343.0170	192.5047	78.0000	4635
4	336.5170	192.5047	117.5000	4636
5	343.5170	192.5047	77.5000	4637
6	336.0170	192.5047	77.5000	4638
7	343.5170	192.5047	118.0000	4639
8	336.0170	192.5047	118.0000	4640
9	357.5170	192.5047	117.5000	4641
10	351.0170	192.5047	78.0000	4642
11	357.5170	192.5047	78.0000	4643
12	351.0170	192.5047	117.5000	4644
13	358.0170	192.5047	77.5000	4645
14	350.5170	192.5047	77.5000	4646
15	358.0170	192.5047	118.0000	4647

Figure 3.8: Vertex Data loaded in MATLAB.



	1	2	3	4	5	6	7	8	9
1	4637	4635	4638	4639	4633	4636	4634	4640	1776
2	4633	4636	4634	4635	NaN	NaN	NaN	NaN	1777
3	4645	4643	4646	4647	4641	4644	4642	4648	1778
4	4641	4644	4642	4643	NaN	NaN	NaN	NaN	1779
5	4653	4651	4654	4655	4649	4652	4650	4656	1780
6	4649	4652	4650	4651	NaN	NaN	NaN	NaN	1781
7	4661	4659	4662	4663	4657	4660	4658	4664	1782
8	4657	4660	4658	4659	NaN	NaN	NaN	NaN	1783
9									

Figure 3.9: Face Definitions stored loaded in MATLAB.

The data contained in the last column of the vertices matrix in Figure 3.8 defines the vertex number or index as shown within the green box, while the last column of the surfaces matrix in Figure 3.9 defines the face number in the model (again, shown in the green box). To match the number of columns of the surface defined by the maximum number of vertices, faces defined by lesser number of points are appended with **NaN** entries. It can be observed in Figure 3.9 that faces defined by more than 4 points exist in the model, despite quadrangulating the entire model. This evidence suggests that only newly meshed surfaces were converted into quad-elements and other meshes were left unaffected.

Therefore, a 3D object from a 3D library such as 3dwarehouse has been successfully imported into the toolbox by converting the data and storing it in a text-file based format. With this information, it will now be possible to refine the model, eliminate inaccuracies present in the model and define the scene for PV calculations by placing the solar panels in it.

3.3 SUMMARY

In this chapter, a workflow was developed for use in the PVMD toolbox as mentioned in section 3.1, where 3D files from open-source 3D libraries are imported into the toolbox to perform PV calculations. In this workflow, Google's 3dwarehouse has been used as the starting point to obtain 3d objects which will be part of the scene used in the PVMD toolbox. This file, in the .skp format, is loaded into a 3D CAD software such as RHINO to maintain and scrutinise model quality. To preserve the geometric data, the model is to be pre-processed by re-meshing and quadrangulating the surfaces. To facilitate the inclusion of the processed object into MATLAB, it is exported into a text-file based format, where the files contain the geometric data and material texture data. Once the file has been loaded and parsed into a MATLAB format, PV panels will be placed in the scene. Optical properties of the individual surfaces are then assigned and the object is finally converted back into a file format which is suitable for use in backward RADIANCE's ray-tracing engine. This workflow ensures minimal user intervention in performing the PV simulation by eliminating the need for other softwares such as Ladybug and Honeybee to assign the optical properties.

In section 3.2, the format suitable for inputting 3D objects into the toolbox was selected, which is the .obj file format. Once this file has been loaded, the procedure to read the necessary parameters from both the geometry file and corresponding texture file was elucidated, along with the explanation of the various variables loaded into the toolbox. This data will be used to place the PV panels in the scene before assigning the optical properties, and a detailed overview of this step is presented in chapter 4.

4 COMPLETING THE SCENE

The third objective in this project is to refine the loaded 3D geometry to eliminate errors and complete the scene by placing the panel geometry in it. Even though the object has been refined once in the CAD software, errors could still exist due to the following reasons: firstly, the quadrangulation step used in the pre-processing step can generate non-planar surfaces (a set of four-vertices need not be co-planar) and secondly, if a model already consisted of ill-defined polygons, these would remain intact within the geometry definition. Therefore, a second refining step is necessary to maintain the accuracy of ray-tracing simulations before assigning optical properties.

This chapter is thus divided into two parts: section 4.1 will illustrate the algorithm used to eliminate errors in the loaded geometry, while section 4.2 explains the possibilities and the procedure used to place the solar panels to complete the scene. Once again, the chapter is concluded with a summary of the findings in section 4.3.

4.1 REFINING THE 3D OBJECT

Minimising computational requirements by reducing the file size, in this case defining a geometry with minimal surfaces leads to a compromise in the accuracy of results as model quality will be lost. Although the idea of defining an object with the least amount of data is the correct approach, the potential sources for errors while doing so should not be neglected. As discussed previously in Chapter 3, defining surfaces with quad elements (surfaces made up of four vertices) could result in non-planar surface definitions. If an element consists of four vertices with one non-planar vertex as shown in Figure 4.1, it can be triangulated in two different ways as shown in the Figure 4.2.

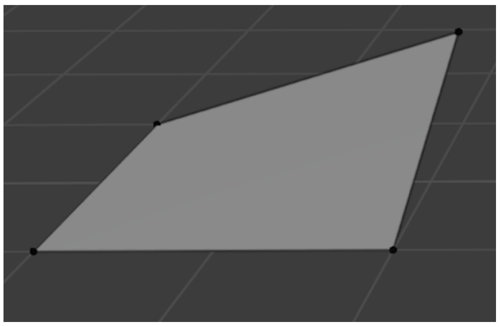


Figure 4.1: Example of a Non-Planar Surface with four vertices.

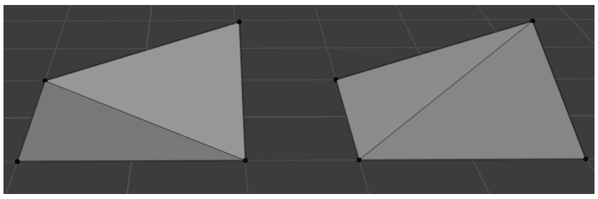


Figure 4.2: Two Possibilities for Triangulating Quad Element from Figure 4.1.

These possibilities shown in Figure 4.2 highlight that the ray-tracing engine will encounter difficulties in determining the orientation of the reflected ray during the simulations as the surface can be defined in multiple ways. Such non-planar surfaces are commonly observed while meshing objects defining vegetation in the scene, as these objects consist of complex polygons arranged in 3D space. Therefore, these surfaces need to be triangulated by preserving the surface definition. Apart from these surfaces, polygons could also be made-up of collinear vertices, which would not get affected during the import or triangulation process. These surfaces need to be eliminated as well and these two problems will be targeted in this section.

To determine the planarity of a surface defined by a certain number of vertices, the rank of the matrix containing the vertex data is computed. If the rank of the matrix is greater than the size of the matrix, it is considered to be a non-planar surface. This is used for the entire data set consisting of the vertex and surface definitions to segregate non-planar and planar surfaces. Let's consider a non-planar surface defined as illustrated in Figure 4.3. Even though this surface seems planar, it is clearly visible from Figure 4.4 that the surface is non planar due to its curvature. There are multiple ways to triangulate this surface, and the preferred method is the Delaunay Triangulation as it does not create skewed triangles. In other words, this means that such an algorithm triangulates vertices in such a way that it generates the best possible outcome.

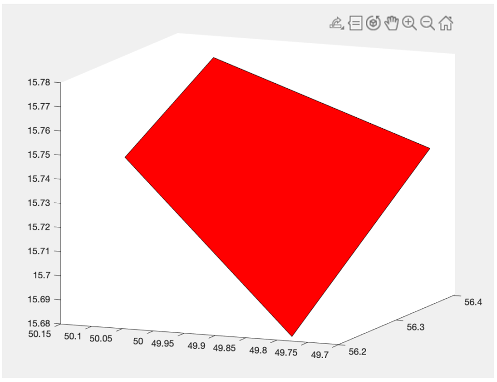


Figure 4.3: A non-planar quad element.

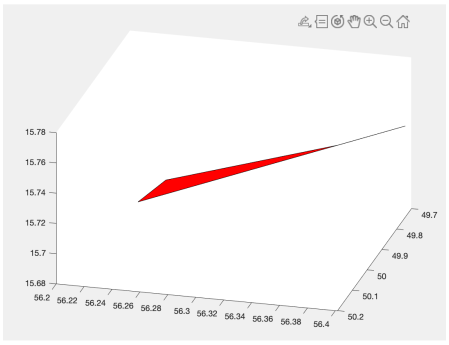


Figure 4.4: Curvature in the Non-planar Surface shown in Figure 4.3.

Preserving the shape of a non-planar surface can become complicated during a triangulation, especially when concave surfaces have been used to define a geometry. This is because triangulation methods usually consider the convex hull of the vertices used to define the surface. In order to prevent this from happening, the MATLAB command ***Polyshape*** is used. To make use of this command, it is essential to translate and rotate the surface onto a 2D plane, such as the XY plane. This is because most of the surfaces are defined in 3D space (3D coordinates) and the *Polyshape* command only works well for surfaces defined by 2 coordinates. Figure 4.5 shows the triangulation of the polygon with the help of *Polyshape* and Delaunay triangulation. In this case, much difference is not observed as the chosen surface is already convex in nature.

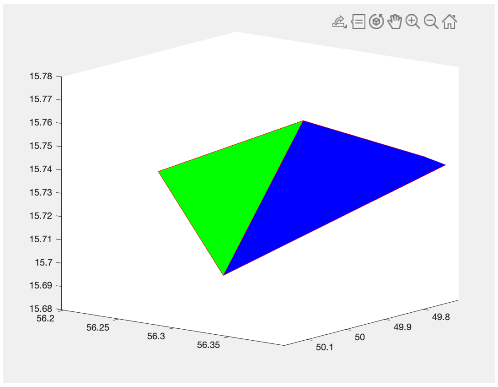
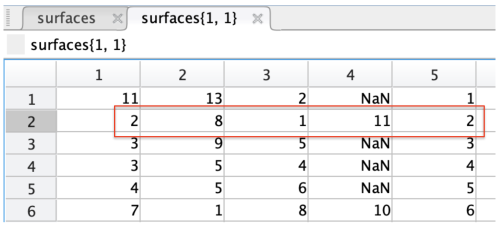


Figure 4.5: Triangulated Output of Non-planar element from Figure 4.3.

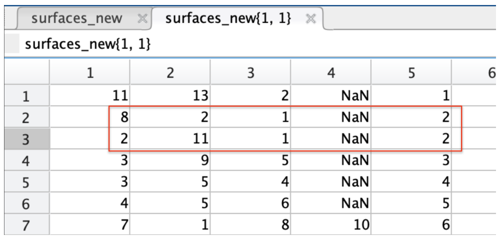
Once this has been performed, the data-set containing the non-planar surface definition is replaced with the new surface definitions containing only triangular elements. This is highlighted in Figure 4.6 and 4.7 respectively, with the old and new surface elements highlighted in the red box. This procedure is performed for every non-planar surface encountered in the defined geometry.


surfaces | surfaces{1, 1}

surfaces{1, 1}

	1	2	3	4	5
1	11	13	2	NaN	1
2	2	8	1	11	2
3	3	9	5	NaN	3
4	3	5	4	NaN	4
5	4	5	6	NaN	5
6	7	1	8	10	6

Figure 4.6: Old Surface definition with 4 vertices.



	1	2	3	4	5	6
1	11	13	2	NaN	1	
2	8	2	1	NaN	2	
3	2	11	1	NaN	2	
4	3	9	5	NaN	3	
5	3	5	4	NaN	4	
6	4	5	6	NaN	5	
7	7	1	8	10	6	

Figure 4.7: New surface with triangulated faces.

With the non-planar polygons refined, the second objective is to eliminate zero-area polygons. Zero area polygons are those surfaces which are invalid because they are made up of collinear vertices, or are surfaces which are not closed. Invalid surfaces would be present in the model only when there are errors during the import/export step and are extremely uncommon. Surfaces defined by collinear points however do exist and need to be removed from the face definitions. To delete these surfaces, the rank of each matrix containing the data of the vertices defining a surface is computed, with a tolerance value. The tolerance value takes into account the number of decimal places of the vertices in the matrix. If the rank of the matrix is lower than the number of rows of the matrix, then the vertices are collinear. This is used to delete surfaces which have no area. Thus, the loaded 3D Object has been refined further to improve the accuracy of the ray-tracing simulations. The next step is to complete the scene by arranging the PV panels, which will be extensively discussed in section 4.2.

4.2 COMPLETING THE SCENE

Once the object has been refined, PV panels can be assigned to the scene to perform the solar potential simulations. There are two possibilities to place PV panels within the scene: placing PV panels beside an object, or placing the PV panels on the roof or facade of a building. This section will elaborate on the algorithm used to achieve this.

Before including the panels within the scene, the 3D object or building is translated to the positive octant in 3D space and is made to lie on the XY plane. Additionally, the edges of the base of the building are made parallel to the X or Y-axis in order to ease the rotation of the building by the user defined azimuth angle. These steps are carried out because the panel geometry is generated starting at the origin and within the positive octant. The convention for the azimuth angle is as follows: 0° is South, 90° is West, 180° is North and 270° is East. After assigning an azimuth angle or orientation of the building, its bounding box or in other words, the bounds of the building are found. This will be further used to place the panels away from the object if necessary. A building with no azimuth assigned and a building with an azimuth angle of 45° are depicted in Figures 4.8 and 4.9 respectively.

The panel geometry consists of the panels themselves, and the frames which are used to hold them in place. If the panels are placed on a roof or facade of the loaded building, then only the PV modules are generated. The following parameters are defined to generate a single panel:

- Number of cells in a row
- Number of rows of cells
- Cell length

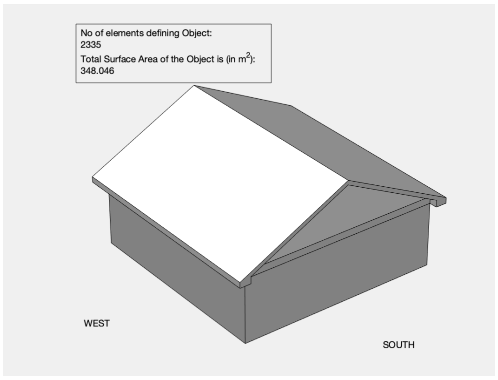

No of elements defining Object:
2335
Total Surface Area of the Object is (in m^2):
348.046
WEST
SOUTH

Figure 4.8: A Building facing South, Azimuth 0 °.

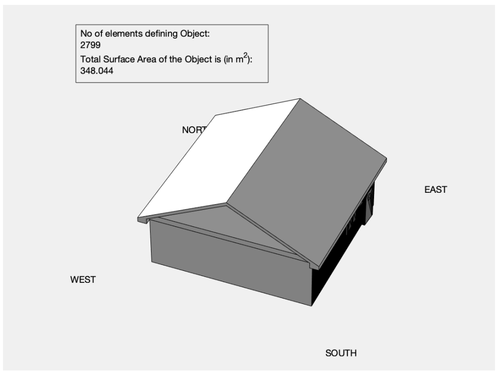

No of elements defining Object:
2799
Total Surface Area of the Object is (in m^2):
348.044
NORTH
EAST
WEST
SOUTH

Figure 4.9: Building from Figure 4.8 facing South-West with an Azimuth of 45°.

- Cell width
- Cell spacing
- Space between edge of module and cell
- Tilt angle of the module
- Azimuth angle of the module
- Module Thickness
- Height of module from the ground
- Bifaciality of a module

All the dimensions of a module and cell are input in **centimeters**, which is obtained from the datasheet of the module considered for the simulations. If a panel is placed on a surface of the 3D object, then the azimuth angle and module tilt angle are directly computed from the surface selected for placing it. The bifaciality of a module is a binary value: 1 means that the module is bifacial and 0 means that the panel is mono-facial. If the modules are placed on the ground at a distance from the 3D object, the geometry of the frames is generated by creating a rectangular box with a square cross-section of edge length 5 cm and height defined by the height at which the modules are mounted from the ground. For modules placed on a surface, bifaciality is automatically assigned to 0.

As mentioned earlier in this section, the panels are generated in the positive octant of 3D Space and lie on the XY Plane. The geometry of a single panel is generated as follows: The module is designed as a cuboid with length and width calculated using the cell dimensions and spacing values. The height of the box is equal to the thickness of the module defined, and the cell are placed onto the top surface of the module like "stickers". For bifacial modules, the cells are similarly placed on the bottom surface as well. The geometry of a single mono-facial panel with frames is illustrated in Figure 4.10. The generated module will be in portrait orientation if the number of cells in a row is greater than the number of rows of cells and in landscape orientation if otherwise.

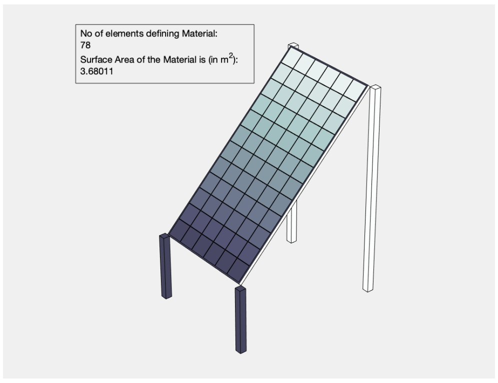

No of elements defining Material:
78
Surface Area of the Material is (in m^2):
3.68011

Figure 4.10: Geometry of a single mono-facial panel, Portrait Mode.

It is also possible to simulate the irradiance for multiple panels at the same time by defining the following parameters:

- Number of modules in a row
- Number of rows of modules
- Row spacing

The Row spacing is also defined in **centimeters**. Before placing the modules in the scene, it is important to refer to the units in which the 3D Object has been designed. This because the panels are always defined in centimeters and it is possible that the object is defined in inches or meters, which in turn means that the panels could either be too large or too small when compared to the object. To achieve this conversion, the following scaling factors are used in the model:

- Centimeters to Inches: 0.3937 (1/2.54)
- Centimeters to Meters: 0.01
- Centimeters to Millimeters: 10

With the panels successfully defined and scaled to the units of the model, it can be placed in the scene. If the modules are to be placed beside the object, the modules are translated using the equation:

$$X = X_{lim} + distance_1 \tag{4.1}$$

$$Y = Y_{lim} + distance_2 \tag{4.2}$$

where,
X_{lim} is the boundary of the 3D object along the X-axis
Y_{lim} is the boundary of the 3D object along the Y-axis
$distance_1$ is the defined distance for translation along X-axis

$distance_2$ is the defined distance for translation along Y-axis

The values of X_{lim} and Y_{lim} will be the maximum value of the X and Y coordinate of the object respectively if the distances are positive, or will be the minimum value of the X and Y coordinate of the object respectively if the distances are negative. A negative distance just means that the panels are translated to the left of the object (as the object in itself predominantly lies in the positive octant). The values for X_{lim} and Y_{lim} are obtained from the bounding box of the 3D object. Figures 4.11 and 4.12 depict the placement of panels beside the object in portrait and landscape mode respectively. In the given figures, it is to be noted that the colours shown here are a default setting from MATLAB and do not represent the actual colours of the scene.

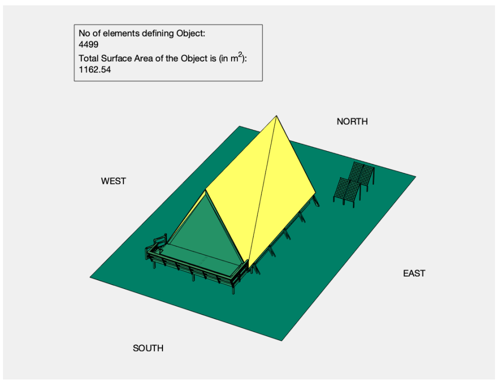

No of elements defining Object:
4499
Total Surface Area of the Object is (in m^2):
1162.54
NORTH
WEST
EAST
SOUTH

Figure 4.11: Panels arranged beside a building facing South, Portrait mode.

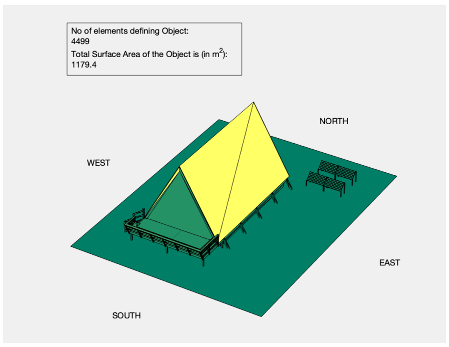

No of elements defining Object:
4499
Total Surface Area of the Object is (in m^2):
1179.4
NORTH
WEST
EAST
SOUTH

Figure 4.12: Panels arranged beside a building facing South, Landscape mode.

If the panels are to be placed on a surface (roof or facade) of the loaded building, it is first translated to the vertex of the plane which is closest to the origin. The Panels are made to slightly hover above the plane and not placed exactly on it in order to prevent errors in the ray-tracing simulations due to merging surfaces. Then, the panels are translated along the edges of the plane by the defined distance as:

$$[X_{dist}, Y_{dist}, Z_{dist}] = distance_1 * edge_1 + distance_2 * edge2 \tag{4.3}$$

where,
X_{dist} is the translation distance along X-axis,
Y_{dist} is the translation distance along Y-axis,
Z_{dist} is the translation distance along Z-axis,
$distance_1$ is the translation distance along edge 1,
$edge_1$ is the normalised edge vector of first edge,

distance$_2$ is the translation distance along edge 2,
edge$_2$ is the normalised edge vector of second edge,

The left hand side of equation 4.3 represents the translation vector which is to be used to move the panel along the selected surface of interest. Figure 4.13 shows an example where panels are placed on a tilted roof of a mountain cabin. To complete the scene, a ground is added on the XY plane with boundaries derived from the limits of the scene with both panels and the object, which is seen from Figures 4.11 to 4.13.

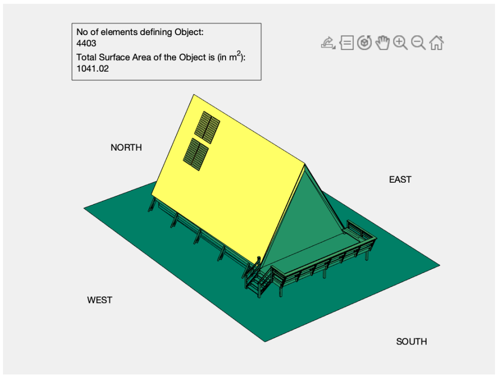

No of elements defining Object:
4403
Total Surface Area of the Object is (in m^2):
1041.02

Figure 4.13: Panels placed on a Tilted Surface, facing West.

Thus, the scene for simulation is completed, with the toolbox now capable of visualising urban scenes with PV panels and computing available irradiance for scenes with panels placed on a surface of the 3D object or beside the 3D object. With the geometrical features completely defined, the next step is to assign the optical properties to the surfaces and convert the data into a format suitable for ray-tracing, which is discussed in detail in Chapter 5.

4.3 SUMMARY

This chapter focused on the third objective of the project: to refine the loaded 3D geometry and to complete the scene by placing PV panels in it. 3D objects imported from external sources contain two kinds of errors: presence of non-planar surface definitions and zero-area polygons, made up of collinear vertices. The first issue was resolved by triangulating the non-planar surfaces by maintaining polygon shape and the second issue was eliminated by deleting the zero-area polygons present in the model. The triangulation method followed was the Delaunay Algorithm on MATLAB.

Secondly, the panels were placed within the scene once the model was refined. There are two possibilities to place it in the scene: beside the 3D object or on a surface of the 3D object, such as roofs or facades. The building is initially translated to the positive octant in 3D space and rotated by the defined azimuth angle of interest before placing the panels. Afterwards, the panel geometry is generated by creating the module as a cuboid and placing the solar cells in the form of "stickers" on top of it. If the panels are placed beside the object, frame geometry is defined with a square cross-section and height equal to mounting height of the panels. The panels and frames together are placed beside the object by translating it from the boundaries of the 3D object input in the toolbox. If the module geometry is to be placed on a surface of the object, the modules are translated to the plane of interest and moved along the edges of the plane. Finally, a ground is assigned to the scene by defining boundaries on the XY plane to complete the scene. This data will be combined with optical data before the ray-tracing simulations are performed and this is elaborated in Chapter 5 of this report.

5 OPTICAL PROPERTIES, DATA CONVERSION AND RAY-TRACING

The fourth objective of this thesis is to collect the reflectance properties of various materials, and use this data to assign optical properties to the surfaces in the model. This opto-geometric data will then be converted to the RADIANCE syntax for carrying out the ray-tracing.

This chapter is thus divided as follows: Section 5.1 discusses the source for the optical data, how this is converted into the RADIANCE suitable format and assigned to each surface present in the scene. Section 5.2 presents the science behind the RADIANCE ray-tracing engine and explains its selection for use in the toolbox. Similar to the previous chapters, this chapter is also concluded with a summary in Section 5.3.

5.1 OPTICAL PROPERTIES AND THE RADIANCE SYNTAX

There exist several sources for obtaining optical properties of all kinds of materials, ranging from natural substances to man-made materials. One such open source library for obtaining optical properties is the NASA ECOSTRESS Library, which was also observed aplenty in Chapter 2 of this thesis report. A satellite imaging technique is used to obtain this spectral reflectance data and the primary reason for choosing this source apart from its accessibility is its data abundance. The data set provided contains spectral reflectance data of various materials in a wide range of wavelengths and has been used widely in previous research due to its data accuracy. As the simulations carried out in the toolbox work within the wavelength range of the solar spectrum or 250 - 1500 nm (nm), data beyond 1500 nm has been neglected for this study. Most of the data provided by NASA only begins from 300 nm, and therefore reflectance data below and up to 250 nm has been linearly extrapolated from the reflectance data present in the early ultraviolet range of the electromagnetic spectrum. A visual representation of the spectral reflectance of a few materials from this material library has been presented in Figure 5.1.

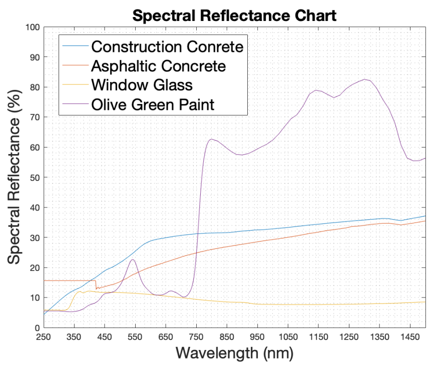


Figure 5.1: Spectral Reflectance of Materials from the NASA ECOSTRESS Library.

The spectral reflectance data provided by NASA cannot be directly used to assign the optical properties of a surface in the toolbox, as RADIANCE has its own syntax for assigning optical properties to each geometric entity in the ray-tracing simulation. A material is defined using the following syntax in RADIANCE:

void *primitive* **materialname**
0
0
5 **R** **G** **B** **Sp.** **Ro.**

In the above definition for a material, the term *primitive* represents the material primitive or the type of a material. According to the RADIANCE definitions, this "primitive" is defined as plastic, metal, etc. In this project, materials are classified as either plastic, metal, mirror or glass. Mirror is used to define highly reflective materials while glass is used to define highly transparent materials. Other general materials are classified as either plastic or metal. Glass and mirror type materials are defined using only three values and therefore, the 5 is to be replaced by three in the aforementioned syntax.

As explained in the syntax, each material is defined with the help of 5 parameters. The first three parameters R, G and B represent the reflectance property of the material in the three channels respectively. As RADIANCE is primarily a visual rendering tool, these 3 parameters represent the R, G and B values used in representing objects in computer graphics. However, these 3 channels within RADIANCE are agnostic and therefore could be used to represent the reflectance property of the surface in any part of the spectrum. This capability of RADIANCE will be used in the toolbox, with the three channels used to represent Ultraviolet UV (uv) (R), Visible range reflectance (G) and Infrared reflectance IR (ir) (B) for carrying out the irradiance calculations.

In order to convert the reflectance data from the NASA library into the RADIANCE format, the spectral reflectance data has to be averaged in the UV, visible and IR range respectively. A normal average does not provide correct results and therefore a weighted average obtained in the form of spectrally integrated albedo is to be incorporated Vogt et al. [2018], which is defined using the equation:

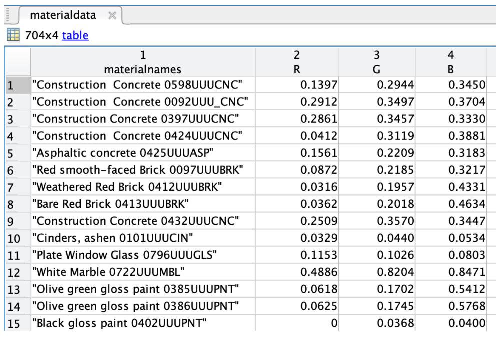
$$\alpha(y) = \frac{\int_{\lambda_1}^{\lambda_2} R_y(\lambda) * E_x(\lambda) * d\lambda}{\int_{\lambda_1}^{\lambda_2} E_x(\lambda) * d\lambda} \qquad (5.1)$$

where,
$\alpha(y)$ is the spectrally weighted albedo for material y
R_y is the spectral reflectance of the chosen material (%)
E_x is the spectral irradiance (W/m^2)
$\lambda_1$and λ_2 define the wavelengths (m) in the spectrum used as the limits of integration

In the simulations used in this study, the default limits for integration are defined as

- Channel 1: 250-380 nm (UV range)
- Channel 2: 381 - 760 nm (Visible range)
- Channel 3: 761 - 1500 nm (IR range)

These limits can be customised during the simulations within the toolbox to include the effects of spectral responsiveness of the PV cell chemistry. This is because different cell chemistries respond differently to incident spectrum, with some parts of the incident radiation contributing more towards the solar power output than the others. Using equation 5.1, the spectral reflectance data provided by NASA is converted into a RADIANCE compatible format and a sample from this data-set is illustrated in Figure 5.2, with the channels R, G and B containing the default spectrally weighted albedo values of a few man-made materials. As composite materials such as construction concrete could be obtained using different ratios of raw-materials, a material code is assigned to each entity to differentiate similar materials. For the ground, the albedo is user defined and this value will be assigned to the plane defining the ground.


materialdata

704x4 table

	1 materialnames	2 R	3 G	4 B
1	"Construction Concrete 0598UUUCNC"	0.1397	0.2944	0.3450
2	"Construction Concrete 0092UUU_CNC"	0.2912	0.3497	0.3704
3	"Construction Concrete 0397UUUCNC"	0.2861	0.3457	0.3330
4	"Construction Concrete 0424UUUCNC"	0.0412	0.3119	0.3881
5	"Asphaltic concrete 0425UUUASP"	0.1561	0.2209	0.3183
6	"Red smooth-faced Brick 0097UUUBRK"	0.0872	0.2185	0.3217
7	"Weathered Red Brick 0412UUUBRK"	0.0316	0.1957	0.4331
8	"Bare Red Brick 0413UUUBRK"	0.0362	0.2018	0.4634
9	"Construction Concrete 0432UUUCNC"	0.2509	0.3570	0.3447
10	"Cinders, ashen 0101UUUCIN"	0.0329	0.0440	0.0534
11	"Plate Window Glass 0796UUUGLS"	0.1153	0.1026	0.0803
12	"White Marble 0722UUUMBL"	0.4886	0.8204	0.8471
13	"Olive green gloss paint 0385UUUPNT"	0.0618	0.1702	0.5412
14	"Olive green gloss paint 0386UUUPNT"	0.0625	0.1745	0.5768
15	"Black gloss paint 0402UUUPNT"	0	0.0368	0.0400

Figure 5.2: Spectrally Weighted Albedo of Different Materials in the UV, Visible and IR ranges.

With the spectrally integrated albedo data assigned to the three channels, the next step is to define the fourth and fifth parameters of the material definition: **Specularity** and **Roughness**. The first value defines how specular the reflection of light is from the defined surface, while the second parameter highlights the texture of the surface and how that affects the reflection of light. These two parameters vary from material to material and is usually obtained through experiments. However, due to the abundance in material data, these parameters will be assigned within accepted limits. The range of values for these materials according to the RADIANCE documentation Crone [1992] given as:

Plastics:

- Specularity: 0 to 0.07
- Roughness: 0 to 0.02

Metals:

- Specularity: 0.5 to 1
- Roughness: 0 to 0.5

To understand the effect of specularity and roughness on simulation accuracy, these parameters will be randomly varied within this acceptable range in Chapter 6 of this thesis.

For the solar cells, the BRDF data obtained from the GenPRO simulations will be converted into the RADIANCE syntax using the following set of equations (Equations 5.2 to 5.4) and represent the reflectance of the solar cell in the default ranges:

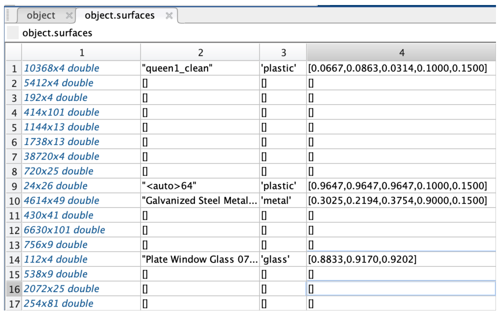
$$R = \frac{\int_{UV} \int_{aoi} \alpha_{UV}(\theta, \lambda) d\theta d\lambda}{\int_{UV} \int_{aoi} d\theta d\lambda} \tag{5.2}$$

$$G = \frac{\int_{vis} \int_{aoi} \alpha_{vis}(\theta, \lambda) d\theta d\lambda}{\int_{vis} \int_{aoi} d\theta d\lambda} \tag{5.3}$$

$$B = \frac{\int_{IR}\int_{aoi}\alpha_{IR}(\theta,\lambda)d\theta d\lambda}{\int_{IR}\int_{aoi}d\theta d\lambda} \tag{5.4}$$

where,
R- Radiance channel 1 reflectance (0-1)
G- Radiance channel 2 reflectance (0-1)
B- Radiance channel 3 reflectance (0-1)
α_{UV} - reflectance in UV range as a function of angle of incidence and wavelength (0-1)
α_{vis} - reflectance in visible range as a function of angle of incidence and wavelength (0-1)
α_{IR} - reflectance in IR range as a function of angle of incidence and wavelength (0-1)
θ - angle of incidence (°)
λ - wavelength of light (nm)

As presented earlier in this chapter, these values lie between 0 and 1; and by default represent the reflectance of a solar cell in the UV, visible and IR ranges of the spectrum respectively. However, if the user desires, these values can be used to represent the reflectance in other ranges of the spectrum by changing the integration limits and this has been explained in appendix C. The solar spectrum is taken in the wavelength limits of 250-1200 nm in accordance with the GenPRO simulations. Thus, the cell level reflectance data can be directly assigned to the cells in the scene. Once the optical properties have been converted into the appropriate format, it can be assigned to the surfaces defining the geometry. Figure 5.3 shows how the data is stored within MATLAB before being exported to the RADIANCE syntax. As seen in Figure 5.3, some material names are awkward, such as "<auto>64". These textures definitions are obtained directly from the .mtl file and therefore do not represent the actual reflective property of the material, but rather the visual aspects of the material. This is also one of the reasons why a material library is separately created, instead of using texture data directly obtained with the geometric information. In this manner, material properties are designated to each surface and opto-geometric data is obtained. The first column contains the matrices defining each polygon or surface, the second column has the material name, the third column highlights the RADIANCE material primitive and the final column has the reflectance data stored for each material.



	1	2	3	4
1	10368x4 double	"queen1_clean"	'plastic'	[0.0667,0.0863,0.0314,0.1000,0.1500]
2	5412x4 double	[]	[]	[]
3	192x4 double	[]	[]	[]
4	414x101 double	[]	[]	[]
5	1144x13 double	[]	[]	[]
6	1738x13 double	[]	[]	[]
7	38720x4 double	[]	[]	[]
8	720x25 double	[]	[]	[]
9	24x26 double	"<auto>64"	'plastic'	[0.9647,0.9647,0.9647,0.1000,0.1500]
10	4614x49 double	"Galvanized Steel Metal...	'metal'	[0.3025,0.2194,0.3754,0.9000,0.1500]
11	430x41 double	[]	[]	[]
12	6630x101 double	[]	[]	[]
13	756x9 double	[]	[]	[]
14	112x4 double	"Plate Window Glass 07...	'glass'	[0.8833,0.9170,0.9202]
15	538x9 double	[]	[]	[]
16	2072x25 double	[]	[]	[]
17	254x81 double	[]	[]	[]

Figure 5.3: Assigning Materials to Surfaces within the Geometry.

This opto-geometric data will be converted into the RADIANCE syntax for performing the ray-tracing calculations and this procedure is explained in Section 5.2.

5.2 DATA CONVERSION AND RAY-TRACING

Section 5.1, illustrated the open source library used to obtain the spectral reflectance data of various materials and explained the methodology adopted to convert this data into the RADIANCE format. This process was also used to convert GenPRO data into the RADIANCE format and assign these properties to the surfaces and solar cells to generate opto-geometric data. In this section, this opto-geometric data will be used as input to convert the files into the RADIANCE format and an explanation of the BRDF equation used in the ray-tracing algorithm will be provided.

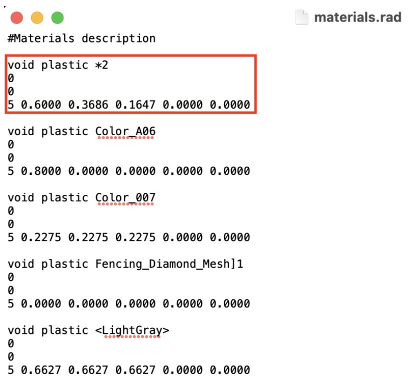
In order to execute the ray-tracing simulation with the BRDF, the opto-geometric data has to be split into the following files:

- A Material definition file
- A Scene definition file
- A Test-points file
- A Sky file

The material file contains the reflectance data and its format and syntax has already been discussed in detail in section 5.1. In order to define the geometries within the scene along with its corresponding material data, the geometry data is stored in the scene definition file as:

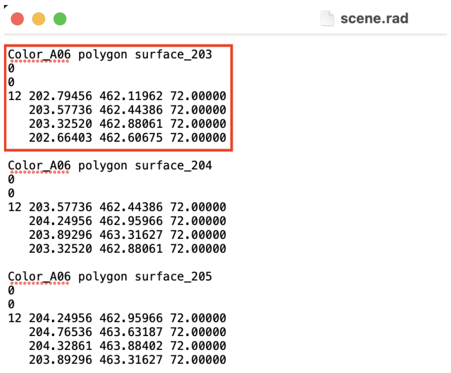
materialname *primitive* **surfacename**
0
0
$3 * n \quad x_1 \quad y_1 \quad z_1$
$x_2 \quad y_2 \quad z_2$
.
.
$x_n \quad y_n \quad z_n$

The **materialname** is the name of the material pre-defined in the material file and RADIANCE follows a chain rule to assign materials to surfaces. Similar to the earlier definition, the *primitive* here represents the type of geometry. A few examples of a RADIANCE geometry primitive would be polygon, sphere, cone, cylinder, etc. As the opto-geometric file consists of surface data in the toolbox, the **Polygon** primitive will be used to define the surfaces in the RADIANCE scene description. Each surface is defined with the help of $3 * n$ parameters, where n represents the number of vertices making up the surface. The values x_n, y_n and $z_n$denote the X, Y and Z coordinates of the n^{th} vertex respectively. Figures 5.4 and 5.5 outline how the data is stored in the material file and scene description file respectively. The red boxes in both images indicate the description of 1 entity in the file, so a material in the first image and a surface in the second. The examples shown in these figures uses data directly assigned from the .mtl file and is only used for representative purposes.



```
#Materials description
void plastic *2
0
0
5 0.6000 0.3686 0.1647 0.0000 0.0000

void plastic Color_A06
0
0
5 0.8000 0.0000 0.0000 0.0000 0.0000

void plastic Color_007
0
0
5 0.2275 0.2275 0.2275 0.0000 0.0000

void plastic Fencing_Diamond_Mesh]1
0
0
5 0.0000 0.0000 0.0000 0.0000 0.0000

void plastic <LightGray>
0
0
5 0.6627 0.6627 0.6627 0.0000 0.0000
```

Figure 5.4: Material Description in the RADIANCE Syntax.



```
Color_A06 polygon surface_203
0
0
12 202.79456 462.11962 72.00000
   203.57736 462.44386 72.00000
   203.32520 462.88061 72.00000
   202.66403 462.60675 72.00000

Color_A06 polygon surface_204
0
0
12 203.57736 462.44386 72.00000
   204.24956 462.95966 72.00000
   203.89296 463.31627 72.00000
   203.32520 462.88061 72.00000

Color_A06 polygon surface_205
0
0
12 204.24956 462.95966 72.00000
   204.76536 463.63187 72.00000
   204.32861 463.88402 72.00000
   203.89296 463.31627 72.00000
```

Figure 5.5: Geometry Description in the RADIANCE Syntax.

The third and final file which will be generated from the defined scene is a "Test-point" file. This file contains the data of the sensors which will be used as the starting point for the backward ray-tracing. A visual representation of a test-point above a cell has been shown in Figure 5.6. The green dot in this figure is the test-point and the blue line is the normal to the surface representing the cell, with the solar cell shown in red. An example file has been portrayed in Figure 5.7 and in each line, the first 3 values defining the Cartesian coordinates of the point and the last 3 values defining the normal of the point. This has been highlighted using the 2 red boxes. Each test point is a point which hovers above a single solar cell at a distance of 0.2 units and thus the plane of array irradiance will be computed at this point. For cells on the rear surface, this point will be present below the cell by the same distance. A single cell can have 4 test-points at the most and the results from each point will be averaged to obtain the irradiance per cell on a module. In this example, a cell with a single test point has been shown.

The reason for considering more than a single test-point is to account for shading phenomenon properly. If only a single point is defined, the shading scenario is binary: A cell is either completely shaded or completely un-shaded. Therefore choosing more than one point allows for a more realistic distribution of irradiance across different cells within a module.

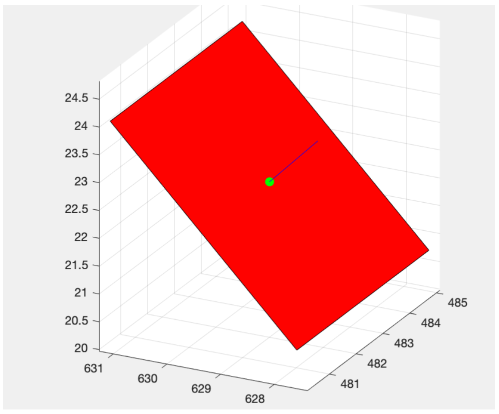
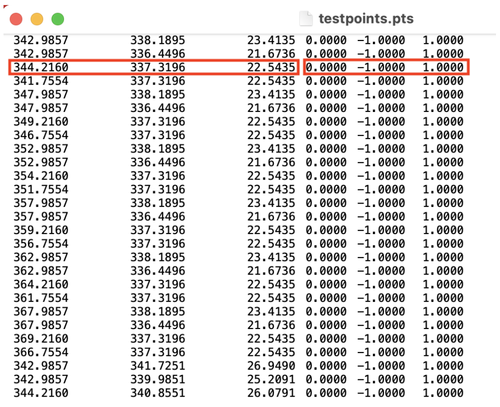


Figure 5.6: Visual Representation of a Test-point.

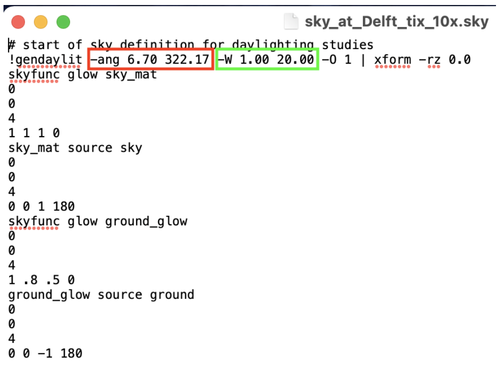


testpoints.pts

```
342.9857        338.1895        23.4135 0.0000 -1.0000  1.0000
342.9857        336.4496        21.6736 0.0000 -1.0000  1.0000
344.2160        337.3196        22.5435 0.0000 -1.0000  1.0000
341.7554        337.3196        22.5435 0.0000 -1.0000  1.0000
347.9857        338.1895        23.4135 0.0000 -1.0000  1.0000
347.9857        336.4496        21.6736 0.0000 -1.0000  1.0000
349.2160        337.3196        22.5435 0.0000 -1.0000  1.0000
346.7554        337.3196        22.5435 0.0000 -1.0000  1.0000
352.9857        338.1895        23.4135 0.0000 -1.0000  1.0000
352.9857        336.4496        21.6736 0.0000 -1.0000  1.0000
354.2160        337.3196        22.5435 0.0000 -1.0000  1.0000
351.7554        337.3196        22.5435 0.0000 -1.0000  1.0000
357.9857        338.1895        23.4135 0.0000 -1.0000  1.0000
357.9857        336.4496        21.6736 0.0000 -1.0000  1.0000
359.2160        337.3196        22.5435 0.0000 -1.0000  1.0000
356.7554        337.3196        22.5435 0.0000 -1.0000  1.0000
362.9857        338.1895        23.4135 0.0000 -1.0000  1.0000
362.9857        336.4496        21.6736 0.0000 -1.0000  1.0000
364.2160        337.3196        22.5435 0.0000 -1.0000  1.0000
361.7554        337.3196        22.5435 0.0000 -1.0000  1.0000
367.9857        338.1895        23.4135 0.0000 -1.0000  1.0000
367.9857        336.4496        21.6736 0.0000 -1.0000  1.0000
369.2160        337.3196        22.5435 0.0000 -1.0000  1.0000
366.7554        337.3196        22.5435 0.0000 -1.0000  1.0000
342.9857        341.7251        26.9490 0.0000 -1.0000  1.0000
342.9857        339.9851        25.2091 0.0000 -1.0000  1.0000
344.2160        340.8551        26.0791 0.0000 -1.0000  1.0000
```

Figure 5.7: Test-Points for the Ray-Tracing Simulation.

The last file to be used as input in the ray-tracing simulation is a sky file, which contains the information about the sky and the position of the sun which is to be used in the ray-tracing. This file is generated using the in-built RADIANCE command ***gendaylit*** which creates a sky based on the Perez Model Perez et al. [1988]. To generate a sky file, a climate file with solar irradiation data such as DNI (dni), DHI (dhi) and sun-position is to be used as input. This climate file can be obtained from weather software such as Meteonorm®. This sky file is defined for every-time step present in the climate file and therefore if hourly data is used for the simulations, 8760 sky files will be generated. A sample of a single sky-file has been depicted in Figure 5.8.



sky_at_Delft_tix_10x.sky

```
# start of sky definition for daylighting studies
!gendaylit -ang 6.70 322.17 -W 1.00 20.00 -O 1 | xform -rz 0.0
skyfunc glow sky_mat
0
0
4
1 1 1 0
sky_mat source sky
0
0
4
0 0 1 180
skyfunc glow ground_glow
0
0
4
1 .8 .5 0
ground_glow source ground
0
0
4
0 0 -1 180
```

Figure 5.8: Sky File Definition for a Single Data-point.

The ***gendaylit*** command uses the position of the sun (its azimuth angle and elevation) along with the DHI and DNI to create the sky definition. The argument *-ang* is used to fix the position of the sun and the *-W* argument is used for using the DNI and DHI data as input. This has been higlighted using the red and green boxes respectively in Figure 5.8. The sun is considered to be a glowing source of direct irradiance in the diffuse sky and this can be seen from the keywords used to describe the sun in the file.

These four files can now be used to perform the ray-tracing by converting these files into a single file called an **octree** file. An octree file is a branched data-structure which organises data in such a way that it can be stored and retrieved quickly, thus minimising computational time Meagher [1995]. This file can be obtained using the RADIANCE command ***oconv***. With the help of the octree file, the ray-tracing can now be carried out.

The ray-tracing is carried out with the help of the RADIANCE command ***rtrace***, and involves five input parameters:

- Ambient bounces: This parameter defines the number of reflections a ray undergoes from multiple surfaces before it reaches the source.
- Ambient divisions: The error in calculating the indirect illuminance using the Monte Carlo method will be inversely proportional to the square root of this value.
- Ambient resolution: This parameter defines the resolution of the file below which the error in simulation starts increasing.
- Ambient accuracy: The value of ambient divisions will determine the error in results due to the interpolation of indirect illuminance values.
- Ambient Super Samples: This parameter is used only for those ambient divisions which have a significant impact on the results.

During the ray-tracing, RADIANCE uses an improved version of the Ward-Dür BRDF model Geisler-Moroder and Dür [2010], to generate a BRDF for each surface in the scene and is expressed using the equation

$$f_w(\theta_l,\phi_l;\theta_v,\phi_v) = \frac{\rho_s}{\pi * \alpha * \beta} * \exp(-tan^2\delta * (\frac{cos^2\phi}{\alpha^2} + \frac{sin^2\phi}{\beta^2})) * \frac{1}{4 * \sqrt{(cos\theta_l * cos\theta_v)}} \tag{5.5}$$

where
f_w is the BRDF, which is a function of the incident ray and viewer ray direction
ρ_s is the specularity of the surface
α and β are the roughness values perpendicular to plane of surface
δ and ϕ are the zenith and azimuth angles of the halfway vector
θ_l and ϕ_l are the zenith and azimuth angles of the incident ray vector
θ_v and ϕ_v are the zenith and azimuth angles of the reflected ray vector or direction of the viewing angle

The halfway vector is defined as the vector along which maximum specular reflection is observed from the surface. The three values ρ_s, α and β are obtained from the five material parameters defined in section 5.1. This BRDF model (equation 5.5) is derived from the Ashikhmin-Shirley BRDF model Shirley and Ashikhmin [2000] and is implemented in RADIANCE due to its ability to model anisotropic reflection better, especially at grazing angles of incidence of light.

With this data, the ray-tracing simulation is carried out for the PVMD Monitoring station to determine the accuracy of the proposed workflow and methodology. This is discussed in detail in Chapter 6.

5.3 SUMMARY

This chapter focused on the fourth objective of this project, and illustrated three things: 1) the source for material reflectance data; 2) the generation of opto-geometric data and data conversion into the RADIANCE syntax; 3) the explanation of the BRDF equation used in the simulations. The reflectance data for different materials were procured from the open-source ECOSTRESS Library of NASA, which has the spectral reflectance for a wide range of materials across a very large wavelength span (from 300 nm to 15000 nm). This data is then made RADIANCE syntax compatible by converting the spectral reflectance data into spectrally weighted albedo, which is then easily assigned to the materials to obtain opto-geometric data. A similar approach is followed for converting solar cell reflectance data from the GenPRO simulations to be used in the ray-tracing process.

Secondly, in order to perform the ray-tracing, the opto-geometric data from section 5.1 is converted into three files: one with the optical information of each material, the second with geometric information of each surface and the third file with the information on the sensors which will be used as the starting points in the backward ray-tracing simulation. Along with these three files, a sky file is generated using a climate file as input, which contains information on the position of the sun and the amount of energy incident at the desired location.

Finally, these four input files from section 5.2 were converted into an octree file in-order to speed up the ray-tracing problem and will be used as the sole input file. This octree file is used as an input to perform the ray-tracing simulation, which involves the use of the Ward-Dür BRDF model to accurately determine reflected irradiance. In the next chapter, the validation of the proposed model will be done by comparing the results of the simulation for the PVMD monitoring station.

6 RESULTS AND VALIDATION

This chapter deals with the fifth objective of the thesis, which is to validate the proposed workflow and determine the accuracy of the simulations. The accuracy of the workflow depends also on the settings adopted for the ray-tracing simulations and the value of these parameters are very scene dependent and thus need constant updating. However, as the toolbox is meant to be an all-purpose software, the value of these parameters have to be fixed.

Therefore, this chapter begins with the optimisation of the RADIANCE parameters and their contribution to the accuracy of the result will be presented, in section 6.1. Once settings have been fixed, simulations will be run for the 3D Scene of the PV Monitoring station located within the TU Delft campus and compared with the monitored data to analyse the accuracy of the ray-tracing program. This will be highlighted in section 6.2. Afterwards, section 6.3 will discuss the effect of the specularity and roughness parameters which have been omitted in the simulations so far. Finally, this chapter is concluded with a summary in section 6.4.

6.1 OPTIMISING RADIANCE PARAMETERS

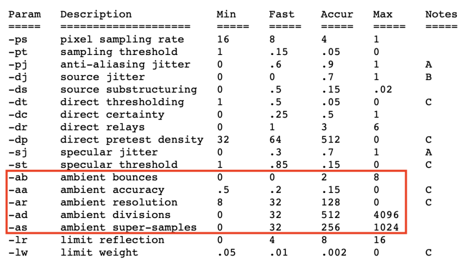
As explained in section 5.2, five parameters dictate the accuracy of the ray-tracing simulation. These five variables are:

- Number of ambient bounces
- Ambient Accuracy
- Ambient Resolution
- Ambient Divisions
- Ambient Super Samples

The definitions of these five values have been provided in chapter 5 of this report. The values to be used for the simulations depend on the geometries present in the scene and the number of sensors (ray-tracing points) used in the simulations. As mentioned earlier, the toolbox is meant to be an all-purpose software and therefore should not request user intervention to decide the ray-tracing parameters. Therefore, the ideal scenario would be to perform simulations for a large number of samples with varying details of geometries and number of sensor points to determine the optimal value for the software. However, due to time constraints, the optimisation has been carried out by considering the default settings for accurate rendering recommended by RADIANCE and vary the values around this point to determine if a better setting can be used.

According to the RADIANCE websiteWard [1997] , the values recommended for accurate rendering is provided below in the form of a list and the possible set of values for other settings have been listed in Figure 6.1:

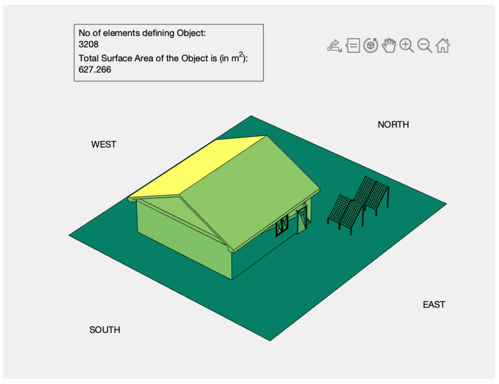
- Ambient bounces: 2
- Ambient Accuracy: 0.15
- Ambient Resolution: 128
- Ambient Divisions: 512
- Ambient Super Samples: 256



Param	Description	Min	Fast	Accur	Max	Notes
-ps	pixel sampling rate	16	8	4	1	
-pt	sampling threshold	1	.15	.05	0	
-pj	anti-aliasing jitter	0	.6	.9	1	A
-dj	source jitter	0	0	.7	1	B
-ds	source substructuring	0	.5	.15	.02	
-dt	direct thresholding	1	.5	.05	0	C
-dc	direct certainty	0	.25	.5	1	
-dr	direct relays	0	1	3	6	
-dp	direct pretest density	32	64	512	0	C
-sj	specular jitter	0	.3	.7	1	A
-st	specular threshold	1	.85	.15	0	C
-ab	ambient bounces	0	0	2	8	
-aa	ambient accuracy	.5	.2	.15	0	C
-ar	ambient resolution	8	32	128	0	C
-ad	ambient divisions	0	32	512	4096	
-as	ambient super-samples	0	32	256	1024	
-lr	limit reflection	0	4	8	16	
-lw	limit weight	.05	.01	.002	0	C

Figure 6.1: Possible Configurations for the RADIANCE Parameters.

To understand the effect of different geometries and number of samples on the simulation time and accuracy, two different scenes were chosen. The first scene chosen consists of 4 panels placed in portrait orientation beside a house, facing south and is shown in Figure 6.2. Each cell within the modules have four sensor points each for the simulations. In the second scene, 3 modules have been placed behind a building as shown in Figure 6.3 , facing south and in landscape orientation. Each cell has 4 sensor points again, similar to the previous case.



No of elements defining Object:
3208
Total Surface Area of the Object is (in m^2):
627.266

Figure 6.2: Scene used for First Set of Simulations

The benchmark setting used to compare the simulation time and results are a combination of the "ACCUR" and "MAX" settings depicted in Figure 6.1 and is as follows:

- Ambient Bounces: 2
- Ambient Accuracy: 0.15
- Ambient Resolution: 128
- Ambient Divisions: 4096
- Ambient Super Samples: 1024

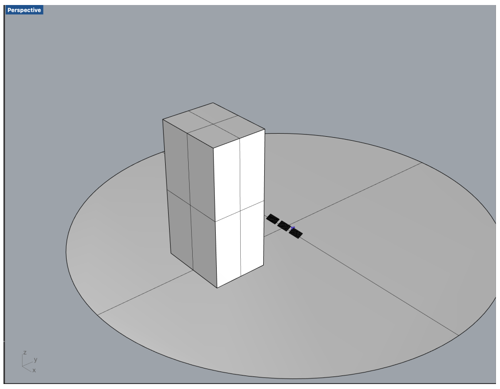


Figure 6.3: Scene used for Second Set of Simulations

For both the simulations, the albedo of the ground was taken as 0.3 and material properties for the surfaces were directly assigned from the .mtl file. This was because the material definitions were vague in the geometry file and the use of these scenes were just to compare computational time and record variation in accuracy with respect to results from the benchmark setting. The simulations were carried out for the summer day of 21st of June, and this instant was chosen as the sun is out for the longest time and thus would help us determine the maximum simulation time required to simulate one day of the year. The variation of results has been shown in Table 6.1, where the global variations observed in the simulated results have been arranged in the decreasing order of simulation time.

Table 6.1: Comparison of Simulation Time and Accuracy for the First Scene.

Setting	Ambient Bounces	Ambient Accuracy	Ambient Resolution	Ambient Divisions	Ambient Super Samples	Simulation Time (s)	Max Variation (%)	Min Variation (%)
Setting 1	3	0.15	128	512	256	420.46	3.1	-0.3
Accur	2	0.15	128	512	256	90.09	3.2	0.1
Setting 2	2	0.2	128	512	256	70.51	1.3	-0.4
Setting 3	2	0.2	32	512	256	40.85	0.2	-0.47
Setting 4	4	0.2	32	32	32	14.98	1.5	-1.6
Setting 5	3	0.2	32	32	32	9.643	1.35	-1.7
Setting 6	2	0.2	32	32	256	5.86	2.1	-1.66
Setting 7	2	0.2	32	32	32	5.6	1.32	-1.2

It can be seen that the variation in results is not large over the wide range of settings used here. This could be due to the simple nature of the scene used in the simulations. However, it can be seen that the number of ambient bounces and the ambient resolution affect the simulation time heavily and therefore these values have to be chosen towards the faster settings in the simulations. As the variations in results are very similar, a second simulation was carried out with a different scene to understand the effect of a different geometry on the results. The variation in accuracy of results and computational time has been similarly recorded for the second scene and the results have been provided in Table 6.2 , with each set of settings arranged in the descending order of simulation time until it diverges again similar to the previous case. All the errors shown in this table is again, with respect to the results produced in the benchmark simulation and thus the benchmark simulation is not included here.

Table 6.2: Comparison of Simulation Time and Accuracy for the Second Scene.

Setting	Ambient Bounces	Ambient Accuracy	Ambient Resolution	Ambient Divisions	Ambient Super Samples	Simulation Time (s)	Max Variation (%)	Min Variation (%)
Setting 1	3	0.1	128	512	256	1010	2.58	1.36
Accur	2	0.15	128	512	256	19.5	2.03	0.85
Setting 2	2	0.2	128	512	256	14.2	2.04	1.02
Setting 3	2	0.2	32	512	256	1.96	2.32	0.8
Setting 4	2	0.2	32	32	256	0.31	10.81	-5.21
Setting 5	2	0.2	32	32	32	0.3	7.62	-5.61
Setting 6	3	0.2	32	512	256	5.01	2.49	0.95

It is visible that the accuracy again does not vary so much for the settings which are a combination of fast rendering and accurate rendering values. This was also observed in the first scene, where such a combination also had minimal deviation from the benchmark results. However, it is evident that the results diverge very quickly in the second scene for very fast simulations settings, which was not the case in the first. Therefore, a middle ground was reached and the following values will be chosen as the ray-tracing settings for the PVMD Toolbox:

- Ambient Bounces: 2
- Ambient Accuracy: 0.2
- Ambient Resolution: 32
- Ambient Divisions: 512
- Ambient Super Samples: 256

However, as suggested at the start of this chapter, it would be ideal to perform simulations on a large set of samples and a variety of complexities to zero-in on the best possible combination for the simulations.

6.2 COMPARISON WITH DATA FROM THE PV MONITORING STATION

In the previous section, the RADIANCE settings used for carrying out the ray-tracing simulations were fixed for use in the toolbox. In this section, these settings will be used to perform the ray-tracing simulations for the 3D scene of the PV monitoring station present within the TU Delft Campus and the results from the simulations will be compared with real-time monitored data to compute the accuracy of the simulations and the possible sources of error in the model.

The 3D scene for the simulations was obtained directly from TU Delft's library in the form of a grasshopper file, and this file is compatible with RHINO. Therefore, this file was opened in RHINO and the geometric entities were baked to obtain the 3D model of the scene. As the input file also contains the reflectance data of each material assigned to the surfaces in the model, this data was directly exported into the input file for the toolbox. Even in this set of simulations, the values of specularity and roughness have been neglected for the ray-tracing. Figure 6.4 shows the 3D scene which was used for the simulations, with the position of the modules highlighted in the red circle present in the figure. The specifications of the modules present in the scene have been provided in the form of Table 6.3.

The data used for validation from the PV Monitoring station has been recorded from 19th August 2020 to the 12th of March 2021. Data recorded earlier has not

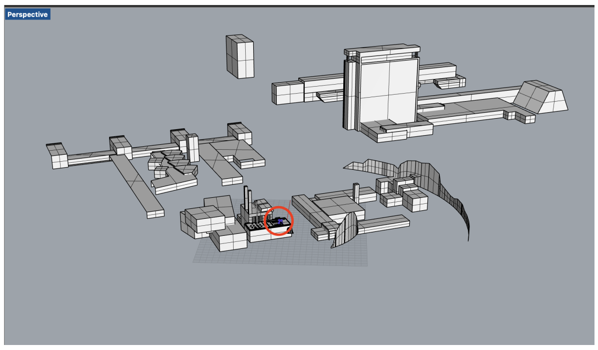


Figure 6.4: 3D Model used for Validation, with the modules highlighted in the red circle.

Table 6.3: Module Specifications at the PV Monitoring Station.

Parameter	Value
Module Tilt	31 +- 1°
Module Azimuth	0 +- 1°(Facing South)
Number of Sensors per module in the ray-tracing study	4
Cell Type (in the ray-tracing)	Perfect Absorbers

been considered due to the fact that the monitoring station was under maintenance and reliable data was not available. Since the 12th of March 2021, the settings used for measuring the POA irradiance were modified at the site and thus not included in this study. As this data only includes data from the start of Autumn to the start of spring, the results have to be taken with a grain of salt as the data from the summer months have not been included in this study. For the study, as shown in Table 6.3, the cells have been considered as perfect absorbers.

The results shown in Figure 6.5 displays a comparison of the simulated amount of irradiance versus the measured value from the station on the 19th of August, 2020. As seen from the image, the simulated irradiance closely follows the measured value of irradiance on the day. There are two points on the plot where the simulated irradiance is lower than the measured value, and this has been observed during hours 13 and 17 respectively. As seen from the scene, there is an object big enough to shade the panels on both occasions, located in the western side of the scene. As this object is not completely defined, this could mean that it accounts for shading rather than the reflected part of the irradiance.

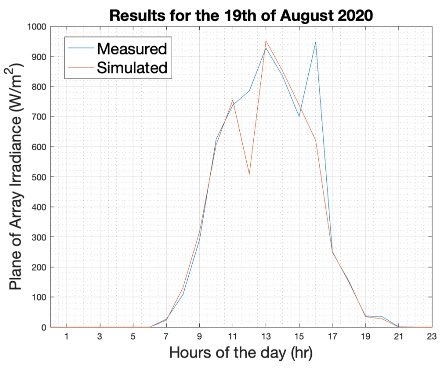


Figure 6.5: Simulated Irradiance vs Measure Irradiance on 19th August 2020.

Such data was found for the entire length of the data-set. As it becomes cumbersome to plot the value for each day this way, the errors have been plotted in Figure 6.6. This plot represents the absolute error encountered in the entirety of the simulations. There are a few **large spikes** which are observed in the simulations, and this has been seen for almost **250 data-points** in the simulation (**5% of the total data-set**). These data-points represent times which are early in the morning or early into the evening, and the readings from the monitoring station are not very reliable in this case. This is because the pyrheliometer is located close to a very white wall which is extremely reflective and can reflect light when the Sun is at very low angles. Moreover, the presence of the very tall EWI building in the campus can cast shadows which will lower the measured irradiance greatly, particularly in the mornings. Thus, the DNI and DHI readings from the site could be higher or lower on such occasions. Excluding such data-points due to their low reliability, the **mean error** observed across the data-set is **0.0097** W/m^2 , with an observed **standard deviation** of **8.3412** W/m^2. These values are well within acceptable ranges of error and therefore show that the software is accurate in modelling the irradiance from the panels.

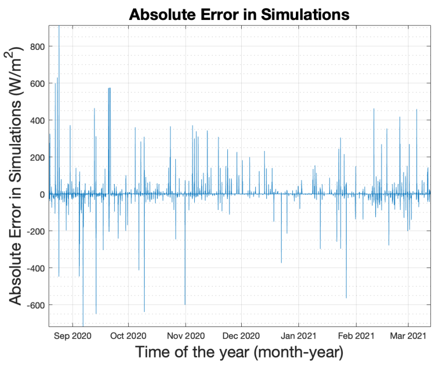


Figure 6.6: Absolute Errors Observed in Simulations for given Data-set.

Similarly, the errors are plotted in the form of percentages and presented in Figure 6.7. Again, the large spikes are observed for data-points similar to the ones observed in the previous case, and account for the inaccuracies in monitoring data early in the mornings or into the evenings in the winter months. Excluding these data-points, the observed **mean percentage error** is **1.89%**, with a **percentage standard deviation** of **12.49%** observed in the data. These values are again within acceptable limits and therefore prove that the RADIANCE ray-tracing engine is capable of producing very accurate results.

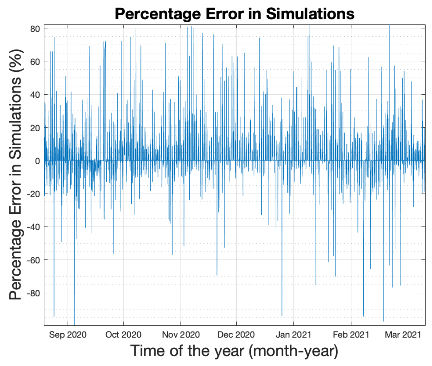


Figure 6.7: Percentage Errors Observed in Simulations for given Data-set.

Unfortunately, due to the unavailability of data, the ray-tracing results for summer months are not presented, which would have given a clearer picture of how accurate the results are when inaccuracies such as shading or reflected irradiance at low sun angles do not exist. Moreover, as seen from the scene defined in Figure 6.4, the level of detail is very low on the objects surrounding the modules and such inputs could also vary the results. However, Figure 6.5 shows that the software package is more than capable of producing extremely good results and thus suitable for use in the toolbox. The results from the graphs shown in Figures 6.6 and6.7 are summarised and presented in Table 6.4.

Table 6.4: Summary of Results for the Validation Simulation.

Parameter	Value
Mean Error (W/m^2)	0.0097
Standard Deviation in Error (W/m^2)	8.34
Mean Error (%)	1.89
Percentage Standard Deviation in Error (%)	12.49

So far, we have neglected the parameters of specularity and roughness from the ray-tracing simulations. In the next section, the effect of these parameters will be discussed in detail to study their impact on the results and the simulations.

6.3 EFFECT OF SPECULARITY AND ROUGHNESS

In all the simulations that have been carried out in this project up until this point, the last 2 values used to define the material reflectance in the RADIANCE syntax have been neglected: the specularity and the roughness. Each surface exhibits two types of reflective behaviour: specular and diffuse behaviour. A completely specular surface acts like a mirror and is extremely glossy, while a completely diffuse surface scatters light equally in all directions and can be called a matte surface. In the RADIANCE environment, specularity for a material ranges from 0 to 1 and represents how specular a surface is, with 0 being completely diffuse and 1 being completely specular. The roughness on the contrary is used to define the texture of the chosen surface and the nature of indentations on it. This parameter also varies from 0 to 1, with 0 representing completely smooth surfaces and 1 representing a very rough surface.

In real life, it is extremely difficult to obtain open source data on specularity and roughness for several materials as each surface is unique. Therefore, these values are usually obtained experimentally using spectrophotometers. However, as this is impossible to do for the wide collection of materials included in the material library, these values were neglected for the simulations. This section will thus discuss the impact they can have on the irradiance values.

As mentioned in Chapter 5, the two kinds of material primitives considered in the toolbox to define general materials (Plastic & Metal) have acceptable ranges of specularity and roughness, and these are provided again as a refresher:

- Plastic: Specularity: 0 - 0.07; Roughness: 0 to 0.02
- Metal: Specularity: 0.5 to 1; Roughness: 0 to 0.5

Thus, two sets of simulations are carried out to understand how they play a role in manipulating the results. The first one considers intermediate values within the acceptable range, while the second one considers a scenario with the maximum possible value assigned to each material. Again, the 3D Scene used for validation in section 6.2 will be used here with updated specularity and roughness values. The values used for both scenarios have been provided in Table 6.5.

Table 6.5: Specularity and Roughness values used in the Simulations.

Parameter	Scenario 1	Scenario 2
Specularity (Plastic)	0.035	0.07
Roughness (Plastic)	0.01	0.02
Specularity (Metal)	0.75	1
Roughness (Metal)	0.25	0.5

With the values provided in Table 6.5, the ray-tracing was performed to visualise its effect on the results. The comparison of results produced in the first scenario has been provided in the form of a plot in Figure 6.8, where the variation in results with respect to the original results obtained in section 6.2 have been plotted.

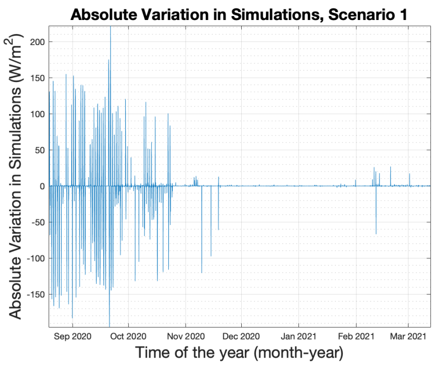


Figure 6.8: Variation in Results with respect to Original Results, Scenario 1.

From Figure 6.8, it can be observed that there is an overestimation or underestimation of irradiance, especially in the summer and autumn months. In the winter months, the variations are very low and we observe a few spikes again during the start of spring. Most of these very large positive variations have been observed to lie in the middle of the day, while most of the negative variations occur early in the morning or late in the evening. The positive variations can be attributed to the fact that the specular and roughness properties which have been introduced reflect more direct light in the middle of the day due the to Sun being brighter in the summer and early Autumn months. However, early in the mornings or late in the evenings, the sun's altitude is too low and therefore the angle of reflected light from the surface is too low to reach the panels and thus not accounted for. During the winter months, as the incident energy from the low altitude Sun is much lower, this does not affect the results as much. This pattern repeats itself during the start of spring, as seen on the right hand side of Figure 6.8.

The same set of variations are also observed in Scenario 2, and this has been plotted in Figure 6.9. The difference between the two scenarios has also been plotted and shown in Figure 6.10 as the results seem very similar. As explained by this figure, it is evident that the variation in results between the 2 scenarios is very minimal, but the effects of specularity and roughness on the results are very fascinating to observe.

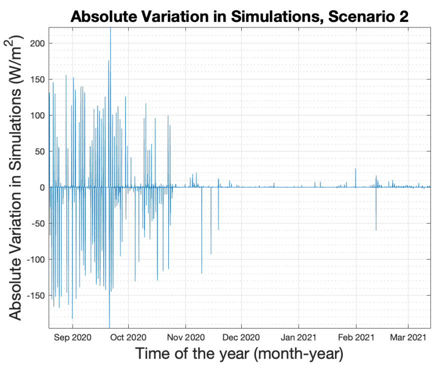


Figure 6.9: Variation in Results with respect to Original Results, Scenario 2.

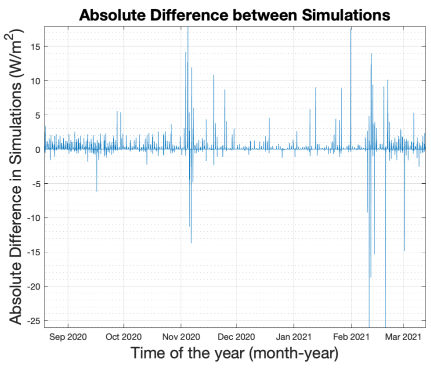


Figure 6.10: Difference in Results from Both Scenarios.

In conclusion, it is clear that the specularity and roughness do have an impact on the results. Therefore, it is recommended that these values be determined and assigned to the materials used for the simulations.

6.4 SUMMARY

This chapter focused on validating the developed workflow, the ray-tracing program and understanding the effects of the input parameters and RADIANCE settings on the results. In section 6.1, the optimisation of the RADIANCE parameters used in the ray-tracing calculations was completed, in order to reduce computational time while preserving the accuracy of results. It was observed that the RADIANCE simulations can be quickly completed while maintaining the accuracy for values which are a combination of the prescribed accurate and fast rendering settings, especially for very simple scenes. It is also recommended that such simulations be carried out for a variety of geometries and complexities to narrow down the problem and determine the ideal setting. This could not be carried out due to time restrictions.

In Section 6.2, the ray-tracing simulations were performed for the 3D scene of the PV monitoring station situated in the TU Delft campus, and results were compared with real-time measured data between the months of August 2020 and March 2021. There were a few large errors observed for data-points taken early in the morning or late in the evening, due to the position and surroundings of the sensor used to measure the data. However, for the reliable data- points, the mean error was computed to be 1.89%, with a standard deviation of 12.49%. These values prove that the RADIANCE ray-tracing engine is a good software package which produces reliable results. However, these results could be affected by the quality of the input model and therefore care must be taken to ensure complete models are included in the simulations.

Finally, section 6.3 discussed the effects of the specularity and roughness values on the results, which were omitted up until this point. The effect of these parameters is more prominent in the summer months, as specular surfaces can reflect more incident energy onto the modules. However, this effect is negligible in the winter months as the sun's altitude is too low to produce enough energy which can be reflected onto the panels. It was also observed that the variations in results between the two simulations involving these parameters is marginal and therefore the recommendation is that these values be determined and assigned to the materials used in the simulations.

7 THE NEW POWER OF THE TOOLBOX

This small chapter is dedicated to the final objective of this report, which is to present a "case-study" which will explore the new functionalities of the improved toolbox. As explained at the start of this report, the primary objective of the project is to accurately model and visualise the effects of reflected irradiance and shading on solar panels, similar to panels installed in urban environments. The precision of the model, the effects of various input parameters and the sources for errors have been extensively put forth in chapter 6 of this report. To visualise this new power of the toolbox, a simple scenario is considered in this chapter where a wall is placed beside a set of panels and its reflective properties modified.

7.1 MODELLING REFLECTED IRRADIANCE AND SHADING PERFORMANCE

A visual description of the scene is provided in Figure 7.1 as shown below. We will call this simulation a "Case-Study". Two modules are placed beside each other in a single row and a wall is placed away from it towards its west. For this simple scenario, the solar cells are considered to be perfect absorbers, and no specularity or roughness is assigned to the materials in the scene. As highlighted in chapter 5, the toolbox can directly assign the reflective property of the solar cell by converting the GenPRO simulation output. The specifications of the modules used for the simulations have been tabulated and provided in Table 7.1.

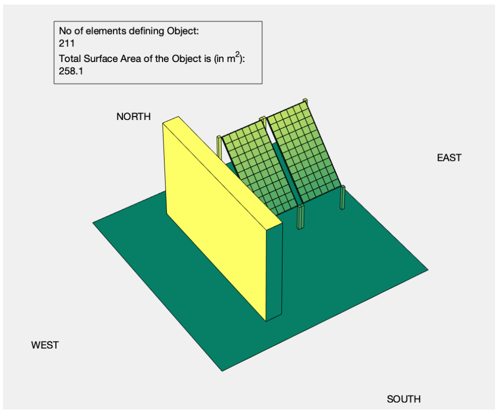

No of elements defining Object:
211
Total Surface Area of the Object is (in m^2):
258.1
NORTH
EAST
WEST
SOUTH

Figure 7.1: Simple Scenario for Case-Study Simulations.

We consider 3 scenarios to visualise the new functions of the toolbox: The first two consider the modelling of reflected irradiance and the last one considers a shading simulation. In the first reflected irradiance model, the wall is placed at the same distance from the panels and the material definition is changed. In the second reflected irradiance model, the wall position is changed across simulations to understand the effect of reflected irradiance, while assigning it the same material. The materials considered for the simulations are Bare-red brick and Aluminium. The material reflectance values in the UV, visible and IR range has been provided

Table 7.1: Module Specifications for the Case Study.

Parameter	Value
Number of Cell Rows	12
Number of Cell Columns	6
Cell Length (cm)	12.5
Cell Width (cm)	12.5
Cell Spacing (cm)	0.2
Edge Spacing (cm)	1
Module Thickness (cm)	2.5
Module tilt (deg)	45
Module Azimuth (deg)	0 (Facing South)
Height of Modules from ground (lower point) (cm)	50
Number of modules in a row	2
Number of rows of modules	1
Module Bifaciality	NIL

in Table 7.2 and visually depicted in Figure 7.2. As observed, Aluminium has a greater reflectance across the 3 ranges of the solar spectrum by a large margin. The specifications of the wall used in both simulations for the reflected irradiance have been provided in Table 7.3 and Table 7.4 respectively.

Table 7.2: Material Properties Used in Case Study.

Material	UV Reflectance (0-1)	Visible Reflectance (0-1)	IR Reflectance (0-1)
Bare Red Brick	0.0362	0.2018	0.4634
Aluminium Metal	0.3848	0.5243	0.7212

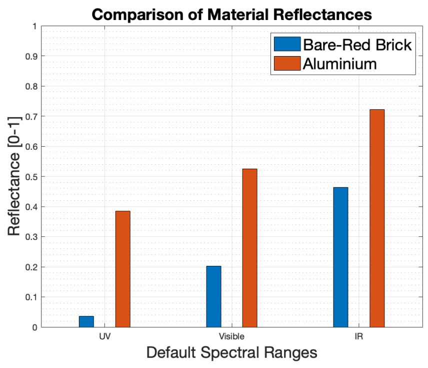


Figure 7.2: Material Reflectances used in the Case Study.

In this case study, the panels are facing south with a tilt of 45°, and the location used for the simulations is Delft, the Netherlands. The chosen year for the simulations is 2019, and the results are plotted for the 21st of June at 0800 hours. This time has been chosen because the wall is to the west of the modules and it is easier to visualise reflected irradiance in this case. However, for modelling the shading across the module, the results are presented for the 5th of January at 3 PM. This is due to the fact that the sun is low enough on the horizon to cause shading in this simple scene. The parameters/ RADIANCE settings used for the simulations were briefly discussed in chapter 6 of this report and the settings chosen from that chapter will be used for this study as well.

Table 7.3: Wall Properties used for First Reflected Irradiance Simulation.

Parameter	Simulation 1	Simulation 2
Wall Dimensions (cm)	300x200x30	300x200x30
Distance from Panels (cm)	50	50
Wall Material	Bare-Red Brick	Aluminium Metal

Table 7.4: Wall Properties used for Second Reflected Irradiance Simulation.

Parameter	Simulation 1	Simulation 2
Wall Dimensions (cm)	300x200x30	300x200x30
Distance from Panels (cm)	50	100
Wall Material	Aluminium Metal	Aluminium Metal

The results for all scenarios in this case study have been given in the form of a set of images, shown from Figures 7.3 to Figure 7.14. Three types of results can be visualised with the help of the toolbox: Plane of Array Irradiance (POA (poa)) available on the panels at a particular time instant (shown in the form of a gradient plot across the different cells), the plot of the Monthly POA irradiance and finally, the plot of the Yearly POA irradiance for the modules.

The computed reflected irradiance for the first scenario is shown in Figures 7.3 and 7.4. As seen from both figures, the scene with the aluminium wall reflects more energy and therefore more POA irradiance is observed. This can be seen from the Irradiance color bar in Figure 7.4, where the irradiance exceeds 250 W/m^2 while in the first scenario only exceeds 200 W/m^2. Thus, the toolbox is able to take into account reflected irradiance in its simulations and calculate POA irradiance on the modules.

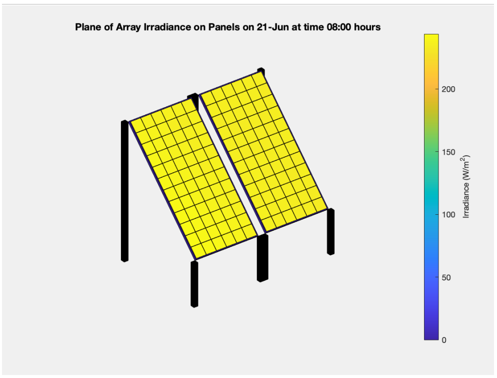

Plane of Array Irradiance on Panels on 21-Jun at time 08:00 hours
200
150
100
50
0
Irradiance (W/m^2)

Figure 7.3: Plane of Array Irradiance in the Scene with Brick Wall placed 50 cm away.

Similarly, in the second set of simulations where the wall is now pushed back to 100 cm away from the modules, it can be seen from Figure 7.5 that the total POA irradiance calculated is lower than the one in the simulation shown in Figure 7.4 and only barely touches $250W/m^2$. This once again emphasises the fact that the tool is able to model the effect of reflected irradiance accurately.

To quantify the contribution of reflected irradiance on the modules, a mock simulation was run where all the materials were considered to be perfect absorbers. This ensures that direct irradiance from the sun is computed and the difference between the total energy computed at that time instant and the direct irradiance will determine the contribution of reflected irradiance in the simulations. Figure 7.6 shows the contribution of reflected irradiance to the total POA irradiance and this data has been split among the default channels (UV, Visible and IR ranges) to visualise how

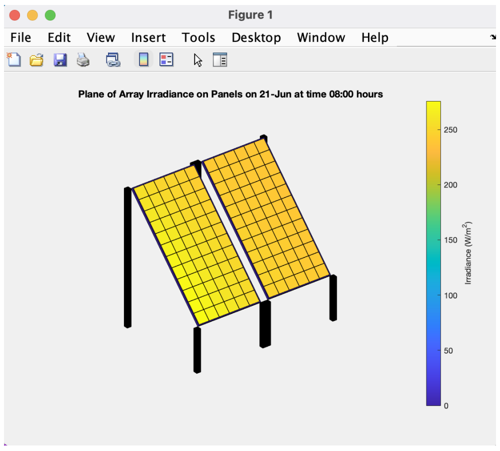


Figure 1
File Edit View Insert Tools Desktop Window Help
Plane of Array Irradiance on Panels on 21-Jun at time 08:00 hours
Irradiance (W/m^2)
250
200
150
100
50
0

Figure 7.4: Plane of Array Irradiance in the Scene with the Aluminium wall placed 50 cm away.

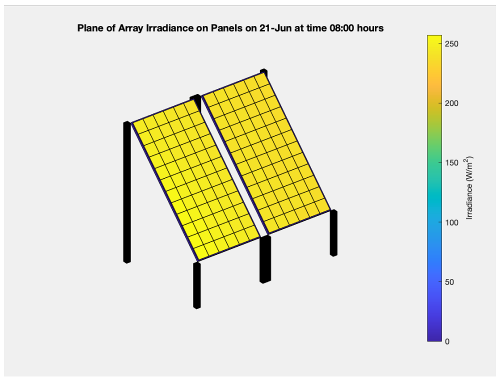

Plane of Array Irradiance on Panels on 21-Jun at time 08:00 hours
Irradiance (W/m^2)
250
200
150
100
50
0

Figure 7.5: Plane of Array Irradiance in the Scene with the Aluminium wall placed 100 cm away.

the material responds to different regions in the solar spectrum. This data is very useful as this can be used to understand the effect of incident light changes as time varies. The total value of reflected irradiance incident on the modules have been shown in Figure 7.7.

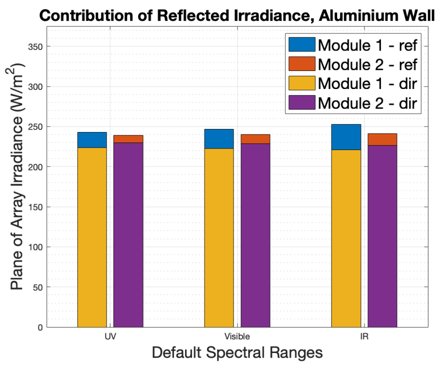


Figure 7.6: Split-up of POA Irradiance across the Solar Spectra in the 3 Spectral Regions, Aluminium wall.

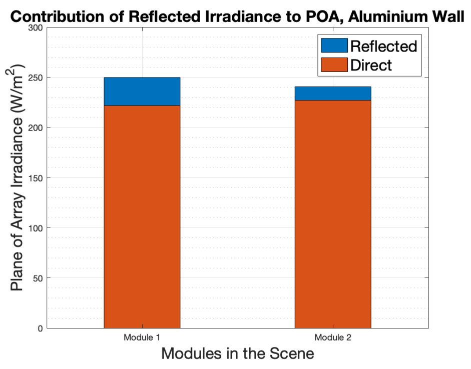


Figure 7.7: Contribution of Reflected Irradiance to the total POA Irradiance, Aluminium Wall.

Similarly, the contribution of reflected irradiance to the total POA irradiance in the case of the brick wall has been depicted in Figure 7.8 and its distribution across the three regions of the spectra is illustrated in Figure 7.9. Again, comparing the Figures 7.6 and 7.9, it is visible that the contribution of reflected irradiance to the total POA irradiance is much higher in the case of the aluminium wall when compared to the scene with the brick wall. This also indicates that the toolbox is correctly predicting reflected irradiance.

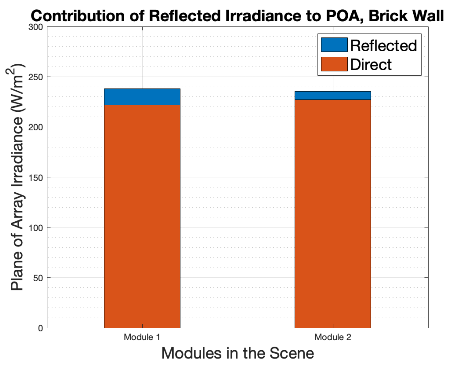


Figure 7.8: Contribution of Reflected Irradiance to the total POA Irradiance, Brick Wall.

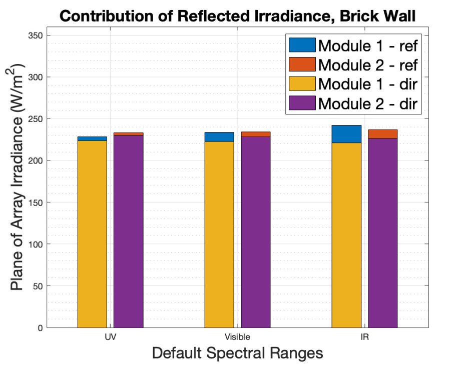


Figure 7.9: Split-up of POA Irradiance across the Solar Spectra in the 3 Spectral Regions, Brick wall.

Moving on to shading performance, the Figures 7.10 and 7.11 show a sample of the shading simulations which could be carried out using the toolbox. The Figure 7.10 show the POA irradiance for the modules on the 5th of January at 3 PM, while Figure 7.11 shows the shading for the panels at 2 PM on the same day. As seen in both figures, the toolbox is now capable of performing shading simulations and depict the spread of varying irradiance across different cells. The color blue indicates low irradiance and the color yellow indicates high irradiance, with green indicating an intermediate level of irradiance. Due to the lower angle of the sun with respect to the horizon at 3 PM, more cells within the modules have been shaded when compared to the panels 2 PM, where only a handful of cells have been shaded due to the wall. Therefore, the toolbox can now model partial shading in modules in a scene and thus is extremely useful for visualising solar PV panel performance in urban scenes where features such as chimneys and dormers can cause shading on panels.

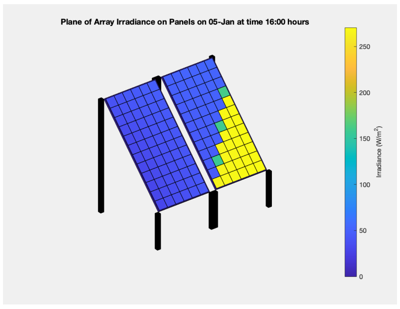

Plane of Array Irradiance on Panels on 05-Jan at time 16:00 hours
250
200
150
100
50
0
Irradiance (W/m^2)

Figure 7.10: Shading Simulations for the Case-Study on the 5th of January at 3 PM.

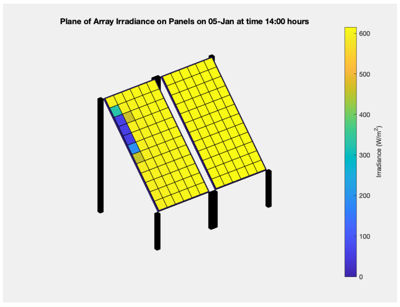

Plane of Array Irradiance on Panels on 05-Jan at time 14:00 hours
600
500
400
300
200
100
0
Irradiance (W/m^2)

Figure 7.11: Shading Simulations for the Case-Study on the 5th of January at 2 PM.

The toolbox can also help view the energy incident at a module level over larger periods of time, such as the energy available over a month or during the whole year. This is done by averaging the energy incident on the cells across the modules for the timestamps present in the month or year. Although this is not a fair indication of how much power the panels will produce during the time interval of interest, it is just used to view the energy available for the panels to produce energy from. Figures 7.12 and 7.13 show the energy available to modules in the month of June and December respectively, while the Figure 7.14 shows the energy available to the modules during the entire year of 2019. The data obtained from these simulations can then be used as the input to compute the DC yield of the solar panels by generating the JV curves of each cell and computing mismatch losses. This is however out of the scope of this project and will be taken up in the future.

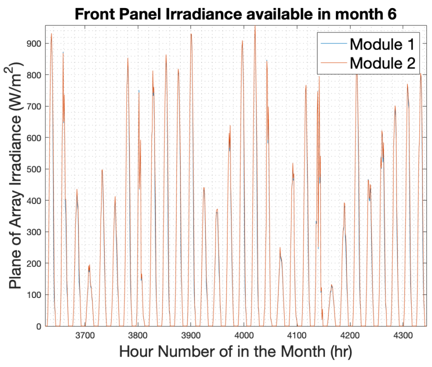


Figure 7.12: Plane of array Irradiance in the Month of June, 2019.

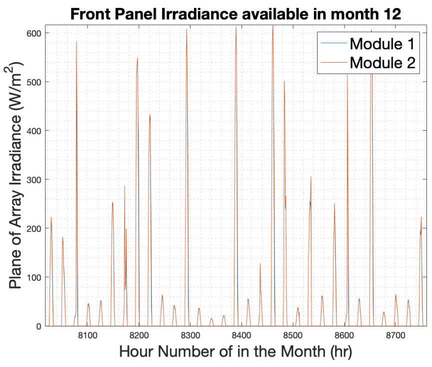


Figure 7.13: Plane of Array Irradiance in the Month of December, 2019.

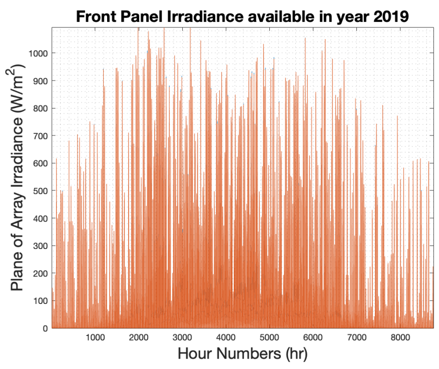
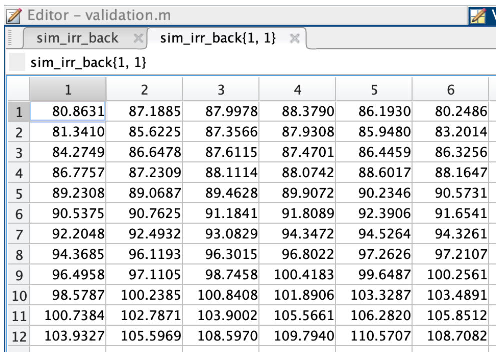


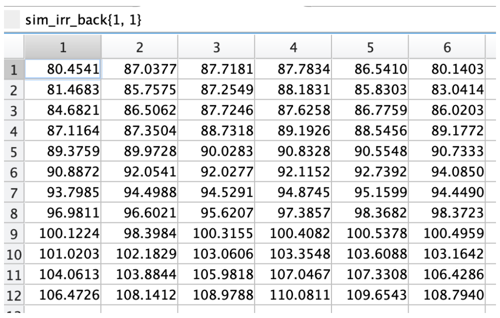
Figure 7.14: Variation in Plane of Array Irradiance in the Year 2019.

This newfound power of the toolbox is extremely convenient to also model the performance of bifacial modules placed in urban environments. Although the toolbox was able to model bifacial module performance earlier, it has been strengthened with this addition as the toolbox can now take into account the presence of urban objects in the scene, which it previously could not. The results for the simulations have been shown in Figure 7.15 and 7.16, where the first case is the rear irradiance of a bifacial module without the wall and the second case with the wall, for the same instant in time. The albedo of the ground has been considered to be 0.3 in both cases. As the difference in gradient is extremely small between the two results due to the fact that they represent rear irradiance, the results have been shown in the form of a matrix. On careful observation, it can be noticed that the values of irradiance on the module are higher when the wall is present and reflecting incident solar radiation to its surroundings. Thus, the toolbox can be used to model rear irradiance on bifacial modules placed in an urban scene as well. Therefore, the new features included in the toolbox have enabled it to model PV potential in urban environments for small urban scenes.



	1	2	3	4	5	6
1	80.8631	87.1885	87.9978	88.3790	86.1930	80.2486
2	81.3410	85.6225	87.3566	87.9308	85.9480	83.2014
3	84.2749	86.6478	87.6115	87.4701	86.4459	86.3256
4	86.7757	87.2309	88.1114	88.0742	88.6017	88.1647
5	89.2308	89.0687	89.4628	89.9072	90.2346	90.5731
6	90.5375	90.7625	91.1841	91.8089	92.3906	91.6541
7	92.2048	92.4932	93.0829	94.3472	94.5264	94.3261
8	94.3685	96.1193	96.3015	96.8022	97.2626	97.2107
9	96.4958	97.1105	98.7458	100.4183	99.6487	100.2561
10	98.5787	100.2385	100.8408	101.8906	103.3287	103.4891
11	100.7384	102.7871	103.9002	105.5661	106.2820	105.8512
12	103.9327	105.5969	108.5970	109.7940	110.5707	108.7082

Figure 7.15: Rear Irradiance on a module without the Wall in its Vicinity.


sim_irr_back{1, 1}

	1	2	3	4	5	6
1	80.4541	87.0377	87.7181	87.7834	86.5410	80.1403
2	81.4683	85.7575	87.2549	88.1831	85.8303	83.0414
3	84.6821	86.5062	87.7246	87.6258	86.7759	86.0203
4	87.1164	87.3504	88.7318	89.1926	88.5456	89.1772
5	89.3759	89.9728	90.0283	90.8328	90.5548	90.7333
6	90.8872	92.0541	92.0277	92.1152	92.7392	94.0850
7	93.7985	94.4988	94.5291	94.8745	95.1599	94.4490
8	96.9811	96.6021	95.6207	97.3857	98.3682	98.3723
9	100.1224	98.3984	100.3155	100.4082	100.5378	100.4959
10	101.0203	102.1829	103.0606	103.3548	103.6088	103.1642
11	104.0613	103.8844	105.9818	107.0467	107.3308	106.4286
12	106.4726	108.1412	108.9788	110.0811	109.6543	108.7940

Figure 7.16: Rear Irradiance on a module when Wall is present.

7.2 SUMMARY

In this chapter, a case study was presented to explore the new capabilities of the toolbox in detail. Firstly, the scene which has been used as the "Case-Study" was defined and the opto-geometric properties of the objects present in the scene were presented. Three scenarios were used to comprehend the new abilities of the toolbox: the first two scenarios described how it models reflected irradiance and how its contribution to the POA irradiance can be computed. In the first simulation, the material properties were varied from a low reflectance material to a high reflectance one while the distance of the obstruction from the panels was changed in the second simulation. In the third scenario, results of a shading simulation were provided to show how the new toolbox takes into account partial shading across different cells and modules. Along with the POA irradiance results for a single time instant, the POA irradiance available at a module level at a monthly and yearly scale was pictorially provided in the chapter. Finally, the capability of the toolbox to model rear irradiance of bi-facial modules was described in detail. With these new additions to the toolbox, users can model the PV potential in urban environments accurately and with ease.

8 CONCLUSIONS & SCOPE FOR FUTURE WORK

8.1 CONCLUSIONS

The objective of this thesis was to develop a workflow and create a new functionality in the toolbox which would help compute the effect of reflected irradiance on panels installed in an urban scene. This new functionality would also include the ability to observe shading across various cells within a module due to surrounding objects. To begin with, a vast study was conducted to find out the methodology adopted in literature and previous research ventures. It was found that solar PV calculations were done primarily for large scenes with minimal emphasis on reflected irradiance, which was completely neglected or assumed to be a constant value. It was also observed that reflected irradiance contributes significantly to the final energy yield of solar panels, especially bifacial ones. Therefore, it was concluded that a smaller scene is to be chosen with very high detailing to its features so as to model reflected irradiance accurately. This also included the incorporation of a BRDF to determine how solar energy is reflected off various surfaces.

Initially, a workflow was determined for use in the toolbox, along with the sources available to import 3D geometries. The workflow described in this report was chosen keeping in mind ease of use and simplicity without compromising heavily on the accuracy of simulations. Google's 3D Warehouse was used as the source for the 3D objects to be used in the simulations and RHINO was used as the CAD software to export the file into a MATLAB readable format. This was achieved by converting the data into the .obj format, which consists of geometric data. Once this geometric data was imported into the toolbox, it was refined in order to eliminate ill-defined surfaces. Panels were then assigned to the scene and this can be done in two ways: A panel could be placed on an existing surface of the 3D object or beside the object itself.

Optical properties were assigned to the corresponding surfaces in the scene, in the RADIANCE syntax. The spectral reflectance of various materials was obtained from NASA's ECOSTRESS Library and converted into spectrally integrated albedo for use in the toolbox. This opto-geometric data was obtained without the use of external softwares/plugins such as Ladybug. The opto-geometric data was then converted into the RADIANCE syntax and used as inputs in the ray-tracing algorithm. The Sky used in the simulations was generated using the Perez model, which was built in the RADIANCE environment. The software tool RADIANCE develops its own BRDF and uses a backward ray-tracing algorithm to accurately compute the irradiance available at the point of interest in the scene.

The developed workflow was then validated by comparing simulated data for the modules present in the PV monitoring station with the real data recorded by the sensors there. It was found that the mean error lies well within the acceptable range, with a mean accuracy of 98% observed in the model. Although large deviations were observed for a few data points, this was attributed to the fact that the scene had lower levels of detail and that the data used for comparison also had errors in measurement due to the location/surroundings of the measuring device. Finally, these results were visualised and scrutinised to determine the effect of different parameters on the results. The effect of Specularity and roughness on the simulations were prominent in the summer months due to the higher altitude and incident energy from the sun during these periods. However, this effect is negligible in the winter months.

Thus, a new and powerful functionality has been added to the PVMD Toolbox which helps us predict the energy available for panels installed in urban environments by accurately modelling reflected irradiance and shading performance.

8.2 SCOPE FOR FUTURE WORK

Although this project has made a positive step towards precisely modelling renewable energy systems, a lot of possibilities were left untouched due to time constraints. To improve on the work carried out in this thesis, the following topics are proposed as add-ons to this project:

- Inclusion of other sources of 3D Geometries as inputs to the toolbox such as LiDAR point clouds, aerial images of a location or other geo-spatial datasets to eliminate the use of external CAD software.
- Conversion of geometric data into syntaxes involving Constructive Solid Geometry (CSG (csg)). This reduces file size while preserving model quality used in ray-tracing. The datasets used here for geometries involve polygonal definitions which occupy a lot of space and increase the file size for complex scenes.
- Inclusion of spectral reflectance data in the form of spectrally responsive albedo rather than spectrally integrated albedo. As different solar cells interact differently with different parts of the incident spectrum, it is essential to convert optical data into this format.
- Conversion of computed irradiance data in the form of a BRDF data-set to accurately determine the energy absorbed by different layers of a solar cell. This is extremely useful for visualising energy absorbed by tandem solar cells, or to observe phenomena such as parasitic absorption by different layers of a solar cell.
- Development of Sensitivity Maps for various modules present in the scene to optimise panel placements in the urban environment.

BIBLIOGRAPHY

Aamir, M., Pu, Y. F., Rahman, Z., Tahir, M., Naeem, H., and Dai, Q. (2019). A framework for automatic building detection from low-contrast satellite images. *Symmetry*, 11(1):1–19.

Andrews, R. W. and Pearce, J. M. (2013). The effect of spectral albedo on amorphous silicon and crystalline silicon solar photovoltaic device performance. *Solar Energy*, 91:233–241.

Baldridge, A. M., Hook, S. J., Grove, C. I., and Rivera, G. (2009). The ASTER spectral library version 2.0. *Remote Sensing of Environment*, 113(4):711–715.

Bergamasco, L. and Asinari, P. (2011). Scalable methodology for the photovoltaic solar energy potential assessment based on available roof surface area: Application to Piedmont Region (Italy). *Solar Energy*, 85(5):1041–1055.

Biljecki, F., Ledoux, H., Du, X., Stoter, J., Soon, K. H., and Khoo, V. H. (2016a). THE MOST COMMON GEOMETRIC and SEMANTIC ERRORS in CITYGML DATASETS. *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 4(2W1):13–22.

Biljecki, F., Ledoux, H., and Stoter, J. (2016b). An improved LOD specification for 3D building models. *Computers, Environment and Urban Systems*, 59:25–37.

Brennan, M. P., Abramase, A. L., Andrews, R. W., and Pearce, J. M. (2014). Effects of spectral albedo on solar photovoltaic devices. *Solar Energy Materials and Solar Cells*, 124:111–116.

Brito, M. C., Gomes, N., Santos, T., and Tenedório, J. A. (2012). Photovoltaic potential in a Lisbon suburb using LiDAR data. *Solar Energy*, 86(1):283–288.

Carneiro, C., Morello, E., and Gilles, D. (2009). Assessment of Solar Irradiance on the Urban Fabric for the Production of Renewable Energy using LIDAR Data and Image Processing Techniques. *Notes*, pages 25–42.

Catita, C., Redweik, P., Pereira, J., and Brito, M. C. (2014). Extending solar potential analysis in buildings to vertical facades. *Computers and Geosciences*, 66:1–12.

Cheng, V., Steemers, K., Montavon, M., and Compagnon, R. (2006). Urban form, density and solar potential. *PLEA 2006 - 23rd International Conference on Passive and Low Energy Architecture, Conference Proceedings*, (September):6–8.

Compagnon, R. (2004). Solar and daylight availability in the urban fabric. *Energy and Buildings*, 36(4):321–328.

Crone, S. (1992). Radiance Users Manual. (November).

Eicker, U., Nouvel, R., Duminil, E., and Coors, V. (2014). Assessing passive and active solar energy resources in cities using 3D city models. *Energy Procedia*, 57:896–905.

Fath, K., Stengel, J., Sprenger, W., Wilson, H. R., Schultmann, F., and Kuhn, T. E. (2015). A method for predicting the economic potential of (building-integrated) photovoltaics in urban areas based on hourly Radiance simulations. *Solar Energy*, 116:357–370.

Fogl, M. and Moudrý, V. (2016). Influence of vegetation canopies on solar potential in urban environments. *Applied Geography*, 66:73–80.

Freitas, S., Catita, C., Redweik, P., and Brito, M. C. (2015). Modelling solar potential in the urban environment: State-of-the-art review. *Renewable and Sustainable Energy Reviews*, 41:915–931.

Geisler-Moroder, D. and Dür, A. (2010). A new Ward BRDF model with bounded albedo. *Computer Graphics Forum*, 29(4):1391–1398.

Gostein, M., Marion, B., and Stueve, B. (2020). Spectral Effects in Albedo and Rear-side Irradiance Measurement for Bifacial Performance Estimation. (July):1–5.

Grena, R. (2012). Five new algorithms for the computation of sun position from 2010 to 2110. *Solar Energy*, 86(5):1323–1337.

Gribnau, D. P. (2020). *Physically based irradiance model for photovoltaic applications*. PhD thesis, Technische Universiteit Delft.

Gueymard, C. A., Lara-Fanego, V., Sengupta, M., and Xie, Y. (2019). Surface albedo and reflectance: Review of definitions, angular and spectral effects, and intercomparison of major data sources in support of advanced solar irradiance modeling over the Americas. *Solar Energy*, 182(February):194–212.

Hofierka, J. and Kaňuk, J. (2009). Assessment of photovoltaic potential in urban areas using open-source solar radiation tools. *Renewable Energy*, 34(10):2206–2214.

Horváth, M., Kassai-Szoó, D., and Csoknyai, T. (2016). Solar energy potential of roofs on urban level based on building typology. *Energy and Buildings*, 111:278–289.

Jacques, D. A., Gooding, J., Giesekam, J. J., Tomlin, A. S., and Crook, R. (2014). Methodology for the assessment of PV capacity over a city region using low-resolution LiDAR data and application to the City of Leeds (UK). *Applied Energy*, 124:28–34.

Jakica, N. (2018). State-of-the-art review of solar design tools and methods for assessing daylighting and solar potential for building-integrated photovoltaics. *Renewable and Sustainable Energy Reviews*, 81(July 2017):1296–1328.

Jakubiec, J. A. and Reinhart, C. F. (2013). A method for predicting city-wide electricity gains from photovoltaic panels based on LiDAR and GIS data combined with hourly Daysim simulations. *Solar Energy*, 93:127–143.

Jochem, A., Hofle, B., Hollaus, M., and Rutzinger, M. (2009). Object detection in airborne LIDAR data for improved solar radiation modeling in urban areas. *International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences*, XXXVIII(Part 3/W8):1–6.

Kodysh, J. B., Omitaomu, O. A., Bhaduri, B. L., and Neish, B. S. (2013). Methodology for estimating solar potential on multiple building rooftops for photovoltaic systems. *Sustainable Cities and Society*, 8:31–41.

Lau, K. K. L., Lindberg, F., Johansson, E., Rasmussen, M. I., and Thorsson, S. (2017). Investigating solar energy potential in tropical urban environment: A case study of Dar es Salaam, Tanzania. *Sustainable Cities and Society*, 30:118–127.

Lindberg, F., Jonsson, P., Honjo, T., and Wästberg, D. (2015). Solar energy on building envelopes - 3D modelling in a 2D environment. *Solar Energy*, 115:369–378.

Lukač, N. and Žalik, B. (2013). GPU-based roofs' solar potential estimation using LiDAR data. *Computers and Geosciences*, 52:34–41.

Lukač, N., Žlaus, D., Seme, S., Žalik, B., and Štumberger, G. (2013). Rating of roofs' surfaces regarding their solar potential and suitability for PV systems, based on LiDAR data. *Applied Energy*, 102:803–812.

M. Martín, A., Domínguez, J., and Amador, J. (2015). Applying LIDAR datasets and GIS based model to evaluate solar potential over roofs: a review. *AIMS Energy*, 3(3):326–343.

Mainzer, K., Killinger, S., McKenna, R., and Fichtner, W. (2017). Assessment of rooftop photovoltaic potentials at the urban level using publicly available geodata and image recognition techniques. *Solar Energy*, 155:561–573.

Meagher, D. (1995). Octree File Format.

Melo, E. G., Almeida, M. P., Zilles, R., and Grimoni, J. A. (2013). Using a shading matrix to estimate the shading factor and the irradiation in a three-dimensional model of a receiving surface in an urban environment. *Solar Energy*, 92:15–25.

Miyazaki, H., Kuwata, K., Ohira, W., Guo, Z., Shao, X., Xu, Y., and Shibasaki, R. (2016). Development of an automated system for building detection from high-resolution satellite images C3 - 4th International Workshop on Earth Observation and Remote Sensing Applications, EORSA 2016 - Proceedings. pages 245–249.

Montes, R. and Ureña, C. (2012). An Overview of BRDF Models. *Technical Report, University of Granada*, pages 1–26.

Nguyen, H. T. and Pearce, J. M. (2012). Incorporating shading losses in solar photovoltaic potential assessment at the municipal scale. *Solar Energy*, 86(5):1245–1260.

Perez, R., Stewart, R., Seals, R., and Guertin, T. (1988). The Development and Verification of the Perez Diffuse Radiation Model. *Sandia National Labs.: Albuquerque, NM, USA; State University of New York: Albany, NY, USA*, (October):176.

Phong, B. T. (1975). The Phong Reflection Model.

Ratti, C., Di Sabatino, S., and Britter, R. (1999). Urban texture analysis with image processing techniques: Winds and dispersion. *Theoretical and Applied Climatology*, 84(1-3):77–90.

Redweik, P., Catita, C., and Brito, M. (2013). Solar energy potential on roofs and facades in an urban landscape. *Solar Energy*, 97:332–341.

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

Robledo, J., Leloux, J., Lorenzo, E., and Gueymard, C. A. (2019). From video games to solar energy: 3D shading simulation for PV using GPU. *Solar Energy*, 193(March):962–980.

Romero Rodríguez, L., Duminil, E., Sánchez Ramos, J., 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.

Ronzino, A., Osello, A., Patti, E., Bottaccioli, L., Danna, C., Lingua, A., Acquaviva, A., Macii, E., Grosso, M., Messina, G., and Rasconà, G. (2015). The energy efficiency management at urban scale by means of integrated modelling. *Energy Procedia*, 83:258–268.

Roopa, E., Shubha, P., and Hebbar, R. (2018). Buildings detection from very high resolution satellite images using segmentation and morphological operations. *Proceedings - 2018 International Conference on Design Innovations for 3Cs Compute Communicate Control, ICDI3C 2018*, pages 106–110.

Russell, T. C., Saive, R., Augusto, A., Bowden, S. G., and Atwater, H. A. (2017). The Influence of Spectral Albedo on Bifacial Solar Cells: A Theoretical and Experimental Study. *IEEE Journal of Photovoltaics*, 7(6):1611–1618.

Salimzadeh, N. and Hammad, A. (2017). High-Level Framework for GIS-Based Optimisation of Building Photovoltaic Potential at Urban Scale Using BIM and LiDAR. *International Conference on Sustainable Infrastructure*, (Zimmerman 2014):14–25.

Santos, T., Gomes, N., Freire, S., Brito, M. C., Santos, L., and Tenedório, J. A. (2014). Applications of solar mapping in the urban environment. *Applied Geography*, 51:48–57.

Shirley, P. and Ashikhmin, M. (2000). An anisotropic phong light reflection model. pages 0–14.

Sonmez, F. (2017). *An albedo Irradiance Model for Bifacial PV Modules Based on LiDAR Data And Ray Casting*. PhD thesis, Technische Universiteit Delft.

Strzalka, A., Alam, N., Duminil, E., Coors, V., and Eicker, U. (2012). Large scale integration of photovoltaics in cities. *Applied Energy*, 93:413–421.

Verso, A., Martin, A., Amador, J., and Dominguez, J. (2015). GIS-based method to evaluate the photovoltaic potential in the urban environments: The particular case of Miraflores de la Sierra. *Solar Energy*, 117:236–245.

Vogt, M., Gewohn, T., Bothe, K., Schinke, C., and Brendel, R. (2018). Impact of using Spectrally Resolved Ground Albedo data for Performance Simulations of Bifacial Modules. *European Photovoltaic Solar Energy Conference and Exhibition*, pages 1011–1016.

Ward, G. (1997). RADIANCE Ray-tracing Settings.

Wavefront-Technologies (1995). Wavefront Material Template Library File Format.

Willenborg, B., Pültz, M., and Kolbe, T. H. (2018). Integration of semantic 3D city models and 3D mesh models for accuracy improvements of solar potential analyses. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, 42(4/W10):223–230.

Xu, S., Huang, Z., Wang, J., Mendis, T., and Huang, J. (2019). Evaluation of photovoltaic potential by urban block typology: A case study of Wuhan, China. *Renewable Energy Focus*, 29(June):141–147.

Yang, Q., Liu, X., and Wu, W. (2020). A hyperspectral bidirectional reflectance model for land surface. *Sensors (Switzerland)*, 20(16):1–19.

Ziar, H., Sönmez, F. F., Isabella, O., and Zeman, M. (2019). A comprehensive albedo model for solar energy applications: Geometric spectral albedo. *Applied Energy*, 255(August):113867.

A ALGORITHMS USED FOR MODELLING

In this appendix, the algorithms used in modelling PV potential in complex geometries will be provided. This section only provides basic information on the logic of the workflow, and a detailed explanation of these algorithms have been provided from chapters 3 to 5.

A.1 LOADING THE 3D OBJECT

In this section, the pre-processing stages on RHINO are also included apart from the logic used in MATLAB.

- Step 1: Obtain the 3D file from 3DWarehouse in .skp format
- Step 2: Load object into RHINO
- Step 3: Mesh un-meshed surfaces in the model
- Step 4: Quadrangulate the newly meshed surfaces/objects
- Step 5: Export into .obj file format
- Step 6: Load the .mtl file which has texture definitions
- Step 7: Parse data containing the phrase "newmtl" to read texture names
- Step 8: Parse data contained in the line with value "Kd", where this parameter contains texture information used in visual rendering
- Step 9: Load .obj file with geometric data
- Step 10: Parse data contained in lines with value "v", where v stands for vertices
- Step 11: Parse data contained in lines with value "f", where f stands for faces
- Step 12: Read unique data from the parsed variables and ignore repeated vertices/faces
- Step 13: Add indices to the loaded vertices and face definitions
- Step 14: Store variables

A.2 REFINING THE SURFACES

- Step 1: Determine non-planar surfaces present in the model
- Step 2: Triangulate non-planar surfaces using the Delaunay method and preserving the shape of non-planar polygon
- Step 3: Eliminate non-planar surfaces and replace them with generated planar polygons

- Step 4: Determine polygons which are of zero area by verifying for surfaces defined by collinear points
- Step 5: Remove the polygons defined with collinear points
- Step 6: Replace old geometric data with new data

A.3 PLACING PV PANELS

If modules are to be placed beside an object

- Step 1: Define units used to model Scene
- Step 2: Assign azimuth angle to the building
- Step 3: Define module's geometric parameters
- Step 4: Generate the panel and frame geometry
- Step 5: Place the modules beside the object at user-defined distances
- Step 6: Rotate the building by the defined azimuth angle
- Step 7: Define the ground for the scene

If modules are to be placed on the surface of an object

- Step 1: Define the units used to model Scene
- Step 2: Assign azimuth angle to the building
- Step 3: Select the points on the surface to put the panels on
- Step 4: Verify planarity of selected points, if non-planar then re-select the points
- Step 5: Define module's geometric parameters
- Step 6: Generate the panel geometry
- Step 7: Place the module on surface at user-defined distances from the vertex closest to origin on the selected plane
- Step 8: Rotate the building by the defined azimuth angle
- Step 9: Define the ground for the scene

The module's geometric parameters include data such as cell dimensions, spacing, module tilt, azimuth and thickness.

A.4 GENERATING GRID POINTS FOR RAY-TRACING

- Step 1: Find the plane of the solar cells, for both front and rear surfaces in case of bi-facial modules
- Step 2: Define number of grid points per cell
- Step 3: Generate grid points equidistant from the centre of the cell. A maximum of 4 points can be generated, each located at the midpoint between the centre of the cell and neighbouring edge.

- Step 4: Verify the normal of the grid points by comparing with the normal of the cell surface
- Step 5: Place the grid points above the cell, by translating the points along the defined surface normal
- Step 6: Index the points in order for averaging results of ray-tracing
- Step 7: Add data of the surface normals to the grid coordinates
- Step 8: Translate the points above the surface of the cells to find available irradiance

A.5 CONVERSION TO THE RADIANCE SYNTAX

To perform the RADIANCE backward ray-tracing simulations, the following files are required:

- The material descriptions
- The geometric data
- Grid points for backward ray-tracing
- Climate file for generating the sky

The climate file is an external file which can be obtained from weather softwares such as Metoenorm®. The grid points file which will be used has been described already in section A.4 of this appendix. In this section, the parsing of files back to an ASCII based text file with the extension .rad will be presented.

To get the data with the material descriptions:

- Step1: Obtain the material name from the variable stored in the base work space
- Step 2: Obtain the RADIANCE subclass of material definition. This can be plastic, metal, glass or mirror
- Step 3: Convert the data in the form *void* **subclass materialname** to add to the text file
- Step 4: Obtain the material parameters from the stored variable. This can be 3 or 5 values depending on the RADIANCE subclass
- Step 5: If the number of parameters is 3, write data into text file as: *3* **c1 c2 c3**.
- Step 6: If number of parameters is 5, write data into text file as: *5* **c1 c2 c3 c4 c5**

To convert geometric data into the RADIANCE Syntax:

- Step 1: Obtain the polygon number by referring to the index of the surface
- Step 2: With the material name procured earlier, add material and surface definition as: **materialname** *polygon* **polygon-index**
- Step 3: Obtain the vertex data defining the surface being parsed
- Step 4: If the number of vertices defining the surfaces is **n**, convert surface data into the RADIANCE syntax as follows: **3*n** $x_1 x_2 x_3 .... x_n y_n z_n$

To get more information on how the data is stored across the various files generated, refer Appendix C

A.6 THE RAY-TRACING PROCEDURE

- Step 1: Define the year for Simulation.
- Step 2: Choose Climate file. The usual source for climate data is Meteonorm and is stored in the .dat format
- Step 3: Specify the path in which RADIANCE files are stored during the ray-tracing.
- Step 4: Calculate the Sun's position for every time instant in the climate file (azimuth angle and altitude)
- Step 5: Generate the Sky data using the *gendaylit* command on RADIANCE, with data from the climate file as inputs.
- Step 6: Convert the opto-geometric input files (sky file, materials file and geometry file) into an octree file to reduce computational time with the help of RADIANCE command *oconv*
- Step 7: Ray-trace from the sensor positions defined in the scene (testpoints file) using RADIANCE command *rtrace*.
- Step 8: Read results file and average the results from each grid point belonging to a cell to obtain the plane of array irradiance of a cell at the given time instant in the climate file.

B SOFTWARE AND INSTALLATION

In this appendix, a detailed overview of the software requirements and the programming languages involved will be elucidated. The installation procedure for platform independent softwares will be discussed first. After this has been presented, the installation of softwares which change with OS platforms will be introduced in three sections: section B.1 discusses the installation procedure for MacOS users; section B.2 and B.3 will similarly highlight the installation procedures for Windows and Linux OS users respectively.

For all users interested in utilising the new capabilities of the PVMD toolbox , it is essential to have the following softwares:

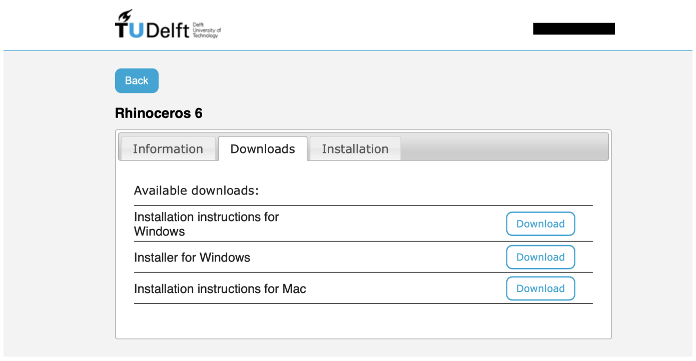
- Rhinoceros (RHINO) 6 or later, or any other 3D CAD software capable of exporting geometries in the **.obj** format. RHINO 6 is available for free on for TU Delft employees and students alike
- RADIANCE lighting engine
- Latest version of MATLAB. It is essential to install the Parallel Computing Toolbox to be able to use GenPRO before running module level simulations.

In the case of programming languages, the user is expected to be competent in shell scripting or command line programming for the installation of RADIANCE. The basic commands and steps to install RADIANCE via the command prompt will be explained in this appendix. However, if the user wishes to maximise the use of RADIANCE for other problems such as rendering, the user manuals available on the RADIANCE website are to be referred. It is highly recommended to follow the basic tutorials on RADIANCE to get used to command line scripting.

RHINO is available only for windows and MacOS. Linux users would have to install another software. The procedure to obtain RHINO for Windows/MacOS is explained below:

- Go to software.tudelft.nl
- Login with your NET-id
- Click on Rhinoceros-6 from the list on the screen
- Download the installation manual and the software from the downloads option on the screen, as shown in figure B.1.
- Follow the procedure in the installation manual to successfully install RHINO on the system.

For all other 3D software packages, refer to the respective websites for the installation procedures and file compatibilities.


[redacted]

Back

Rhinoceros 6

Information | Downloads | Installation

Available downloads:

Installation instructions for Windows	Download
Installer for Windows	Download
Installation instructions for Mac	Download

Figure B.1: Installation page for RHINO 6.

To install MATLAB, the following steps are to be carried out:

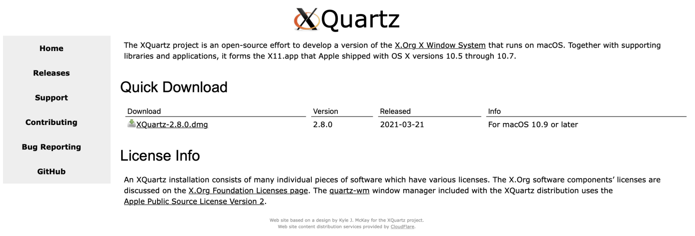
- Go to www.mathworks.com
- Login using your mathworks account. If a new account is to be created, follow the steps when prompted by verifying university for obtaining access
- Follow the installation procedure for the corresponding OS from the downloads page to complete MATLAB installation.

B.1 RADIANCE FOR MAC-OS USERS

The installation of RADIANCE for MacOS also requires the installation of the XQuartz (X11) software. It is a compiler similar to the command prompt application *Terminal*, but includes additional features specific to RADIANCE to maximise ease of use. It is preferred to use XQuartz instead of the actual command prompt to perform the commands of RADIANCE.

To install XQuartz, follow the steps mentioned below:

- Download the latest version of XQuartz from the website www.xquartz.org. Refer to figure B.2 for more information.
- Run the disk image as the administrator and follow the instructions in the prompt to install XQuartz on the Mac-OS system.



The XQuartz project is an open-source effort to develop a version of the X.Org X Window System that runs on macOS. Together with supporting libraries and applications, it forms the X11.app that Apple shipped with OS X versions 10.5 through 10.7.

Quick Download

Download	Version	Released	Info
XQuartz-2.8.0.dmg	2.8.0	2021-03-21	For macOS 10.9 or later

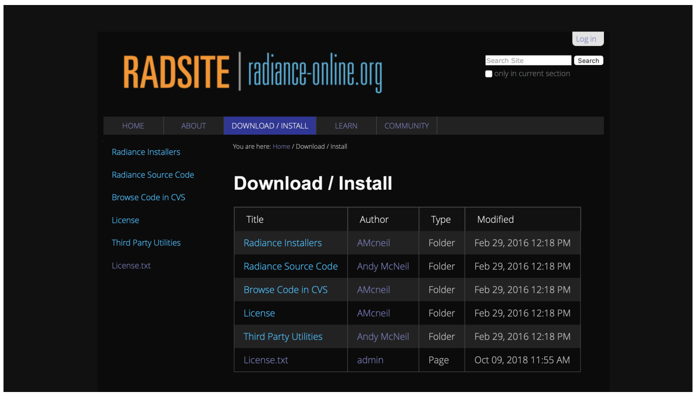
License Info

An XQuartz installation consists of many individual pieces of software which have various licenses. The X.Org software components' licenses are discussed on the X.Org Foundation Licenses page. The quartz-wm window manager included with the XQuartz distribution uses the Apple Public Source License Version 2.

Web site based on a design by Kyle J. McKay for the XQuartz project.
Web site content distribution services provided by CloudFlare.

Figure B.2: Installation page for XQuartz and supporting software.

To obtain the latest package of RADIANCE, go to the website www.radiance-online.org and go to the downloads tab to install the software.This will redirect the user to the **github** releases page, as shown in figure B.3. There are two files available, one is an installer with the extension **.pkg** and the other is a **.zip** file. Select the file with the **.pkg** extension, and follow the installation steps on prompt to complete installation.



You are here: Home / Download / Install

Download / Install

Title	Author	Type	Modified
Radiance Installers	AMcneil	Folder	Feb 29, 2016 12:18 PM
Radiance Source Code	Andy McNeil	Folder	Feb 29, 2016 12:18 PM
Browse Code in CVS	AMcneil	Folder	Feb 29, 2016 12:18 PM
License	AMcneil	Folder	Feb 29, 2016 12:18 PM
Third Party Utilities	Andy McNeil	Folder	Feb 29, 2016 12:18 PM
License.txt	admin	Page	Oct 09, 2018 11:55 AM

Figure B.3: The RADIANCE website.

After running the installer package from the downloads folder, the files for installation will be stored in the directory **/usr/local/radiance**. To access or view the extracted files, use the keyboard shortcut **shift+command+g** in finder.

Open the XQuartz application from finder, which opens the command prompt for Mac-OS. Here, it is important to pay attention to the shell which is used for command line scripting. This is because the scripting environment stores the executed code or variables and will need to be accessed in-case installation de-bugging is required.

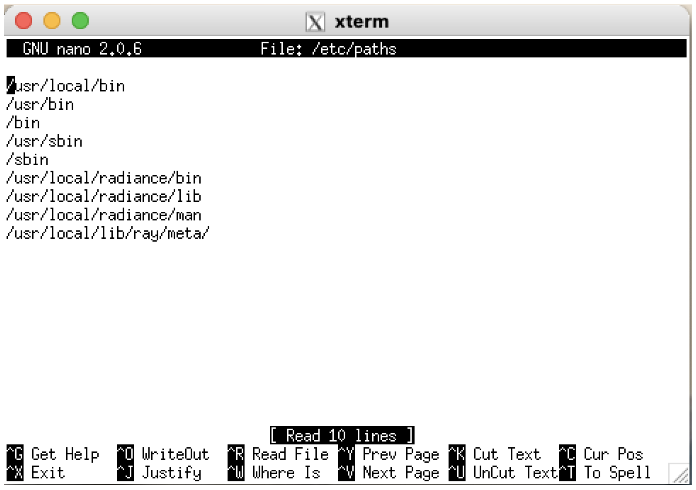
To verify the installation, type **sudo nano /etc/paths** on the command prompt. Type the unlock password on being requested by the system. Here, the following paths should be seen:

```
/usr/local/radiance/bin
/usr/local/radiance/lib
/usr/local/radiance/man
/usr/local/lib/ray/meta
```

The screen should look like figure B.4. If these paths are not visible, manually add them to the system path. Once entered, press **control+x** and type **y** on prompt. Press **return** to come back to home screen of the command prompt. This should include the libraries in the system.
To verify if the installation is successful, restart the command prompt and type **man *command***, where *command* is a RADIANCE command as mentioned in the RADIANCE reference manual. The **man** command is the abbreviation for man pages or manual entry, which is a document containing the syntax and description of each command which has been installed. If no entry shows up, it means RADIANCE has not been installed correctly.

In-order to add the environment variables to the shell, use the following command in the prompt: **nano .bash_profile**. In the pop-up window, type the following commands:

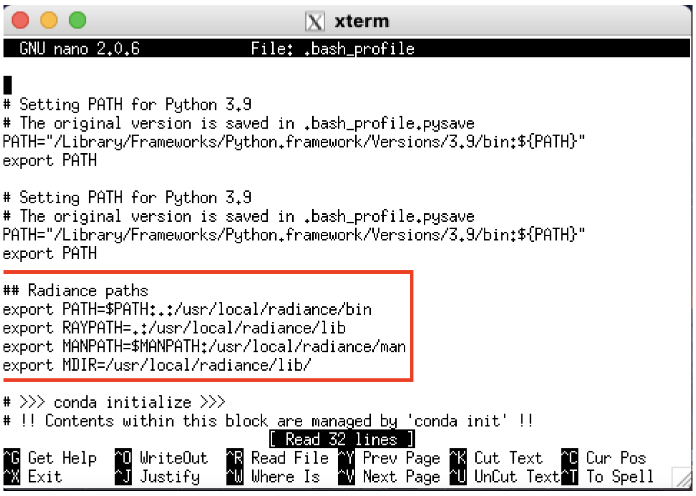
```
#exporting radiance files
export PATH=$PATH:.:/usr/local/radiance/bin
export RAYPATH=.:/usr/local/radiance/lib
export MANPATH=$MANPATH:/usr/local/radiance/man
export MDIR=/usr/local/lib/ray/meta/
```



```
xterm
  GNU nano 2.0.6              File: /etc/paths

/usr/local/bin
/usr/bin
/bin
/usr/sbin
/sbin
/usr/local/radiance/bin
/usr/local/radiance/lib
/usr/local/radiance/man
/usr/local/lib/ray/meta/

                                [ Read 10 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Figure B.4: Command Window for RADIANCE Paths.

The command window should look like figure B.5, and this figure represents the shell environment in which the variables are to be stored. It will represent the bash environment if that is the default shell:



```
xterm
  GNU nano 2.0.6              File: .bash_profile


# Setting PATH for Python 3.9
# The original version is saved in .bash_profile.pysave
PATH="/Library/Frameworks/Python.framework/Versions/3.9/bin:${PATH}"
export PATH

# Setting PATH for Python 3.9
# The original version is saved in .bash_profile.pysave
PATH="/Library/Frameworks/Python.framework/Versions/3.9/bin:${PATH}"
export PATH

## Radiance paths
export PATH=$PATH:.:/usr/local/radiance/bin
export RAYPATH=.:/usr/local/radiance/lib
export MANPATH=$MANPATH:/usr/local/radiance/man
export MDIR=/usr/local/radiance/lib/

# >>> conda initialize >>>
# !! Contents within this block are managed by 'conda init' !!
                                [ Read 32 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Figure B.5: Exporting paths for RADIANCE.

Finally, it is essential to install the **vchars.mta file** for using RADIANCE. If the installation steps have been performed correctly as previously described, this file would already be present in the directory: **/usr/local/lib/ray/meta**. If this file is not available, it cannot run all features available with RADIANCE. This file can be obtained in this URL: www.radiance-online.org/cgi-bin/viewcvs.cgi/ray/lib/meta/vchars.mta?hideattic=0&revision=1.2&view=markup.

Download this file and add it to the aforementioned directory to complete RADIANCE installation. RADIANCE should perfectly work after following these steps. Verify the installation by using the **man** command again in the command prompt. To verify if the environment variables have been correctly installed, use the command **echo $RAYPATH** and the directory in which radiance commands exist will be displayed. If it is not displayed, then the installation is incorrect.

B.2 RADIANCE FOR WINDOWS USERS

Windows users can directly install RADIANCE by downloading the .exe file from the downloads page of the RADIANCE website, as shown in figure B.3. In the installation page, download the file with the **.exe** extension and run the file. Follow the instructions on the prompt to complete installation.

Note: During the installation, it is important to change the installation location as blank spaces are not accepted in the installation path. Therefore, change the installation location of the RADIANCE path to C:\RADIANCE.

B.3 RADIANCE FOR LINUX USERS

A step-by step guide to install RADIANCE in Linux platforms has been mentioned in the webpage:

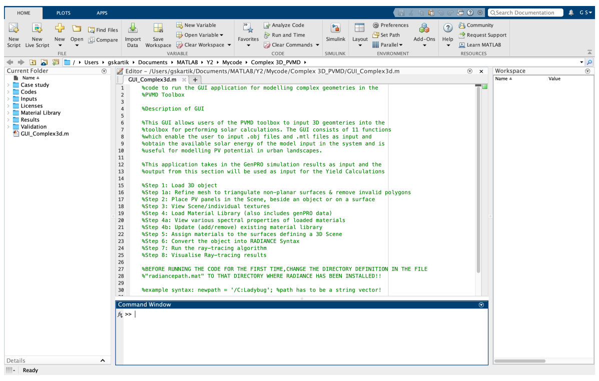
https://discourse.radiance-online.org/t/installing-radiance-in-ubuntu-7-10-step-by-step-guide/1284.

Refer this page and follow the procedure to successfully install RADIANCE.

C USER MANUAL FOR THE PVMD TOOLBOX

In this part of the appendices, a user manual for using the new and improved PVMD Toolbox is presented. As a reminder, it is expected that the user has already installed the requisite softwares and is aware of the programming language requirements to edit and make changes to the code for updating functionalities. These have been mentioned for various OS platforms in Appendix B. It is also to be noted that this appendix only deals with modelling complex 3D scenes in the toolbox and not the entire toolbox.

Once MATLAB has been opened and the folder concerned with the PVMD Toolbox has been opened, the command window looks like figure C.1.



```
%code to run the GUI application for modelling complex geometries in the
%PVMD Toolbox

%Description of GUI

%This GUI allows users of the PVMD toolbox to input 3D geomteries into the
%toolbox for performing solar calculations. The GUI consists of 11 functions
%which enable the user to input .obj files and .mtl files as input and
%obtain the available solar energy of the model input in the system and is
%useful for modelling PV potential in urban landscapes.

%This application takes in the GenPRO simulation results as input and the
%output from this section will be used as input for the Yield Calculations

%Step 1: Load 3D object
%Step 1a: Refine mesh to triangulate non-planar surfaces & remove invalid polygons
%Step 2: Place PV panels in the Scene, beside an object or on a surface
%Step 3: View Scene/individual textures
%Step 4: Load Material Library (also includes genPRO data)
%Step 4a: View various spectral properties of loaded materials
%Step 4b: Update (add/remove) existing material library
%Step 5: Assign materials to the surfaces defining a 3D Scene
%Step 6: Convert the object into RADIANCE Syntax
%Step 7: Run the ray-tracing algorithm
%Step 8: Visualise Ray-tracing results

%BEFORE RUNNING THE CODE FOR THE FIRST TIME,CHANGE THE DIRECTORY DEFINITION IN THE FILE
%"radiancepath.mat" TO THAT DIRECTORY WHERE RADIANCE HAS BEEN INSTALLED!!

%example syntax: newpath = '/C:Ladybug'; %path has to be a string vector!
```

```
>>
```

Figure C.1: GUI code for modelling 3D geometries.

Here, the following sub-folders are of most importance:

- 3D Files: Contains the .obj and .mtl file of the scene to be imported into the toolbox.
- Codes: Has all the codes used for modelling. These are not to be edited unless unavoidable
- Material Library: Contains material library information from the NASA ECOSTRESS Library and recently loaded .mtl file
- Results: Contains the results of the simulation.

To obtain the 3D files in the .obj format, the user has to run RHINO or any 3D Software in order to export the file. The file also needs to be pre-processed before export to refine the model, which has been discussed extensively in chapter 3 of this thesis report. This can be achieved as follows:

- Log-in to Google SketchUP and 3dWarehouse for obtaining the 3D Object to be loaded into the Toolbox. Care is to be taken while downloading the file. For users of Rhino 6, only SketchUP 2019 files/lower versions can be edited. For users of Rhino 7 or other 3D CAD softwares, keep in mind the compatibility of the file exported from 3dWarehouse.

- Open the 3D object on Rhino
- On Rhino, use the **Mesh** command to create meshes of objects which were un-meshed while exporting the model.
- After completing the mesh, use the command **QuadrangulateMesh**, to convert all new meshes into quad elements, or elements with 4 vertices defining a surface.
- Export the object into the .obj format and save it in the 3D Files folder, using the **Export** command on Mac-OS or the **Save-As** option on Windows systems.

Once this has been completed, open MATLAB, change the directory into the PVMD Toolbox folder and run the code as shown in figure C.1. Running the code opens the user-interface to run the sub-codes with the following functionalities as shown in figure C.2:

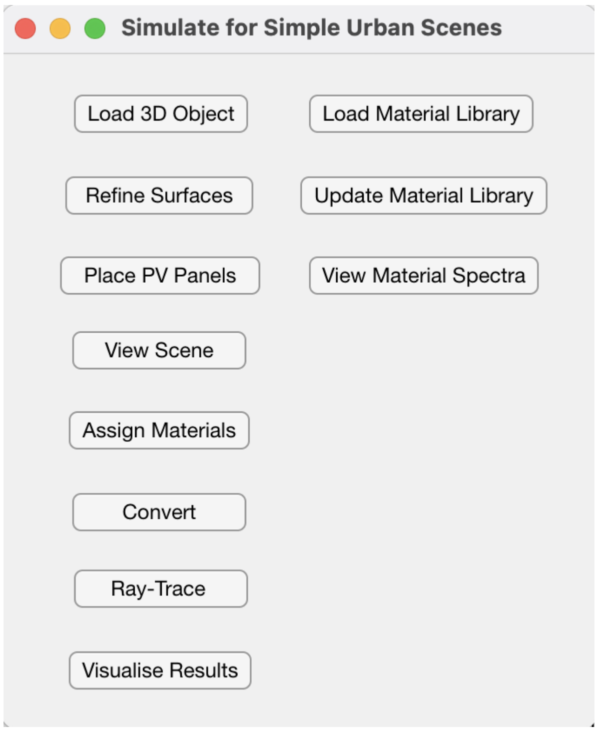


Figure C.2: GUI and its Sub-functions.

The user can perform the following operations:

- Load 3D Object: Loads the geometry and texture files (.obj & .mtl files)
- Refine Surfaces: Refines model to improve surface definitions
- Place PV Panels: Add the panels to the scene
- View Scene: View the loaded 3D Scene
- Load Material Library
- Update Material Library: Users can add, remove material files or update RADIANCE integration limits for spectral reflectance
- View Material Spectra: View and compare the spectral reflectance of various materials from NASA's ECOSTRESS library
- Assign Materials: Assign the materials to the surfaces

- Convert: Generate the files to be used for ray-tracing by converting files into RADIANCE syntax
- Ray-Trace: Run the backward ray-tracing simulation on RADIANCE to obtain the irradiance values
- Visualise Results: View the results from the Ray-tracing Simulation

Follow the steps in the order prescribed in the list above for modelling PV potential in complex scenes. In step 1, the user is prompted two times to select files from the 3D Files folder: once to load the texture data and then load the geometry. It is important to remember to select the correct files; there will be errors if otherwise. This step has been visually represented from figures C.3 to C.5 . As a reminder, if changes were made to the .obj or .mtl file manually after exporting, the code might throw errors while loading the 3d object into the toolbox. If such changes were made, the code needs to be edited to accommodate the changes.

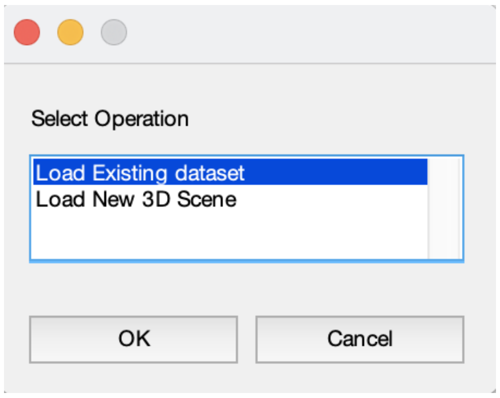


Figure C.3: Step 1 in Modelling Complex Geometries.

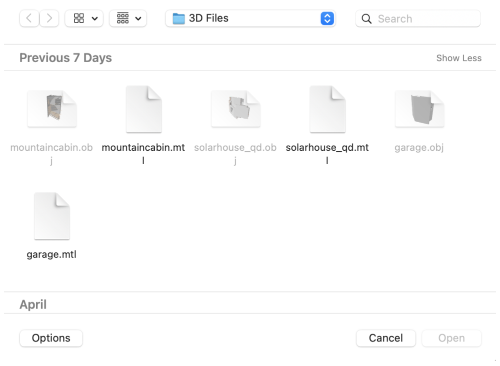


Figure C.4: Selecting Texture File.

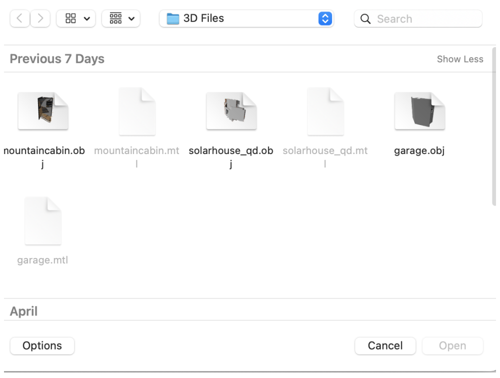
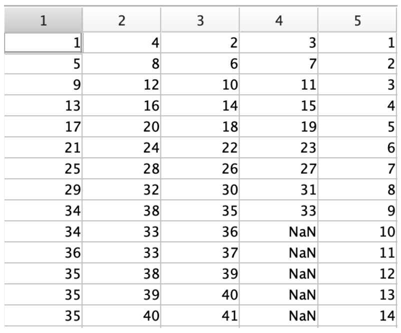


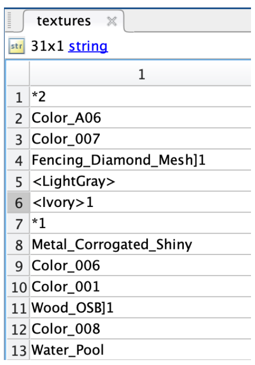
Figure C.5: Prompt to Input Geometry File.

Once the objects have been loaded, it will be stored in the data-set "**3dobject.mat**", which can be accessed thorough the results folder. Once the object has been loaded, the second step is to refine the surfaces. In-depth information on how the code works, its purpose and outcomes is available on chapter 4 of this thesis report. This step is crucial to ensure that surfaces are defined properly and model geometry is preserved to obtain correct ray-tracing results. The outcome of this step is a refined model which replaces the data stored in the data-set "**3dobject.mat**", and the contents of the data-set are shown from figures C.6 to C.8 . Figure C.6 shows the surfaces/face definition, with each row defining the vertex number defining the face, while figure C.8 shows the vertex coordinates in the first 3 columns. The last column of both these matrices contain the surface and vertex index, respectively.



1	2	3	4	5
1	4	2	3	1
5	8	6	7	2
9	12	10	11	3
13	16	14	15	4
17	20	18	19	5
21	24	22	23	6
25	28	26	27	7
29	32	30	31	8
34	38	35	33	9
34	33	36	NaN	10
36	33	37	NaN	11
35	38	39	NaN	12
35	39	40	NaN	13
35	40	41	NaN	14

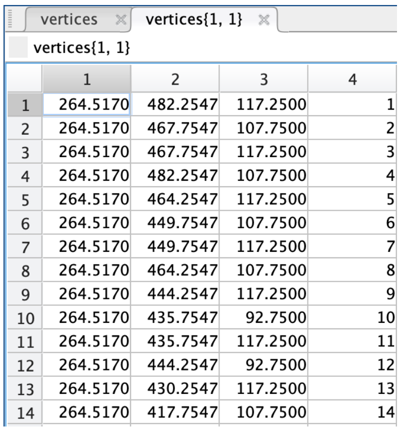
Figure C.6: Contents of Face Definitions.



textures
31x1 string

	1
1	*2
2	Color_A06
3	Color_007
4	Fencing_Diamond_Mesh]1
5	<LightGray>
6	<Ivory>1
7	*1
8	Metal_Corrogated_Shiny
9	Color_006
10	Color_001
11	Wood_OSB]1
12	Color_008
13	Water_Pool

Figure C.7: Texture Definitions in file.



	1	2	3	4
1	264.5170	482.2547	117.2500	1
2	264.5170	467.7547	107.7500	2
3	264.5170	467.7547	117.2500	3
4	264.5170	482.2547	107.7500	4
5	264.5170	464.2547	117.2500	5
6	264.5170	449.7547	107.7500	6
7	264.5170	449.7547	117.2500	7
8	264.5170	464.2547	107.7500	8
9	264.5170	444.2547	117.2500	9
10	264.5170	435.7547	92.7500	10
11	264.5170	435.7547	117.2500	11
12	264.5170	444.2547	92.7500	12
13	264.5170	430.2547	117.2500	13
14	264.5170	417.7547	107.7500	14

Figure C.8: Vertex Definitions in File.

With the file loaded into MATLAB without errors, the next step is to place the PV Panels in the scene. There are two possibilities here: Place the PV panels on a surface of the building, or place it next to the object which has been loaded. If the user decides to place the panels on a surface, it highly recommended to select higher surfaces (such as roofs) as the code is in its primitive stage. Moreover, it is also not recommended to model PV panels placed in surfaces surrounded by multiple objects, as selection of required face becomes difficult. The new file with the panel and frame geometry data will be saved as **"Scene.mat"** in the results folder of the toolbox.

Initially, the user has to input the model units, which is to be noted down while pre-processing the model on RHINO or 3D CAD software. This is shown in figure C.9 . The user has 4 options only: **meters**, **inches**, **millimeters**, **centimeters**, Other units would throw errors. These units will also be stored as a .mat file in the "Inputs" folder.

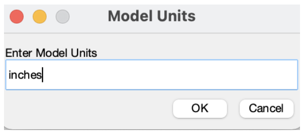


Figure C.9: Define the units of the model.

Once this has been input, the user is requested to input the azimuth of the building. The azimuth angle follows the following convention: South is 0, West is 90, North is 180 and East is 270 degrees. After defining the azimuth of the object, the user is presented with the two choices as described earlier, which is shown in figure C.10.

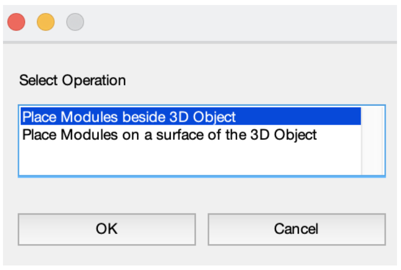
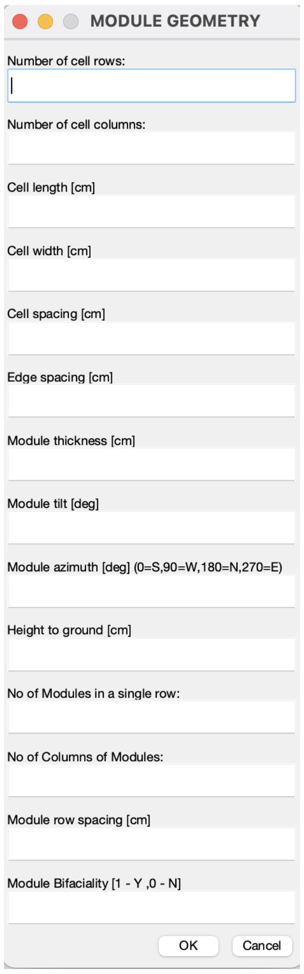

Select Operation
Place Modules beside 3D Object
Place Modules on a surface of the 3D Object
OK
Cancel

Figure C.10: User Operation to place PV panels.

If option one is selected from figure C.10, the user is presented with a dialog box, in which the mechanical parameters of the module are to be input. These parameters include cell dimensions, module tilt, bifaciality, etc. and is illustrated in figure C.11.


MODULE GEOMETRY

Number of cell rows:

Number of cell columns:

Cell length [cm]

Cell width [cm]

Cell spacing [cm]

Edge spacing [cm]

Module thickness [cm]

Module tilt [deg]

Module azimuth [deg] (0=S,90=W,180=N,270=E)

Height to ground [cm]

No of Modules in a single row:

No of Columns of Modules:

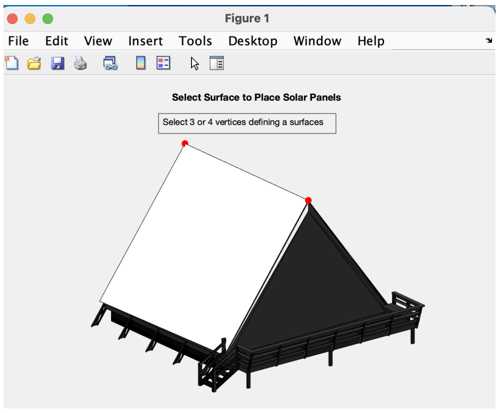
Module row spacing [cm]

Module Bifaciality [1 - Y ,0 - N]

Figure C.11: Geometric Parameters of the modules.

If option 2 is chosen, then a GUI is presented where the user can interactively select the surfaces on the object to place the panel. This is highlighted in figure C.12 , where the red dots indicate the vertex selected by the user. A minimum of 3 vertices is to be selected, with 4 recommended for better accuracy.

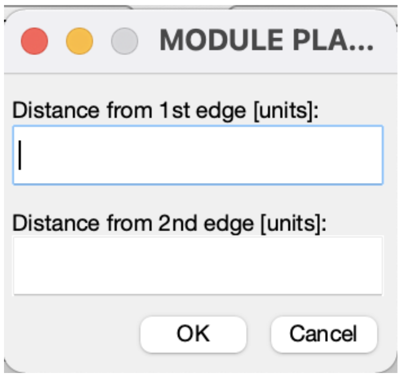
A pop-up window similar to figure C.11 is presented after the surfaces are selected, and once these parameters are defined, the final step is to define the location of the modules. In option 1, the panels will be moved from the boundaries of the object as defined by the user, while in option 2 the panels will be placed away from the edges of the surfaces by the distance defined. This distance is to be input when the application prompts the user, which is as shown in figure C.13. The user can also input negative values at the prompted location. A negative location would mean that the panels are translated along the -ve axes.
The modules are now placed in the scene, and can be viewed using the 4th step of the entire process. The user can decide whether to view a single material, the entire 3D scene with the panels or just the 3D object, by selecting from the pop-up window. This command displays the object/selected surface, the number of elements defining the object/surface and the surface area of the object/surface. Two complete scenes, one with panels on a surface and the other with panels beside an



Select Surface to Place Solar Panels

Select 3 or 4 vertices defining a surfaces

Figure C.12: Vertices of Plane on which panels will be placed.

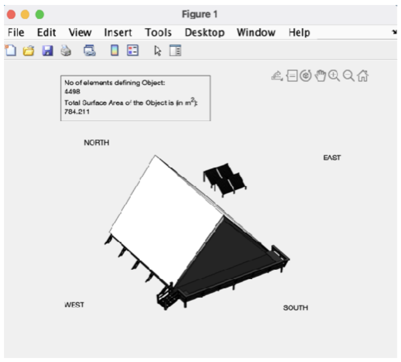

MODULE PLA...

Distance from 1st edge [units]:

Distance from 2nd edge [units]:

Figure C.13: Define Module Placing.

object are shown in figures C.14 and C.15 respectively. The directions in which the panels and the building are pointing can also be seen in the two figures.



No of elements defining Object:
4498
Total Surface Area of the Object is (in m^2):
784.211

Figure C.14: A scene with modules beside the 3D object.

The next step is to load the material library which contains data from the NASA ECOSTRESS Library, and this data is stored as seen in figure C.16. The three values stored (by default) in the parameters R,G and B are the spectrally integrated albedos in the UV, visible and IR range respectively, with the data ranging between wavelength limits 250-1500 nm. It is also possible to remove materials from the library, add extra materials to the existing folder or update the integration limits of the R G and B parameters by using the Update Material Library option, which throws a pop-up as shown in figure C.17 . The user can also view and compare different materials and their spectral properties by using the View Material Spectra Option.

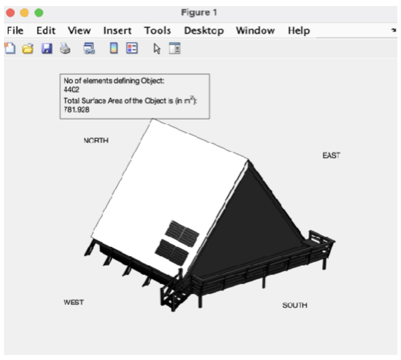
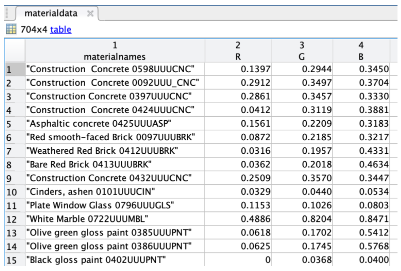


Figure 1
File Edit View Insert Tools Desktop Window Help
No of elements defining Object:
4402
Total Surface Area of the Object is (in m^2):
781.928
NORTH
EAST
WEST
SOUTH

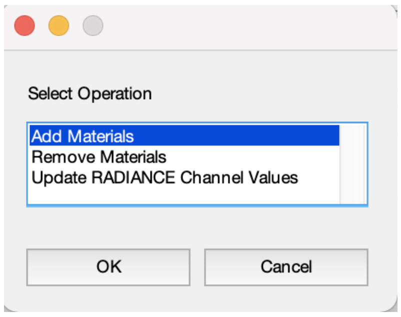
Figure C.15: A Scene with Modules on a surface.


materialdata

704x4 table

	1 materialnames	2 R	3 G	4 B
1	"Construction Concrete 0598UUUCNC"	0.1397	0.2944	0.3450
2	"Construction Concrete 0092UUU_CNC"	0.2912	0.3497	0.3704
3	"Construction Concrete 0397UUUCNC"	0.2861	0.3457	0.3330
4	"Construction Concrete 0424UUUCNC"	0.0412	0.3119	0.3881
5	"Asphaltic concrete 0425UUUASP"	0.1561	0.2209	0.3183
6	"Red smooth-faced Brick 0097UUUBRK"	0.0872	0.2185	0.3217
7	"Weathered Red Brick 0412UUUBRK"	0.0316	0.1957	0.4331
8	"Bare Red Brick 0413UUUBRK"	0.0362	0.2018	0.4634
9	"Construction Concrete 0432UUUCNC"	0.2509	0.3570	0.3447
10	"Cinders, ashen 0101UUUCIN"	0.0329	0.0440	0.0534
11	"Plate Window Glass 0796UUUGLS"	0.1153	0.1026	0.0803
12	"White Marble 0722UUUMBL"	0.4886	0.8204	0.8471
13	"Olive green gloss paint 0385UUUPNT"	0.0618	0.1702	0.5412
14	"Olive green gloss paint 0386UUUPNT"	0.0625	0.1745	0.5768
15	"Black gloss paint 0402UUUPNT"	0	0.0368	0.0400

Figure C.16: Material Spectral Reflectances.


Select Operation

Add Materials
Remove Materials
Update RADIANCE Channel Values

Figure C.17: Operations to Update Material Library.

The following step is to assign the materials to the scene. This presents the user with three options: To assign materials directly from the .mtl file, add data from the NASA spectral Library or edit an existing scene (figure C.18) . The first option is highly not recommended, as the values present in the .mtl file only contain reflectances to visually render the scene and not values corresponding to the solar spectrum. The material properties to be assigned to the PV cells will be directly taken from the cell simulation data generated by the GenPRO simulation from the toolbox.
The materials can be assigned with the subclass required to define the material in RADIANCE, with all these options presented in the form of a dialog box. Based on the selected subclass, surface specularity and roughness values will be assigned to the surfaces. The data-set containing the opto-geometric data will be stored in the file "**ObjectforRadiance.mat**" in the Results folder, and the data in this file is stored as shown figure C.19. The file has information on the surfaces, its material, RADIANCE subclass and optical properties.
The final step before running the RADIANCE ray-tracing simulation is to convert the data into the RADIANCE syntax, it this step generates three files: **materials.rad**, the file containing optical data; **scene.rad**, the file containing geometric data and

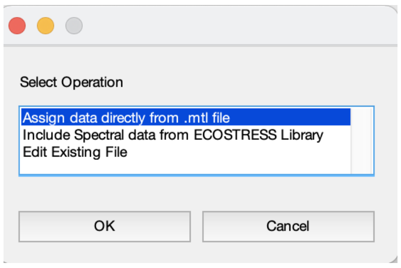
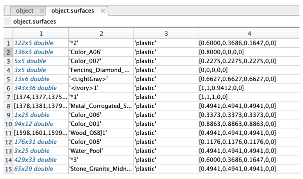

Select Operation
Assign data directly from .mtl file
Include Spectral data from ECOSTRESS Library
Edit Existing File
OK
Cancel

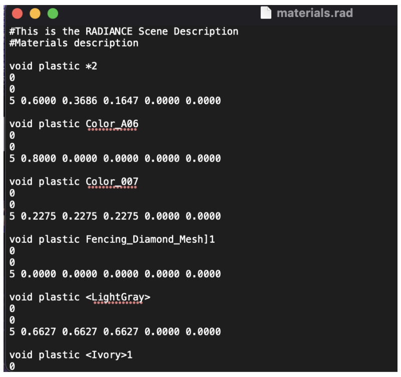
Figure C.18: Assigning Materials to Surfaces.


object
object.surfaces
object.surfaces

	1	2	3	4
1	122x5 double	'*2'	'plastic'	[0.6000,0.3686,0.1647,0,0]
2	136x5 double	'Color_A06'	'plastic'	[0.8000,0,0,0,0]
3	5x5 double	'Color_007'	'plastic'	[0.2275,0.2275,0.2275,0,0]
4	3x5 double	'Fencing_Diamond_...	'plastic'	[0,0,0,0,0]
5	13x6 double	'<LightGray>'	'plastic'	[0.6627,0.6627,0.6627,0,0]
6	343x36 double	'<Ivory>1'	'plastic'	[1,1,0.9412,0,0]
7	[1374,1377,1375...	'*1'	'plastic'	[1,1,1,0,0]
8	[1378,1381,1379...	'Metal_Corrogated_S...	'plastic'	[0.4941,0.4941,0.4941,0,0]
9	1x25 double	'Color_006'	'plastic'	[0.3373,0.3373,0.3373,0,0]
10	94x12 double	'Color_001'	'plastic'	[0.8863,0.8863,0.8863,0,0]
11	[1598,1601,1599...	'Wood_OSB]1'	'plastic'	[0.4941,0.4941,0.4941,0,0]
12	176x31 double	'Color_008'	'plastic'	[0.1176,0.1176,0.1176,0,0]
13	3x25 double	'Water_Pool'	'plastic'	[0.4941,0.4941,0.4941,0,0]
14	429x33 double	'*3'	'plastic'	[0.6000,0.3686,0.1647,0,0]
15	65x29 double	'Stone_Granite_Midn...	'plastic'	[0.4941,0.4941,0.4941,0,0]

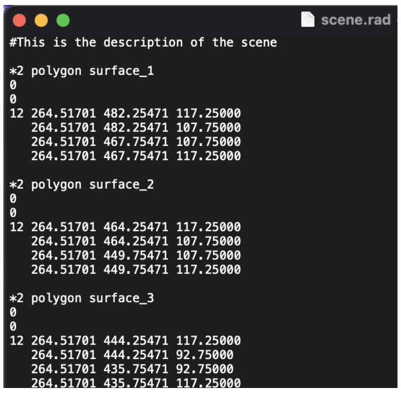
Figure C.19: Opto-Geometric Data of the Scene.

testpoints.pts, the file containing the grid points which will be used for the ray-tracing simulation. The files will be stored in the **"Radiance Files"** subfolder within the **Results** folder. The data stored in these files is illustrated from figures C.20 to C.22 .



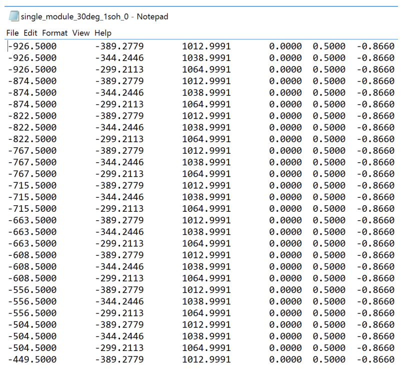
```
#This is the RADIANCE Scene Description
#Materials description

void plastic *2
0
0
5 0.6000 0.3686 0.1647 0.0000 0.0000

void plastic Color_A06
0
0
5 0.8000 0.0000 0.0000 0.0000 0.0000

void plastic Color_007
0
0
5 0.2275 0.2275 0.2275 0.0000 0.0000

void plastic Fencing_Diamond_Mesh]1
0
0
5 0.0000 0.0000 0.0000 0.0000 0.0000

void plastic <LightGray>
0
0
5 0.6627 0.6627 0.6627 0.0000 0.0000

void plastic <Ivory>1
0
```

Figure C.20: Material Definitions on RADIANCE.



```
#This is the description of the scene

*2 polygon surface_1
0
0
12 264.51701 482.25471 117.25000
   264.51701 482.25471 107.75000
   264.51701 467.75471 107.75000
   264.51701 467.75471 117.25000

*2 polygon surface_2
0
0
12 264.51701 464.25471 117.25000
   264.51701 464.25471 107.75000
   264.51701 449.75471 107.75000
   264.51701 449.75471 117.25000

*2 polygon surface_3
0
0
12 264.51701 444.25471 117.25000
   264.51701 444.25471 92.75000
   264.51701 435.75471 92.75000
   264.51701 435.75471 117.25000
```

Figure C.21: Scene Description on RADIANCE.


single_module_30deg_1soh_0 - Notepad
File Edit Format View Help

```
-926.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-926.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-926.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-874.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-874.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-874.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-822.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-822.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-822.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-767.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-767.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-767.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-715.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-715.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-715.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-663.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-663.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-663.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-608.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-608.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-608.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-556.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-556.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-556.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-504.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
-504.5000       -344.2446       1038.9991       0.0000  0.5000  -0.8660
-504.5000       -299.2113       1064.9991       0.0000  0.5000  -0.8660
-449.5000       -389.2779       1012.9991       0.0000  0.5000  -0.8660
```

Figure C.22: Grid Points used for Simulation.

A single row of the test-points file contains the points on the grid above the PV cells which will be used for the ray-tracing simulations, and the surface normal adhering to that cell. 5 parameters are used to define the material optical properties: 3 from the R,G and B values; the specularity and surface roughness form the other 2 values. The geometric data is defined by converting each surface into a polygon defined by 3 or more vertices, along with their coordinates.

Once the files for the ray-tracing are generated, perform the ray-tracing simulation by clicking on the button "Ray-trace" in the GUI. Before running the simulation, the directory containing has to be specified in a .mat file as a character vector. This .mat file is called **"radiancepath.mat"** and has to be edited before running the first simulation. On clicking the button, the user is prompted to enter the year of simulations and then requested to select the climate file. The climate file is taken from Meteonorm and must be stored in the **".dat"** format, in the "Inputs" folder. Once the file has been selected, RADIANCE will perform the ray-tracing. The results of the simulation will be stored in the main directory of RADIANCE, which will then be transferred to the MATLAB directory for computing the irradiance values of a cell at any given time instant.

The files will be stored in the file "results.mat", in the "Results" folder and can be visualised by clicking on the "Visualise results" button in the GUI. This provides 3 options for the user: 1) view the irradiance plot across the different modules at a given instant in time; 2) view the monthly irradiance plot and 3) view the irradiance plot for the year. While viewing the results at a single instant, the date and time have to be entered in the following format: **"DD-Mon hh:00"**. This format will also be shown as an example at the prompt. The results for the first option have been shown in Figure C.23. If the user selects option 2, the results will be as depicted in Figure C.24 and if the user selects option 3, the results will be presented as in Figure C.25 .

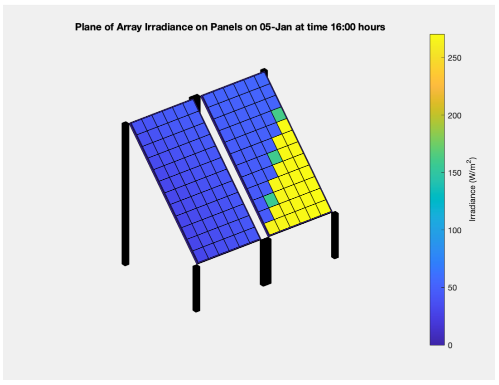


Figure C.23: Plane of Array Irradiance viewed for a single instant in time.

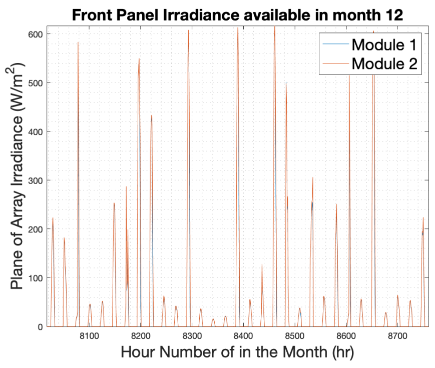


Figure C.24: Plane of Array Irradiance in the Month of December, 2019.

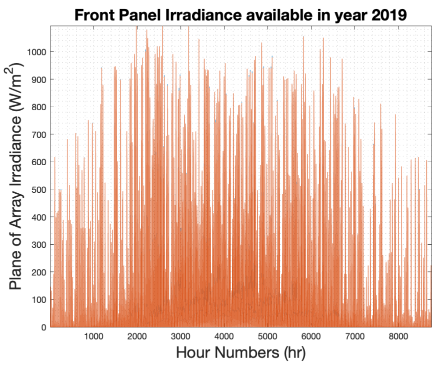


Figure C.25: Plane of Array Irradiance plotted for the whole year.

The plots presented are for a mono-facial module. If a bifacial module is used two graphs will be produced, with each indicating front and rear irradiance respectively. Thus, the toolbox can be used to model the results for modules installed in urban environments.

COLOPHON

This document was typeset using LaTeX. The document layout was generated using the arsclassica package by Lorenzo Pantieri, which is an adaption of the original classicthesis package from André Miede.