DS-20k G4DS Weekly Report

Niamh Fearon

October 22, 2019

Contents

1 Introduction **2**

2 SiPM placement in current DS-20k geometry **2**

3 Simple G4DS Simulation to investigate Optics and SiPM efficiency **4**

3.1 Week 5 Summary: Creating a new detector configuration with a single SiPM . 5

3.2 Week 6 Summary: Optics Tests in the TPC 8

3.2.1 Small VUV photon event simulation - 10,000 events 8

3.2.2 1,000,000 VUV photon event simulation 9

3.2.3 1,500,000 VUV photon event simulation 9

3.2.4 Large data simulation plan (many millions of events) 12

3.3 Week 7 Summary: Light Maps 12

3.3.1 200,000,000 VUV photon event simulation 12

3.4 Week 8 Summary: New light maps and MC Veto Presentation 14

3.4.1 MC Veto Team Presentation 12th Nov 14

3.4.2 300 million VUV photon event simulation 15

3.4.3 DarkSide Collaboration Meeting - Madrid 16

3.5 Week 9 Summary: 16

3.6 Week 10 Summary: DIRAC Grid Distributed Computing System and new multi-device SiPM geometry 17

3.6.1 Large working DIRAC Grid job for simulating VUV photons 17

3.6.2 Multi-device SiPM geometry 18

3.7 Week 11 Summary: Editing the Multi-device SiPM geometry 19

3.8 Week 12 Summary: 22

3.8.1 Continuation of multi-device SiPM geometry 22

3.8.2 Report Plan Feedback 22

List of Figures

1 SiPM Volume Schematic 2

2 Current DS-20k Geometry Oct 2018 4

3 Current SiPM Placement picture 4

4 First experiment design 5

5 First visual of the new TPC design, no SiPMs included 6

6 Single SiPM inside the initial TPC configuration 7

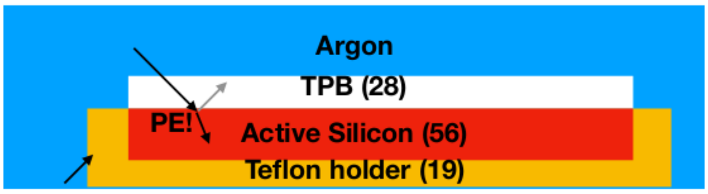


Figure 1: SiPM structure shown with a metal teflon holder, active silicon collecting area and a TPB coating. Only the outward facing surface area of the SiPM collects light and creates photoelectrons.

7 Photon wavelength of VUV photon events generated directly above the TPC at varying *z* coordinates 9
8 Test for number of photons per event in a small MC data set 10
9 The *xyz* distribution of VUV photons for events where nph ≥ 2 10
10 Random spatial distribution of 1500000 VUV photon events in the TPC 11
11 Distribution of 1500000 VUV photon events in the TPC with the added cut for nph ≥ 2. 11
12 200 million events cut events and light maps 13
13 200 million events light maps for *xz* and *yz* planes. 14
14 300 million events light maps for *xy* planes, binned per 1cm^2 and 4cm^2. 15
15 300 million events light maps for *xy* planes in terms of number of photons per voxel, binned per 1cm^2. 16
16 Multi-device geometry - 4 SiPMs in an approximate 100 kg LAr volume 19
17 Multi-device SiPM geometry placed directly in the fMotherVolume 20
18 LAr box within the ProtoDUNE cryostat 21
19 4 SiPMs aligned with the LAr TPC box edges 22

1 Introduction

ZOOM Meeting at 2:30pm GMT 19th Oct with Paolo Agnes (and possibly Andrzej Szelc depending on time). Topics of discussion:

- Current code in DSDetectorPlasticVeto.cc to place SiPMs in current DS-20k geometry
- Stripped back G4DS simulation to calculate SiPM efficiency as a function of position

Details of the discussion follow and work done to complete the tasks.

2 SiPM placement in current DS-20k geometry

In the long-term we need to place SiPMs in the Veto volumes. As of Fri 19th October there are 6 functions in *DSDetectorPlasticVeto.cc* in the *sol_niamh_andrzej _g410.1.2* GitLab branch that place SiPMs in the Vetoes. The SiPMs are created in lines 184-210 in this file. The code that places the SiPMs is given in lines 213-236.
As the geometry of the Vetoes is cylindrical we need two separate functions to place the SiPMs on the top and bottom 'caps' of the cylinders, as well as around the curved sides. These functions are declared in the *DSDetectorPlasticVeto.hh* header file and are called:

```
void PlaceSiPMsInCaps (G4double,G4double,G4double,
G4int,G4LogicalVolume *,G4VPhysicalVolume *,int)

void PlaceSiPMsInSides (G4double,G4double,G4double,
G4int,G4int,G4LogicalVolume *,G4VPhysicalVolume *,int)
```

The specifics of these two functions are written in the *DSDetectorPlasticVeto.cc* file from lines 926-1005. The arguments of these functions include:

- Radius value, R.
- Vertical height value, Z.
- R and Z rotation angles.
- Number of lines of SiPMs.
- Number of steps in placement loop.
- Logical volume (smallSiPMLogic).
- Physical volume.
- Overlap checker.

The macro commands needed to set the volume sizes for each simulation are:

```
/ds/detector/configuration 11
/ds/detector/ds20cryo_tpcEdge 150 cm
/ds/detector/ds20cryo_tpcHeight 262 cm
/ds/detector/ds20_AcrylicWalls_Thick 5 cm
/ds/detector/ds20_LArBuffers_Thick 40 cm
/ds/detector/ds20_VetoShell_Thick 10 cm
```

The current code is the repo is good for the long term, but before December we will want to do a simpler investigation of optical photons in LAr that does not require the full scale simulation.

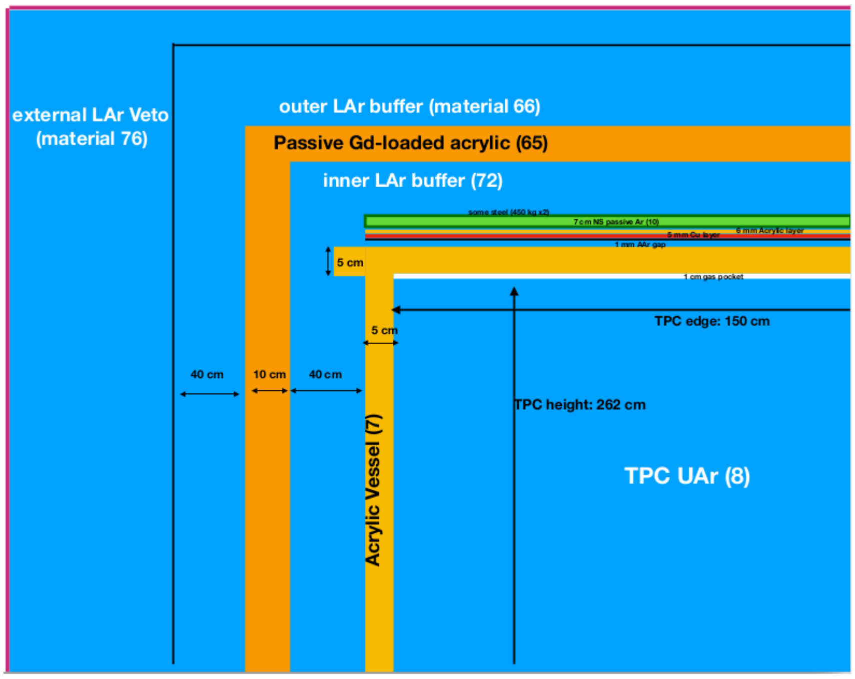


Figure 2: DS-20k Geometry at the current development stage. The LArBuffer Thickness, Acrylic Wall Thickness and Veto Shell Thickness can all be set in the *.mac* files. This is for the team that is optimising the dimensions.

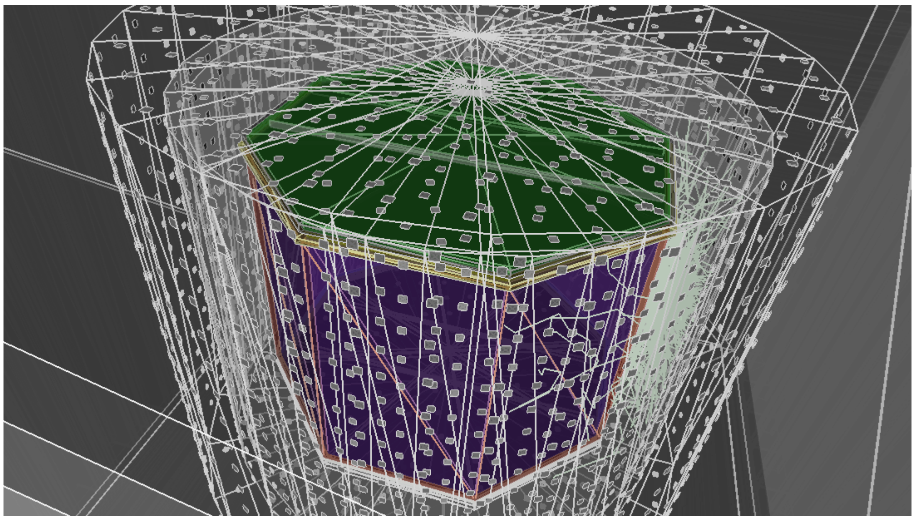


Figure 3: SiPMs placed in all of the Veto volumes as of 19th Oct. Leave these for now until we know how many SiPMs we will need to collect light efficiently.

3 Simple G4DS Simulation to investigate Optics and SiPM efficiency

TASK GIVEN BY PAOLO : First step: 1 SiPM observing a "large" volume of LAr and determine which volume is effectively illuminating the device (probability map).
To do the first initial study of the optics in the LAr medium we are aiming to start with a single SiPM in a box. We want to randomly generate a distribution of optical photons and find the

effective area that the SiPM can observe. The geometry will be completely simplified. Aim to place one SiPM on the base of the TPC Cathode facing into the LAr. The current TPC geometry is in *DSDetectorDS20k.cc* from lines 445 onwards. Note: there are multiple outdated TPC designs still in this code file.

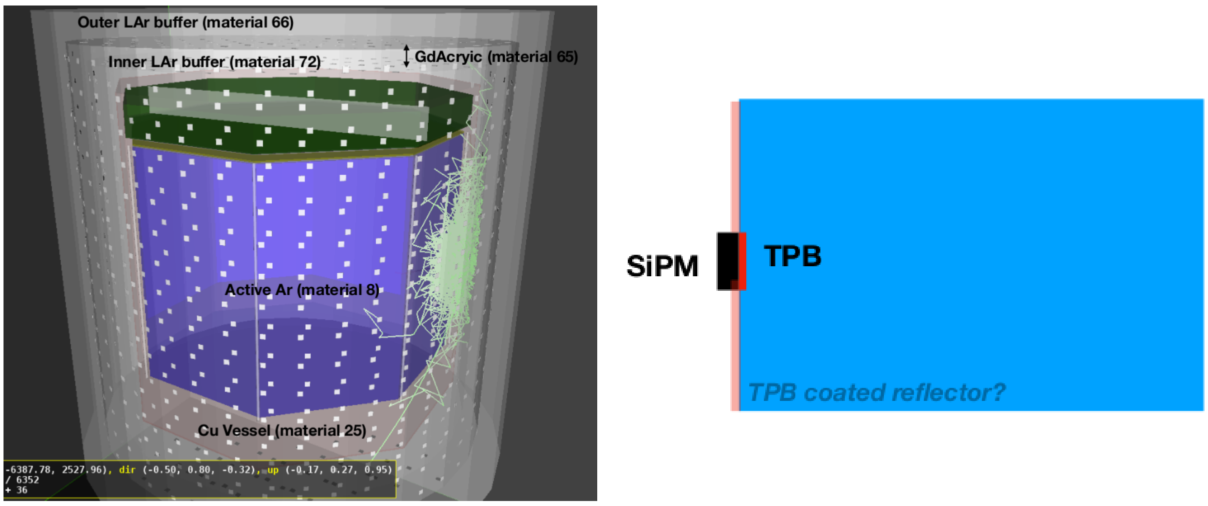

Outer LAr buffer (material 66)
Inner LAr buffer (material 72)
GdAcryic (material 65)
Active Ar (material 8)
Cu Vessel (material 25)
-6387.78, 2527.96), dir (-0.50, 0.80, -0.32), up (-0.17, 0.27, 0.95)
/ 6352
+ 36
SiPM
TPB
TPB coated reflector?

Figure 4: The current DS-20k TPC will be used as our box for the simple simulations. The TPB coating will be removed and replaced with Stainless Steel, as we want an absorbing material. One SiPM in a large LAr volume, which will be the LAr inside the TPC.

The factors that will limit the SiPM efficiency are mainly the Rayleigh scattering of LAr and the purity of the LAr. The VUV photons that we generate as a first step are generated by the macro commands:

```
/ds/generator/particle opticalphoton
/ds/generator/energy 9.7 eV
```

This generates VUV 128nm photons. We will need to test that these are converted into visible photons by the TPB coating and detected as photoelectrons by the SiPMs.
The next step is to generate these events in a random spatial distribution inside the TPC volume. Take a look in *DSVGeneratorMessenger.cc* to look at this distribution function.

```
/ds/generator/tpcdistribution 1
```

The TPB coating also needs to be removed from the TPC. In *DSDetectorDS20k.cc* search for any instances of *GetTPB()*. Change both instances to Stainless Steel (look at *DSMaterial.cc* to find the specific variable name). The optical properties of LAr are in *DSOptics.dat*; could possibly vary these during the study but not a major concern.
Next Step: Create a copy of the *DSDetectorDS20k.cc* files and call it *DSDetectorDS20k.cc.xx*. The *.cc.xx* file will not compile during the source and make commands but we want to keep a copy of the working original code. Make the necessary edits to the *.cc* file to make the TPC box and a single SiPM.

3.1 Week 5 Summary: Creating a new detector configuration with a single SiPM

I have created a new file in the *src* directory called *DSDetectorDS20kTPC.cc*, which is a copy of the *DSDetectorDS20k.cc* file. The new file is the one I will edit to make a new TPC configuration with a single SiPM. To make just the TPC volume there needs to be a new detector configuration which sets the TPC to be the world volume. The new detector configuration is made in the *DSDetectorConstruction.cc* file.

```
else if(fDetectorConfiguration == 12) {
  new DSDetectorDS20kTPC(fPhysicWorld);
  DSStorage::Get()->Set20KGeometry(true);
  DSStorage::Get()->Set20KAlternateVeto(true);
}
```

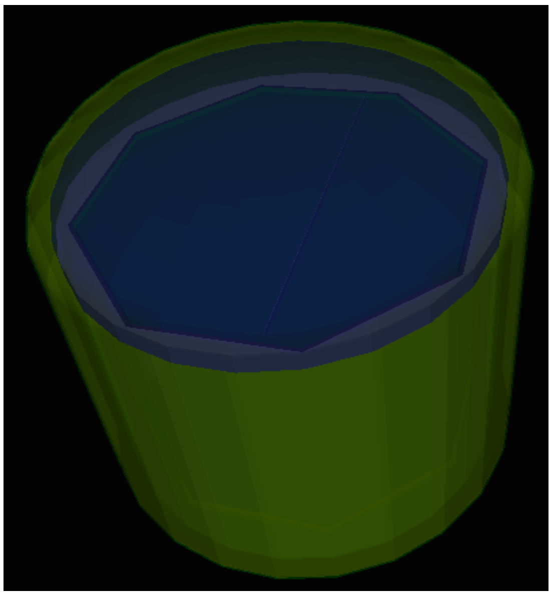


Figure 5: This is the visual that is created when running *viswrl.mac* with the new detector configuration (12). There have not been any SiPMs added yet, this was just to test the code builds the TPC without difficulties.

The code that creates the SiPMs is taken from *DSDetectorPlasticVeto.cc* and used in *DSDetectorDS20kTPC.cc*. See lines 993-1018.
We want to place the SiPM at (0,0,-myTPCHeight/2. + 5*mm) facing in the +z direction. This is so that the SiPM is placed near to the TPC cathode floor, and the extra 5*mm is to avoid overlaps when building the volumes.

```
G4RotationMatrix* mySiPM_FacePZ  = new G4RotationMatrix();
mySiPM_FacePZ->rotateY(-90*degree);

G4PVPlacement *SingleSiPM = new
G4PVPlacement(
mySiPM_FacePZ,
G4ThreeVector(0,0,-myTPCHeight/2. + 5*mm),
"single_SiPM",
smallSiPMLogic,
fPhysicActiveLAr,
false,
0,
myCheckOverlap);
```

The rotation matrix is needed to place the SiPM facing in the +z direction (in the *xy* plane). Without the matrix the SiPM is orientated in the *xz* plane. The variable *myTPCHeight* is set

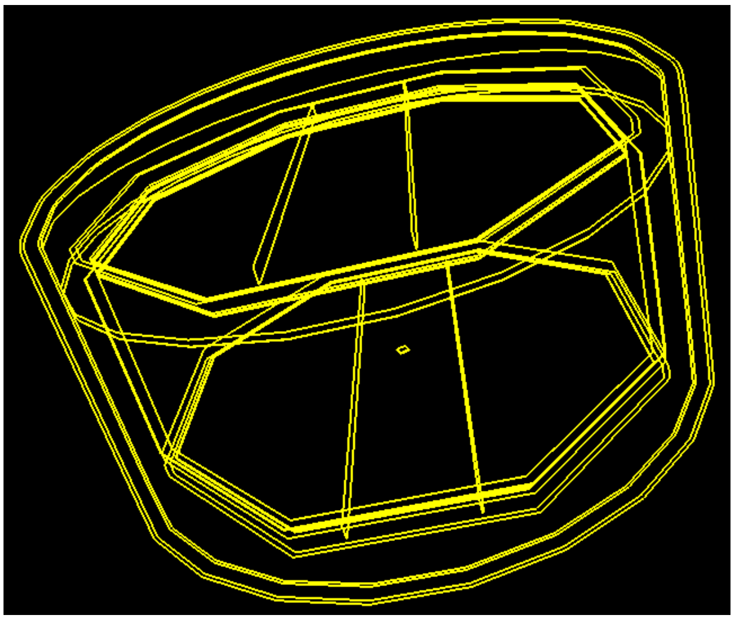


Figure 6: A single SiPM has been placed just above the cathode of the TPC volume. There is enough space between the cathode floor and the TPC for there to be no physical volume overlaps.

in the macro commands when selecting a detector configuration. The is the total height of the TPC, i.e the z coordinate of the top of the TPC is *myTPCHeight*/2 relative to the origin . The value is passed by the following line:

```
G4double myTPCHeight =
DSStorage::Get()->GetDS20kTPCheight();
```

To apply all of these changes in a macro file the new commands needed are:

```
/ds/detector/configuration 12
/ds/detector/ds20cryo_tpcEdge 50 cm
/ds/detector/ds20cryo_tpcHeight 50 cm
/ds/detector/ds20_AcrylicWalls_Thick 5 cm
```

We do not need a large LAr volume to do the light map studies, so TPC dimensions 50 cm x 50 cm should be enough.
Problems: No PEs generated when running VUV 128nm photons at position (0,0, -myTPCHeight/2 + 10*cm) directed in the -z direction towards the SiPM.
Solutions: (input from Paolo)

- Removed some useless volumes in the TPC (grid,gaspocket,outer supports...). This solves the overlap problems.
- Restored some of the optical parameters of LAr.

  ```
  fLiquidArgon->SetMaterialPropertiesTable
  (myLiquidArgon);
  ```

- Changed optical properties of TPB, LAr and MetalSilicon. The reason is not straightforward.

The tracking of an optical photon requires optical properties to be assigned to the material the photon travels in. For LAr, we need to set absorption length (disruptive process that kills the

photon - likely absorbed on impurities) and scattering length (change in momentum). For our purposes, we only care about the optical properties of LAr, and they are set in *data/detector/DSOptics.dat* (LiquidArgonUVAbs, LiquidArgonVisAbs, LArRayleighScale).
At the interface between two volumes, Geant4 is using by default the Fresnel optics (every material need a refractive index to be defined), unless the std optics is override by user-defined properties. This might be the case for dielectric-metal surfaces. In our implementation, we need:

- The TPB must absorb and convert any VUV photon.
- The TPB must emit isotropically.
- The Silicon must absorb each photon (the detection efficiency of these devices is not 1, but we can scale offline).

We can only use Fresnel optics and I am setting the RINDEX's of LAr, TPB and Silicon to be equal, in order to avoid any unwanted reflection. On the other hand, the diffusive nature of the TPB is conserved (it emits isotropically). This is done in *data/detector/DSOptics.dat*, where:

- TPBUVRind = PhotocathodeUVRind (actually this one is for the Silicon) = 1.47 = rindex of LAr in the VUV.
- TPBVisRind = PhotocathodeVisRind = 1.22 = rindex of LAr in the visible.

Files changed in Paolo's git commit to the sol_niamh _andrzej_g410.1.2 branch (26th Oct):

- data/detector/DSOptics.dat
- src/DSDetectorDS20kTPC.cc
- src/DSMaterial.cc

3.2 Week 6 Summary: Optics Tests in the TPC

3.2.1 Small VUV photon event simulation - 10,000 events

Using *run.mac* I have generated 10000 VUV photons using the following macro commands:

```
/ds/generator/select G4Gun
/ds/generator/particle opticalphoton
/ds/generator/direction 0 0 -1
/ds/generator/position 0 0 -ZZZ cm
/ds/generator/energy 9.7 eV
/run/beamOn 10000
```

This generates VUV photons of wavelength 128nm towards the SiPM. For the initial z coordinate I have chosen 3 different positions to generate the photons at. This is to check that there should be more visible photons produced if the initial VUV photon is generated closer to the SiPM, as these photons will have a larger probability of hitting the TPB coating and being wavelength shifted.
Z positions of generated photons:

- -15 cm

- -20 cm
- -22.5 cm

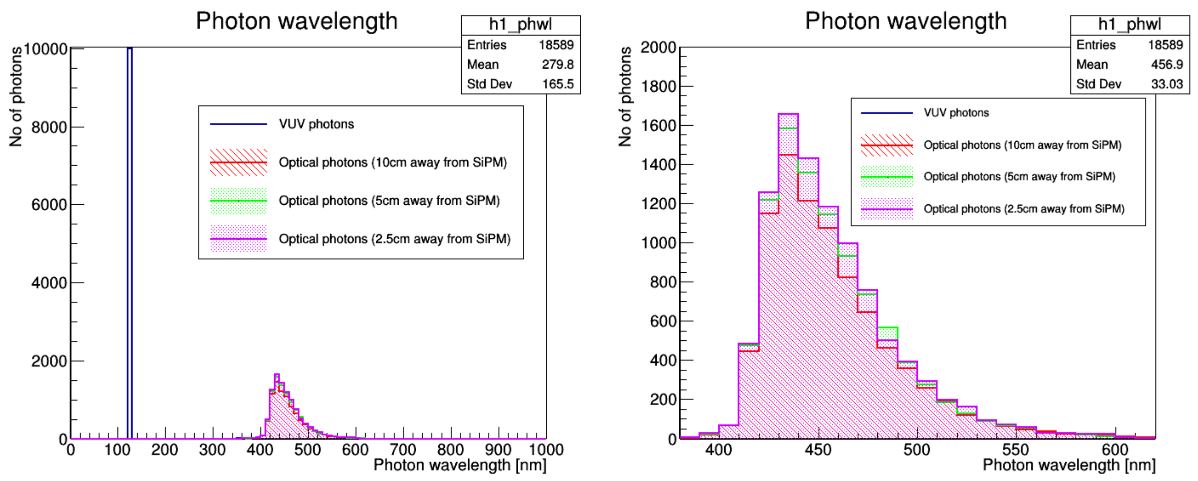


Figure 7: 10000 VUV photons generated at the z coordinates listed for the initial TPC checks. The large blue peak is the 10000 initial VUV photons at 128nm and the bump at around 420nm shows the visible photons created by the TPB wavelength shifter. Note that from 10000 events details for 18000 photons are stored. This is because when a VUV photon is converted by the TPB the VUV photon dies and the visible photon is created. We need to account for this when calculating the detection efficiency.

3.2.2 1,000,000 VUV photon event simulation

The run time to generate a fairly large number of VUV photons is not long. I ran a 1000000 event run.mac to see how many events the rooter could store. The ROOT file created from this run stored 441436 events. This was due to a space issue and I will move to *hepgpu1-data1* to generate events. We are interested in the events where the number of photons is greater than or equal to 2, i.e from the single initial VUV photon another photon is created from interaction with the TPB.

3.2.3 1,500,000 VUV photon event simulation

From Figure 8 we can see that much larger data sets will be needed to investigate the interesting events, i.e. nph $\geq$ 2, so that we can produce light maps and calculate the SiPM efficiency.
To do this we have:

- Written a bash script that edits the *run.mac* file to find and replace the output filename and the HEP random seed in a loop. The bash script then can compile the rooter and convert the binary *.fil* files into *.root* files. The bash script is called *runmulti.sh* and is saved in the *Linux_g++* directory. To run this bash script:

  ```
  $ bash runmulti.sh
  ```

- Used 'hadd' to add together the ROOT outputs of the bash script.

  ```
  $ hadd total_file.root *.root
  ```

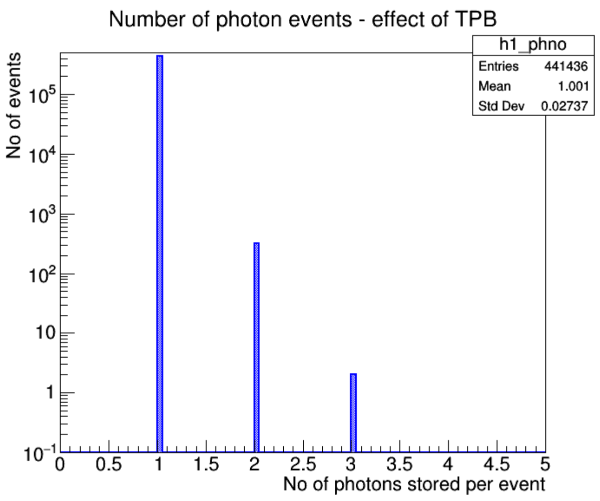


Figure 8: The number of photons stored per event. The large number of events with only one photon will be the events where VUV photons do not reach the SiPM and are absorbed by the TPC component materials. Only a small percentage of events have more than one photon generated.

- I have set the loop to run from 0 to 14, so 15 ROOT files have been created each with 100000 events → 1500000 events in a ROOT file created called *total_file.root* in the *Linux_g++* directory on the *hepgpu1-data1* disk.

I have written a simple macro file to analyse this ROOT file to produce simple preliminary plots for this optics investigation. The C macro *LightMap.C* can also be found in the *Linux_g++* directory. To run the macro:

```
$ root -l
root[0] .L LightMap.C
root[1] light_maps()
```

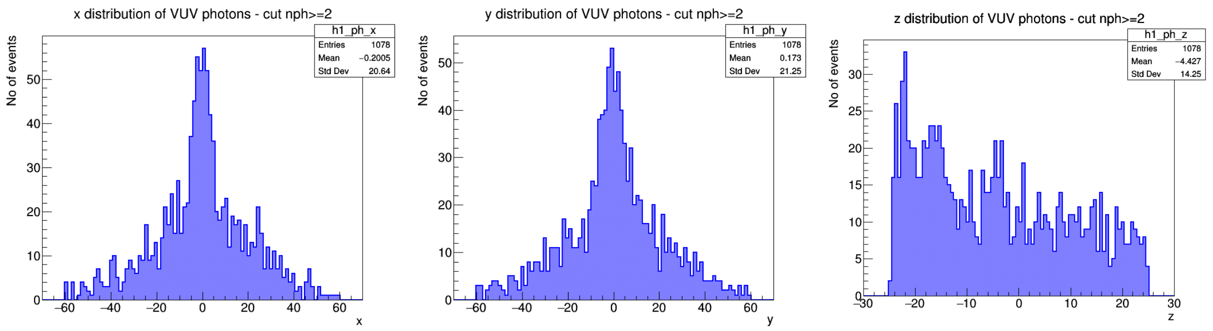


Figure 9: The xyz initial generation position of the VUV photon events where nph $\geq$ 2. As expected the x and y histograms look almost identical in shape due to the symmetry of the detector. The z histrogram will peak closest to the position of the SiPM, which is approximately at a z position of -25 cm.

The graphs in Figure 9 tell us that our simulation is working as expected: events which are generated at xy coordinates (0,0) and with a z coordinate increasingly close to the SiPM position will most likely reach the SiPM and be converted to visible photons by the TPB coating.

We can also check that the event generation in randomly and uniformly distributed in the TPC volume by plotting the postions of the initial events in both the *xy* and *xz* planes (assuming that *yz* is almost identical to the *xz* plane). The graphs for this check are shown in Figure 10.

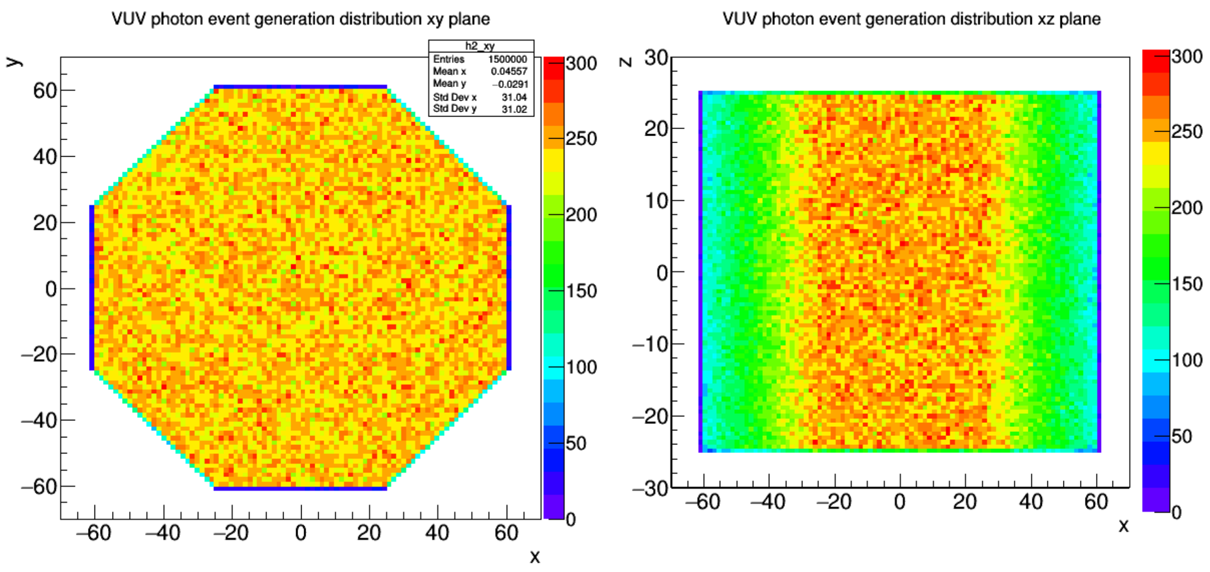


Figure 10: Slices in the *xy* and *xz* planes to check that the spatial distribution of events in random and uniform for 1500000 events. We need to fill the entire TPC volume with enough events so that we get a 'good' amount of statistics to work with.

Figure 11 shows that we need lots more events if we want to get reasonable statistics for our SiPM efficiency test. Out of 1500000 events only 1078 have created 2 or more photons, 1 VUV and 1 or more visible. We are not cutting on the number of photoelectrons as to increase the statistics. If we did cut for photoelectrons instead of photons this would cut to an even smaller percentage of events. We want as many events as possible.

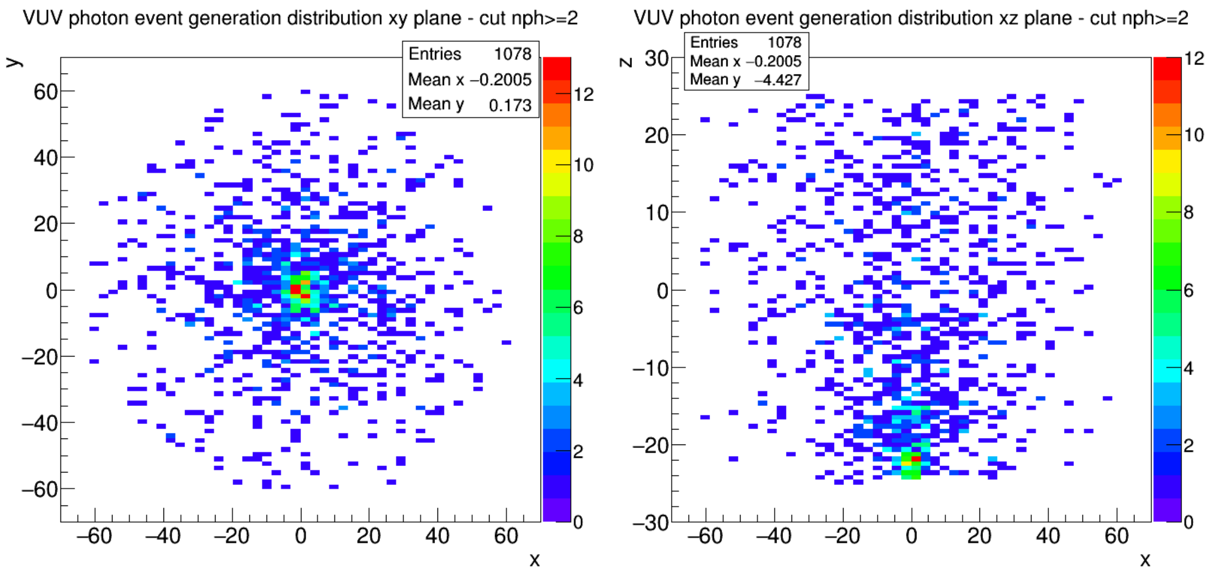


Figure 11: The *xy* and *xz* distribution of 1500000 events after the cut for nph $\geq$ 2. There are only 1078 events left after this cut, implying we need many more events to plot light maps for the SiPM efficiency.

3.2.4 Large data simulation plan (many millions of events)

Procedure for the large light map data set we need to create:

- Move the old *.root*, *.fil*, *.log* and *.mac* files from the *Linux_g++* directory on *hepgpu1-data1* to directory *1500000_event_files*.
- Run the *runmult.sh* bash script in 200 file blocks, each run containing 100000 events, using the HEP random seed option in the *run.mac*.
- Cut these ROOT files and write into a new ROOT file only filled with branch variables *x*,*y*,*z* and *ev* using a cut for nph ≥ 2.To do this we have made a new Make Class file.

  ```
  $ nohup bash cut.sh &
  ```

  The script *cut.sh* needs to be edited for each run and in the run_X _files directory .
- See how many events are left from this cut and run more 200 run batches.
- Check track of the total number of events, this is important for computing the SiPM efficiency.
- 'hadd' the cut ROOT files to get a large ROOT file with only events nph ≥ 2.
- **NOTE:** Look up the bash command 'nohup' (No Hangup) which allows you to execute a job in the background in a way that logging out of the terminal will not stop it.

  ```
      $ nohup <command> &
      $ exit
  ```

3.3 Week 7 Summary: Light Maps

3.3.1 200,000,000 VUV photon event simulation

We ran into issues using this method to generate a large data set.

- The HEP random seed was not changing correctly in the .sh script loop. This meant that several runs out of the 200 had the exact same simulation data. This caused large bumps to appear in the histogram plots after using hadd to create the total ROOT file.
- The new events were not given unique event ID numbers. This may have caused further issues when trying to separate events or hadd the ROOT files.

The solutions we have implemented are:

- Using the autoSeed option in the *run.mac* which uses a seed based on the system time. This solves the issue with copies of the same data being generated.

  ```
  /run/autoSeed 1
  ```

- Edit the new event ID number in the *runCut.C* file which is an edited version of the *myClass.C* file.

  ```
  ev_new = ev + 100000*(200*fRun+fFile);
  ```

The ROOT files are cut and a new TTree is made by the *runCut.C* script.

```
void runCut(TString fname, int run, int file)
{
myClass::myClass m(fname, run, file);
m.Loop();
m.WriteOutputTree();
}
```

This is placed in a loop to cut for all ROOT files generated in the *cut.sh* script.

Completed simulation runs:

1. Run Number 1 : Tues 12:15 *total_cut_file_1.root*
2. Run Number 2 : Tues 14:13 *total_cut_file_2.root*
3. Run Number 3 : Tues 15:05 *total_cut_file_3.root*
4. Run Number 4 : Tues 16:35 *total_cut_file_4.root*
5. Run Number 5 : Tues 17:42 *total_cut_file_5.root*
6. Run Number 6 : Thurs 10:28 *total_cut_file_6.root*
7. Run Number 7 : Thurs 11:36 *total_cut_file_7.root*
8. Run Number 8 : Thurs 13:27 *total_cut_file_8.root*
9. Run Number 9 : Thurs 15:01 *total_cut_file_9.root*
10. Run Number 10 : Thurs 16:28 *total_cut_file_10.root*

Out of the 200 million initial VUV photon events, only 148727 events survived the nph ≥ 2 cut. This means that for this cut there is an approximate 0.07% survival rate. We need a very large data set to investigate this study further.

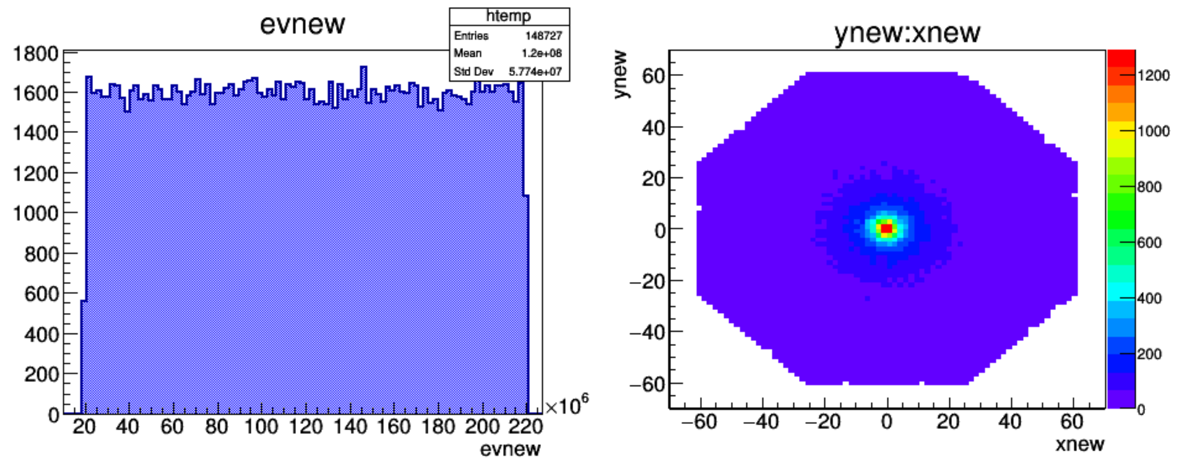
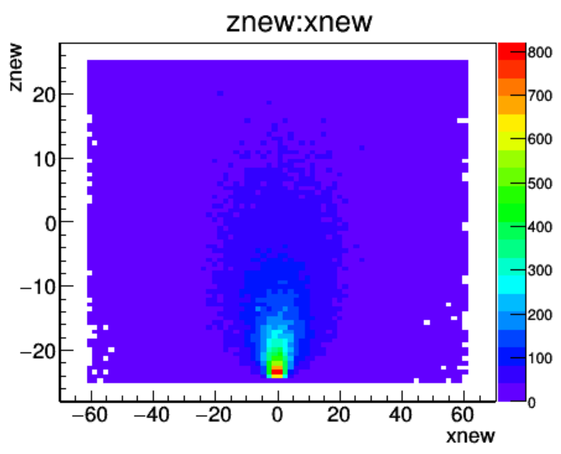


Figure 12: New event ID numbers allocated to events based on the run number and file number of each separate simulation. This was done to avoid grouping events together by mistake. For the *xy* plane light map plot there are 80 bins in both directions. Note that this is not binned per cm, this will be changed in future plots.



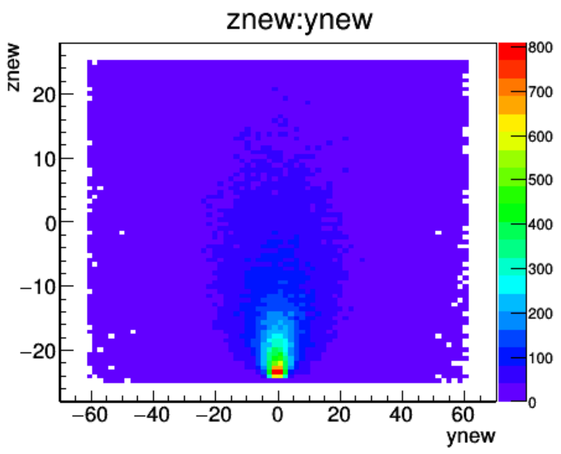


Figure 13: For the *xz* and *yz* planes the light map plots are also plotted with 80 bins in both directions. Note that this is not binned per cm, this will be changed in future plots.

Empty holes in these histogram plots may be attributed to border effects where the VUV photons are absorbed by the TPC walls and do not propagate through the LAr. It could also be that there are no events generated at these positions during the initial random spatial distribution, but this is not obvious as the binning is not done per cm.
The next steps are to split the TPC volume up into small 3D cube volumes called *voxels*. From this we can calculate the efficiency per voxel from the number of photons detected per voxel divided by the number of photons produced per voxel.
Area of the octagon = 11950 cm^2
Volume of the TPC = 597500 cm^3

3.4 Week 8 Summary: New light maps and MC Veto Presentation

3.4.1 MC Veto Team Presentation 12th Nov

I gave a presentation to the DarkSide-20k MC Veto Team on the 12th Nov about my work on the Optics task. The slides are uploaded on the INDICO DarkSide page. I presented my work from Week 5 onwards, from the point of creating the new geometry to the light maps I have at the moment.

Questions and Comments:

- There are 2 events with nph = 3. This may be a fault in the code. Paolo suggested that the conversion ratio from VUV to optical photons should be 1:1.
- Why use the TPC volume for the study, not a box? I said that this is best for time management as it is already a volume avaliable and defined in the *g4ds* repository.
- Instead of generating more statistics should we use a TPB coating to create more scattered events? Andrzej said this is not a sensible approach to first order as when TPB is included the results depend highly on geometry. Instead generate more events and move simulations to the GRID.

3.4.2 300 million VUV photon event simulation

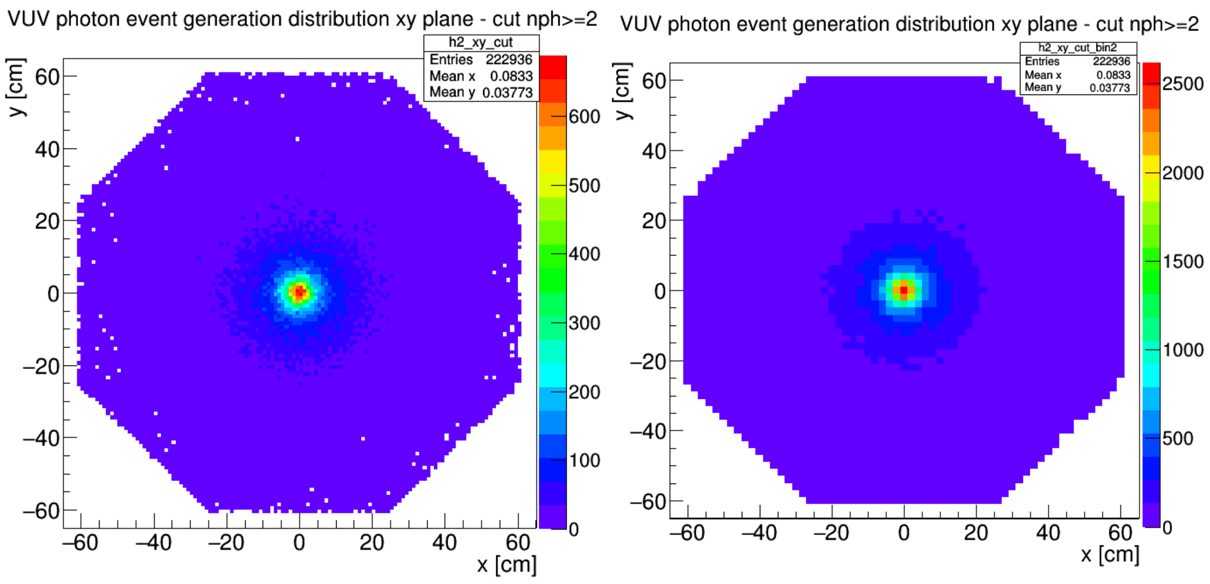


Figure 14: These graphs show the events that survive the cut throughout the entire TPC volume. Events that survive are most likely to be located at *x*,*y* = (0,0). The graph on the left is binned per 1cm, so 130 bins in each direction. The graph on the right is binned per 2cm, so 65 bins in each direction. The 2cm binning is more appropriate as this reduces the number of empty bins in the light map plots.

The plots in Figure 14 suggest that 2cm binning in the *x* and *y* directions is more appropriate that 1cm binning due to the holes that appear when the data is binned in the later way. Voxel sizes in these plots are (1cm x 1cm x 50cm) and (2cm x 2cm x 50cm) so the numbers given by the *z* scaling are the number of photons that are detected per voxel volume. To calculate an efficiency of photon detection we need to scale this to find the number of photons that survive per voxel. We can calculate efficiency by dividing the number of photons in each bin by the total number of voxels. In this case we do this by dividing each bin by the area of the octagon. For the plot with binning per 2cm plots this would be done by dividing each bin by a quarter of the area of the octagon.

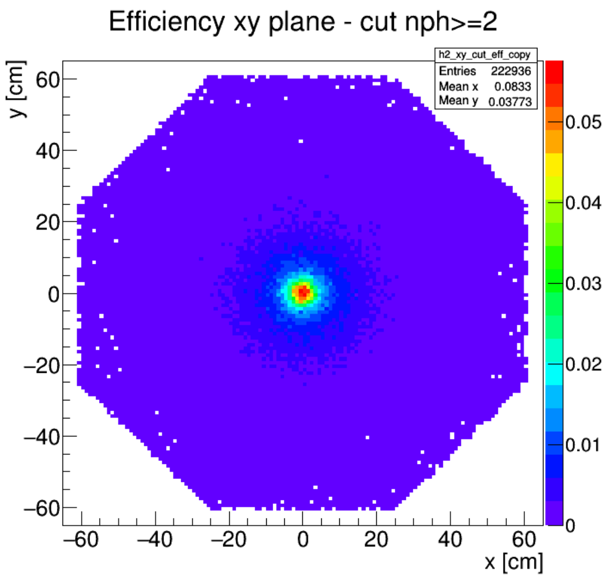


Figure 15: This plots show the number of photons per voxel that make it to the SiPM. The previous 1cm^2 bin plot has been divided by the area of the TPC in the *xy* plane as this is the number of voxels in this specific case.

The number of photons produced initially per 1cm^3 voxel is $\approx$ 500 photons per 1cm^3. The efficiency can be calculated from the ratio of nph$_{survive}$ and nph$_{init}$.

3.4.3 DarkSide Collaboration Meeting - Madrid

I presented at the DarkSide Collaboration Meeting hosted at Ciemat in Madrid. My talk was given on Sat 17th Nov during the series of talk given to on the topic of the DarkSide-20k Veto. The slides can be found on the INDICO DarkSide page. These slides are very similar to those used for the MC Team meeting on the Monday, except they include the 300 million data set and slices in the *z* plane.

3.5 Week 9 Summary:

Problems this week with the CVMFS repository update, i.e. cannot source the *configDarkSide_Manchester.sh* script as the path to the versions we were using has changed and the old software has been removed from the repository. The new repository path is:

```
/cvmfs/larsoft.opensciencegrid.org/products
```

We have updated to:

- geant4 v4_10_103a -q e10:prof
- root v6_06_04b -q e10:nu:prof
- g4photon v3_0

Files that have been changed and pushed to GitLab are:

- configDarkSide_Manchester.sh
- GNUmakefile

- configDarkSide_GridPP.sh

Grid Update:

We have the first working grid job that successfully retrieves the job outputs!!!
The *.jdl* file looks like:

```
[
JobName = "DS20k-g4sim-test";
JobGroup = "DS20K-g4sim-test";
InputSandbox = {"g4ds10_test_submission.sh","LFN:/vo.northgrid.ac.uk/user/n/
niamh.fearon/g4ds10_new.tar.gz"};
Executable = "g4ds10_test_submission.sh";
Arguments = "NameOfScript";
StdOutput = "StdOut";
StdError = "StdErr";
OutputSandbox = {"StdOut","StdErr", "outDS20k_bg.log","g4ds10/Linux-g++/
outDS20k_bg.root"};
OutputSE = "UKI-NORTHGRID-MAN-HEP-disk";
OutputData = "LFN:/vo.northgrid.ac.uk/user/n/niamh.fearon/g4ds10/Linux-g++/
outDS20k_bg.root";
]
```

3.6 Week 10 Summary: DIRAC Grid Distributed Computing System and new multi-device SiPM geometry

3.6.1 Large working DIRAC Grid job for simulating VUV photons

We have decided to use the DIRAC Grid Distributed Computing System to run large simulation jobs to get a large data set for the VUV photon event generation. More information about setting up the DIRAC user interface and general DIRAC commands can be found on the user guide website. We need to setup a grid proxy to use the commands that submit jobs to the grid:

```
$ source setupGrid.sh
```

I then had to add the tarball of the g4ds10 directory that I have been using locally on *hepgpu1* to a grid storage element (SE).

```
$ dirac-dms-add-file /vo.northgrid.ac.uk/user/n/niamh.fearon/g4ds10_new.tar.gz
g4ds10_new.tar.gz UKI-NORTHGRID-MAN-HEP-disk
```

It is possible to run many jobs at once using parametric job submission commands in the *.jdl* file. The file extension *jdl* stands for 'Job Description Language' and describes the submission criteria of the job. Including the parametric arguments as below generates 85 separate jobs, each with an individual Job ID and an assigned number. The %s instances are replaced by a number from 1-85. This file is called *parametric_jobs.jdl* in the *dirac_ui* directory. The shell script *g4ds10_test_submission.sh* is a combination of the old script of the same name, added to *runmulti.sh* and *cut.sh*.

```
[
JobName = "DS20k-g4sim-test_%n";
JobGroup = "DS20K-g4sim-test";
InputSandbox = {"g4ds10_test_submission.sh","LFN:/vo.northgrid.ac.uk/user/n/
niamh.fearon/g4ds10_new.tar.gz"};
Executable = "g4ds10_test_submission.sh";
Arguments = "%s";
Parameters = 85;
ParameterStart = 1;
ParameterStep = 1;
StdOutput = "StdOut_%s";
StdError = "StdErr_%s";
OutputSandbox = {"StdOut_%s","StdErr_%s","g4ds10/Linux-g++/run_files/
cut_root_files/total_cut_file_%s.root"};
OutputSE = "UKI-NORTHGRID-MAN-HEP-disk";
OutputData = "LFN:/vo.northgrid.ac.uk/user/n/niamh.fearon/g4ds10/Linux-g++/
run_files/cut_root_files/total_cut_file_%s.root";
]
```

To submit this job to the grid:

```
$ dirac-wms-job-submit parametric_jobs.jdl
```

To retrieve the outputs of this job:

```
$ dirac-wms-job-get-output --JobGroup DS20K-g4sim-test
```

3.6.2 Multi-device SiPM geometry

One of the outcomes of the Madrid meeting is that we have converged on a total number of channels for the two LAr buffers ($\approx$ 3000). The inner and outer coverage of SiPMs may be different and still need to be optimised. The two buffer configuration is efficient at tagging Gd-capture events. However, it is not optimised for captures on H in the TPC walls or captures on ^{40}Ar in the inner buffer. Measuring the energy deposited in the inner buffer only and setting a threshold at $\approx$ 800 keV would allow us to reach a background-free goal. The high threshold is required to suppress the ^{39}Ar background, which has an endpoint at 565 keV. The real detector will have poor light coverage and light collection non-uniformity which determine the leakage of ^{39}Ar events above threshold and induce a large dead time. One possible choice to mitigate this is the optical segmentation and the coverage of segment walls with TPB coated reflector.
Assuming we have 2000 5×5 cm^2 SiPM tiles covering the inner buffer, which is filled with 50t of LAr. The outer buffer is instrumented with 1000 SiPMs and can be neglected for the moment. This assumption corresponds to 4 SiPM tiles per 100 kg of LAr. This is approximately the equivalent to 4 SiPMs in a cubic volume of 0.4×0.4×0.5 m^3. From this geometry we want to determine the light yield and best resolution achievable and the related Data Acquisition (DAQ) requirements.
Creating this new geometry requires modification to the *DSDetectorDS20kTPC.cc* files:

- Transform the octagonal prism volumes into boxes by editing the number of edges in the *GVPolyhedra* functions.
- Repeat the SiPM geometry 3 more times and place the SiPMs on one surface.
- Define the TPB->reflector surfaces (dielectric_metal with some fixed reflectively around the LAr volume).
- Introduce the LAr scintillation properties in *DSMaterial.cc*, if not implemented already.

- Use the *SCS* generator to simulate ^{39}Ar decays uniformly distributed in this new volume.

First test:
I have reduced all of the sides in the G4Polyhedra functions from 8 to 4 to create cubic volumes. 4 SiPMs are located on the cathode floor in an equally spaced array, and the dimensions of the SiPMs are as before. The Stainless Steel walls have been switched back to TPB coated walls. The LAr scintillation properties are already implemented from the previous configuration.

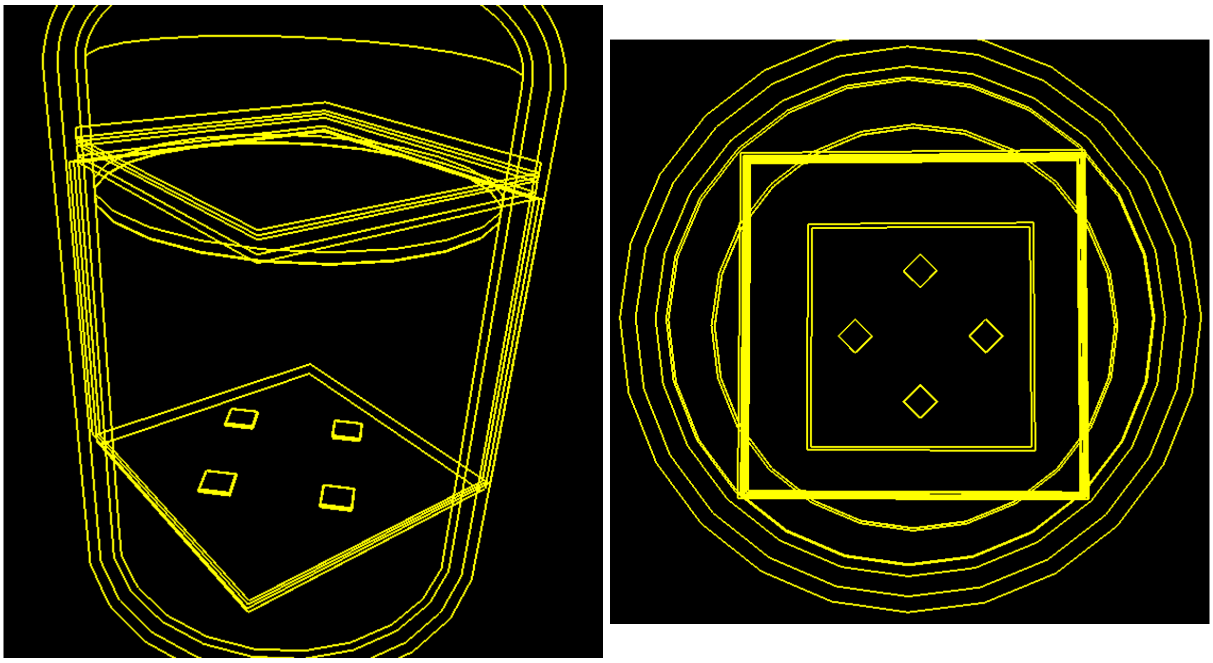


Figure 16: First look at the new geometry described to do the ^{39}Ar decay study. The dimensions for this configuration can be set by macro commands as in previous configurations.

Macro commands:

```
/ds/detector/configuration 12
/ds/detector/ds20cryo_tpcEdge 20 cm
/ds/detector/ds20cryo_tpcHeight 50 cm
```

Advice from Paolo from first look:
The geometry is almost ready. There are some overlaps between the inner cubic volume and the surrounding volumes (the steel cryostat) which we can remove. The optical properties of the TPB anode and cathode surfaces need to also be changed to be reflective instead of transparent.

3.7 Week 11 Summary: Editing the Multi-device SiPM geometry

Overview of the current geometry:

There are two LAr volumes fPhysicActiveLAr and fPhysicTopCryoFiller. fPhysicTopCryoFiller sits on top of fPhysicActiveLAr, and is only 1 cm thick (with the dual-phase layout, this is the volume filled with the gaseous argon). fPhysicTPBSide surrounds the ActiveLAr in all directions but the top surface, fPhysicTPBTop is the TPB coating of the top window.
The optical surfaces already defined are:

- fPhysicActiveLAr <—-> fPhysicTPBSide (bi-directional)

- fPhysicTPBSide —-> fPhysicTeflonBottom (uni-directional, dielectric)

Using the two surface above already defined in *DSDetector20kTPC.cc*, add the missing surfaces:

- fPhysicTopCryoFiller <—-> fPhysicTPBTop (bi-directional)
- fPhysicTPBTop —-> fPhysicTopWindow (uni-directional, dielectric_metal)
- fPhysicTPBSide —-> fPhysicBotWindow (uni-directional, dielectric_metal)

The overlapping volumes are constructed in the

```
if(!IsAlternateDesign)
```

block starting at the line 300 in *DSDetectorDS20kTPC.cc*.
To remove all unnecessary overlapping volumes add the following lines above line 300.

```
IsAlternateDesign = true;
isNoCuVessel = true;
```

This selects the last else block at line 432 and the LAr box is placed directly in the fMotherVolume. However, we encountered an error when running *viswrl.mac*. It was complaining about rotating the fMotherVolume. We initially solved this by removing the line:

```
fMotherVolume -> SetRotation(myDefaultRotation2);
```

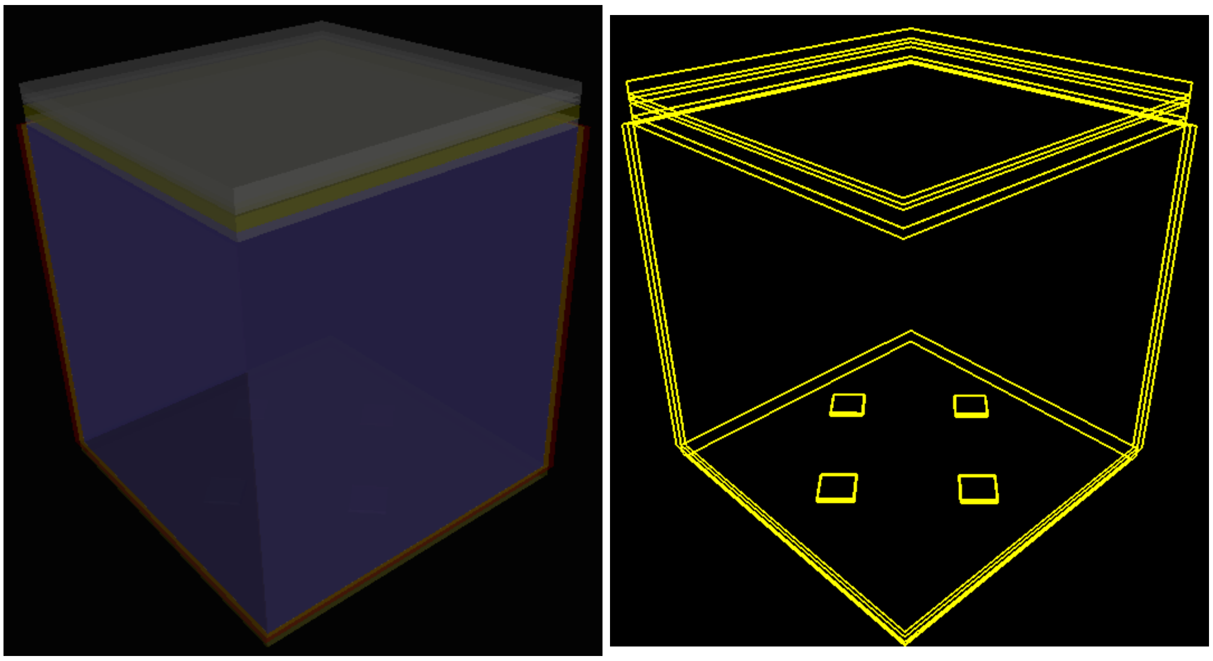


Figure 17: New schematics of the box of LAr with 4 SiPMs placed on the cathode floor. The box is placed directly in the fMotherVolume in this configuration. The geometry is created with the same macro commands as listed in the section above.

When trying to simulate some simple events using *run.mac* to see if the light simulation works correctly we ran into problems. When the events that are being simulated reach the air world volume the Monte Carlo simulation breaks. We debugged this by printing the material using *G4OpBoundaryProcess.cc*. We then tried to change the world volume to be made of non-scintillating LAr in *DSDetectorConstruction.cc* and then ran into issues with *DSTrackingAction*. We have had a similar issue to this before and solved it by implementing the ProtoDUNE cryostat to be an outer volume that encases the DarkSide-20k Detector. However this did not work either.

```
DSDetectorDuneCryostat* ProtoDuneCryo = new DSDetectorDuneCryostat(fPhysicWorld)
new DSDetectorDS20kTPC(ProtoDuneCryo->GetDetectorComponent());
```

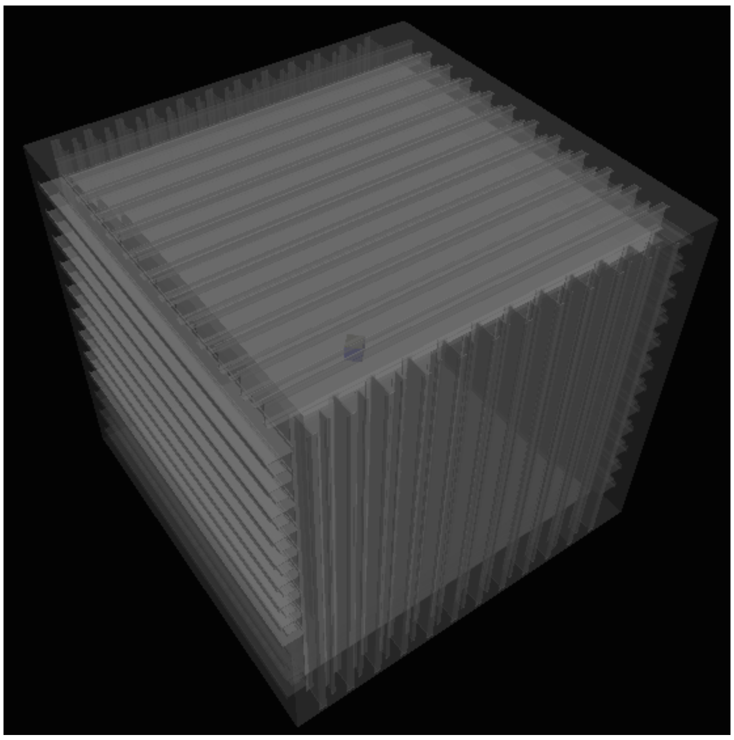


Figure 18: Test of creating the ProtoDUNE cryostat outside of the LAr box. This was not a successful fix.

Solutions from Paolo:

- Outermost material must be made of air, otherwise this causes an error in *DSSteppingAction.cc*.
- Remove completely the gas pocket in the dual-phase TPC layout. This simplifies the problem.
- Create a new Tubs volume to solve verbose output problem.
- Group all relevant optical surface definitions into two blocks at the bottom of *DSDetectorDS20kTPC.cc*.
- The reflectively of the surfaces behind the TPB can be changed in *data/detector/DSOptics.dat*. Parameter TeflonTPBVisRef is now set to 0.98.
- SiPMs are not in the optimal position with the current rotation.
- Possible debugging may be required.

Need to include pre-lim check graphs from Paolo

3.8 Week 12 Summary:

3.8.1 Continuation of multi-device SiPM geometry

We need to optimise the placement of the 4 SiPMs to place them in a "sensible" array. Rotate them by 45° with respect to the z axis so that they are aligned with the edges of the LAr TPC box. This however changes the coordinate system of xy plane due to this rotation. We must account for this when placing the SiPMs using *G4ThreeVector*.

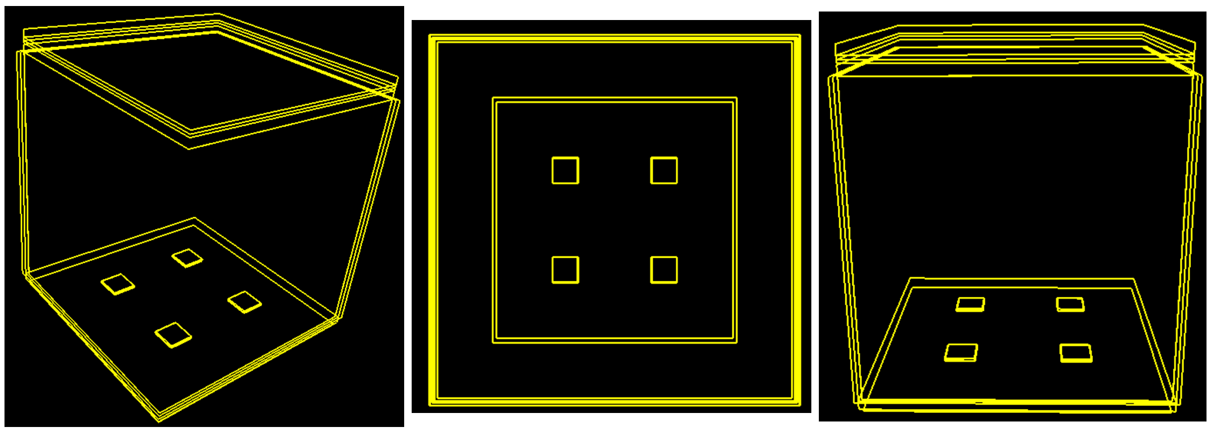


Figure 19: The 4 SiPMs are now rotated and placed so that they are aligned with the edges of the LAr TPC box configuration. This configuration will not matter so much with the TPB reflector foils coating the TPC walls due to the high reflectively and the detection probability with be uniform across the volume. In this case there should be little sensitivity to SiPM position.

Tasks:

1. Light maps and efficiency plots with the 2 billion event root file for VUV photons in the single SiPM geometry with the octagonal DarkSide-20k TPC.
2. LAr TPC box with no TPB walls (Stainless Steel), store x,y,z,nph,npe ROOT variables. Cut for nph $\geq$ 2.
3. LAr TPC box with TPB walls, store x,y,z,nph,npe ROOT variables. Cut for npe $\geq$ 1.
4. LAr TPC box with TPB walls, generate ^{39}Ar events using *scs.mac*. Calculate the light yield.
5. Generate Marley Supernova events in the LAr TPC box. How far from a single SiPM are you likely to see it?

3.8.2 Report Plan Feedback

First plan feedback from Andrzej

- Introduction is best written at the end. This should include why the project is interesting and also what is included in the whole report.
- The Theory section should be where a large amount of "meat" should be. Split this section into Dark Matter Theory and the DarkSide-20k Detector.

- Mention what G4DS is and that it is based on G4.
- Mention the Dirac GRID, possibly a small subsection of 1 or 2 pages.
- For the test geometries, consider doing a number of small test studies:
    1. Generate a Marley SN event, how far from the SiPM can these events be detected?
    2. If a neutron deposits N MeV, how likely is it to be detected vs. distance from the SiPM?
- This adds some physics to the plots. Include these studies if I have time.

This document has been typeset using LaTeX.