Caustic Maps

with OpenGL

Chakrit Watcharopas

watcharo@cs.clemson.edu

Clemson University

Outline

- Motivation
- Caustic Texture Generation
- Caustic Mapping
- Multitexturing
- Demo

Motivation

- Generating "Real" Caustics in computer graphics is quite costly
- Using techniques such as bidirectional path tracing or the photon map

Motivation

- Generating "Real" Caustics in computer graphics is quite costly
- Using techniques such as bidirectional path tracing or the photon map
- However, some applications are interested in capturing the general "feel" and "look" of a caustic environment, i.e. Hunley Project

Motivation

- Generating "Real" Caustics in computer graphics is quite costly
- Using techniques such as bidirectional path tracing or the photon map
- However, some applications are interested in capturing the general "feel" and "look" of a caustic environment, i.e. Hunley Project

How do we do it?

We fake it!

- We precompute a caustic texture map which is both periodic in space and time

We fake it!

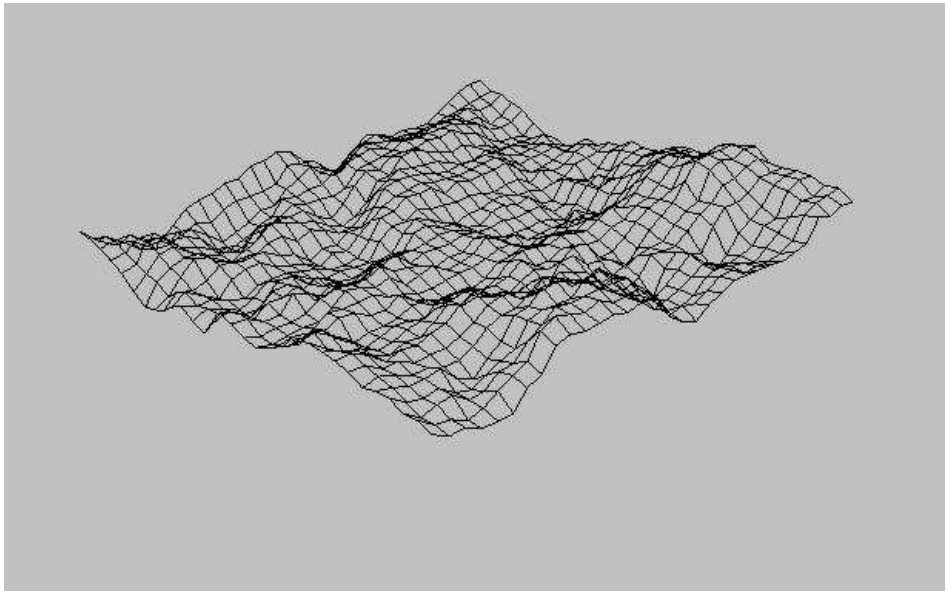
- We precompute a caustic texture map which is both periodic in space and time
- We render the scene using texture mapping in OpenGL
  - 2 Pass Rendering
  - Multitexturing

Caustic Texture Generation

- Caustic Map
  - Developed by Jos Stam
  - Consists of a set of picture files, i.e. 32 textures
  - Each picture is spatially periodic
  - The sequence of 32 tiles also wraps around in time

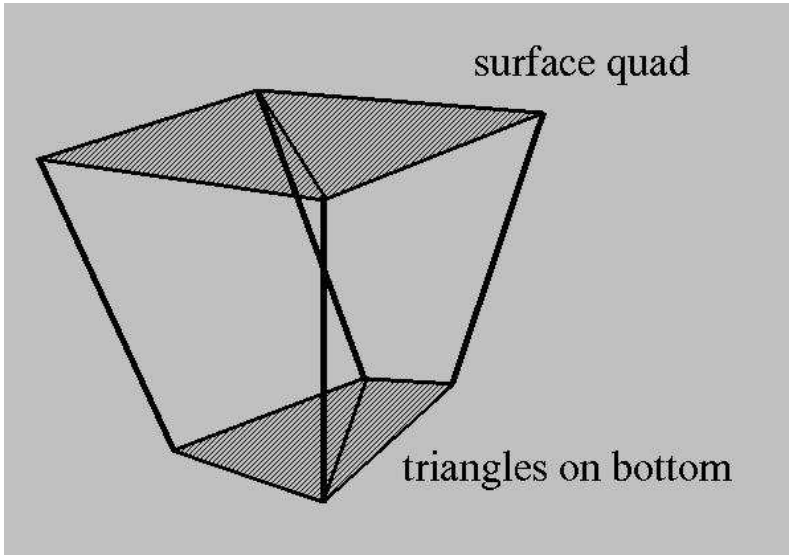
Caustic Texture Generation (1)

- Use the Fast Fourier transform (FFT)
- The usefullness of the FFT comes from the fact that most phenomena in nature tend to be approximately cyclical



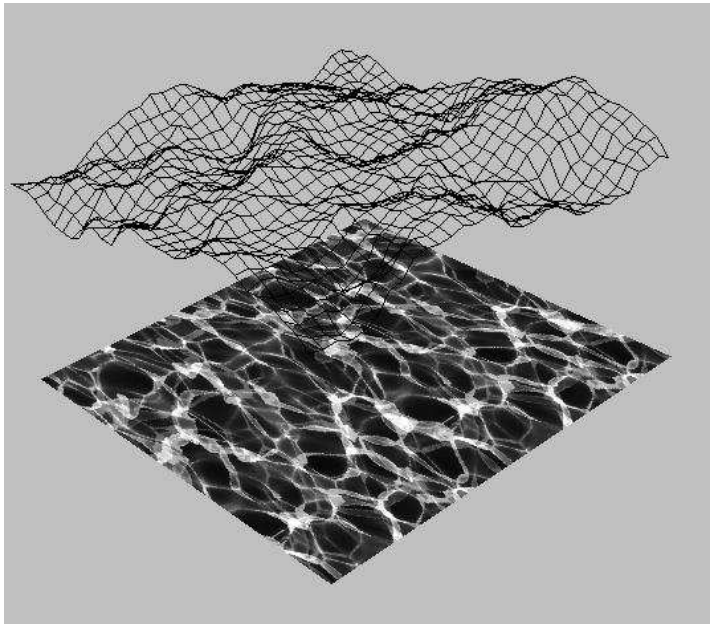
Caustic Texture Generation (2)

- Assume that the light source points straight at this surface
- For each vertex of every quad on the surface, compute the intersection of the refracted ray with a plane at some depth D



Caustic Texture Generation (3)

- Shade and blend with OpenGL
- The color intensity is proportional to the area of the triangle



Caustic Mapping

- Preload caustic textures
  - `glTexEnvf (..,..,GL_MODULATE);`
  - `glBindTexture (GL_TEXTURE_2D,caustic_id_#nn);`
  - `gluBuild2DMipmaps (..,GL_LUMINANCE,caustic_buf_#nn);`
- 2 Pass Rendering with blending

Caustic Mapping: 2 Pass Rendering (1)

- First Pass: Render as normal
- Second Pass: Render with caustics
  - `glColor3f (1.0,1.0,1.0);`
  - `glDisable (GL_LIGHTING);`
  - `GLfloat sPlane[4]={1.0,1.0,0,0};`
  - `GLfloat tPlane[4]={0,1.0,1.0,0};`
  - `glTexGenfv (GL_S,GL_OBJECT_PLANE,sPlane);`
  - `glTexGenfv (GL_T,GL_OBJECT_PLANE,tPlane);`
  - :

Caustic Mapping: 2 Pass Rendering (2)

- Second Pass: Render with caustics (Cont.)
  - `glBindTexture (GL_TEXTURE_2D,caustic_id);`
  - `glutIdleFunc ();`
    looping caustic_id from 1 to 32
  - `glCallList (Draw_Object);`

Caustic Mapping: Multitexturing (1)

- Possibly gives twice higher frame rate
- `glActiveTextureARB (GL_TEXTURE1_ARB);`
- `glColor3f (1.0,1.0,1.0);`
- `glDisable (GL_LIGHTING);`
- `glEnable (GL_TEXTURE_2D);`
- `glBindTexture (GL_TEXTURE_2D,caustic_id);`
- :

Caustic Mapping: Multitexturing (2)

- `glActiveTextureARB (GL_TEXTURE0_ARB);`
- `glEnable (GL_LIGHTING);`
- `glDisable (GL_TEXTURE_2D);`
- `glCallList (Draw_Object);`

Tada