kernel_device.h File Reference

Go to the source code of this file.

Functions

__global__ void **newthreshold** (float *d_threshold, cudaTextureObject_t texObjsurface, float threshold, int res)
this method is to adjust the surface offset of a model More...

__global__ void **resolution_kernel** (float *d_newres, cudaTextureObject_t texObj3, int res, int zres)
this method is to adjust the resolution of SDF model More...

__global__ void **collision_render** (bool *d_collision, cudaTextureObject_t texObj, cudaTextureObject_t texObj1, cudaExtent volumeSize)
this method is to render a red background if collision detected More...

__global__ void **d_rendermoveskeleton** (int *d_belongcluster, int clusternum, bool segmentation, bool pivotget, float3 pivot3d, float4 *skeletonarray, **uint** *d_output, **uint** imageW, **uint** imageH, float brightness, cudaExtent volumeSize, int originaldim, **uint** skeleton, int No_skeleton, float xmin, float xmax, float ymin, float ymax, float zmin, float zmax, bool selection)
this method is to render when the skeleton are transforming More...

__global__ void **d_renderskeleton** (bool color, int *d_internalneighborcount, int *d_belongcluster, int clusternum, bool segmentation, bool pivotget, float3 pivot3d, float4 *skeletonarray, **uint** *d_output, **uint** imageW, **uint** imageH, float brightness, cudaExtent volumeSize, int originaldim, **uint** skeleton, int No_skeleton, float xmin, float xmax, float ymin, float ymax, float zmin, float zmax, bool selection)
this method is to render the skeleton More...

__global__ void **d_render** (int sweepfun, bool sweepconsrender, bool sweeprender, float maxnegdis, cudaTextureObject_t texObj, cudaTextureObject_t texObj1, cudaTextureObject_t texObjplate, **uint** *d_output, **uint** imageW, **uint** imageH, float brightness, float threshold, float threshold1, float threshold2, **uint** objn, float tolerance, bool collision, cudaExtent volumeSize, int originaldim, int spefun)
this method is for models before boolean operations More...

__global__ void **d_renderbool** (int sweepfun, bool sweepconsrender, cudaTextureObject_t texObj, **uint** *d_output, **uint** imageW, **uint** imageH, float brightness, float threshold, **uint** objn, float tolerance, float maxnegdis, float xcross, float ycross, float zcross, cudaExtent volumeSize, int originaldim, float objcoorminx, float objcoormaxx, float objcoorminy, float objcoormaxy, float objcoorminz, float objcoormaxz, **uint** material)
this method is for models after boolean operations More...

__global__ void **levelset_kernel** (bool *d_needfix, cudaSurfaceObject_t SurfObj1, cudaSurfaceObject_t SurfObj2, cudaExtent volumeSize, int originaldim, float inf, float boundary)
this method is to use upwind differencing fixing distance field More...

__global__ void	**copysurface_kernel** (cudaSurfaceObject_t SurfObj2, cudaSurfaceObject_t SurfObj1, cudaExtent volumeSize) this method is to copy texture memory from SurfObj1 to SurfObj2 More...
__global__ void	**flipandset_kernel** (cudaSurfaceObject_t SurfObj1, cudaSurfaceObject_t SurfObj2, cudaExtent volumeSize, float inf) this method is to takes surface memory of SurfObj2 and flip the sign. Setting the positive valu as infinite More...
__global__ void	**set_kernel** (cudaSurfaceObject_t SurfObj1, cudaSurfaceObject_t SurfObj2, cudaExtent volumeSize, float inf) this method is to takes surface memory of SurfObj2 and Setting the positive value as infinite More...
__global__ void	**repeat** (float *d_volume, cudaTextureObject_t texObj, cudaTextureObject_t texObj1, cudaExtent volumeSize, float threshold1, float threshold2) this method is to union texObj with threshold1 and texObj1 with threshold2 More...
__global__ void	**green_dif_blue** (float *d_volume, cudaTextureObject_t texObj, cudaTextureObject_t texObj1, cudaExtent volumeSize, float threshold1, float threshold2) this method is to apply difference operation on texObj with threshold1 and texObj1 with threshold2 More...
__global__ void	**sweepcons_bool** (int blend, float k, float *d_volume, cudaTextureObject_t texObj, cudaTextureObject_t texObj1, cudaExtent volumeSize, float threshold1, float threshold2) this method is to apply union operation on texObj with threshold1 and texObj1 with threshold2 More...
__global__ void	**skeletonincluster** (int cluster_No, float4 *d_volume, int *d_belongcluster, float4 *skeletonarray_total, int No_skeleton_total) this method is to copy skeleton with certain ID to d_volume More...
__device__ float	**solvequadratic** (cudaSurfaceObject_t SurfObj, int x, int y, int z, cudaExtent volumeSize, int originaldim) this method is to solve eikonal equation with upwind differencing More...
__global__ void	**initsurf** (cudaSurfaceObject_t SurfObj1, cudaSurfaceObject_t SurfObj2, cudaExtent volumeSize, int originaldim, int No_skeleton) this method is to initialize the surface memory More...
__global__ void	**flipandcopy_kernel** (float *d_volume, cudaSurfaceObject_t SurfObj1, float threshold, cudaExtent volumeSize) this method is to flip the dsitance from SurfObj2 and copy to d_volume More...
__global__ void	**copyfromtextodevice_uint** (**uint** *d_skeleton, cudaTextureObject_t texObjskeleton, cudaExtent volumeSize, int originaldim) this method is to convert positive distance to 1 and negatrive to 0 More...
__global__ void	**dis2material_kernel** (float *d_volume, cudaTextureObject_t texObjsurface, int sweepfun, cudaExtent volumeSize, int originaldim, float maxnegdis, float objcoorminx, float objcoormaxx, float objcoorminy, float objcoormaxy, float objcoorminz, float objcoormaxz)

this method is to convert distance value to material value More...

__global__ void	**material_function** (float *d_newres, int res, int zres, int material, float maxnegdis, float objcoorminx, float objcoormaxx, float objcoorminy, float objcoormaxy, float objcoorminz, float objcoormaxz) this method is to convert distance value to material value More...
__global__ void	**distance_data_cpu** (float *d_skeleton, cudaTextureObject_t texObjskeleton, cudaExtent volumeSize, int originaldim) this method is to copy skeleton data from texture memory to global memory More...
__global__ void	**skeleton_data_share** (**uint** *d_skeleton2, float *skeleton2, cudaExtent volumeSize, int originaldim, float exterial_threshold, float interial_threshold, float exterial_dis, float interial_dis) this method is to compute the gradient field and label skeleton data More...
__global__ void	**copyskeletontotal** (**uint** *d_array, **uint** *scan_array, float4 *skeletonarray_total, cudaTextureObject_t texObjskeleton, cudaExtent volumeSize) this method is to compact skeleton coordinate and distance into global memory More...
__global__ void	**skeleton_data1_uint_exterial** (**uint** *d_skeleton, cudaTextureObject_t texObjskeleton, cudaExtent volumeSize, int originaldim, float exterial_threshold, float interial_threshold, float exterial_dis, float interial_dis) this method is to compute the gradient field and label the external skeleton data More...
__global__ void	**closetpoint** (float3 dif, float3 pos, float4 *d_volume, float4 *skeletonarray_ex, int No_skeleton_ex) this method is to compute the distance from each external skeleton piont to the ray More...
__global__ void	**pivotfinding** (float4 tempmin, float4 *d_volume, float4 *skeletonarray, int No_skeleton, float maxnegdis) this method is to compute the distance from each internal skeleton point to the selected external skeleton point More...
__global__ void	**shape_change** (cudaExtent volumeSize, float4 *skeletonarray, int No_skeleton, float xmin, float xmax, float ymin, float ymax, float zmin, float zmax, float newradii, int mode) this method is to transform the skeleton to new position and edit the radius of skeleton data More...
__global__ void	**reconstructed_to_d_volume** (float *d_volume2, cudaSurfaceObject_t SurfObj1, cudaExtent volumeSize, int originaldim) this method is to copy sdf from surface memory to global array memory More...
__global__ void	**copyskeleton** (bool *bo, float4 *skeletonarray, cudaSurfaceObject_t SurfObj1, cudaSurfaceObject_t SurfObj2, cudaExtent volumeSize, int originaldim, int No_skeleton) this method is to copy internal skeleton data information from skeletonarray to SurfObj1 and SurfObj2 More...
__global__ void	**levelset_skeleton** (bool *bo, bool *d_needfix, cudaSurfaceObject_t SurfObj1, cudaSurfaceObject_t SurfObj2, cudaExtent volumeSize, int originaldim) this method is to use upwind differencing for updating distance value on grid points More...

__global__ void	**vertical_vector** (float3 mid, float3 *d_volume, float4 *exter_in_cluster, int exterialincluster) this method is to compute the normal vector of the external skeleton sheet More...
__global__ void	**closest_plane** (float3 verticalvec, float4 tempmin, float4 *d_volume, float4 *skeletonarray, int No_skeleton, int **skeletonincluster**) this method is to obtain the internal skeleton points closest to a plane More...
__global__ void	**compute_usdf** (float *d_grid, float3 *d_pointcloud, int pointsize, cudaExtent volumeSize, int originaldim) this method is to compute the unsigned distance from each voxel to closest cloud point More...
__device__ float3	**normalvector** (cudaTextureObject_t texObj, float3 pos, cudaExtent volumeSize) this method is to compute normal vector of a location in the signed distance field More...
__global__ void	**surfIntegral_kernel** (float *d_volume, float *d_maxnegdis, cudaTextureObject_t texObj, cudaExtent volumeSize, int originaldim, float inf) this method is to compute the surface area of the signed distance model More...
__global__ void	**importprimitive** (int spefun, int roundn, float3 oricenter, float3 oricenter1, float r1, float r2, float h, int primitive, float *d_volume, cudaExtent volumeSize, int originaldim) this method is to import a primitive defined by the signed distance function More...

Function Documentation

```
__global__ void closest_plane ( float3   verticalvec,
                                float4   tempmin,
                                float4 * d_volume,
                                float4 * skeletonarray,
                                int      No_skeleton,
                                int      skeletonincluster
                              )
```

this method is to obtain the internal skeleton points closest to a plane

Parameters

verticalvec	is the normal vector of the external skeleton sheet
tempmin	is centroid of that seletcet external skeleton sheet
d_volume	is the array for storing the result closest internal skeleton
skeletonarray	is the array of internal skeleton
No_skeleton	is the amount of internal skeleton
skeletonincluster	is the amount of internal skeleton in that segmented sheet

```
__global__ void closetpoint ( float3   dif,
                              float3   pos,
                              float4 * d_volume,
                              float4 * skeletonarray_ex,
                              int      No_skeleton_ex
                            )
```

this method is to compute the distance from each external skeleton piont to the ray

Parameters

dif	is the ray direction vector
pos	is the ray starting point
d_volume	is the global memroy for storing all the distance
skeletonarray_ex	is the external skeleton array
No_skeleton_ex	is the amount of external skeleton data

```
__global__ void collision_render ( bool *                d_collision,
                                   cudaTextureObject_t   texObj,
                                   cudaTextureObject_t   texObj1,
                                   cudaExtent            volumeSize
                                 )
```

this method is to render a red background if collision detected

Parameters

d_collision	is the global output array for storing whether collision found
texObj	is the texture memory of a sdf model
texObj1	is the texture memory of a sdf model1
volumeSize	is the grid size

```
__global__ void compute_usdf ( float *        d_grid,
                               float3 *       d_pointcloud,
                               int            pointsize,
                               cudaExtent     volumeSize,
                               int            originaldim
                             )
```

this method is to compute the unsigned distance from each voxel to closest cloud point

Parameters

d_grid	is global array memory of all the grid pionts or voxels
d_pointcloud	is array of all point cloud data
pointsize	is the amount of cloud points
originaldim	is the original grid size if the grid spacing is 1
volumeSize	is the grid size

```
__global__ void copyfromtextodevice_uint ( uint *                d_skeleton,
                                           cudaTextureObject_t   texObjskeleton,
                                           cudaExtent            volumeSize,
                                           int                   originaldim
                                         )
```

this method is to convert positive distance to 1 and negatrive to 0

Parameters

texObjskeleton	is texture memory of skeleton
d_skeleton	is global memory for storing the result
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1

```
__global__ void copyskeleton ( bool *                bo,
                               float4 *              skeletonarray,
                               cudaSurfaceObject_t   SurfObj1,
                               cudaSurfaceObject_t   SurfObj2,
                               cudaExtent            volumeSize,
                               int                   originaldim,
                               int                   No_skeleton
                             )
```

this method is to copy internal skeleton data information from skeletonarray to SurfObj1 and SurfObj2

Parameters

bo	is the bool flag to label the internal skeleton
SurfObj1	is surface memory of distance field with copied internal skeleton information
SurfObj2	is surface memory of distance field with copied internal skeleton information
originaldim	is the original grid size if the grid spacing is 1
volumeSize	is the grid size
No_skeleton	is the amount of internal skeleton data

```
__global__ void copyskeletontotal ( uint *               d_array,
                                    uint *               scan_array,
                                    float4 *             skeletonarray_total,
                                    cudaTextureObject_t  texObjskeleton,
                                    cudaExtent           volumeSize
                                  )
```

this method is to compact skeleton coordinate and distance into global memory

Parameters

d_array	is global memory of label
scan_array	is the global memory of index mapping
volumeSize	is the grid size
skeletonarray_total	is the global memory for storing the compacted skeleton
texObjskeleton	texture memory of sdf model

```
__global__ void copysurface_kernel ( cudaSurfaceObject_t  SurfObj2,
                                     cudaSurfaceObject_t  SurfObj1,
                                     cudaExtent           volumeSize
                                   )
```

this method is to copy texture memory from SurfObj1 to SurfObj2

Parameters

SurfObj1	is the texture memory of a sdf model
SurfObj2	is the texture memory of a sdf model for alternative read and write
volumeSize	is the grid size

```
__global__ void d_render ( int                  sweepfun,
                           bool                 sweepconsrender,
                           bool                 sweeprender,
                           float                maxnegdis,
                           cudaTextureObject_t  texObj,
                           cudaTextureObject_t  texObj1,
                           cudaTextureObject_t  texObjplate,
                           uint *               d_output,
                           uint                 imageW,
                           uint                 imageH,
                           float                brightness,
                           float                threshold,
                           float                threshold1,
                           float                threshold2,
                           uint                 objn,
                           float                tolerance,
                           bool                 collision,
                           cudaExtent           volumeSize,
                           int                  originaldim,
                           int                  spefun
                         )
```

this method is for models before boolean operations

Parameters

sweepfun	indicates what kind of sweep function used: distance or z
sweepconsrender	indicates whether the sweep construction is on
sweeprender	indicates whether the sweep cut mode is on
maxnegdis	is the max negative distance value
texObj	is the texture memory of model 1
texObj1	is the texture memory of model 2
texObjplate	is the texture memory of plate for sculpturing
d_output	is the registered cuda array for mapping to cpu side later
imageW	is the width of the display window
imageH	is the height of the display window
brightness	is the bright adjust the display
threshold	is surface offset for model 1
threshold1	is the surface offset for model 2

threshold2	is the surface offset for both models
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1
spefun	indicates whether the user special material function is applied to primitives

```
__global__ void d_renderbool ( int                  sweepfun,
                               bool                 sweepconsrender,
                               cudaTextureObject_t  texObj,
                               uint *               d_output,
                               uint                 imageW,
                               uint                 imageH,
                               float                brightness,
                               float                threshold,
                               uint                 objn,
                               float                tolerance,
                               float                maxnegdis,
                               float                xcross,
                               float                ycross,
                               float                zcross,
                               cudaExtent           volumeSize,
                               int                  originaldim,
                               float                objcoorminx,
                               float                objcoormaxx,
                               float                objcoorminy,
                               float                objcoormaxy,
                               float                objcoorminz,
                               float                objcoormaxz,
                               uint                 material
                             )
```

this method is for models after boolean operations

Parameters

sweepfun	indicates what kind of sweep function used: distance or z
sweepconsrender	indicates whether the sweep construction is on
maxnegdis	is the max negative distance value
texObj	is the texture memory of model 1
d_output	is the registered cuda array for mapping to cpu side later
imageW	is the width of the display window
imageH	is the height of the display window
brightness	is the bright adjust the display
threshold	is surface offset for model 1
objn	is the object number user selected

tolerance	is the threshold value when determing hitting the surface
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1
material	indicates what kind of material function is used
xcross	is the value of x-axis plane for cut view
ycross	is the value of y-axis plane for cut view
zcross	is the value of z-axis plane for cut view
objcoorminx	is the min of x coordinate of the model
objcoormaxx	is the max of x coordinate of the model
objcoorminy	is the min of y coordinate of the model
objcoormaxy	is the max of y coordinate of the model
objcoorminz	is the min of z coordinate of the model
objcoormaxy	is the max of z coordinate of the model

```
__global__ void d_rendermoveskeleton ( int *        d_belongcluster,
                                       int          clusternum,
                                       bool         segmentation,
                                       bool         pivotget,
                                       float3       pivot3d,
                                       float4 *     skeletonarray,
                                       uint *       d_output,
                                       uint         imageW,
                                       uint         imageH,
                                       float        brightness,
                                       cudaExtent   volumeSize,
                                       int          originaldim,
                                       uint         skeleton,
                                       int          No_skeleton,
                                       float        xmin,
                                       float        xmax,
                                       float        ymin,
                                       float        ymax,
                                       float        zmin,
                                       float        zmax,
                                       bool         selection
                                     )
```

this method is to render when the skeleton are transforming

Parameters

d_belongcluster	is the skeleton belonging cluster number array
clusternum	number of clusters
segmentation	indicates whether the external skeleton is segmented
pivotget	indicates whether the pivot is obtained for rotating skeleton
pivot3d	is pivot positioin
skeletonarray	is the array for storing skeleton data
d_output	is the registered cuda array for mapping to cpu side later
imageW	is the width of the display window
imageH	is the height of the display window
brightness	is the bright adjust the display
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1

No_skeleton	is the amount of skeleton
skeleon	is the skeleon render model
xmin	is min of x coordinates
xmax	is max of x coordinates
ymin	is min of y coordinates
ymax	is max of y coordinates
zmin	is min of z coordinates
zmax	is max of z coordinates
selection	indicates whether there is a selected skeleon as pivot

```
__global__ void d_renderskeleton ( bool        color,
                                   int *       d_internalneighborcount,
                                   int *       d_belongcluster,
                                   int         clusternum,
                                   bool        segmentation,
                                   bool        pivotget,
                                   float3      pivot3d,
                                   float4 *    skeletonarray,
                                   uint *      d_output,
                                   uint        imageW,
                                   uint        imageH,
                                   float       brightness,
                                   cudaExtent  volumeSize,
                                   int         originaldim,
                                   uint        skeleton,
                                   int         No_skeleton,
                                   float       xmin,
                                   float       xmax,
                                   float       ymin,
                                   float       ymax,
                                   float       zmin,
                                   float       zmax,
                                   bool        selection
                                 )
```

this method is to render the skeleton

Parameters

d_internalneighborcount	indicats the amount of skeleton neighbor of a skeleton data
d_belongcluster	is the skeleton belonging cluster number array
clusternum	number of clusters
segmentation	indicates whether the external skeleton is segmented
pivotget	indicates whether the pivot is obtained for rotating skeleton
pivot3d	is pivot positioin
skeletonarray	is the array for storing skeleton data
d_output	is the registered cuda array for mapping to cpu side later
imageW	is the width of the display window
imageH	is the height of the display window

brightness	is the bright adjust the display
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1
No_skeleton	is the amount of skeleton
skeleon	is the skeleon render model
xmin	is min of x coordinates
xmax	is max of x coordinates
ymin	is min of y coordinates
ymax	is max of y coordinates
zmin	is min of z coordinates
zmax	is max of z coordinates
selection	indicates whether there is a selected skeleon as pivot

```
__global__ void dis2material_kernel ( float *                d_volume,
                                      cudaTextureObject_t    texObjsurface,
                                      int                    sweepfun,
                                      cudaExtent             volumeSize,
                                      int                    originaldim,
                                      float                  maxnegdis,
                                      float                  objcoorminx,
                                      float                  objcoormaxx,
                                      float                  objcoorminy,
                                      float                  objcoormaxy,
                                      float                  objcoorminz,
                                      float                  objcoormaxz
                                    )
```

this method is to convert distance value to material value

Parameters

d_volume	is global memory storing the result
texObjsurface	is the texture memory of the SDF models
sweepfun	is the material function for sweeping: distance or z-axis
volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1
maxnegdis	is the max negative distane of a object
objcoorminx	is the min x coordinate of the model
objcoormaxx	is the max x coordinate of the model
objcoorminy	is the min y coordinate of the model
objcoormaxy	is the max y coordinate of the model
objcoorminz	is the min z coordinate of the model
objcoormaxz	is the max z coordinate of the model

```
__global__ void distance_data_cpu ( float *                 d_skeleton,
                                    cudaTextureObject_t     texObjskeleton,
                                    cudaExtent              volumeSize,
                                    int                     originaldim
                                  )
```

this method is to copy skeleton data from texture memory to global memory

Parameters

d_skeleton	is global memory storing the result
texObjskeleton	is the texture memory of SDF model
volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1

```
__global__ void flipandcopy_kernel ( float *                 d_volume,
                                     cudaSurfaceObject_t     SurfObj1,
                                     float                   threshold,
                                     cudaExtent              volumeSize
                                   )
```

this method is to flip the dsitance from SurfObj2 and copy to d_volume

Parameters

SurfObj1	is surface memory of sdf model
d_volume	is global memory for storing the result
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1
threshold	is the surface offset of the object.

```
__global__ void flipandset_kernel ( cudaSurfaceObject_t  SurfObj1,
                                    cudaSurfaceObject_t  SurfObj2,
                                    cudaExtent           volumeSize,
                                    float                inf
                                  )
```

this method is to takes surface memory of SurfObj2 and flip the sign. Setting the positive valu as infinite

Parameters

SurfObj1	is the texture memory for storing the fliped data
SurfObj2	original data source
volumeSize	is the grid size
inf	is the user defined large value as infinite

```
__global__ void green_dif_blue ( float *              d_volume,
                                 cudaTextureObject_t  texObj,
                                 cudaTextureObject_t  texObj1,
                                 cudaExtent           volumeSize,
                                 float                threshold1,
                                 float                threshold2
                               )
```

this method is to apply difference operation on texObj with threshold1 and texObj1 with threshold2

Parameters

texObj	is the texture memory for sdf model
texObj	is the texture memory for sdf model 1
threshold1	is the surface offsetting of sdf model
threshold2	is the surface offsetting of sdf model 1
volumeSize	is the grid size
d_volume	is for storing the result

```
__global__ void importprimitive ( int          spefun,
                                  int          roundn,
                                  float3       oricenter,
                                  float3       oricenter1,
                                  float        r1,
                                  float        r2,
                                  float        h,
                                  int          primitive,
                                  float *      d_volume,
                                  cudaExtent   volumeSize,
                                  int          originaldim
                                )
```

this method is to import a primitive defined by the signed distance function

Parameters

spefun	indicates whether a user specified material function is applied
roundn	is the degree of distance: 3,4,5...
primitive	is the primitive users selected to create
oricenter	is the coordinate a characteristic center
oricenter	is the coordinate another characteristic center
r1	is the radius
r2	is the radius
h	is the height
d_volume	is for storing the result primitive's signed distance field
originaldim	is the original grid size if the grid spacing is 1
volumeSize	is the grid size

```
__global__ void initsurf ( cudaSurfaceObject_t  SurfObj1,
                           cudaSurfaceObject_t  SurfObj2,
                           cudaExtent           volumeSize,
                           int                  originaldim,
                           int                  No_skeleton
                         )
```

this method is to initialize the surface memory

Parameters

SurfObj1	is surface memory of sdf model
SurfObj2	is surface memory of sdf model for alternating
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1
No_skeleton	is the amount of internal skeleton

```
__global__ void levelset_kernel ( bool *               d_needfix,
                                  cudaSurfaceObject_t  SurfObj1,
                                  cudaSurfaceObject_t  SurfObj2,
                                  cudaExtent           volumeSize,
                                  int                  originaldim,
                                  float                inf,
                                  float                boundary
                                )
```

this method is to use upwind differencing fixing distance field

Parameters

d_needfix	indicates whether the updated value is witin error tolerance
SurfObj1	is the texture memory of a sdf model
SurfObj2	is the texture memory of a sdf model for alternative read and write
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1
inf	is the user defined large number treated as unupdated points
boundary	is the distance value, beyond which, the pionts's distance are unchangable

```
__global__ void levelset_skeleton ( bool *                bo,
                                    bool *                d_needfix,
                                    cudaSurfaceObject_t   SurfObj1,
                                    cudaSurfaceObject_t   SurfObj2,
                                    cudaExtent            volumeSize,
                                    int                   originaldim
                                  )
```

this method is to use upwind differencing for updating distance value on grid points

Parameters

bo	is the bool flag to label the internal skeleton
d_needfix	is label whether the error tolerance is within certain threshold for stopping
SurfObj1	is surface memory of distance field with copied internal skeleton information
SurfObj2	is surface memory of distance field with copied internal skeleton information, alternative
originaldim	is the original grid size if the grid spacing is 1
volumeSize	is the grid size

```
__global__ void material_function ( float *  d_newres,
                                    int      res,
                                    int      zres,
                                    int      material,
                                    float    maxnegdis,
                                    float    objcoorminx,
                                    float    objcoormaxx,
                                    float    objcoorminy,
                                    float    objcoormaxy,
                                    float    objcoorminz,
                                    float    objcoormaxz
                                  )
```

this method is to convert distance value to material value

Parameters

d_newres	is global memory storing the result
res	is the grid size in x and y axis
zres	is the grid size in z axis
material	is the material funcion mode
maxnegdis	is the max negative distane of a object
objcoorminx	is the min x coordinate of the model
objcoormaxx	is the max x coordinate of the model
objcoorminy	is the min y coordinate of the model
objcoormaxy	is the max y coordinate of the model
objcoorminz	is the min z coordinate of the model
objcoormaxz	is the max z coordinate of the model

```
__global__ void newthreshold ( float *                d_threshold,
                               cudaTextureObject_t    texObjsurface,
                               float                  threshold,
                               int                    res
                             )
```

this method is to adjust the surface offset of a model

Parameters

d_threshold	is the global output array for storing thresholded SDF
texObjsurface	is the texture memory of a sdf model
threshold	is the surface offset of the model
res	is the grid size

```
__device__ float3 normalvector ( cudaTextureObject_t  texObj,
                                 float3               pos,
                                 cudaExtent           volumeSize
                               )
```

this method is to compute normal vector of a location in the signed distance field

Parameters

texObj	is texture memory of sigend distance field
pos	is location in that distance field
volumeSize	is the grid size

```
__global__ void pivotfinding ( float4    tempmin,
                               float4 *  d_volume,
                               float4 *  skeletonarray,
                               int       No_skeleton,
                               float     maxnegdis
                             )
```

this method is to compute the distance from each internal skeleton point to the selected external skeleton point

Parameters

tempmin	is selected external skeleton point
d_volume	is the global memory for storing the distance
skeletonarray	is the global array of all internal skeleton points
No_skeleton	is the amount of internal skeleton data

```
__global__ void reconstructed_to_d_volume ( float *                d_volume2,
                                            cudaSurfaceObject_t    SurfObj1,
                                            cudaExtent             volumeSize,
                                            int                    originaldim
                                          )
```

this method is to copy sdf from surface memory to global array memory

Parameters

d_volume2	is the global memory for storing the distance field
SurfObj1	is surface memory of distance field
volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1

```
__global__ void repeat ( float *                 d_volume,
                         cudaTextureObject_t     texObj,
                         cudaTextureObject_t     texObj1,
                         cudaExtent              volumeSize,
                         float                   threshold1,
                         float                   threshold2
                       )
```

this method is to union texObj with threshold1 and texObj1 with threshold2

Parameters

texObj	is the texture memory for sdf model
texObj	is the texture memory for sdf model 1
threshold1	is the surface offsetting of sdf model
threshold2	is the surface offsetting of sdf model1
volumeSize	is the grid size
d_volume	is for storing the result

```
__global__ void resolution_kernel ( float *                 d_newres,
                                    cudaTextureObject_t     texObj3,
                                    int                     res,
                                    int                     zres
                                  )
```

this method is to adjust the resolution of SDF model

Parameters

d_newres	is the global output array for storing SDF model with new resolution
texObj3	is the texture memory of a sdf model
res	is the x and y resolution or grid size
zres	z resolution or grid size

```
__global__ void set_kernel ( cudaSurfaceObject_t  SurfObj1,
                             cudaSurfaceObject_t  SurfObj2,
                             cudaExtent           volumeSize,
                             float                inf
                           )
```

this method is to takes surface memory of SurfObj2 and Setting the positive value as infinite

Parameters

SurfObj1	is the texture memory for storing the fliped data
SurfObj2	original data source
volumeSize	is the grid size
inf	is the user defined large value as infinite

```
__global__ void shape_change ( cudaExtent  volumeSize,
                               float4 *    skeletonarray,
                               int         No_skeleton,
                               float       xmin,
                               float       xmax,
                               float       ymin,
                               float       ymax,
                               float       zmin,
                               float       zmax,
                               float       newradii,
                               int         mode
                             )
```

this method is to transform the skeleton to new position and edit the radius of skeleton data

Parameters

volumeSize	is grid size
skeletonarray	is the global memory of internal skeleton data
No_skeleton	is the amount of internal skeleton data
xmin	is the min value of x coordinate
xmax	is the max value of x coordinate
ymin	is the min value of y coordinate
ymax	is the max value of y coordinate
zmin	is the min value of z coordinate
zmax	is the max value of z coordinate
newradii	is the radius value
mode	is the redius modification mode: 0 indicates to a new raius, 1 indicates add newradii to old one

```
__global__ void skeleton_data1_uint_exterial ( uint *                d_skeleton,
                                               cudaTextureObject_t   texObjskeleton,
                                               cudaExtent            volumeSize,
                                               int                   originaldim,
                                               float                 exterial_threshold,
                                               float                 interial_threshold,
                                               float                 exterial_dis,
                                               float                 interial_dis
                                             )
```

this method is to compute the gradient field and label the external skeleton data

Parameters

d_skeleton	is global memory for storing the label
texObjskeleton	is the texture memory of SDF model
volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1
exterial_threshold	is the threshold of gradient for external skeleton
interial_threshold	is the threshold of gradient for internal skeleton
exterial_dis	is the threshold of distance value for external skeleton
interial_dis	is the threshold of distance value for external skeleton

```
__global__ void skeleton_data_share ( uint *       d_skeleton2,
                                      float *      skeleton2,
                                      cudaExtent   volumeSize,
                                      int          originaldim,
                                      float        exterial_threshold,
                                      float        interial_threshold,
                                      float        exterial_dis,
                                      float        interial_dis
                                    )
```

this method is to compute the gradient field and label skeleton data

Parameters

d_skeleton	is global memory for storing the label
skeleton2	is the global memory of SDF model
volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1
exterial_threshold	is the threshold of gradient for external skeleton
interial_threshold	is the threshold of gradient for internal skeleton
exterial_dis	is the threshold of distance value for external skeleton
interial_dis	is the threshold of distance value for external skeleton

```
__global__ void skeletonincluster ( int        cluster_No,
                                    float4 *   d_volume,
                                    int *      d_belongcluster,
                                    float4 *   skeletonarray_total,
                                    int        No_skeleton_total
                                  )
```

this method is to copy skeleton with certain ID to d_volume

Parameters

cluster_No	is the specified skeleton ID
d_belongcluster	is the array of ID matching with skeleton data
d_volume	is for storing the result
skeletonarray_total	is array for storing all the skeleton data
is	the amount of total skeleton data

```
__device__ float solvequadratic ( cudaSurfaceObject_t SurfObj,
                                  int                 x,
                                  int                 y,
                                  int                 z,
                                  cudaExtent          volumeSize,
                                  int                 originaldim
                                )
```

this method is to solve eikonal equation with upwind differencing

Parameters

SurfObj	is surface memory of sdf model
x	is the x coordinate of updating pionts
y	is the y coordinate of updating points
z	is the z coordinate of updating points
volumeSize	is the grid size
originaldim	is the original grid size if grid spacing is 1

```
__global__ void surfIntegral_kernel ( float *             d_volume,
                                      float *             d_maxnegdis,
                                      cudaTextureObject_t texObj,
                                      cudaExtent          volumeSize,
                                      int                 originaldim,
                                      float               inf
                                    )
```

this method is to compute the surface area of the signed distance model

Parameters

d_volume	is global memory for storing the surface area
d_maxnegdis	global memory for storing the maximum negative distance
texObj	is the texture memory of signed distance field
originaldim	is the original grid size if the grid spacing is 1
volumeSize	is the grid size
inf	is the infinite value

```
__global__ void sweepcons_bool ( int                 blend,
                                 float               k,
                                 float *             d_volume,
                                 cudaTextureObject_t texObj,
                                 cudaTextureObject_t texObj1,
                                 cudaExtent          volumeSize,
                                 float               threshold1,
                                 float               threshold2
                               )
```

this method is to apply union operation on texObj with threshold1 and texObj1 with threshold2

Parameters

blend	indicates whether the blending function is used
k	is what degree of blending function
texObj	is the texture memory for sdf model
texObj	is the texture memory for sdf model 1
threshold1	is the surface offsetting of sdf model
threshold2	is the surface offsetting of sdf model 1
volumeSize	is the grid size
d_volume	is for storing the result

```
__global__ void vertical_vector ( float3   mid,
                                  float3 * d_volume,
                                  float4 * exter_in_cluster,
                                  int      exterialincluster
                                )
```

this method is to compute the normal vector of the external skeleton sheet

Parameters

mid	is the centroid point of the external skelton sheet
d_volume	is for storing the computed normal vector
exter_in_cluster	is the array of all the external skeleton points in that segmented sheet
exterialincluster	is the amount of external skeleton in that segmented sheet