CMSC 491G/691G

Computer Graphics for Games

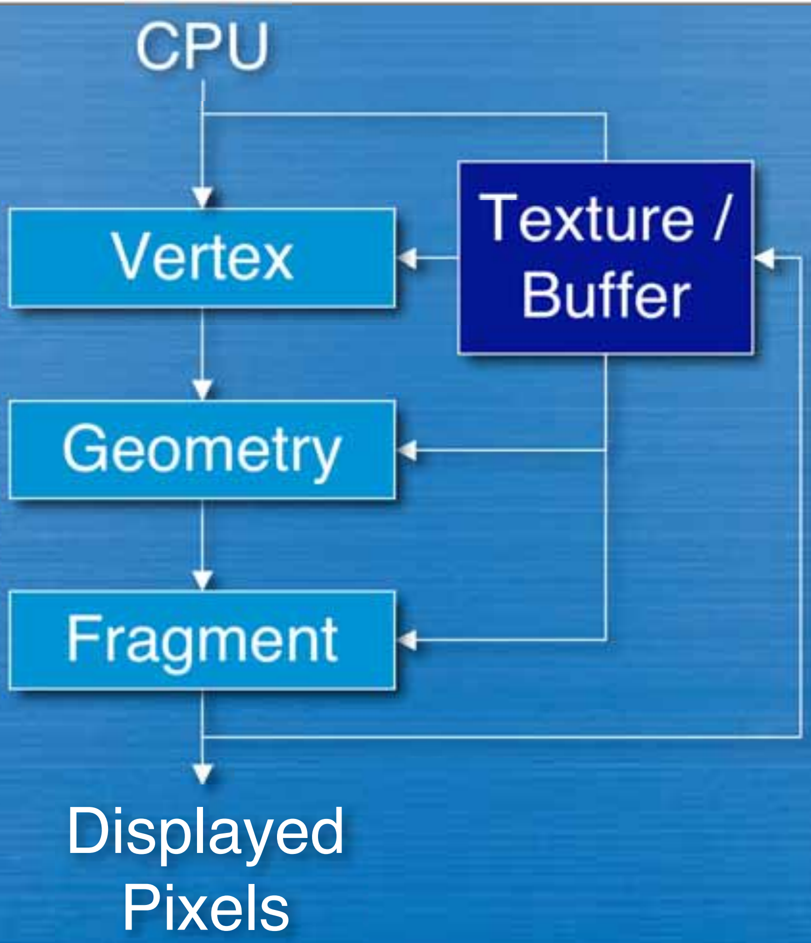
Marc Olano

GPU

- GPU: Graphics Processing Unit
  - Designed for real-time graphics
  - Present in almost every PC
  - Increasing realism and complexity

Americas Army

GPU computation



Low-level code

```
!!ARBvp1.0
# Transform the normal to view space
TEMP Nv,Np;
DP3 Nv.x,state.matrix.modelview.invtrans.row[0],vertex.normal;
DP3 Nv.y,state.matrix.modelview.invtrans.row[1],vertex.normal;
DP3 Nv.z,state.matrix.modelview.invtrans.row[2],vertex.normal;
MAD Np,Nv,{.9,.9,.9,0},{0,0,0,1};

# screen position from vertex
TEMP Vp;
DP4 Vp.x, state.matrix.mvp.row[0], vertex.position;
DP4 Vp.y, state.matrix.mvp.row[1], vertex.position;
DP4 Vp.z, state.matrix.mvp.row[2], vertex.position;
DP4 Vp.w, state.matrix.mvp.row[3], vertex.position;
[…]
# interpolate
MAD Np, Np, -vertex.color.x, Np;
MAD result.position, Vp, vertex.color.x, Np;
END
```

High-level code

```
void main() {
    vec4 Kin = gl_Color;        // key input

    // screen position from vertex, texture and normal
    vec4 Vp = ftransform();
    vec4 Tp = vec4(gl_MultiTexCoord0.xy*1.8-.9, 0,1);
    vec4 Np = vec4(nn*.9,1);

    // interpolate between Vp, Tp and Np
    gl_Position = Vp;
    gl_Position = mix(Tp,gl_Position,pow(1.-Kin.x,8.));
    gl_Position = mix(Np,gl_Position,pow(1.-Kin.y,8.));

    // copy to output
    gl_TexCoord[0] = gl_MultiTexCoord0;
    gl_TexCoord[1] = Vp;
    gl_TexCoord[3] = Kin;
}
```

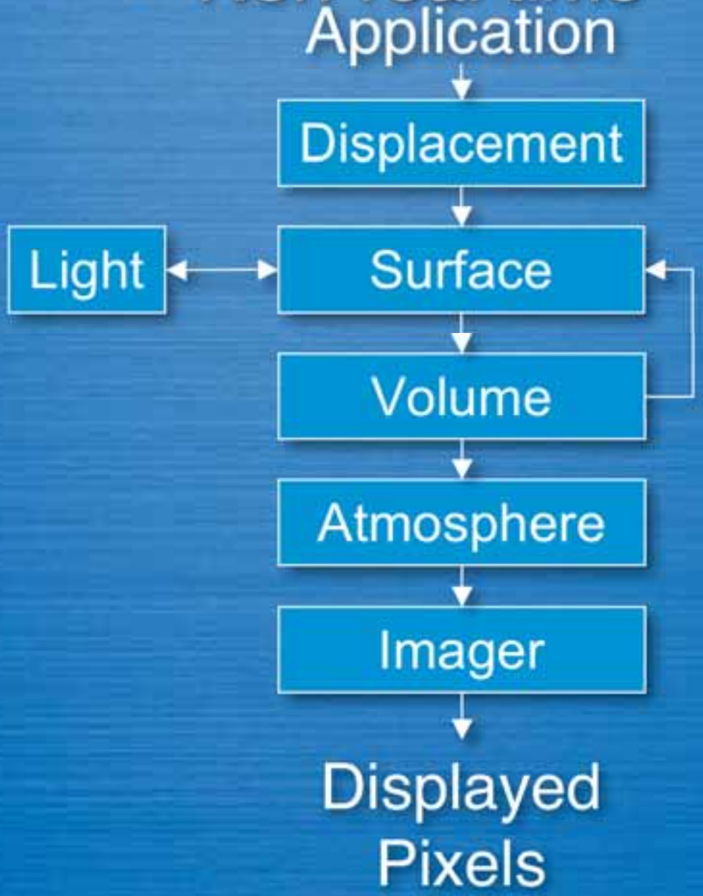
Non-real time vs. Real time

- Not real-time
  - Developed from General CPU code
  - Seconds to hours per frame
  - 1000s of lines
  - “Unlimited” computation, texture, memory, …

- Real-time
  - Developed from fixed-function hardware
  - Tens of frames per second
  - 1000s of instructions
  - Limited computation, texture, memory, …

Non-real time vs. Real-time

- Non-real time



Application

Displacement

Light

Surface

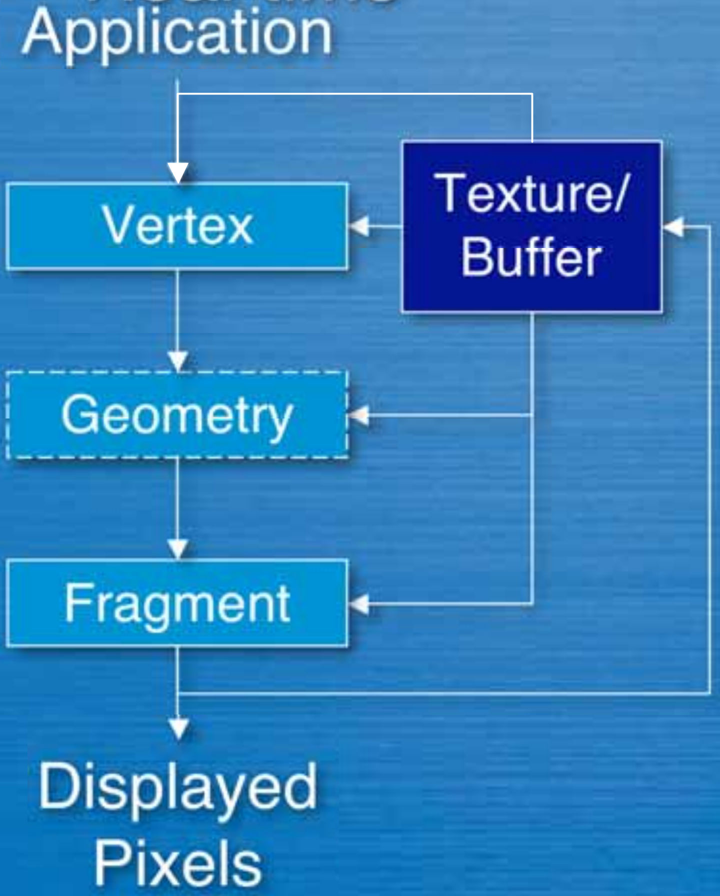
Volume

Atmosphere

Imager

Displayed Pixels

- Real-time



Application

Vertex

Texture/ Buffer

Geometry

Fragment

Displayed Pixels

History (not real-time)

- Testbed [Whitted and Weimer 1981]
- Shade Trees [Cook 1984]
- Image Synthesizer [Perlin 1985]
- RenderMan [Hanrahan and Lawson 1990]
- Multi-pass RenderMan [Peercy et al. 2000]
- GPU acceleration [Wexler et al. 2005]

History (real-time)

- Custom HW [Olano and Lastra 1998]
- Multi-pass standard HW [Peercy et al. 2000]
- Register combiners [NVIDIA 2000]
- Vertex programs [Lindholm et al. 2001]
- Compiling to mixed HW [Proudfoot et al. 2001]
- Fragment programs
- Standardized languages
- Geometry shaders [Blythe 2006]

Choices

- OS: Windows, Mac, Linux
- API: DirectX, OpenGL
- Language: HLSL, GLSL, Cg, …
- Compiler: DirectX, OpenGL, Cg, ASHLI
- Runtime: CgFX, ASHLI, OSG (& others), sample code

Major Commonalities

- Vertex, Geometry & Fragment/Pixel
- C-like, if/while/for
- Structs & arrays
- Float + small vector and matrix
  - Swizzle & mask (a.xyz = b.xxw)
- Common math & shading functions