Illumination Networks Demo Reference Manual

Generated by Doxygen 1.4.6-NO

Thu Apr 27 15:37:35 2006

Chapter 1

Illumination Networks Demo Class Documentation

1.1 PreIllumSystem Class Reference

This class represents a particle system that uses the Illumination Networks technique.

Public Member Functions

- void **Init** (int particlecount, int directioncount)

  Initializator function.

- void **Display** ()

  Render the particle system.

- void **Refresh** (Vector lightpos, Vector lightpos2, Vector lightcolor, Vector lightcolor2)

  Refreshes the system in a frame.

- char ∗ **DisplayTexture** (int tex)

  Displays one of the textures used by the system.

Public Attributes

- ParticleSystem **m_System**

  a system that stores particle positions and can render them as sprites

- int **m_DirectionCount**

  the number of directions the technique should use

- int **m_ParticleCount**

  the number of particles in the system

- float **m_Albedo**

the albedo of one particle

- float **m_Opacity**

  the desirer opacity of the medium

- float **m_Symmetry**

  the symmetry of scattering used in the phase function

- int **m_LightWindowSize**

  resolution of the lightsources viewports

- int **m_IterateCount**

  number of iterations in a frame

Private Member Functions

- void **CreateGivenDirections** ()

  generates directions equally along the unit sphere

- void **CreateRandomDirections** (bool fillarray)

  generates random directions

- float **Phase** (Vector diri, Vector dirj, float symmetry)

  calculates the scattering phase function value for two directions and a symmetry value

- **GetNearestDirection** (Vector LightPosition)

  searches the stored directions and returns the one closest to a given direction

- void **InitSystem** (int particlecount, int directioncount)

  Initializator function.

- void **CreateVisibilityTexture** ()

  Creates a texture that stores the visibility information of the particles.

- void **CreateNearestDirectionTexture** ()

  Creates a texture that stores.

- void **CreatePhaseTexture** ()

  Creates a look-up texture to speed up phase function calculation.

- void **CreateLVisMap** ()

  Creates a texture that can be used to determine which particles are visible from the lightsource.

- void **CreateTauTexture** ()

  Creates a texture tha stores the tau value for each particle.

- void **RefreshDirectIllumTexture** ()

  Refreshes the texture that stores direct illumination information.

- void **Iterate** ()

  Updates the illumination texture.

- void **CreateEyeRadTexture** ()

  Updates the eye radiance texture.

- void **FindVisiblesWithRendering** (Vector LightPosition, int row)

  Finds the visible particles from a point of view.

- void **RenderToImpostor** ()

  not used

Private Attributes

- Vector **m_SkyColor**

  color of the sky

- Impostor **m_ScreenQuad**

  used for fullscreen quad rendering

- Camera ∗ **m_EyeCamera**

  view camera

- Vector **m_LightPosition**

  position of the first lightsource

- Vector **m_LightColor**

  color of the first lightsource

- Vector **m_LightPosition2**

  position of the second lightsource

- Vector **m_LightColor2**

  color of the second lightsource

- int **m_NearestDir**

  the closest direction from the predefined directions to the light's direction

- int **m_NearestDir2**

  the second closest direction from the predefined directions to the light's direction

- float **m_Weight1**

  weight of m_NearestDir

- float **m_Weight2**

  weight of m_NearestDir2

- GLuint **m_VisibilityTexID**

stores the visibility information of the particles

- GLuint **m_DirectionsTexID**

  stores predefined directions to use

- GLuint **m_PhaseTextureID**

  a look-up texture to speed up phase function calculation

- GLuint **m_LVisMapID**

  a texture that can be used to determine which particles are visible from the lightsource

- GLuint **m_RenderedVisID**

  used when determining licible particles

- GLuint **m_TauTextureID**

  stores tau value for each particle

1.1.1 Detailed Description

This class represents a particle system that uses the Illumination Networks technique.

This particle system can be lit with two dinamic directional light sources and a sky light color. The direction and color of the light sources can freely change.

1.1.2 Member Function Documentation

1.1.2.1 void PreIllumSystem::CreateEyeRadTexture () `[private]`

Updates the eye radiance texture.

The eye radiance texture stores the amount of light headig from each particle to the eye.

1.1.2.2 void PreIllumSystem::CreateTauTexture () `[private]`

Creates a texture tha stores the tau value for each particle.

The tau values are calculated from the given desired opacity and the size of the particle.

1.1.2.3 void PreIllumSystem::CreateVisibilityTexture () `[private]`

Creates a texture that stores the visibility information of the particles.

For each particle for each direction the first visible (from that direction) particle's id is stored.

1.1.2.4 char $*$ PreIllumSystem::DisplayTexture (int *tex*)

Displays one of the textures used by the system.

Used for debugging and presentation.

1.1.2.5 void PreIllumSystem::FindVisiblesWithRendering (Vector *LightPosition*, int *row*) `[private]`

Finds the visible particles from a point of view.

The light visibility texture stores the id of the visible particles (with occlusion) from the light sources. The param "row" means the id of the lightsource (0 or 1).

The visibility is calculated with rendering the particles from the lightsource. Each particle has a color corresponding it's id. The resulting image is read back, and the pixels are counted. If the number of pixels with a particle's id found is greather than some limit, the particle is visible.

1.1.2.6 void PreIllumSystem::Refresh (Vector *lightpos*, Vector *lightpos2*, Vector *lightcolor*, Vector *lightcolor2*)

Refreshes the system in a frame.

The actual light positions and colors should be passed.

1.1.2.7 void PreIllumSystem::RefreshDirectIllumTexture () `[private]`

Refreshes the texture that stores direct illumination information.

Direct illumination is the amount of light coming directly from the lightsource.

1.1.3 Member Data Documentation

1.1.3.1 int PreIllumSystem::m_IterateCount

number of iterations in a frame

As the result of the last frame is used, this should be set to one.

1.1.3.2 GLuint PreIllumSystem::m_RenderedVisID `[private]`

used when determining licible particles

See also:
FindVisiblesWithRendering(p. 5)

1.1.3.3 GLuint PreIllumSystem::m_VisibilityTexID `[private]`

stores the visibility information of the particles

For each particle for each direction the first visible (from that direction) particle's id is stored.

1.1.3.4 float PreIllumSystem::m_Weight1 `[private]`

weight of m_NearestDir

m_NearestDir and m_NearestDir2 will be interpolated

1.1.3.5 float PreIllumSystem::m_Weight2 `[private]`

weight of m_NearestDir2

m_NearestDir and m_NearestDir2 will be interpolated