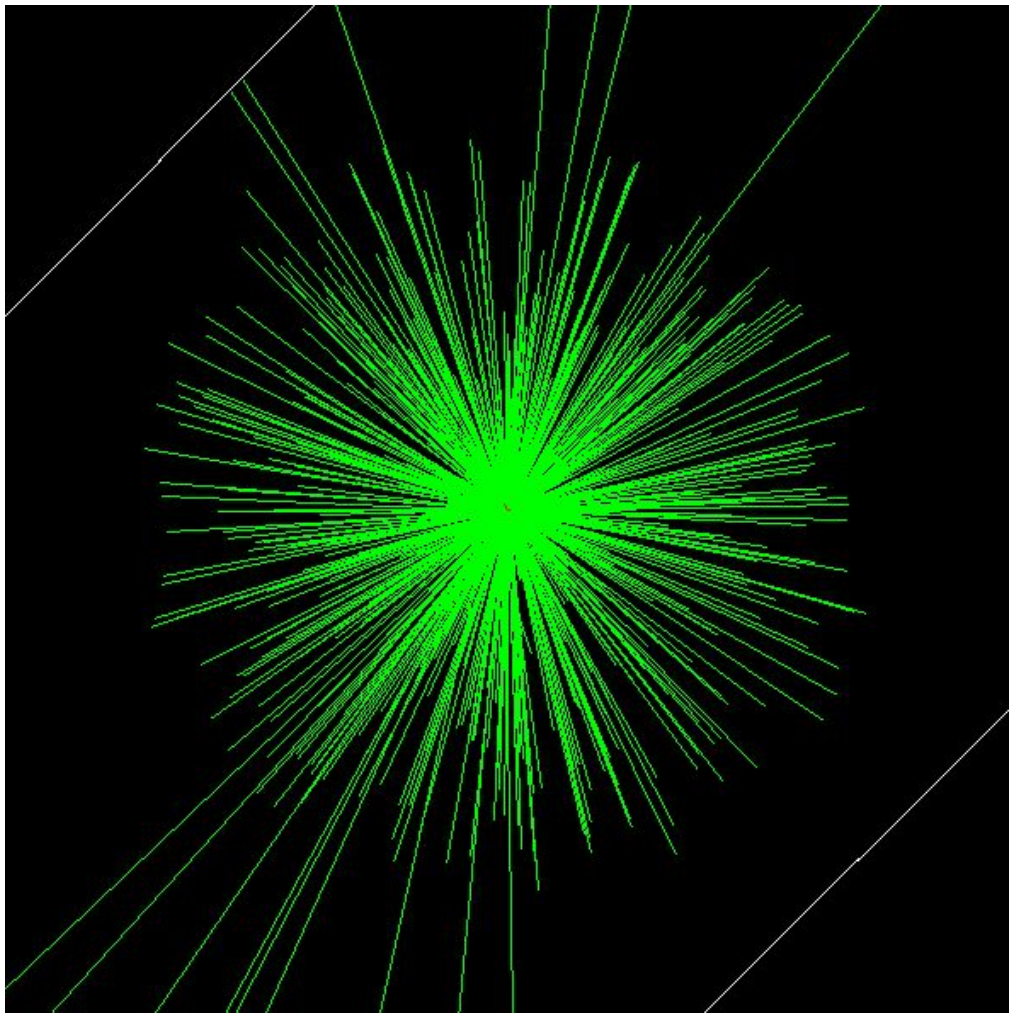
Introduction to GEANT4

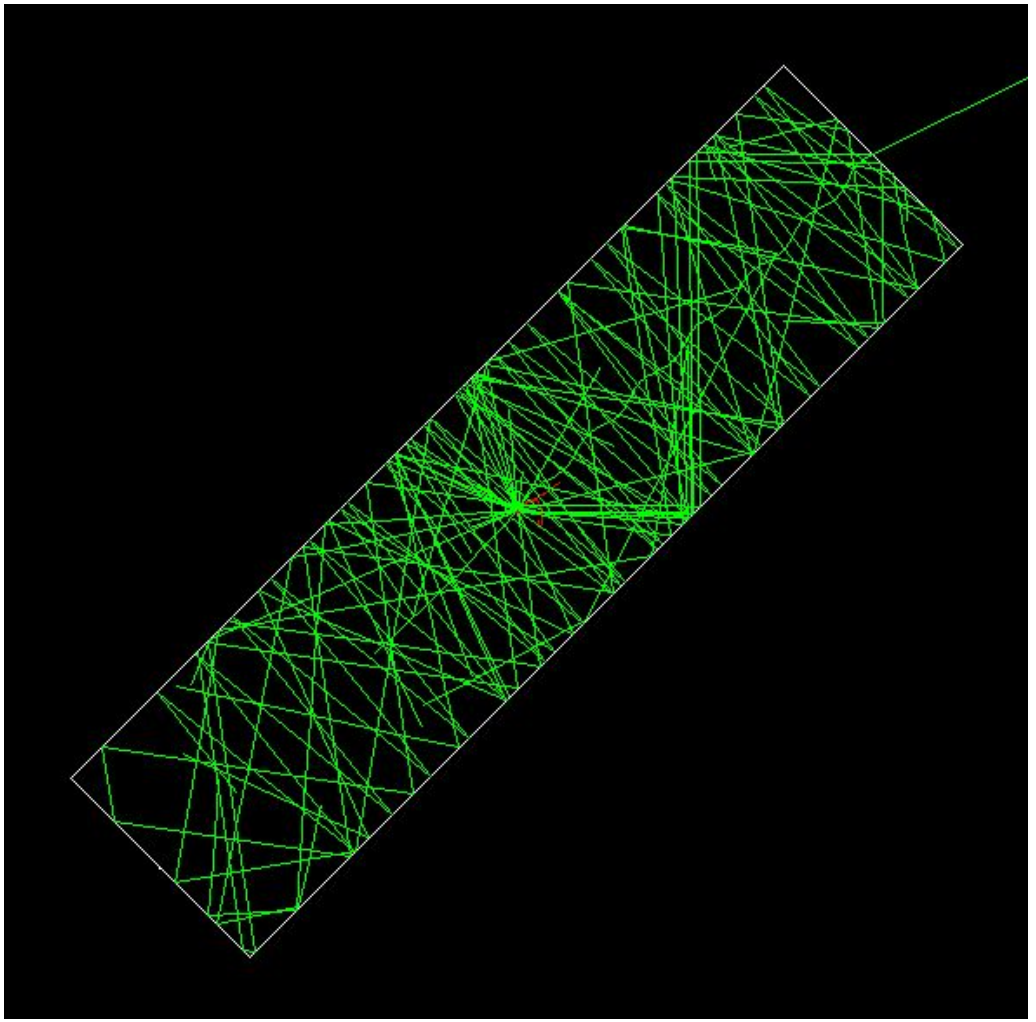
Laura Paulucci and Franciole Marinho

material adapted from M. Asai / P. Gumplinger / G. Santin / J. McCormick

Third UK-Latin America Workshop on Advanced Light Collection Systems

Simulating Optical Processes





Optical Photons

- Physically optical photons should be covered by the electromagnetic category, but
  - optical photon wavelength is >> atomic spacing
  - treated as waves → no smooth transition between optical and gamma particle classes

- G4OpticalPhoton: wave like nature of EM radiation → new particle type!
  - G4OpticalPhoton <=|=> G4Gamma

Optical Processes

- Optical Photon Production
- Cerenkov Process
- Scintillation Process
- Transition Radiation

- Processes affecting Optical Photons
- Refraction and Reflection
- Bulk Absorption
- Rayleigh Scattering

Processes undergone by optical photons

- Optical photons undergo:
  - bulk absorption
  - Rayleigh scattering
  - wavelength shifting
  - refraction and reflection at medium boundaries

- Geant4 keeps track of polarization
  - but not overall phase → no interference

Optical properties

- Properties table in G4Material: reflectivity, transmission efficiency, dielectric constants, surface properties
- Photon spectrum properties → G4Material: scintillation yield, time structure (fast, slow components)

```
const G4int NUMENTRIES = 32;

G4double photmom[NUMENTRIES]     = {2.034*eV, ……, 4.136*eV};
G4double rindex[NUMENTRIES]      = {1.3435, ……, 1.3608};
G4double absorption[NUMENTRIES] = {344.8*cm, ……, 1450.0*cm};

G4MaterialPropertiesTable *MPT = new G4MaterialPropertiesTable();
MPT -> AddProperty("RINDEX",photmom,rindex,NUMENTRIES};
MPT -> AddProperty("ABSLENGTH",photmom,absorption,NUMENTRIES};

aG4Material -> SetMaterialPropertiesTable(MPT);
```

as function of the photon's momentum

rindex → refraction index

absorption → absorption length

G4BoundaryProcess and Surfaces

- Reflection and refraction at physical volume surfaces
  - GLISUR (Geant3), UNIFIED (DETECT / TRIUMF) or DICHROIC models
- Ways to define surface:
  - G4LogicalBorderSurface(name, physVol1, physVol2, G4OpticalSurface)
  - G4LogicalSkinSurface

no partial refraction / reflection

- G4OpticalSurfaceType:
  - dielectric_metal, dielectric_dielectric, dielectric_dichroic
- G4OpticalSurfaceFinish:
  - polished, polishedfrontpainted, polishedbackpainted, ground, groundfrontpainted, groundbackpainted

Surface Parameters

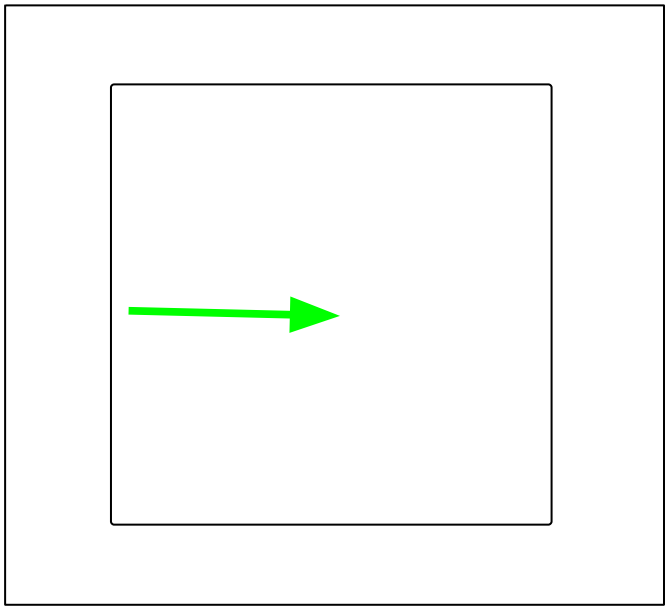
Parameters

- Finish
- Model
- Type
- RINDEX
- SPECULARLOBECONSTANT
- BACKSCATTERCONSTANT
- REFLECTIVITY
- EFFICIENCY

G4OpAbsorption

- Bulk absorption
  - uses photon attenuation length from material properties to get mean free path
  - photon is simply killed after a selected path length

```
G4double PhotonEnergy[nEntries] =
   {6.6*eV,6.8*eV,7.0*eV,7.2*eV,7.4*eV};
G4double AbsLength[nEntries] =
    {0.1*mm,0.3*mm,1.0*cm,1.0*m,10.0*m};

MPT->AddProperty("ABSLENGTH",PhotonEnergy,
AbsLength,NUMENTRIES};
```



Rayleigh Scattering

- Elastic scattering including polarization of initial and final photons
  - The scattered photon direction is perpendicular to the new photon polarization in such a way that the final direction, initial and final polarization are all in one plane

- The diff. cross section is proportional to $\cos^2(\alpha)$ where α is the angle between the initial and final photon polarization

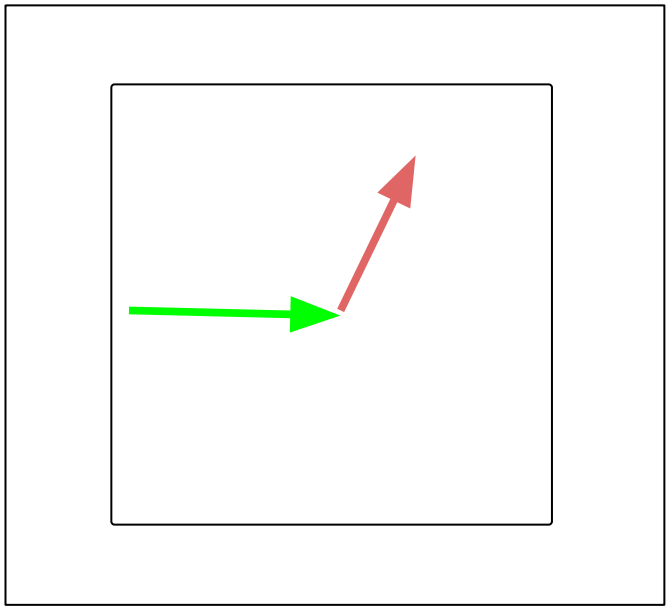
G4OpRayleigh

- Rayleigh scattering attenuation coefficient is calculated for “Water” material following the Einstein-Smoluchowski formula, but in all other cases it must be provided by the user:

```
MPT -> AddProperty(“RAYLEIGH”,PhotonEnergy,Scattering,NUMENTRIES};
```

Wavelengh Shifting

- initial photon is killed, one with new wavelength is created
- User must supply:
  - Absorption length as function of photon energy
  - Isotropic emission
  - With random linear polarization
  - Emission spectra parameters as function of energy
  - Possible time delay between absorption and re-emission



G4OpWLS

- In Detector Construction:

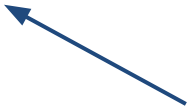
```
G4double RIndexFiber[nEntries] = {1.6, 1.6, 1.6, 1.6, 1.6};
G4double AbsFiber[nEntries] =    {0.1*mm,0.3*mm,1.0*cm,1.0*m,10.0*m};
G4double EmissionFiber[nEntries] = {0.0, 0.0, 0.5, 5.0, 10.0};

G4Material* WLSFiber;

G4MaterialPropertiesTable* MPTFiber = new
   G4MaterialPropertiesTable();

MPTFiber->AddProperty("RINDEX", PhotonEnergy,RIndexFiber,nEntries);
MPTFiber->AddProperty("WLSABSLENGTH",PhotonEnergy,AbsFiber,nEntries);
MPTFiber->AddProperty("WLSCOMPONENT",PhotonEnergy,EmissionFiber,nEntr
   ies);
MPTFiber->AddConstProperty ("WLSTIMECONSTANT", 0.5*ns);

WLSFiber->SetMaterialPropertiesTable(MPTFiber);
```

Let's shoot an optical photon

- Edit g4workshop/src/PrimaryGeneratorAction.cc
  - Make "opticalphoton" as primary particle
    G4ParticleTable::GetParticleTable()->FindParticle("opticalphoton");
  - Energy at ~eV range
    fParticleGun->SetParticleEnergy(9.68*eV);
  - Set polarization too (not necessary now but we keep it for later)
    G4ThreeVector polar = Polarisation(dir_vec);
    fParticleGun->SetParticlePolarization(polar);

- Optical properties of materials need to be included as well
  edit g4workshop/src/DetectorConstruction.cc and on DefineMaterials() function include:

```
const G4int nEntries = 5;
G4double energy[nEntries] = { 1.0*eV, 3.*eV, 6.*eV, 9.*eV, 12.*eV };
G4double n_lAr[nEntries] =  { 1.5, 1.5, 1.5, 1.5, 1.5};
```



These are dummy values

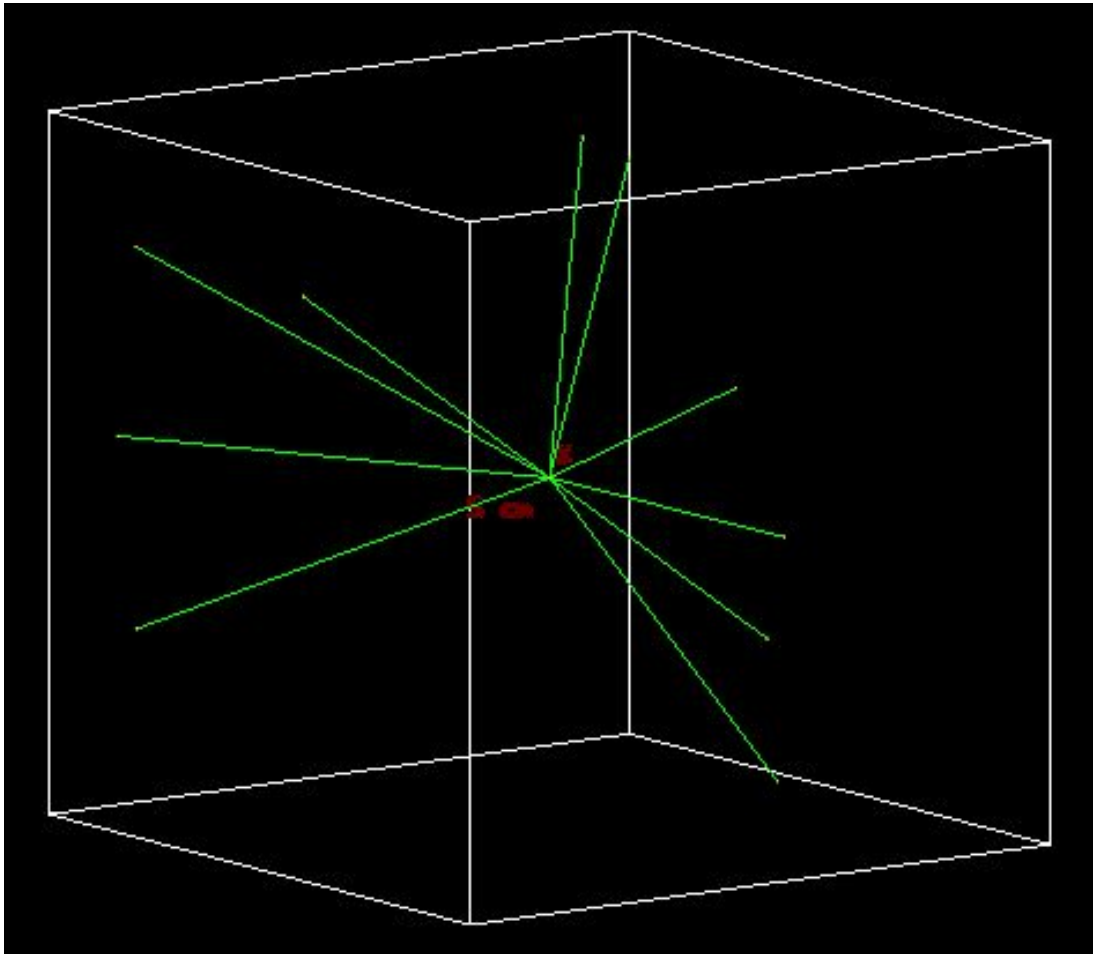
Let's shoot an optical photon

- Also on DefineMaterials() function include:

```
G4MaterialPropertiesTable* lAr_pt = new G4MaterialPropertiesTable();
lAr_pt->AddProperty("RINDEX", energy, n_lAr, nEntries);
env_mat->SetMaterialPropertiesTable(lAr_pt);
```

- Go back to terminal on "g4workshop_build/" compile and do "./g4workshop"

Are we happy
w/ this picture?



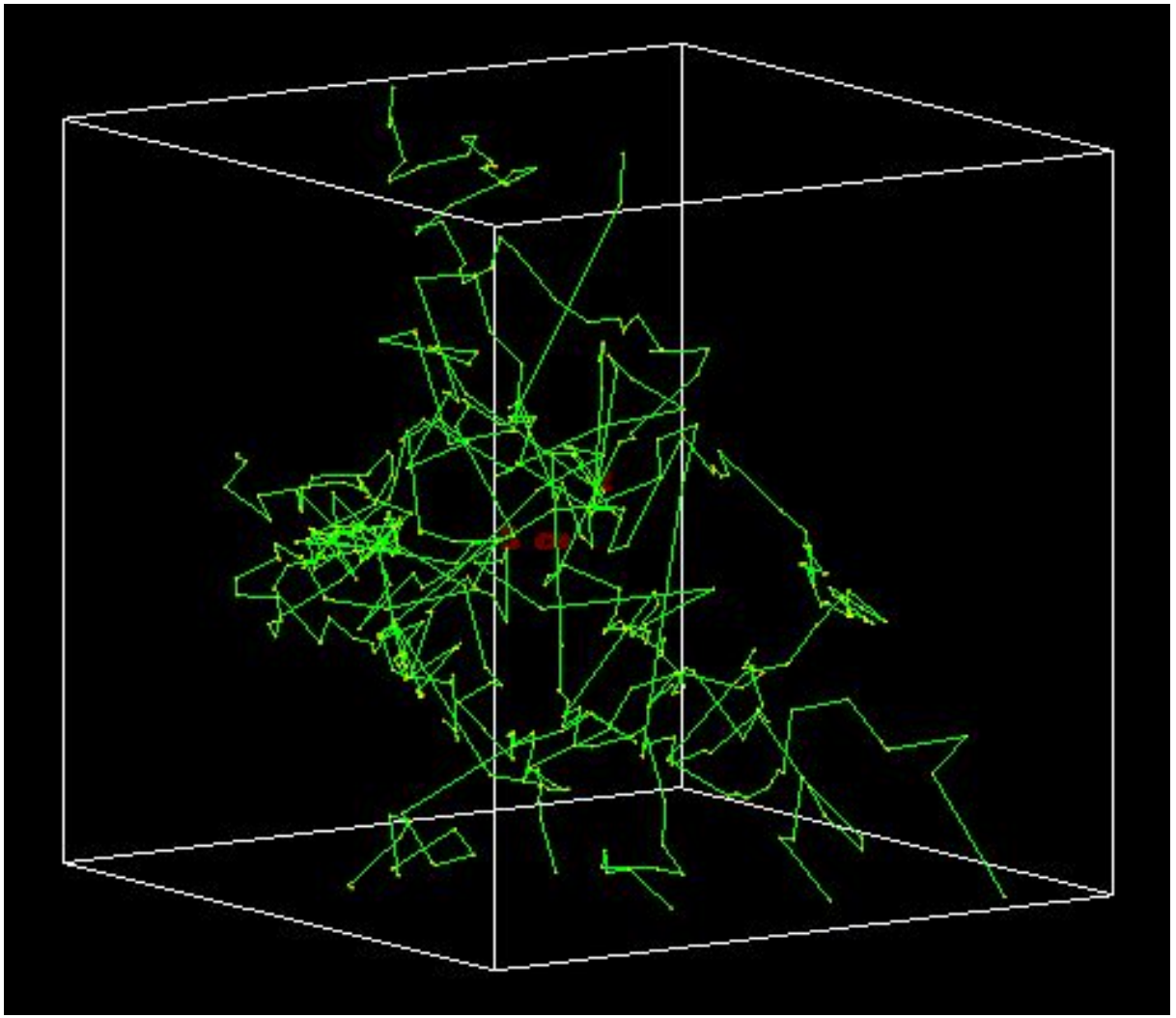
Let’s shoot an optical photon

- On DefineMaterials() function update code to:

```
const G4int nEntries = 5;
G4double energy[nEntries] = { 1.*eV, 3.*eV, 6.*eV, 9.*eV, 12.*eV };
G4double n_lAr[nEntries] =  { 1.5, 1.5, 1.5, 1.5, 1.5};
G4double lR_lAr[nEntries] = {0.55*m,  0.55*m,  0.55*m,  0.55*m, 0.55*m};

G4MaterialPropertiesTable* lAr_pt = new G4MaterialPropertiesTable();
lAr_pt->AddProperty("RINDEX", energy, n_lAr, nEntries);
lAr_pt->AddProperty("RAYLEIGH", energy, lR_lAr, nEntries);
env_mat->SetMaterialPropertiesTable(lAr_pt);
```

- Go back to terminal on “g4workshop_build/” compile and do “./g4workshop”

Let’s shoot an optical photon



Wavelength shifters

- Now let's shoot optical photons on something
- Edit g4workshop/include/DetectorConstruction.hh
  - Insert TPB as a class member material

```
G4Material*   fTPB;

G4VPhysicalVolume*  tpbPhys;
G4LogicalVolume*    tpbLogic;
G4Box*              tpbSolid;
```

- Edit g4workshop/src/DetectorConstruction.cc
  On DefineMaterials() function include the following:

```
//tpb material instance
G4Material* tpb_mat = new G4Material("tpb_mat",1.079*g/cm3,2);
G4Element* H = new G4Element ("Hydrogen","H",1.,1.01*g/mole);
G4Element* C = new G4Element ("Carbon","C",6.,12.01*g/mole);
tpb_mat->AddElement (C, 28);
tpb_mat->AddElement (H, 22);
```

Wavelength shifters

- Also on DefineMaterials() function include the following:

```
//TPB optical info
const int tpb_entries = 10;

G4double tpb_e_energy[tpb_entries] = {3.125*eV, 3*eV, 2.875*eV, 2.75*eV, 2.625*eV, 2.5*eV,
2.375*eV, 2.25*eV, 2.125*eV, 2*eV};
G4double tpb_e[tpb_entries] = {0.120479, 0.718092, 1.00175, 0.640985, 0.389969, 0.157005,
0.0509217, 0.0080711, 0.000595335, 2.04258e-05};

G4double tpb_a_energy[tpb_entries] = {10.9729*eV, 7.07431*eV, 5.22486*eV, 4.2614*eV,
3.71444*eV, 3.36*eV, 3.16*eV, 2.96*eV, 2.9064*eV, 2.901*eV};
G4double tpb_l[tpb_entries] = {0.0594703*um, 0.0594703*um, 0.0573079*um, 0.106937*um,
0.0323873*um, 0.0608373*um, 0.581187*um, 13.2291*um, 87.4751*um, 950.347*um};
```

Wavelength shifters

- Also on DefineMaterials() function include the following:

```
G4MaterialPropertiesTable* tpb_pt = new G4MaterialPropertiesTable();
tpb_pt->AddProperty("RINDEX", energy, n_lAr, nEntries);
tpb_pt->AddProperty("WLSABSLENGTH", tpb_a_energy, tpb_l, tpb_entries);
tpb_pt->AddProperty("WLSCOMPONENT", tpb_e_energy, tpb_e, tpb_entries);
tpb_pt->AddConstProperty("WLSTIMECONSTANT", 5*ns);
tpb_mat->SetMaterialPropertiesTable(tpb_pt);

fTPB = tpb_mat;
```

Wavelength shifters

- Now on ConstructLine():
  - reduce LAr volume size to 10cm
- Create a thin layer volume (2cm x 2cm x 2μm)

```
tpbSolid = new G4Box("tpbSolid",1.0*cm,1.0*cm,1*um);
tpbLogic = new G4LogicalVolume(tpbSolid,fTPB,"tpbSolid");
tpbPhys = new G4PVPlacement(0,G4ThreeVector(0,0,0),"tpbSolid", tpbLogic,
fPhysiWorld, false, 0);
```

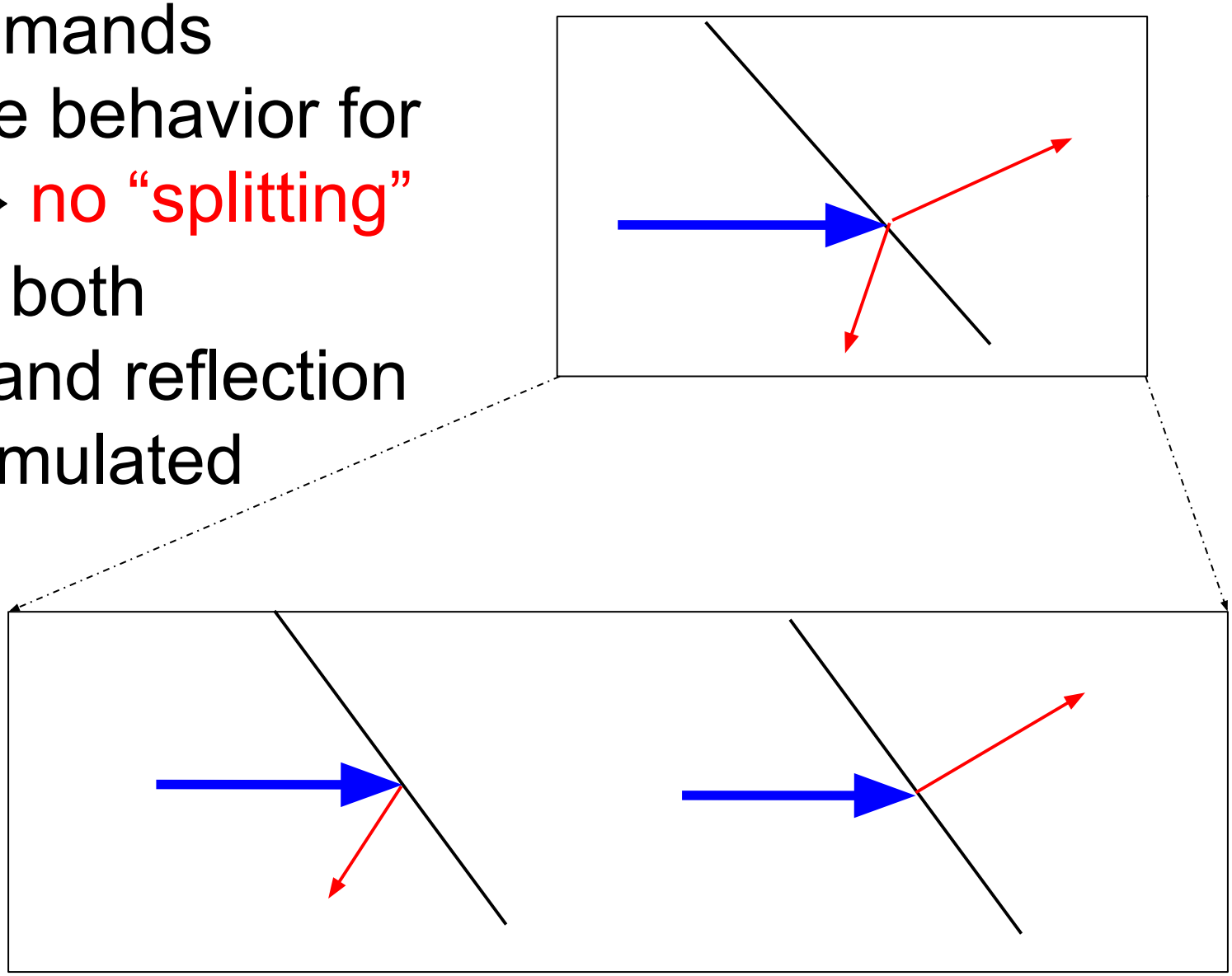
- Now change g4workshop/src/PrimaryGeneratorAction.cc accordingly

```
z0=1.0*cm;
G4ThreeVector dir_vec (0.,0.,-1.);
```

- Go back to terminal on “g4workshop_build/” compile and do “./g4workshop”

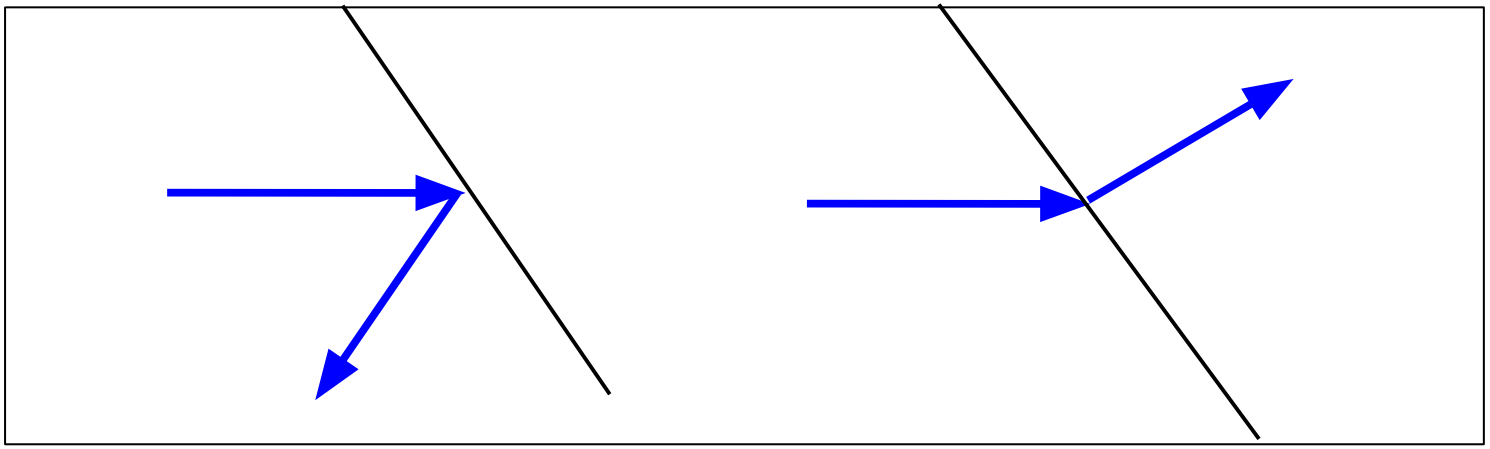
Boundary Interactions

- Geant4 demands particle-like behavior for tracking → no “splitting”
- event with both refraction and reflection must be simulated by at least two events



Boundary Interactions

- Handled by G4OpBoundaryProcess
  - Refraction
  - Reflection
- User must supply surface properties using G4OpticalSurface models
- Boundary properties
  - dielectric-dielectric
  - dielectric-metal
  - dichroic
- Surface properties:
  - polished
  - ground
  - front- or back-painted, ...



Surface Concept

Split into two classes:

- Geometrical class: G4LogicalSurface

  (in the **geometry** category)

  – pointers to the relevant physical or logical volumes
  – pointer to a G4OpticalSurface

  Are stored in a table and need:

  – an ordered pair of physical volumes touching at the surface [G4LogicalBorderSurface]
    - in principle allows for different properties depending on which direction the photon arrives

Surface Concept

- Geometrical class: G4LogicalSurface

Are stored in a table and need:

- a logical volume entirely surrounded by this surface [G4LogicalSkinSurface]
  - useful when the volume is coded by a reflector and placed into many volumes
  - limitation: only one and the same optical property for all the enclosed volume's sides)

Surface Concept

Split into two classes:

- Physical class: G4OpticalSurface
  (in the **materials** category)
  keeps information about the physical properties of the surface itself

Optical Surface Types

- Dielectric - Dielectric
  - Depends on
    - photon's wavelength, angle of incidence, (linear) polarization
    - refractive index on both sides of the boundary

a) total internal reflected
b) Fresnel refracted
c) Fresnel reflected

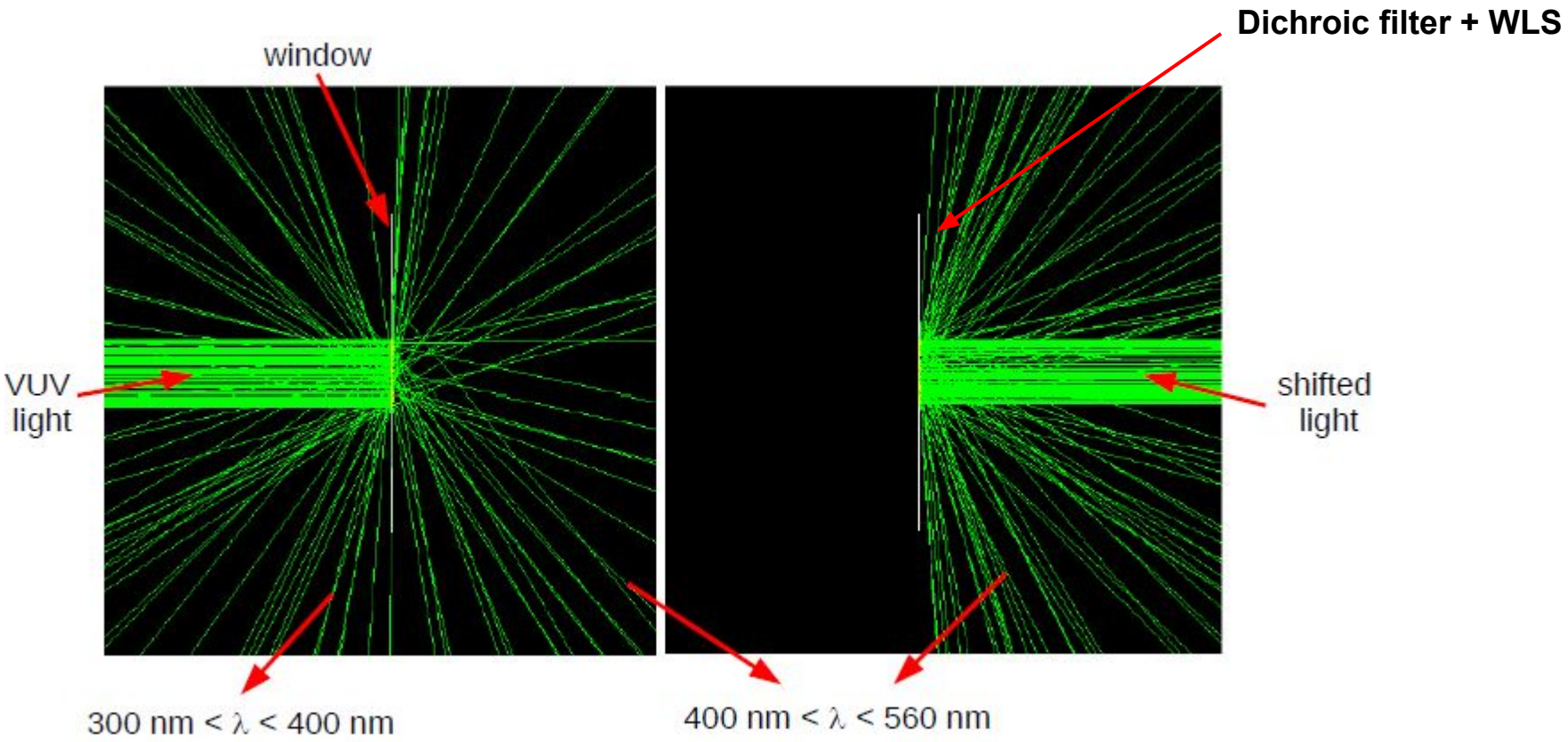
Optical Surface Types

- Dielectric – Metal
  - The photon cannot be transmitted

a) absorbed (detected), with probability estimated according to the table provided by the user
b) reflected

Optical Surface Types

- Dielectric – Dichroic
  - The photon is transmitted or reflected according to wavelength and angle of incidence



Surface Effects

- POLISHED: surface between two bodies is perfectly polished, the normal used by the G4BoundaryProcess is the normal to the surface defined by:
  - the daughter solid entered; or else
  - the solid being left behind

- GROUND: The incidence of a photon upon a rough surface requires choosing the angle, a, between a 'micro-facet' normal and that of the average surface

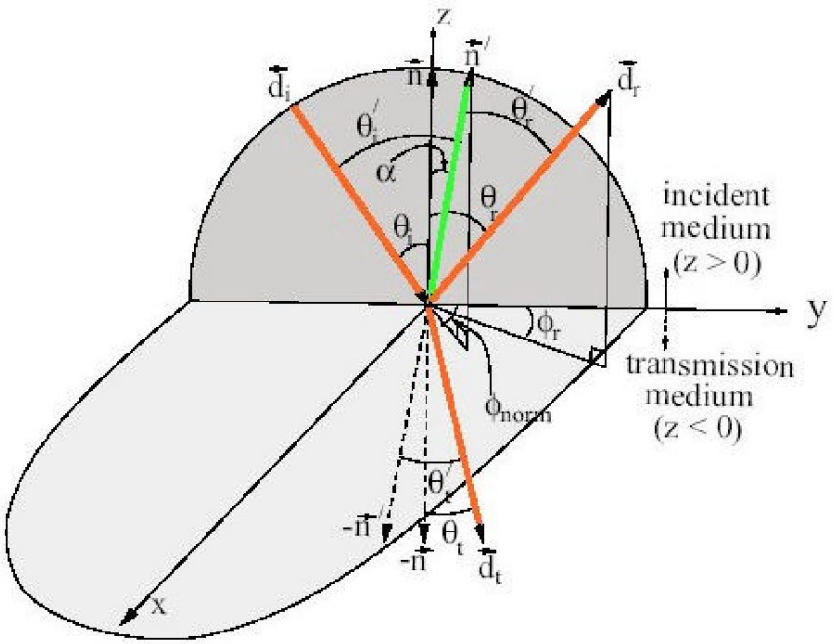
Surface Effects

- UNIFIED model: probability of micro-facet normals that populates the annulus of solid angle $\sin(\alpha)d\alpha$ will be proportional to a gaussian of SigmaAlpha:
  theOpSurface -> SetSigmaAlpha(0.1); [rad]

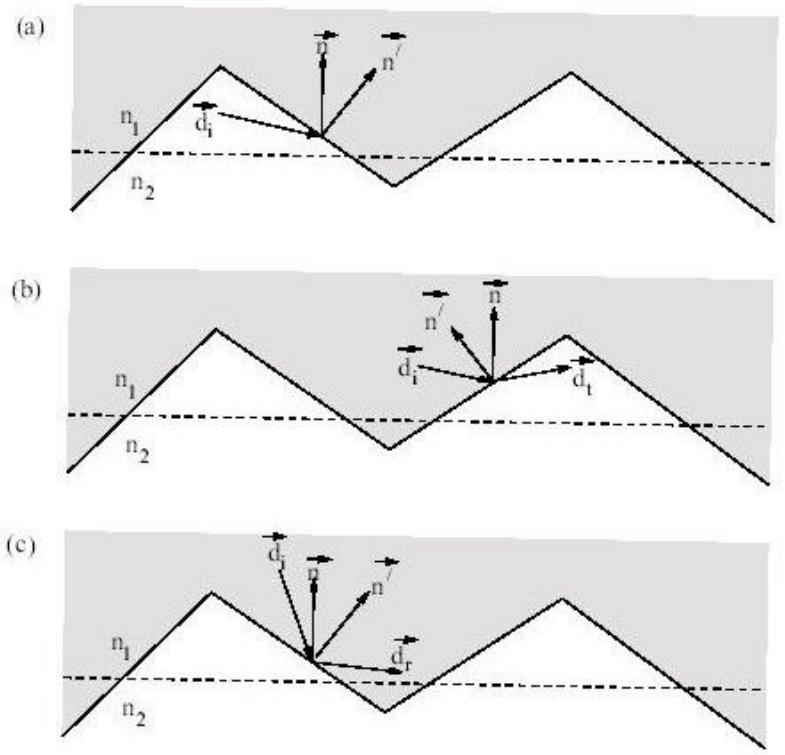
- In the GLISUR model this is indicated by the value of polish; when it is <1, then a random point is generated in a sphere of radius (1-polish), and the corresponding vector is added to the normal. The value 0 means maximum roughness with effective plane of reflection distributed as **cos(α)**.
  theOpSurface -> SetPolish(0.0);
- The 'facet normal' is accepted if the refracted wave is still inside the original volume.

Microfacets

- Assumption: a rough surface is a collection of ‘microfacets’

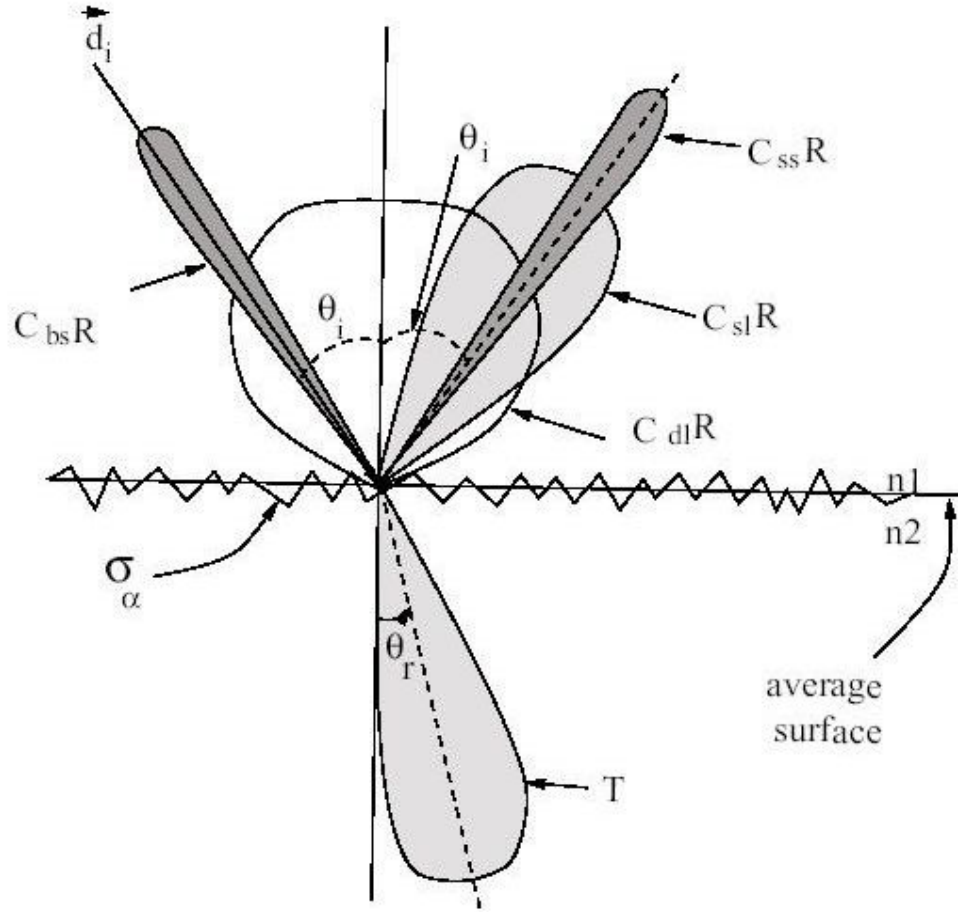


Coordinate system used for ***ground*** surfaces along with the definition of geometrical parameters.



Special cases handled by the UNIFIED model:
(a) when the incident photon does not aim toward the local micro-facet; or when the transmitted (b) or reflected (c) photon heads in the wrong direction with respect to the average surface normal.

- In cases (b) and (c), multiple interactions with the boundary are possible within the process itself and without the need for relocation by the G4Navigator.

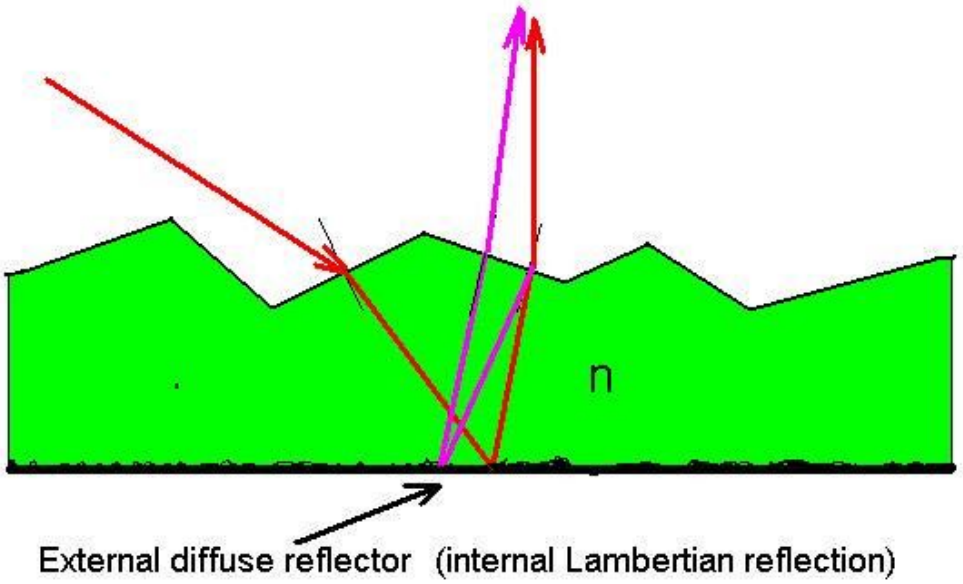


Polar plot of the radiant intensity in the UNIFIED model

- Csl: Reflection prob. about the normal of a micro facet
- Css: Reflection prob. about the average surface normal
- Cdl: Prob. of internal Lambertian reflection
- Cbs: Prob. of reflection within a deep grove with the ultimate result of exact back scattering.

The G4OpticalSurface also has a pointer to a G4MaterialPropertiesTable

- If surface is painted, wrapped, or has a cladding: table may include the thin layer's index of refraction
- This allows the simulation of boundary effects both
  - at the intersection between the medium and the surface layer and
  - at the far side of the thin layer



all within the process itself and without invoking the G4Navigator

  - the thin layer does not have to be defined as a G4 tracking volume

Example

```
G4OpticalSurface * OpWaterSurface = new G4OpticalSurface("WaterSurface");
OpWaterSurface -> SetModel(unified);
OpWaterSurface -> SetType(dielectric_dielectric);
OpWaterSurface -> SetFinish(groundbackpainted);

Const G4int NUM = 2;

G4double pp[NUM] = {2.038*eV, 4.144*eV};
G4double specularlobe[NUM] = {0.3, 0.3};
G4double specularspike[NUM] = {0.2, 0.2};
G4double backscatter[NUM] = {0.1, 0.1};
G4double rindex[NUM] = {1.35, 1.40};
G4double reflectivity[NUM] = {0.3, 0.5};
G4double efficiency[NUM] = {0.8, 1.0};
G4MaterialPropertiesTable *SMPT = new G4MaterialPropertiesTable();

SMPT -> AddProperty("RINDEX", pp, rindex, NUM);
SMPT -> AddProperty("SPECULARLOBECONSTANT",pp,specularlobe,NUM);
SMPT -> AddProperty("SPECULARSPIKECONSTANT",pp,specularspike,NUM);
SMPT -> AddProperty("BACKSCATTERCONSTANT",pp,backscatter,NUM);
SMPT -> AddProperty("REFLECTIVITY",pp,reflectivity,NUM);
SMPT -> AddProperty("EFFICIENCY",pp,efficiency,NUM);

OpWaterSurface -> SetMaterialPropertiesTable(SMPT);
```

Reflection and transmission

- Edit g4workshop/src/DetectorConstruction.cc
  On DefineMaterials() function do the following:

```
G4double n_tpb[nEntries] =  {1.35, 1.35, 1.35, 1.35, 1.35};

tpb_pt->AddProperty("RINDEX", energy, n_tpb, nEntries);
```

  and comment all the WLS properties out

- g4workshop/src/PrimaryGeneratorAction.cc

```
theta = CLHEP::pi/4;
y0=-sin(theta)*cm; z0=cos(theta)*cm;
G4ThreeVector dir_vec (0.,sin(theta),-cos(theta));
```

- Go back to terminal on "g4workshop_build/" compile and do "./g4workshop"
- If you got to this point try to implement a different reflection model

G4OpticalSurface

- Set the simulation model used by the boundary process:
  - GLISUR-Model: original G3 model
  - UNIFIED-Model: adopted from DETECT (TRIUMF)

```
enum G4OpticalSurfaceModel {glisur, unified};
```

- Set the type of interface:

```
enum G4OpticalSurfaceType { dielectric_metal,
   dielectric_dielectric};
```

G4OpticalSurface

- Set the surface finish:

```
enum G4OpticalSurfaceFinish {
  polished,      // smooth perfectly polished surface
  polishedfrontpainted,    // polished top-layer paint
  polishedbackpainted, // polished (back) paint/foil
  ground,        // rough surface
  groundfrontpainted,   // rough top-layer paint
  groundbackpainted     // rough (back) paint/foil
};
```