## neBEM - a surface discretization approach


$3^{\text {rd }}$ RD 51 Collaboration meeting 16 June, Kolympari, Crete

## Surface discretization - a closer look



Elements of various sizes and aspect ratios can be used
$>$ Field variation around the anode is of importance - very small elements
$>$ The lines join the element centroids
It is necessary to supply only the interface and the information on adjoining volumes

## neBEM - the solution procedure



- Consider a system of two conductors, each having been discretized into two elements.
- In order to generalize the situation, let us even consider one of the conductors to be floating. Thus, one of these conductors is at a known voltage, V. The other conductor is at a floating voltage $\mathrm{V}_{\mathrm{F}}$, which is unknown.
- Number the elements on the conductor with known voltage to be 1,2 , and those on the floating conductor to be 3, 4.
- Denote charge densities by $\rho_{i}$, area by $A_{i}$, on each element
- Resulting system of equation is as shown - the last equation reflecting the fact that the total charge on a floating conducting object is zero
- In the above system, $I_{\mathrm{ij}}$ denotes the influence of the $j^{\text {th }}$ element on the $i^{\text {th }}$ element.
- Please note that if we have more than one floating conductor, they cannot be assumed to be at the same potential, and one column and one row as shown above needs to be added for each floating conductor.
Please note that the matrix is sparse.


## neBEM as a toolkit

It is now easier to use, interface and integrate neBEM
Stand-alone
A driver routine
An interface routine
Post-processing, if necessary
Charge density at all the interfaces
Potential at any arbitrary point
Garfield
Garfield prompt
Garfield script
ROOT
ROOT prompt
ROOT script
py-ROOT script
Flux at any arbitrary point
Capacitance, forces on device
 components properties can be obtained by post-processing

Other interfaces can also be developed easily - specially exciting could be experimental and CAD interfaces

Following slides contain few examples of the above

## neBEM as a toolkit - stand-alone application

```
// argv[1] is a MotherInFile
int main(int argc, char *argv[])
{
int PostProcessing(void); // a typical post-processing function
if(argc != 2)
    {
    printf("Usage:\n\tmuMegas <MotherInputFile>\n");
    exit(-1);
    }
```

// The following function allows input files to be read in from files int fstatus = neBEMGetInputsFromFiles(argv[1]); assert(fstatus ==0);
// beginning of neBEM computation - assign the state variable neBEMState $=0$;
fstatus = neBEMInitialize(OutDir, DebugLevel); assert(fstatus == 0);
// The following function uses neBEMGetNbPrimitives, neBEMGetPrimitive, // neBEMVolumeDescription and neBEMVolumePoint, the last one used sparingly fstatus = neBEMReadGeometry(); assert(fstatus == 0);

## neBEM as a toolkit - stand-alone application

// It is likely that a routine will be supplied either by neBEM or the user
// to analyze the pritimives and to decide the size and shape of elements
// necessary on each primitive
int **elementNbs;
elementNbs = imatrix(1, NbPrimitives, 1, 2);
for(register int prim = 1; prim <= NbPrimitives; ++prim)
\{
elementNbs[prim][1] = tmpNbXSegs[prim];
elementNbs[prim][2] = tmpNbZSegs[prim];
\}
fstatus = neBEMDiscretize(elementNbs);
assert(fstatus == 0);
// Since a boundary condition is likely to be
// associated with each element, this routine can be called only after the // primitives have been discretized
fstatus = neBEMBoundaryConditions();
assert (fstatus == 0);
assert(fstatus == 0);

## neBEM as a toolkit - stand-alone application

```
// A flag needs to be set and transmitted to neBEMSolve to indicate whether
// the matrix inversion for this geometry has been carried out already
if(!PostProcess)
    {
    int inverted = 0; // assume that this is a fresh calculation
    fstatus = neBEMSolve(inverted); assert(fstatus == 0);
    }
else
{
int inverted = 0; // A temporary solution - this has to be a user input
if(inverted) // skips matrix inversion step ,but since the boundary conditions have changed
    { // read in the inverted matrix and find new solution
        fstatus = neBEMSolve(inverted); assert(fstatus == 0);
        }
    else // read in the solution vector directly
        {
        fstatus = neBEMReadSolution(); // reads in the solution vector directly
        assert(fstatus == 0); // implying no change in the boundary condns
        }
}
```


## neBEM as a toolkit - stand-alone application

// Post-processing can be carried out with separate function(s) written within // this code, or function(s) residing in a separate source code, or a mix. fstatus $=$ PostProcessing(); assert(fstatus $==0$ );
// End neBEM gracefully
fstatus = neBEMEnd(); assert(fstatus $==0$ );
return 0;
\} // main ends
// As needed
int PostProcessing(void)
\{
double xpos, ypos, zpos, potential; Point3D point; Vector3D field;
point. $\mathrm{X}=0.0 ;$ point. $\mathrm{Y}=0.0$; point. $\mathrm{Z}=0.0$;
fstatus = neBEMField(point, \&potential, \&field); assert(fstatus == 0);
// print or carry on other analysis
\} // Post-Processing ends
neBEM - stand-alone: Example analysis of a ThGEM


## Preprocess :

-Device definition using various primitives / surfaces -Discretization of primitives into triangular or rectangular elements -Preparation of input files for neBEM with geometrical and electrical parameters of the elements


## neBEM as a toolkit - stand-alone application

neBEM gnuplot VIEWER

wowish

## neBEM as a toolkit - stand-alone application

Flux surface close to mesh


## neBEM integrated to Garfield

\&CELL
Global metal=0.0100
Global g10=0.1
Global hole=0.0250
solids
box centre 0 0-1
half-lengths 1.01 .00 .1 ...
conductor voltage -2000
hole centre $00\{-\mathrm{g} 10 / 2-\mathrm{metal} / 2\}$ half-lengths 1.01 .0 \{metal/2\} radius \{hole\} ... conductor voltage $-750 n=3$
hole centre 000 half-lengths 1.01 .0 \{g10/2\} radius \{hole\} ...
dielectric eps=4 n=3
hole centre $00\{\mathrm{~g} 10 / 2+\mathrm{metal} / 2\}$ half-lengths 1.01 .0 \{metal/2\} radius \{hole\} ... conductor voltage $750 n=3$
box centre 001 half-lengths 1.01 .00 .1 ... conductor voltage 3500

## neBEM integrated to Garfield

```
&FIELD
area -0.1 -0.1 -0.1 0.1 0.1 0.1 view }\textrm{x}+2*\mathrm{ * y +3*z=0 3d
pl vect
```

area $-0.1-0.1-0.10 .10 .10 .1$ view $\mathrm{x}=0$ 3d
pl vect
area view $\mathbf{z = 0}$
pl surf v
area view $\mathrm{x}=0$
pl surf v
track 00-1001
pl gr v
pl gr ez

## neBEM integrated to Garfield





## neBEM integrated to Garfield



## neBEM interfaced to ROOT

\#Final Volumes making up the device
\#The XY plane is co-incident with the bottom of the substrate \#volAnode
\#heights: anode strips are placed on the top surface of the substrate heightSubstrate = thicknessSubstrate/2.0
heightAnodeStrip = thicknessSubstrate + thicknessAnodeStrip/2.0
\#Anode volume is made of the epoxy substrate and a number of anode strips
nStrips $=\operatorname{int}(($ lengthDevice-2.* widthAnodeStrip)/(pitchAnodeStrip))
volAnode = TGeoVolumeAssembly('volAnode')
volAnode.AddNodeOverlap(volSubstrate, 1, TGeoTranslation( 0., 0., heightSubstrate), "'")
\#One of the anode strips is right at $(0, Y, 0)$
volAnode.AddNodeOverlap(volAnodeStrip, 2, TGeoTranslation(0., 0.0, heightAnodeStrip), ""')
\#Place anode strips in the +ve Z direction
for i in range(1, nStrips/2):
volAnode.AddNodeOverlap(volAnodeStrip, 2 + i, TGeoTranslation(0., i*pitchAnodeStrip, heightAnodeStrip), "") \#Number of nodes laid till now: Substrate+CentralStrip+(nStrips/2)
$\mathrm{nb}=1+1+(\mathrm{nStrips} / 2)-1$
\#Place anode strips in the -ve Z direction
for i in range(1, nStrips/2):
volAnode.AddNodeOverlap(volAnodeStrip, nb + i, TGeoTranslation(0., -i*pitchAnodeStrip, heightAnodeStrip), "")

## neBEM interfaced to ROOT

\#volCathode
\#Cathode volume is made of a PCB and a metal cathode- - simple volumes and
heightMetalCathode $=$ thicknessSubstrate + thicknessAnodeStrip + gapAmplification + thicknessMesh + gapDrift

+ thicknessMetalCathode/2.
heightPCB = thicknessSubstrate + thicknessAnodeStrip + gapAmplification + thicknessMesh + gapDrift + thicknessMetalCathode + thicknessPCB/2.
volCathode $=$ TGeoVolumeAssembly('volCathode')
volCathode.AddNodeOverlap(volMetalCathode, 1, TGeoTranslation(0., 0., heightMetalCathode), "'") volCathode.AddNodeOverlap(volPCB, 2, TGeoTranslation(0., 0., heightPCB), "") \#The microMEGAS (volmicroMEGAS) is composed of several volume elements: \#a volAnode, a 2D matrix of volMeshUnits and a volCathode volmicroMEGAS = TGeoVolumeAssembly('volmicroMEGAS') volmicroMEGAS.AddNode(volAnode, 1, TGeoTranslation(0., 0, 0.0)) volmicroMEGAS.AddNode(volCathode, 2, TGeoTranslation(0., $0 ., 0.0)$ )
\#The first mesh unit is at $(0,0,0)$
volmicroMEGAS.AddNode(volMeshUnit, 3, TGeoTranslation(0., 0., 0.))
volTop.AddNode(volmicroMEGAS, 1, TGeoTranslation(0., 0., 0.))
geoManager.CloseGeometry() volTop.Draw()


## neBEM interfaced to ROOT

After creating the desired device, the user needs to execute another macro,

> neBEMInputFiles.C
which in turn calls the macro
ExtractPrimitives.C
This gives the required interfaces between volumes with all the necessary properties.

At present, it possible to extract primitive information for simple structures such micromegas.
Complex devices are being worked upon.

[^0]

## Particles on Surface (ParSue)

## An improved model to represent space charge



Possible through the use of neBEM formalism


## Space charge Particles on Surface (ParSue)




- Both potential and field within the cell has been estimated far more accurately by ParSue than the PIC

PARticles on SUrfacE (PARSUE) seems to be the new model to pursue!!

## Charging dynamics, magnetostatics

These are some of the other areas that could prove useful and need attention
Charging dynamics can be modeled using a quasi-static approach
Magnetostatics should be an easy extension of the existing formulation
We hope to report some advancement in the next collaboration meeting, especially on the former

## Periodicity, mirroring, efficiency

-Periodicity has been implemented - that should save computational time of a significant amount. It is naturally assumed that the repeated primitives are exact replicas of the base primitive in every aspect, i.e., including the charge density distributed on the primitive elements. Effectively, this approach defies the existence of a finite boundary of the problem.
Periodicity, at present, is assumed to be the same for all the primitives. This will be made primitive-specific in a very near future release.
'However, we plan to explore how accurately this model can be used to study effects of edges / boundaries.
-Mirroring is expected to be effective for several MPGD geometries. We plan to implement mirroring very soon. This will need algorithms similar to the periodicity codes and, hence, should not be difficult.
-We are aware of efficiency -related problems. We know the cause (branchcuts in transcendental functions )of this problem and expect to correct it quite soon using special expressions to avoid the numerical difficulties.

## Conclusions and future plans

neBEM is expected to provide us with one of the important missing pieces in the projected simulation framework
oWritten in a toolkit fashion, it is available for users to be used in a standalone fashion, or interfaced / integrated with other codes
${ }^{*}$ Present version seem to provide us with reliable and precise estimation of electrostatic configuration
${ }^{\text {Efficiency issues need to be resolved - especially those related to the }}$ evaluation of the foundation expressions near branch-cuts
-Dynamics charging is an issue that can be tackled using the same formulation in a quasi-static fashion - needs lot of work though
"Space charge can be modeled in a more accurate fashion using ParSue -proof-of-concept seems to be successful. Needs implementation
-Magnetostatics is another aspect that should be easily tackled using a similar formulation
${ }^{\bullet}$ Documentation is in a very bad shape - we plan to put good effort into this within the coming couple of weeks
UUser feed-backs will be essential in further improvements

## Acknowledgments

We happily acknowledge the support extended by
Prof. Bikas Sinha, Director, SINP
Department of Science and Technology, Govt of India



[^0]:    winverit

