

Automate Radiance Workflows through Python

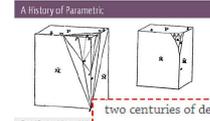
15th International Radiance Workshop
August 30, 2016

Mostapha Sadeghipour Roudsari
University of Pennsylvania

Sarith Subramaniam
Penn State University

Parametric Design

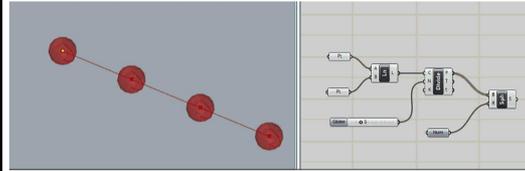
- What is [the History of] Parametric Design?
Go read this link: <http://www.danieldavis.com/a-history-of-parametric/>



two centuries of developments. A fairly convoluted path that missed potholes of theory and architecture in order to idle past idolised technology. Ultimately this history wasn't scholarly enough and wasn't needed for the argument of my thesis. I deleted it.

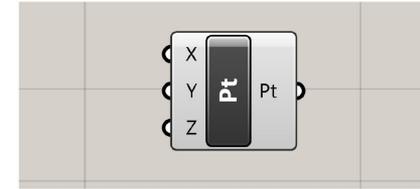
The first words I wrote in my deleted A History of parametric: The main before Challenging Technology Corporation and Dan Goldblatt's Identical, before the invention of the computer, and the birth of death. I assumed that I should start my thesis here in order to write the reader up on the past potholes of theory and architecture in order to idle past idolised technology. Ultimately this history wasn't scholarly enough and wasn't needed for the argument of my thesis. I deleted it.

What is Grasshopper? Why does it matter?



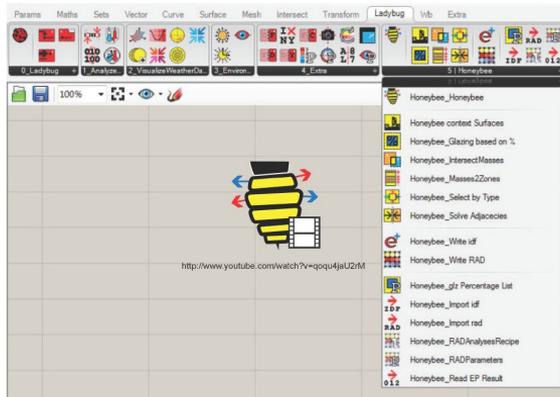
Visual Scripting

Pt = Rhino.Geometry.Point3D(X, Y, Z)

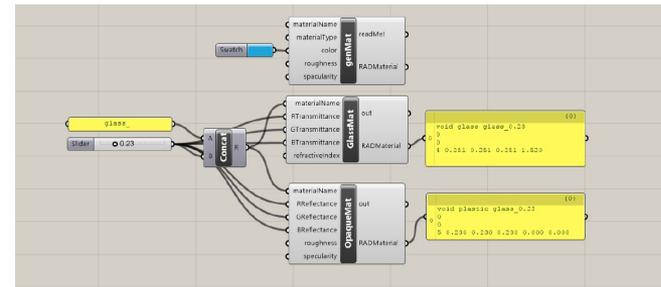


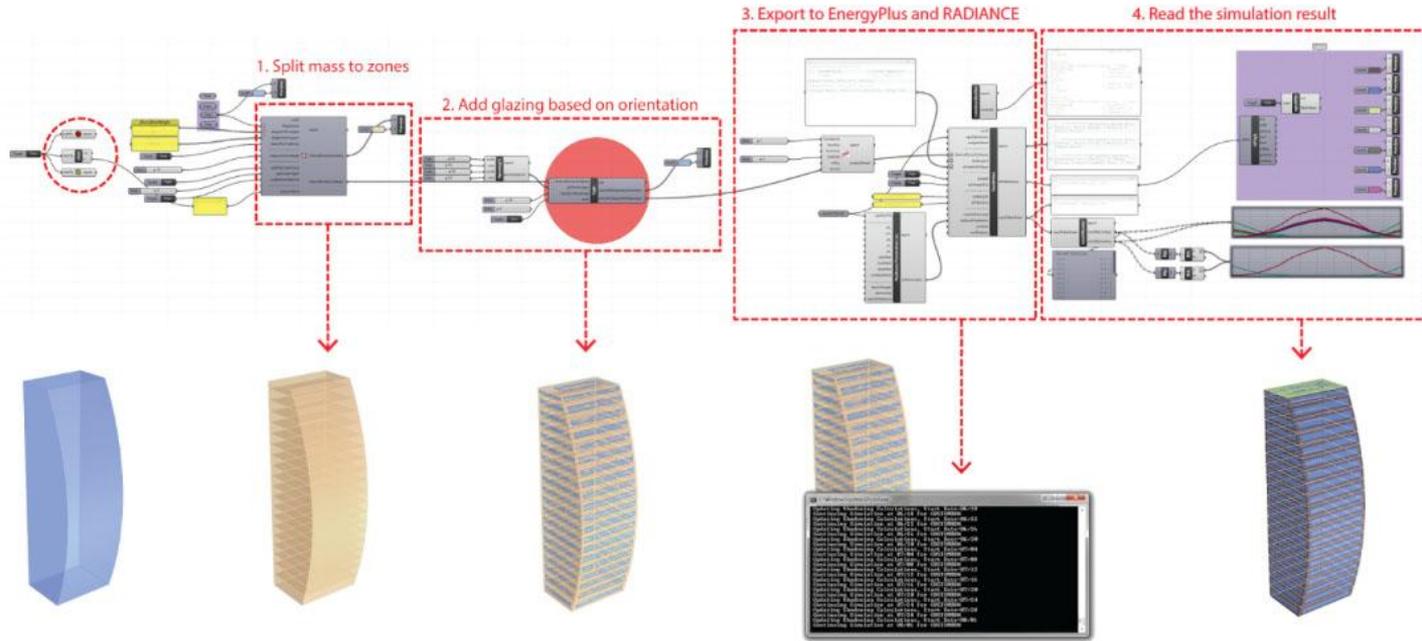
Parametric Analysis is the new !

Honeybee: Grasshopper <-> Radiance/Daysim/EnergyPlus

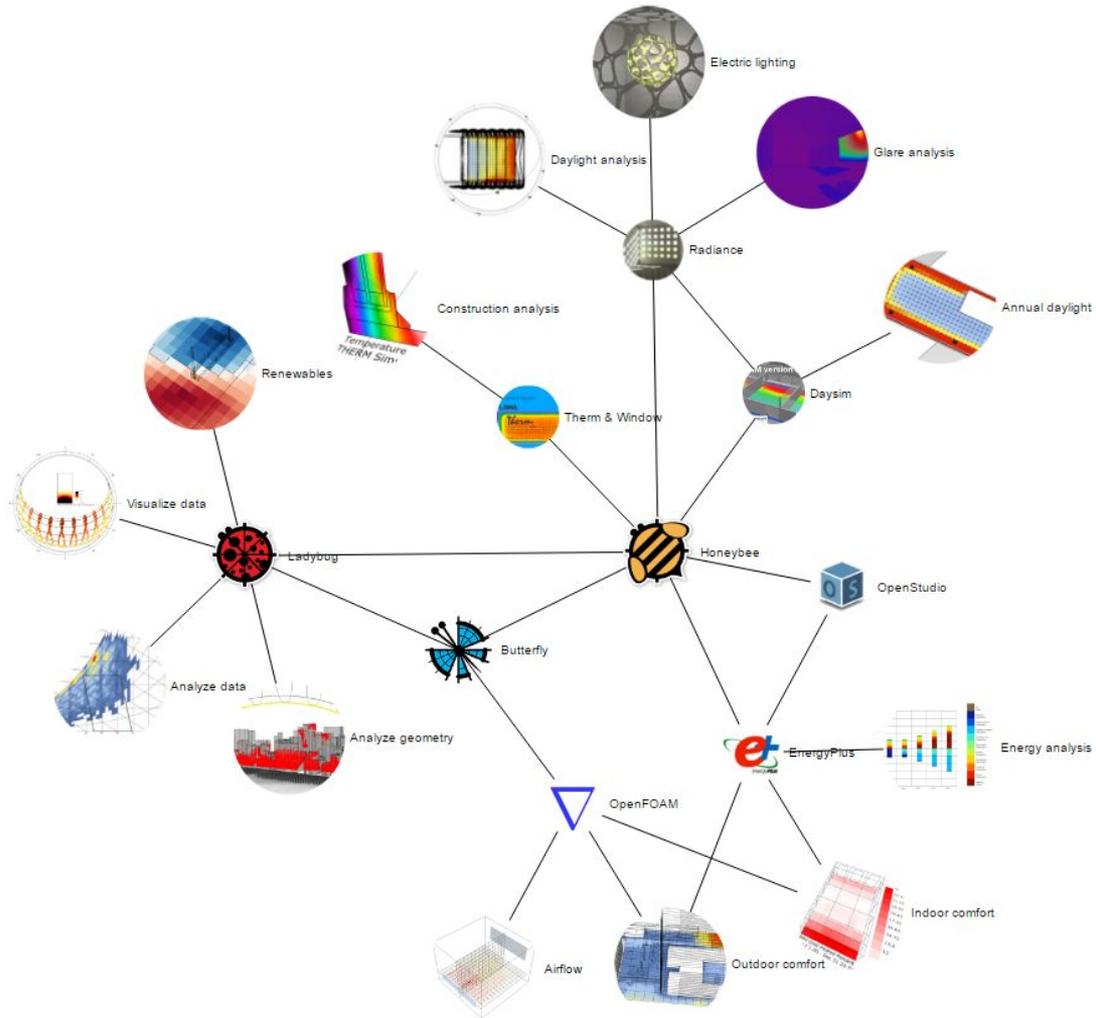


Honeybee: Multiple Material Components





2013



2016

All Groups My Groups Options Invite

Ladybug + Honeybee

Created by [Mostapha Sadeghipour Roudsari](#)
[View Groups](#)

Information



Ladybug and Honeybee are two open source environmental plugins for Grasshopper to help designers and engineers create an environmentally-conscious building design.

[Installation Instructions](#)
[Download Ladybug and Honeybee](#)
[Remove Old Version](#)

[Ladybug Primer](#), [Honeybee Primer](#)

[Example Files for Ladybug](#)
[Example Files for Honeybee](#)

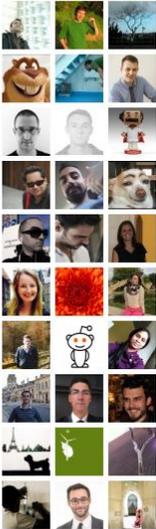
[Ladybug on GitHub](#)
[Honeybee on GitHub](#)

Use [this Reference](#) for your Publications.

Licensed under @license GPL-3.0+
<http://spdx.org/licenses/GPL-3.0+>

Send Message to Group

Members (1218)

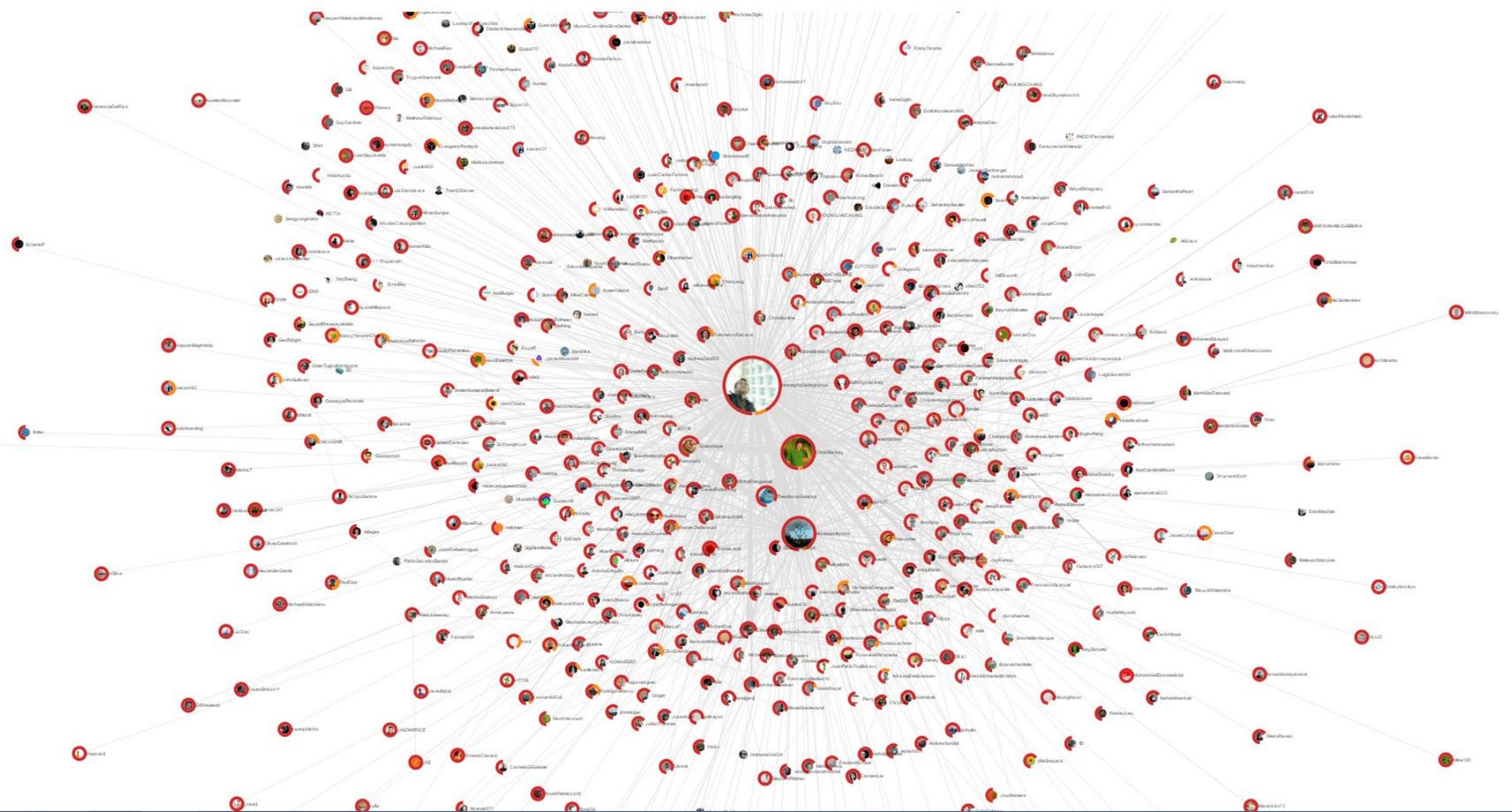


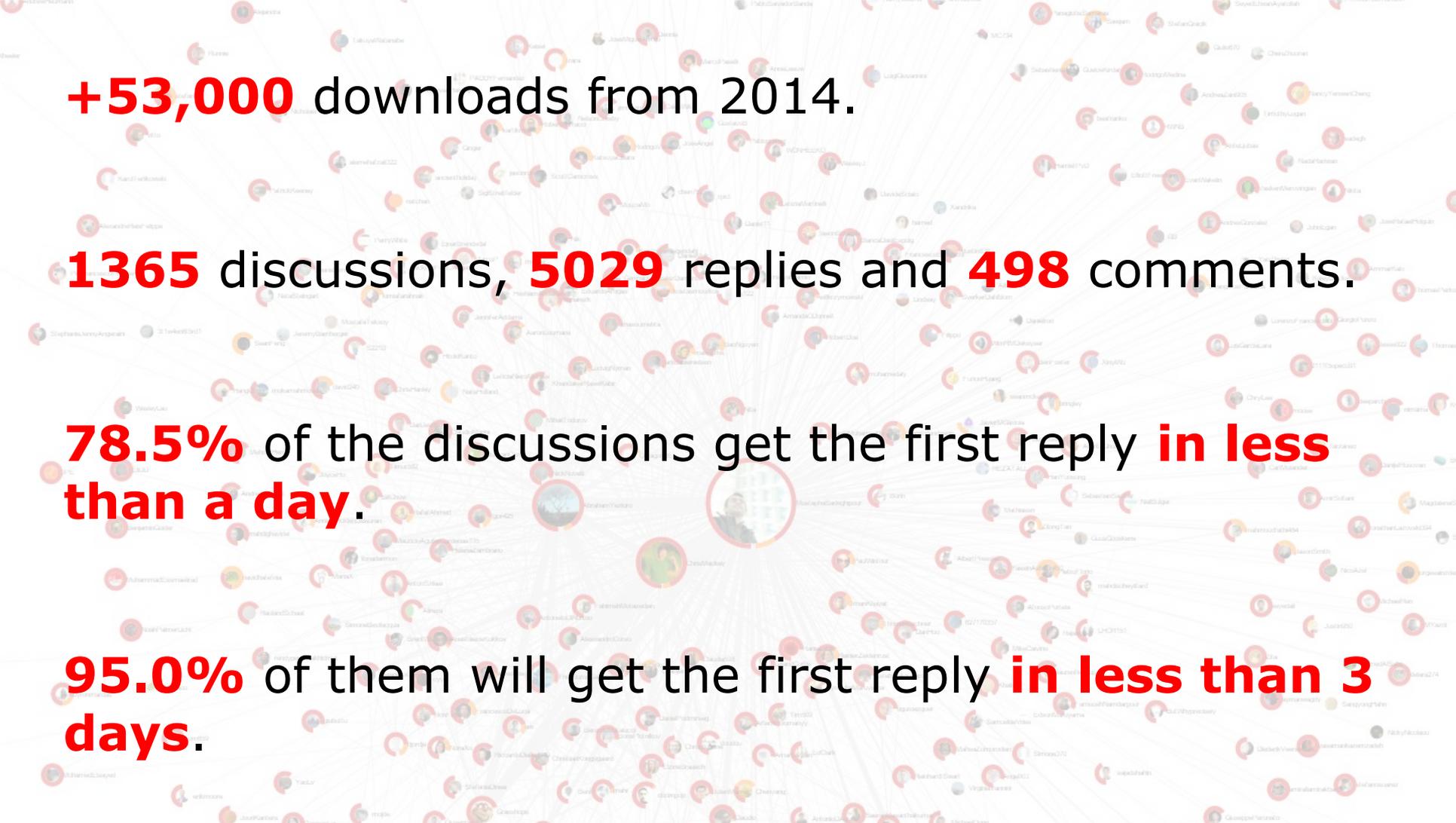
Stop Following New Members

[+ Invite More](#) [View All](#)

<http://www.grasshopper3d.com/group/ladybug>

. 1365 Discussions . 5029 Replies . 498 Comments . 660 Connections .



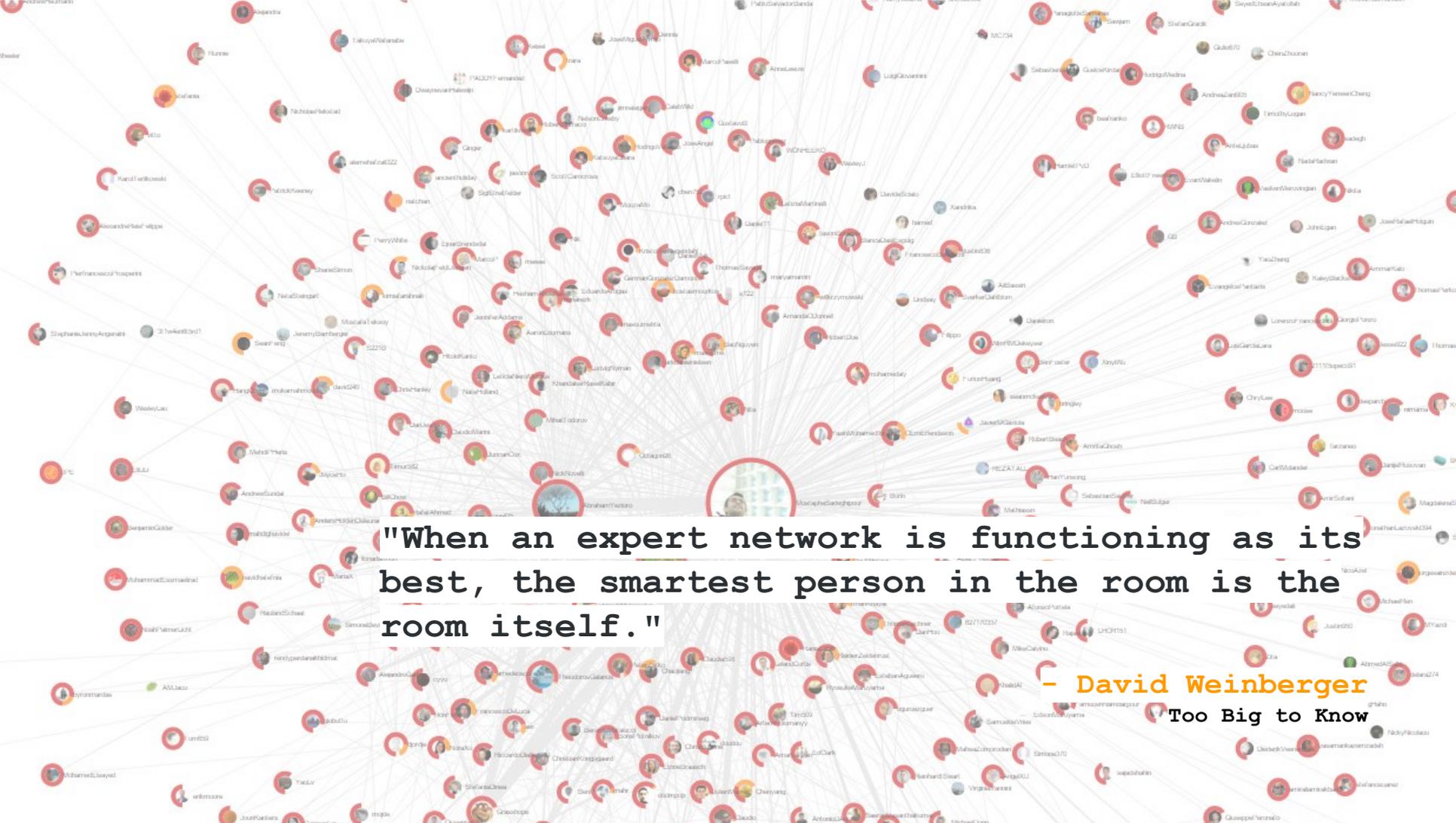


+53,000 downloads from 2014.

1365 discussions, **5029** replies and **498** comments.

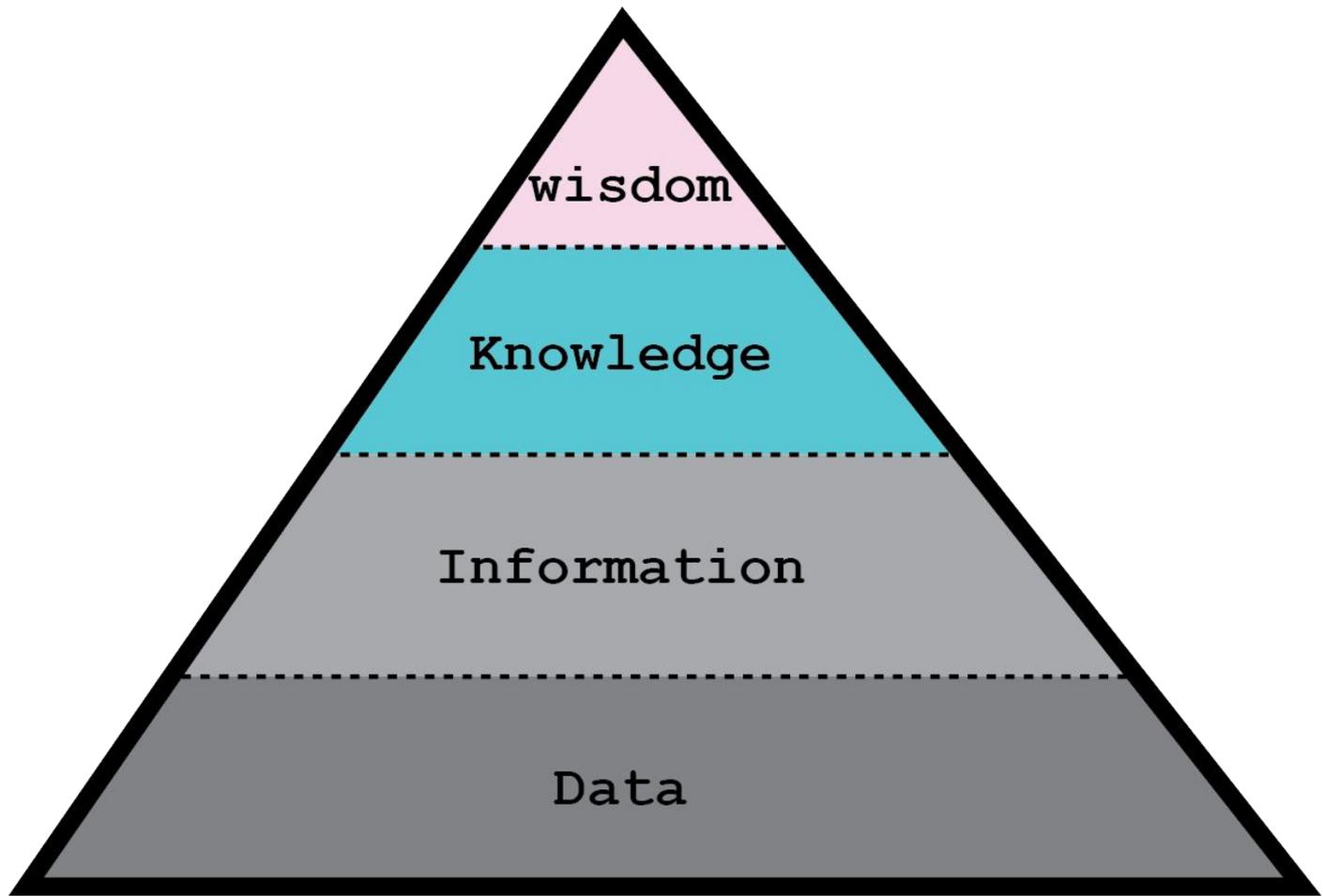
78.5% of the discussions get the first reply **in less than a day**.

95.0% of them will get the first reply **in less than 3 days**.



"When an expert network is functioning as its best, the smartest person in the room is the room itself."

- David Weinberger
Too Big to Know



wisdom

Knowledge

Information

Data

Share Code.

Share Knowledge.

Share Examples.

Share Stories!

Grasshopper

ALGORITHMIC MODELING FOR RHINO

[Home](#) [Galleries](#) [Download](#) [Tutorials](#) [Discussions](#) [Events](#) [My Page](#)

Options + Add



[discussion] Sky View Factor

Posted by [Grasshops](#) on February 10, 2016 at 10:52am in [Ladybug + Honeybee](#)
[← Back to Ladybug + Honeybee Discussions](#)

Dear bee and bugs,

I'd like to share and discuss with you my understanding of Sky View Factor, considering that it is an important concept frequently misunderstood.

Sky View Factor is first and foremost defined from the discussion of **radiation exchange between urban surfaces and the sky** in urban heat island research (See Oke's literature list below). It will be affected by the proportion of sky visible from a given calculation point on a surface (vertical or horizontal) as a result of the obstruction of urban geometry, but it is not entirely associated with the solid angle subtended by the visible sky patch/patches.

So, I think using "geometry way" to approximate Sky View Factor is not correct. Sky View Factor calculation shall be based on the first principle defining the concept: radiation exchange between urban surface and sky hemisphere:

(image extracted from Johnson, G. T., & Watson, 1984)

We may now define the *sky view-factor* (ψ_s) for the surface element ΔA as the fraction of radiant flux leaving ΔA which is intercepted by the sky. Then

$$\psi_s = \frac{1}{\pi R^2} \int_{S_v} \cos\phi dS. \quad (8)$$

It may be helpful to note that ψ_s is equal to the ratio of the radiant flux received by ΔA from the visible sky, to that which would be received by ΔA from an unobscured sky. This is shown by replacing F_e by F_s in (6) and integrating over S_v .

Therefore, I always refer to the following "theoretical" Sky View Factors calculated at the centre of an infinitely long street canyon with different Height-to-width ratios in Oke's original paper (1981) as the

Mostapha Sadeghipour Roudsari

Sign Out

Inbox (4 new)

Alerts

Friends - Invite

Settings

New Invites

6 Group Invites

Translate

Select Language

Powered by Google Translate

Search Grasshopper

Google Custom Search

Search

Photos



by [Jens Pedersen](#)

0 0



by [Jens Pedersen](#)

0 0



Share Code.

Share Knowledge.

Share Workflows.

Share Stories!

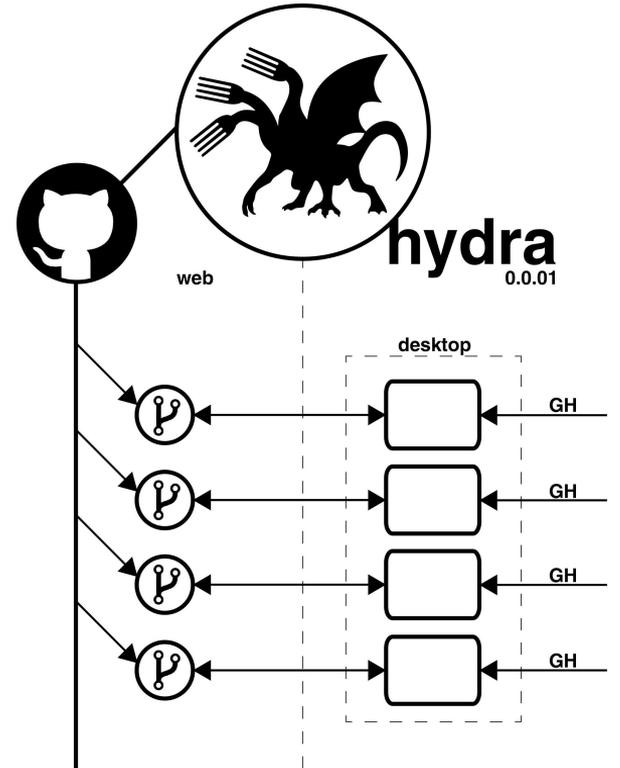
Welcome to Hydra!

last updated at Sun Apr 10 2016 05:08:51 GMT-0400 (Eastern Daylight Time)

username, component name, Newest

Radiant Asymmetry Discomfort	HUMAN UI - Adjusting Appearance	DynamoTest1	Urban Microclimate - Simple Spatial-UTCI
Adaptive Comfort Chart	RorschachShadowDiagram	Sky Exposure, Sky View, and Sky Component	Test Points and Legend Parameters: Ladybug Dynamo
Sunpath-Ladybug Dynamo	Sunlighthours-ladybug Dynamo	Import EPW Ladybug Dynamo	Import LBNL WINDOW Glazing Assembly for EnergyPlus
Import Glazing System from LBNL WINDOW	THERM Window Frame Construction	Analyze THERM Results	THERM Comparison of Stud Wall Constructions
IES Electric Lighting Grid-based calculations	IES Electric Lighting Basic Example	IES Electric Lighting Demo Setup	Thermal Bridging with THERM and EnergyPlus
THERM Export Workflow	EnergyPlus Window Shade Generator	Shapefile.to.Dynamo	Evaporative Cooling Tower
Area Capture	Urban Weather Generator Workflow	LadyBug Combine solarEnvelopes	Radiation in an Urban Canyon
Sky View in an Urban Canyon	Render Water Cube with Honeybee	Perforated mesh	Estimate Glare Potential Over a Year
Integrated Daylight and Energy simulation	Map Annual Comfort on Sunpath	Microclimate Map - Simple	Green Roof In Energy Model

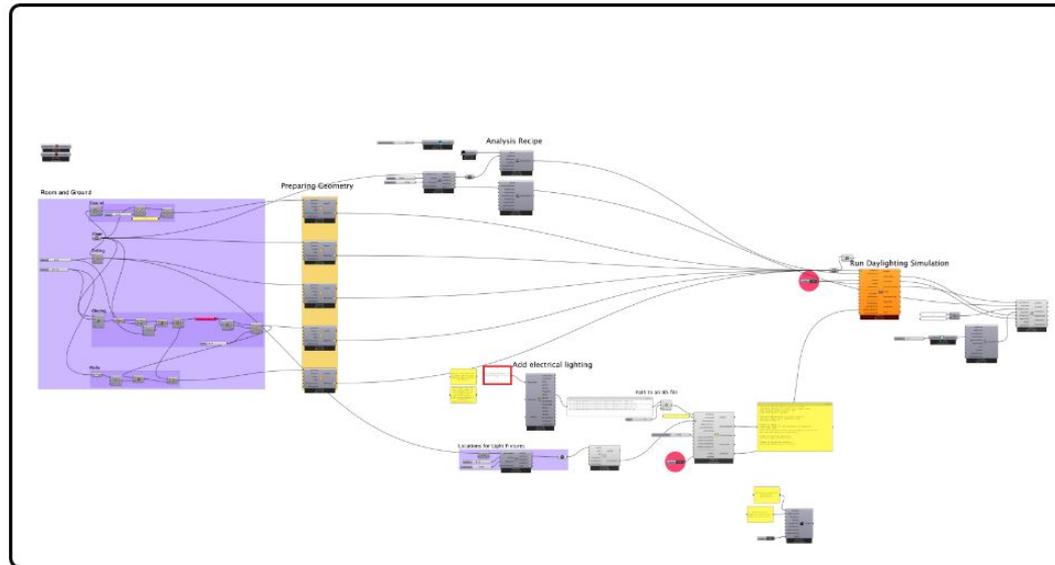
<http://hydrashare.github.io/hydra/>



IES Electric Lighting Grid-based calculations

[sharethis](#) 25 [Email](#) 0 [Share](#) 0 [Tweet](#) 0 [Share](#) 0 [Google +](#) 0

[ZoomToExtent](#)



[Download](#)

[#ES](#) [#Electric](#) [#Lighting](#) [#Grid-based](#) [#calculations](#)

Description

Indoor electric lighting calculations based on grid points

This file has been submitted by [sariths](#)

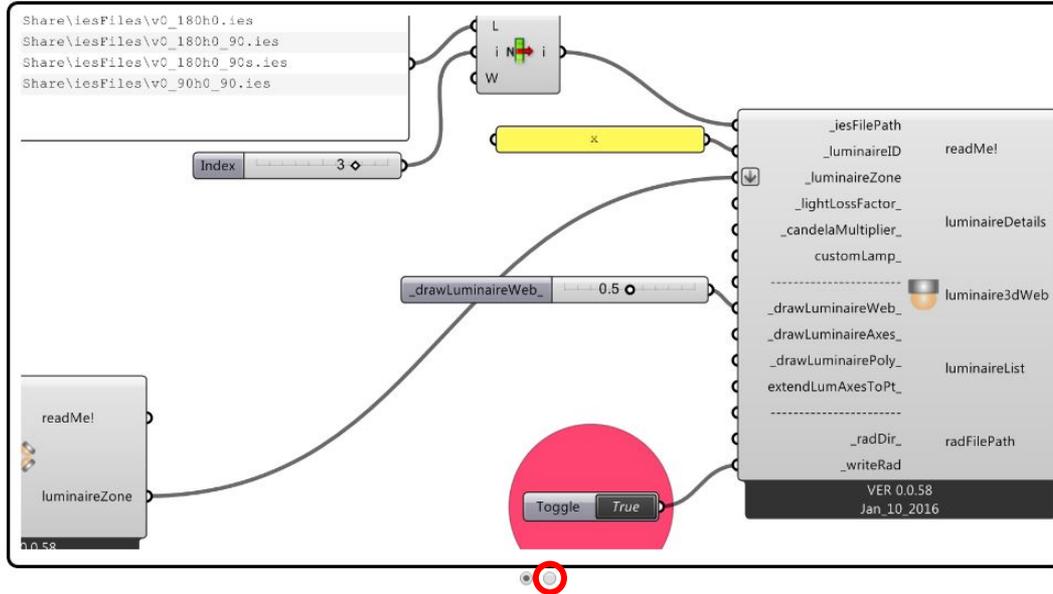
[Check out this example on Hydral](#)

Tags

IES Electric Lighting Grid-based calculations

[sharethis](#) 25
 [Email](#) 0
 [Share](#) 0
 [Tweet](#) 0
 [Share](#) 0
 [Google +](#) 0

[ZoomToExtent](#)



[Download](#)

[#ES #Electric #Lighting #Grid-based #calculations](#)

Description

Indoor electric lighting calculations based on grid points

This file has been submitted by [sariths](#)

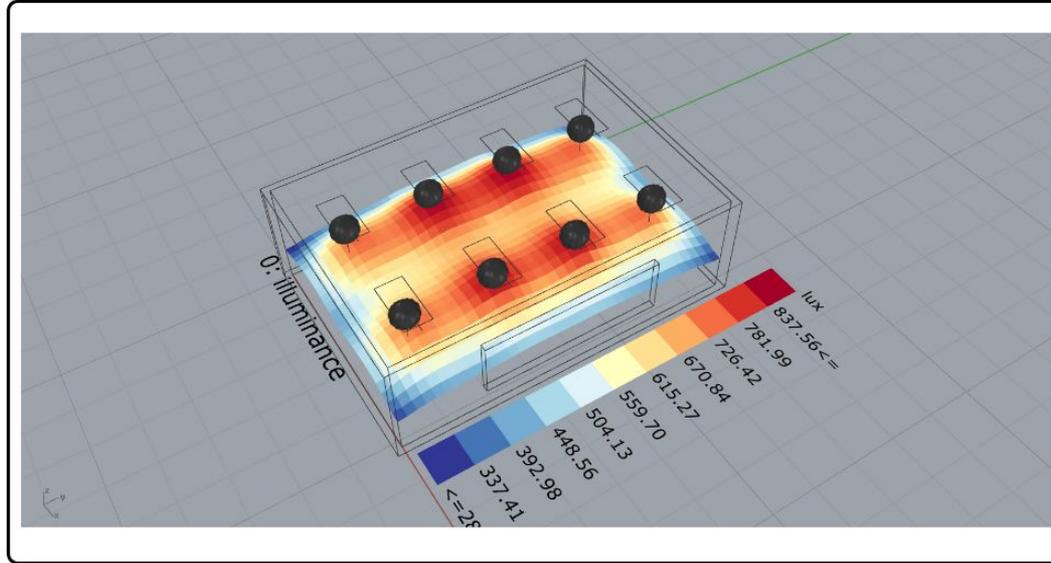
[Check out this example on Hydral](#)

Tags

IES Electric Lighting Grid-based calculations

[sharethis](#) 25 [Email](#) 0 [Share](#) 0 [Tweet](#) 0 [Share](#) 0 [Google +](#) 0

[ZoomToExtent](#)



[Download](#)

[#ES](#) [#Electric](#) [#Lighting](#) [#Grid-based](#) [#calculations](#)

Description

Indoor electric lighting calculations based on grid points

This file has been submitted by sariths

[Check out this example on Hydra!](#)

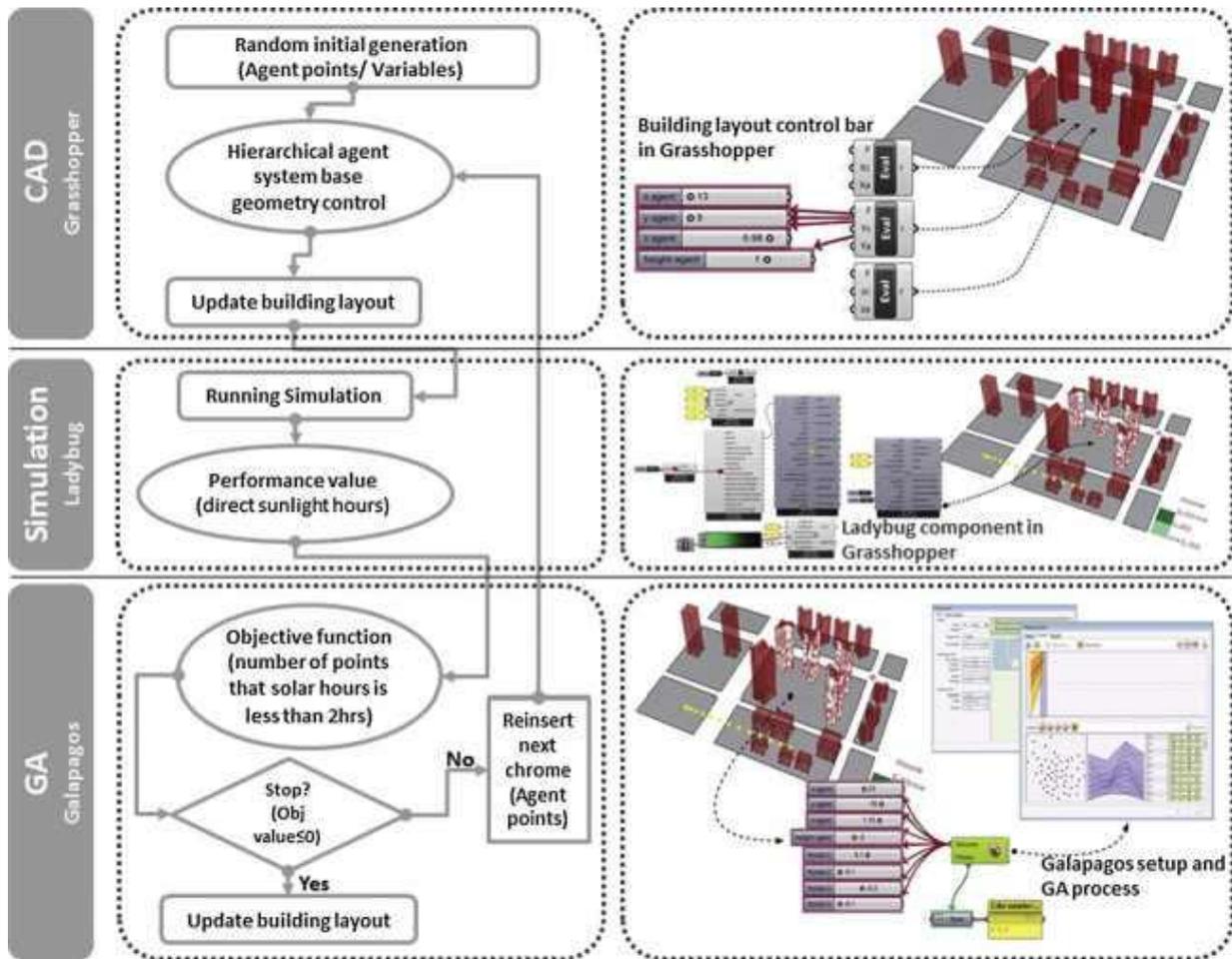
Tags

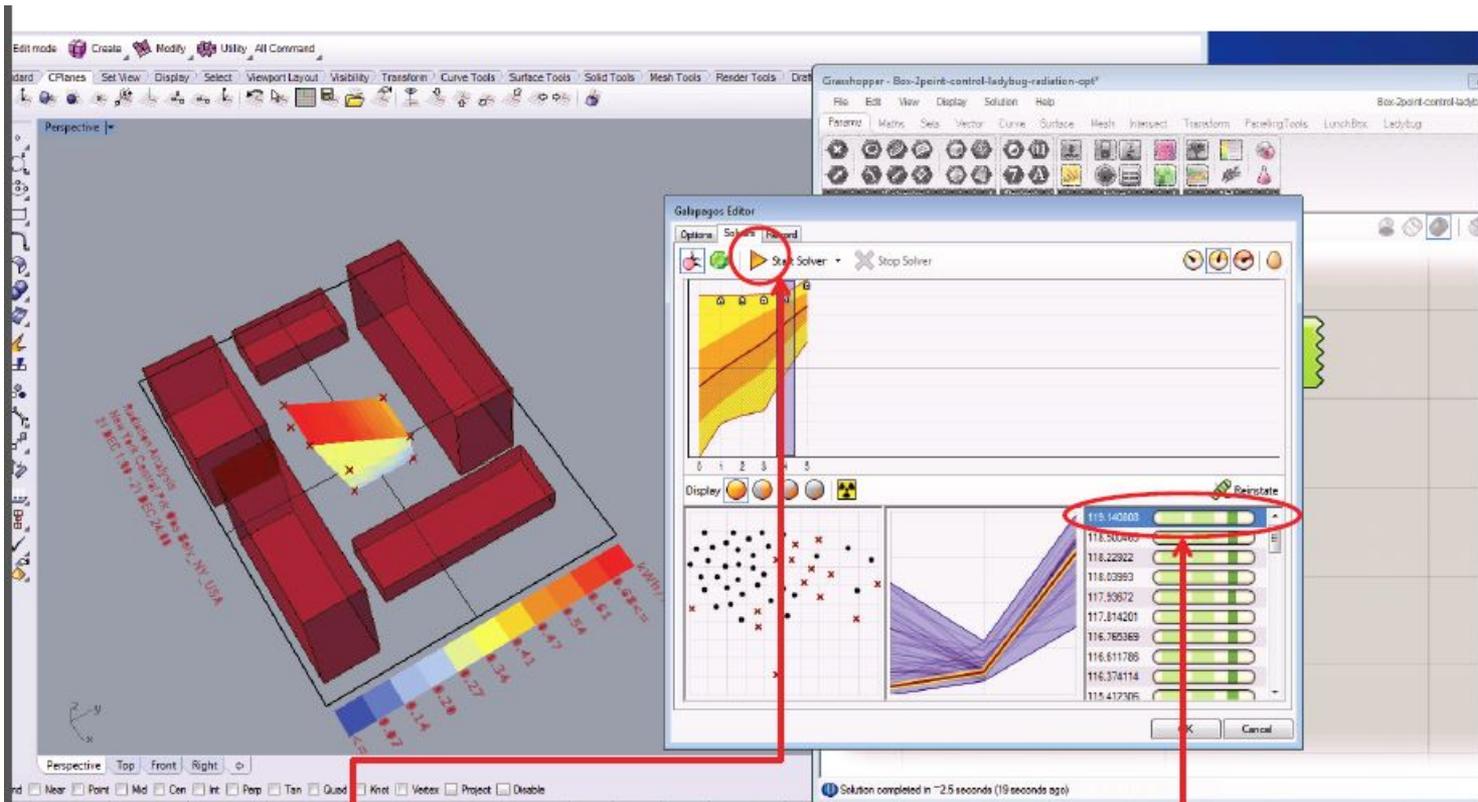
Share Code.

Share Knowledge.

Share Examples.

Share Stories!





Select start and it will continue optimization

Process stops when it reach the goal or max iterations

Make your own tool!



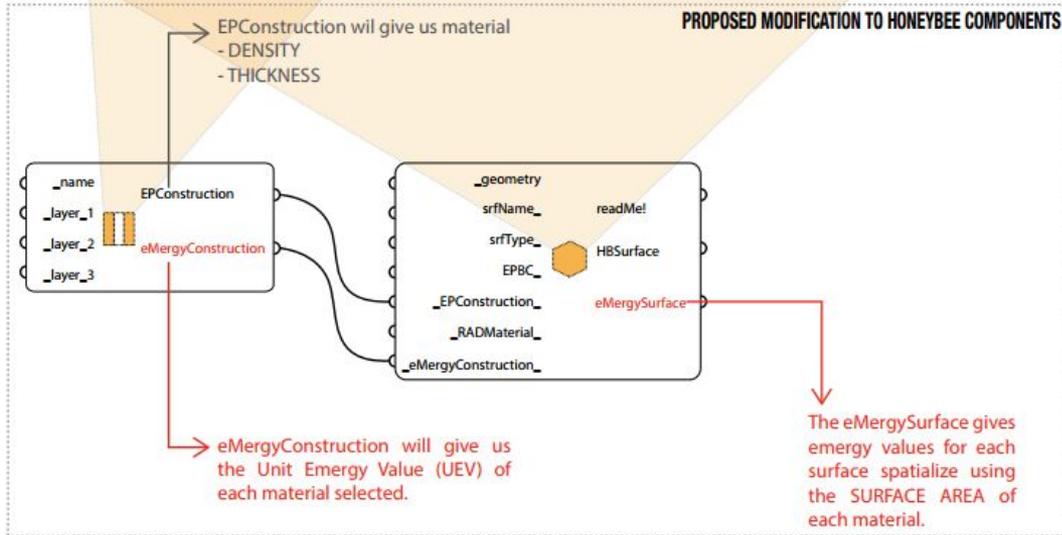
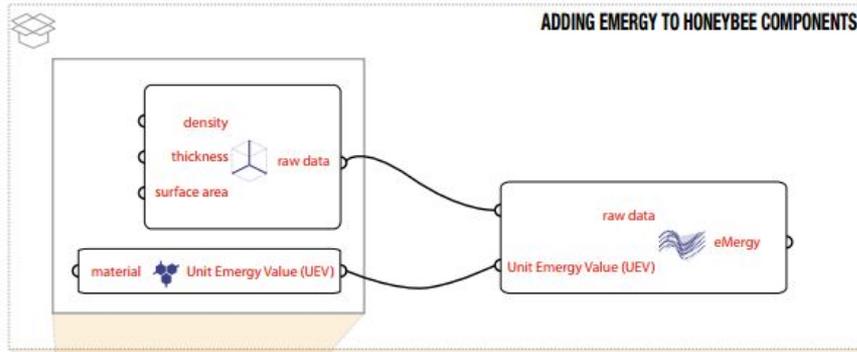
Clark's Crow



HoneyBee

Key:

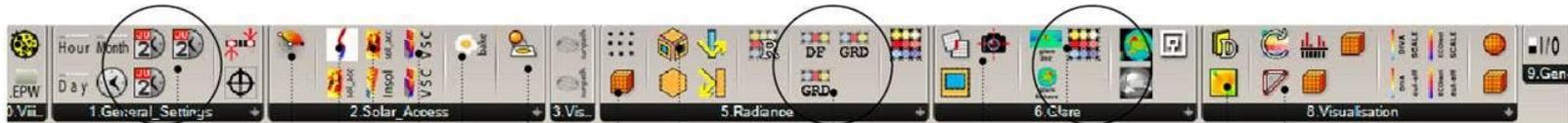
- existing component
- proposed component



Naomi Keena, Mohamed Aly Etman, Nancy Diniz, Alexandra Rempel, Anna Dyson

Center for Architecture Science and Ecology (CASE), Rensselaer Polytechnic Institute





Mar/Dec/Jun 21st
for the extremely
lazy people

Sun path that
gives vectors for
solar access simu-
lations. (there is a
second sun path
there that runs
for 21st June to
get maximum
hours for legend
high)

Solar access with
and without
minimum mark
cutoff (+ average
passing points for
regulations)

VSC only visual
and then VSC for
regulations were
only non-hori-
zontal surfaces
are calculated
and also overlap-
ing surfaces are
filtered. (this
takes ages to run
on more complex
models)

Simple bake
command
that gets the
layer name from
the "readme"
of analysis
period.

Very small and
full of bugs
component
that works like
Ecotect-shading
profile.

3D grid crea-
tor for radiance
(assign floor and
ceiling and
cube size)

Geometry filter
(when HBSrfs
crushes). **This is
very usefull** for
complex "dirty"
models. It filters
surfaces with very
small sides (line-
like surfaces) that
make radiance
crush

Just a simpli-
fied set of
radiance
simulations
for ease

Camera and
viewport
setup for
imagebased
simulations

Glare setup for fisheye
and cylinder (360°) views
&
Visualization of those

Average
period for .ill
files

Filters the mesh
bellow and above
target value and
creates a con-
tour line (with a
smooth factor to
make it visually
better)

Now this is a tricky
and messy one. It was
created on a project where
we wanted to test
reflection on a specific
target after solar rays
reflect from a adjacent
tower. It works with
assigning target and
context. It is very simi-
lar to the component
you already have but
for some reason i made
a new (can't remember
exactly why)



 *LadyBee*

Contains a cluster of Grasshopper components

 *This cluster occurs **once** in this document.*

A light gray grid background with two horizontal orange bars centered on it. The top bar is slightly longer than the bottom bar.

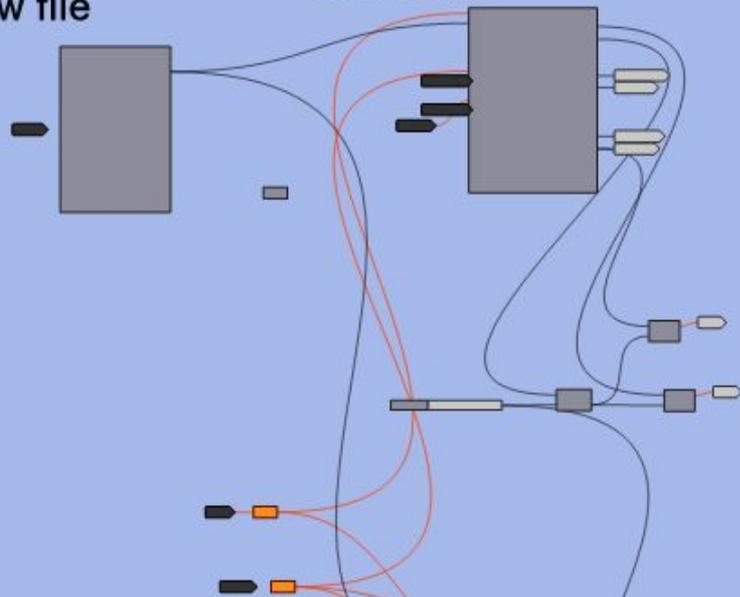
analysisPeriod_	sunSpheresMesh
sunPathScale	sunPathCrvs
centerPt	sunPathCenPts
_epwFile	sunPositions
North	sunPositions_10°
Timestep	Sun_Vectors_For_Analysis
	Total_hours_for_Legend

107ms (1%)

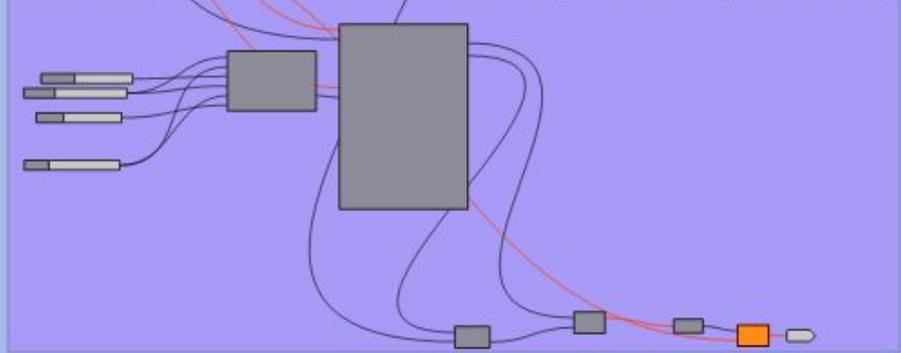
The other side of the coin!

Load .epw file

Sun path



Summer Solstice Max Hours (after angle cutoff)



Direct sunlight hour analysis



REPLY

**PAUL WINTOUR**

March 13, 2015 at 12:47 am

Hi

I'm wondering if there is a Dynamo node that can calculate the number of hours of direct sunlight a surface receives. Basically I want it to work like the 'Sunlight hours analysis' component found in Ladybug in Grasshopper. All I can currently see in Dynamo is daylighting and solar insolation. I would like to avoid constructing it from scratch if possible. Any thoughts?

**PAUL WINTOUR**

March 15, 2015 at 11:56 pm

anyone?



[Radiance-general] REVIT to Radiance pipeline

Shakespeare, Robert A. [shakespe at indiana.edu](mailto:shakespe@indiana.edu)

Thu Nov 12 06:35:11 PST 2015

- Previous message: [\[Radiance-general\] Luminaire modelling using Radiance](#)
 - Next message: [\[Radiance-general\] REVIT to Radiance pipeline](#)
 - **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#)
-

At the last Radiance Workshop, there was some discussion regarding converting REVIT models to Radiance, with materials and named surfaces parsed into readable .rad datasets. As REVIT has essentially become the architectural industry modeling tool, the pipeline to Radiance is an important one. It seems that the latest version of Sketchup has some challenges with the very helpful su2rad, written quite awhile ago. I heard that others were using some other pipelines, perhaps integrating obj2rad, etc.

I would appreciate your sharing conversion pipelines which interface with the MOST CURRENT releases of REVIT, SKETCHUP, etc.

Hopefully a generous and expert code person would consider writing a direct REVIT plugin for this purpose. There might be modest financial encouragement to fill this gap. I do realize that it may be short-lived as AutoDESK releases are moving targets, yet to engage the full Radiance tool kit for lighting design and analysis on REVIT generated models, will keep Radiance on the forefront and accessible to a broader audience.

Appreciatively,

Rob Shakespeare

[shakespe at indiana.edu](mailto:shakespe@indiana.edu)

----- next part -----

An HTML attachment was scrubbed...

URL: <<http://www.radiance-online.org/pipermail/radiance-general/attachments/20151112/abe112fe/attachment.html>>

- Previous message: [\[Radiance-general\] Luminaire modelling using Radiance](#)
 - Next message: [\[Radiance-general\] REVIT to Radiance pipeline](#)
 - **Messages sorted by:** [\[date \]](#) [\[thread \]](#) [\[subject \]](#) [\[author \]](#)
-

[More information about the Radiance-general mailing list](#)

Let the `bug fly free!

Visual Programming

DynamoBIM

Grasshopper3D

IronPython
Geometrical calculations

honeybeeX

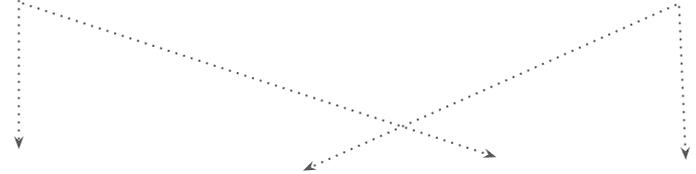
ladybugX

Python
Numerical calculation

honeybee

ladybug

Radiance, EnergyPlus



What does it really mean?

```
from honeybee.room import Room
from honeybee.radiance.material.glass import GlassMaterial
from honeybee.radiance.sky.certainIlluminance import SkyWithCertainIlluminanceLevel
from honeybee.radiance.recipe.gridbased import HBGridBasedAnalysisRecipe
```

0

```
room = Room(origin=(0, 0, 3.2), width=4.2, depth=6, height=3.2, rotationAngle=45) # create a test room
room.addFenestrationSurface(wallName='back', width=2, height=2, sillHeight=0.7) # add a window to the back wall
glass_60 = GlassMaterial.bySingleTransValue('tvis_0.6', 0.6) # add another window with custom material.
room.addFenestrationSurface('right', 4, 1.5, 1.2, radianceMaterial=glass_60) # This time to the right wall
```

1

```
# run a grid-based analysis for this room
sky = SkyWithCertainIlluminanceLevel(illuminanceValue=2000) # generate the sky
testPoints = room.generateTestPoints(gridSize=0.5, height=0.75) # generate grid of test points
rp = HBGridBasedAnalysisRecipe(sky=sky, pointGroups=(testPoints,), simulationType=0, hbObjects=(room,))
```

2

```
# write and run the analysis
rp.writeToFile(targetFolder=r'c:\ladybug', projectName='room')
rp.run(debug=False)
```

3

```
results = rp.results(flattenResults=True)
print 'Average illuminacen level in this room is {} lux.'.format(sum(results) / len(results))
```

4

```
from honeybee.room import Room
from honeybee.radiance.material.glass import GlassMaterial
from honeybee.radiance.sky.certainIlluminance import SkyWithCertainIlluminanceLevel
from honeybee.radiance.recipe.gridbased import HBGridBasedAnalysisRecipe

# create a test room
room = Room(origin=(0, 0, 3.2), width=4.2, depth=6, height=3.2, rotationAngle=45)

# add a window to the back wall
room.addFenestrationSurface(wallName='back', width=2, height=2, sillHeight=0.7)

# add another window with custom material. This time to the right wall
glass_60 = GlassMaterial.bySingleTransValue('tvis_0.6', 0.6)
room.addFenestrationSurface('right', 4, 1.5, 1.2, radianceMaterial=glass_60)
```

```
...  
# run a grid-based analysis for this room  
# generate the sky  
sky = SkyWithCertainIlluminanceLevel(illuminanceValue=2000)  
  
# generate grid of test points  
testPoints = room.generateTestPoints(gridSize=0.5, height=0.75)  
  
# put the recipe together  
rp = HBGridBasedAnalysisRecipe(sky=sky, pointGroups=(testPoints,),  
                               simulationType=0, hbObjects=(room,))  
  
# write and run the analysis  
rp.writeToFile(targetFolder=r'c:\ladybug', projectName='room')  
rp.run(debug=False)
```

...

```
results = rp.results(flattenResults=True)
print 'Average illuminance level in this room is {} lux.' \
      .format(sum(results) / len(results))
```

Number of total materials: 5

Number of total surfaces: 1

Files are written to: c:\ladybug\room\gridbased

C:\Users\Administrator\Documents\GitHub\hydrashare.github.io>c:

C:\Users\Administrator\Documents\GitHub\hydrashare.github.io>cd c:\ladybug\room\gridbased

c:\ladybug\room\gridbased>c:\radiance\bin\oconv -f c:\ladybug\room\gridbased\Uniform_CIE_2000.sky

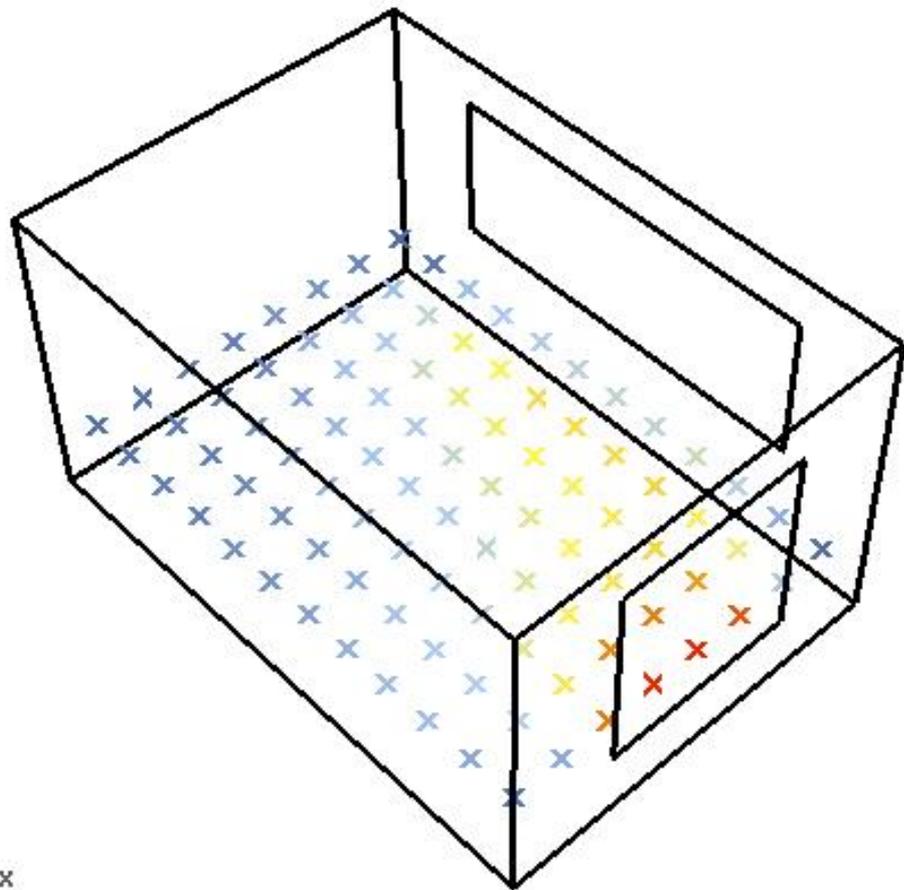
c:\ladybug\room\gridbased\room.mat c:\ladybug\room\gridbased\room.rad 1>room.oct

c:\ladybug\room\gridbased>c:\radiance\bin\rtrace -aa 0.25 -ab 2 -dj 0.0 -ad 512 -ss 0.0 -h -dc 0.25 -st 0.85 -lw 0.05 -as 128 -ar 16 -lr 4 -l -dt 0.5
-dr 0 -ds 0.5 -dp 64 -e error.txt c:\ladybug\room\gridbased\room.oct 0<c:\ladybug\room\gridbased\room.pts 1>room.res

Average illuminance level in this room is 158.901223225 lux.

3 + 4 / 4 Results

Perspective ▾

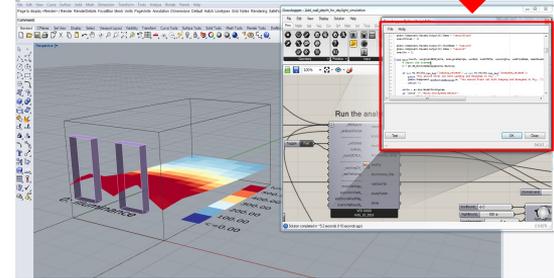
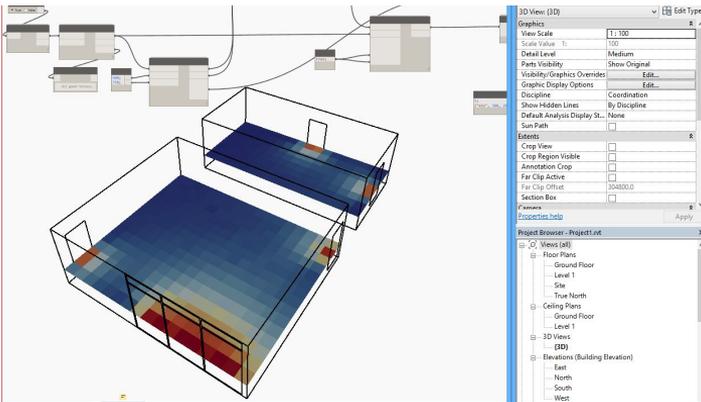


```

1 import sys
2 sys.path.extend(IN[0])
3
4 try:
5     from honeybee.radiance.recipe.gridbased import GridBasedAnalysisRecipe
6     from honeybee import wrapper
7 except ImportError:
8     raise ImportError("Failed to import Honeybee libraries.\n" + \
9                       "Set-up path to Honeybee libraries in honeybee_honeybee component")
10
11
12 _HBSky, _testPoints, ptsVectors_, _simType, _radParameters_ = IN[1:]
13
14 if _HBSky and _testPoints and ptsVectors_:
15     if _radParameters_:
16         _radParameters_ = _radParameters_.unwrap()
17
18     analysisRecipe = GridBasedAnalysisRecipe(
19         _HBSky.unwrap(), _testPoints, ptsVectors_, _simType, _radParameters_
20     )
21
22     OUT = wrapper.Wrapper(analysisRecipe)
  
```

```

File Help
1 import sys
2 from scriptcontext import sticky
3
4 if "hb_lib_path" in sticky:
5     sys.path.append(sticky["hb_lib_path"])
6
7 try:
8     from honeybee.radiance.recipe.gridbased import GridBasedAnalysisRecipe
9     from honeybee import wrapper
10 except ImportError:
11     raise ImportError("Failed to import Honeybee libraries.\n" + \
12                      "Set-up path to Honeybee libraries in honeybee_honeybee component")
13
14 if _HBSky and _testPoints and ptsVectors_:
15     if _radParameters_:
16         _radParameters_ = _radParameters_.unwrap()
17
18     analysisRecipe = GridBasedAnalysisRecipe(
19         _HBSky.unwrap(), _testPoints, ptsVectors_, _simulationType_, _radParameters_
20     )
21
22     analysisRecipe = wrapper.Wrapper(analysisRecipe)
  
```



File Path
Browse...
C:\EnergyPlus\8.3.0\WeatherData\USA_CA_San.Francisco.Intl.AP.724940_TMY3.epw

Import Location
_epwFile location

Calculate sunpath and sun vectors

north_	sunVectors
_location	sunAltitudes
hoursOfYear	sunAzimuths
centerPt	sunSpheres
scale	geometry
sunScale	centerPt
annualSunpath	sunPositions
	dacetimes

Calculate HOY

month	HOY
day	DOY
hour	datetime
minute	

Code Block
0; >

Code Block
7..20; >

Generate test points

Number of segments: 14; >

Distance from base: 0..1; >

Generate Test Points

testSurfaces	testPts
numOfSegments	ptsNormal
disFromSurface	

Calculate Sunlight hours

testPoints	sunlightHours
geometries	
sunVectors	

Create test geometries!

Plane.XY
Plane

Rectangle.ByWidth.Length
plane width length

Code Block
100; >

Surface.ByPatch
closedCurve Surface

Point.ByCoordinates
x y z Point

Vector.Yaxis
Vector

Code Block
45; >

Geometry.Rotate
geometry origin axis degrees

Surface.ByPatch
closedCurve Surface

Geometry.Scale
geometry xamount yamount zamount

Code Block
0..5; >

List.Create
item0 item1 list

Color surfaces based on results

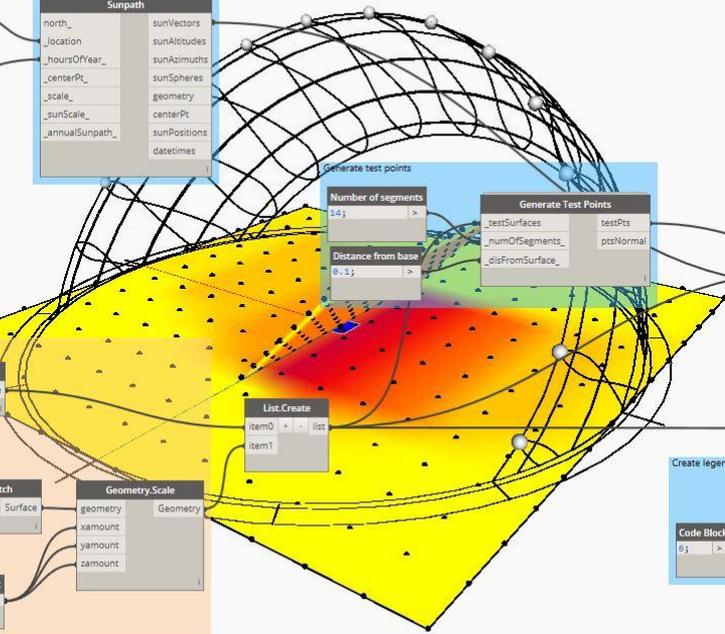
analysisSurfaces	Display
values	
legendPar	

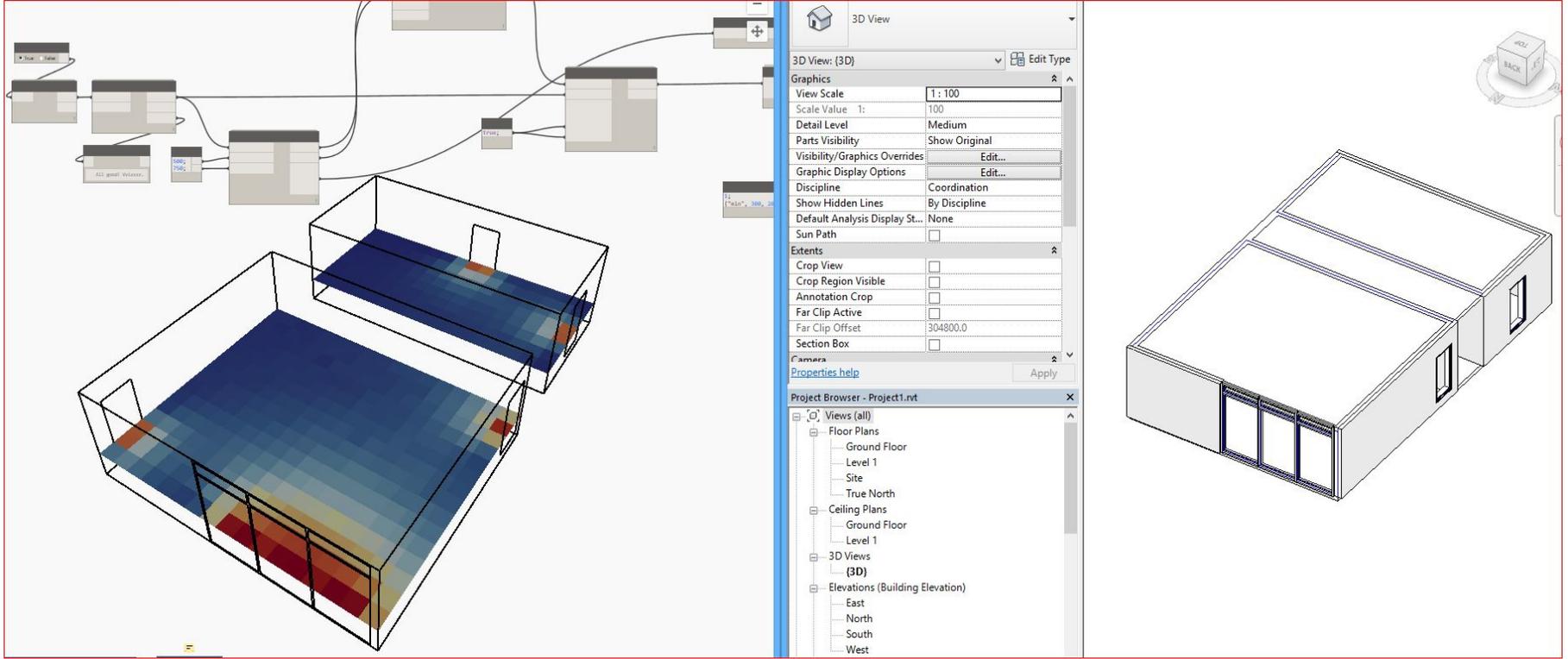
Create legend parameters

Code Block
0; >

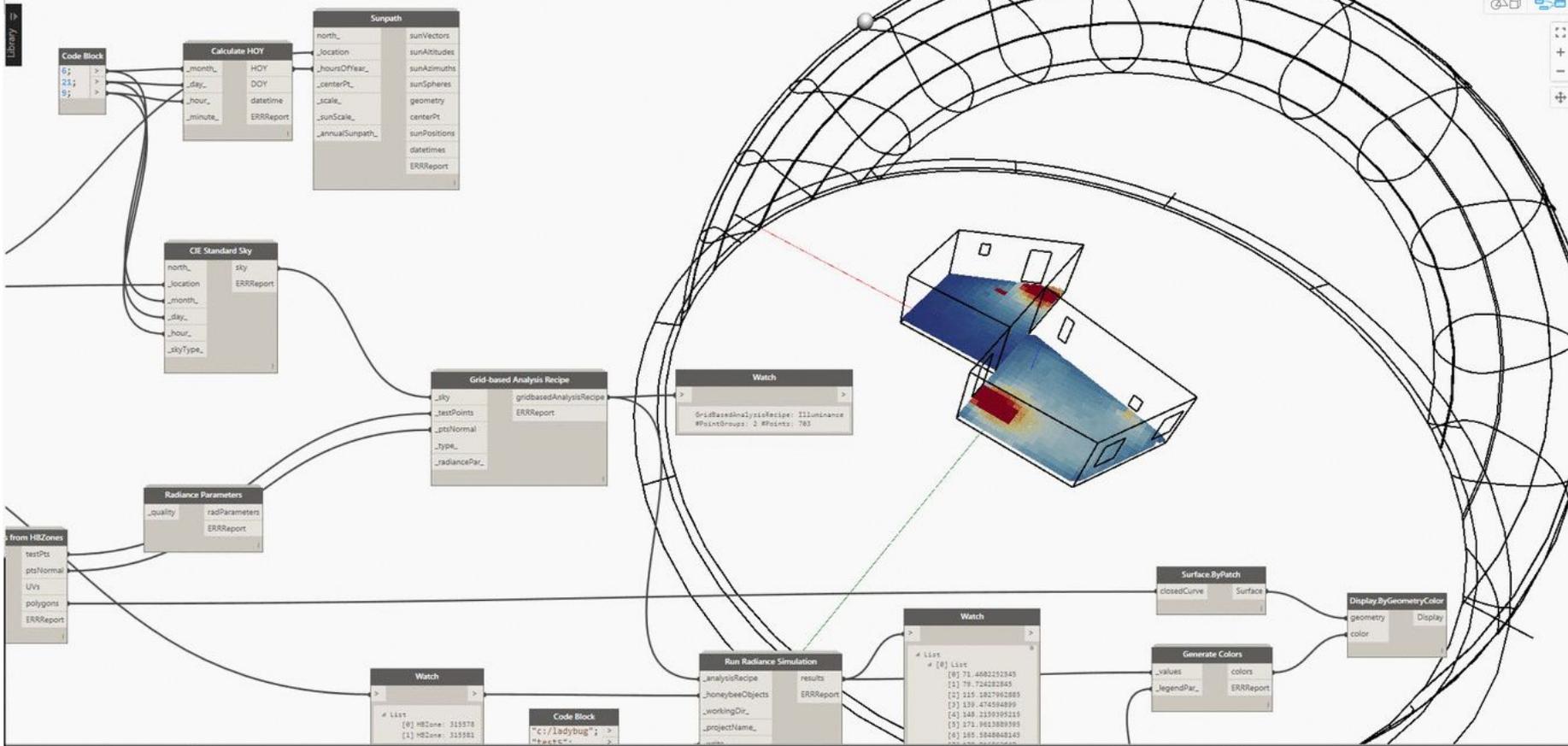
LB Color Range
index colors

Legend Parameters
type legendPar_ _domain_ _colors_





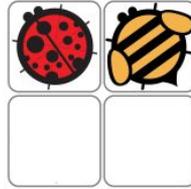
Library



Goals

Goals

Enable seamless Cross-OS compatibility



```
honeybee - [C:\Users\Smith\Projects\honeybee\ - scripts\ThreePhaseImage.py - PyCharm Community Edition 2016.2.1]
honeybee |> scripts |> ThreePhaseImage.py
ThreePhaseImage.py x
274 dot.matrixFile = 'Matrix
275 dot.matrixName = 'temp\3d.mat'
276 dot.matrixFile = @Matrix
277 dot.matrixName = @Matrix
278 dot.outputFile = @Matrix
279 dot.outputFile = @Matrix or 'temp\results.mat'
280 dot.execute()
281 return 'temp\results.txt'
282
283
284 if __name__ == '__main__':
285 phase = ['WT', 'Turb', '1', 'True', 'd1', 'False', 's', 'False]
286 description = ['matrixName.mat', 'matrixFile\matrix.mat', 'matrixFile\matrix.mat_37111.mat']
287
288 epwFiles = ['epw\USA_AK_Anchorage_1911.AP_702700_7003.epw',
289            'epw\USA_WL_London-Corbin-Wyome_Field_724243_7003.epw',
290            'epw\USA_WL_Indian-City_700_700.epw',
291            'epw\USA_WL_Charlotte-Douglas_1911.AP_723140_7003.epw',
292            'epw\USA_WL_Christian-Burns-Lakefront.AP_722540_7003.epw',
293            'epw\USA_WL_Philadelphia_1911.AP_704000_7003.epw']
294
295 for idx, matrix in enumerate(epwFiles):
296     resultFile = runPhaseCalculationType='angle', matrixFile=matrix,
297                 phase=CalculatePhase, epwFile=epwFiles[idx],
298                 numProcessors=4)
299
```



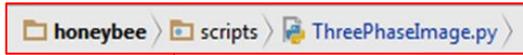
```
honeybee - [C:\work\Scripts\honeybee - scripts\ThreePhaseImage.py - PyCharm Community Edition 2016.1.4]
honeybee |> scripts |> ThreePhaseImage.py
ThreePhaseImage.py x
4. Start generate results
dot = @Matrix
dot.matrixFile = @Matrix
dot.matrixName = 'temp\3d.mat'
dot.outputFile = @Matrix
dot.outputFile = @Matrix or 'temp\results.mat'
dot.execute()
return 'temp\results.txt'

if __name__ == '__main__':
phase = ['WT', 'Turb', '1', 'True', 'd1', 'False', 's', 'False]
description = ['matrixName.mat', 'matrixFile\matrix.mat', 'matrixFile\matrix.mat_37111.mat']
epwFiles = ['epw\USA_AK_Anchorage_1911.AP_702700_7003.epw',
            'epw\USA_WL_London-Corbin-Wyome_Field_724243_7003.epw',
            'epw\USA_WL_Indian-City_700_700.epw',
            'epw\USA_WL_Charlotte-Douglas_1911.AP_723140_7003.epw',
            'epw\USA_WL_Christian-Burns-Lakefront.AP_722540_7003.epw',
            'epw\USA_WL_Philadelphia_1911.AP_704000_7003.epw']
for idx, matrix in enumerate(epwFiles):
    resultFile = runPhaseCalculationType='angle', matrixFile=matrix,
                phase=CalculatePhase, epwFile=epwFiles[idx],
                numProcessors=4)

```

A 3 Phase Method Workflow in Windows

1



```
honeybee > scripts > ThreePhaseImage.py
Project > ThreePhaseImage.py x
honeybee C:\Users\Sarith\Projects\ho
135
136
137
141
142
143
144
145
146
147
148
149
150
151
152
153
154
155
156
157
158
159
160
161
162
163
164
165
166
167
168
169
170
171
172
173
174
175
176
177
178
179
180
181
182
183
184
185
186
187
188
189
190
191
192
193
194
195
196
197
198
199
200
201
202
203
204
...
# Step5: Generate results
dct = Dctimestep()
dct.tmatrixFile = tMatrix
dct.vmatrixSpec = r'temp/*03d.hdr'
dct.dmatrixFile = dMatrix
dct.skyVectorFile = skyVector
dct.outputFileName = hdrResultsFileName or r'temp/results.hdr'
dct.execute()
return 'temp/results.txt'
if __name__ == '__main__':
    phases = {'v': True, 't': True, 'd': False, 's': True}
    tmatrices = ['xmls/clear.xml', 'xmls/diffuse50.xml', 'xmls/ExtVenetianBlind
    epwFiles = ['epws/USA_AK_Anchorage.Intl.AP.702730_TMY3.epw',
               'epws/USA_KY_London-Corbin-Magee.Field.724243_TMY3.epw',
               'epws/USA_MA_Boston-City.WS0_TMY.epw',
               'epws/USA_NC_Charlotte-Douglas.Intl.AP.723140_TMY3.epw',
               'epws/USA_OH_Cleveland-Burke.Lakefront.AP.725245_TMY3.epw',
               'epws/USA_PA_Philadelphia.Intl.AP.724080_TMY3.epw']
    for idx, matrix in enumerate(tmatrices):
        resultFile = run3Phase(calculationType='single', tmatrixFile=matrix,
                               phasesToCalculate=phases, epwFile=epwFiles[idx],
                               hdrResultsFileName=r'temp/results.hdr' & idx,
                               numProcessors=40)
```

2

```
numProcessors=40
```

3

rcontrib.exe	Sarith	13	4,512 K
conhost.exe	Sarith	00	2,492 K
cmd.exe *32	Sarith	00	1,576 K
taskeng.exe	Sarith	00	3,360 K
rfluxmtx.exe	Sarith	00	1,404 K

Image Name	User Name	CPU	Memory ...
rcontrib.exe	Sarith	13	4,512 K
conhost.exe	Sarith	00	2,492 K
cmd.exe *32	Sarith	00	1,576 K
taskeng.exe	Sarith	00	3,360 K
rfluxmtx.exe	Sarith	00	1,404 K
spWow64.exe	Sarith	00	3,524 K
cmd.exe	Sarith	00	2,008 K
CCC.exe	Sarith	00	26,168 K
MOM.exe	Sarith	00	7,468 K
IEMonitor.exe *32	Sarith	00	2,408 K
acrotray.exe *32	Sarith	00	2,576 K
Times4ft.exe *32	Sarith	00	7,940 K
OPENITEM.EXE	Sarith	00	608 K
jusched.exe *32	Sarith	00	2,608 K
EvernoteClipper.exe *32	Sarith	00	2,388 K
sw2_tray.exe	Sarith	00	7,648 K
wweb32.exe *32	Sarith	00	4,140 K
iTunesHelper.exe *32	Sarith	00	4,112 K
netsession_win.exe *32	Sarith	00	8,812 K
CodeMeterCC.exe *32	Sarith	00	4,056 K
netsession_win.exe *32	Sarith	00	3,632 K
SpotifyWebHelper.exe *32	Sarith	00	2,432 K
AdAppMgr.exe *32	Sarith	00	20,168 K
explorer.exe	Sarith	00	46,160 K
PanGPA.exe	Sarith	00	5,880 K
dum.exe	Sarith	00	23,756 K
ccSvcHst.exe *32	Sarith	00	1,600 K
FDISpPos.exe	Sarith	00	1,756 K
taskhost.exe	Sarith	00	6,140 K
SnagPrv.exe *32	Sarith	00	3,844 K
MCTDUtil.exe	Sarith	00	1,916 K

System Config: Remote Cluster, Xeon, 40-Core

Behind the scenes: genskyvec(.pl)



```
.genskv = Genskyvec()  
.genskv.inputSkyFile = r'temp/sky.rad'  
.genskv.outputFile = r'temp/sky.vec'  
.genskv.skySubdivision = 4  
.genskv.execute()
```



```
"C:\Program Files\OpenStudio 1.11.0\strawberry-perl-5.16.2.1-32bit-portable-reduced\perl\bin\perl.exe"  
"C:\Program Files\OpenStudio 1.11.0\share\openStudio\Radiance\bin\genskyvec.pl" -m 4 < temp\sky.rad >  
temp\sky.vec
```



```
/gpfs/home/sxs1106/work/Radiance/bin/genskyvec -m 4 < temp/sky.rad > temp/sky.vec
```



PERL Interpreter Genskyvec Inputs

Goals

Enable seamless Cross-OS compatibility

Simplify syntax with abstractions

Human readable inputs for infrequently used commands

```
gendayParam = GendaymtxParameters()
gendayParam.skyDensity = 4

genday = Gendaymtx(weaFile=r'temp/te
genday.gendaymtxParameters = gendayPa
genday.execute()

skyVector = r'temp/day.smx'
else:
    genskPar = GenskyParameters()
    genskPar.c
        f cloudySky
    gensk.
        f __class__
    gensk.
        f uniformCloudySky
    gensk.
        m addRadianceBoolFlag(self, n
    gensk.
        m addRadianceNumber(self, nam
```

```
genday.gendaymtxParameters = gen
genday.execute()

skyVector = r'temp/day.smx'
else:
    genskPar = GenskyParameters()
    genskPar.cloudySky
        gensk = Gensky()
        gensk.genskyParameters = genskPa
        gensk.monthDayHour = (11,11,11)
        gensk.outputFile = 'temp/sky.rad'
        gensk.execute()

    genskv = Genskyvec()
    genskv.inputSkyFile = r'temp/sky
    genskv.outputFile = r'temp/sky.v
```

Documentation cloudySky

Instance attribute cloudySky of class GenskyParameters

[-c] A boolean value to generate cloudy sky

Look-up class documentation while coding workflows.

Code completion on compatible IDEs.
(API remembers and prompts for inputs)

A self-documenting API

Index

Sub-modules

- `honeybee.radiance.command`
- `honeybee.radiance.datatype`
- `honeybee.radiance.filemanager`
- `honeybee.radiance.geometry`
- `honeybee.radiance.material`
- `honeybee.radiance.parameters`
- `honeybee.radiance.postprocess`
- `honeybee.radiance.properties`
- `honeybee.radiance.recipe`
- `honeybee.radiance.runmanager`
- `honeybee.radiance.view`

`honeybee.radiance` module

Honeybee Radiance libraries.

[SHOW SOURCE ↗](#)

Sub-modules

`honeybee.radiance.command`

`honeybee.radiance.datatype`

Descriptors, factory classes etc for the Radiance library.

`honeybee.radiance.filemanager`

A collection of auxiliary funtions for working with radiance files and objects.

`honeybee.radiance.geometry`

A collection of methods for writing Radiance geometry file.

`honeybee.radiance.material`

`honeybee.radiance.parameters`

`honeybee.radiance.postprocess`

`honeybee.radiance.properties`

<http://ladybug-analysis-tools.github.io/honeybee/doc/radiance/index.html>

Goals

Enable seamless Cross-OS compatibility

Simplify syntax with abstractions

Type and Error Checking

Gensky
Command

```
genskPar = GenskyParameters()  
genskPar.turbidity = 0.01  
gensk = Gensky()  
gensk.genskyParameters = genskPar  
gensk.monthDayHour = (11, 11, 11)  
gensk.outputFile = 'temp/sky.rad'  
gensk.execute()
```

[Gensky Manpage](https://radsite.lbl.gov/radiance/man_html/gensky.1.html)

radsite.lbl.gov/radiance/man_html/gensky.1.html

“Values less than 1.0 are physically impossible.”

Error
Message

```
Traceback (most recent call last):  
... File "C:/Users/Sarith/Projects/honeybee/scripts/ThreePhaseIllum.py", line 153, in <module>  
... phasesToCalculate=phases,epwFile=epwFiles[1])  
... File "C:/Users/Sarith/Projects/honeybee/scripts/ThreePhaseIllum.py", line 113, in run3phas  
... genskPar.turbidity = 0.01  
... File "C:/Users/Sarith/Projects/honeybee/honeybee/radiance/parameters/_frozen.py", line 53,  
... object.__setattr__(self, key, value)  
... File "C:/Users/Sarith/Projects/honeybee/honeybee/radiance/datatype.py", line 368, in __set  
... raise ValueError(msg)  
ValueError: The specified input for t (turbidity) is 0.01. This is below the valid range. Th
```

```
ValueError: The specified input for t (turbidity) is 0.01. This is below the valid range.
```

Goals

Enable seamless Cross-OS compatibility

Simplify syntax with abstractions

Type and Error Checking

Leverage Python to reduce effort



```
rfluxPara = RfluxmtxParameters()  
rfluxPara.I = True  
rfluxPara.aa = 0.1  
rfluxPara.ad = 4096  
rfluxPara.ab = 12  
rfluxPara.lw = 0.0000001  
rflux = Rfluxmtx()  
skyFile = rflux.defaultSkyGround(r'temp/rfluxSky.rad', skyType='r4')  
rflux.receiverFile = skyFile  
rflux.sender = '-'  
rflux.rfluxmtxParameters = rfluxPara  
rflux.radFiles = [r"room.mat", r"room.rad"]  
rflux.pointsFile = r"indoor_points.pts"  
rflux.outputMatrix = r"temp/dayCoeff.dc"  
rflux.execute()
```

Radiance

```
rfluxmtx" -y 6 -aa 0.1 -ab 12 -ad 4096 -lw 1e-07 -I - temp\rfluxSky.rad room.mat room.rad < indoor_points.pts > temp\dayCoeff.dc
```

Automate predictable tasks: Create a sky definition

```
rfluxPara = RfluxmtxParameters()  
rfluxPara.I = True  
rfluxPara.aa = 0.1  
rfluxPara.ad = 4096  
rfluxPara.ab = 12  
rfluxPara.lw = 0.0000001  
rflux = Rfluxmtx()  
skyFile = rflux.defaultSkyGround(r'temp/rfluxSky.rad', skyType='r4')  
rflux.receiverFile = skyFile  
rflux.sender = '-'  
rflux.rfluxmtxParameters = rfluxPara  
rflux.radFiles = [r"room.mat", r"room.rad"]  
rflux.pointsFile = r"indoor_points.pts"  
rflux.outputMatrix = r"temp/dayCoeff.dc"  
rflux.execute()
```

1. Create sky definition

2. Add rfluxmtx comments

```
#@rfluxmtx h=u u=Y  
void glow ground_glow  
0  
0  
4 1 1 1 0  
  
ground_glow source ground  
0  
0  
4 0 0 -1 180  
  
#@rfluxmtx h=r4 u=Y  
void glow sky_glow
```

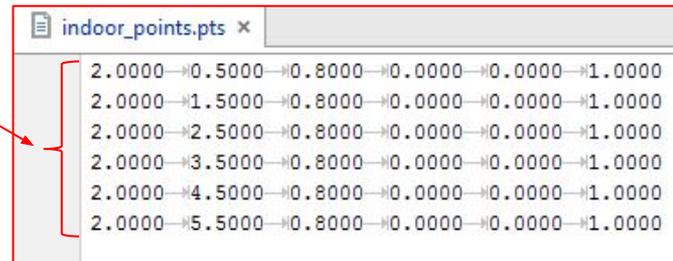
```
rfluxmtx" -y 6 -aa 0.1 -ab 12 -ad 4096 -lw 1e-07 -I - temp\rfluxSky.rad room.mat room.rad < indoor_points.pts > temp\dayCoeff.dc
```

3. Assign sky as "sender"

Automate predictable tasks: Count a points-file

```
rfluxPara = RfluxmtxParameters()  
rfluxPara.I = True  
rfluxPara.aa = 0.1  
rfluxPara.ad = 4096  
rfluxPara.ab = 12  
rfluxPara.lw = 0.0000001  
rflux = Rfluxmtx()  
skyFile = rflux.defaultSkyGround(r'temp/rfluxSky.rad', skyType='r4')  
rflux.receiverFile = skyFile  
rflux.sender = '-'  
rflux.rfluxmtxParameters = rfluxPara  
rflux.radFiles = [r"room.mat", r"room.rad"]  
rflux.pointsFile = r"indoor_points.pts"  
rflux.outputMatrix = r"temp/dayCoeff.dc"  
rflux.execute()
```

1.Count the number of points



```
rfluxmtx" -y 6 -aa 0.1 -ab 12 -ad 4096 -lw 1e-07 -I - temp\rfluxSky.rad room.mat room.rad < indoor_points.pts > temp\dayCoeff.dc
```

2. Assign the number of points in the command

Make the workflow more flexible

```
rfluxPara = RfluxmtxParameters()  
rfluxPara.I = True  
rfluxPara.aa = 0.1  
rfluxPara.ad = 4096  
rfluxPara.ab = 12  
rfluxPara.lw = 0.0000001  
rflux = Rfluxmtx()  
skyFile = rflux.defaultSkyGround(r'temp/rfluxSky.rad', skyType='r4')  
rflux.receiverFile = skyFile  
rflux.sender = '-'  
rflux.rfluxmtxParameters = rfluxPara  
rflux.radFiles = [r"room.mat", r"room.rad"]  
rflux.pointsFile = r"indoor_points.pts"  
rflux.outputMatrix = r"temp/dayCoeff.dc"  
rflux.execute()
```

Inputs can be in any order.

```
rfluxmtx" -y 6 -aa 0.1 -ab 12 -ad 4096 -lw 1e-07 -I - temp\rfluxSky.rad room.mat room.rad < indoor_points.pts > temp\dayCoeff.dc
```

The ordering of inputs needs to follow a certain order.

```
rfluxmtx [ -v ] [ rcontrib options ] { sender.rad | - } receivers.rad [ -i system.oct ] [ system.rad .. ]
```

Goals

Enable seamless Cross-OS compatibility

Simplify syntax with abstractions

Type and Error Checking

Leverage Python to reduce effort

Interoperability between simulation platforms



```
from honeybee.energypplus import filemanager, geometryrules
from honeybee.hbsurface import HBSurface
from honeybee.hbfensurface import HBFenSurface
from honeybee.hbshadesurface import HBShadingSurface
from honeybee.hbzone import HBZone
import os

def idfToRadString(idfFilePath):
    """Convert an idf file geometry to a radiance definition.

    Radiance materials are assigned based on surface types and not from
    EnergyPlus materials or construction. You can create and map your
    own Radiance materials by adding a few number of lines to the code.
    """
    objects = filemanager.getEnergyPlusObjectsFromFile(idfFilePath)

    # if the geometry rules is relative then all the points should be added
    # to X, Y, Z of zone origin
    geoRules = geometryrules.GlobalGeometryRules(
        *objects['globalgeometryrules'].values()[0][1:4]
    )

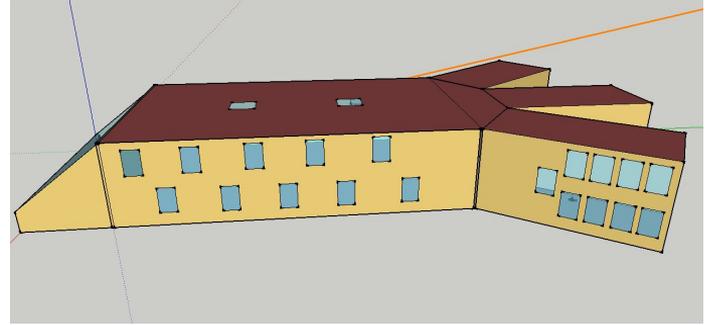
    hbObjects = {'zone': {}, 'buildingsurface': {}, 'shading': {}}

    # create zones
    for zoneName, zoneData in objects['zone'].iteritems():
        # create a HBZone
        zone = HBZone.fromEPString(",".join(zoneData), geometryRules=geoRules)
        hbObjects['zone'][zoneName] = zone

    # create surfaces
    for surfaceName, surfaceData in objects['buildingsurface:detailed'].iteritems():
        surface = HBSurface.fromEPString(",".join(surfaceData))
        surface.parent = hbObjects['zone'][surfaceData[4]]
        hbObjects['buildingsurface'][surfaceName] = surface
```



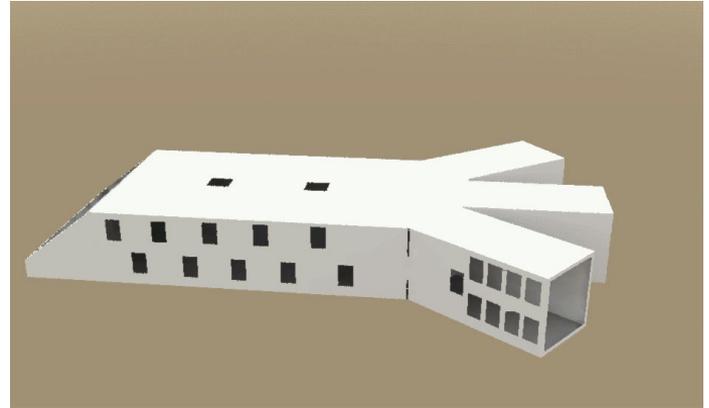
IDF



Radiance



HDR



Goals

Enable seamless Cross-OS compatibility

Simplify syntax with abstractions

Type and Error Checking

Leverage Python to reduce effort

Interoperability between simulation platforms

Apply Object Oriented Programming (OOP) where required

Radiance syntax alludes to OOP patterns

NAME

rcontrib - compute contribution coefficients in a RADIANCE scene

SYNOPSIS

```
rcontrib [ -n nprocs ][ -V ][ -c count ][ -fo | -r ][ -e expr ][ -f source ][ -o ospec ][ -p p1=V1,p2=V2 ]  
[ -b binv ][ -bn nbins ] { -m mod | -M file } [ SEVAR ] [ @file ] [ rtrace options ] octree  
rcontrib [ options ] -defaults
```

NAME

rfluxmtx - compute flux transfer matrix(es) for RADIANCE scene

SYNOPSIS

```
rfluxmtx [ -v ] [ rcontrib options ] { sender.rad | - } receivers.rad [ -i system.oct ] [ system.rad .. ]
```

DESCRIPTION

NAME

vwrays - compute rays for a given picture or view

SYNOPSIS

```
vwrays [ -i -u -f{a|f|d} -c rept | -d ] { view opts .. | picture [zbuf] }
```

DESCRIPTION

Vwrays takes a picture or view specification and computes the ray origin and direction corresponding to

Honeybee “wraps” Radiance syntax through OOP

Base Class

Implements **rtrace** options ab, ad, as etc.

```
@frozen
class GridBasedParameters(AdvancedRadianceParameters):
    """Radiance Parameters for grid based analysis..."""

    def __init__(self, quality=None):
```

Subclass

Implements **rcontrib**.

```
@frozen
class RcontribParameters(GridBasedParameters):
    """Radiance Parameters for rcontrib command including rtrace parameters.

    def __init__(self, modFile=None):
```

Subclass

Implements **rfluxmtx**.

```
@frozen
class RfluxmtxParameters(GridBasedParameters):

    def __init__(self, sender=None, receiver=None, octree=None, systemFile:
        """Init parameters."""
```

Goals

Enable seamless Cross-OS compatibility

Simplify syntax with abstractions

Type and Error Checking

Leverage Python to reduce effort

Interoperability between simulation platforms

Apply Object Oriented Programming (OOP) where required

Recipes !

Use the API to create “recipes” for common workflows

ThreePhaseIllum.py

```
def run3phase(phasesToCalculate={'v':True,'t':True,'d':True,'s':True},
             calculationType='annual',epwFile=None,tmatrixFile=None):
    # Step 1: Create the view matrix.
    rfluxPara = RfluxmtxParameters()
    rfluxPara.I = True
    rfluxPara.aa = 0.1
    rfluxPara.ab = 10
    rfluxPara.ad = 65536
    rfluxPara.lw = 1E-5
    # Step 1.1 Invert glazing surface with xform so that it faces inwards
    xfrPara = XformParameters()
    xfrPara.invertSurfaces = True
    xfr = Xform()
    xfr.xformParameters = xfrPara
    xfr.radFile = 'glazing.rad'
    xfr.outputFile = 'glazingI.rad'
    xfr.execute()
    rflux = Rfluxmtx()
    rflux.sender = '-'
    # Klems full basis sampling and the window faces +Y
    recCtrlPar = rflux.ControlParameters(hemiType='hf',hemiUpDirection='+Z')
    rflux.receiverFile = rflux.addControlParameters('glazingI.rad',
                                                    {'Exterior_Window':recCtrlPar})
    rflux.rfluxmtxParameters = rfluxPara
    rflux.pointsFile = 'indoor_points.pts'
```

ThreePhaseImage.py

```
def run3phase(phasesToCalculate={'v':True,'t':True,'d':True,'s':True},
             calculationType='annual',epwFile=None,tmatrixFile=None):
    # Step 1: Create the view matrix.
    rfluxPara = RfluxmtxParameters()
    rfluxPara.I = True
    rfluxPara.aa = 0.1
    rfluxPara.ab = 10
    rfluxPara.ad = 65536
    rfluxPara.lw = 1E-5
    # Step 1.1 Invert glazing surface with xform so that it faces inwards
    xfrPara = XformParameters()
    xfrPara.invertSurfaces = True
    xfr = Xform()
    xfr.xformParameters = xfrPara
    xfr.radFile = 'glazing.rad'
    xfr.outputFile = 'glazingI.rad'
    xfr.execute()
    rflux = Rfluxmtx()
    rflux.sender = '-'
    # Klems full basis sampling and the window faces +Y
    recCtrlPar = rflux.ControlParameters(hemiType='hf',hemiUpDirection='+Z')
    rflux.receiverFile = rflux.addControlParameters('glazingI.rad',
                                                    {'Exterior_Window':recCtrlPar})
    rflux.rfluxmtxParameters = rfluxPara
    rflux.pointsFile = 'indoor_points.pts'
```

DayCoeffImage.py

```
def run3phase(phasesToCalculate={'v':True,'t':True,'d':True,'s':True},
             calculationType='annual',epwFile=None,tmatrixFile=None):
    # Step 1: Create the view matrix.
    rfluxPara = RfluxmtxParameters()
    rfluxPara.I = True
    rfluxPara.aa = 0.1
    rfluxPara.ab = 10
    rfluxPara.ad = 65536
    rfluxPara.lw = 1E-5
    # Step 1.1 Invert glazing surface with xform so that it faces inwards
    xfrPara = XformParameters()
    xfrPara.invertSurfaces = True
    xfr = Xform()
    xfr.xformParameters = xfrPara
    xfr.radFile = 'glazing.rad'
    xfr.outputFile = 'glazingI.rad'
    xfr.execute()
    rflux = Rfluxmtx()
    rflux.sender = '-'
    # Klems full basis sampling and the window faces +Y
    recCtrlPar = rflux.ControlParameters(hemiType='hf',hemiUpDirection='+Z')
    rflux.receiverFile = rflux.addControlParameters('glazingI.rad',
                                                    {'Exterior_Window':recCtrlPar})
    rflux.rfluxmtxParameters = rfluxPara
    rflux.pointsFile = 'indoor_points.pts'
```

DayCoeffImage.py

```
def run3phase(phasesToCalculate={'v':True,'t':True,'d':True,'s':True},
             calculationType='annual',epwFile=None,tmatrixFile=None):
    # Step 1: Create the view matrix.
    rfluxPara = RfluxmtxParameters()
    rfluxPara.I = True
    rfluxPara.aa = 0.1
    rfluxPara.ab = 10
    rfluxPara.ad = 65536
    rfluxPara.lw = 1E-5
    # Step 1.1 Invert glazing surface with xform so that it faces inwards
    xfrPara = XformParameters()
    xfrPara.invertSurfaces = True
    xfr = Xform()
    xfr.xformParameters = xfrPara
    xfr.radFile = 'glazing.rad'
    xfr.outputFile = 'glazingI.rad'
    xfr.execute()
    rflux = Rfluxmtx()
    rflux.sender = '-'
    # Klems full basis sampling and the window faces +Y
    recCtrlPar = rflux.ControlParameters(hemiType='hf',hemiUpDirection='+Z')
    rflux.receiverFile = rflux.addControlParameters('glazingI.rad',
                                                    {'Exterior_Window':recCtrlPar})
    rflux.rfluxmtxParameters = rfluxPara
    rflux.pointsFile = 'indoor_points.pts'
```

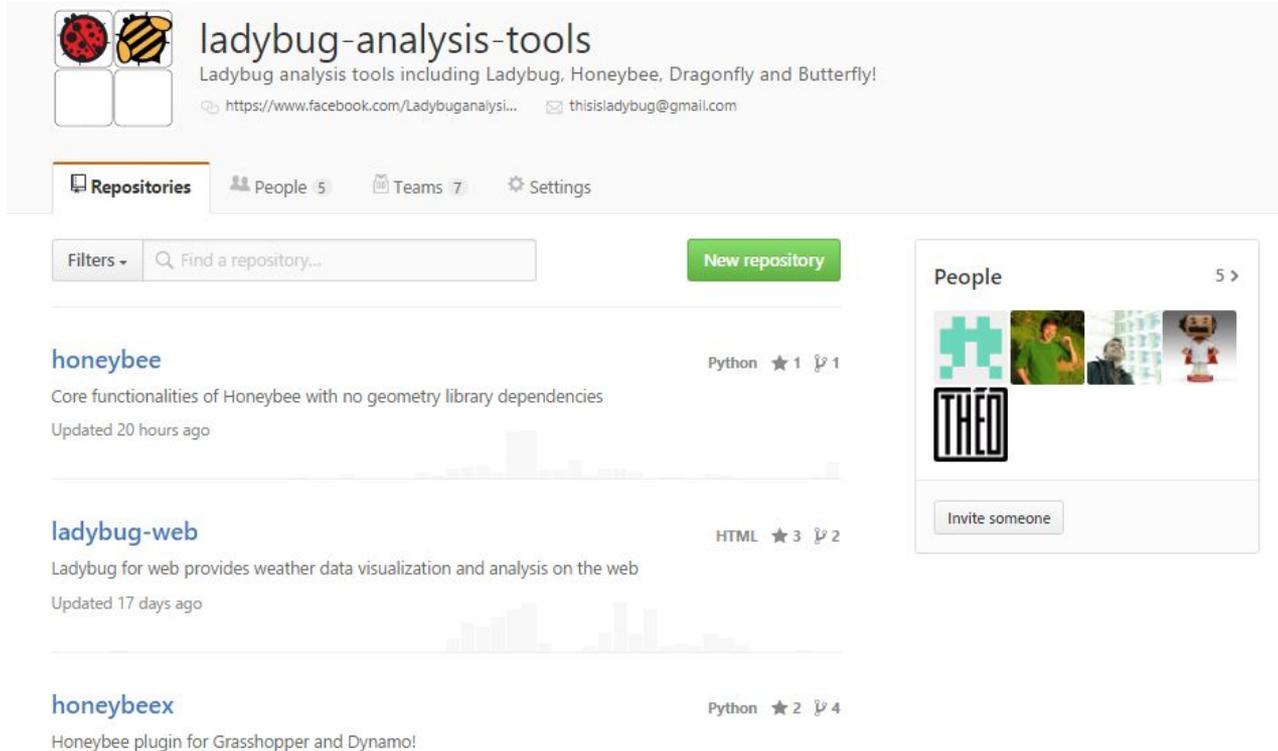
PhotonMapImage.py

FivePhaseImage.py

```
def run3phase(phasesToCalculate={'v':True,'t':True,'d':True,'s':True},
             calculationType='annual',epwFile=None,tmatrixFile=None):
    # Step 1: Create the view matrix.
    rfluxPara = RfluxmtxParameters()
    rfluxPara.I = True
    rfluxPara.aa = 0.1
    rfluxPara.ab = 10
    rfluxPara.ad = 65536
    rfluxPara.lw = 1E-5
    # Step 1.1 Invert glazing surface with xform so that it faces inwards
    xfrPara = XformParameters()
    xfrPara.invertSurfaces = True
    xfr = Xform()
    xfr.xformParameters = xfrPara
    xfr.radFile = 'glazing.rad'
    xfr.outputFile = 'glazingI.rad'
    xfr.execute()
    rflux = Rfluxmtx()
    rflux.sender = '-'
    # Klems full basis sampling and the window faces +Y
    recCtrlPar = rflux.ControlParameters(hemiType='hf',hemiUpDirection='+Z')
    rflux.receiverFile = rflux.addControlParameters('glazingI.rad',
                                                    {'Exterior_Window':recCtrlPar})
    rflux.rfluxmtxParameters = rfluxPara
    rflux.pointsFile = 'indoor_points.pts'
```


Live Demo

How to get involved?



The screenshot shows the GitHub profile for 'ladybug-analysis-tools'. The profile header includes the repository name, a description 'Ladybug analysis tools including Ladybug, Honeybee, Dragonfly and Butterfly!', and contact information for Facebook and email. Below the header are navigation tabs for 'Repositories', 'People 5', 'Teams 7', and 'Settings'. A search bar with 'Find a repository...' and a 'New repository' button are visible. The repository list shows three items: 'honeybee' (Python, 1 star, 1 fork), 'ladybug-web' (HTML, 3 stars, 2 forks), and 'honeybeex' (Python, 2 stars, 4 forks). A 'People' sidebar on the right shows 5 members and an 'Invite someone' button.

ladybug-analysis-tools
Ladybug analysis tools including Ladybug, Honeybee, Dragonfly and Butterfly!
🌐 <https://www.facebook.com/Ladybuganalysis...> ✉ thisisladybug@gmail.com

Repositories People 5 Teams 7 Settings

Filters - Find a repository... **New repository**

honeybee Python ★ 1 🍴 1
Core functionalities of Honeybee with no geometry library dependencies
Updated 20 hours ago

ladybug-web HTML ★ 3 🍴 2
Ladybug for web provides weather data visualization and analysis on the web
Updated 17 days ago

honeybeex Python ★ 2 🍴 4
Honeybee plugin for Grasshopper and Dynamo!

People 5 >
Invite someone

<https://github.com/ladybug-analysis-tools>

Thank You!