OpenVDB Implicit Field Plugin

> ⚠ This documentation is intended for developers working directly with the RenderMan interface or RIB files.

The OpenVDB Implicit Field Plugin impl_openvdb.so supports the use of OpenVDB files, either as implicit surfaces or as volumes. Two string arguments are required (a third argument is optional, as described below). The first argument is the name of the OpenVDB file. The second argument is the name of the grid which will be used as the source of the field function. This grid must be of float type, and can be either a level set (getGridClass() == openvdb::GRID_LEVEL_SET) or a fog volume (openvdb::GRID_FOG_VOLUME). If the grid is a level set, it will be converted to a fog volume where the interior voxels have a field function value of 1, the exterior voxels have a value of 0, and the narrow-band voxels have values varying linearly from 0 to 1. If the field function grid's class is not specified in the OpenVDB file, or is neither a level set nor a fog volume, an override can be supplied by appending ":levelset" or ":fogvolume" to the name of the grid.

The plugin supports rendering of OpenVDB data both as a surface (using RiBlobby) and a volume (RiVolume). Here is an example of the syntax for rendering the OpenVDB dragon dataset as a surface.

```
Blobby 1 [1004  0 0 0 2 1] [0] ["${RMANTREE}/lib/plugins/impl_openvdb.so/impl_openvdb.so" "dragon.vdb"
"ls_dragon"]
```

And here is an example of the syntax for rendering the OpenVDB dragon dataset as a volume.

```
Volume "blobbydso:${RMANTREE}/lib/plugins/impl_openvdb.so" [-99 103.4 -41.6 49.6 -57.6 77.3]
    [0 0 0] "constant string[2] blobbydso:stringargs" ["dragon.vdb" "ls_dragon"]
```

Deformation motion blur is supported by supplying the name of a grid containing velocity data in the optional third string argument. This grid must of type op envdb::Vec3f, and must be a fog volume. (If the velocity data's grid class is set otherwise in the OpenVDB file, it can be overriden by appending ":fogvolume" to the name of the grid.) Here is an example of the syntax for rendering a OpenVDB dataset containing a volume whose field function is contained in the grid named "density", and a velocity in the grid named "velocity".

```
Volume "blobbydso:${RMANTREE}/lib/plugins/impl_openvdb.so" [-30 30 -30 30 -30 30]
    [0 0 0] "constant string[3] blobbydso:stringargs" ["sphere.vdb" "density"
    "velocity"] "constant float[2] time" [0 1] "varying vector[1] dPdtime" []
```

Primitive variables are supported and are bound to grids of the same name in the OpenVDB file. RenderMan primitive variables of type color, vector, point, and normal are bound to grids of type openvdb::Vec3fGrid while float variables are bound to grids of type openvdb::FloatGrid. In the following example, we use three grids (density, fuel, and glow) to drive the field function and two attached primvars; the result of vdb_print file.vdb is shown as well.

```
Volume "blobbydso:${RMANTREE}/lib/plugins/impl_openvdb.so" [-100 100 -100 100 -100 100]
    [0 0 0] "constant string[2] blobbydso:stringargs" ["file.vdb" "density"]
     "varying float fuel" [] "varying color glow" []

fuel         float -100 -100 -100  100 100 100  200x200x200   1.8MVox 17.1MB
density      float -100 -100 -100  100 100 100  200x200x200   1.8MVox 17.1MB
glow         vec3s -100 -100 -100  100 100 100  200x200x200   1.8MVox 51.2MB
```

By default, the plugin performs unfiltered lookups of voxel data. It can optionally also perform fully filtered lookups of voxel data in the OpenVDB file in order to avoid aliasing. Because OpenVDB does not currently support a pre-filtered hierarchical representation, internally a mipmap of voxels is maintained with downsampled levels generated on the fly as needed, as a means of accelerating the filtered lookups. Generating this mipmap will incur some overhead in time, especially if a large grid must be filtered down to only a few voxels. There is also some memory overhead to storing these mipmaps, but due to their nature the total memory should never amount to more than twice the memory needed by the base unfiltered grid.

If filtering is enabled, the default width of the filter used over OpenVDB voxels favors antialiasing over detail (the DSO picks the maximum dimension of a PRMan microvoxel); this may provide a blurry result if the PRMan microvoxels are very anisotropic due to a high relative shadingrate. This blurriness can be alleviated at the cost of potential aliasing by passing an optional float argument to the DSO: this argument will be used as a multiplier on the filterwidth. The following is an example of how to pass a filterwidth of 0.5 to the DSO.

```
Volume "blobbydso:${RMANTREE}/lib/plugins/impl_openvdb.so" [-99 103.4 -41.6 49.6 -57.6 77.3]
    [0 0 0] "constant string[2] blobbydso:stringargs" ["dragon.vdb" "ls_dragon"]
    "constant float[1] blobbydso:floatargs" [0.5]
```

Filtered lookups (and mipmap generation) can be entirely disabled by setting the filterwidth to 0, which is the default setting.

By default, the velocity data is used directly by the plugin without any further transformation. Should the magnitude of the velocity vectors be inappropriate, a second optional float argument can be passed to the DSO which will be used as a multiplier on the velocity data. Note that if this float argument is used the filter width MUST also be specified. The following is an example of how to pass a velocity scaling argument of 0.25 to the DSO for use by a RiBlobby.

```
Blobby 1 [1004 0 2 0 3 1] [1 0.25] ["${RMANTREE}/lib/plugins/impl_openvdb.so"
    "water.vdb" "surface" "vel:fogvolume"]
```