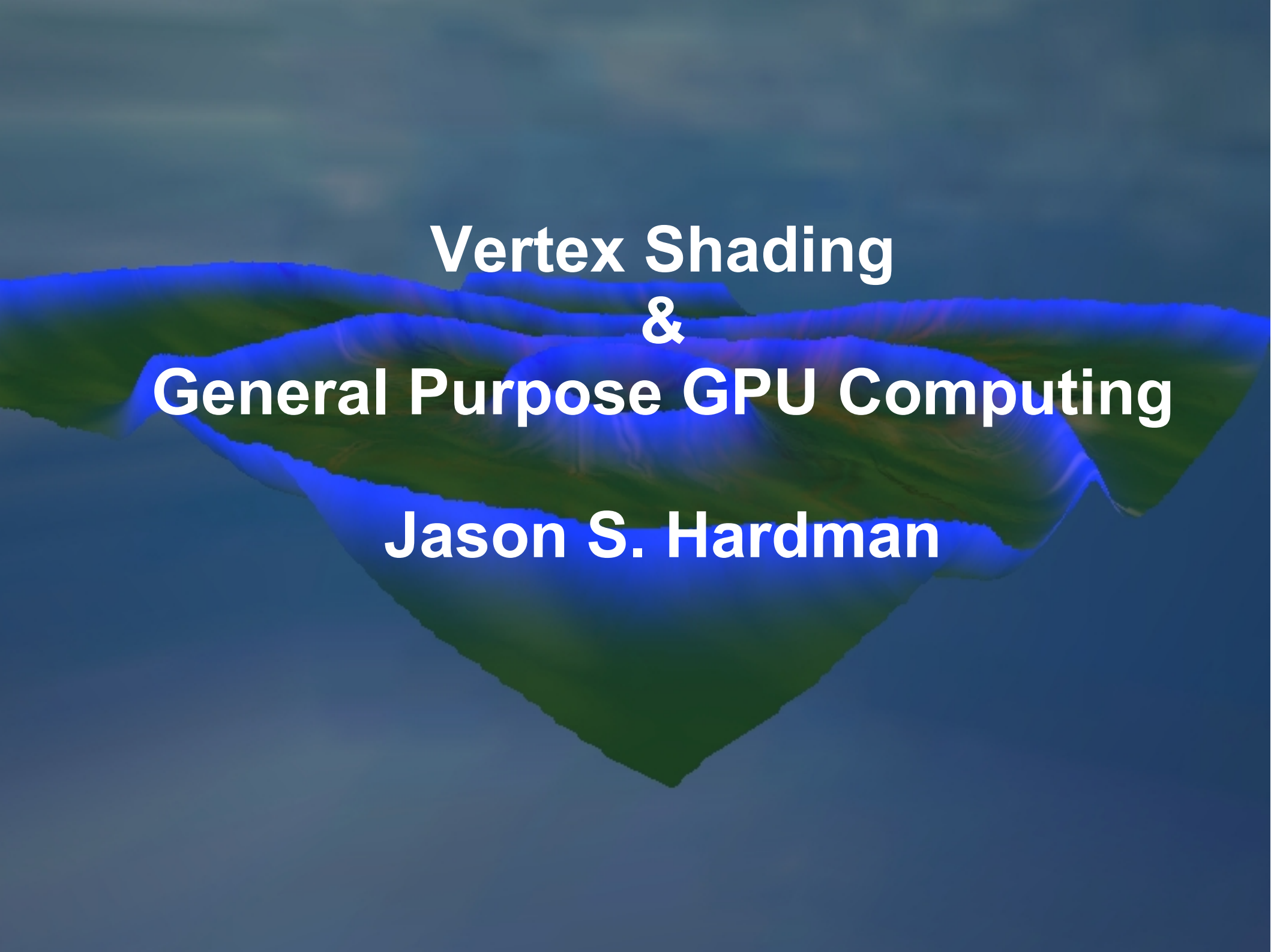

Vertex Shading
&
General Purpose GPU Computing
Jason S. Hardman

The Vertex Shader

What is the Vertex Shader?

Allows parallel manipulation of vertices on graphics hardware.

Vertex processor allows SIMD style processing.

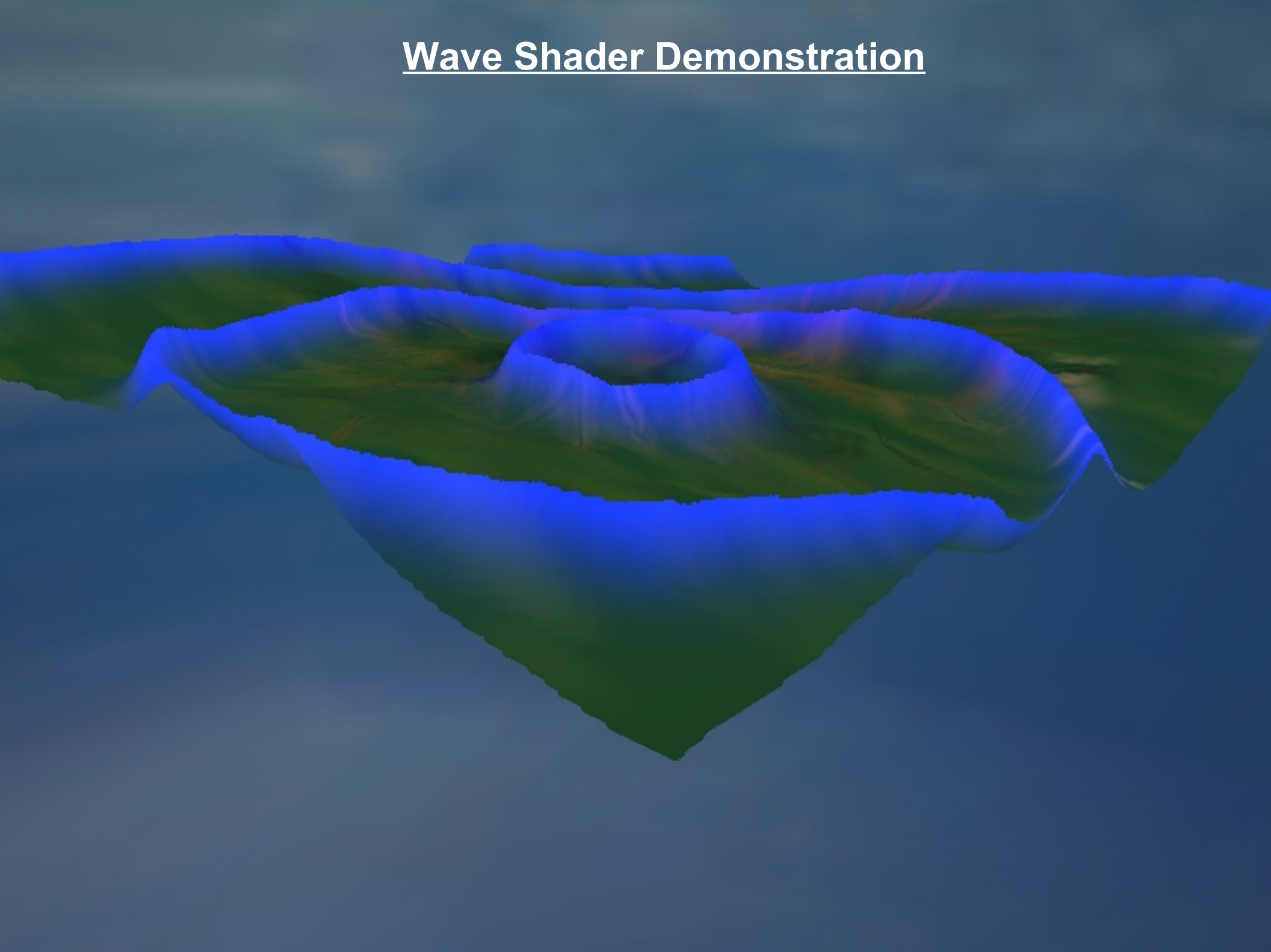
C-style programming language is easy to use.

- GLSL(OpenGL)
- HLSL (DirectX)

Calculate per-vertex values for the fragment shader.

Extremely usefull for physics based simulations.

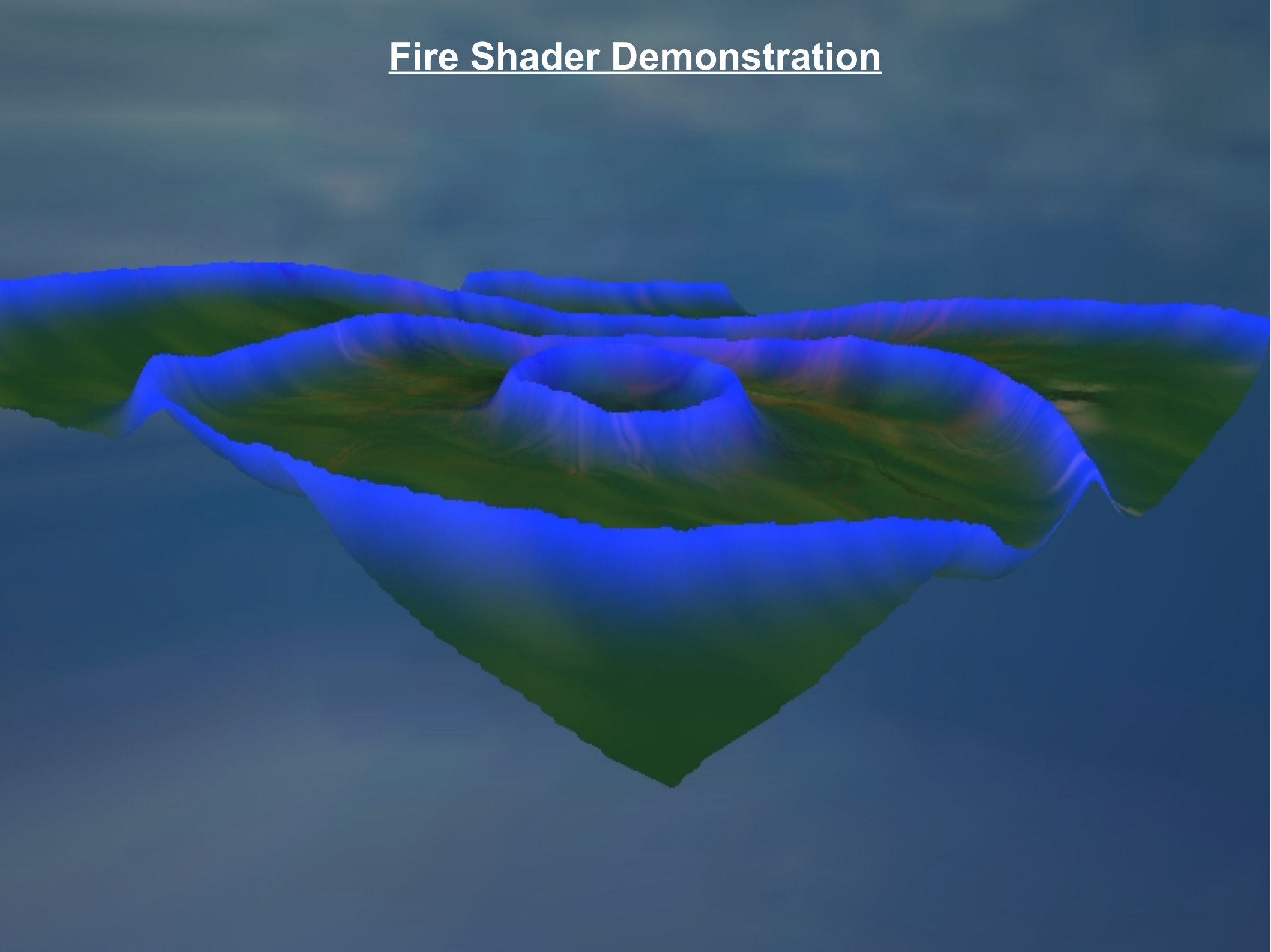
Wave Shader Demonstration



Wave Shader Demonstration

```
// Distance between Player and Vertex
float groundZeroDistance = abs( distance( inPos, playerPosition) );


// Distance between vertex and wave crest
float waveDistance = (groundZeroDistance - waveRadius);


// Create Waveform
float waveForm = 1.0f / (1.0f + amplitude + cos(waveDistance * 3.14f / frequency ));


// New vertex height based on wave form
inPos.y = inPos.y + richtor * waveForm;


// Modify the color based on height
Output.Color.b = waveForm * amplitude + blueShift;
```

Fire Shader Demonstration



Fire Shader Demonstration

```
// Distance between Player and Vertex
float groundZeroDistance = abs( distance( inPos, playerPosition) );

// Distance between vertex and wave crest
float waveDistance = abs( (groundZeroDistance - waveRadius) );

// Create Fire
float waveForm =
      1.0f/ (amplitude + sqrt( abs( waveDistance * cos( waveDistance / frequency) )));

// New vertex height based on wave form;
inPos.y = inPos.y + richtor * waveForm ;

// Now burn it down based on time
inPos.y -= waveRadius/numFrames * inPos.y;

if(inPos.y < 0.0f){
      inPos.y = 0.0f;
}

// Pass red to fragment shader based on height
Output.Color.r = waveForm * redShift;
```

General Purpose GPU Computing

What Is General Purpose GPU Computing?

- Harnesses the GPU to process general purpose data.
- Trick the GPU into thinking it's processing graphics data.
- Extremely fast.... 13 GFlops or four 3.0 GHZ CPUs in parallel.

How Does It Work?

- Textures == Memory Pages
- Shader Code == Atomic Instruction Units
- One-to-one Pixel-to-Texel mapping
- Render viewport sized quad to a texture

Why Aren't We All Using GP-GPU?

- Translating from Graphics to General Computation terminology.
- Operations must be highly parallel to achieve good results.
- Limited data type capabilities (eg- floats between 0-1 or ints).
- Very slow bridge between CPU and GPU.
- The bleeding edge... uncharted technology.

Current State Of Research

- Bringing the GPU to the (programmer) masses.
- Created a harness to simplify and translate operations.
- Operating system architecture simulates CPU instruction cycle.

Next Steps

- Using the vertex processor to handle memory management.
- Minimizes data reorganization by CPU.
- Less data traveling across the *SLOW* CPU-GPU bridge.

Further Down The Road... Back To Basics

- Independent GPU computation... a GPU mini-OS.
- Robust “assembly-code” library of shaders.
- Each rendering pass becomes a clock-cycle.


Thank You
:- j