Implementation of SVT Sensor Noise

Joshua Sobajic[1,2], Beatrice Liang-Gilman[1,2], Shujie Li[1,2]

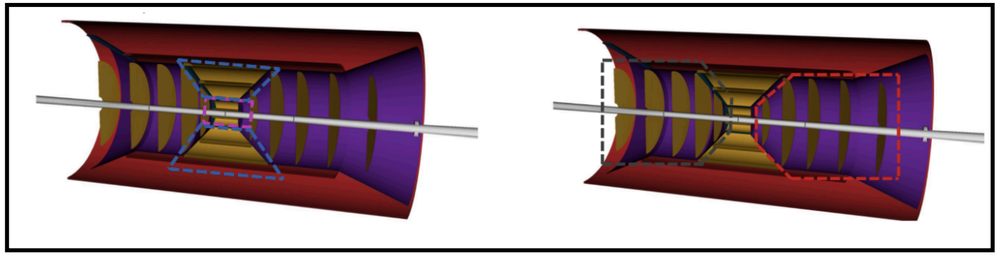
Joint Track and Vertex Reconstruction and Tracking WG Meeting, May 21, 2026

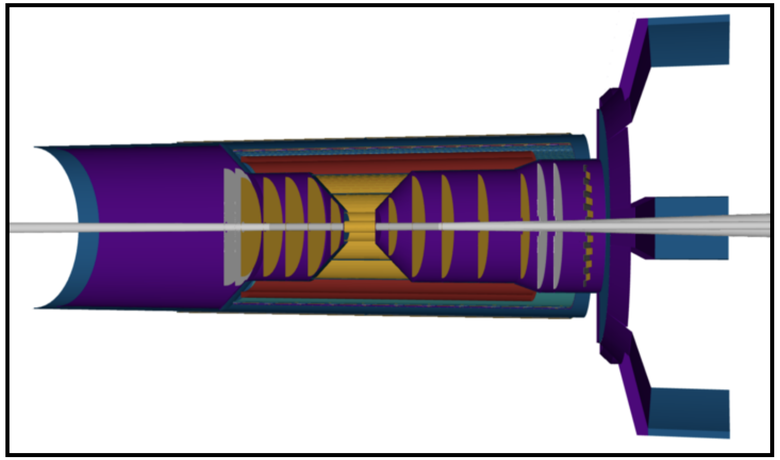
[1]Physics Department, University of California, Berkeley, [2]Nuclear Science Division, LBNL

Background + Motivation

- We want to generate noise hits across the various SVT layers with a uniform and random distribution.
- Previous work was done by Minjung Kim who implemented the RandomNoise algorithm for the silicon tracker subsystems (BVTX/BTRK/ECTRK).
- Sampled fake-hit rate: FHR $< 5 \times 10^{-7}$ per event per pixel.
- Fake hits/event/collection: FHR $\times$ total pixels. Noise hits $\propto$ surface area of SVT layer.
- Pixel sizes: $20 \times 20 \mu\mathrm{m}^2$
- *The following changes are implemented in PR #2670

	Inner Barrel	Outer Barrel	Endcaps
Est. total pixels	8.65E+08	7.83E+09	1.18E+10
Fake hits/event	4.33E+02	3.92E+03	5.91E+03



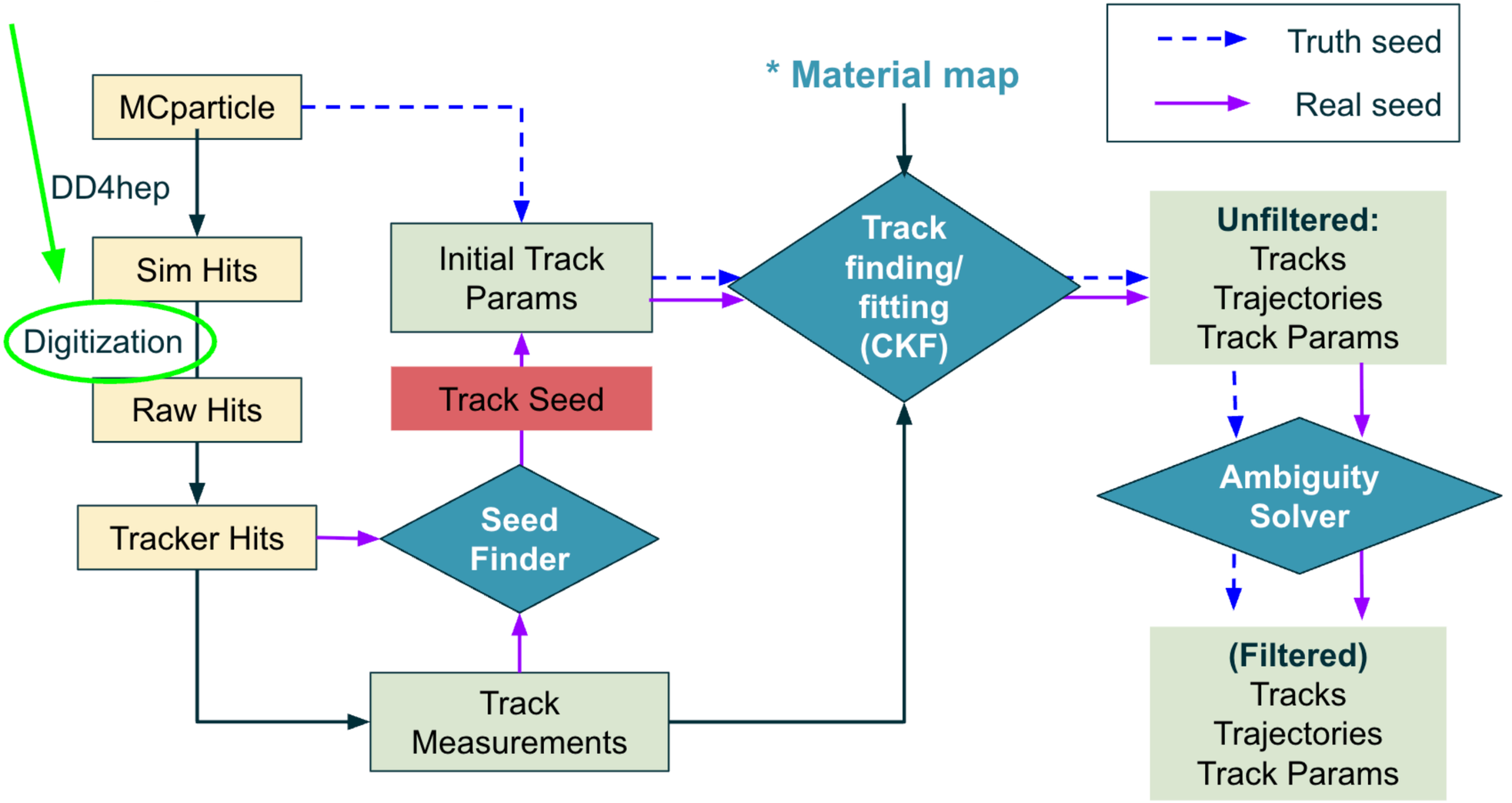


Number of noise hits in 2μs per subdetector

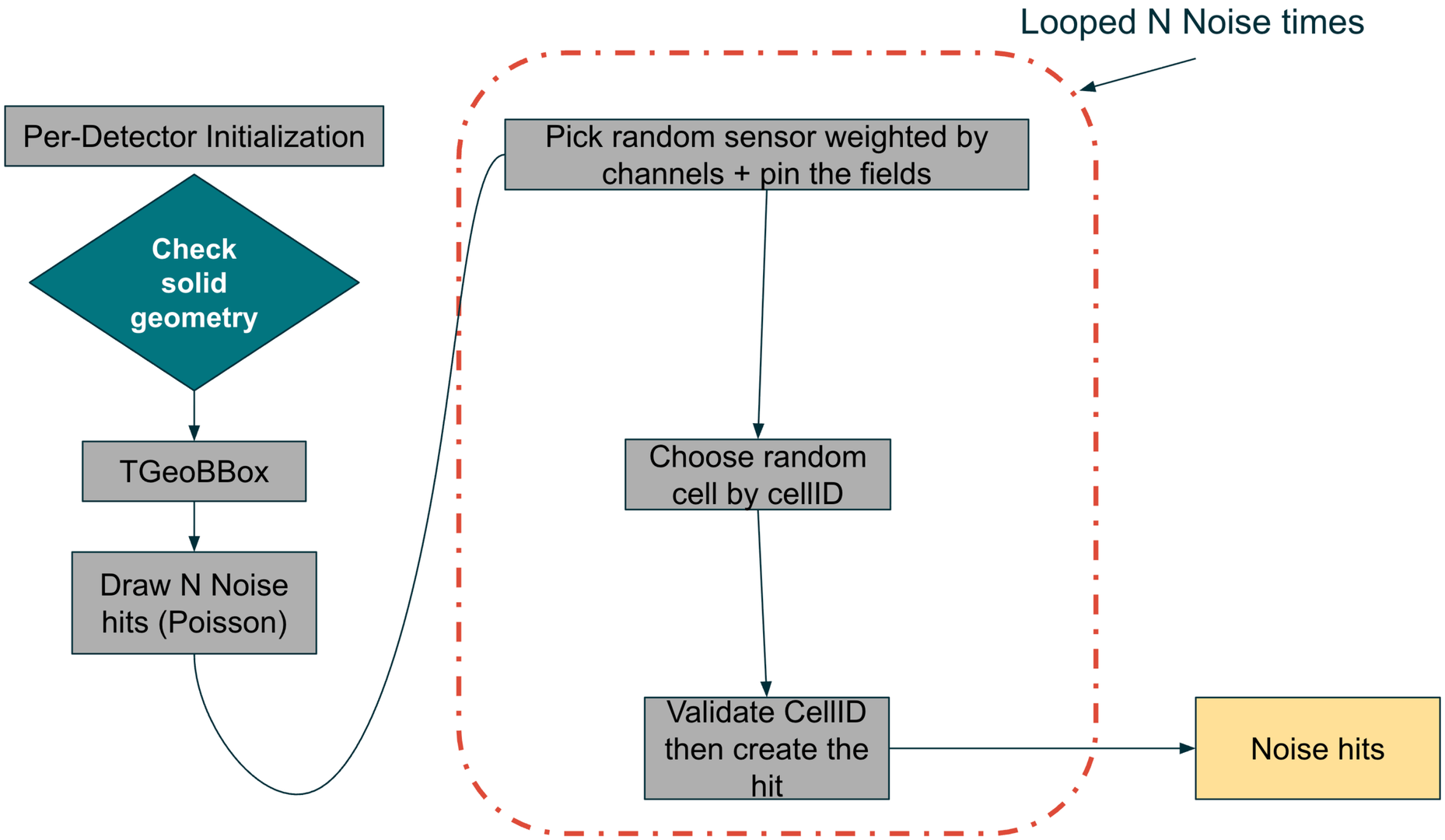
Detector name	Barrel layers	R [mm]	L [mm]	Area [mm]	# of pixels	# of noise hits per event*	# of background hits (10x275, 10um Au) per 2us
VertexBarrel	L0	36	270	6.11E+04	1.53E+08	76	722
	L1	48	270	8.14E+04	2.04E+08	102	503
	L2	120	270	2.04E+05	5.09E+08	254	309
SagittaSiBarrel	L3	270	540	9.16E+05	2.29E+09	1,145	987
OuterSiBarrel	L4	420	800	2.11E+06	5.28E+09	2,639	630
Detector name	**Disks**	**R1 [mm]**	**R2 [mm]**	**Area [mm]**	**# of pixels**	**# of noise hits per event***	**# of background hits (10x275, 10um Au) per 2us**
InnerTrackerEndcapN	E-Disk1	36.76	230	3.24E+05	8.10E+08	405	258
MiddleTrackerEndcapN	E-Disk2	36.76	430	1.15E+06	2.88E+09	1,442	463
OuterTrackerEndcapN	E-Disk3	36.76	430	1.15E+06	2.88E+09	1,442	445
	E-Disk4	40.06	430	1.15E+06	2.88E+09	1,440	290
	E-Disk5	46.35	430	1.15E+06	2.87E+09	1,435	62
InnerTrackerEndcapP	H-Disk1	36.76	230	3.24E+05	8.10E+08	405	223
MiddleTrackerEndcapP	H-Disk2	36.76	430	1.15E+06	2.88E+09	1,442	311
OuterTrackerEndcapP	H-Disk3	38.42	430	1.15E+06	2.88E+09	1,441	73
	H-Disk4	54.43	430	1.14E+06	2.86E+09	1,429	13
	H-Disk5	70.14	430	1.13E+06	2.83E+09	1,414	10

*1 event = 2μs

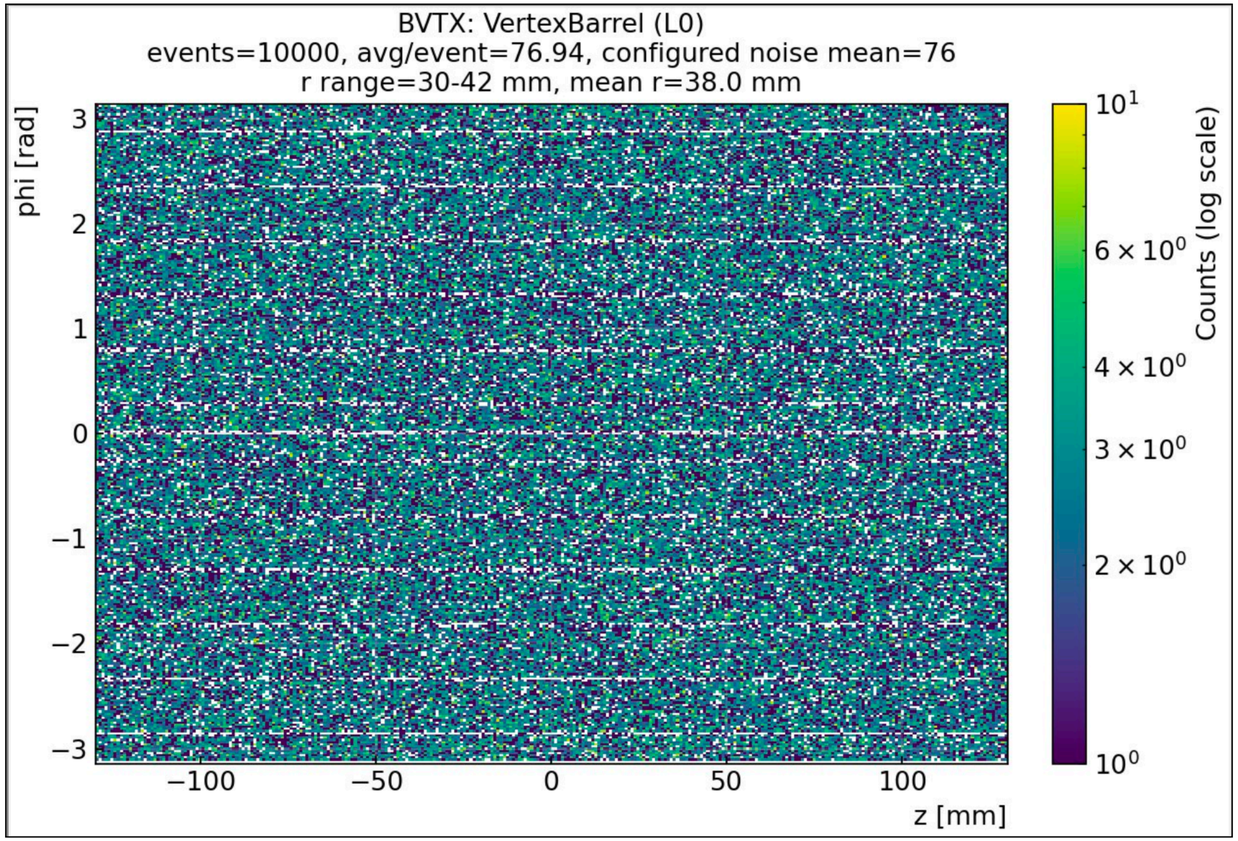
Noise Implementation

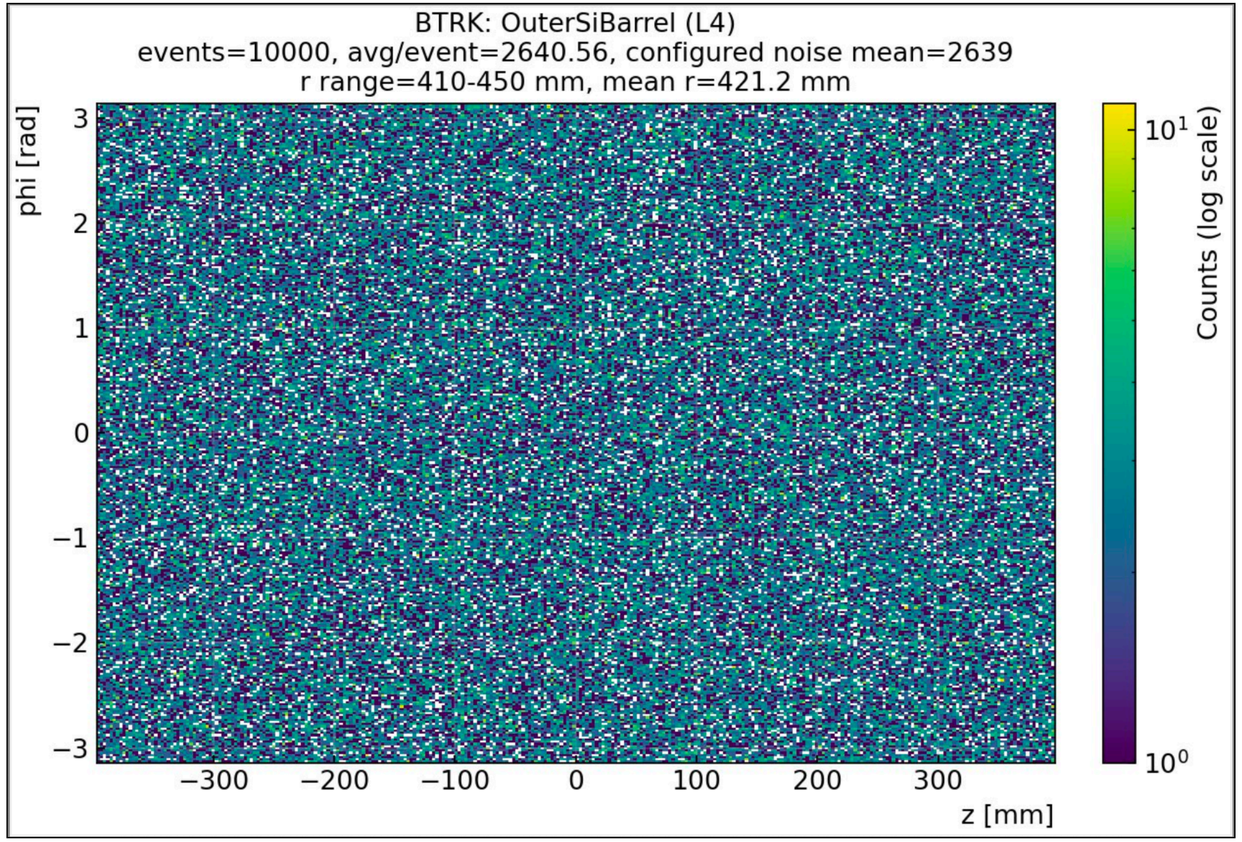


Prior Noise Implementation within the SVT barrels



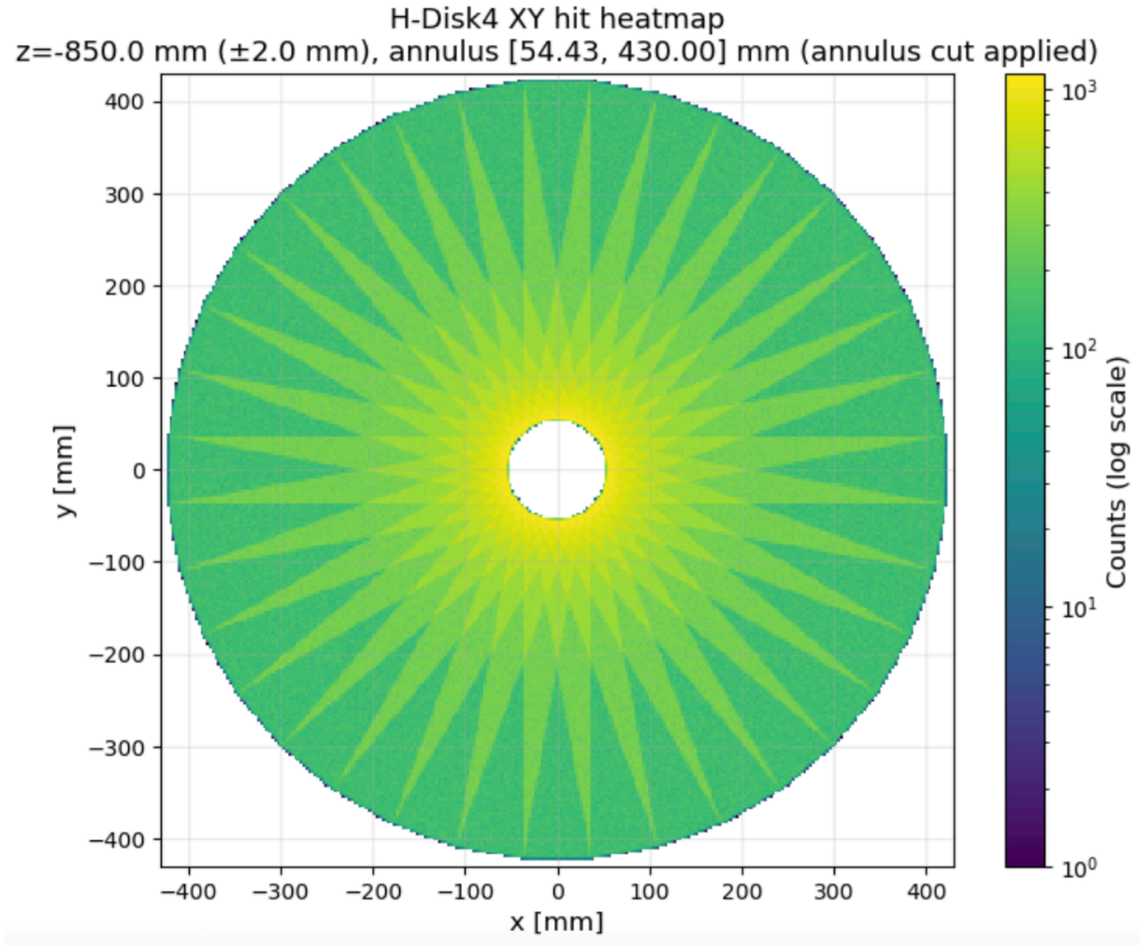
Noise Injection before the fixes





Here is the cross sectional noise distribution within one arbitrarily selected inner and outer barrel.

Prior noise implementation in the endcaps



Problem: needs to be fixed b/c the noise hit distribution is not uniform

Adding Noise to Endcap Disks:

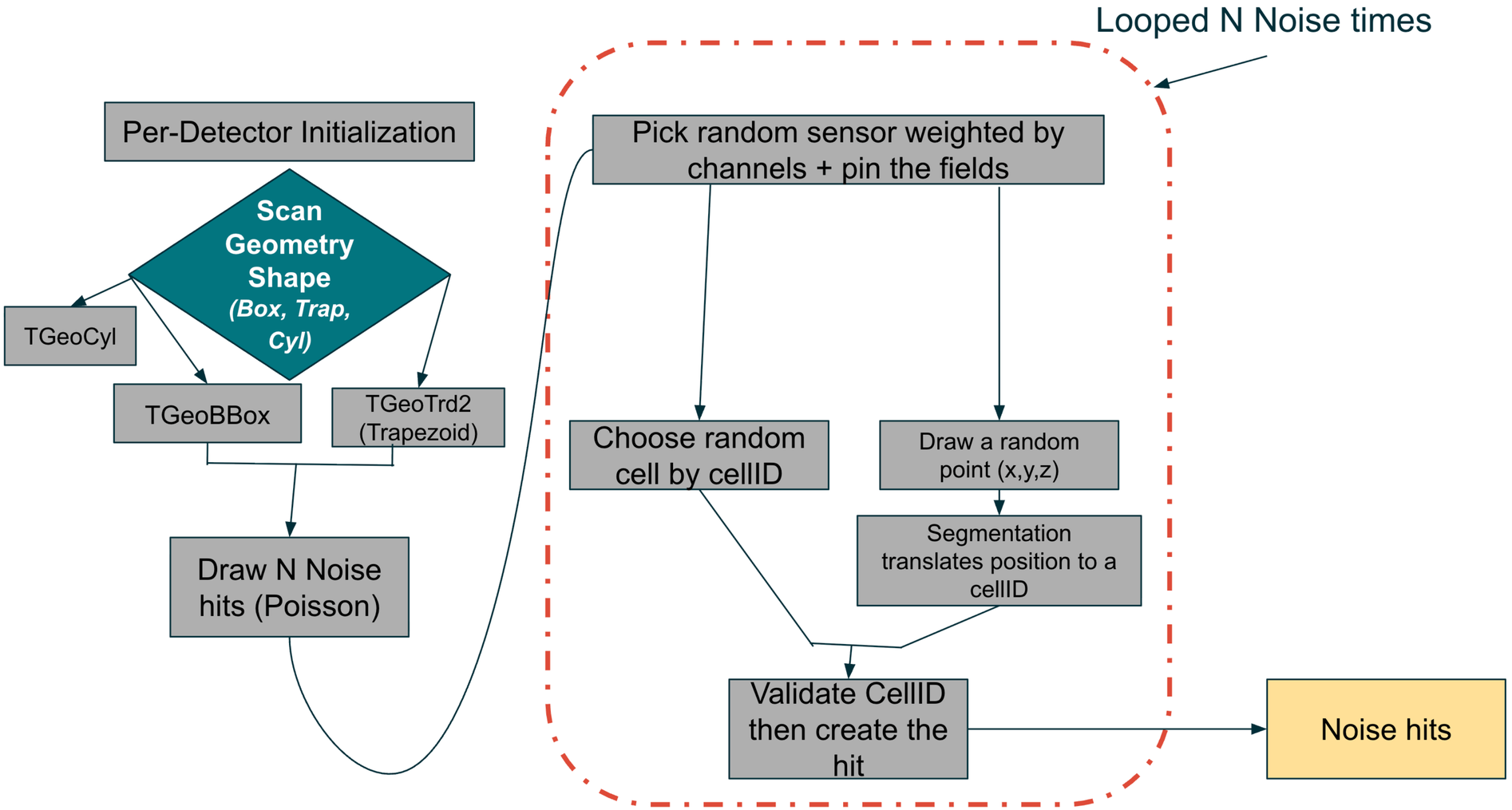
```
564
565   dd4hep::PlacedVolume leafPV;
566   auto findLeafPV = [&](dd4hep::DetElement d, auto&& self) -> bool {
567     if (!d.isValid()) return false;
568     auto pv = d.placement();
569     if (!pv.isValid()) return false;
570
571     if (pv.volume().sensitiveDetector().isValid()) {
572       bool childSensitive = false;
573       for (auto const& [_,c] : d.children()) {
574         auto cpv = c.placement();
575         if (cpv.isValid() && cpv.volume().sensitiveDetector().isValid()) { childSensitive = true; break; }
576       }
577       if (!childSensitive) { leafPV = pv; return true; }
578     }
579
580     for (auto const& [_,c] : d.children())
581       if (self(c,self)) return true;
582
583     return false;
584   };
585
586   const TGeoTrd2* trd = (findLeafPV(det,findLeafPV) ? dynamic_cast<const TGeoTrd2*>(leafPV.volume().solid().ptr()) : nullptr);
587
```

```
592     std::vector<double> w;
593     w.reserve(sensors.size());
594     for (auto* b : sensors) {
595       std::size_t n = 1;
596       for (auto const& [fname, r] : bounds) {
597         if (b->find(fname) != b->end())
598         continue;
599         n *= static_cast<std::size_t>(std::max<long>(1,r.second - r.first + 1));
600       }
601       w.push_back(static_cast<double>(std::max<std::size_t>(1,n)));
602     }
603
604     std::discrete_distribution<std::size_t> pickSensor(w.begin(),w.end());
```

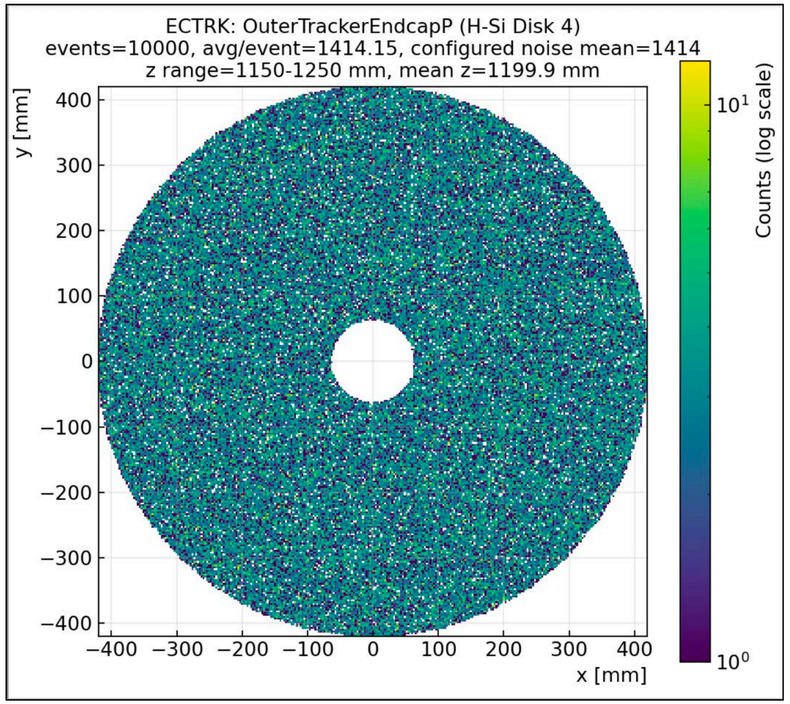
The trapezoidal geometry of the endcap modules are parameterized here and a random cell within this trapezoid is picked uniformly.

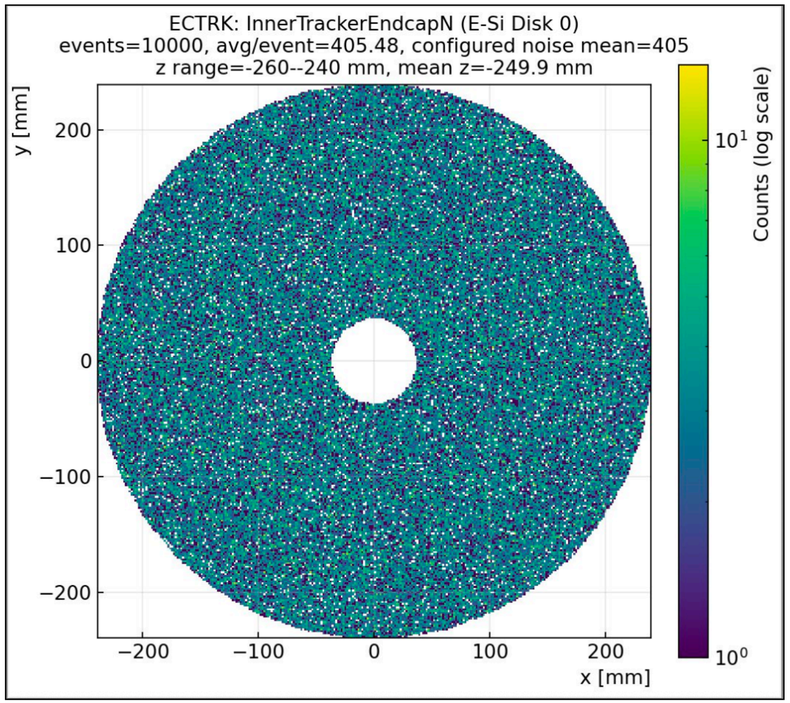
```
624     for (auto const& kv : base)
625       decoder -> set(cid, kv.first, kv.second);
626
627     if (trd) {
628
629       const double dx1 = trd->GetDx1();
630       const double dx2 = trd->GetDx2();
631       const double dy1 = trd->GetDy1();
632       const double dy2 = trd->GetDy2();
633       const double dz = trd->GetDz();
634
635       auto dxAt = [&](double z) {
636         const double t = (z + dz) / (2.0 * dz);
637         return dx1 + t * (dx2 - dx1);
638       };
639
640       auto dyAt = [&](double z) {
641         const double t = (z + dz) / (2.0 * dz);
642         return dy1 + t * (dy2 - dy1);
643       };
644
645       double aMax = 0.0;
646       for (int i = 0; i <= 16; ++i) {
647         const double z = -dz + (2.0 * dz) * (double(i) / 16.0);
648         aMax = std::max(aMax,dxAt(z)*dyAt(z));
649       }
650       if(aMax <= 0.0) continue;
651
652       std::uniform_real_distribution<double> uz(-dz,dz);
653       std::uniform_real_distribution<double> u01(0.0,1.0);
654
655       double dx = 0.0;
656       double dy = 0.0;
657       double z = 0.0;
658       for (;;) {
659         z = uz(rng);
660         dx = dxAt(z);
661         dy = dyAt(z);
662         if (dx <= 0.0 || dy <= 0.0) continue;
663         if (u01(rng) * aMax <= dx * dy) break;
664       }
665
666       std::uniform_real_distribution<double> ux(-dx,dx);
667       std::uniform_real_distribution<double> uy(-dy,dy);
668
669       dd4hep::Position p{ux(rng), uy(rng), z};
670       cid = segH.cellID(p,p,cid);
671     } else {
672       // Same as 3.2
673       for (auto& kv : fieldDists) {
674         if (base.find(kv.first) != base.end())
675           continue;
676         decoder->set(cid,kv.first,kv.second.uni(rng));
677       }
678     }
```

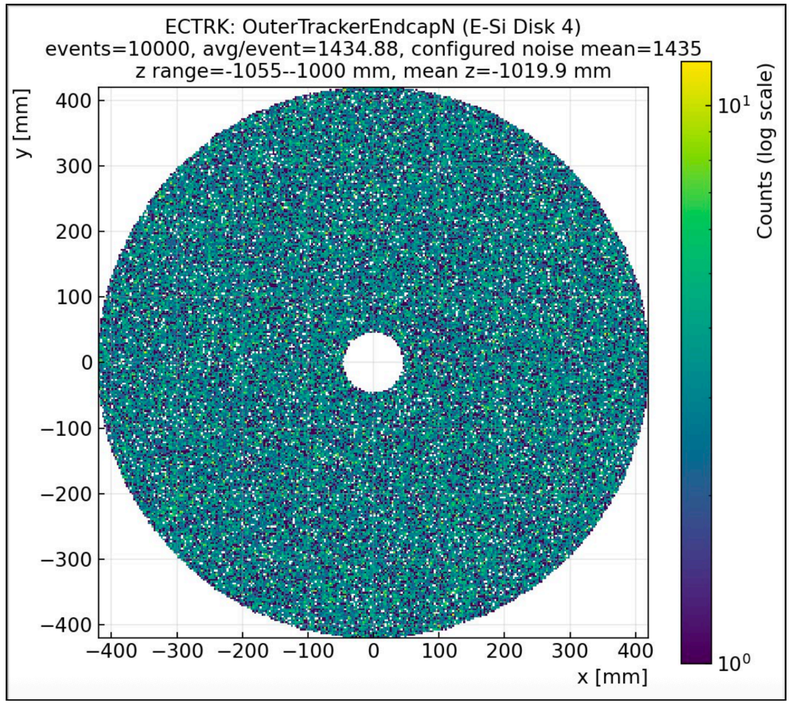
Adding Noise to Endcap Disks:



Noise Injection post-fix

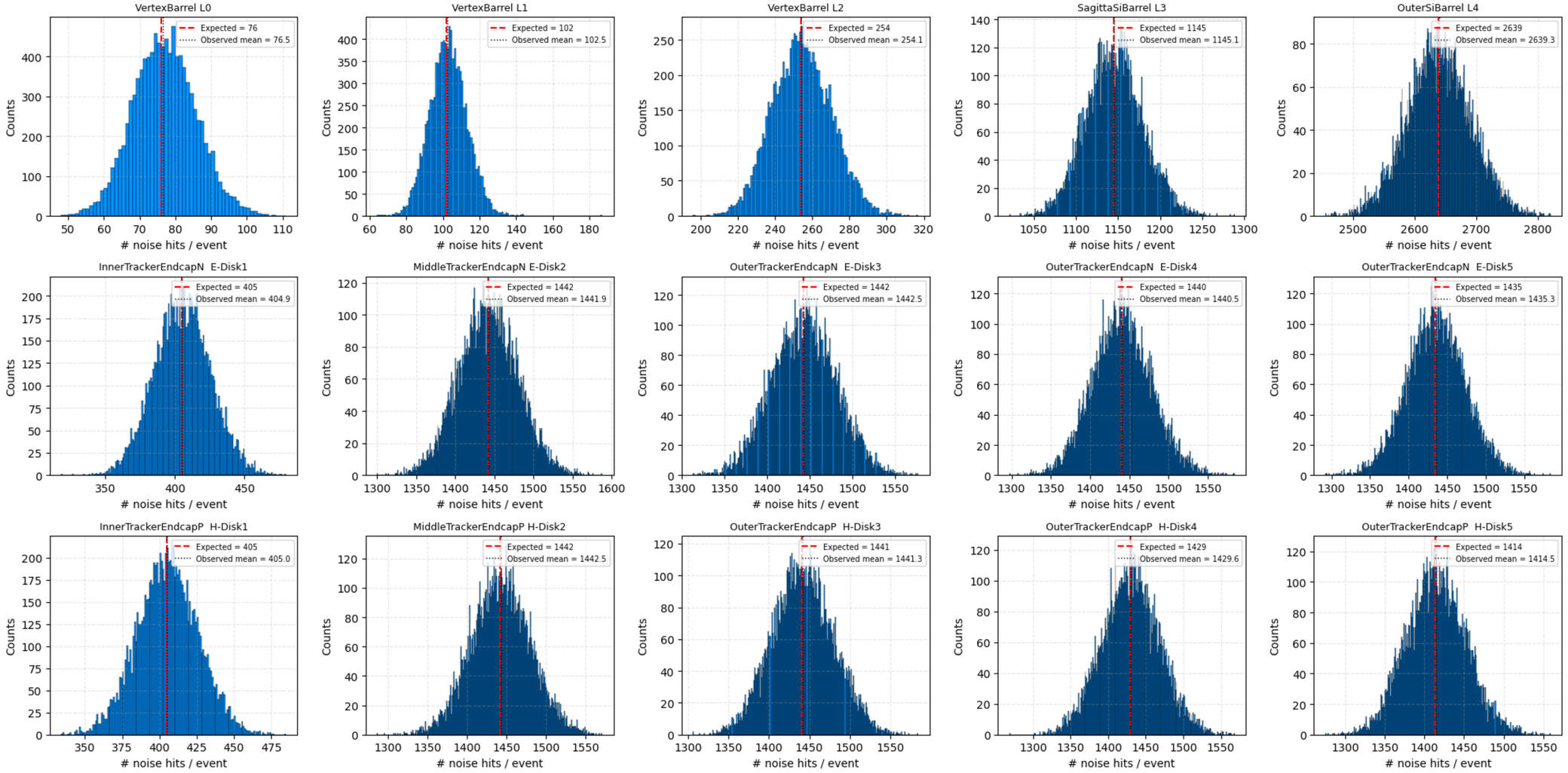






Endcap noise injection and geometry parameterization is altered. Noise is now uniformly distributed among the trapezoidal modules/staves of the endcaps.

Checking the noise implementation



The hits are Poisson-distributed about the fixed noise hit count.

Conclusion

- SVT noise injection into EICrecon is now completed and the initial checks look good
  - Changes were added in PR #2670.
- Currently, there's a lot of 'noise' tracks that need to be filtered.
  - When done, the noise tracking study can be continued.