kernel.h File Reference

Go to the source code of this file.

Typedefs

typedef unsigned int **uint**

Functions

void **initCuda** (void *h_volume, void *h_volume1, cudaExtent volumeSize)
this method will be used to initilize the cuda More...

void **freeCudaBuffers** ()
this method will be used to free all the memory allocated in GPU More...

void **render_kernel** (bool sweep, dim3 gridSize, dim3 blockSize, **uint** *d_output, **uint** imageW, **uint** imageH, float brightness, float threshold, float threshold1, float threshold2, **uint** objn, float tolerance, bool collision, float xcross, float ycross, float zcross, cudaExtent volumeSize, int originaldim, **uint** skeleton)
this method will be used to trigger different render modes More...

void **copyInvViewMatrix** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy inverse matrix from cpu to GPU constant memory More...

void **copypivotrotateViewMatrix** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy inverse pivotmatrix from cpu to GPU constant memory More...

void **copyinvpivotrotateViewMatrix** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy inverse pivotmatrix from cpu to GPU constant memory More...

void **copyInvViewMatrix1** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy inverse matrix1 from cpu to GPU constant memory More...

void **copyInvViewMatrix2** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy inverse matrix2 from cpu to GPU constant memory More...

void **copyInvInvViewMatrix1** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy transform matrix1 from cpu to GPU constant memory More...

void **copyInvInvViewMatrix2** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy transform matrix2 from cpu to GPU constant memory More...

void **copyInvInvViewMatrixskeleton** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy transform skeleton matrix from cpu to GPU constant memory More...

void **copyInvViewMatrixskeleton** (float *invViewMatrix, size_t sizeofMatrix)
this method is for copy inverse skeleton matrix from cpu to GPU constant memory More...

void **booloperation** (**uint** booloperator, cudaExtent volumeSize, float threshold, float threshold1, float threshold2, int originaldim)
this method is the function called to apply boolean operations More...

void	**threshold_booled_object** (float threshold, float threshold1, float threshold2, cudaExtent volumeSize, int originaldim) this method is the function called to adjust the offset surface of booled object More...
void	**skeleton_oper** (**uint** skeleton, cudaExtent volumeSize, int originaldim) this method is to apply transform on skeleton data More...
void	**create_distance** (cudaExtent volumeSize, int originaldim, int No_skeleton) this method is recover the signed distance field based on the skeleton data More...
void	**create_newskeleton** (cudaExtent volumeSize, int originaldim, float newradii, int mode) this method is create the configure the skeleton to new position after transformation More...
void	**convert_dis2material** (cudaExtent volumeSize, int originaldim, int sweepfun) this method is for converting the distance field to material value or gray scale More...
void	**justfindmaxnegdis** (cudaExtent volumeSize, int originaldim) this method is from finding the max negative distance inside the object More...
void	**getpivot** (**uint** width, **uint** height) this method is for finding the pivot point user selected for skeleton rotation More...
void	**piv_direction** () this method is for computing the piv vector More...
void	**repeat_obj** (cudaExtent volumeSize, int originaldim) this method is for repeatly creating the same models along x and y axis More...
void	**addprimitive** (int primitive, cudaExtent volumeSize, int originaldim) this method is for add primitive geometry defined with signed distance function More...
int	**segment_exskeleton** (cudaExtent volumeSize, int originaldim) this method is for segment the external skeleton More...
void	**updateblue** (int primitive, cudaExtent volumeSize, int originaldim) this methold is for adding the created primitive to the system. More...
void	**importfile** (cudaExtent volumeSize, int originaldim) this methold is for import the tethedron edge and point file More...
void	**import_pointcloud** (cudaExtent volumeSize, int originaldim) this method is for importing the point cloud file More...
void	**colorinternal** (cudaExtent volumeSize, int originaldim) this method is for assign different color to the internal skeleton based on the amount of neighbor points More...
void	**justfindmaxcoor** (cudaExtent volumeSize, int originaldim) this method is for finding the min and max coordinates of the model More...
void	**importtrans** () this method is for importing the transformation file for sweeping the models More...
float3	**cross_product_host** (const float3 &temp1, const float3 &temp2) this method is for computing cross production More...

Typedef Documentation

typedef unsigned int uint

Function Documentation

void addprimitive (int primitive,
cudaExtent volumeSize,
int originaldim
)

this method is for add primitive geometry defined with signed distance function

Parameters

primitive is the different primitives user selected

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

void booloperation (uint booloperator,
cudaExtent volumeSize,
float threshold,
float threshold1,
float threshold2,
int originaldim
)

this method is the function called to apply boolean operations

Parameters

booloperator is the selected boolean operation: union, difference, intersection

volumeSize is the grid size

threshold is the threshold value for Model A

threshold1 is the threshold value for Model B

threshold2 is the threshold value for both Model A and B

originaldim is the original grid size if the grid spacing is 1

```
void colorinternal ( cudaExtent  volumeSize,
                     int         originaldim
                   )
```

this method is for assign different color to the internal skeleton based on the amount of neighbor points

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void convert_dis2material ( cudaExtent  volumeSize,
                            int         originaldim,
                            int         sweepfun
                          )
```

this method is for converting the distance field to material value or gray scale

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

sweepfun is to specify whether the sweep mode is on

```
void copyInvInvViewMatrix1 ( float *  invViewMatrix,
                             size_t  sizeofMatrix
                           )
```

this method is for copy transform matrix1 from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

void copyInvInvViewMatrix2 (float * invViewMatrix,
size_t sizeofMatrix
)

this method is for copy transform matrix2 from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

void copyInvInvViewMatrixskeleton (float * invViewMatrix,
size_t sizeofMatrix
)

this method is for copy transform skeleton matrix from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

void copyinvpivotrotateViewMatrix (float * invViewMatrix,
size_t sizeofMatrix
)

this method is for copy inverse pivotmatrix from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

```
void copyInvViewMatrix ( float *  invViewMatrix,
                         size_t  sizeofMatrix
                       )
```

this method is for copy inverse matrix from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

```
void copyInvViewMatrix1 ( float *  invViewMatrix,
                          size_t  sizeofMatrix
                        )
```

this method is for copy inverse matrix1 from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

```
void copyInvViewMatrix2 ( float *  invViewMatrix,
                          size_t  sizeofMatrix
                        )
```

this method is for copy inverse matrix2 from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

```
void copyInvViewMatrixskeleton ( float *  invViewMatrix,
                                 size_t  sizeofMatrix
                               )
```

this method is for copy inverse skeleton matrix from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

```
void copypivotrotateViewMatrix ( float *  invViewMatrix,
                                 size_t  sizeofMatrix
                               )
```

this method is for copy inverse pivotmatrix from cpu to GPU constant memory

Parameters

invViewMatrix is the float array of inverse matrix in cpu

sizeofMatrix is the size of matrix

```
void create_distance ( cudaExtent  volumeSize,
                       int         originaldim,
                       int         No_skeleton
                     )
```

this method is recover the signed distance field based on the skeleton data

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void create_newskeleton ( cudaExtent  volumeSize,
                          int         originaldim,
                          float       newradii,
                          int         mode
                        )
```

this method is create the configure the skeleton to new position after transformation

Parameters

volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1
newradii	is the radius for adjusting the skeleton's distance value
mode	is the radius changing mode: add some constant to the existing radius; use the new radius instead

```
float3 cross_product_host ( const float3 &  temp1,
                            const float3 &  temp2
                          )
```

this method is for computing cross production

Parameters

temp1	is the vector 1
temp2	is the vector 2

```
void freeCudaBuffers (  )
```

this method will be used to free all the memory allocated in GPU

Author

Di Zhang

```
void getpivot ( uint  width,
                uint  height
              )
```

this method is for finding the pivot point user selected for skeleton rotation

Parameters

width is the x coordinate of the point user clicking on the dispaly window

height is the y coordinate of the point user clicking on the dispaly window

```
void import_pointcloud ( cudaExtent  volumeSize,
                         int         originaldim
                       )
```

this method is for importing the point cloud file

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void importfile ( cudaExtent  volumeSize,
                  int         originaldim
                )
```

this methold is for import the tethedron edge and point file

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void importtrans (  )
```

this method is for importing the transformation file for sweeping the models

```
void initCuda ( void *        h_volume,
                void *        h_volume1,
                cudaExtent    volumeSize
              )
```

this method will be used to initilize the cuda

Author

Di Zhang

Parameters

h_volume is the cpu side array pointer of SDF-rep model A

h_volume1 is the cpu side array pointer of SDF-rep model B

volumeSize is the grid size class

```
void justfindmaxcoor ( cudaExtent  volumeSize,
                       int         originaldim
                     )
```

this method is for finding the min and max coordinates of the model

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void justfindmaxnegdis ( cudaExtent  volumeSize,
                         int         originaldim
                       )
```

this method is from finding the max negative distance inside the object

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void piv_direction ( )
```

this method is for computing the piv vector

```
void render_kernel ( bool        sweep,
                     dim3        gridSize,
                     dim3        blockSize,
                     uint *      d_output,
                     uint        imageW,
                     uint        imageH,
                     float       brightness,
                     float       threshold,
                     float       threshold1,
                     float       threshold2,
                     uint        objn,
                     float       tolerance,
                     bool        collision,
                     float       xcross,
                     float       ycross,
                     float       zcross,
                     cudaExtent  volumeSize,
                     int         originaldim,
                     uint        skeleton
                   )
```

this method will be used to trigger different render modes

Author

Di Zhang

Parameters

sweep	is to specify whether the sweep mode is on
gridSize	is the grid size class for configure the kernel function
blockSize	is the block size class for configure the kernel function
d_output	is the array registered in CUDA, which will be mapped to CPU array for rendering
imageW	is the width of the display window
imageH	is the height of the display window
brightness	is the bright of the rendered imageH
threshold	is the offset value for Model A
threshold1	is the offset value for Model B
threshold2	is the offset value for both Model A and B
objn	is the object No. selected

tolerance is the threshold value for surface finding

collision is to specify whether there is collision detected among objects

xcross is the value of x-axis plane for cutaway view

ycross is the value of y-axis plane for cutaway view

zcross is the value of z-axis plane for cutaway view

volumeSize is the grid size

originaldim is the origianl grid size if grid spacing is 1

skeleton is the skeleton render mode selected

```
void repeat_obj ( cudaExtent  volumeSize,
                  int         originaldim
                )
```

this method is for repeatly creating the same models along x and y axis

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
int segment_exskeleton ( cudaExtent  volumeSize,
                         int         originaldim
                       )
```

this method is for segment the external skeleton

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

```
void skeleton_oper ( uint       skeleton,
                     cudaExtent volumeSize,
                     int        originaldim
                   )
```

this method is to apply transform on skeleton data

Parameters

skeleton	is the skeleton render mode: skeleton only, model only, both skeleton and models
volumeSize	is the grid size
originaldim	is the original grid size if the grid spacing is 1

```
void threshold_booled_object ( float       threshold,
                               float       threshold1,
                               float       threshold2,
                               cudaExtent  volumeSize,
                               int         originaldim
                             )
```

this method is the function called to adjust the offset surface of booled object

Parameters

volumeSize	is the grid size
threshold	is the threshold value for Model A
threshold1	is the threshold value for Model B
threshold2	is the threshold value for both Model A and B
originaldim	is the original grid size if the grid spacing is 1

```
void updateblue ( int        primitive,
                  cudaExtent volumeSize,
                  int        originaldim
                )
```

this methold is for adding the created primitive to the system.

Parameters

volumeSize is the grid size

originaldim is the original grid size if the grid spacing is 1

primitive is the different primitives user selected