`topIIvol_DistMesher`

This is a tool to create embarrassingly parallel distributed meshes. The mesher takes in a point-cloud as an input (`.xyz`) and outputs distributed mesh.

This is tool to created distributed mesh from partitioned point cloud

- Examples 3D partitioning of distributed mesher producing `*.mesh` mesh with 24 MPI ranks (with 24 subdomains divided between x, y and z directions):

```
mpirun -n 24 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 2 --partition_y 3 --partition_z 4 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

or

```
mpirun -n 24 topIIvol_DistMesher  --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 3 --partition_y 2 --partition_z 4 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

or

```
mpirun -n 24 topIIvol_DistMesher  --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 2 --partition_y 2 --partition_z 6 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 2D partitioning of distributed mesher producing `*.mesh` mesh with 8 MPI ranks (with the 8 subdomains divided between x and y directions):

```
mpirun -n 8 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 2 --partition_y 4 --partition_z 1 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

or

```
mpirun -n 8 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 4 --partition_y 2 --partition_z 1 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 2D partitioning of distributed mesher producing `*.mesh` mesh with 6 MPI ranks (with the 6 subdomains divided between x and z directions):

```
mpirun -n 6 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 2 --partition_y 1 --partition_z 3 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

or

```
mpirun -n 6 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 3 --partition_y 1 --partition_z 2 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 2D partitioning of distributed mesher producing `*.mesh` mesh with 16 MPI ranks (with the 16 subdomains divided between y and z directions):

```
mpirun -n 16 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 1 --partition_y 8 --partition_z 2 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

or

```
mpirun -n 16 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 1 --partition_y 2 --partition_z 8 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

or

```
mpirun -n 16 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 1 --partition_y 4 --partition_z 4 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 1D partitioning of distributed mesher producing `*.mesh` mesh with 4 MPI ranks (letting the algorithm decide the partition direction):

```
mpirun -n 4 topIIvol_DistMesher --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 1D partitioning of distributed mesher producing `*.mesh` mesh with 4 MPI ranks (enforced partitioning in x direction):

```
mpirun -n 4 topIIvol_DistMesher  --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 4 --partition_y 1 --partition_z 1 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 1D partitioning of distributed mesher producing `*.mesh` mesh with 8 MPI ranks (enforced partitioning in y direction):

```
mpirun -n 8 topIIvol_DistMesher  --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 1 --partition_y 8 --partition_z 1 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

- Examples 1D partitioning of distributed mesher producing `*.mesh` mesh with 3 MPI ranks (enforced partitioning in z direction):

```
mpirun -n 3 topIIvol_DistMesher  --zpoints 50 --xpoints 32 --ypoints 29 \
--depth -1000 --partition_x 1 --partition_y 1 --partition_z 3 \
--out top-ii-vol-mesh  --in ./../etc/DEM_160m
```

Command-line option definitions

Option	Type	Comment
`--xpoints`	`[int]`	These are # of x points present in your point cloud.
`--ypoints`	`[int]`	These are # of y points present in your point cloud.
`--zpoints`	`[int]`	These are # of z points intended in the z direction.
`--partition_x`	`[int]`	These are # of x partitions in x direction.
`--partition_y`	`[int]`	These are # of y partitions in y direction.
`--partition_z`	`[int]`	These are # of z partitions in z direction.
`--in`	`[string]`	Sting to provide the input point cloud file `.xyz`
`--out`	`[string]`	Sting to provide the output mesh file `.mesh`
`-np`	`[int]`	Provide the # of MPI ranks.

To report bugs, issues, feature-requests contact:*

- **mohd-afeef.badri@cea.fr**
- **mohd-afeef.badri@hotmail.com**
- **mohd-afeef.badri@etu.univ-nantes.fr**