# Calculating and Applying BSDFs in *Radiance*

Greg Ward
Anyhere Software

# Goals

- Support WINDOW 6 BSDF output (XML)
- Create WINDOW 6 BSDFs from geometric models using freeware (i.e., *Radiance*)
- Render accurate images with direct shading using **mkillum**
- Perform efficient annual simulations
- Support higher-resolution BSDF data

# Status

- WINDOW 3 BSDF format still subject to revision
- Support for reading and creating BSDFs is currently limited to transmittance data
- **mkillum** has been tested and refined to produce reasonable accuracy for diffusing systems
- New **genBSDF** tool for creating BSDF files from geometric (*Radiance* or MGF) descriptions
- Annual simulation for the **daring** with **rtcontrib**

# What Is a BSDF?

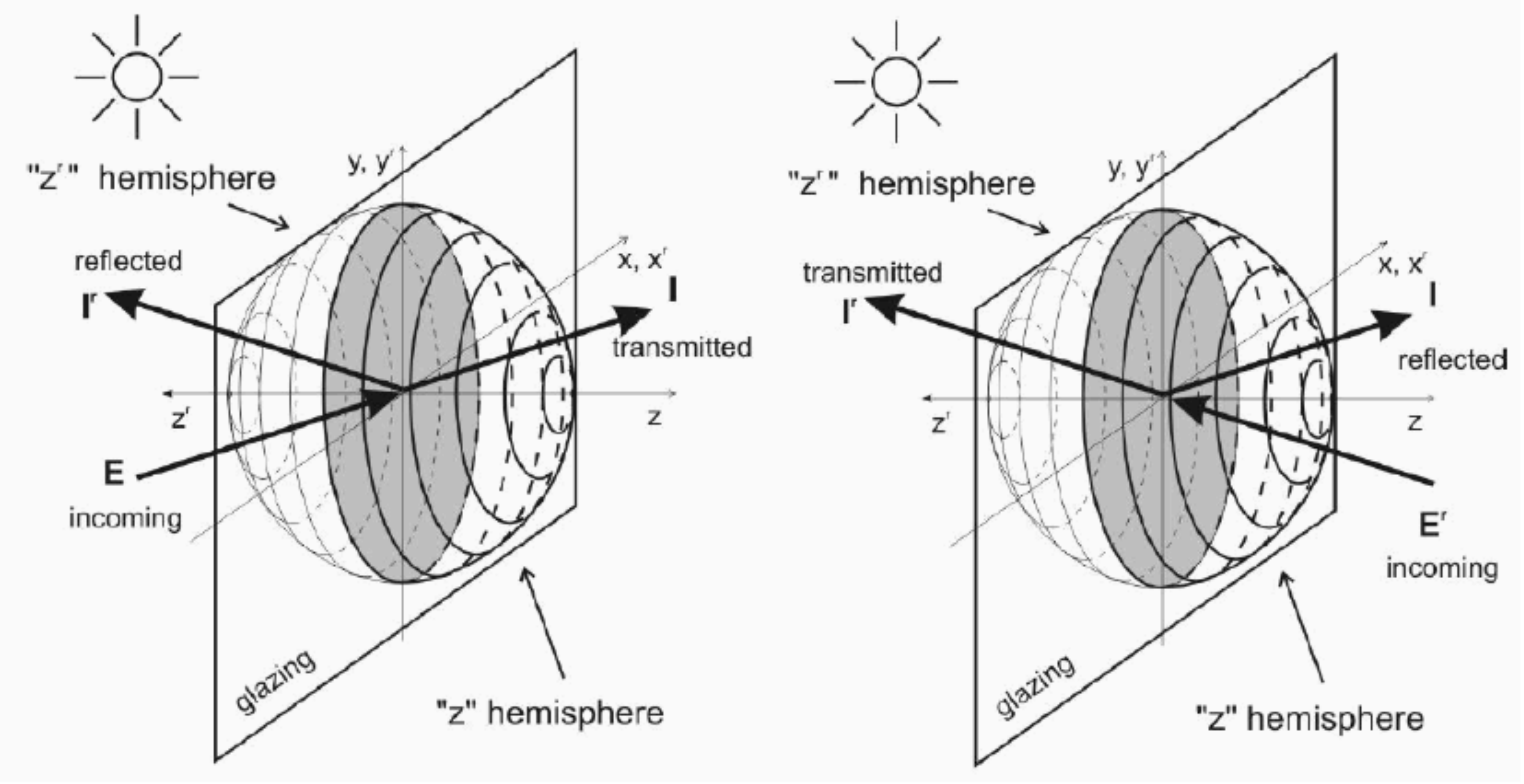
- "BSDF" stands for "Bidirectional Scattering Distribution Function" (BRDF + BTDF)
- Describes how light scatters off a surface
- Incl. wavelength, a 5-dimensional scalar function
  - unitless ratio of outgoing radiance over incoming irradiance (1/steradian)

# WINDOW 6 BSDF

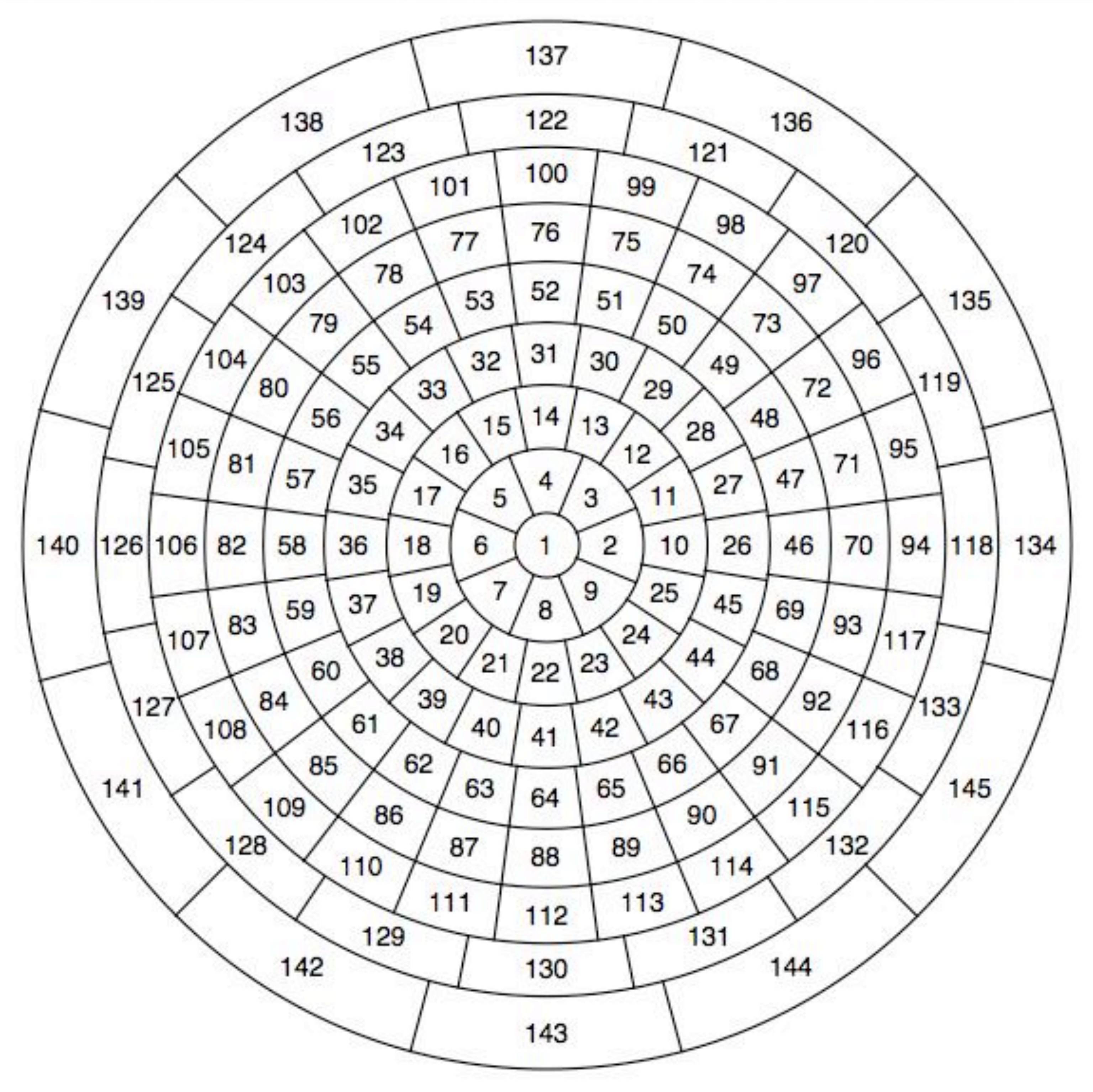
- Breaks BSDF into integrated spectra:
  - Visible (380-780 nm), Solar (300-2500 nm), NIR (780-2500 nm), FIR (5000- 40000 nm)
- Breaks scattering into four components:
  - Front Transmission, Back Transmission, Front Reflection, Back Reflection

# BSDF Coordinates (1)

## Klems/Window 6 system



# BSDF Coordinates (2)



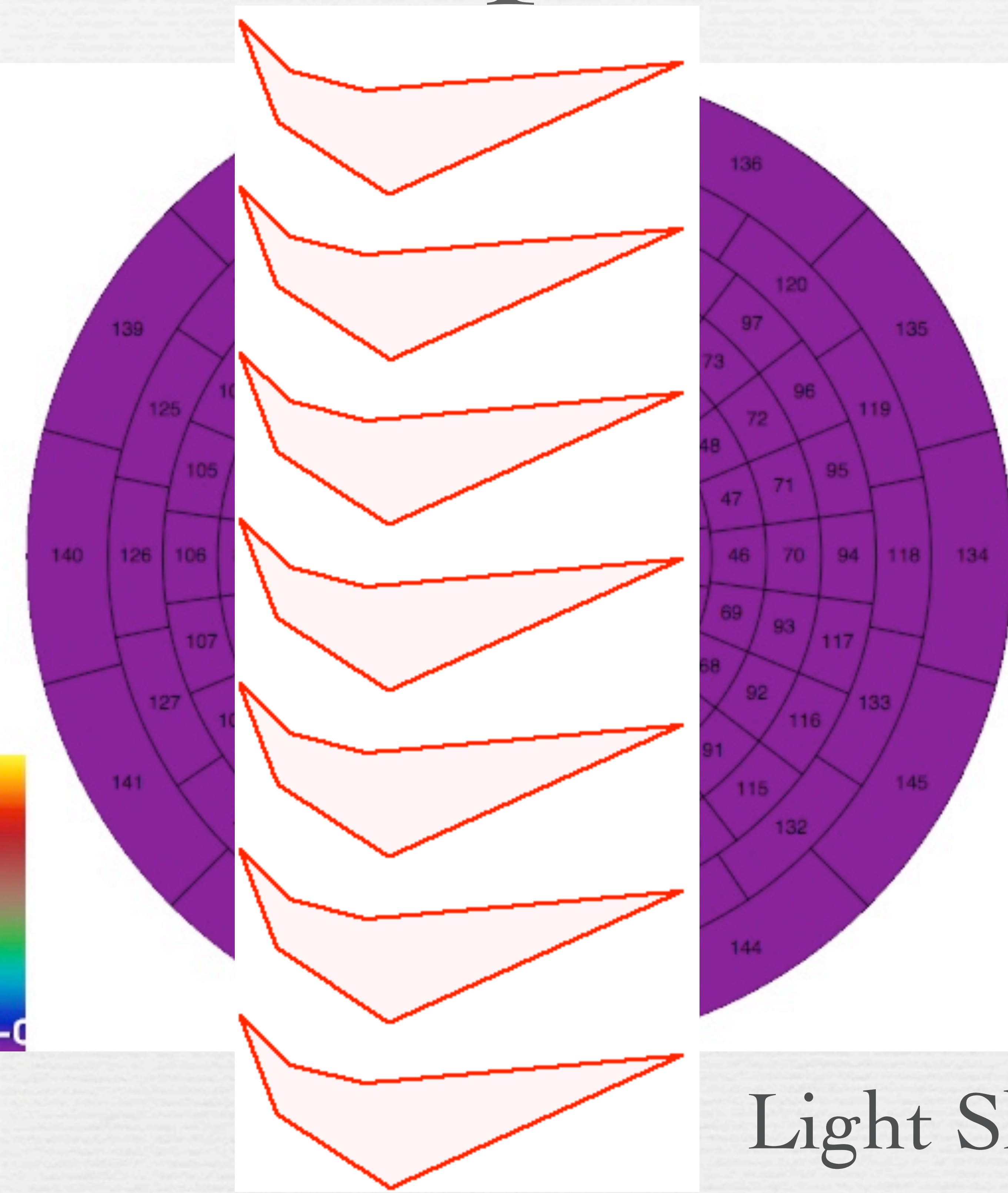
# WINDOW 6 XML File

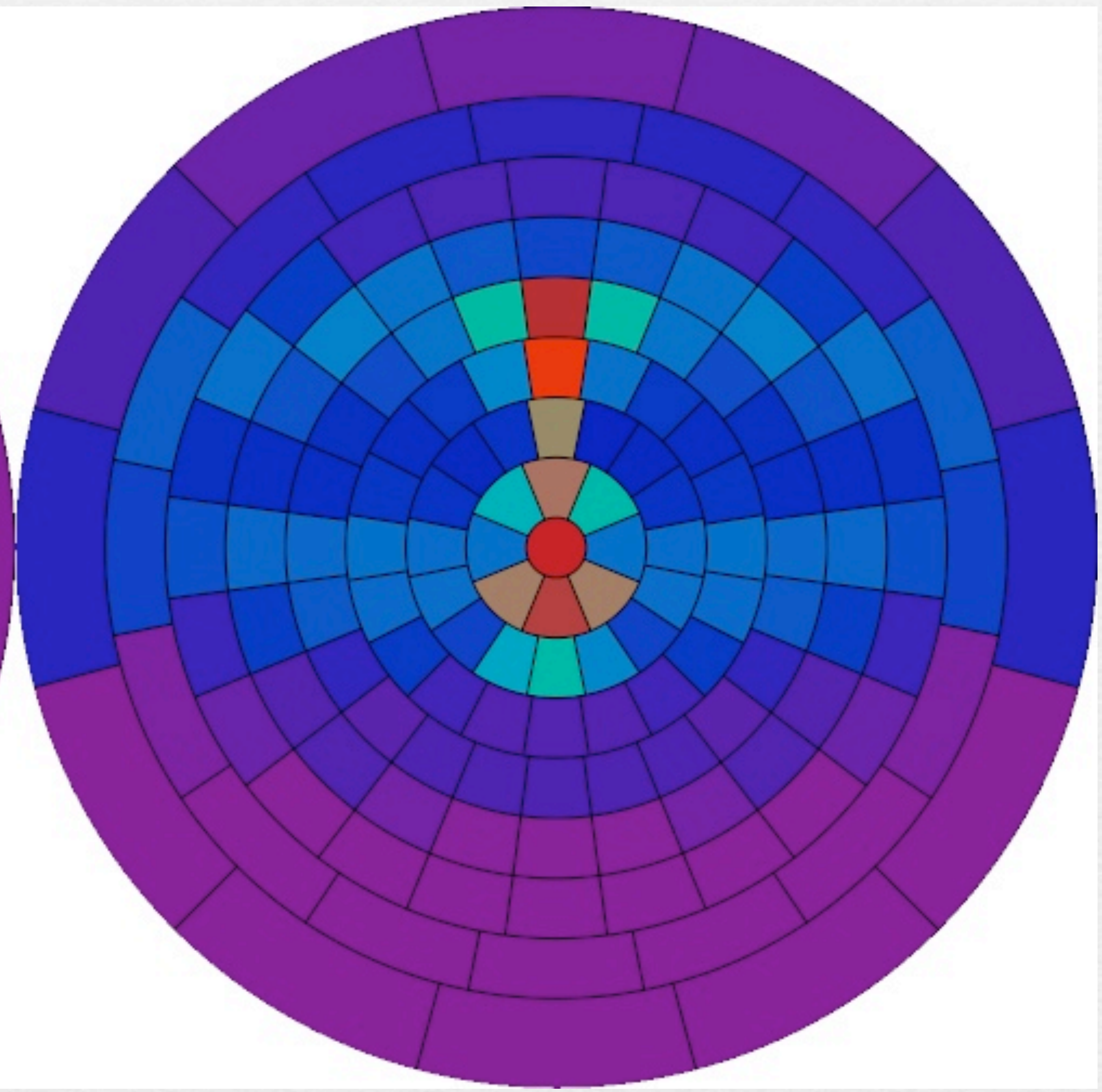

```
  <WavelengthData>
  <Wavelength unit="Integral">NIR</Wavelength>
  <SourceSpectrum>CIE Illuminant D65 1nm.ssp</SourceSpectrum>
  <DetectorSpectrum>ASTM E308 1931 Y.dsp</DetectorSpectrum>
  <WavelengthDataBlock>
      <WavelengthDataDirection>Transmission Front</WavelengthDataDirection>
      <ColumnAngleBasis>LBNL/Klems Full</ColumnAngleBasis>
      <RowAngleBasis>LBNL/Klems Full</RowAngleBasis>
      <ScatteringDataType>BTDF</ScatteringDataType>
      <ScatteringData>
2.443881,   0.047337,   0.041435,   0.038990,   0.041435,
0.047337,   0.048413,   0.046964,   0.048413,   0.047337,
0.040883,   0.035154,   0.031478,   0.030108,   0.031363,
0.035154,   0.040605,   0.047337,   0.048086,   0.044691,
0.042586,   0.042007,   0.042537,   0.044691,   0.047921,
0.047337,   0.038892,   0.031273,   0.025227,   0.021345,
0.020007,   0.021345,   0.025227,   0.031273,   0.038892,
...
```

# Getting BSDF Data

- WINDOW 6 will have access to a database of fenestration systems
- Systems may be combined in "layers" up to a point
- Some BSDFs are measured, most are simulated
  - BSDF measurements go into simulations, too
- New **genBSDF** program in *Radiance* 4.1

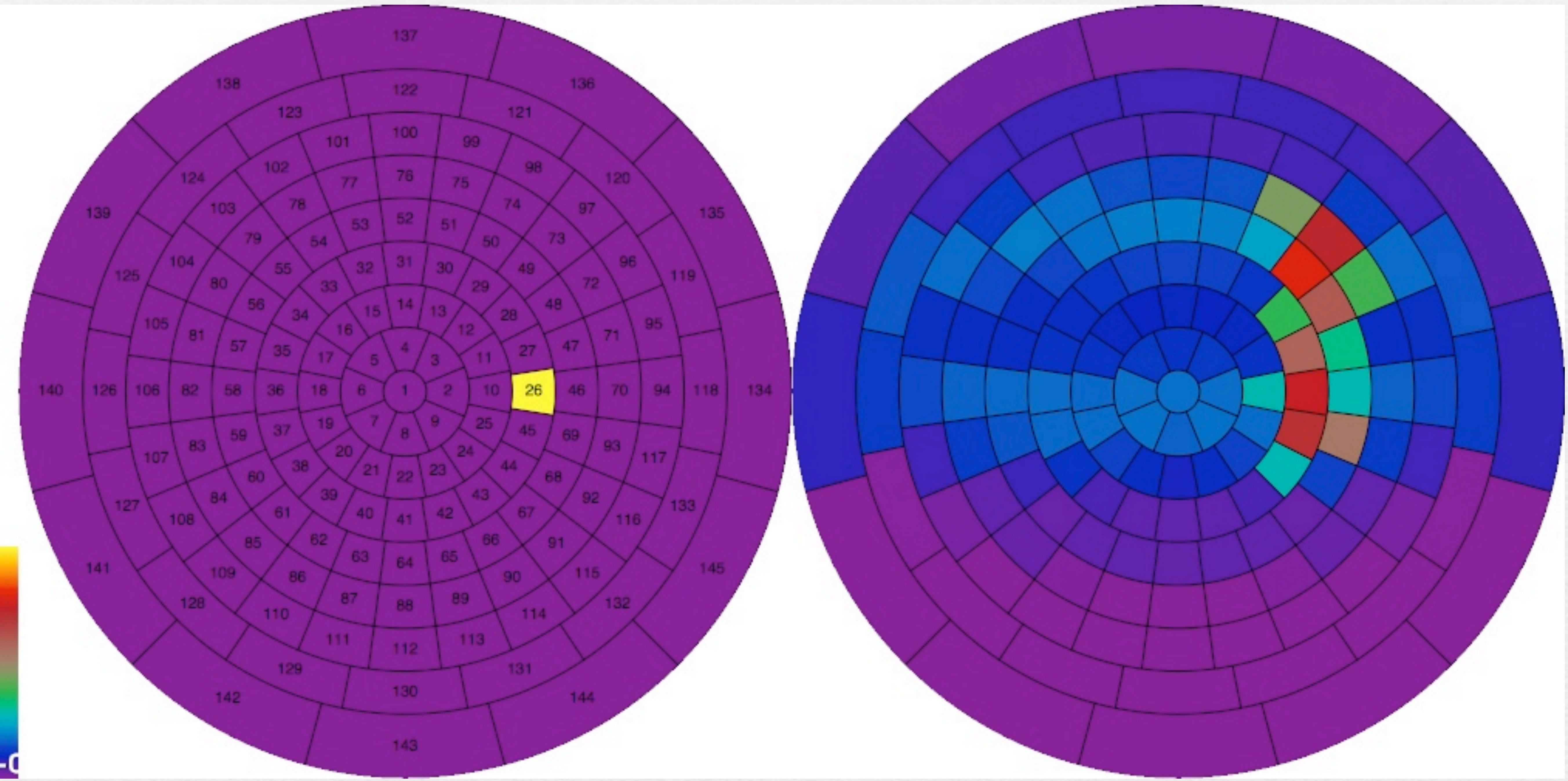
# Sample BTDF Data (1)



Light Shelf

# Sample BTDF Data (2)



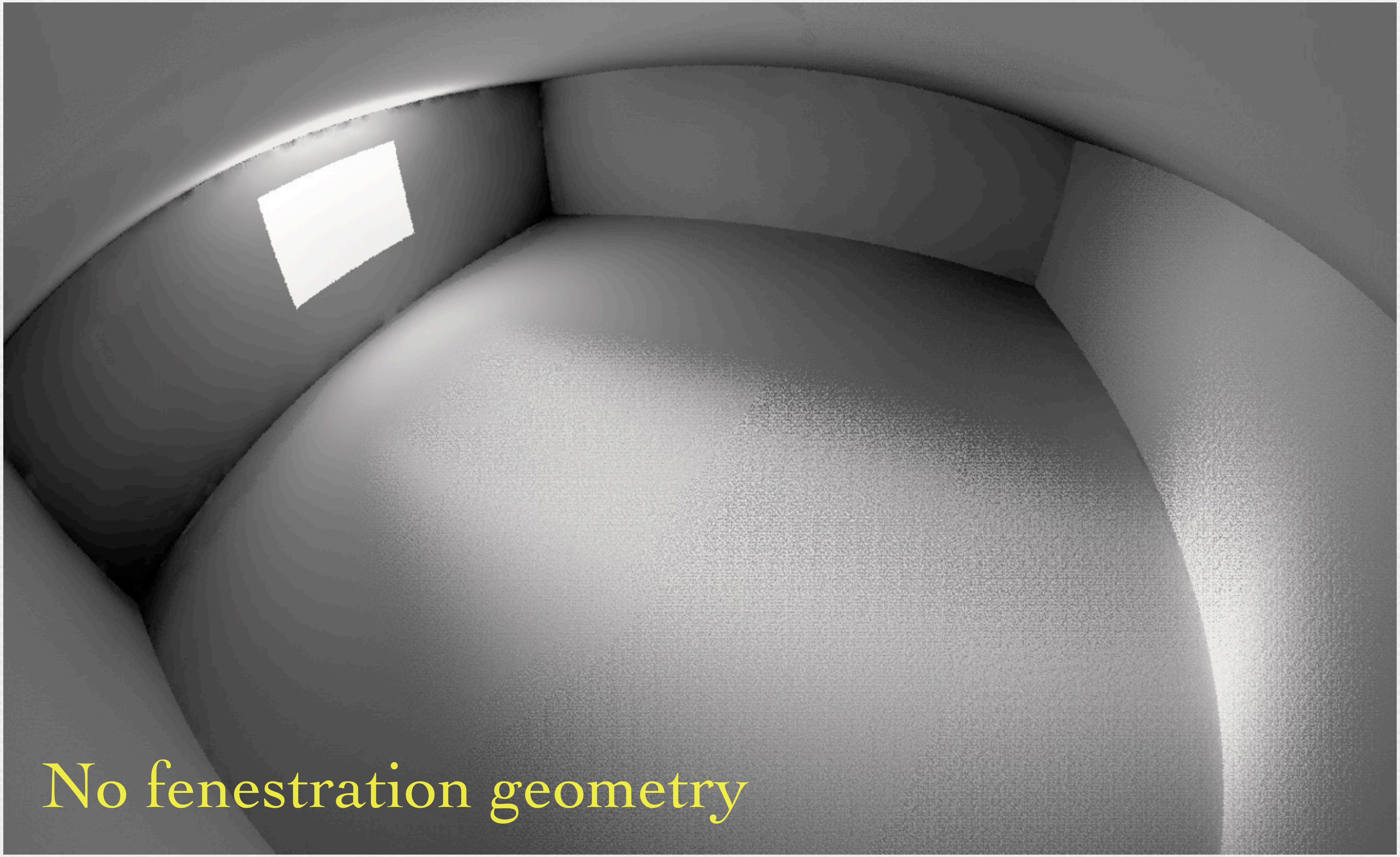

Light Shelf

Visualization by Andrew McNeil

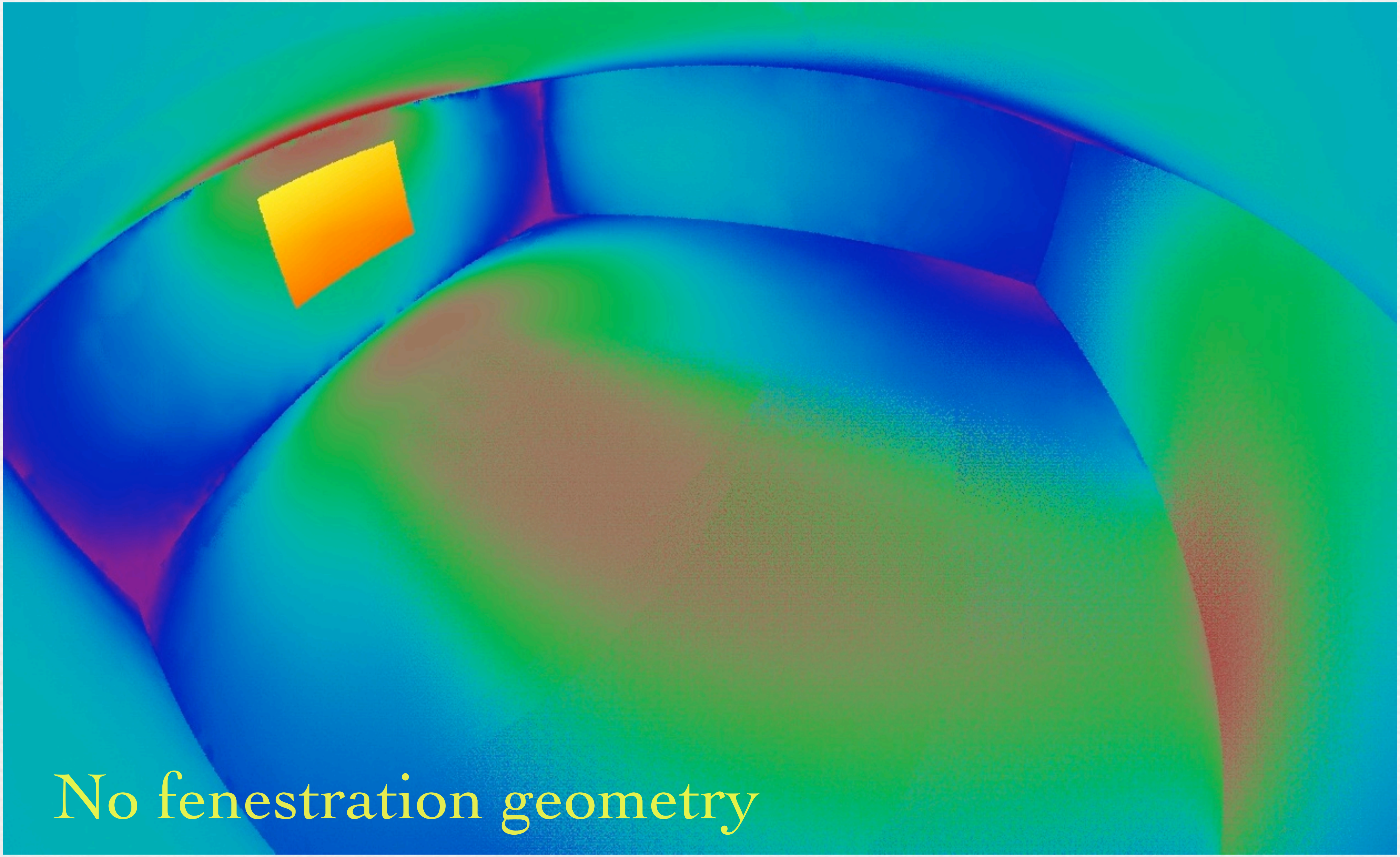
# BSDFs in **mkillum**

- Since 3.9 (2008), **mkillum** has supported BTDFs
  - fully debugged by 4.0 (Spring 2010)
- Works best when:
  - BTDF is somewhat diffuse, or
  - model includes geometry for direct component

# Example Results



# Averages Agree



# Specular Challenge

- 145 hemisphere directions cover 10-30° regions
  - The sun covers about 0.5°
- Breaking hemisphere further leads to drastic increases in computational load
  - 80000 divisions needed for sun-sized patches
  - that's an 80000x80000 BTDF (6.4e9 values)
  - ...and still no direct solar pattern on interior

# BSDF with Geometry

- Incorporate model of fenestration device in XML
- Use MGF (Materials and Geometry Format) developed for IESNA luminaire descriptions
    - simple, public <http://radsite.lbl.gov/mgf>
    - embeddable in XML
    - standard C parser makes support easy

# Sample MGF

XML embedding

```
<Geometry format="MGF" unit="Meter">
# Y-axis points "up", Z-axis into room, right-handed coordinates
m WhitePlastic =
    rd .7
    rs .02 0
    sides 2
o VenetianBlinds
xf -rx -60 -a 67 -t 0 .03 0
    o Slat
    v v1 =
        p -2 0 0
    v v2 =
        p 2 0 0
    v v3 =
        p 2 0 .04
    v v4 =
        p -2 0 .04
    f v1 v2 v3 v4
    o
xf
o
</Geometry>
```

Supports arrays

# New **genBSDF** script

- Perl script calls **rtcontrib** to compute BTDF
  - forward transmittance only at this time
- Simple operation:
  - `genBSDF fenes1.rad > fenes1.xml`
- Default is to embed MGF geometry
  - options for number of samples, processes, etc.

# BSDFs in **mkillum**

Use *illum* BSDF Thickness (flag to use MGF)

```
#@mkillum l- c=a d=fenes.xml t=0 u=+Z
```

Up orientation

```
void polygon window
0
0
12
                    0                      4                      1
                    0                   6.25                      1
                    0                   6.25                    2.5
                    0                      4                    2.5
```

# When To Use "l+" with BTDF Data

- When the window geometry is unknown
- When there is no direct beam component
  - redirected components usually use BTDF

# Annual Simulation

- Using mkillum with BTDFs is fairly quick, but...
- Re-rendering a scene 2000+ times for each hour?
- We need something faster...
- Can we use daylight coefficients with BTDF data?

# Three Phase Method

- **Phase I**:
  Use **rtcontrib** to get daylight coefficients relating sky patches to incident directions

- **Phase II**:
  Use **rtcontrib** to relate exiting portal directions to desired measurement locations (e.g., image)

- **Phase III** (time-step calculation):
  sky * incident * BTDF * exiting

# Our Matrix Equation

**i = VTDs**

where:

| | |
|---|---|
| **i** | is the desired result vector (radiances, irradiances, etc.) |
| **V** | is the "View" matrix defining the lighting connection between results and exiting directions for a window group |
| **T** | is the "Transmission" matrix defining the BTDF of the window group |
| **D** | is the "Daylight" matrix defining the coefficients between incoming directions for the window group and sky patches |
| **s** | is a vector of sky patch luminances for a particular time and date |

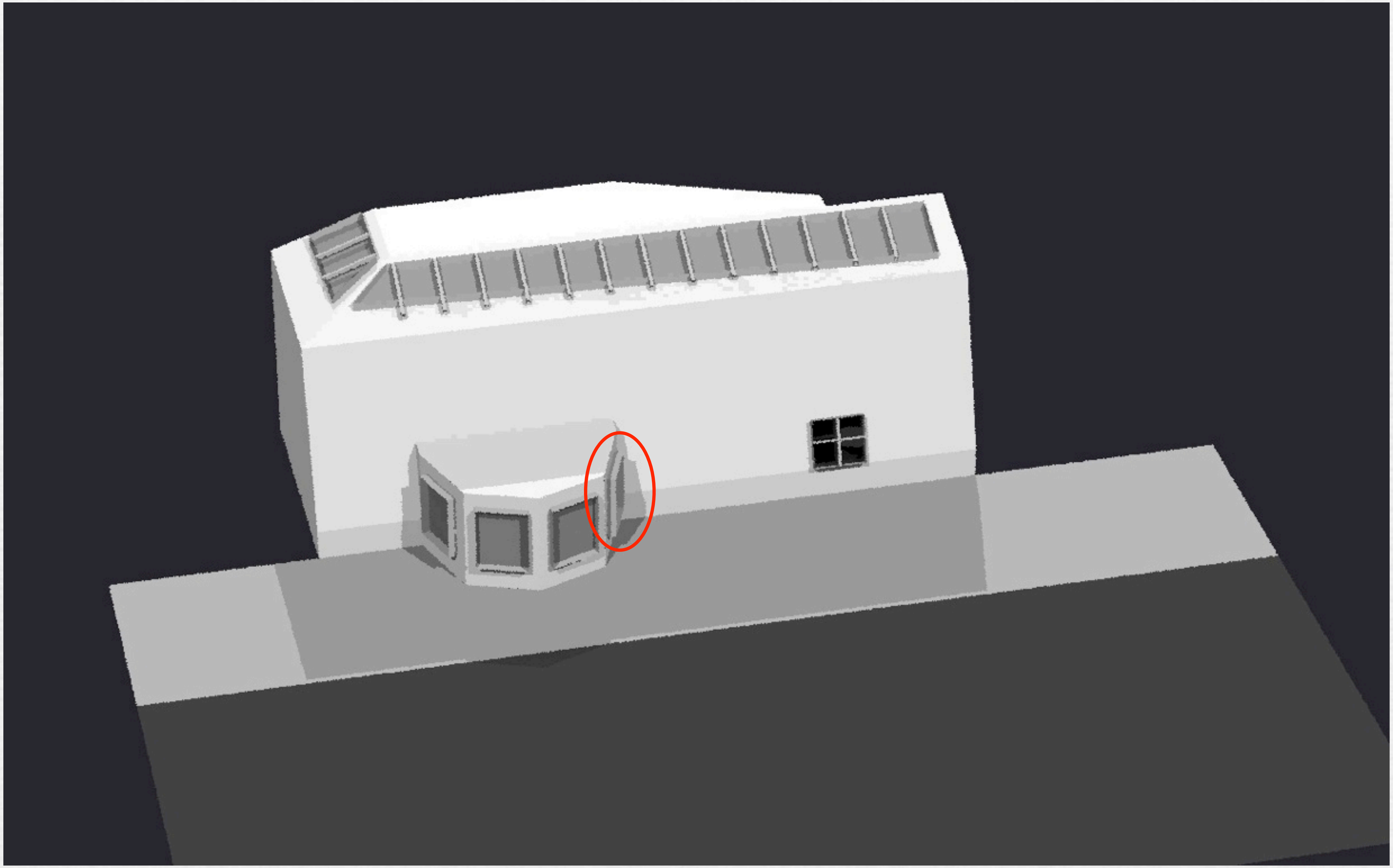
In a more explicit form, this would be:

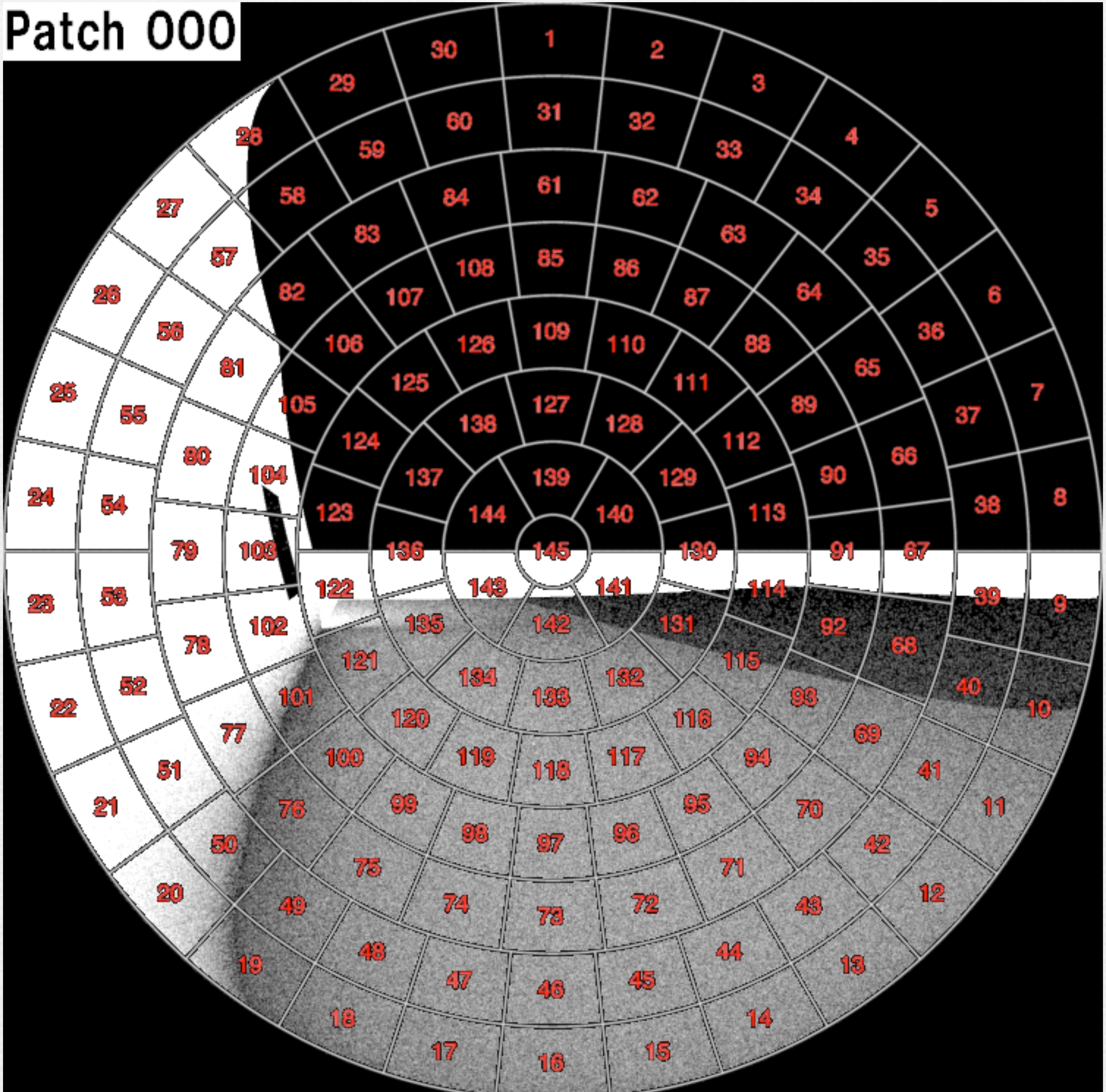
$$\begin{bmatrix} sens1 \\ \dots \\ sensM \end{bmatrix} = \begin{bmatrix} sens1edir1 & \dots & sens1edirN \\ \dots & \dots & \\ sensMedir1 & & sensMedirN \end{bmatrix} \begin{bmatrix} edir1idir1 & \dots & edir1idirN \\ \dots & \dots & \\ edirNidir1 & & edirNidirN \end{bmatrix} \begin{bmatrix} idir1dc1 & \dots & idir1dcK \\ \dots & \dots & \\ idirNdc1 & & idirNdcK \end{bmatrix} \begin{bmatrix} sky1 \\ \dots \\ skyK \end{bmatrix}$$

# Phase I: Compute D

- Apply **rtcontrib** to relate sky patches to incident directions on window exterior
  - Need separate calculation for each orientation and major geometric feature
  - New **genklemsamp** utility generates samples over a given window group
    - written in Perl

# Example Space




Patch 000
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100
101
102
103
104
105
106
107
108
109
110
111
112
113
114
115
116
117
118
119
120
121
122
123
124
125
126
127
128
129
130
131
132
133
134
135
136
137
138
139
140
141
142
143
144
145

# Phase II: Compute **V**

- Use **rtcontrib** to relate sensor locations to exiting directions on window interiors
  - a single run can cover all window groups
  - **`klems_int.cal`** file maps to BTDF coord.

Outgoing Directions for One Window Group

# Phase III: Time Step

- Use **genskyvec** to create sky patch vector **s**
- Use **dctimestep** to multiply it all together

BTDF

Sky Patch Vector

Result Vector

$$i = VTDs$$

View Matrix Daylight Matrix

# Phase III Example

```
gensky 9 21 12:00 -a 37.71 -o 122.21 -m 120 | genskyvec > sky.dat
pcomb '!dctimestep comp/back_SouthGroup%03d.hdr blinds1.xml SouthGroup.dmx sky.dat' \
      '!dctimestep comp/back_WestGroup%03d.hdr blinds2.xml WestGroup.dmx sky.dat' \
      '!dctimestep comp/back_NorthGroup%03d.hdr blinds2.xml NorthGroup.dmx sky.dat' \
      '!dctimestep comp/back_EastGroup_%03d.hdr blinds1.xml EastGroup.dmx sky.dat' \
   > back_9-21_1200.hdr
rm sky.dat
```

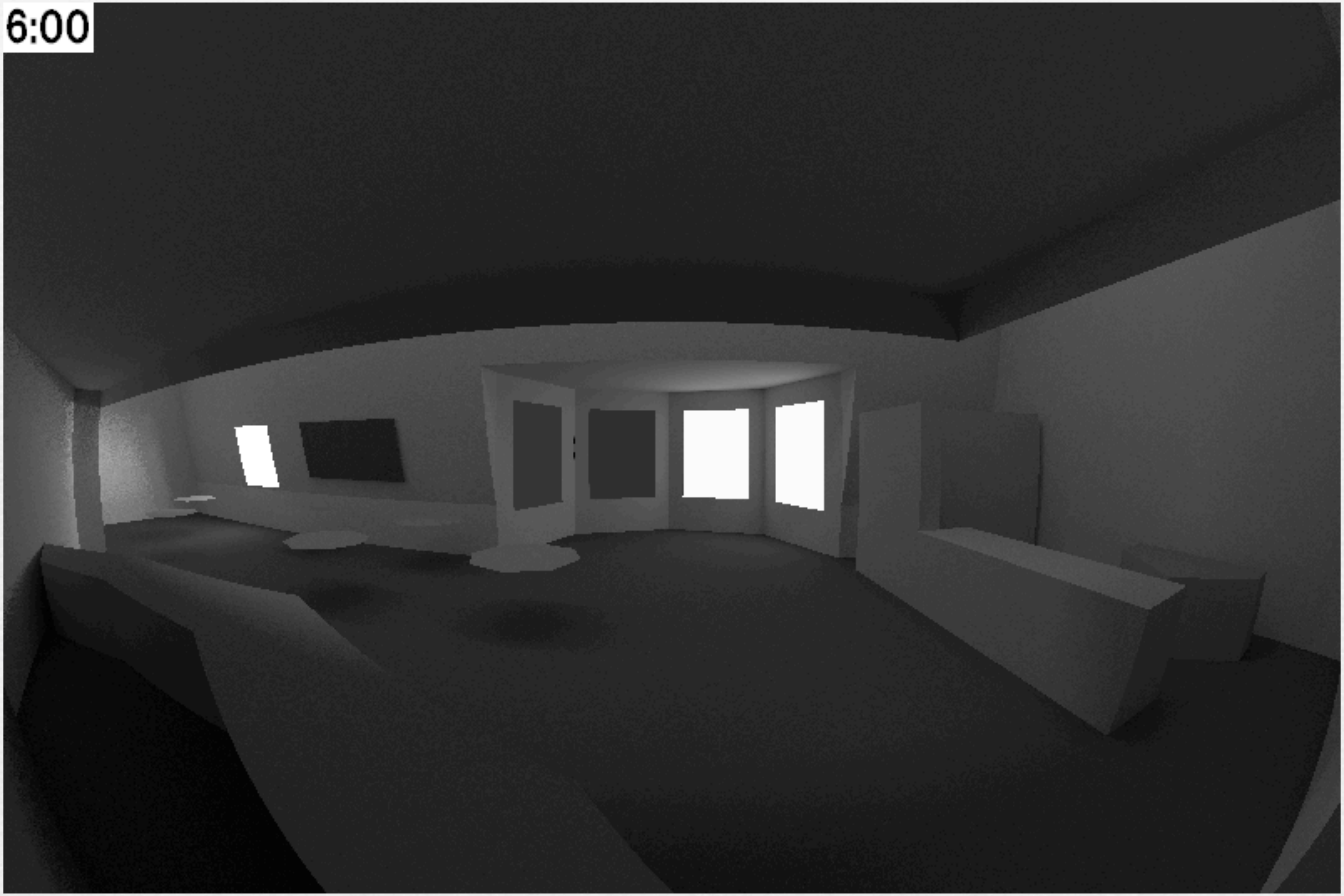
# Phase III Example

Generate sky vector for noon at the Autumn equinox

```
gensky 9 21 12:00 -a 37.71 -o 122.21 -m 120 | genskyvec > sky.dat
pcomb '!dctimestep comp/back_SouthGroup%03d.hdr blinds1.xml SouthGroup.dmx sky.dat' \
      '!dctimestep comp/back_WestGroup%03d.hdr blinds2.xml WestGroup.dmx sky.dat' \
      '!dctimestep comp/back_NorthGroup%03d.hdr blinds2.xml NorthGroup.dmx sky.dat' \
      '!dctimestep comp/back_EastGroup_%03d.hdr blinds1.xml EastGroup.dmx sky.dat' \
   > back_9-21_1200.hdr
rm sky.dat
```

Each call to dctimestep computes contributions of one window group

Time to run the above is less than 8 seconds on my laptop




Equinox Simulation

# Future Work

- Support for BRDF (reflection) data
- Further validation on appropriate use
- Variable-resolution & native BSDF types
- Documentation and tutorials
- What about a user interface?
  - always my last question, always unanswered