frads
framework for radiance simulation
Status

- >30 simulation software tools use Radiance worldwide
- Application: modeling daylight, solar radiation, PV, and more
- Functionality exposed through system calls of a combination of programs: Unix toolbox model
Challenges

- Complicated workflow > user error (difficult to identify)
- Steep learning curve: sampling basis? direction? resolution?
- Slow adoption for advanced modeling capabilities <-> low rate of new tech adoption
frads
intermediate layer to Radiance core engine

- Standardized workflows
- Workflow automation and scripting (command-line interface + python library)
- Command-line programs: for most of the day-to-day use cases
- Python library: for customized workflow and tool development
frads
command-line programs

1. mrad: N-phase method simulation
2. genmtx: generic matrix generation
3. genfmtx: generate matrix for non-coplanar systems
4. rglaze: single and double pane glazing modeling
5. dctsnp: matrix multiplication using BLAS linear algebra library
6. geombsdf: geometric BSDF modeling
7. genglazing: system BSDF through WinCalc(WINDOW)
1. mrad
N-phase method simulation

1. Create folder structure:
   - Objects: materials and scene objects
   - Resources: BSDF xml and weather files

2. mrad init

3. mrad run
1. mrad

N-phase method simulation: five-phase example

mrad init --latlon 37.7 -122.2 -o '*.rad' -m 'material*.mat' -w '*glass.rad' -g floor.rad .6 .7
1. mrad

N-phase method simulation: five-phase example

# default.cfg

[SimulationControl]
vmx_basis = kf
vmx_opt = -ab 3 -ad 64
fmx_basis = kf
fmx_opt
smx_basis = r1
dmx_opt = -ab 1 -ad 64 -c 9
dsmx_opt = -ab 1 -ad 262144 -lw 1e-9
dsmx_opt = -ab 0 -dj 0 -st 0
ray_count = 1
pixel_jitter = 0.7
separate_direct = True
nprocess = 4
overwrite = True
method

[Site]
wea_path =
latitude = 37
longitude = 122
zipcode
daylight_hours_only = True
start_hour
end_hour
orientation = 0

[Model]
material = materials.mat
window_paths = lower_glass.rad upper_glass.rad
scene = windowframe.rad overhang.rad ground.rad
extwalls.rad floor.rad desks.rad horframe.rad
cubefabric.rad deskleg.rad ceiling.rad cubeframe.rad
cubefabric.rad deskleg.rad ceiling.rad cubeframe.rad
chairs.rad walls.rad
ncp_shape
window_xml = blinds30.xml klems_aniso_high.xml
window_control = 0 1
window_cfs

[Raysenders]
view = -vf v1a.vf -x 800 -y 800
grid_surface = floor.rad
grid_height = 2.5
grid_spacing = 2
1. mrad

N-phase method simulation: five-phase example

mrad run default.cfg

Illuminance and luminance results will be produced in the "Results" folder

Demo
2. genmtx
generic matrix generation

>> genmtx -h

Generate flux transport matrix

optional arguments:
-h, --help                      show this help message and exit
-st {s,v,p}                    Sender object type: (s)urface, (v)iew, (p)oint
-s SENDER                      Sender object: view | grid point | .rad file
-r RECEIVER [RECEIVER ...]     Receiver objects, sky | sun | *.rad files
-i OCTREE                      Scene octree file
-o OUTPATH [OUTPATH ...]       Output file path | directory
-env ENV [ENV ...]             Environment files
-rs {r1,r2,r4,r6,kf,sc25}      Receiver sampling basis, ....
-ss SENDER_BASIS               Surface sender sampling basis: kf|l|r1|r2|...
-rc RECEIVER_OFFSET            Move receiver surface in normal direction
-so SENDER_OFFSET              Move sender surface in normal direction
-opt OPTION                    Simulation parameters enclosed in quotes
-rc RAY_COUNT                  Ray count
-res RESOLU RESOLU             Image res., default=[800, 800]
-smx SMX                       Sky matrix file
-wpths WPTH [WPTH ...]         window files paths
-v, --verbose                  verbose mode
2. gen_mtx

generic matrix generation: example

• View matrix:
  >> genmtx -s grid.pts -st p -r window.rad -ro 0.05 -env material.mat room.rad -o view.mtx

• Daylight matrix:
  >> genmtx -s window.rad -st s -so -0.05 -ss kf -r sky -rs r4 -env material.mat room.rad -o daylight.mtx

• Direct sun view matrix with sun and window culling:
  >> genmtx -s view.vf -st v -r sun -rs r6 -env material.mat room.rad window.rad -o sun_r6.mtx -smx oak_sun.smx -wpths window1.rad window2.rad
3. genfmtx
non-coplanar system modeling

```bash
>> genfmtx -w window.rad -ncp awning.rad -rs r4 -ss r4 -opt '-ab 1 -ad 4096 -c 5000' -env material.mat wall.rad -o awning.mtx
```
4. rglaze

glazing unit modeling

• Uses Optics files or accesses directly from IGSDB database

• Allows user to specify color space

• Example

  • rglaze -X layer1.optics layer2.optics

  • rglaze -D 4789 8765 -T $igsdb_api_token

    • *igsdb id
4. rglaze

glazing unit modeling

>> rglaze -X CLEAR_6.DAT CSR42_3.afg

void BRTDfunc Generic_Clear_Glass+Comfort_Select_R-42_on_Clear

if(Rdot,cr(fr(0.115330976),ft(0.714477533),fr(0.0769209297)),cr(fr(0.0769222478),ft(0.855362931),ft(0.173130587)))

if(Rdot,cr(fr(0.148201345),ft(0.722440685),fr(0.0813256)),cr(fr(0.0813254108),ft(0.893129389),fr(0.21087446)))

if(Rdot,cr(fr(0.248413717),ft(0.631038374),fr(0.0828252871)),cr(fr(0.0828252771),ft(0.885143028),fr(0.28307956)))

ft(0.714477533)*ft(0.855362931)

ft(0.722440685)*ft(0.893129389)

ft(0.631038374)*ft(0.885143028)

0 0 0 0 0 0 0 0 0 0 0 0 0
5. dctsnp
matrix multiplication with numpy

• Use numpy (BLAS) to accelerate matrix multiplication

• Takes regular Radiance matrix as inputs (ascii, float, double, xml), RGB weighting built-in

• No image-based matrix

• Example use:

  $\texttt{dctsnp} -m \text{view.mtx shade.xml daylight.mtx -s sky.mtx -w 47.4 119.9 11.6 -o output.txt}$
6. geombsdf (wip)
geometric BSDF generation and modeling

• Blinds or any customized macroscopic shading system

• BSDF created using either:
  • genblinds
  • Custom cross section (novel blind shape)
  • Custom periodic shape

• Automated BSDF generation with proxy geometry

• Places BSDF onto the window
7. genglazing (wip)
calls WINDOW to generate system BSDF

- Create BSDF for multilayer glazing and shading system
- Uses pyWincalc python library which calls Window_CalcEngine (the engine behind WINDOW)
- Can be connected to IGSDB API to pull data directly from glazing and shading database
- Configuration file enables scriptable, transferable, repeatable glazing+shading system modeling
7. genglazing (wip)
calls WINDOW to generate system BSDF

## User generated double_lowe.cfg file

```
[Standards]
optic_standard = standards/W5_NFRC_2003.std

[Blinds]
blind1 = 0.03 0.03 45 0 13919 # spacing depth tilt curve
material_igsdb_id

[Shade]
shade1 = 13810 # silverscreen 2%OF black
shade2 = 13227 # helio 3%of grey
shade3 =

[Glazing]
# glazing1 = 7972
glazing1 = ./products/CSR42_3.afg
glazing2 = 5216
glazing3 =
glazing4 =

[Gap]
gap1 = 0.0127 air
gap2 = .0127 air
gap3 =

[Glazing system]
width = 1
height = 1
system = glazing1 gap1 glazing2 gap2 shade1 # ext -> int
```

genglazing double_lowe.cfg -T
$igsdb_token -M

System BSDF (.xml) will then be generated
Other tools

- ep2rad: converts EnergyPlus model (epjson) to Radiance models
- genradroom: generates a side-lit shoe box model
- gengrid: generates a sensor grid base on a horizontal surface
- getwea: gets the .wea file from lat/lon, or zipcode(US)
Python library

Example: flip window surface normal by modifier

```python
from frads import radutil
windows = radutil.unpack_primitives("windows.rad")
for window in windows:
    polygon = radutil.parse_polygon(wall.real_arg)
    if window.modifier == 'white60':
        new_polygon = polygon.flip()
        new_primitive = radutil.Primitive(
            wall.modifier, wall.ptype, wall.identifier,
            wall.str_arg, new_polygon.to_real())
        print(str(new_primitive))
```

Thank you

Documentation: http://frads.readthedocs.io/

To install:

    pip install frads

or for head

    pip install git+https://github.com/LBNL-ETA/frads

Beta users welcome!!!

contact: taoningwang@lbl.gov

Acknowledgment

This work was supported by the Assistant Secretary for Energy Efficiency and Renewable Energy, Building Technologies Office, of the U.S. Department of Energy under Contract No. DE-AC02-05CH11231.