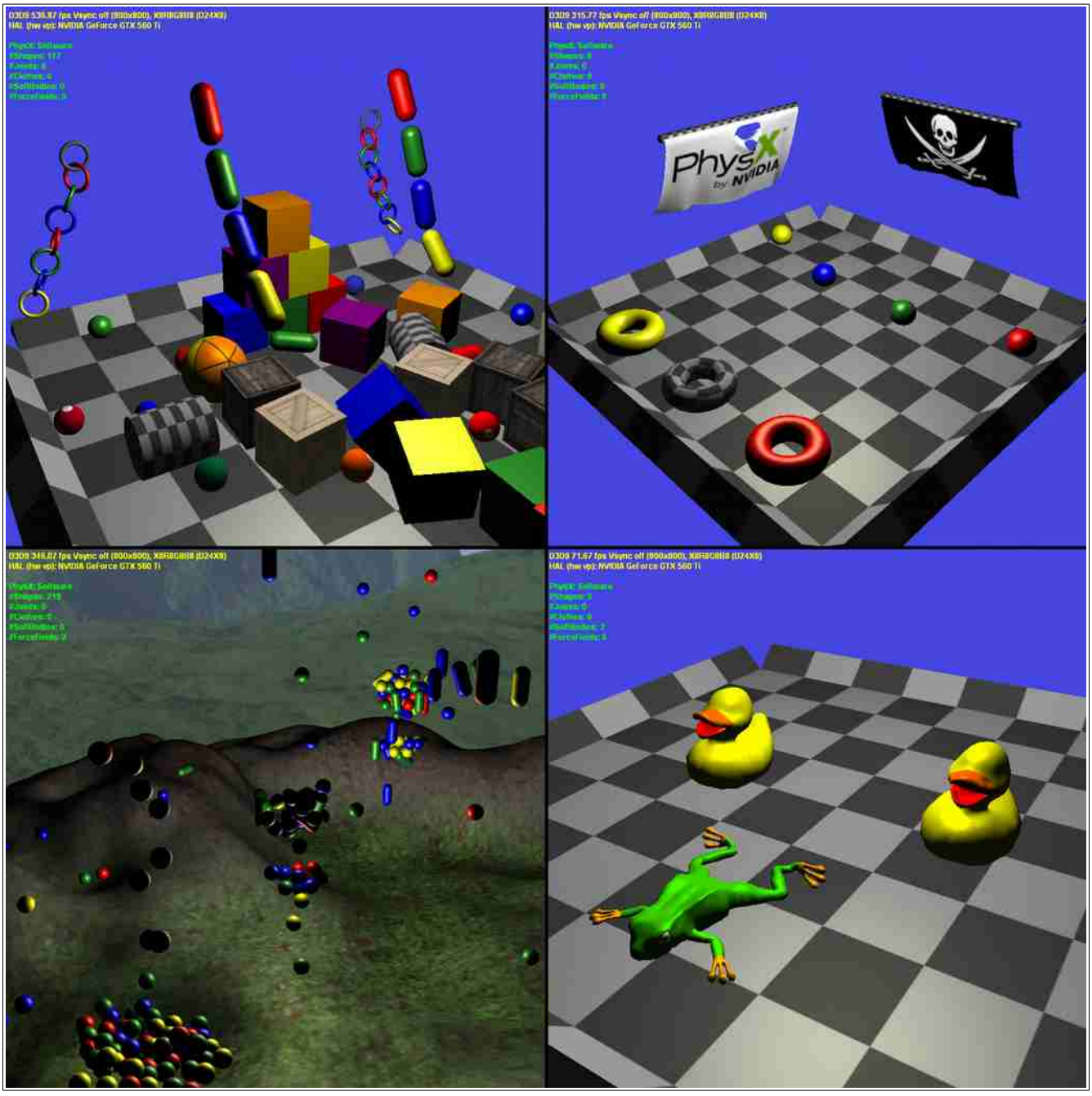
NVIDIA PhysX 2.x Engine



Description: game like physics sandbox application, with data driven "levels" featuring the core components of the PhysX 2.X SDK

Features:

1) simple OO entity hierarchy
2) xml based scene description (TinyXML loader) – meshes, textures, materials, actors, PhysX descriptors, scenes
3) PhysX features – rigidbody, cloth + tearing, pressure cloth, softbody, joints, terrain, forcefields

2D Fluid Simulation

Description: 2D “stable fluid” simulation

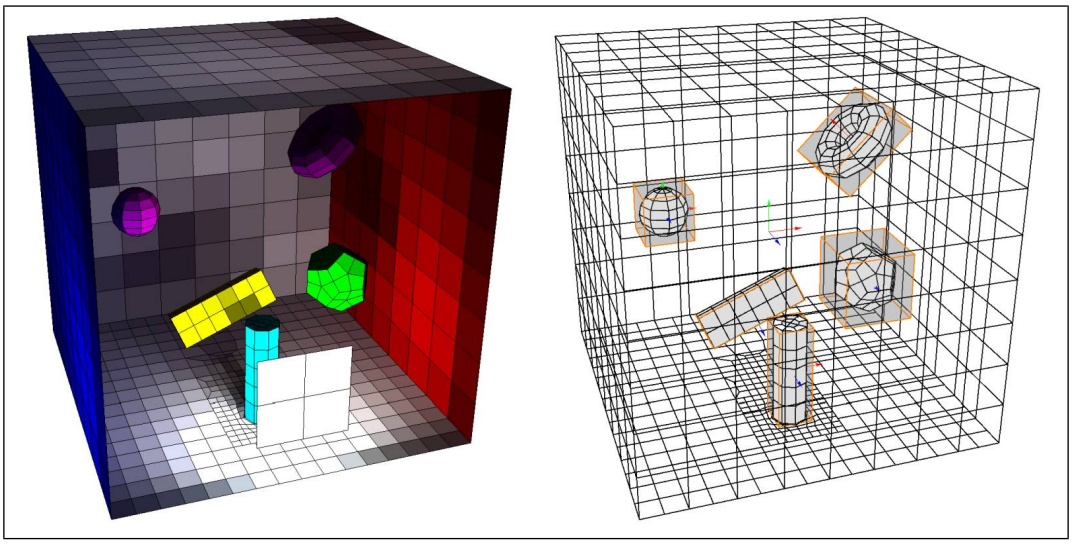
Features:

1) implementation of the renowned algorithm of Jos Stam
2) obstacles
3) vorticity confinement
4) monotonic cubic interpolation, MacCormack advection

Reference:

Jos Stam – Stable Fluids
GPU Gems – Fast Fluid Dynamics Simulation on the GPU
GPU Gems 3 – Real-Time Simulation and Rendering of 3D Fluids

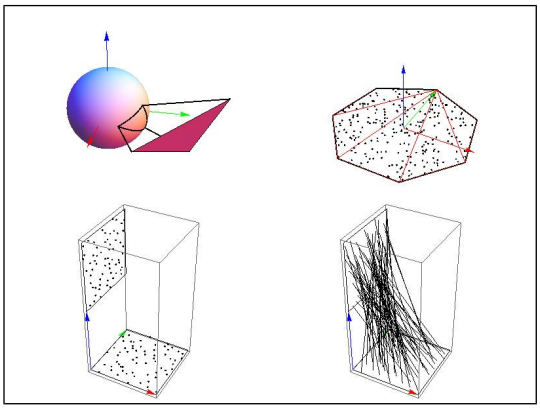
Radiosity



Description: image synthesis based on light reflections off diffuse surfaces

Steps:

1) divide surfaces into patches
2) calculate form factors between patches
   - tested multiple sample generation algorithms (jitter, multi-jitter, Poisson) for visibility calculations
   - compared multiple form factor calculations (closed form, differential area to differential area, diLaura, hemicube, MonteCarlo)
   - accelerated patch visibility calculation with OBB–OBB intersection test (OBB occluder versus OBB formed by two facing patches)

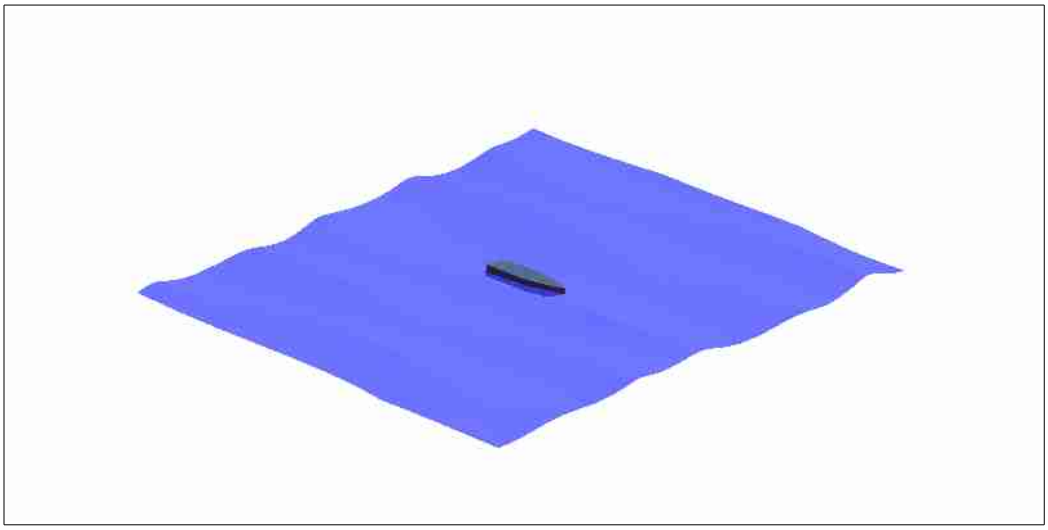


3) solve linear system arising from the radiosity equation

Reference:

Peter Schröder, Pat Hanrahan – On the Form Factor between Two Polygons
D.L. DiLaura – Nondiffuse Radiative Tranfer 2 – Planar Area Sources and Receivers
An-SeopChoi – Practical applications of form factor computation in lighting calculations

Buoyancy Force Calculation

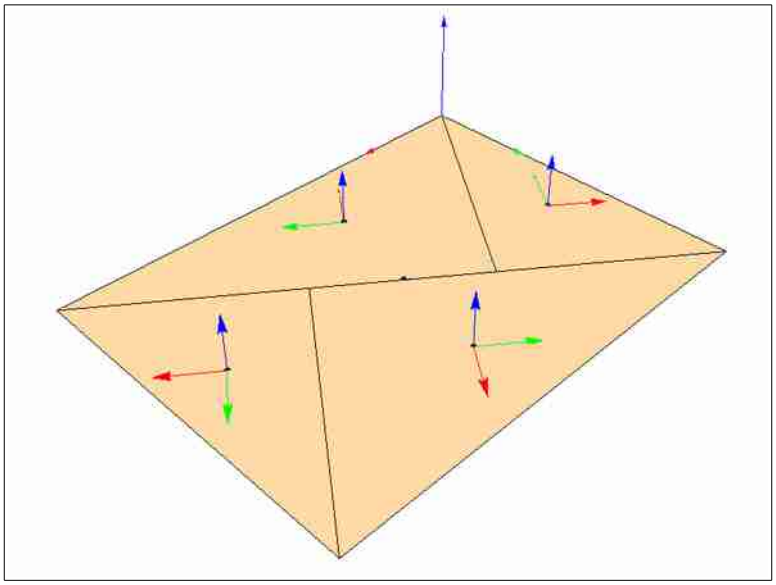


Description: simple boat simulation

Algorithm Overview: calculating hydrostatic forces on immersed bodies represented by a triangular mesh

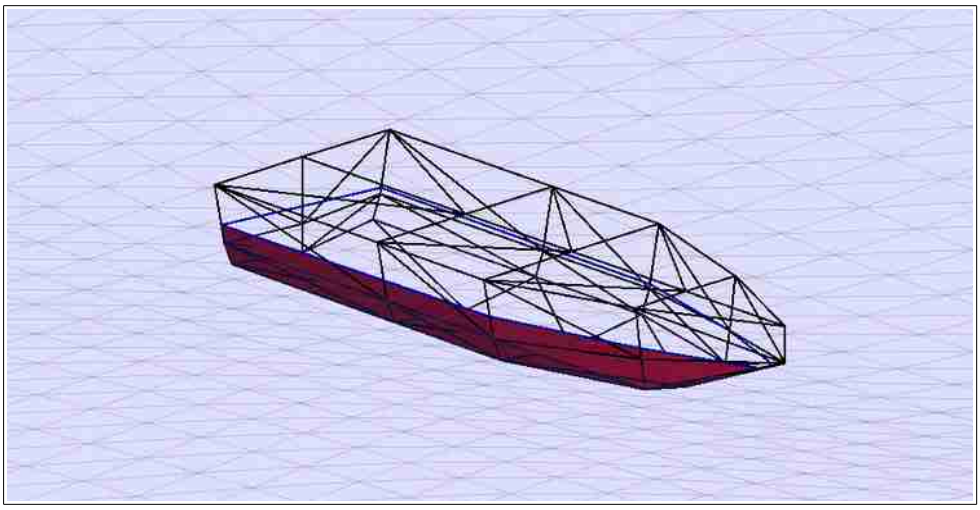
Steps:

1) moment of inertia (rotational inertia) calculation for triangle mesh (hull) – triangles are approximated as plates with infinitely small thickness
   only calculated once, can be calculated before the whole simulation
   - divide every triangle into 2 right-angled triangle
   - with the help of parallel axis theorem (inertia tensor) sum the contribution of the right-angled triangles to the moment of inertia at the center of mass

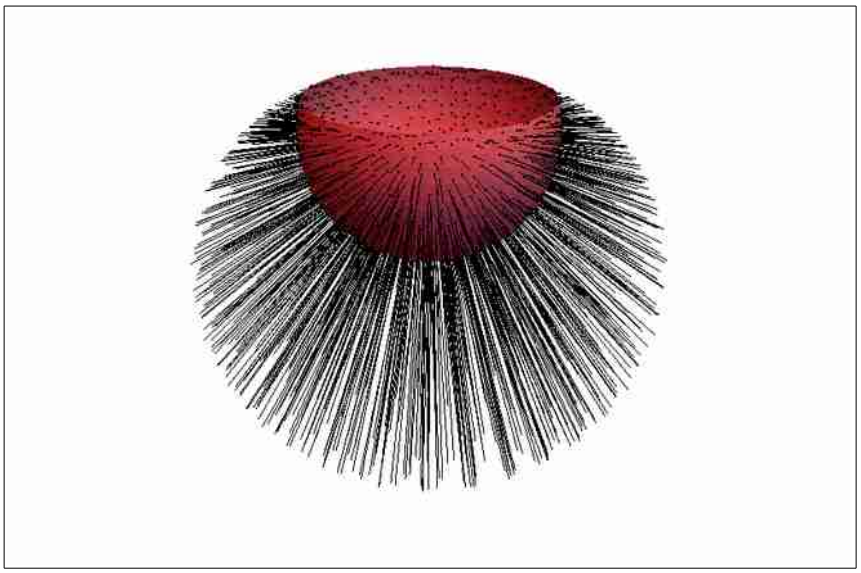


moment of inertia validation – rectangle assembled from 4 right-angled triangles

2) separate and tessellate boat hull according to water height for buoyancy force calculation – divide triangles that are only partially submerged into fully submerged and non-submerged triangles



3) calculate the point of application and magnitude of the hydrostatic force for every triangle resulted in step 2 and accumulate it at the center of mass of the boat

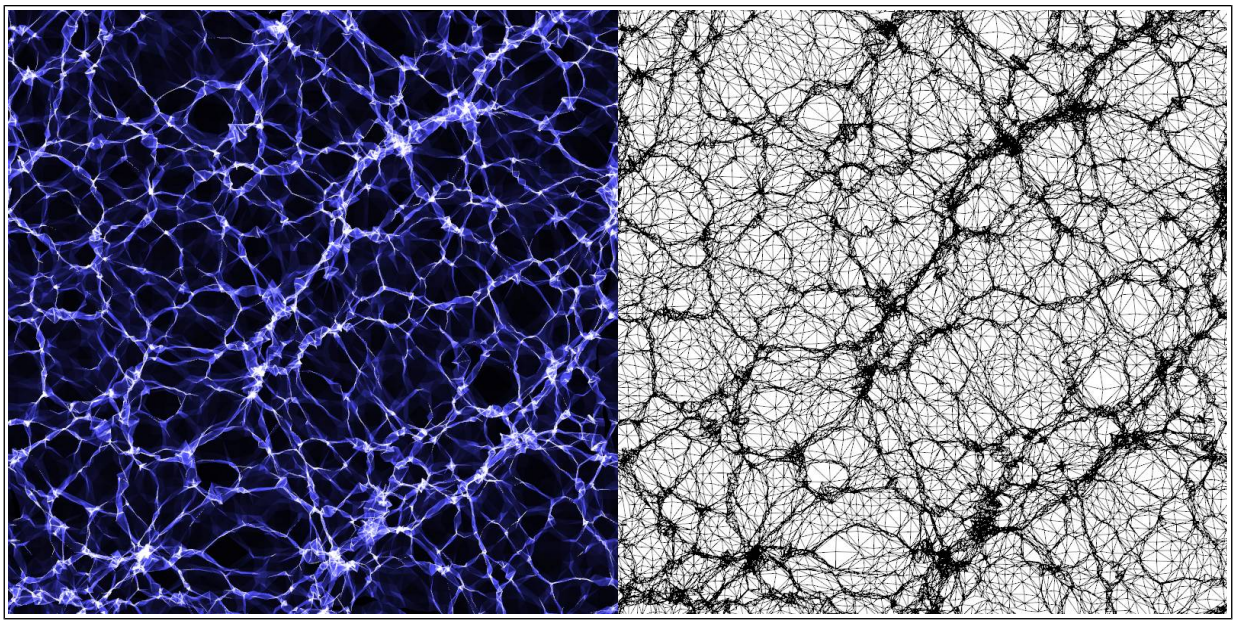


pressure distribution on a partially submerged sphere

4) use the calculated force to update the position of the boat – render image
5) update water height, go back to step 2

References:
Water interaction model for boats in video games by Jacques Kerner

Periodic Caustic Texture

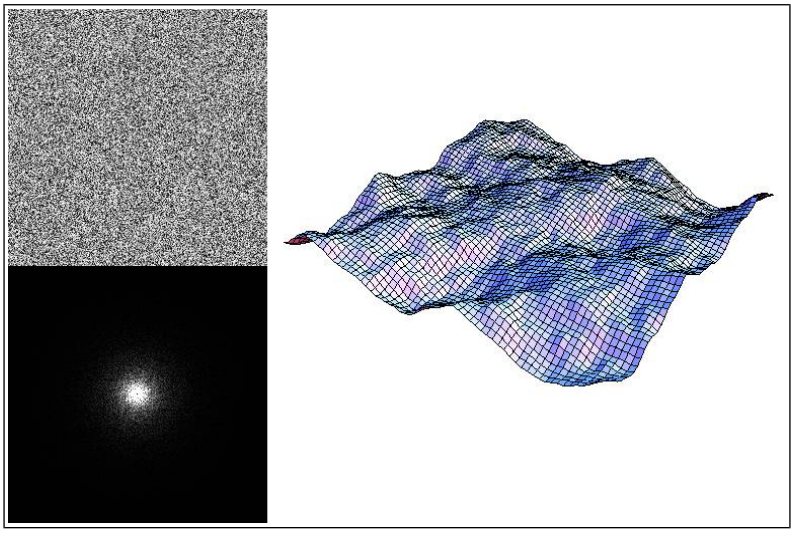


Description: generating periodic caustics pattern in space and time

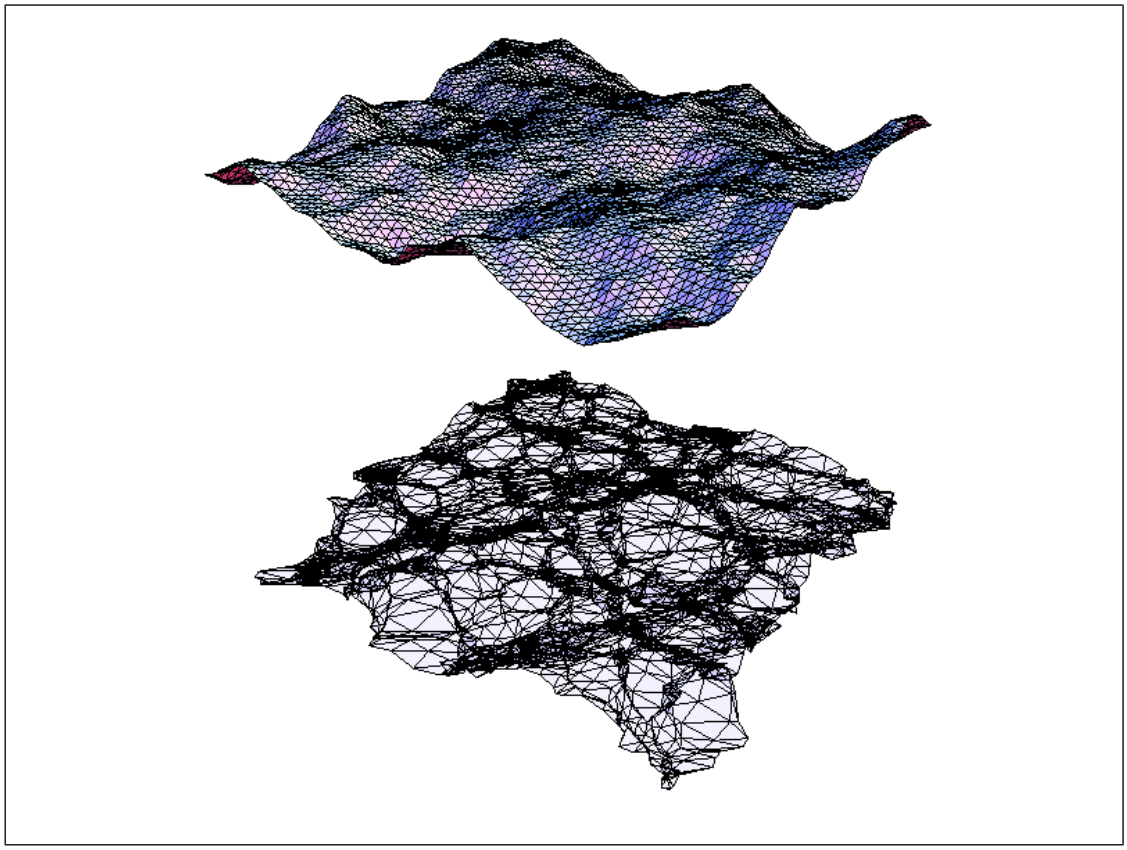
Algorithm Overview: random (fractal) surface profile by filtering a three-dimensional white noise by a power law-like filter in space and a Gaussian-like filter in time

Steps:

1) random height field generation based on frequency synthesis
   - create a random white noise signal
   - apply the fast Fourier transform to transfer the data into the frequency domain and apply the 1/f filter
   - transform it back to the spatial domain with an inverse fast Fourier transform



2) smoothing in time with Gaussian filter
3) projection of the resulting surface (triangles) to the ground plane – assuming directional light source pointing at this surface, computation of the intersection of the refracted rays with a plane at some depth – visualization with fixed-function OpenGL additive blending, triangle intensity is proportional to the area of the triangle

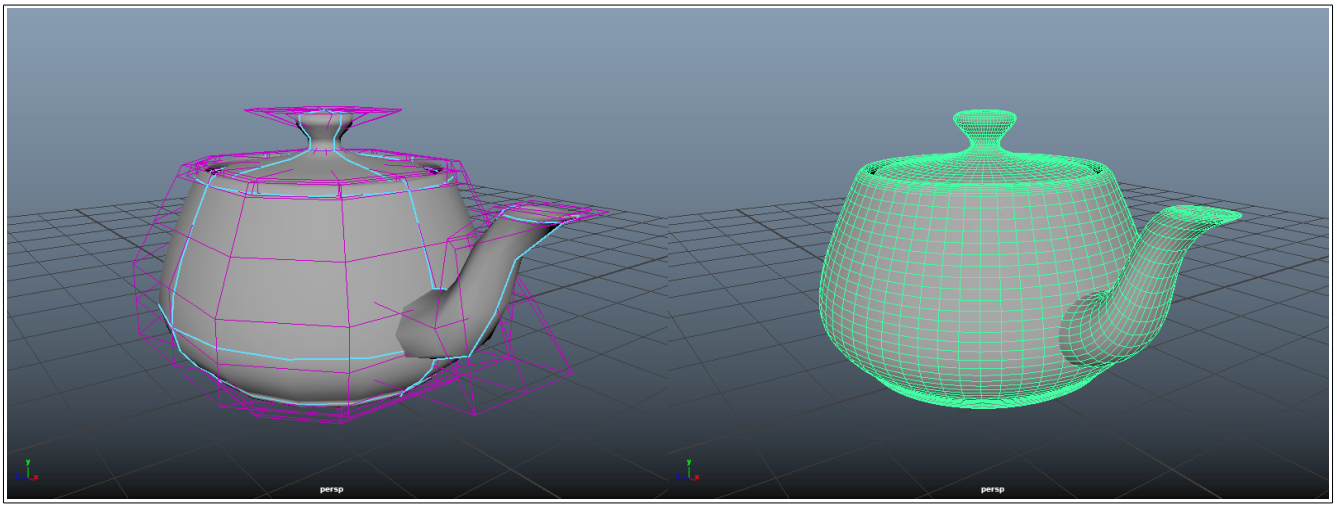


References:

Periodic Caustic Textures by Jos Stam

Frequency Synthesis of Landscapes (and clouds)

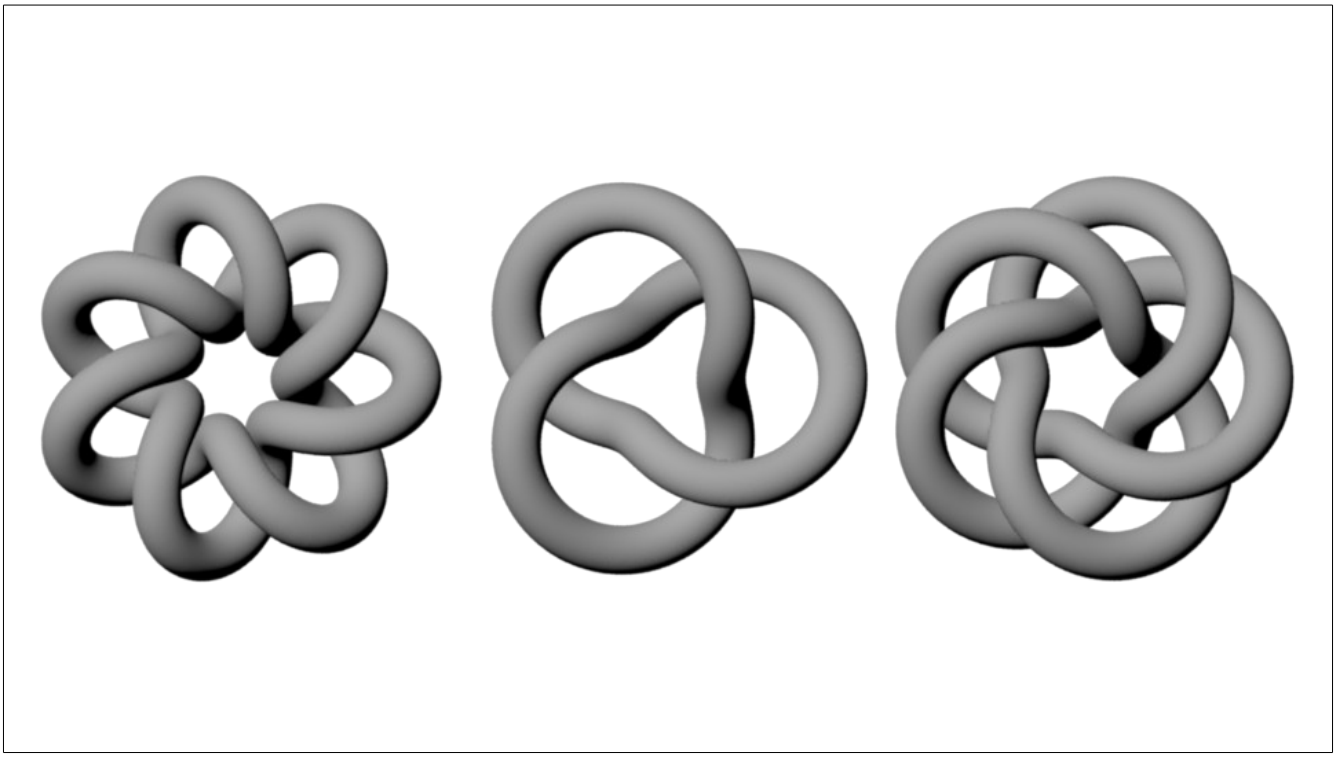
Autodesk Maya – C++ API



Description: Bezier patch visualization with teapot primitive

Features:

1) raw Bezier data to nurbs surface
2) raw Bezier data to polygon mesh

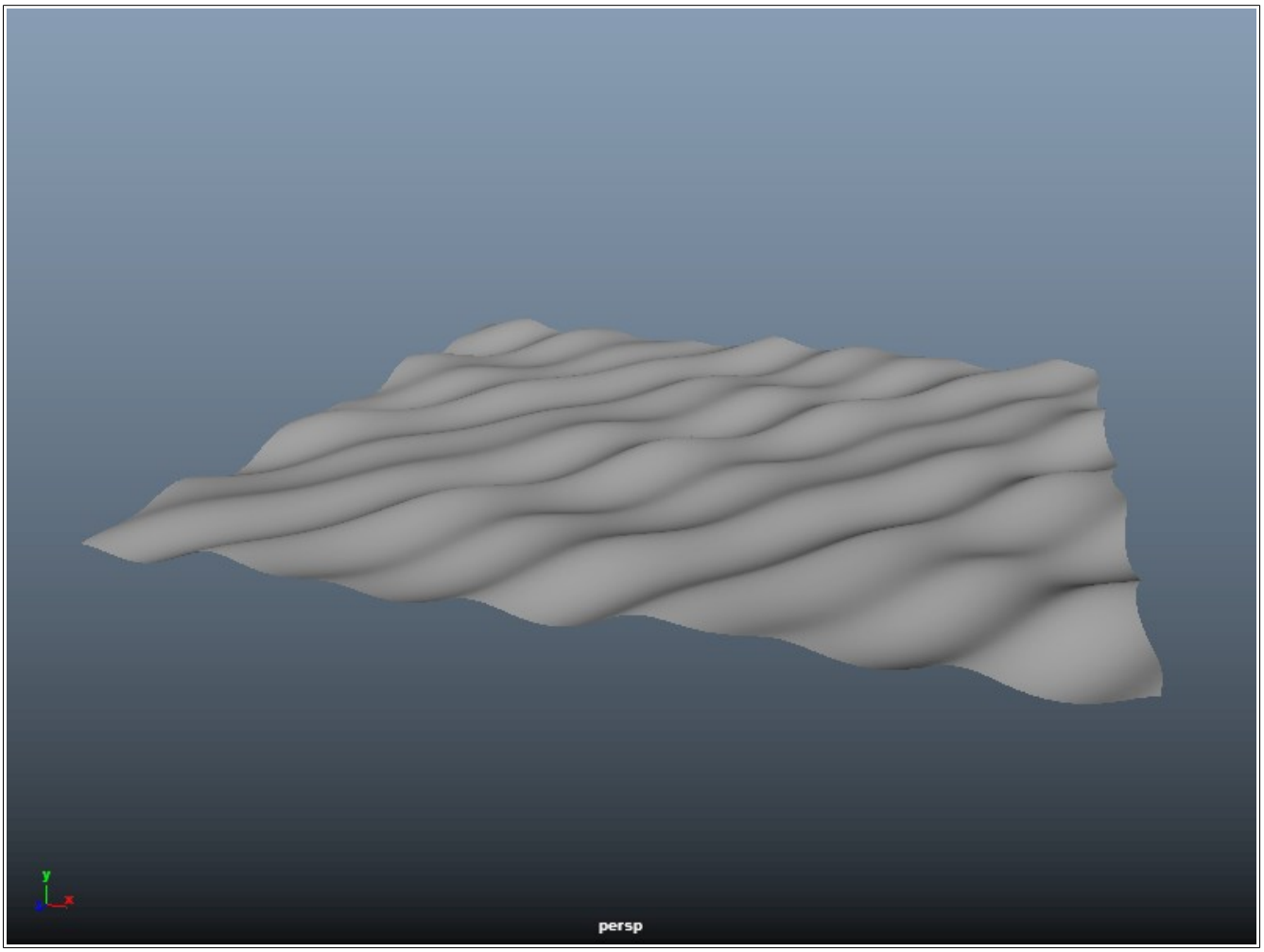


Description: torus knot (TK) curve node

Features:

1) node input params – p, q, segments

2) the algorithm samples the parametric TK equation to construct the Bezier curve control points
3) least-squares algorithm sets the tangents to minimize the distance between the ground truth TK curve and the Bezier curve



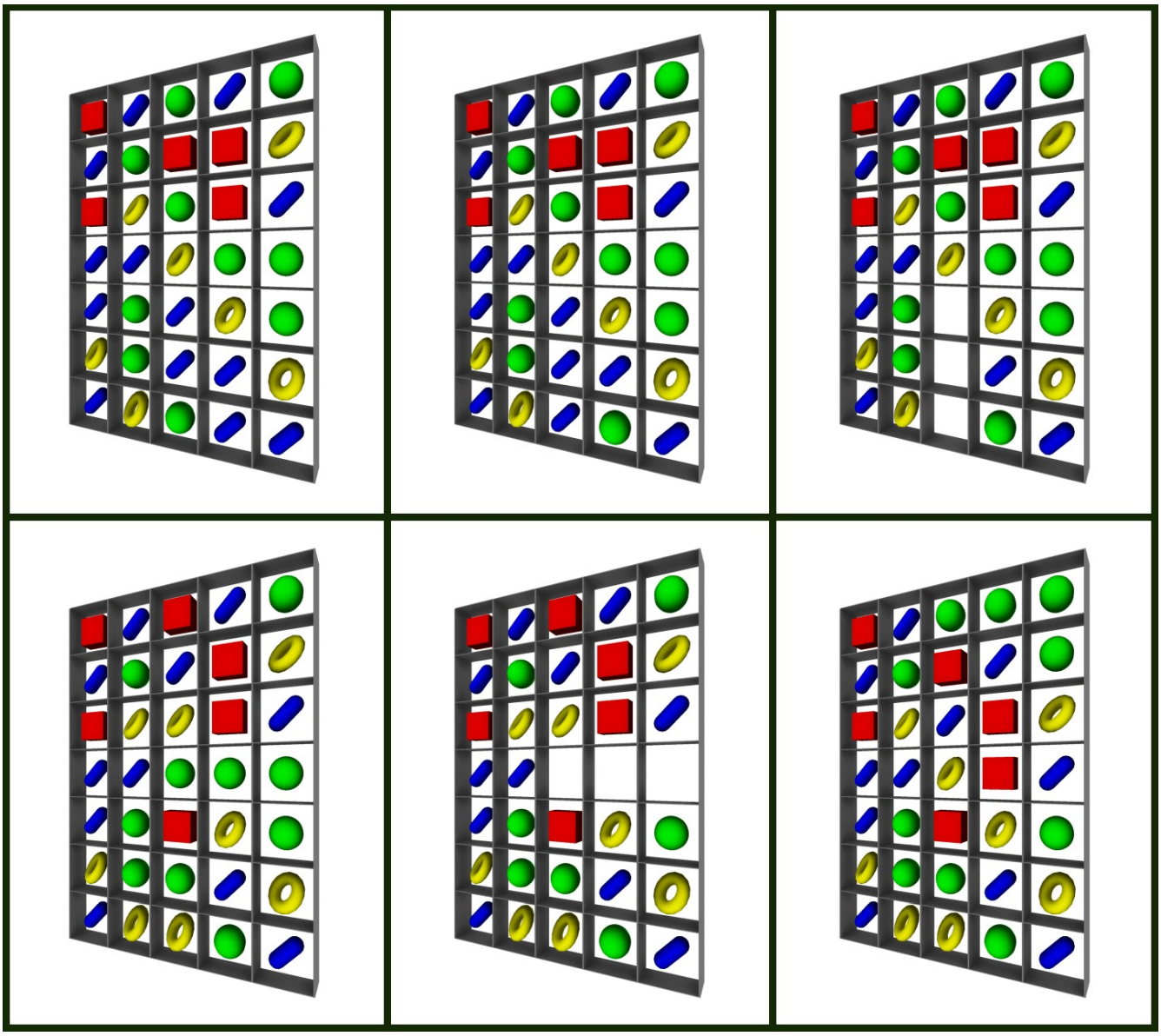
Description: Gerstner wave

Features:

1) the node deforms an input poly plane to Gerstner waves

References:
GPU Gems - Effective Water Simulation from Physical Models

Autodesk Maya – Python + PyQt



Description: Candy Crush Saga style game with Maya primitives

Features:

1) game logic scripted in python
2) mesh creation, key frames, shader setup in python
3) basic PyQt GUI

NextLimit RealFlow

Description: Dyverso SPH liquid simulation

Features:

1) liquid pouring into a glass with solid object inside interacting with the flow

Description: SPH melt simulation

Features:

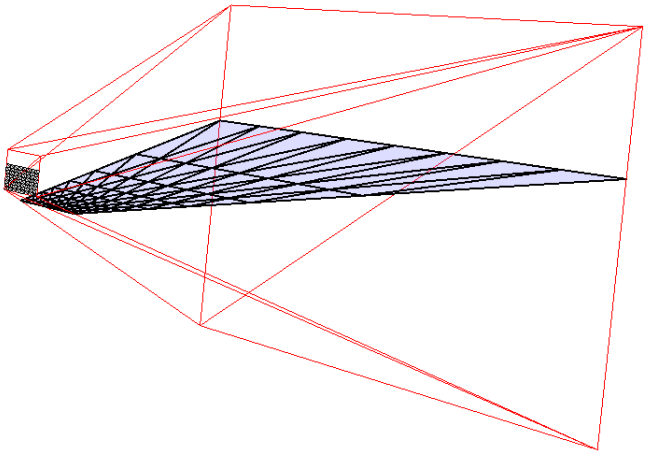
1) base mesh filled with frozen particles
2) particles are set free based on particle neighbor count
3) the object melts from its surface first

Simulating Ocean Water

Description: height field for ocean waves based on a statistical oceanographic model

Algorithm Overview:

1) Fast Fourier Transform (FFT) on the amplitudes defined by the Phillips-spectrum to obtain the wave height realization – OpenCL
2) 3 distinct FFT stages calculate the displacement field – height field + choppy effect
3) 2D FFT – local memory radix with traspose, cl-gl interop for height/normal map
4) Preetham Sky Model for background full screen quad – GLSL
5) screen space projected grid mesh visualization – GLSL



References:

Simulating Ocean Waves by Jerry Tessendorf