New *Radiance* “WGMDfunc” Material Primitive

Greg Ward

Anyhere Software

The Reason (It better be good…)

- Gap between basic types and BSDF materials
  - *plastic*, *metal*, *glass*, etc. have little flexibility
  - *plasfunc*, *metdata*, *BRTDfunc*, *aBSDF* require measurements and/or coding in beloved .cal files
- Spectral colors come in via pattern modifiers and are assigned based on material type
  - difficult to assign to different components
- *mixfunc* awkward for combining materials

The Concept

- Build on “physically plausible” Ward-Geisler-Moroder-Dür reflectance model
  - anisotropic Gaussian lobe + Lambertian
  - includes Fresnel component if smooth (0 roughness)
- Provide functional control over each component & separate modifier paths
  - compromise between simplicity and flexibility
  - direct mechanism to control spectra

Taoning Wang made initial suggestion

David Geisler-Moroder provided guidance/feedback

The Details

- Four independent modifiers for specular and diffuse reflectance and transmission
  - mixtures only other primitives with > 1 modifier path
- Expressions for specular parameters:

  rs = specular reflection (front and back)

  rs_urough, *rs_vrough* = anisotropic rs roughness

  ts = specular transmission (front and back)

  ts_urough, *ts_vrough* = anisotropic ts roughness

  ux, *uy*, *uz* = anisotropic orientation vector

The Caveats

- Specular colors/spectra defined via modifiers
  - requires some outboard *colorfunc* primitives
- Not all functions are valid as parameters
  - variable coefficients and/or roughness often violate Helmholtz reciprocity
    - the same holds true for other BSDF primitive types
- Care required that extra light is not generated
  - sum of coefficients should be < 1.0

Basic Template

```
mod WGMDfunc id
13+  rs_mod  rs  rs_urough rs_vrough
     ts_mod  ts  ts_urough ts_vrough
     td_mod
     ux uy uz  funcfile  [transform]
0
9+   rfdif gfdif bfdif
     rbdif gbdif bbdif
     rtdif gtdif btdif
     [A10 ..]
```

- The *rs* and *ts* expressions determine specular reflection and transmission components, respectively, and either or both may be 0
- The main modifier (*mod*) affects diffuse reflection, but any of the named modifiers may share it by specifying ‘inherit’
- Similarly, a ‘void’ modifier equates to white

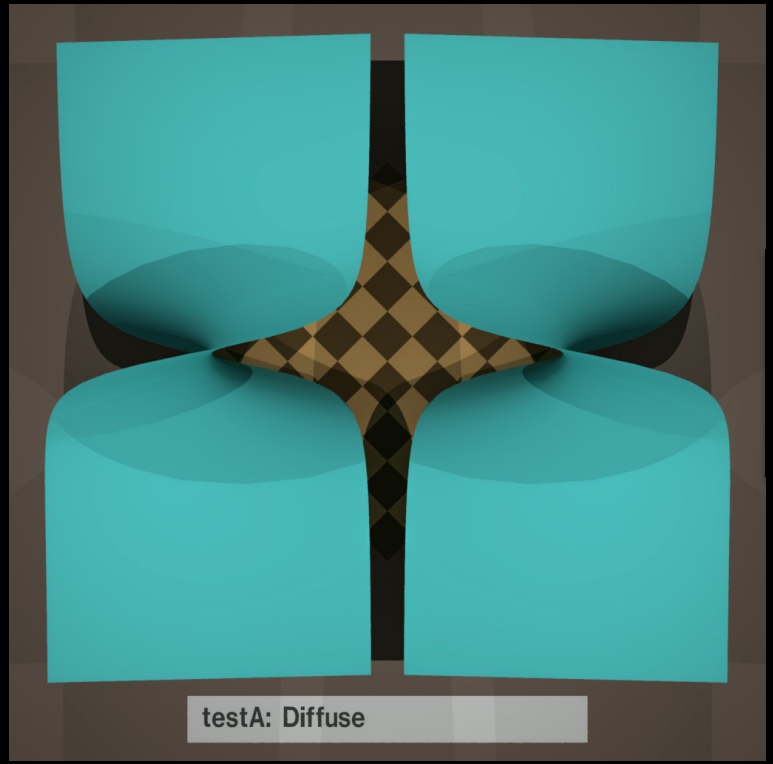
mod WGMDfunc *id*
13+ *rs_mod rs rs_urough rs_vrough*
ts_mod ts ts_urough ts_vrough
td_mod
ux uy uz funcfile [transform]
0
9+ *rfdif gfdif bfdif*
rbdif gbdif bbdif
rtdif gtdif btdif
[A10 ..]

- The *rs_mod*, *ts_mod*, and *td_mod* identifiers specify independent reflected-specular, transmitted-specular, and transmitted-diffuse modifiers
- The *rs_urough* and *rs_vrough* roughnesses apply to reflected-specular
- The *ts_urough* and *ts_vrough* expressions apply to transmitted-specular
- The *ux*, *uy*, and *uz* expressions orient these roughness values
- Real parameters give diffuse RGB values, which may be modified by spectra

WGMDfunc Example (1)

WGMDfunc

```
void WGMDfunc wgmdf_mat
13
      void 0 0 0
      void 0 0 0
      void
      0 0 0 .
0
9
      .1 .4 .4
      .1 .4 .4
      0 0 0
```



Reference

```
void plastic ref_mat
0
0
5 .1 .4 .4 0 0
```

WGMDfunc Example (2)

WGMDfunc

```
void colorfunc cchecked
4 check(.7) check(.05)
check(.05) locheck.cal
0
0

cchecked WGMDfunc wgmdf_mat
13
      void 0 0 0
      void 0 0 0
      void
      0 0 0 .
0
9
      .8 .8 .8
      .8 .8 .8
      0 0 0
```

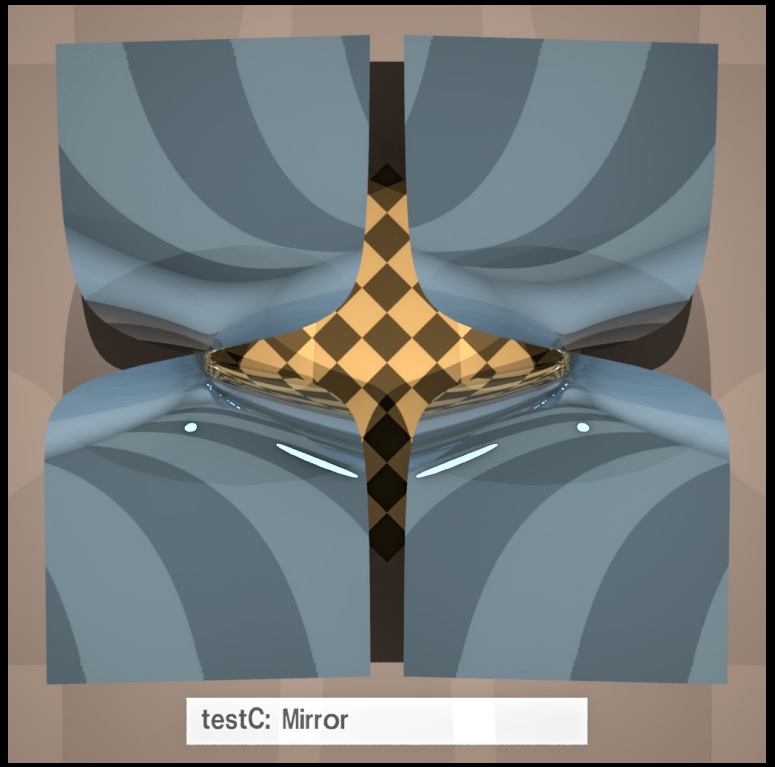
Reference

```
void colorfunc cchecked
4 check(.7) check(.05)
check(.05) locheck.cal
0
0

cchecked plastic ref_mat
0
0
5 .8 .8 .8 0 0
```

WGMDfunc Example (3)

WGMDfunc

```
void colorfunc metal_color
4 .3 .4 .5  .
0
0


metal_color WGMDfunc
wgmdf_mat
13
      inherit .9 0 0
      void 0 0 0
      void
      0 0 0 .
0
9
      .1 .1 .1
      .1 .1 .1
      0 0 0
```



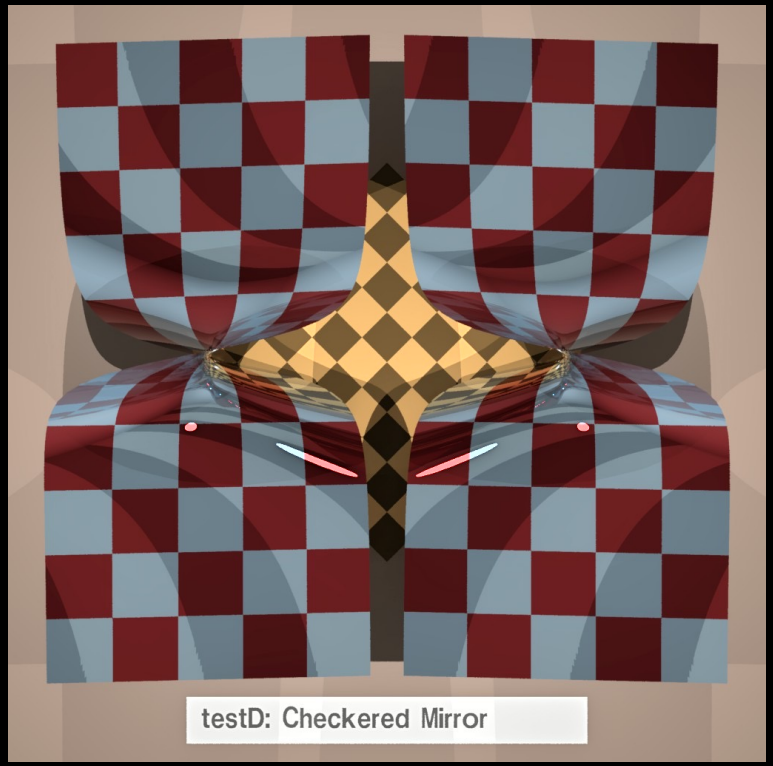
Reference

```
void metal ref_mat
0
0
5 .3 .4 .5 .9 0
```

WGMDfunc Example (4)

WGMDfunc

```
void colorfunc cchecked
4 check(.7) check(.05)
check(.05) locheck.cal
0
0

metal_color WGMDfunc
wgmdf_mat
13
      inherit .9 0 0
      void 0 0 0
      void
      0 0 0 .
0
9
      .1 .1 .1
      .1 .1 .1
      0 0 0
```



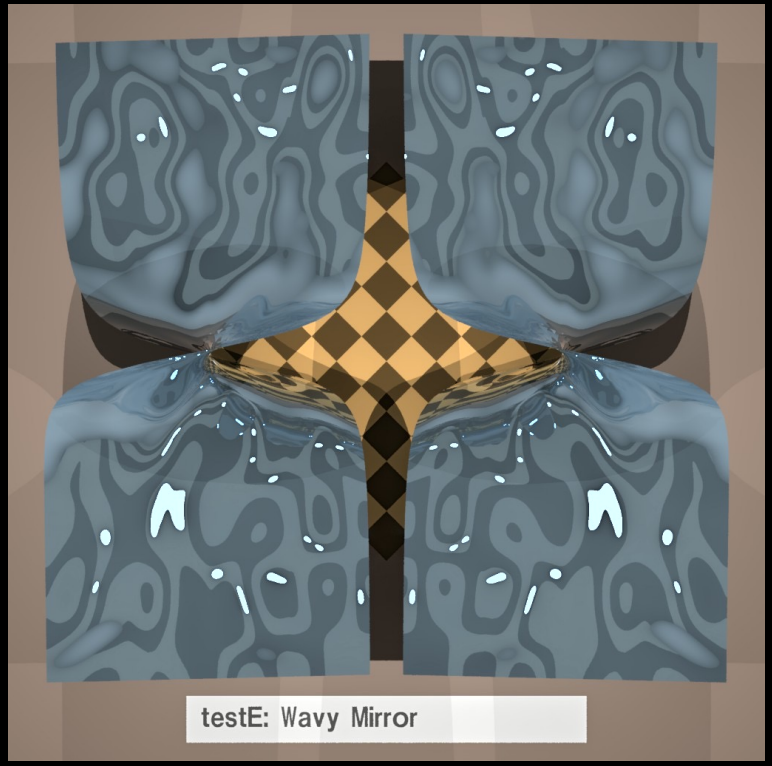
Reference

```
void colorfunc cchecked
4 check(.7) check(.05)
check(.05) locheck.cal
0
0

void colorfunc cchecked
4 check(.7) check(.05)
check(.05) locheck.cal
0
0

cchecked metal ref_mat
0
0
5 .3 .4 .5 .9 0
```

WGMDfunc Example (5)

WGMDfunc

```
void texfunc wavy
4
.2*noise3a(8*Lu,4*Lv,Pz)
"if(Py,.2,-
.2)*noise3b(8*Lu,4*Lv,Pz)"
.2*noise3c(8*Lu,4*Lv,Pz)  .
0
0

wavy colorfunc metal_color
4 .3 .4 .5  .
0
0

metal_color WGMDfunc wgmdf_mat
13
      inherit .9 0 0
      void 0 0 0
      void
      0 0 0 .
0
9
      .1 .1 .1
      .1 .1 .1
      0 0 0
```



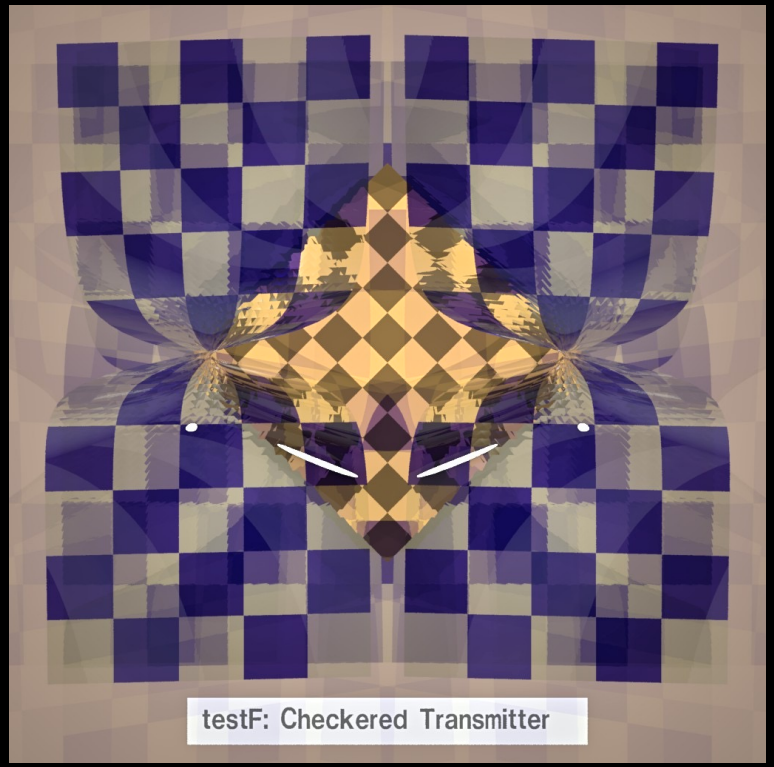
Reference

```
void texfunc wavy
4
.2*noise3a(8*Lu,4*Lv,Pz)
"if(Py,.2,-.2)*
noise3b(8*Lu,4*Lv,Pz)"
.2*noise3c(8*Lu,4*Lv,Pz)  .
0
0

wavy metal ref_mat
0
0
5 .3 .4 .5 .9 0
```

WGMDfunc Example (6)

WGMDfunc

```
void colorfunc cchecked
4 check(.1) check(.05) check(.7)
locheck.cal
0
0

cchecked colorfunc trans_color
4 .85 .9 .8 .
0
0

void WGMDfunc wgmdf_mat
13
      void .07 0 0
      trans_color .93 0 0
      void
      0 0 0 .
0
9
      0 0 0
      0 0 0
      0 0 0
```



Reference

```
void colorfunc cchecked
4 check(.1) check(.05)
check(.7) locheck.cal
0
0

cchecked trans ref_mat
0
0
7 .85 .9 .8 .07 0 1 1
```

WGMDfunc Example (7)

WGMDfunc

```
void colorfunc cchecked
4 check(.1) check(.05) check(.7)
locheck.cal
0
0

cchecked WGMDfunc wgmdf_mat
13
      void .07 0 0
      void 0 0 0
      inherit
      0 0 0 .
0
9
      0.39525 0.4185 0.372
      0.39525 0.4185 0.372
      0.39525 0.4185 0.372
```

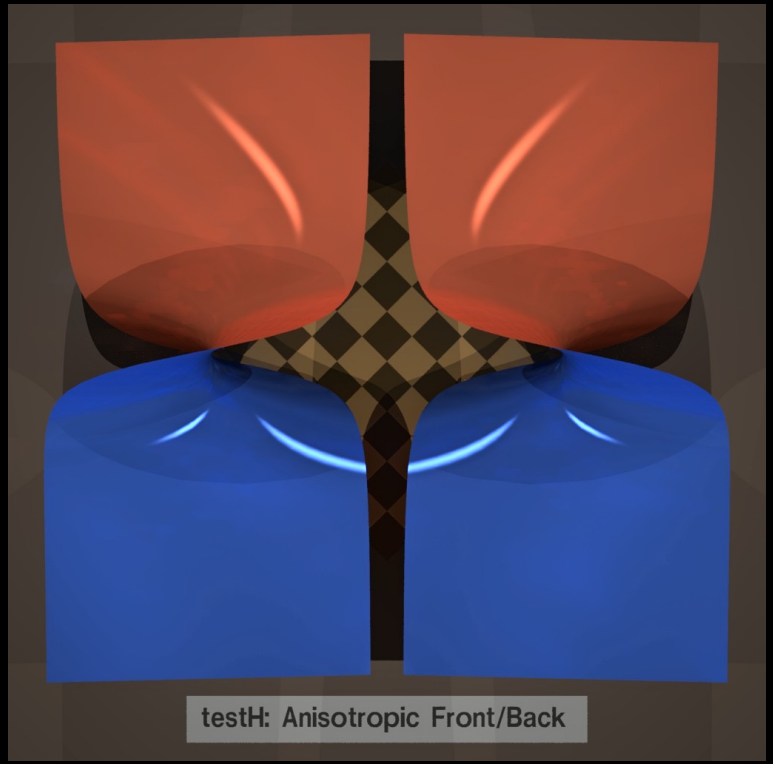
Reference

```
void colorfunc cchecked
4 check(.1) check(.05)
check(.7) locheck.cal
0
0

cchecked trans ref_mat
0
0
7 .85 .9 .8 .07 0 .5 0
```

WGMDfunc Example (8)

WGMDfunc

```
void colorfunc mat_color
4 if(Rdot,.1,.9) .2
if(Rdot,.9,.1) .
0
0

mat_color WGMDfunc wgmdf_mat
13
      inherit .5 .01 .08
      void 0 0 0
      void
      1 if(Rdot,1,-1) 0 .
0
9
      .5 .5 .5
      .5 .5 .5
      0 0 0
```



Reference

```
void metal2 ref_front
4 1 -1 0 .
0
6 .1 .2 .9 .5 .01 .08

void metal2 ref_back
4 1 1 0 .
0
6 .9 .2 .1 .5 .01 .08

void mixfunc ref_mat
4 ref_front ref_back
if(Rdot,1,0) .
0
0
```

More General Examples

- None of the foregoing really exercised the new capabilities of *WGMDfunc*

  …because we wished to compare to existing materials to make sure there were no deviations

- The following scene is part of the regression test suite used during *Radiance* builds

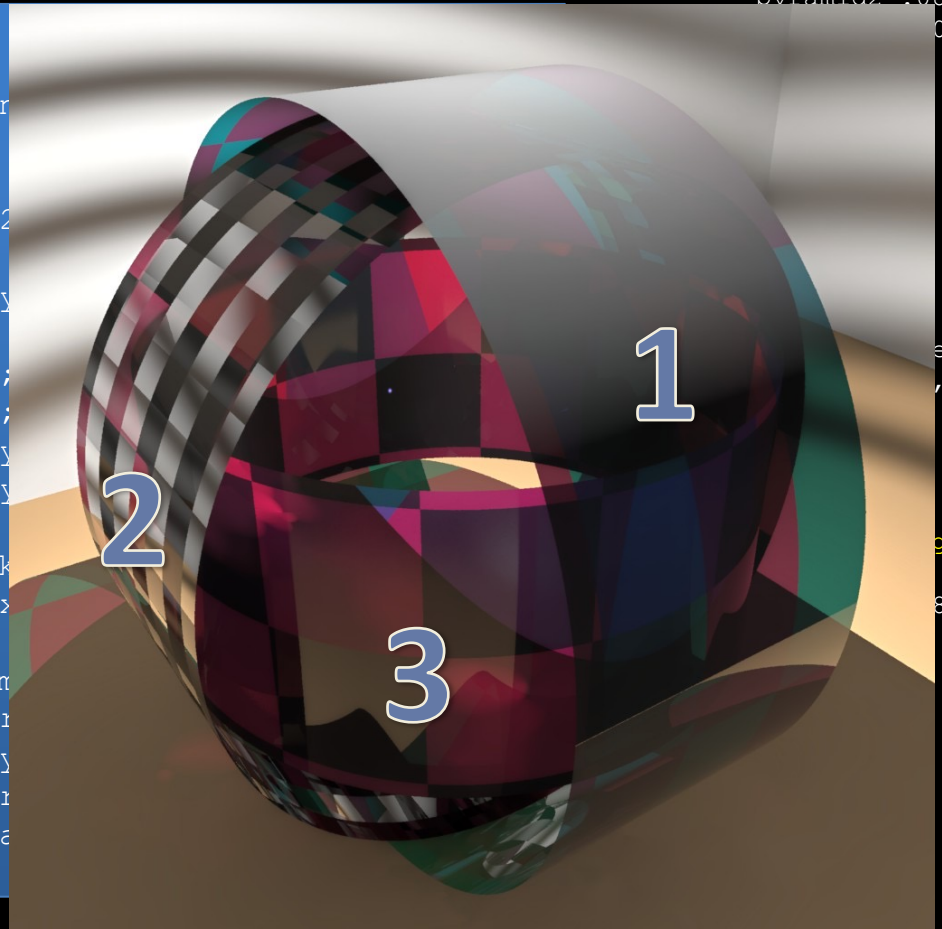
WGMDfunc Examples (9-11)

```
void colorfunc check1
6 if(Check(12),.5,.1) if(Check(12),.1,.7)
if(Check(12),.4,.9)
      cylmods.cal -ry 90
0
0

void WGMDfunc wgmd_mat1
19
      void if(Px,.08,0) 0 0
      check1 .2 0 0
      void
      0 0 1
      cylmods.cal -ry 90 -t -4 2.5 1.1
0
9
      .01    .01    .01
      .05    .2     .05
      0      0      0

void texfunc pyramid2
12 Pyramid_dx(13) Pyramid_dy(13) Pyramid_dz(13)
      cylmods.cal -s .9 -rx 90 -t -4 2.5 1.1
0
0

void colorfunc check2
12 if(Check(9),.9,.1) .1 if(Check(9),.3,.1)
      cylmods.cal -s .9 -rx 90 -t -4 2.5 1.1
0
0
```

```
void WGMDfunc wgmd_mat2
17
      pyramid2 .08 0 0
      check2 .2 .02 .02
      void
      0 0 0
      . -t -4 2.5 1.1
0
9
      0 0 0
      0 0 0
      0 0 0

void colorfunc check3
10 if(Check(7),.9,.1) .1 if(Check(7),.3,.1)
      cylmods.cal -s .8 -t -4 2.5 1.1
0
0

check3 WGMDfunc wgmd_mat3
17
      void .03 .08*max(Pz,0) .03*max(Pz,0)
      void 0 0 0
      inherit
      0 0 1
      . -t -4 2.5 1.1
0
9
      0 0 0
      .4 .1 .1
      .3 .3 .4
```

WGMDfunc Examples (9-11)

```
{
      Unit cylinder fun
      for testing
}
posAngle(a):if(a,a,a+2

Cx = posAngle(atan2(Py
Cy = Pz;
Rx(n) = n/(2*PI) * Cx;
Ry(n) = n/(2*PI) * Cy;
Sx(n) = .5*(Rx(n) - Ry
Sy(n) = .5*(Rx(n) + Ry

                { Check
Check(n) = xor(frac(Rx

                { Pyram
Pyramid_dxy(n) = if(fr
Pyramid_dx(n) = -Py*Py
Pyramid_dy(n) = Px*Pyr
Pyramid_dz(n) = if(fra
```




```
void WGMDfunc wgmd_mat2
17
      pyramid2 .08 0 0
                  02 .02


                   1.1




                  eck3
                  ,.1) .1 if(Check(7),.3,.1)
                   -s .8 -t -4 2.5 1.1


                  gmd_mat3

                  8*max(Pz,0) .03*max(Pz,0)


                   1.1
```

Conclusions

- New *WGMDfunc* material provides a general, flexible way to model surface interactions
- Adds programmable variables for specularity, roughness parameters
- Enables multiple modifier paths - better for including spectral components

Could It Replace Other Materials?

- Maybe, but adds complexity where unneeded
- Also, *glass* and *dielectric* are simple materials that are not so simple
- No possibility of emission for light sources
- However, does provide a better translation target for general MGF materials (i.e., **mgf2rad -s**)