Re-supporting Charge Reflection (G4CMP-515)

Michael Kelsey, Wade Lamberson, David Toback
Texas A&M University

SuperCDMS G4DMC Meeting, 15 Oct 2025
G4CMP Technical Meeting, 16 Oct 2025

Outline

- Physics Expectations
- Existing Implementation
- Bug Report (G4CMP-503)
- Kinematics failures
- Code Modifications, API Changes
- Proposal: Staged Implementation
- Comments, Community Input?

How Should Charges Reflect At Surfaces?

Two sources of consistent information (details in backup slides)

- Wade Lamberson (A&M student) searched literature for information
- Kelsey contacted Jesse Lutz (G4CMP Consortium) at Sandia

At polished, crystal-symmetry surfaces, charges should reflect specularly

- Coherent scattering, should preserve valley assignment

At rough or non-symmetry surfaces, charges should scatter diffusely

- Incoherent scattering, valley should be reassigned (nearest to direction)

Charges should transmit through boundary where ohmic contacts are involved

Real surfaces will likely have a mix of both specular and diffuse scattering

Existing Implementation

`G4CMPDriftBoundaryProcess` handles charge scattering at boundaries

- Specular reflection (only) was implemented in 2014
- For electrons, velocity vector computed from momentum, then reflected
- Deactivated (default settings limited to zero bounces) in 2017

Reflection code hasn't been touched, or used, since then

Exercising this was not included in validation of new electron transport code (G4CMP-276, **V09-00-00** release)

G4CMP-503 Bug Report

Eric Mascot found that charge reflection code could not be successfully reactivated, which we reproduced independently at A&M

Allowing multiple reflections (`/gcmp/chargeBounces N`) has no effect

- **`DriftBoundaryProcess`** absorption decision has incorrect logic
- Overrode implementation in base **`G4CMPBoundaryUtils`** class
- Simple, one line fix (see G4CMP-503 initial comments)

After fixing, charge reflection code substantially failed

Kinematics Failures

Fixing **`DoAbsorption()`** worked to re-enable **`/g4cmp/chargeBounces`**

All reflected electrons were killed with error messages

```
Killing track inconsistent position (nan,nan,nan)
```

Specular reflection in **`DriftBounaryProcess`** no longer works

- Uses **`MapPtoV_el()`** to convert momentum to electron velocity ***v***
- Specular reflection of velocity: $\boldsymbol{v}_{\mathbf{refl}} = \boldsymbol{v} - 2(\boldsymbol{v}\cdot\boldsymbol{n})\boldsymbol{n}$ (***n*** is surface normal)
- Converts new velocity (in same valley) back to momentum; **this fails**

Also got one instance in 100 tracks of electric field transport failure

- **`G4ChordFinder`**, 1000 messages complaining of O(1e-75) step lengths

Code Modifications Needed/Planned

Repair **`DriftBoundaryProcess::DoReflectionElectron()`** algorithm

- Replace use of velocity with wavevector, ensure this eliminates NaNs
- Factor algorithm out to new S**`pecularReflectElectron()`** function

Rename existing **`DoReflectionHole()`** algorithm to Do**`SpecularHole()`**

Implement Do**`Diffuse[Electron/Hole]()`** f for Lambertian scattering.

- Include iteration to ensure inward propagation (for curved surfaces).
- For electrons, reassign valley to correspond to new direction

Support diffuse vs. specular reflection probabilities for charges

Possible API Changes

Add **`qSpecProb`** argument to **`G4CMPSurfaceProperty`** to set specular vs. diffuse reflection probability

- This would change constructor signature, affects all user applications
- Will implement as separate constructor for backward compatibility

Organize reflection-related functions to have parallel structure and names in both phonon and charge processes, or unify in **`G4CMPBoundaryUtils`**

- Move Lambertian functions from **`G4CMPUtils`** to **`G4CMPBoundaryUtils`**,
- Rename function with "while (!Inward)" to **`DoDiffuseReflection()`**

Rename **`PhononVelocityIsInward()`** to **`TransportIsInward()`**, override in **`DriftBoundaryProcess`** to get charge direction instead of phonon

Quick Patch on G4CMP-503 Branch/Tag

`DriftBoundaryProcess::DoAbsoprtion()` fixed to use OR instead of AND

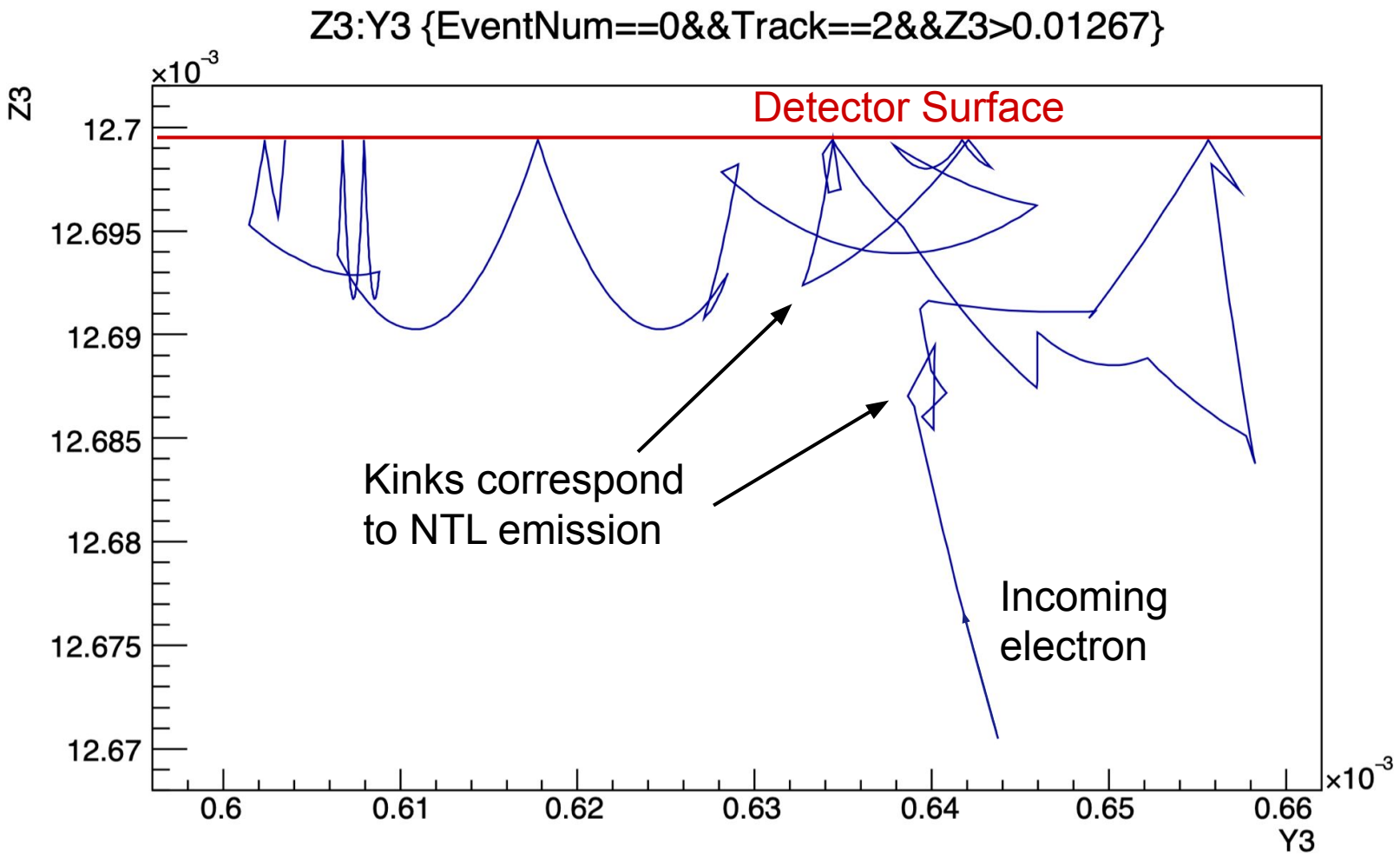
Implemented trial diffuse scattering for electrons, using momentum direction

Temporarily commented out specular reflection algorithm

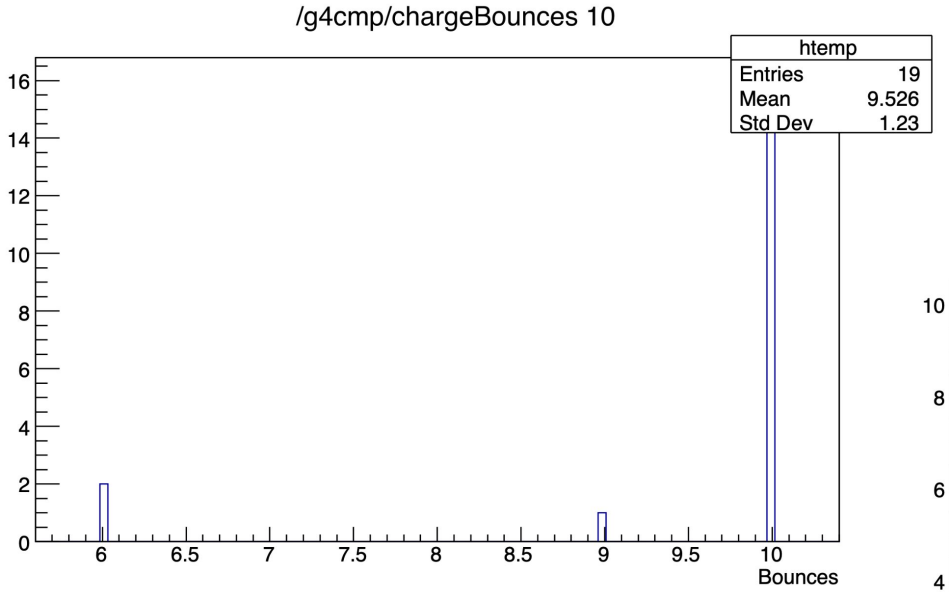
Modified **`MaximumReflections()`** to have associated function in place of **`DoSimpleKill()`**: DriftBoundary overrides function to allow recombination

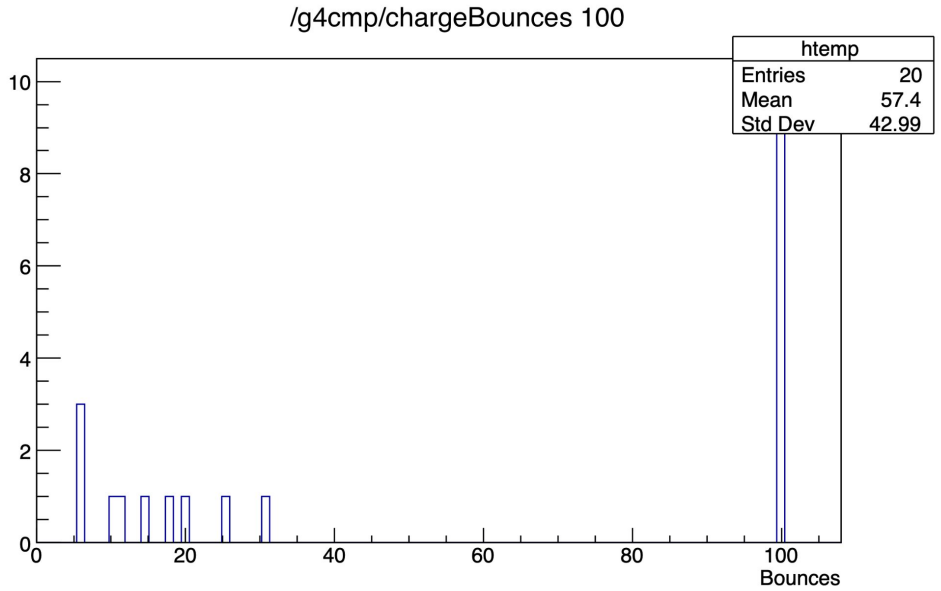
Improved recombination process to include track kinetic energy in conversion

Getting reasonble results, with distribution of actual number of bounces per track from zero (immediate absorption) up to maximum limit set


Z3:Y3 {EventNum==0&&Track==2&&Z3>0.01267}
Z3
×10⁻³
12.7
12.695
12.69
12.685
12.68
12.675
12.67
Detector Surface
Kinks correspond
to NTL emission
Incoming
electron
0.6
0.61
0.62
0.63
0.64
0.65
0.66
×10⁻³
Y3

Number of Bounces when Charge Absorbed





Staged Implementation

Simple patch to electron reflection code, deployed in new **V09-09-00** release

- Restores intended behaviour prior to **V09-00-00** electron transport changes
- Reflections will be entirely diffuse, but will not cause errors

Electron specular reflection should be fixed next

- Factoring into diffuse and specular functions
- Could undo the patch above and make specular the default, or not

SurfaceProperty API changes can be done in backward compatible way

Reorganizing reflection utility functions should wait until after QP tracking finished

- Include in a future **V10-xx-00** release

Associated JIRA tickets and subtasks are listed in backup slides

Comments? Input from Community?

Backup Slides

Debugging Output from DriftBoundaryProcess

```
G4WT0 > BU::Apply: Reflection
G4WT0 > G4CMPChargeBoundary: Track reflected
G4WT0 > G4CMPChargeBoundary: Electron reflected
G4WT0 >   PreStep volume: Zip @ (-0.11839980663398,-0.19898902695724,-12.699147114449)
 PostStep volume: Zip/MaskEnvBottom @ (-0.11832890542117,-0.19869833533439,-12.6994)
G4WT0 > CheckStepBoundary: in prePV (Zip) frame
  preStep @ (-0.11839980663398,-0.19898902695724,-12.699147114449)
 postStep @ (-0.11832890542117,-0.19869833533439,-12.6994)
G4WT0 >  Is postStep location on surface of preStep Volume? surface
G4WT0 >  Old momentum direction (0.18034993669572,0.74143676549097,-0.64633228537041)
 Old velocity direction (0.18034993669572,0.74143676549097,-0.64633228537041)
G4WT0 >  New velocity direction (0.18034993669572,0.74143676549097,0.64633228537041)
G4WT0 >  New momentum direction (nan,nan,nan)
G4WT0 >  Cross-check new v dir  (nan,nan,nan)
[...]
G4WT0 > G4CMPChargeBoundary ERROR: fGeomBoundary status set, but pre- and post-step
volumes are identical!
G4WT0 > G4CMPChargeBoundary ERROR: fGeomBoundary status set, but pre- and post-step
volumes are identical!
```

Error Message (×1000) from Geant4

```
-------- WWWW ------- G4Exception-START -------- WWWW -------
*** G4Exception : GeomField0003
      issued by : G4ChordFinder::FindNextChord()
Exceeded maximum number of trials= 75
Current sagita dist= nan
Max sagita dist= 0.25
Step sizes (actual and proposed):
Last trial =          6.97973e-76
Next trial =          6.97973e-77
Proposed for chord = 1.39595e-75
*** This is just a warning message. ***
-------- WWWW -------- G4Exception-END --------- WWWW -------
```

Three Questions About Charge Bounces

Is reflecting the velocity vector the right thing to do?

Is specular reflection for the charges the right thing to do at all?

- If they reach the surface and don't get trapped, should they "reflect", or just scatter randomly?

Should we be reassigning the valley?

- If the charge direction is getting significantly flipped, it will be far outside its assigned valley (indeed, it will be in the opposite hemisphere!). We don't do that reassignment for NTL, but the latter is a smaller deviation (less than 90 degrees).

Wade's Research (I)

Is reflecting the velocity vector the right thing to do?

- Reflecting the normal component of the velocity vector is the standard approach for modeling specular reflection of charge carriers at a surface. This is supported by the literature on surface scattering in semiconductors.
- "Electrical Properties of Polycrystalline Semiconductor Thin Films" by L.L. Kazmerski (National Renewable Energy Laboratory, 1980)
- "During the scattering process, the carriers (electrons or holes) have only their velocity component perpendicular to the surface reversed and their energy remains constant. Since no losses occur, there is no effect on conductivity. The surface represents a perfect reflector and the scattering is elastic."
- Page 8, Section 3.2, Transport in Thin Crystalline Films, Subsection on Surface Scattering

Wade's Research (II)

Is specular reflection for the charges the right thing to do at all?

- Specular reflection is appropriate for ideal or smooth surfaces, where charges reflect elastically without energy loss, reversing only the perpendicular velocity component. This is a common modeling choice in semiconductor transport, especially for polished or passivated surfaces in cryogenic detectors. However, real surfaces may involve a mix of specular and diffuse (random) scattering, depending on roughness and fabrication.
- "Electrical Properties of Polycrystalline Semiconductor Thin Films" by L.L. Kazmerski (National Renewable Energy Laboratory, 1980)
- "Electron scattering at single crystal Cu surfaces" by T. Miller et al. (Thin Solid Films, Volume 518, Issue 3, 2009
- "Monte Carlo Transport and Heat Generation in Semiconductors" by E. Pop (in "Springer Handbook of Semiconductor Devices," edited by M. Rudan et al., 2021)

Wade's Research (III)

Should we be reassigning the valley?

- Yes, valley reassignment should occur after reflection if the new direction places the charge outside the angular range of its current valley (e.g., in the opposite hemisphere), to maintain consistency with the valley-specific effective mass tensor and oblique propagation.
- "Analytic band Monte Carlo model for electron transport in Si: Including acoustic and optical phonon dispersion and intervalley scattering" by E. Pop et al. (Journal of Applied Physics, Volume 96, Issue 9, 2004)

E-Mail from Jesse Lutz

Carriers which undergo collisions with a generic boundary should indeed be reflected. There are special cases to look out for (e.g., ohmic contacts), but, in general, carriers should not transmit into vacuum.

Interfaces are trickier. In a typical spin-qubit stack, for example, there are rough interfaces and polished interfaces. On polished interfaces, I would expect specular reflection. This is a coherent process and therefore the electron will retain its valley. If the reflection involves scattering on rough surfaces/interfaces, then the valley should be randomized.

If you can reflect the wave vector efficiently, do that. It will be more accurate in low-dimensional volumes and other confined areas (corners).

JIRA Ticket Structure (So Far)

G4CMP-515 – Top-level ticket to resolve charge reflection problems

- ~~G4CMP-503~~ – Used to roll out preliminary patch (diffuse-only electron scatter)
- G4CMP-511 – Move Lambertian functions into G4CMPBoundaryUtils
- G4CMP-513 – Provide specular and diffuse reflection functions for charges
- G4CMP-514 – Add charge specular reflection to G4CMPSurfaceProperty

Once we decide on the implementation sequence, some of the subtasks may be promoted to top-level tickets