Bifacial_Radiance Training
Part 1

Silvana Ayala Peláez and Chris Deline

NREL Webinar
October 17, 2019
Housekeeping

• This webinar is being recorded.
  – A recording will be made available on the “NREL Learning” YouTube channel and the GitHub & readthedocs page a few weeks after the webinar.

• Audio issues?
  – Try switching from computer to phone audio
  – If problems persist, contact GoToWebinar technical support at (877) 582-7011.

• Have a question?
  – Please submit it using the ”Questions” panel in the GoToWebinar interface at any time.
  – We will answer as many questions as possible at several point during the webinar.
Open Source toolkit for working with RADIANCE for the ray-trace modeling of Bifacial Photovoltaics

Available on: https://github.com/NREL/bifacial_radiance
Documentation: https://bifacial-radiance.readthedocs.io
Installation: https://youtu.be/4A9GocfHKyM
V. 0.3.3.1 (10/16/19)
Radiance is a ray-trace software – a suite of tools for performing lighting simulation. Developed in 1985, by Greg Ward at Lawrence Berkeley National Laboratory. Underlying simulation engine for many packages. 
Homepage: [https://floyd.lbl.gov/radiance/HOME.html](https://floyd.lbl.gov/radiance/HOME.html)

We use **backward ray-trace** to evaluate the irradiance (W/m²) at the modules. Reduces complexity and run-time.
Bifacial Challenge: accurate modelling of rear irradiance

- Albedo
- Tilt
- Row-to-row spacing
- Clearance

**Irradiance**
- Location
- Weather
- Sky Diffuse

**Others:**
- Spacing between cells
- #rows, #panels
- Mounting Structure
- Other scene elements

Image courtesy of Opsun trackers, via Francois Gilles-Gagnon
View Factor Model for Rear Irradiance

Simple
- basic geometry
- inexpensive

Fast
- computationally inexpensive

Common
- Behind SAM, Pvsyst, bifacialvf and others

Simple
- basic geometry
Raytrace benefits:
• Any size array
• Sample any module
• Evaluate edge effects
• Complicated geometries
  • Modules
  • Racking
  • Obstructions
• Evaluate shading
• Evaluate electrical mismatch
• Open source
• Dedicated visual interface
• Validated

Cons:
• Complexity
• Run-times

← Visual interface
← Training
← HPC integration
← Simplified models
Comparison of models and validation with test-bed data

Single-axis tracking model

2017

Albedo assessment and database at Duramat

Framework to calculate bifacial gain with field data

Validation with 2 100kW fields

2018

Standards for bifacial rating

Assess system performance impact from rear irradiance shading and electrical mismatch

Capacity testing

2019

2100kW fields


*Submitted as extended journal to PinPV

Complete list of publications: https://github.com/NREL/bifacial_radiance/wiki
Bifacial_radiance
Training Part 1

1. Why/how we raytrace
2. Geometry
3. Skies
4. Github
5. Demo 1: Tutorial 1 in Jupyter Journal
6. Other Tutorials
7. Demo 2: GUI

20 minutes
10 minutes
10 minutes
Why | How we raytrace?
Components

Sky
Source Properties

“Scene”
Modules
Racks
Obstructions etc..

Materials
Properties
What does bifacial_radiance do?
bifacial_radiance is a python wrapper for calling and using Radiance, with specific functions to generate geometry (text files) related to bifacial pv systems.
How a full example might look like

```python
In [ ]:
from bifacial_radiance import *
demo = RadianceObj()
demo.setGround(0.52)
epwfile = demo.getEPW(37.5, -77.6)
metdata = demo.readWeatherFile(epwfile)
demo.gendaylight(metdata, 4020)
demo.makeModule("My_panel", x=1, y=2)
sceneDict={'tilt':30, 'pitch':3, 'clearance_height':0.5, 'azimuth':180, 'nMods':10, 'nRows':4}
scene = demo.makeScene("My_panel", sceneDict)
octfile=demo.makeOct() AnalysisObj(octfile, demo.name)
frontscan, backscan = analysis.moduleAnalysis(scene)
analysis.analysis(octfile, demo.name, frontscan, backscan)
```
1. Make Radiance Object
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky

3. Make Module
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object
2. Make Sky
3. Make Module
4. Make Scene
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky

3. Make Module

4. Make Scene

5. Make Oct
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object
2. Make Sky
3. Make Module
4. Make Scene
5. Make Oct
6. Analysis Obj
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky

3. Make Module

4. Make Scene

5. Make Oct

6. Analysis Obj

7. Analysis
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky

3. Make Module

![Image of Radiance file with commands]

```plaintext
! genbox black PVmodule 1.996 0.991 0.02 | xform -t -0.998 0 0 -a 2 -t 0 0.991 0
! genbox Metal_Grey hextubala 1.996 0.075 0.129903810568 | xform -t -0.998 0.9535 -0.304903810568
! genbox Metal_Grey hextube1b 1.996 0.075 0.129903810568 | xform -t -0.998 -0.0375 -0.0649519052838 -rx 60 -t 0 0.9
! genbox Metal_Grey hextube1c 1.996 0.075 0.129903810568 | xform -t -0.998 -0.0375 -0.0649519052838 -rx -60 -t 0 0
```
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky

3. Make Module

4. Make Scene
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object
2. Make Sky
3. Make Module
4. Make Scene
5. Make Oct
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object
2. Make Sky
3. Make Module
4. Make Scene
5. Make Oct
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky

3. Make Module

4. Make Scene

5. Make Oct

6. Analysis Obj

7. Analysis
MAIN STEPS / INSTRUCTIONS

1. Make Radiance Object

2. Make Sky
   cmd gencumsky

3. Make Module

4. Make Scene

5. Make Oct
   cmd oconv

6. Analysis Obj

7. Analysis
   cmd rtrace
Geometry
makeModule: “module unit” (module, cells, torque tube, etc) to replicate
Assign $x$ and $y$ depending on which value should go along the axis / slope.

For example:
‘PORTRAIT’
$y = 2, x = 1$  
LANDSCAPE
$y = 1, x = 2$
numpanels=2

numpanels=3

Collector Width (CW)
Torque Tube

All based on “diameter”

Modifiable material

Metal_Grey (GREY NOT GRAY) → 44% reflectivity.

Black → Absorptive.
X-gap, Y-gap, Z-gap
Cell Level Module
Axis of rotation Torquetube

- FALSE
- TRUE

Offset from axis of rotation = zgap + diam/2
**makeScene**: replicates whatever was defined as the module unit to make the Scene
MAPPING EXAMPLE:

Azimuth = 180
Tilt = 0
nMod = 5
nRows = 3
2 UP with torquetube,
Xgap and ygap exaggerated too
Sky
“Gendaylit”
Hourly distribution
Perez Model
Must know: sun position, DNI, DHI

Cumulative sky

“Gencumsky”

Bins sky into 145 patches of same angular extent

Irradiances for each patch can be cumulative for any period of time.

→ Increase in speed

Robinson, Stone “Irradiation modelling made simple: the cumulative sky approach” 2004
Gencumulative Sky

- Simulate Hourly: ~4380 simulations
- Simulate Daily: ~365 simulations
- Simulate Monthly: ~12 simulations
- Simulate Yearly: ~1 simulations
Cummulative Sky by Tracker Angle

Cumulative Skies

- Hourly: ~4380 simulations
- Accumulate Daily: ~365 simulations
- Accumulate Monthly: ~12 simulations
- Accumulate Yearly: ~1 simulations

Tracking
Accumulate by tracker angle
-45 to 45: ~19 simulations
Analysis Objects
MAPPING EXAMPLE:

Azimuth = 180
Tilt = 0
nMods = 5
nRows = 3
2 UP with torquetube,
Xgap and ygap exaggerated too

modWanted = 4
rowWanted = 3
Sensors are spaced equidistant to each other No ‘safety distance’ from edges.

The detectors measure irradiance at the first surface they ‘see’.

Sensors are placed close to the surfaces to avoid hitting torque tubes.

Complex geometries might require “Cleaning Up” the results
Azimuth

0
90
210
**Tracking**: same azimuth_axis, tilts are negative for the afternoon. Sensor locations fall on the same place in the module!
<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Why/how we raytrace</td>
</tr>
<tr>
<td>2</td>
<td>Geometry</td>
</tr>
<tr>
<td>3</td>
<td>Skies</td>
</tr>
<tr>
<td>4</td>
<td>Github</td>
</tr>
<tr>
<td>5</td>
<td>Demo 1: Tutorial 1 in Jupyter Journal</td>
</tr>
<tr>
<td>6</td>
<td>Other Tutorials</td>
</tr>
<tr>
<td>7</td>
<td>Demo 2: GUI</td>
</tr>
</tbody>
</table>
Github
Tutorials
Jupyter Notebooks
Website to store and manage code, track and control changes and new versions.

https://github.com/NREL/bifacial_radiance/

Open-source web application that allows you to create and share documents that contain live code, equations, visualizations and narrative text.

Example starting-guide: https://realpython.com/jupyter-notebook-introduction/
## Journals

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Introductory Example - Fixed Tilt simple setup.ipynb</td>
</tr>
<tr>
<td>2</td>
<td>Introductory Example - Single Axis Tracking with cumulative Sky.ipynb</td>
</tr>
<tr>
<td>3</td>
<td>Medium Level Example - Single Axis Tracking - hourly.ipynb</td>
</tr>
<tr>
<td>4</td>
<td>Medium Level Example - Debugging your Scene with Custom Objects (Fixed Tilt 2-up with Torque Tube + CLEAN Routine + CustomObject).ipynb</td>
</tr>
<tr>
<td>5</td>
<td>Medium Level Example - Bifacial Carports and Canopies + sampling across a module!.ipynb</td>
</tr>
<tr>
<td>6</td>
<td>Advanced topics - Understanding trackerdict structure.ipynb</td>
</tr>
<tr>
<td>7</td>
<td>Advanced topics - Multiple SceneObjects Example.ipynb</td>
</tr>
<tr>
<td>8</td>
<td>Advanced topics - Calculating Power Output and Electrical Mismatch.ipynb</td>
</tr>
<tr>
<td>9</td>
<td>Advanced topics - 1 axis torque tube Shading for 1 day (Research documentation).ipynb</td>
</tr>
</tbody>
</table>

[https://github.com/NREL/bifacial_radiance/tree/master/docs/tutorials](https://github.com/NREL/bifacial_radiance/tree/master/docs/tutorials)
Canopies and Carports
Canopies and Carports
Canopies and Carports

Module 1

Scanning

(a0.0.a0, a0.0.a1, a0.0.a2, a0.0.a3)

(Tutorial 5)
Roofs, Cars, and Different albedo sections

A wild white car appears!
Multiple Scene Objects

Sampling this Module from sceneObj 2

https://medium.com/@MySunIndia/limited-time-opportunity-to-get-huge-subsidy-for-rooftop-solar-in-india-for-residential-users-6991f66a1ede

(Tutorial 7)
75 kW HSAT | 5 bifacial technologies

Electrical Mismatch Losses

Averaging Irradiance for the module ~1031 W/m²

Detailed Irradiance value

\[ L_{DC} = M = 1 - \frac{P_1}{P_0} \]
GUI (Visual Interface)
In Spyder Console:
import bifacial_radiance
bifacial_radiance.gui()
A portion of the research was performed using computational resources sponsored by the Department of Energy's Office of Energy Efficiency and Renewable Energy and located at the National Renewable Energy Laboratory.

This work was authored in part by the National Renewable Energy Laboratory, operated by Alliance for Sustainable Energy, LLC, for the U.S. Department of Energy (DOE) under Contract No. DE-AC36-08GO28308. Funding provided by the U.S. Department of Energy's Office of Energy Efficiency and Renewable Energy (EERE) under Solar Energy Technologies Office (SETO) Agreement Number 34910, and Award Number DE-EE0008564. The views expressed in the article do not necessarily represent the views of the DOE or the U.S. Government. The U.S. Government retains and the publisher, by accepting the article for publication, acknowledges that the U.S. Government retains a nonexclusive, paid-up, irrevocable, worldwide license to publish or reproduce the published form of this work, or allow others to do so, for U.S. Government purposes.