New Features in *Radiance* 6.0α

Greg Ward, Anyhere Software
Taoning Wang, LBNL

Minor Fixes/Enhancements

- Added **pcomb** -ff option and support for floating-point 1- and 3-component matrix i/o
  - Improves interoperability, allows for negative values and greater precision
- Improved ambient super-sampling algorithm
  - Better variance reduction in scenes with high-contrast edges
- Added "h=cie" hemisphere type to **rfluxmtx** to support 145-component CIE sky scanner pattern
- Optimized long argument lists in .cal files
  - Mostly for select() call when used to store arrays
- Made **rlam** and **rsplit** handle any number of files up to descriptor limit
- Bug fix in distant source sampling introduced in 5.4

Major Changes/ Additions (1)

- Introduced hyperspectral rendering & associated scene primitives (modifiers rather than materials)
  - new types: *spectrum*, *specfile*, *specfunc*, *specdata*, and *specpict*
  - updated ambient file format to support hyperspectral cache values
- Rendering tools have new **-cw** and **-cs** options to specify wavelength range and number of samples
- Both **rpict** and **rtrace** support **-p*** options to set output color space (**-pRGB**, **-pXYZ**, or **-pc** with custom primaries)
- New **rtrace -co** flag supports full spectral output
  - Also supported in updated **rfluxmtx**, **rcontrib**, and **rtpict**, which can produce hyperspectral matrices or images
  - Single-channel **-pY**, **-pS**, or **-pM** for photopic, scotopic, or melanopic

Major Changes/ Additions (2)

- New hyperspectral picture format (HSR) extends RGBE to multi-channel with common exponent
  - supported by rendering tools plus **pfilt**, **rcrop**, **rmtxop**, and new **rcomb**
  - **rcode2bmp**, and **radcompare** now handle HSR pictures
- Other tools have added spectral support:
  - new **mgf2rad -s** option converts MGF spectral materials to *Radiance*
  - **rsensor** and **rcontrib** produce number of spectral samples set by **-cs**
  - **mkillum**, **ranimove**, and **mksource** support spectral scene descriptions, but still reduce results to RGB
- New tools from Taoning support spectral skies:
  - **genssky** - generate spectral sky using atmospheric model for single time
  - **gensdaymtx** - generate annual spectral sky matrix from weather tape
    - supported by new **epw2wea -a** option

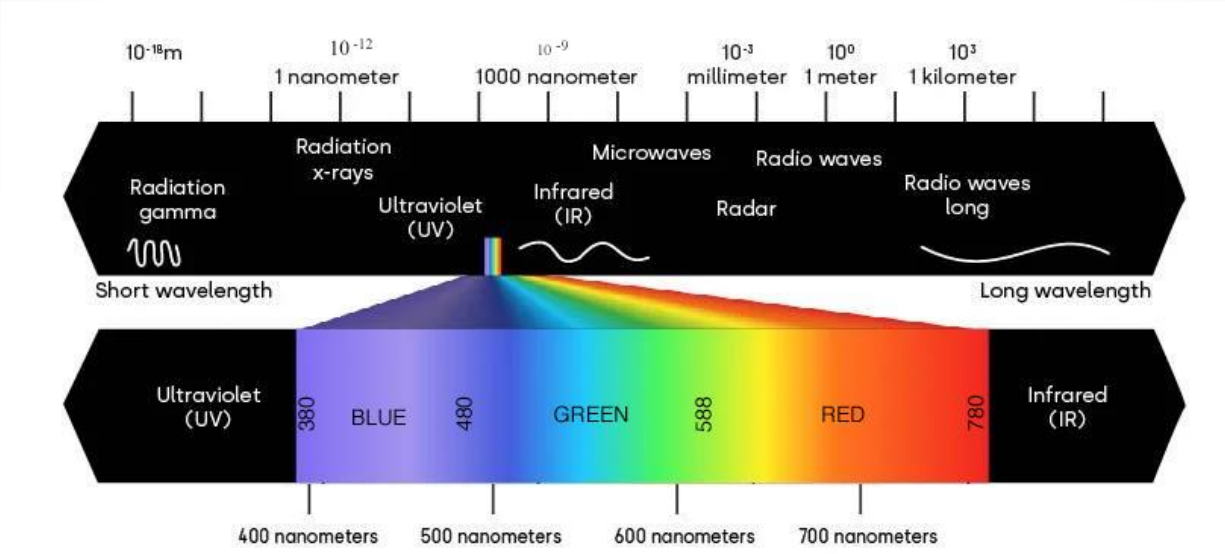
Major Changes/ Additions (3)

- New C++ RtraceSimulManager class in src/rt
  - Applied in experimental **rxtrace** tool, replicating **rtrace** functionality
- New C++ RpictSimulManager class in src/rt
  - Applied in experimental **rxpict** tool, which adds to **rpict** functionality with multi-processing (**-n** option) and hyperspectral (**-co+**) picture output
  - Class supports tiled rendering, with **rxpiece** tool forthcoming
- Thinking on more general rework of rendering process to minimize sampling error
  - Will save discussion for final workshop session

Hyperspectral Rendering

- Wavelength range partitioned into N divisions
  - Standard compilation limits N between 3 (RGB) and 24
    - Macro -DMAXCSAMP=24 can be changed at compile time
  - Input sample ranges and steps are converted during rendering
- Five new *Radiance* scene primitives:
  - *spectrum* is a basic spectral color
  - *specfile* takes static spectrum from file
  - *specfunc* is dynamic and procedural
  - *specdata* is dynamic and data-driven
  - *specpict* maps a hyperspectral (HSR) image
- Updated ambient file (**-af**) format
  - supports N hyperspectral samples as well as RGB
  - converts to/from current rendering mode with some accuracy loss

Partitioning the Spectrum



- Spectrum is partitioned into wavelengths of 380-480nm for BLUE, 480-588nm for GREEN, and 588-780nm for RED for purposes of multiplication between material colors and spectral patterns
- Extrema may be adjusted to cover UV and/or IR if desired, which does not alter the range of GREEN, only BLUE and RED
- The number of spectral samples may be set from 3 to MAXCSAMP

New *spectrum* Primitive

```
void spectrum gold_spec
0
0
82 380 775 0.389133964 0.389133964 0.389255302 0.389980986 0.391051245
     0.392138273 0.392908482 0.393032534 0.392429968 0.391351324
     0.390030615 0.388650157 0.387407915 0.386320218 0.385530031
     0.385510124 0.386040307 0.386026057 0.388298949 0.394215681
     0.404140535 0.419966115 0.443246002 0.473861019 0.511014174
     0.552643396 0.596304788 0.639685049 0.682955923 0.723645597
     0.759200442 0.789126545 0.813469177 0.832955943 0.848265499
     0.860485356 0.870549939 0.878925721 0.885276016 0.891159039
     0.896546999 0.901393946 0.905636686 0.909327028 0.912475235
     0.91521538 0.917680109 0.919998384 0.922297675 0.924966158
     0.927728576 0.930571538 0.933477024 0.936421086 0.939377245
     0.942439249 0.945588396 0.948563802 0.951319598 0.953828111
     0.956075841 0.958059262 0.959789105 0.961306963 0.96261608
     0.963753124 0.964748535 0.965627808 0.966413149 0.96712339
     0.967778124 0.968398351 0.968976934 0.969517604 0.970024269
     0.970500409 0.970949021 0.971372989 0.971775043 0.972157635

gold_spec metal gold_smat
0
0
5 1 1 1 1 0
```

New *specfile* Primitive

```
void specfile copper_spec
1 copper_spec.dat
0
0

copper_spec metal copper_smat
0
0
5 1 1 1 1 0
```

copper_spec.dat:

```
# Calculated spectrum for copper (unoxidized)
1
380 775 80

0.478218395
0.478218395
0.481854591
0.487980345
0.494156055
0.500345772
0.506516269
0.512635248
0.518673493
…
```

New *spectfunc* Primitive

```
# Third real argument is bandwidth and determines brightness/saturation
void specfunc rainbow_spec
8 xrainbow rainbow.cal -s 4 -t -6 0 0
0
3 380 780 45

rainbow_spec plastic rainbow_smat
0
0
5 1 1 1 0 0
```

rainbow.cal:

```
{
      Simple function to return spectral band along (0,1) interval
}

inBand(w,m) = A3/2 - abs(A1 + (A2-A1)*m - w); { wavelength in bandwidth for 0<m<1 ? }

xrainbow(w) = if(inBand(w,Px), 1, 0);    { simple on-off value along X axis }
```

New *spectdata* Primitive

```
void specdata gold+copper_alloy_spec
4 noop copper_gold.dat . ".5+.5*cos(PI*Py)"
0
0

gold+copper_alloy_spec metal gold+copper_alloy
0
0
5 1 1 1 1 0
```

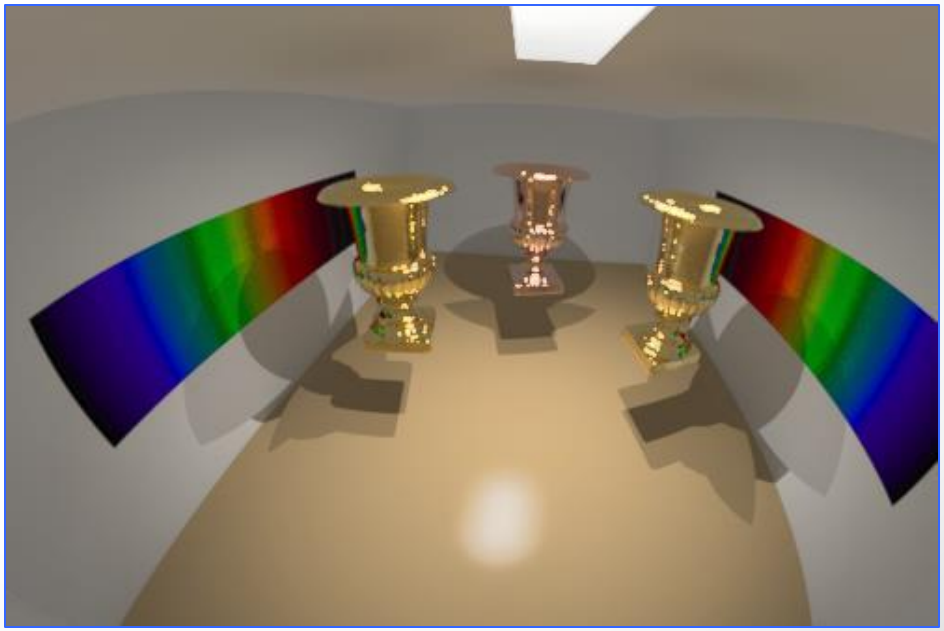
copper_gold.dat:

```
# Interpolate between Gold and Copper spectra
2
0 1 2
380 775 80

# Copper portion (first coord @ 0)
0.478218395
0.478218395
0.481854591
0.487980345
0.494156055
0.500345772
…
```

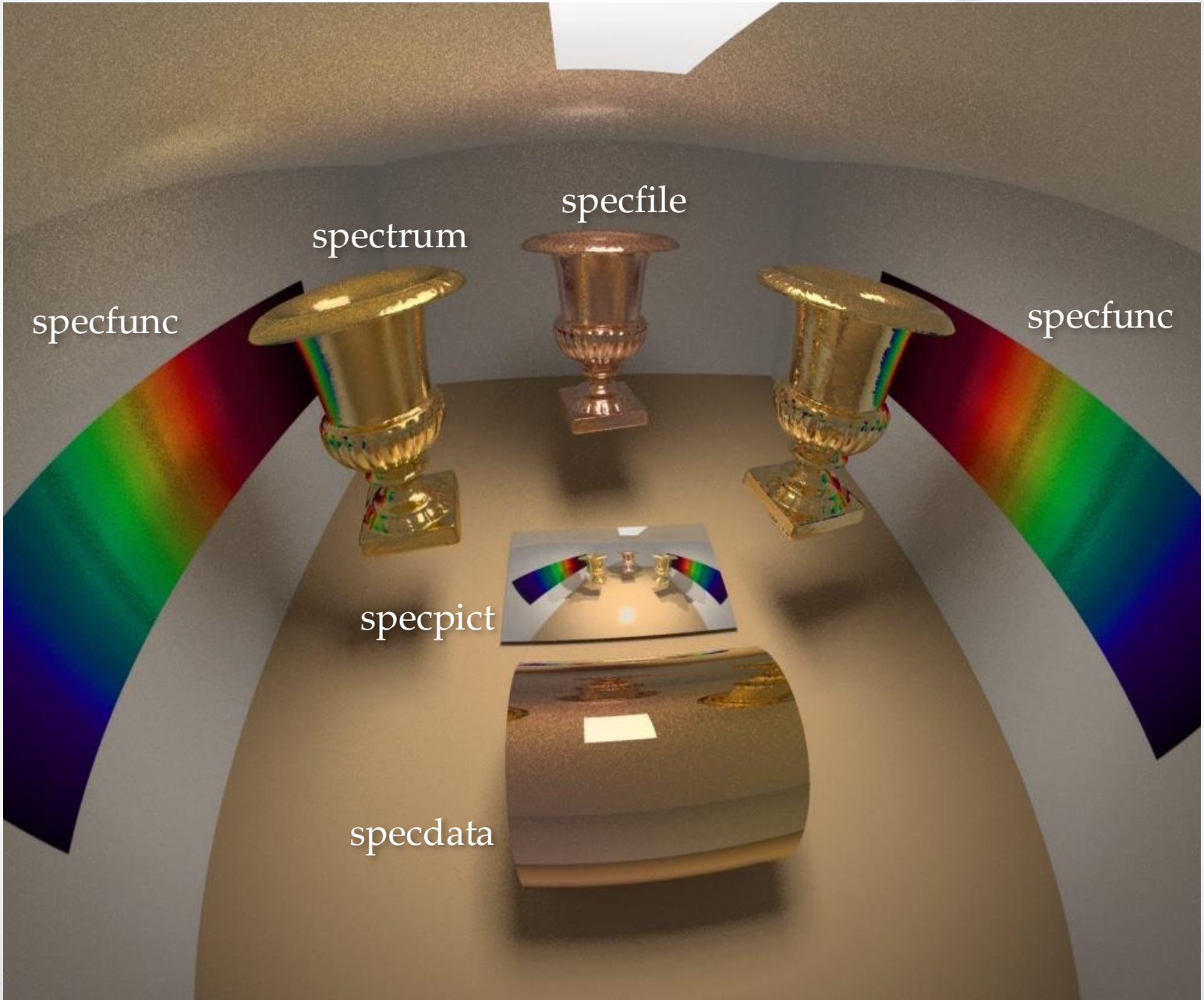
New *spectpict* Primitive

```
void specpict mini_me_spec
5 clip spfish.hsr . 3.25-Py -3-Px
0
0

mini_me_spec plastic mini_me_mat
0
0
5 1 1 1 0 0
```

spfish.hsr



Final Rendering



Hyperspectral Options by 6.0 Rendering Tools

-cw *start_wl end_wl*

starting and ending wavelengths in nanometers

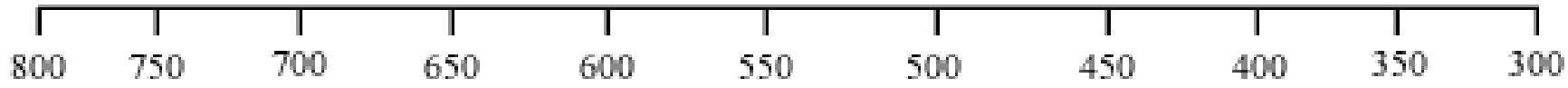
380 and 780 are defaults (ordering ignored)

-cs *number_of_divisions*

spectral sample count from 3 to MAXCSAMP

3 is default, which implies RGB rendering

Example: -cw 300 800 -cs 10



New Color Output Options

Supported by **rpict** and **rtrace**:

-pRGB # output RGB color space (default)

-pXYZ # output XYZ color space

-pc *xr yr xg yg xb yb xw yw* # custom color space

Supported by **rtrace**:

-pY # output single-channel photopic luminance

-pS # output single-channel scotopic luminance

-pM # output single-channel melanopic luminance

Note: **rcontrib** (& **rfluxmtx**) always produce N channels based on **-cs** setting

New Hyperspectral Radiance (HSR) Pictures

- Produced by **rtpict**, **rfluxmtx**, and **rcontrib**
- Manipulated by **pfilt**, **rcomb**, **rcrop**, and **rmtxop**
- **rcode2bmp** and **radcompare** also take HSR's
- Used during rendering by *specpict* primitive

- Encoding: N × 8-bit mantissas sharing 8-bit exponent
  - Unlike RGBE and XYZE, no run-length compression
- Required information header settings:

```
NCOMP=18
WAVELENGTH_SPLITS= 780 588 480 380
FORMAT=Radiance_spectra
```

Additional Hyperspectral Tool Support

- New **mgf2rad -s** option converts spectra
  - Default is to reduce to RGB as before, which is best if spectra unneeded
- Similar to **rcontrib** and **rfluxmtx**, **rsensor** produces number of spectral samples set by **-cs** option
- Updated **mkillum**, **ranimove**, and **mksource** produce RGB results, but employ spectral rendering to get there

New **genssky** and **gensdaymtx** tools

- New spectral sky model based on Bruenton & Nayret (2008)
- Modified Mie scattering profile computed from libradtrans
- Atmospheric scattering values are precomputed and interpolated using *Radiance* N-dimensional interpolation scheme (.dat files)
- **genssky** and **gensdaymtx** uses this model for point-in-time and annual simulation (!CIE or Perez)

Spectral Sky Output

- **genssky** produces an equirectangular HSR sky map, which is used by *spectpict* primitive for rendering
- **gensdaymtx** generates an annual sky matrix based on aerosol optical depth and total cloud cover
  - **rcomb** or **rmtxop** used to multiply matrices for time steps
- New **epw2wea -a** option provides values needed by **gensdaymtx**

Usage: **genssky**

genssky *month day hours* **-a** *lat* **-o** *lon* **-m** *tz*

-n multithreading for precomputation

-d broadband aerosol optical depth scales Mie scattering

-c total cloud cover 0-1

-l custom Mie scattering profile

-p output directory, stores both precomputed scattering and hyperspectral (HSR) picture

-f image file name prefix

-g ground reflectance, default = 0.2

-r hypyspectral image y resolution, x = y/2, default=64

Usage: **epw2wea** and **genssky**

epw2wea -a *sample.epw* [*sample.wea*]

gensdaymtx -m *4* [*sample.wea*]

-n multithreading for precomputation

Precomputation triggered for each new AOD value

-d/**-s** for sky/sun only matrix

-r sky rotation

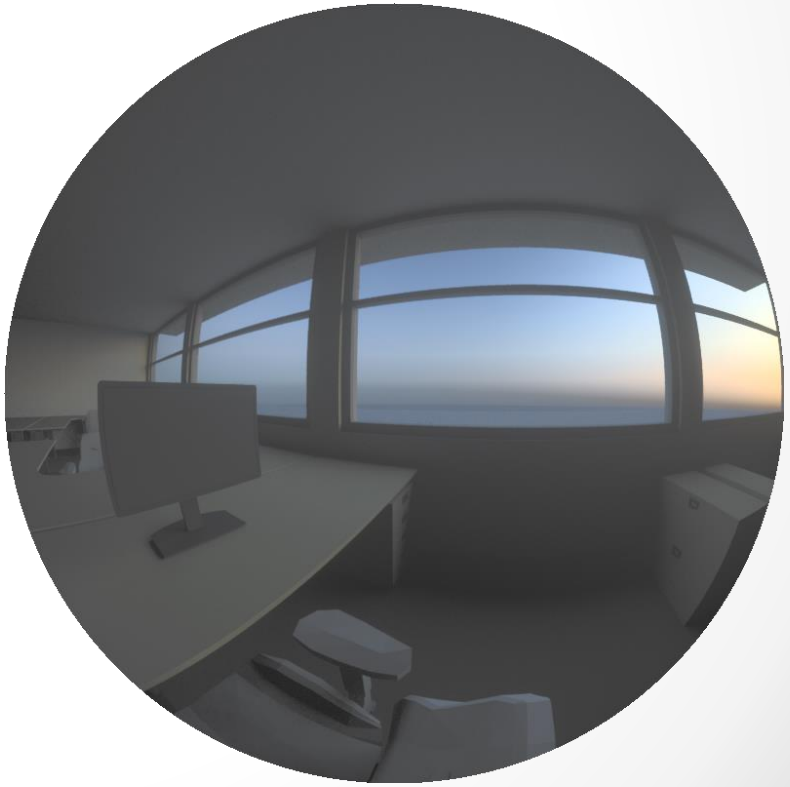
-p directory storing precomputed scattering files

-g ground reflectance, default = 0.2

-u daylight hours only (sun above horizon)

genssky Examples

Sample renderings with **genssky** with 15 and -5 degree solar altitude angle



Unfinished Pieces

- Currently, photon map does not work in hyperspectral calculation
  - Not even sure how it fails, as it hasn't been tested -- best to leave it off with -cs > 3
- Current **dctimestep** does not handle HSR input, as it only ever computes 3 color channels
  - Inconvenient for annual simulations, but **rmtxop** and new **rcomb** can fill in with updated **rsplit** program as shown in yesterday's tutorial
  - Matrix multiplication with hyperspectral can be slow
- New **genssky** and **gensdaymtx** tools always use 20 samples over default visible spectral range
  - Also need further testing and input calibration method

Questions?