


Presentation

RayTracing project






Why using Python version vs. Matlab old one ?






- Scalability
- Performances
- Intuitive interface
- Object-oriented programming
- Python is free!
- 3D animations for debugging purposes




RayTracing



Project ID: 1164

 Ray Tracing  Electromagnetism  Radar Coverage

 **59** Commits  **1** Branch  **0** Tags  **9.3 MB** Files  **9.3 MB** Storage

The RayTracing project provides tools to simulate ray tracing for radar coverage, etc.



master 


RayTracing /  


History


Find file

Web IDE

Clone 

 **Upload New File**
Jérôme Eertmans authored in 5 minutes

49ec6030 

How to setup the project ?

- Go to <https://forge.uclouvain.be/eertmans/RayTracing>
- Follow the 'README.md' instructions:
 1. Clone the repository
 2. Install Python
 3. Setup a virtual environment
 4. Install the packages

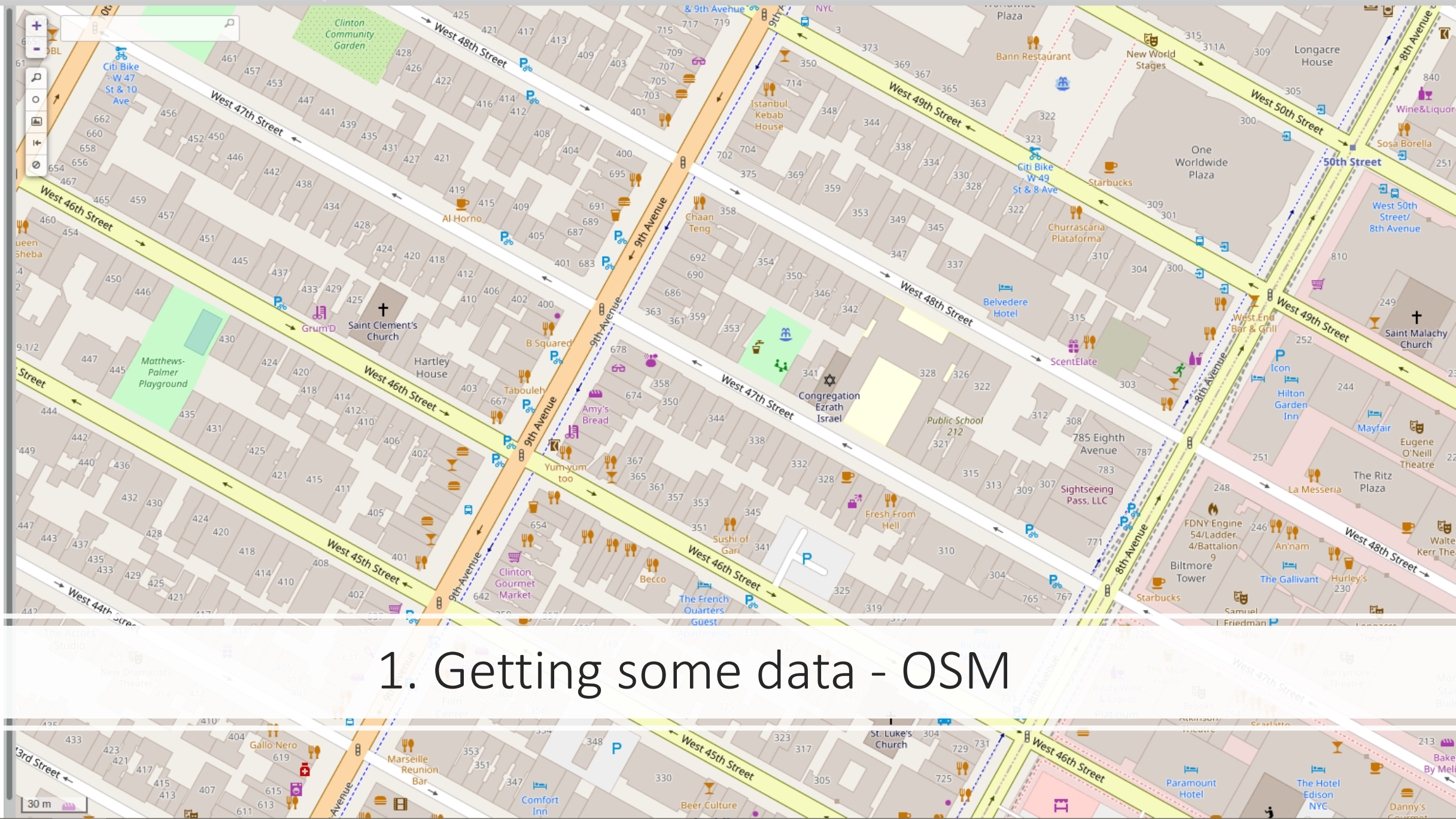
How to use the project ?

Let's go through an example !

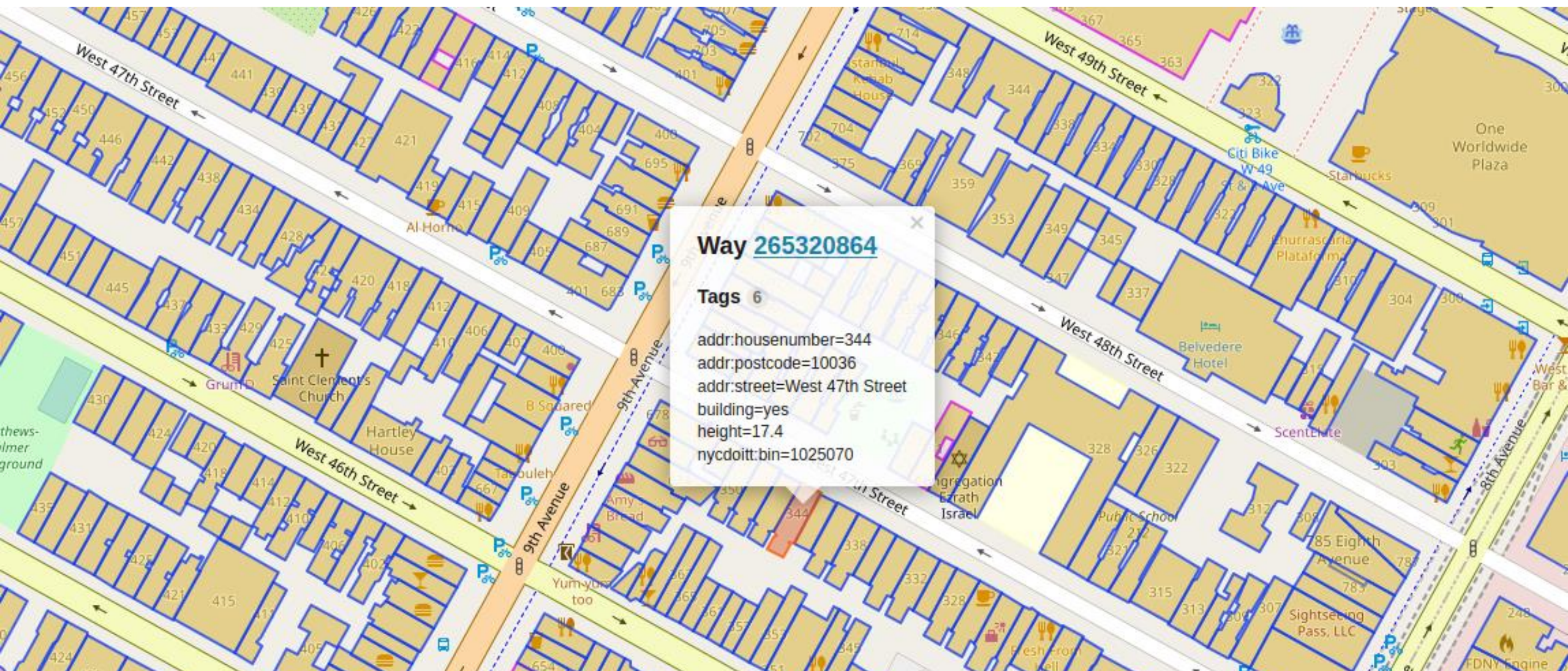
1. Getting some data

- Instructions in <https://forge.uclouvain.be/eertmans/RayTracing/-/tree/master/data>
- Data can be obtained/built via:
 - Open source projects (ex.: OpenStreetMap)
 - Open source softwares (ex.: QGIS)
 - Commands using the `geometry.py` library or the command-line tools

y is
y OSM
s for
visible
to run
click
save the
e looking
reas and
wnloads
al data
e
models
lex
ent
types and
s
available
dbuilding



building
multiply
box}});
ut qt;

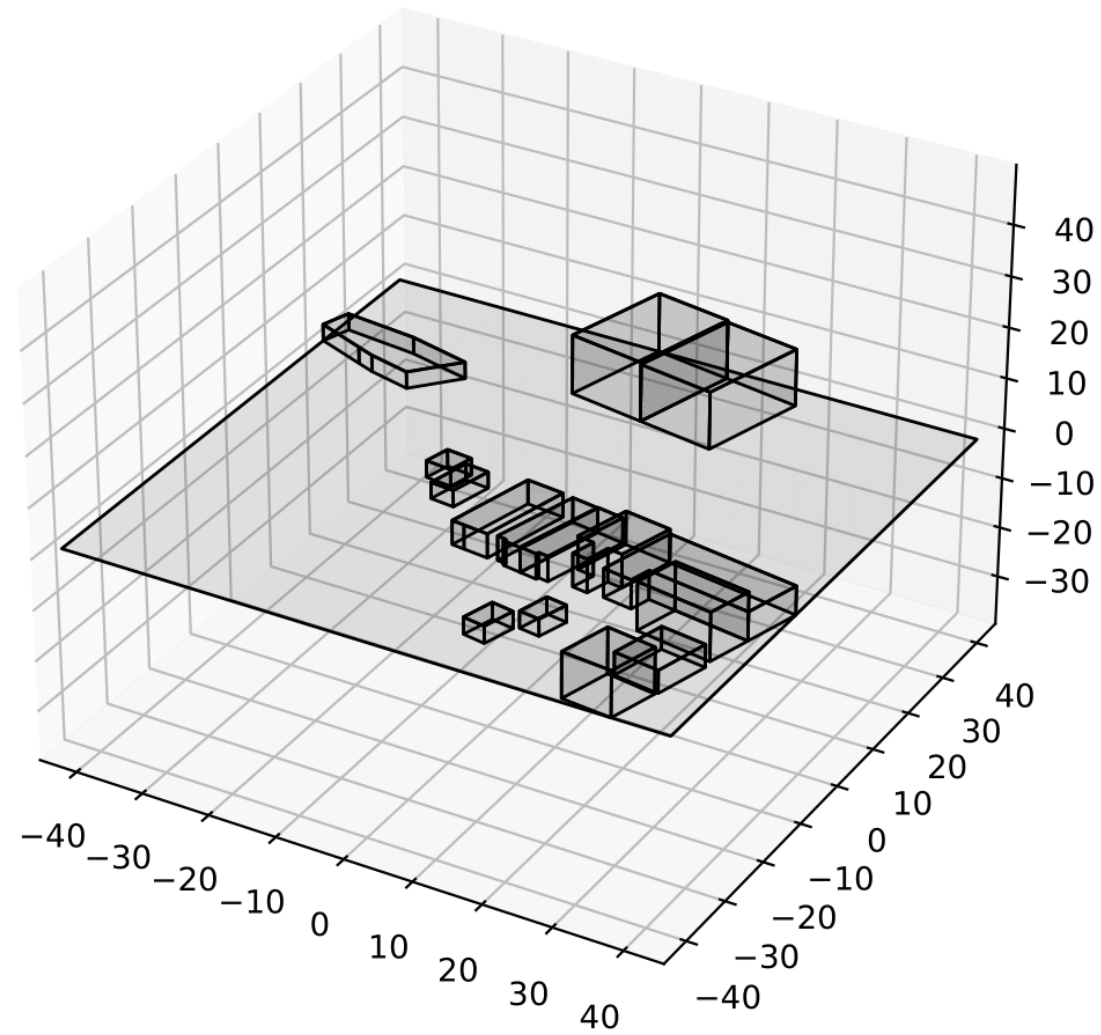


1. Getting some data - OSM

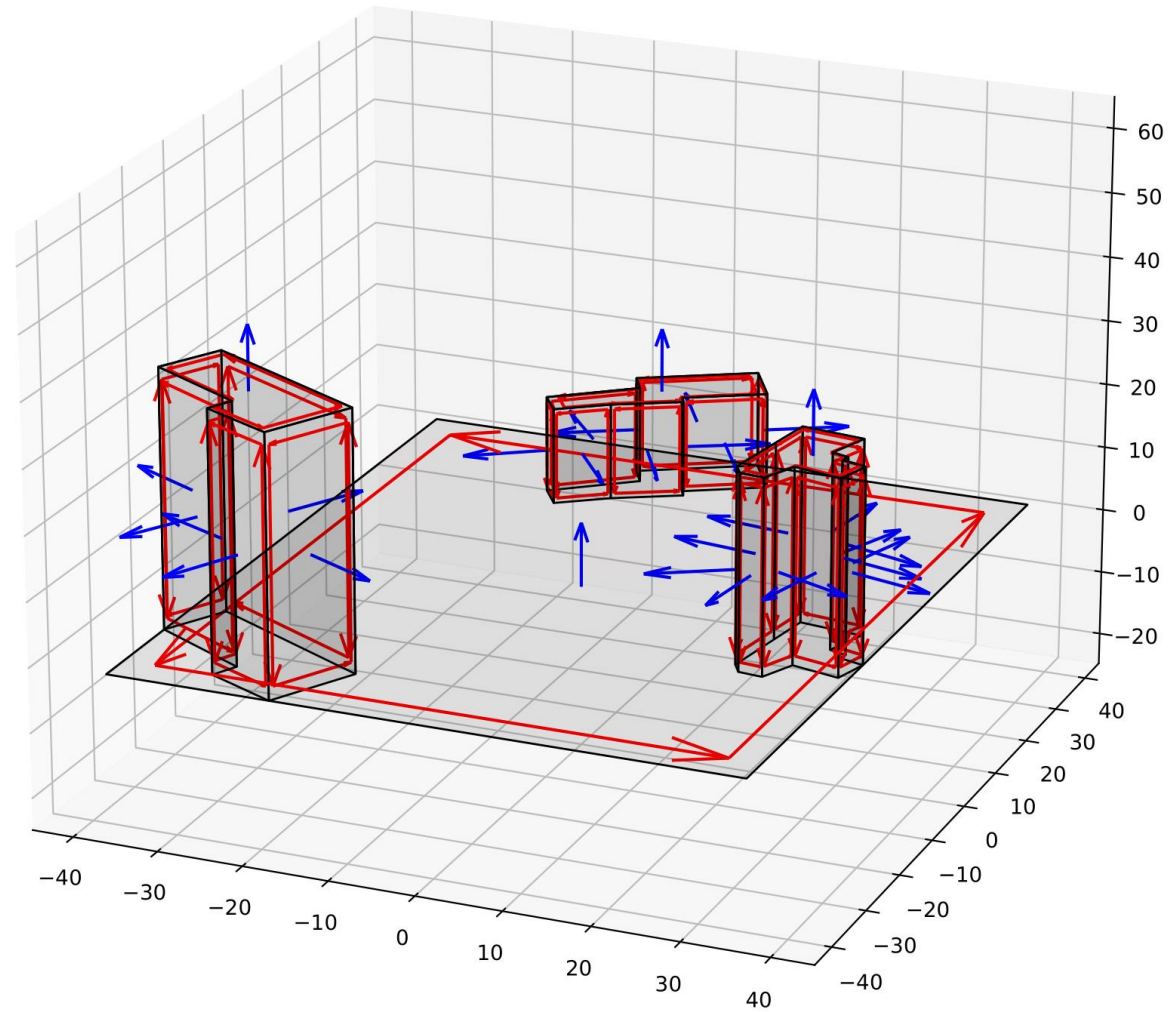
2. Plotting the data

Good to verify:

1. Correctness of data
2. Orientation of surfaces
3. Detection of sharp edges
4. Visibility matrix (NP problem)

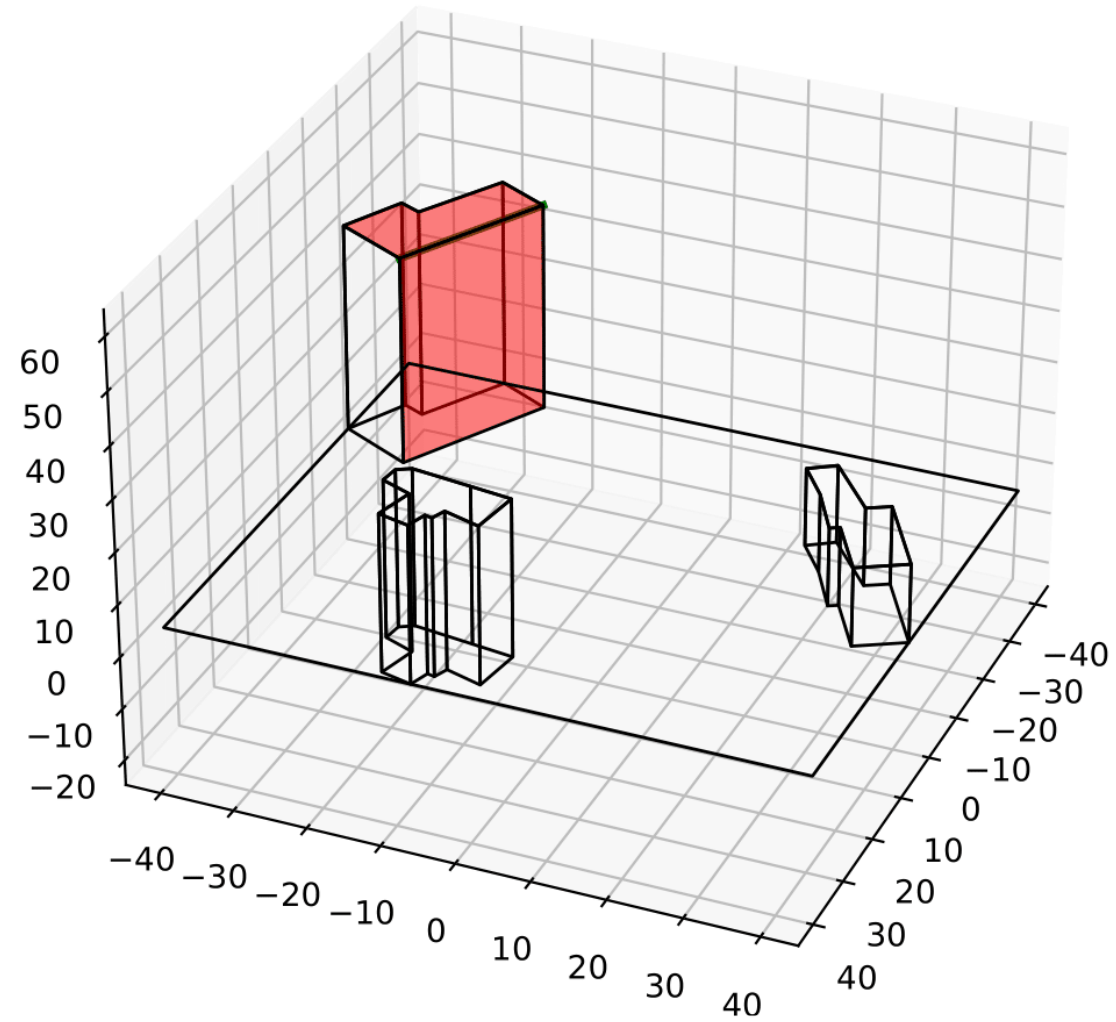


2.1 Correctness of data



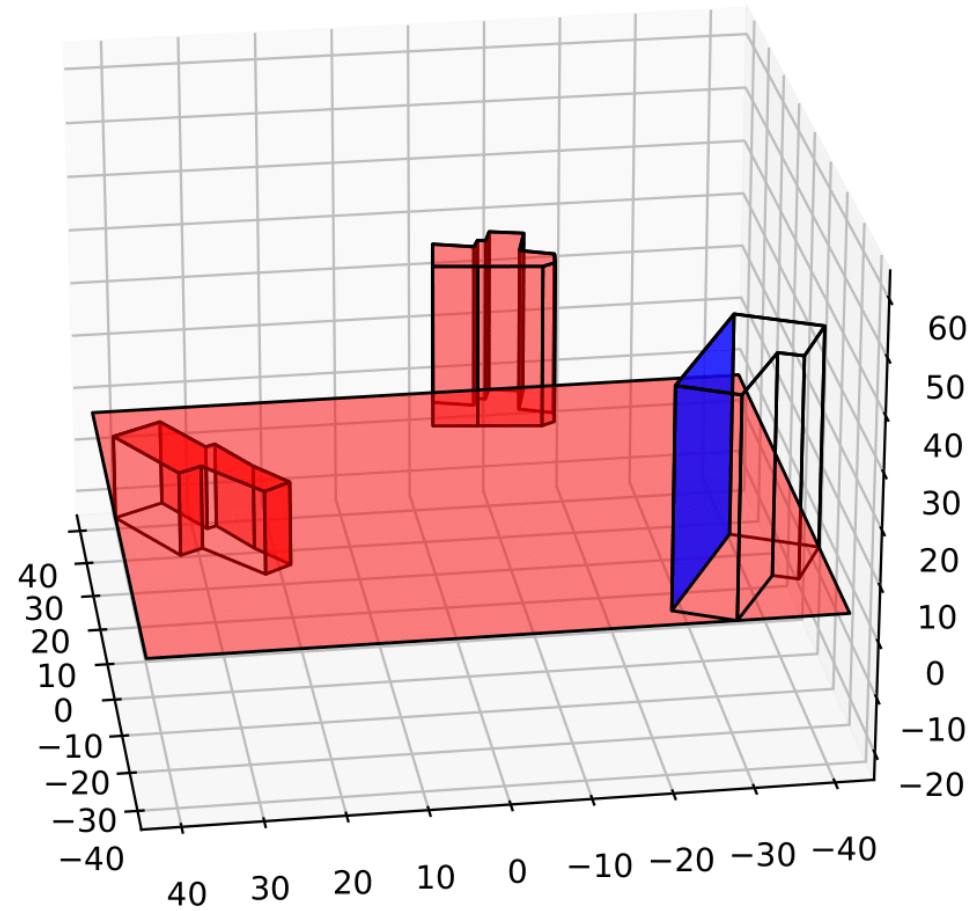
2.2 Orientation of surfaces

Press 'q' to quit, 'space' to play/pause



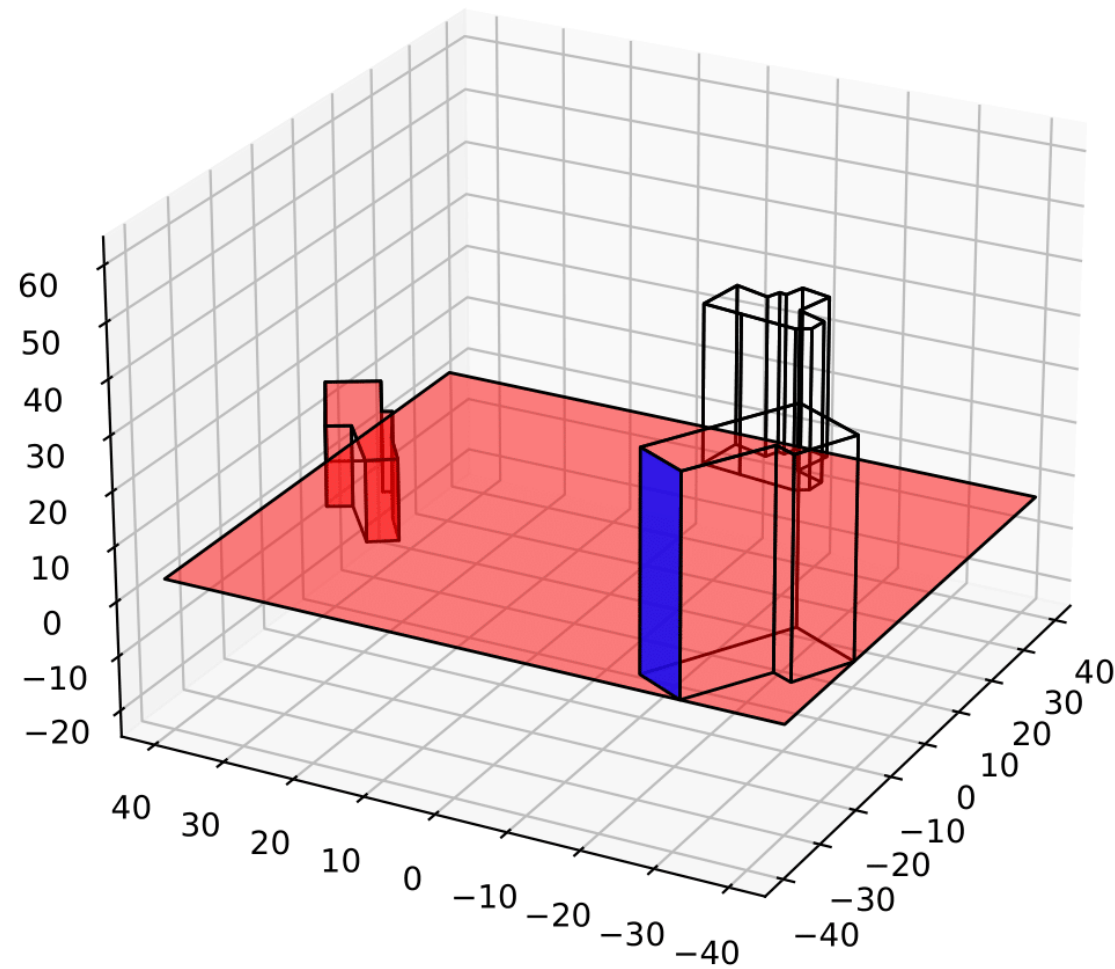
2.3 Detection of sharp edges

Press 'q' to quit, 'space' to play/pause



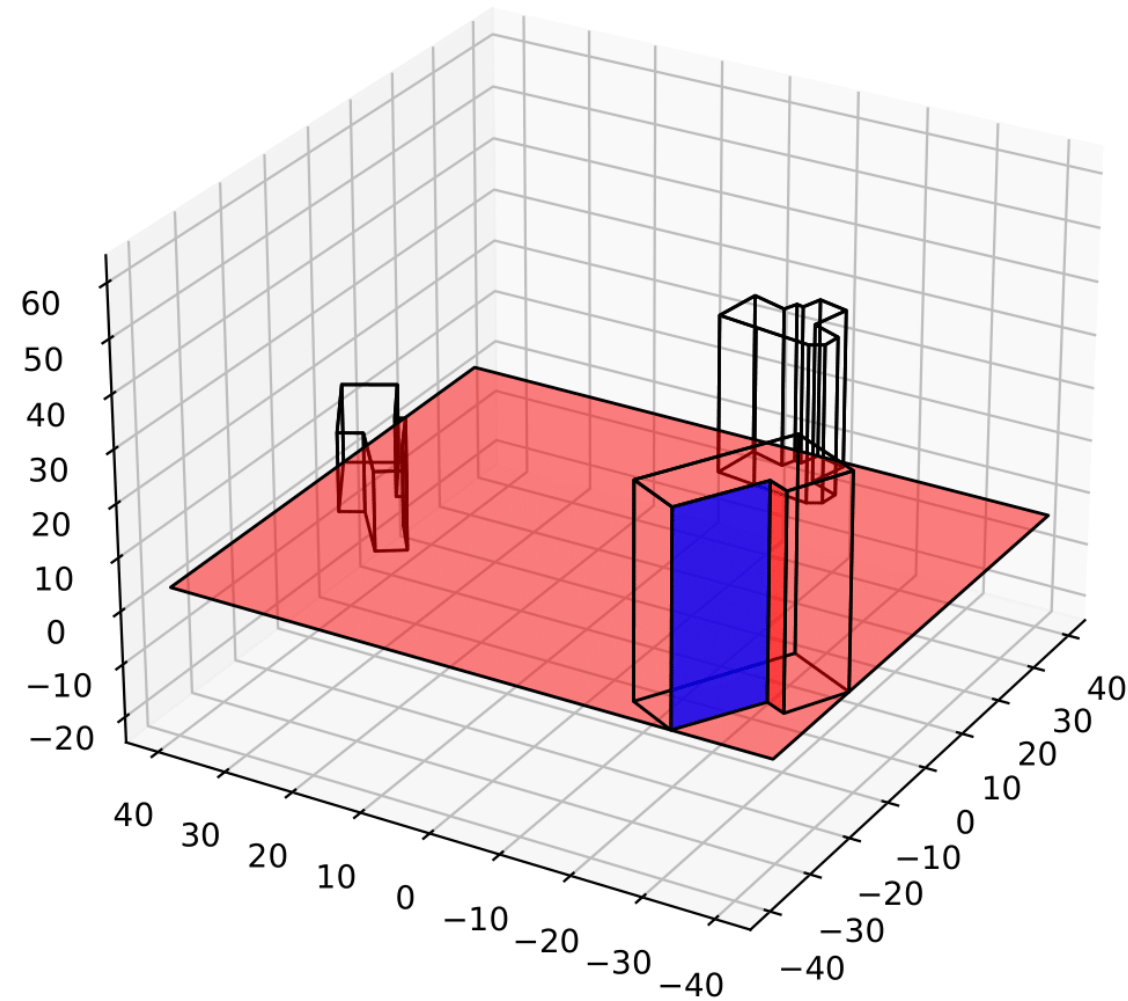
2.4-A Visibility matrix (NP problem)

Press 'q' to quit, 'space' to play/pause



2.4-B Visibility matrix (NP problem)

Press 'q' to quit, 'space' to play/pause



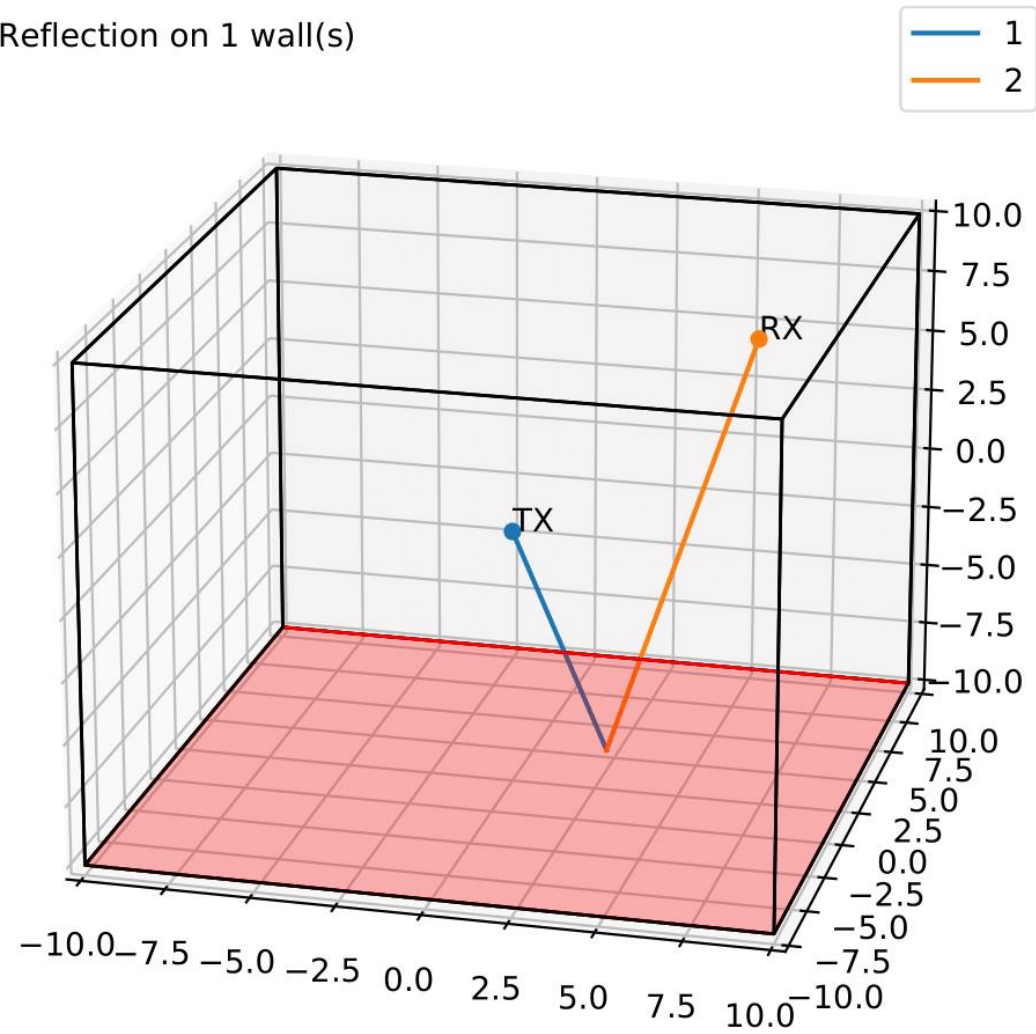
2.4-C Visibility matrix (NP problem)

3. Ray tracing

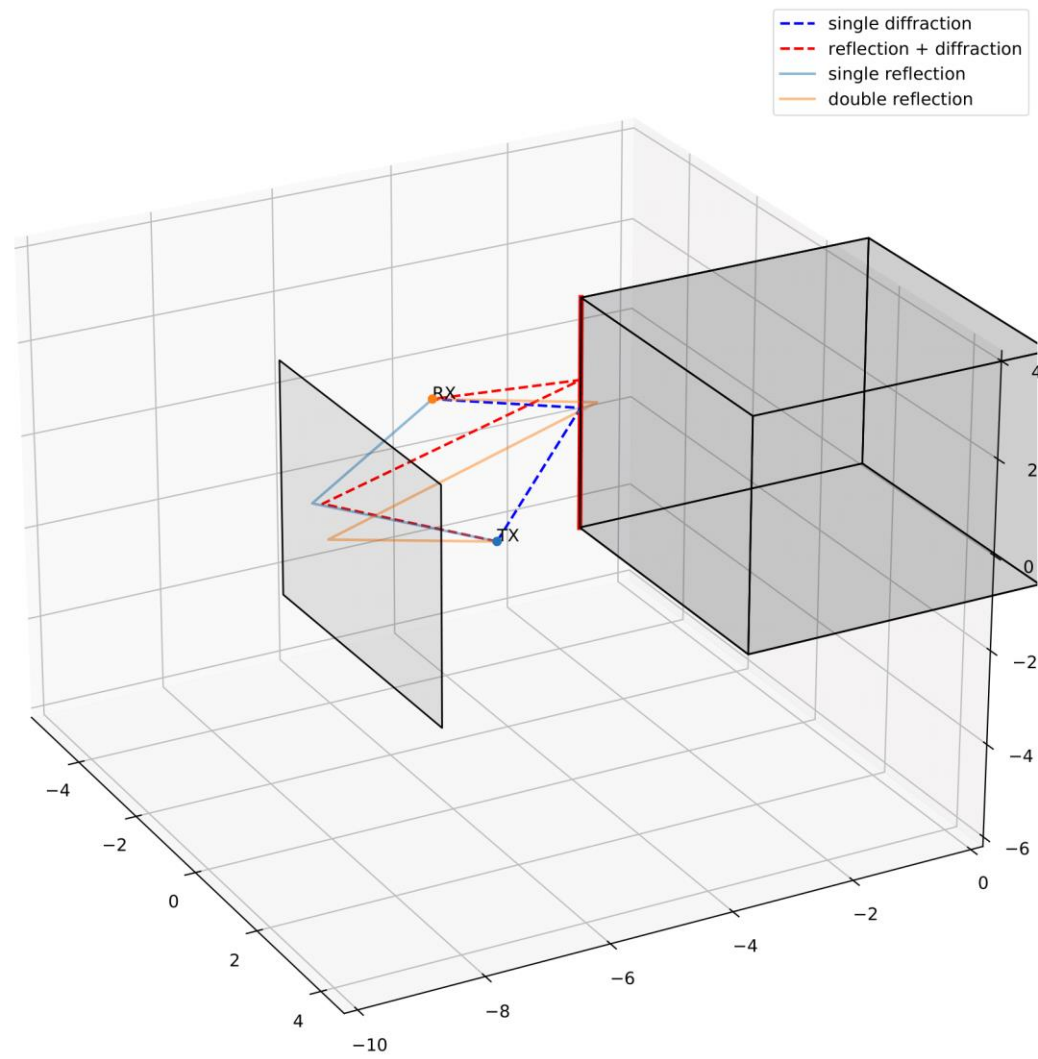
A thick yellow horizontal bar spanning the width of the slide, with a vertical yellow bar on the right side.

1. Reflection(s)
2. Diffraction
3. Multiple reflection(s) and diffraction

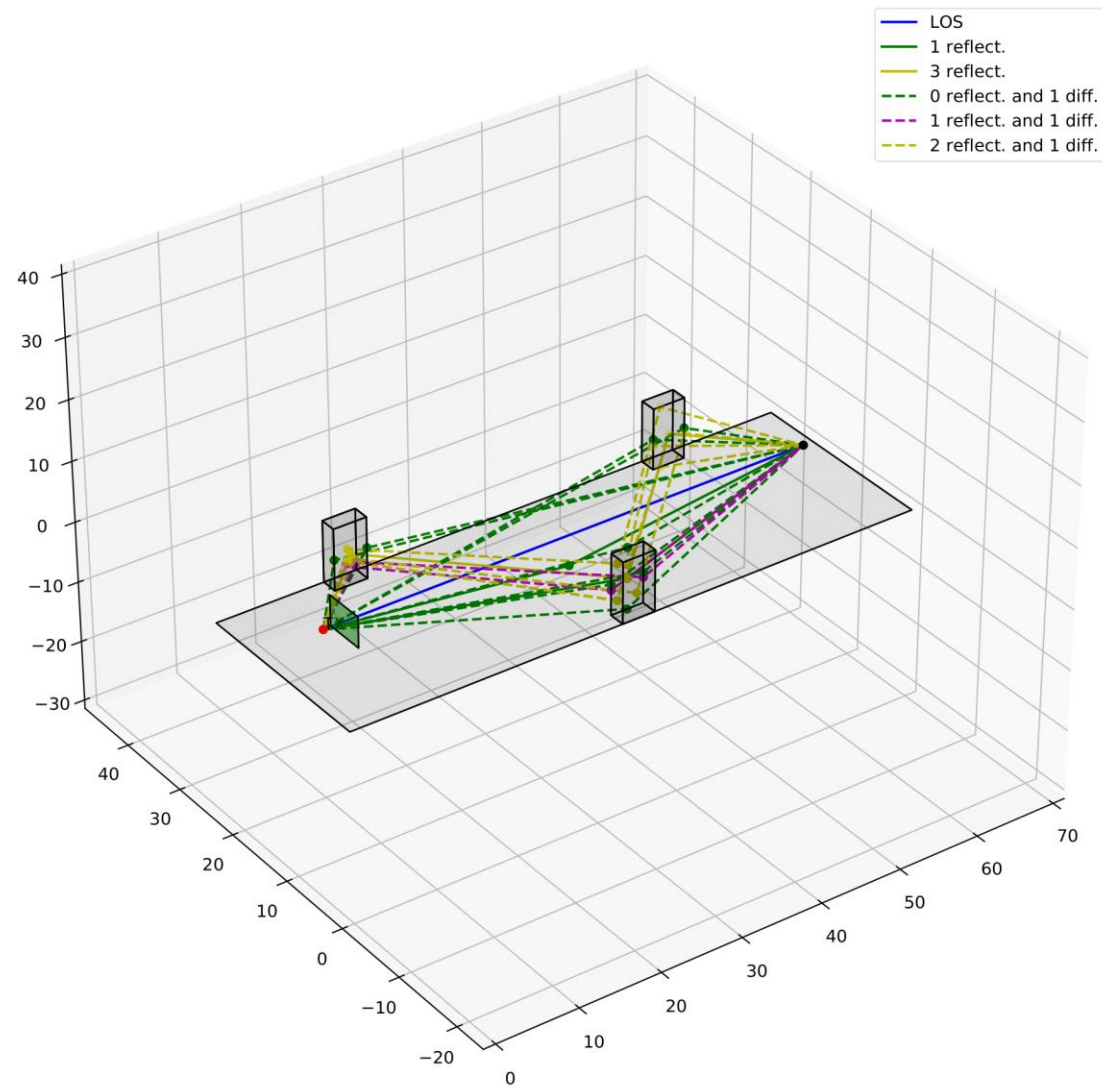
Reflection on 1 wall(s)



3.1 Reflection(s)



3.2 Diffraction

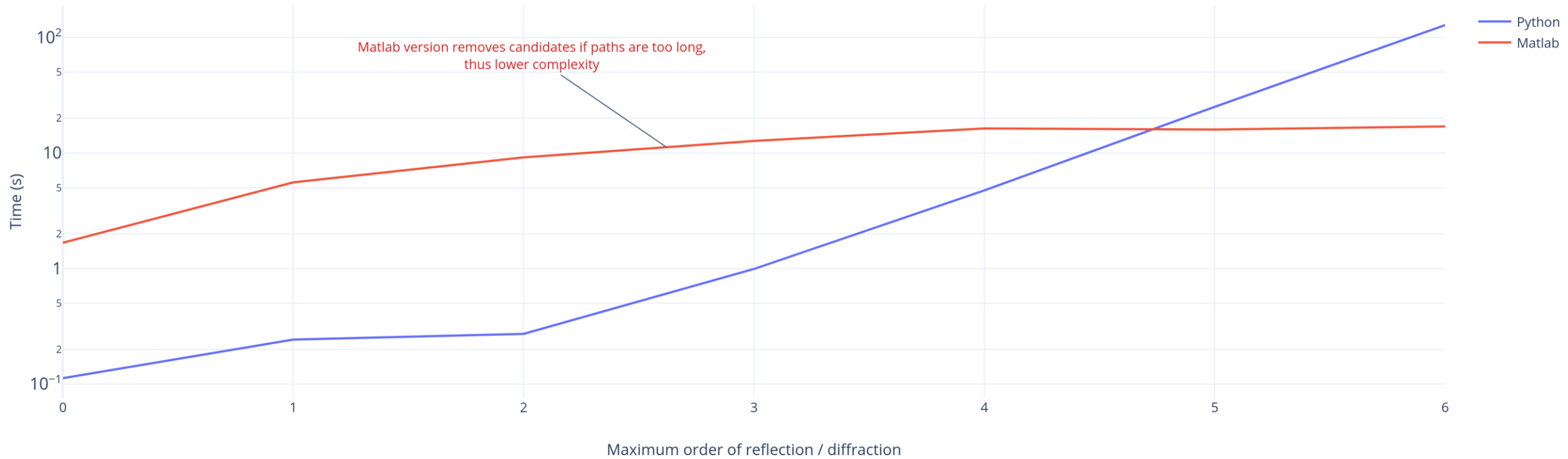


3.3 Multiple reflection(s) and diffraction

4. Post-processing

- Save geometries in json-like files
- Re-use or modify geometries
- Compute received power at given position(s) from ray tracing
- Etc.

Benchmark of RayTracing for various implementations



4.1 Comparing Matlab and Python versions

5. Read the documentation

A thick yellow horizontal bar spans the width of the slide, with a vertical yellow bar extending downwards from its right end.

You can generate the documentation and read it:

- It is better than reading thousands of lines of code
- It can be open in any browser

6. Questions ?

If you find any problem, I encourage you to use Gitlab's issues and merge requests:

- Issues at <https://forge.uclouvain.be/eertmans/RayTracing/-/issues>
- Requests at https://forge.uclouvain.be/eertmans/RayTracing/-/merge_requests

Feel also free to contact me: Jérôme Eertmans