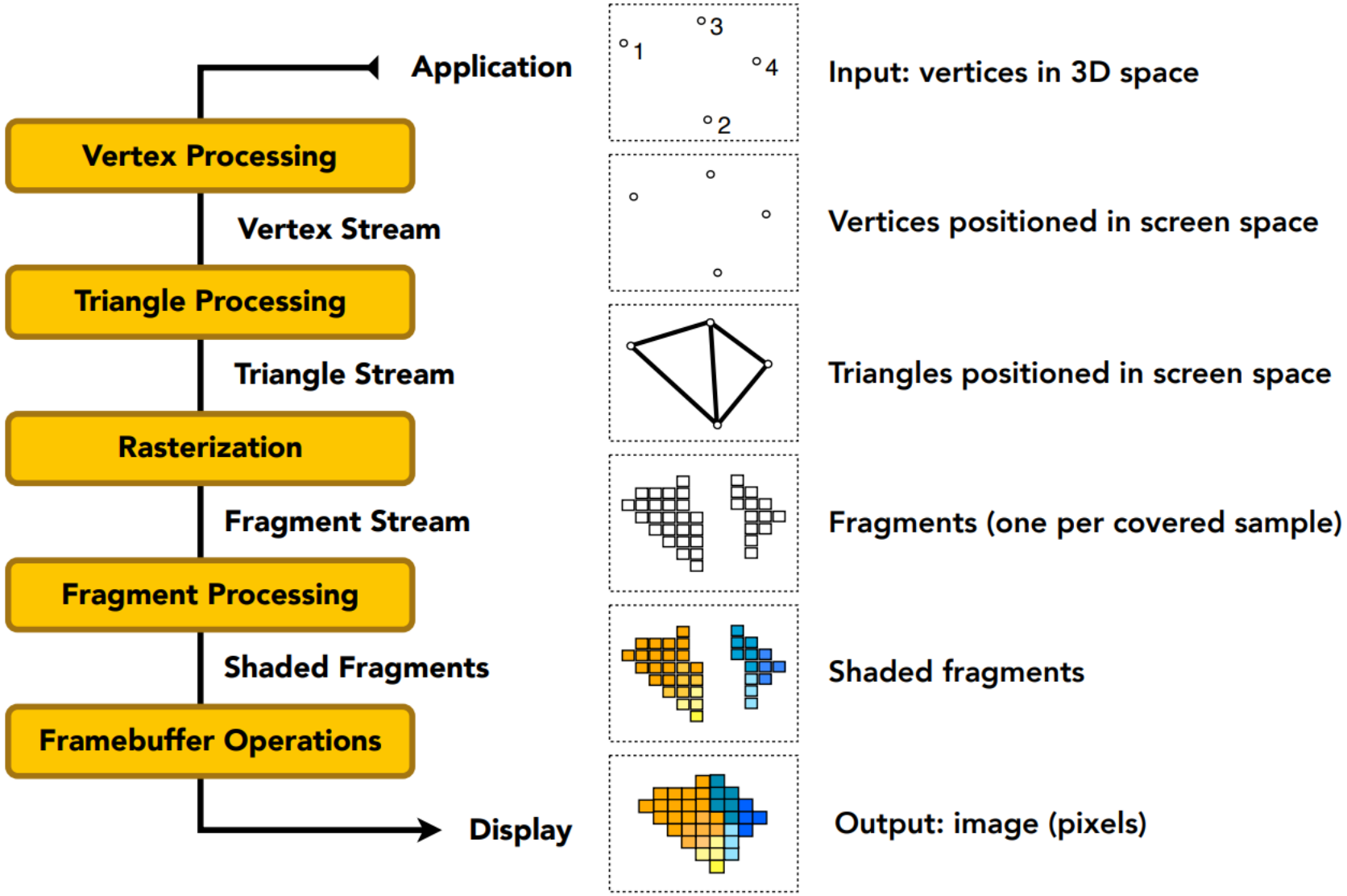
Graphics Pipeline

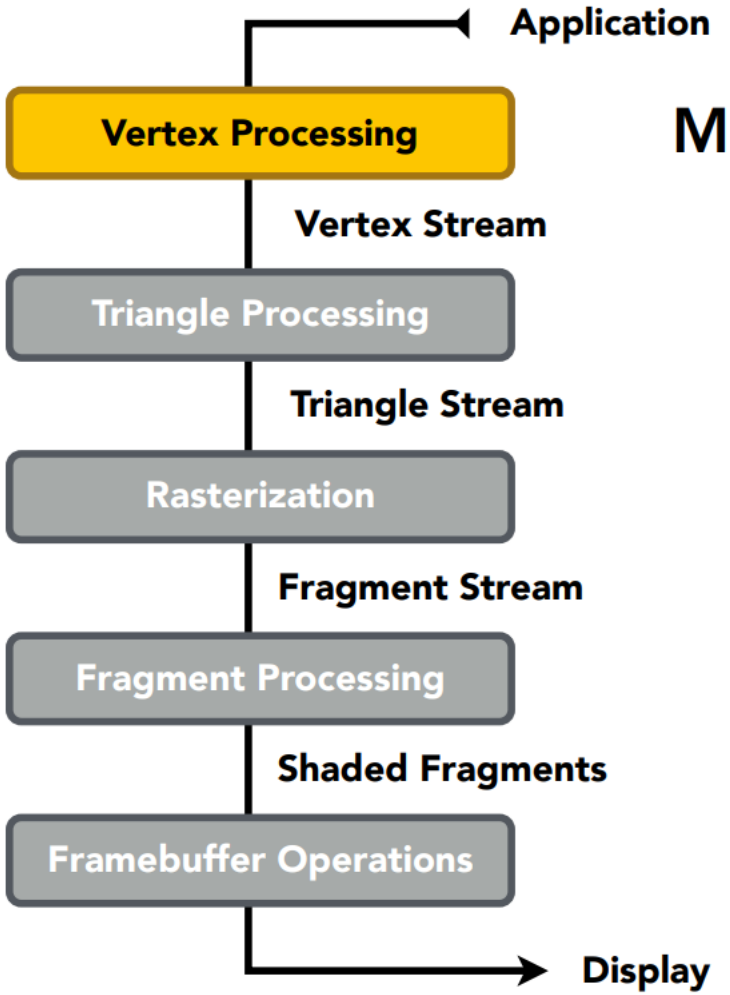
Raster Rendering

Rendering Pipeline

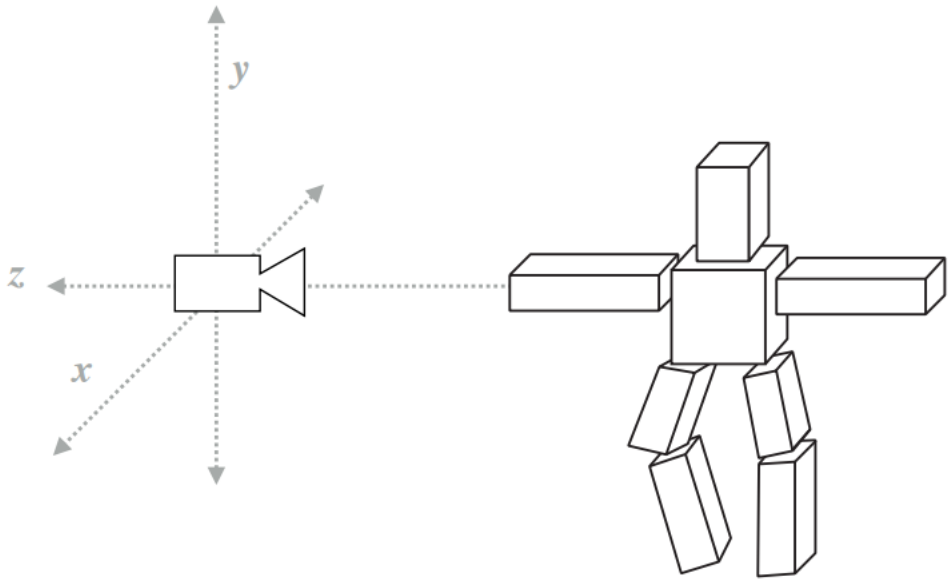


From Lingqi Yan, GAMES 101

Rendering Pipeline

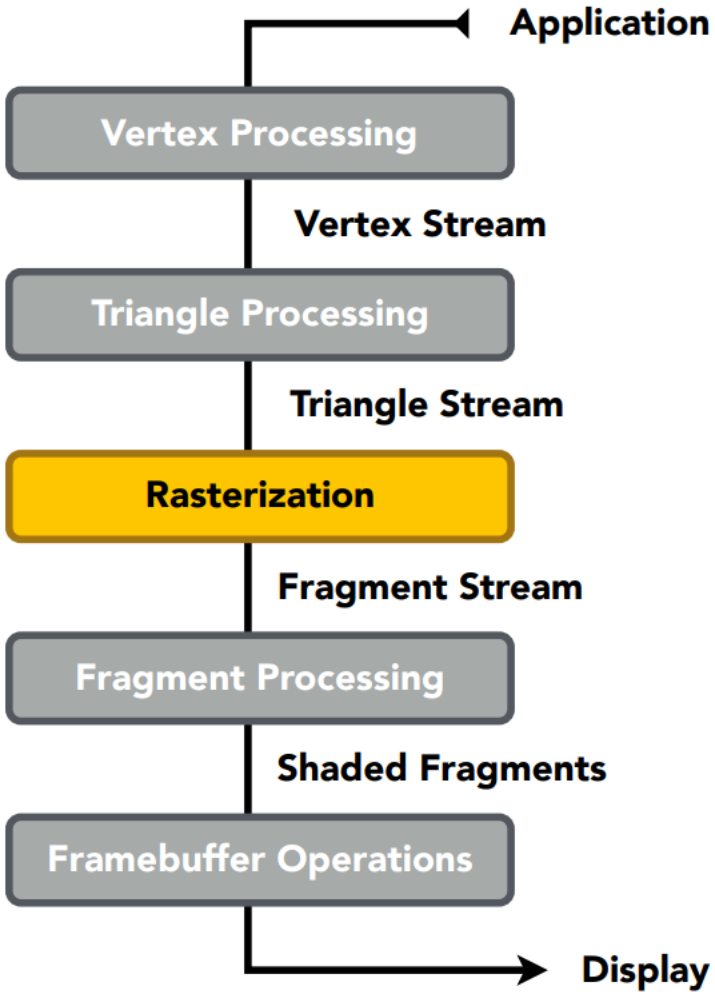


Model, View, Projection transforms

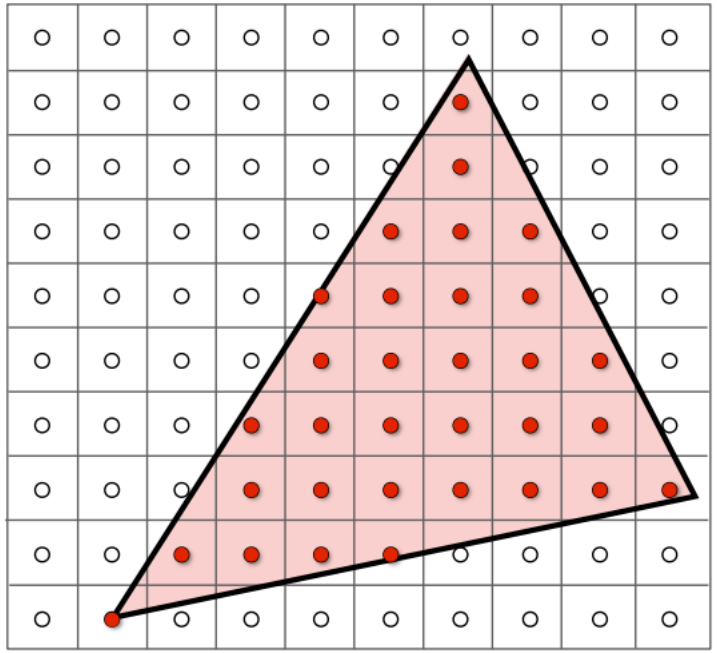


From Lingqi Yan, GAMES 101

Rendering Pipeline

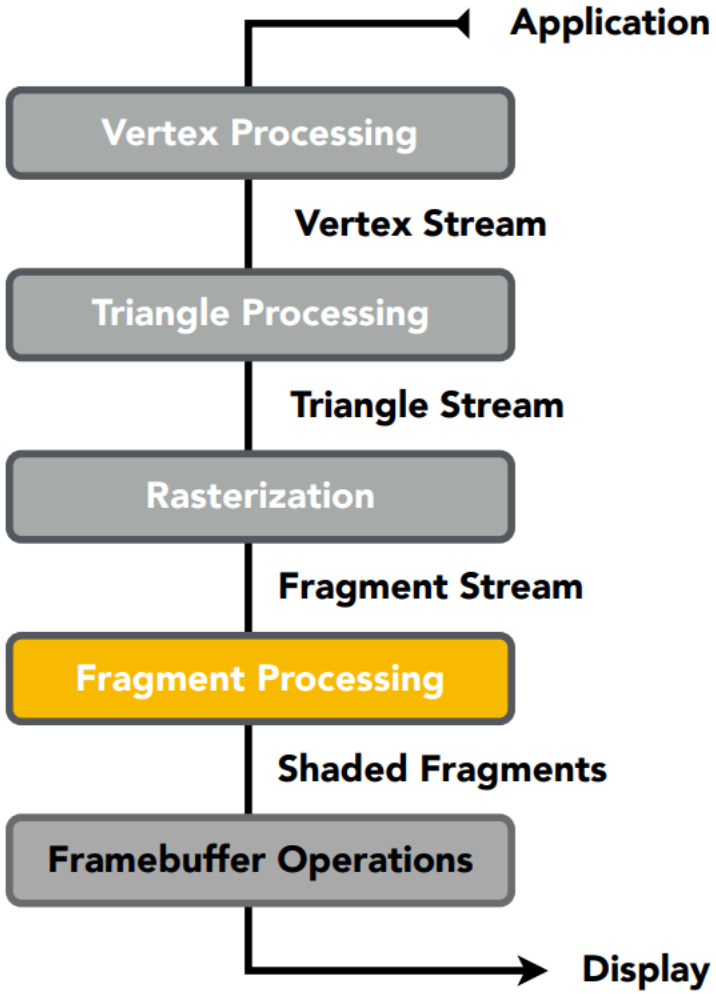


Sampling triangle coverage



From Lingqi Yan, GAMES 101

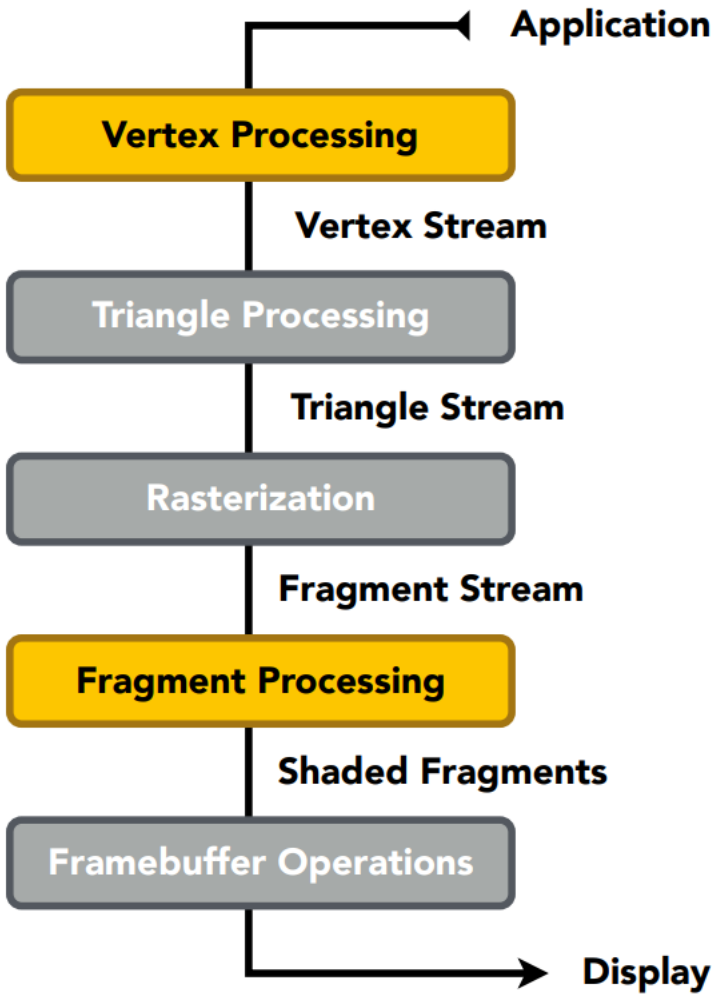
Rendering Pipeline

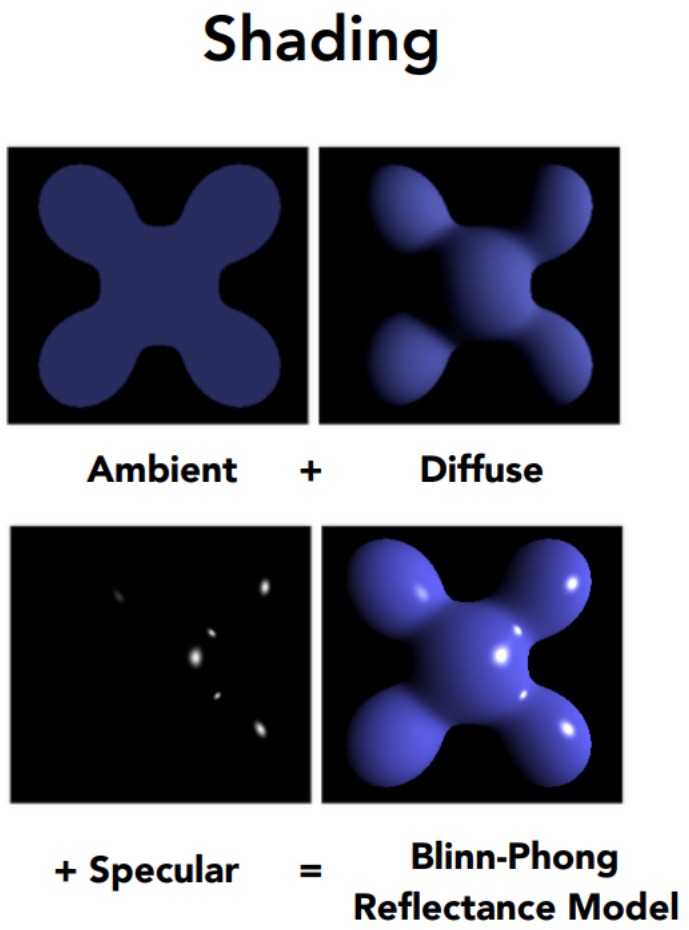


Z-Buffer Visibility Tests

From Lingqi Yan, GAMES 101

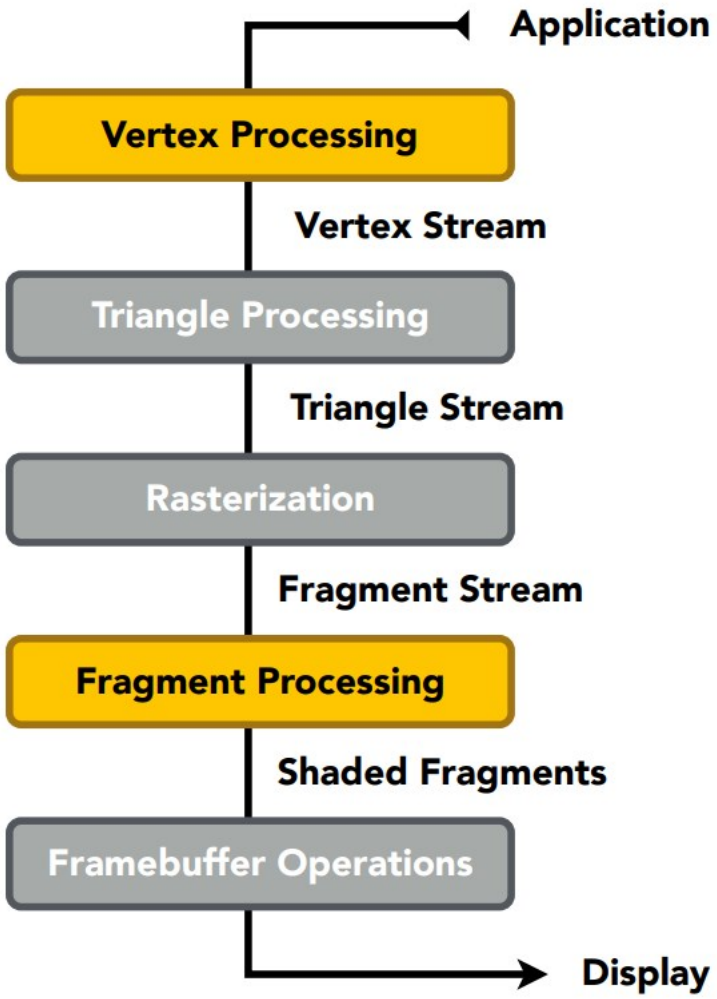
Rendering Pipeline





From Lingqi Yan, GAMES 101

Rendering Pipeline

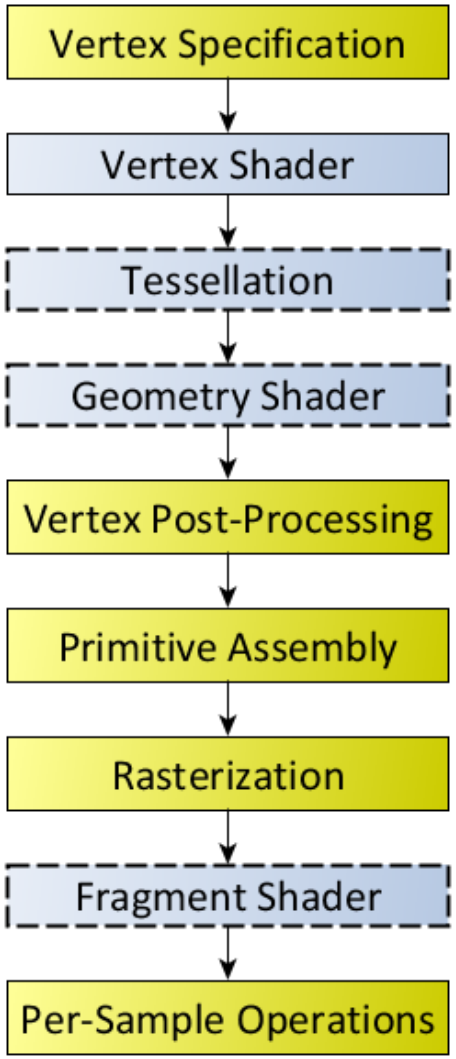


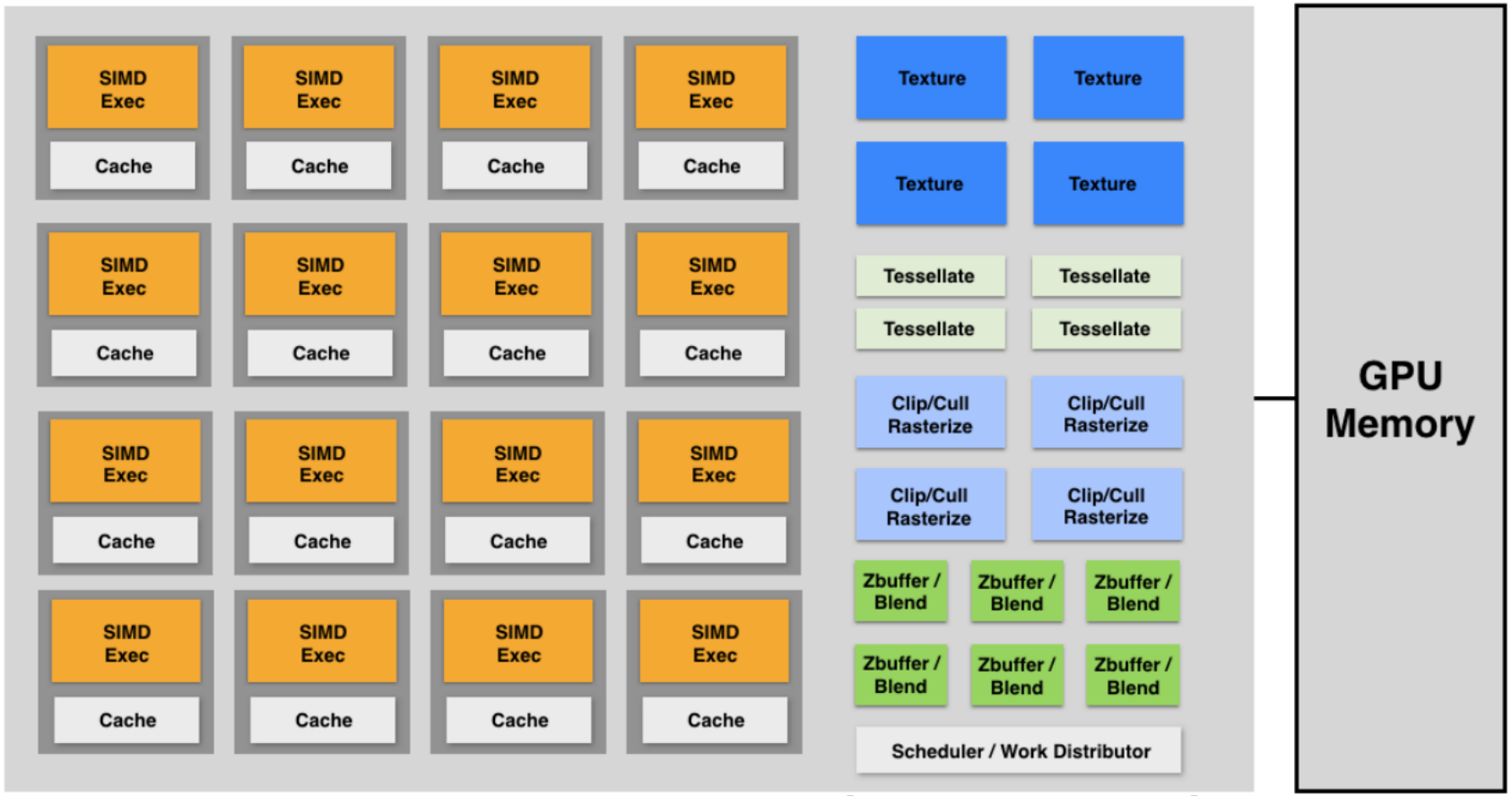
Texture mapping
(introducing soon)

From Lingqi Yan, GAMES 101

Graphics API

Modern GPU





Modern GPUs offer ~2-4 Tera-FLOPs of performance for executing vertex and fragment shader programs

Tera-Op's of fixed-function compute capability over here

Programming on GPU

OpenGL, a cross-language, cross-platform API for 2D and 3D graphics.

WebGL, a JavaScript API for 2D/3D graphics within any compatible web browser without the use of plug-ins.

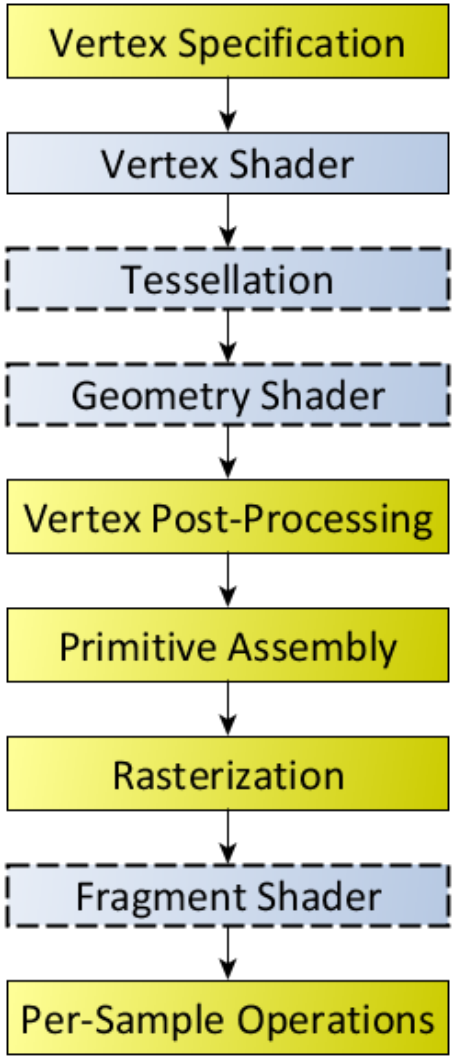
Metal, created by Apple

DirectX 12 Ultimate, the API that features the most advanced features of NVIDIA GPU.

OpenGL ES, a subset of OpenGL API on embedded and mobile systems

Vulkan, built upon components of AMD's Mantle API

Programming on GPU



Programmable with Shaders

Other stages are NOT Programmable!

Programmable with Shaders

Image from OpenGL Wiki

Shader

- Vertex shader: parallel over all vertices
  - Input some attributes of each vertex, like position, normal, color…
  - Output some attributes of each vertex, like normal, color, depth…
- Fragment shader: parallel over all pixels of all segments
  - Input values per pixel: interpolated from vertex shader output
  - Output pixel color, depth…
- Shader language: GLSL (OpenGL), HLSL (DirectX), Houdini VEX (Houdini)…

GLSL Shader Example: Phong Shading

Vertex shader

```
#version 330 core
layout (location = 0) in vec3 aPos;
layout (location = 1) in vec3 aNormal;

out vec3 FragPos;
out vec3 Normal;

uniform mat4 model;
uniform mat4 view;
uniform mat4 projection;

void main()
{
    FragPos = vec3(model * vec4(aPos, 1.0));
    Normal = mat3(transpose(inverse(model))) * aNormal;

    gl_Position = projection * view * vec4(FragPos, 1.0);
}
```

Fragment shader

```
#version 330 core
out vec4 FragColor;

in vec3 Normal;
in vec3 FragPos;

uniform vec3 lightPos;
uniform vec3 viewPos;
uniform vec3 lightColor;
uniform vec3 objectColor;

void main()
{
    // ambient
    float ambientStrength = 0.1;
    vec3 ambient = ambientStrength * lightColor;

    // diffuse
    vec3 norm = normalize(Normal);
    vec3 lightDir = normalize(lightPos - FragPos);
    float diff = max(dot(norm, lightDir), 0.0);
    vec3 diffuse = diff * lightColor;

    // specular
    float specularStrength = 0.5;
    vec3 viewDir = normalize(viewPos - FragPos);
    vec3 reflectDir = reflect(-lightDir, norm);
    float spec = pow(max(dot(viewDir, reflectDir), 0.0), 32);
    vec3 specular = specularStrength * spec * lightColor;

    vec3 result = (ambient + diffuse + specular) * objectColor;
    FragColor = vec4(result, 1.0);
}
```

GPU History

- 1976-1984: only 2D hardware acceleration, 3D on CPU
- Mainly display adapters, video accelerators…

IBM PC's Monochrome Display Adapter

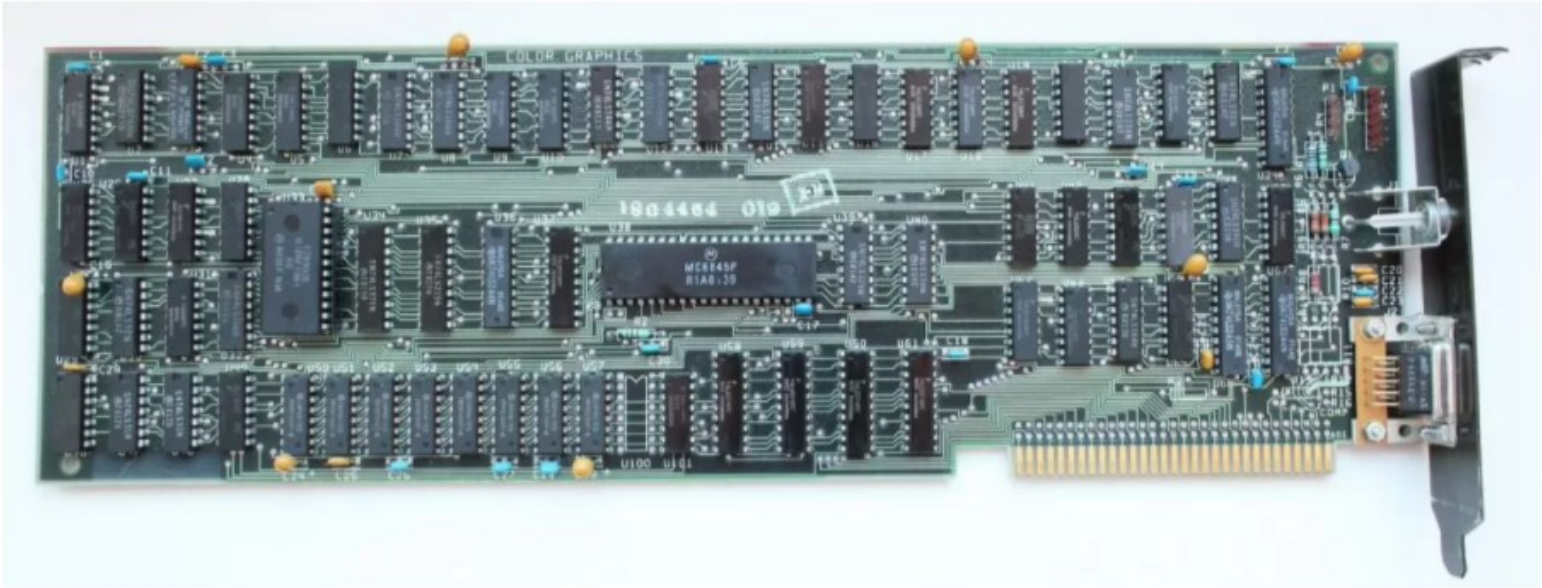


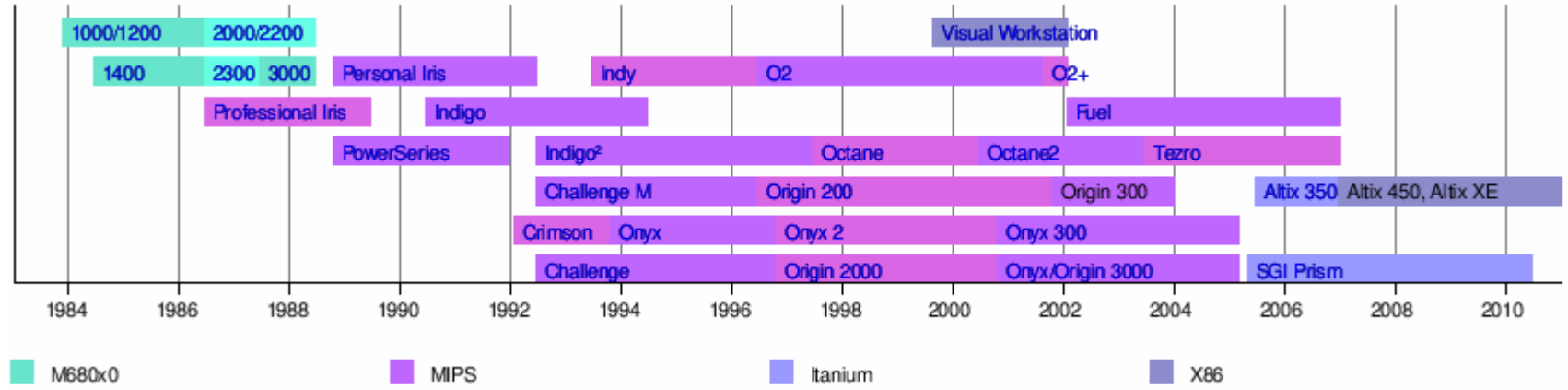
Image from Techspot

GPU History

- 1984-2005: Silicon Graphics (SGI) workstation

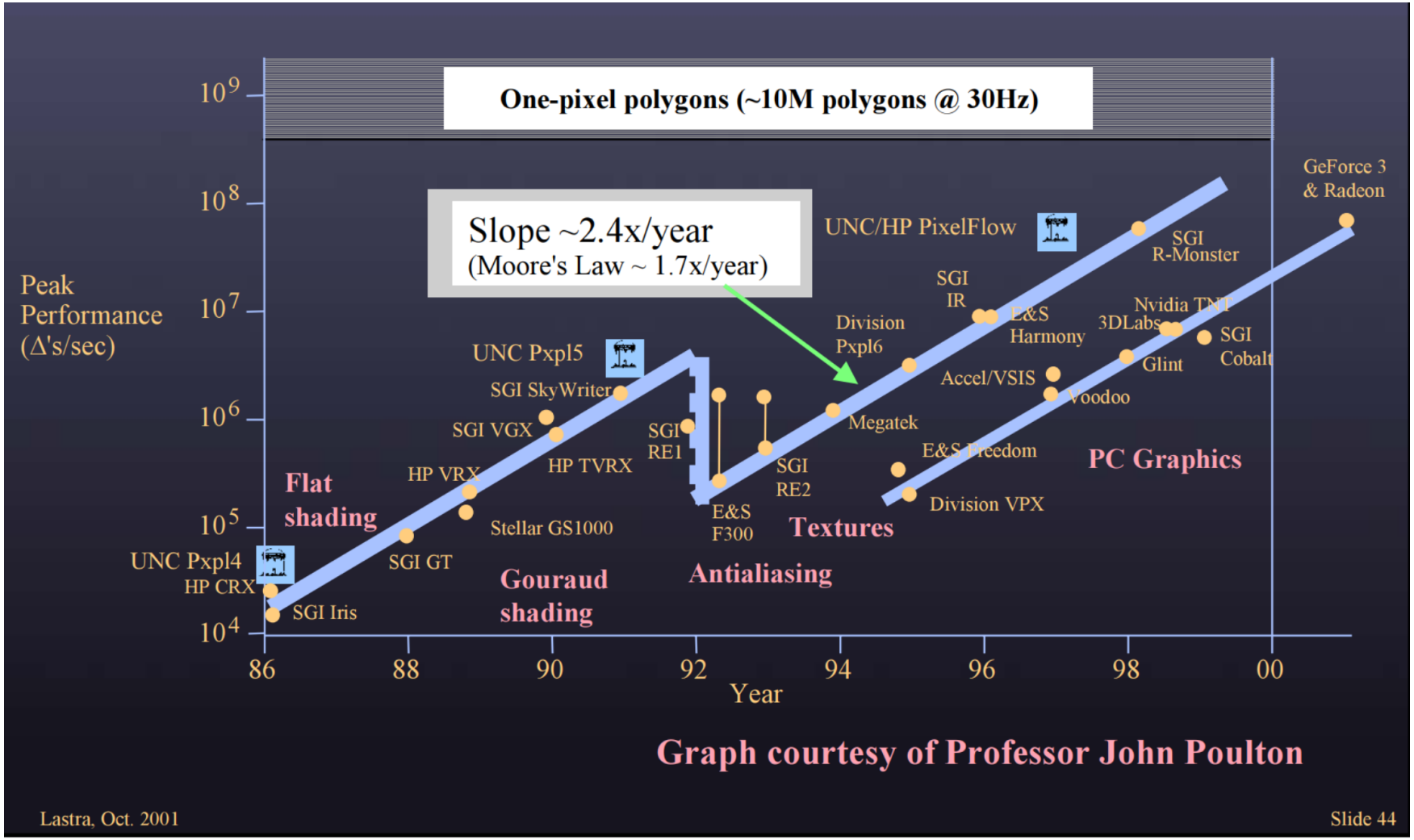
sgi®

Type	Public
Traded as	NYSE: SGI OTC Pink: SGID.pk Nasdaq: SGIC
Industry	Computer hardware and software
Founded	November 9, 1981; 40 years ago Mountain View, California, U.S.[1]
Defunct	May 11, 2009; 12 years ago
Fate	Chapter 11 bankruptcy; assets acquired by Rackable Systems, which renamed itself Silicon Graphics International Corp.
Headquarters	Sunnyvale, California, U.S.
Key people	Jim Clark Wei Yen Kurt Akeley Ed McCracken Thomas Jermoluk Marc Hannah



https://www.youtube.com/watch?v=lEcw_oTKTa0

Graphics Hardware



GPU History

- 1995-1999: 3Dfx VOODOO, 3D hardware acceleration

Fast forward: GLQuake released in 1997 versus original Quake

- 2000-2006: Nvidia vs ATI
- Programmable pipeline
  - vertex/fragment shader
  - 2002: HLSL by DirectX 9, GLSL by OpenGL
  - 2006: Geometry Shader
- 2006: ATI bought by AMD

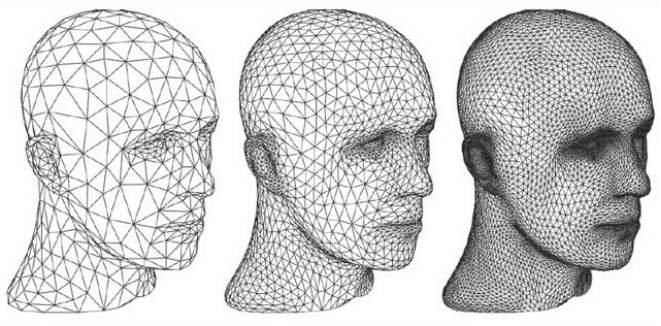
Nvidia's stock GeForce 3

Image from Techspot

GPU History

- 2006-2013: general purpose GPU (GPGPU)
- 2007: CUDA by Nvidia
- 2009: tessellation shader (subdivision),

  compute shader



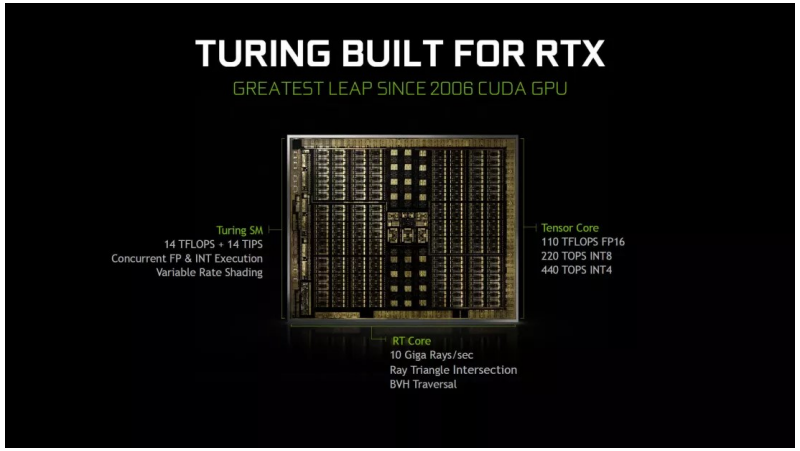
MSI's version of the GeForce 8800 GTX

Image from Techspot

GPU History

- 2013-now: faster and faster, for gamers and AI
- Nvidia RTX: hardware ray tracing
- Nvidia DLSS: deep-learning super sampling
- GPGPU: scientific computing,

  Crypto-mining, AI training…



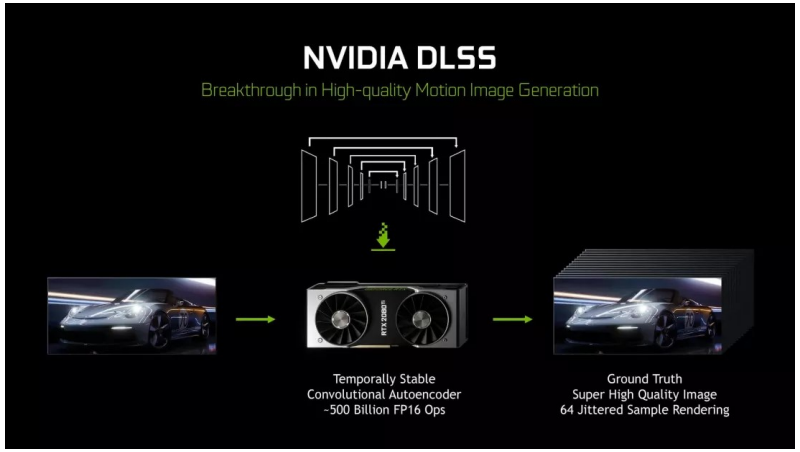
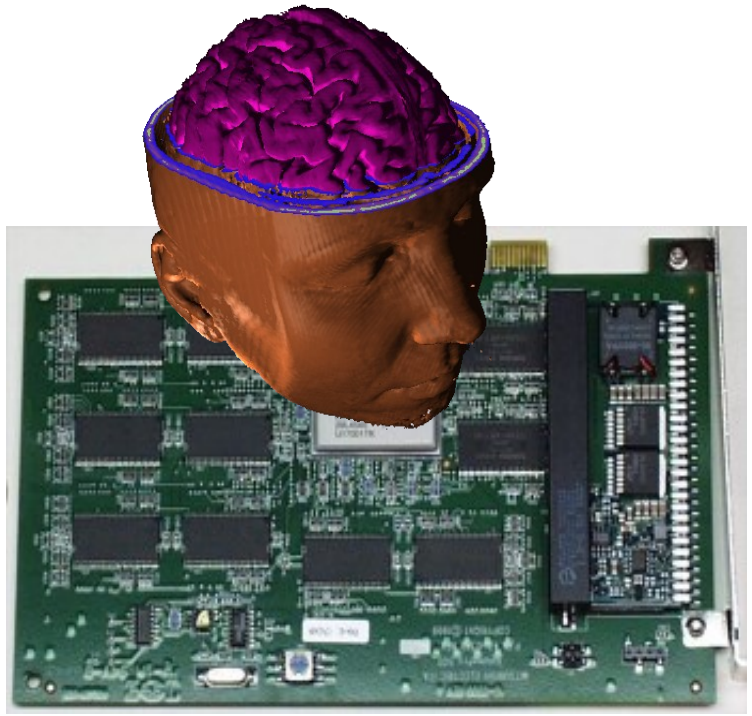


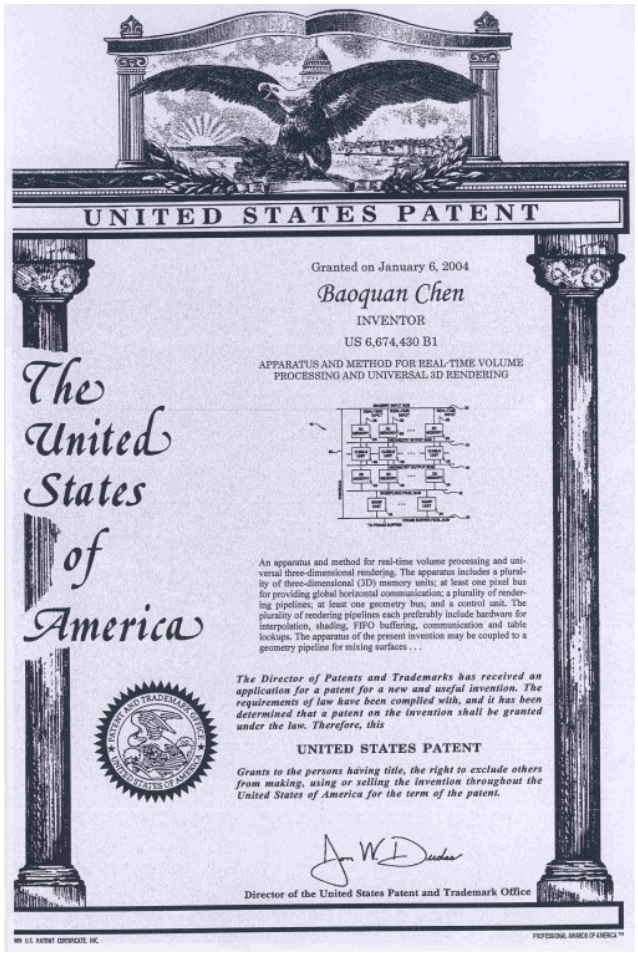
Image from Techspot

VolumePro:
Special purpose hardware for volume rendering



第五代*Cube*芯片设计与实现

- VolumePro (MITSUBISHI ELECTRIC), 1999
- Cube Engine (JRC Japan Radio Co.,Ltd.), 2000



UNITED STATES PATENT

Granted on January 6, 2004

Baoquan Chen

INVENTOR

US 6,674,430 B1

APPARATUS AND METHOD FOR REAL-TIME VOLUME PROCESSING AND UNIVERSAL 3D RENDERING

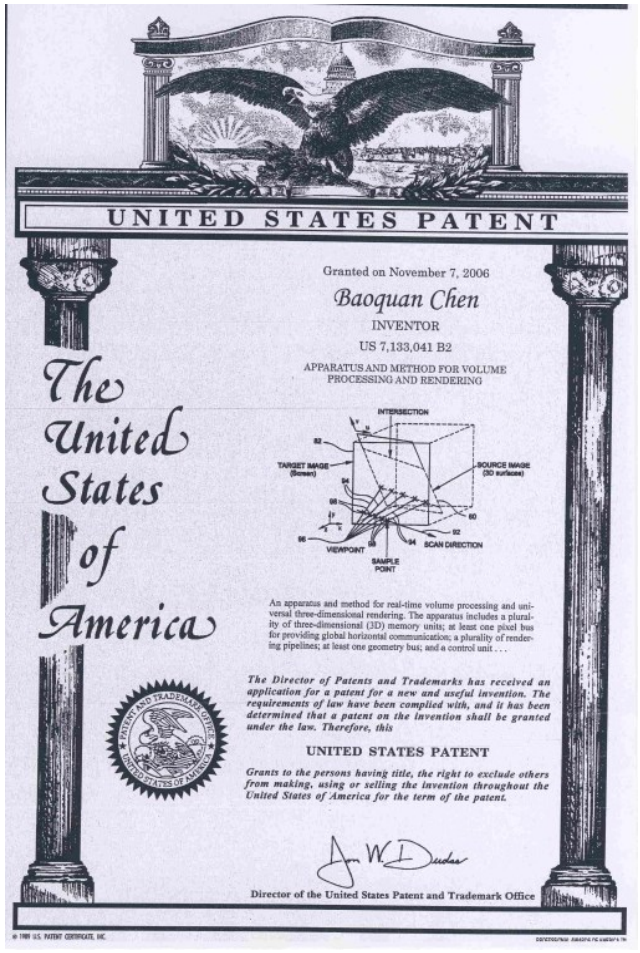
The United States of America

The Director of Patents and Trademarks has received an application for a patent for a new and useful invention. The requirements of law have been complied with, and it has been determined that a patent on the invention shall be granted under the law. Therefore, this

UNITED STATES PATENT

Grants to the persons having title, the right to exclude others from making, using or selling the invention throughout the United States of America for the term of the patent.

Director of the United States Patent and Trademark Office



UNITED STATES PATENT

Granted on November 7, 2006

Baoquan Chen

INVENTOR

US 7,133,041 B2

APPARATUS AND METHOD FOR VOLUME PROCESSING AND RENDERING

The United States of America

The Director of Patents and Trademarks has received an application for a patent for a new and useful invention. The requirements of law have been complied with, and it has been determined that a patent on the invention shall be granted under the law. Therefore, this

UNITED STATES PATENT

Grants to the persons having title, the right to exclude others from making, using or selling the invention throughout the United States of America for the term of the patent.

Director of the United States Patent and Trademark Office

Patents licensed to: GE, MITSUBISHI ELECTRIC, TERARECON, INC, VIATRONIX V3D - The new imaging reality, JRC Japan Radio Co.,Ltd.

Accelerated Rendering

Forward Rendering

- Assume N triangles and M light sources in the shading stage

```
For N triangles:
    For all pixels:
        For M light sources:
            color += shading(triangle, light)
        Test z-buffer and update
```

Complexity O(N*M)! Many shading results are overdrawed and discarded.

Early-Z

- Decide whether to perform shading by current z-buffer

```
For N triangles:
    For pixels not occluded:
        For M light sources:
            color += shading(triangle, light)
        Test z-buffer and update
```

If triangles are sorted by depth, the rendering time can be significantly reduced

No effect in the worst case!

Deferred Rendering

- Convert world-space scene to screen-space G-Buffer

```
For N triangles:
    Test depth and update G-Buffer
For all pixels:
    For M light sources:
        color += shading(G-Buffer, light)
```

Complexity O(N+M); Reduced overdrawing
High bandwidth overhead (G-Buffer can be large)
Computed G-Buffer can be used in post-processing

Deferred Rendering

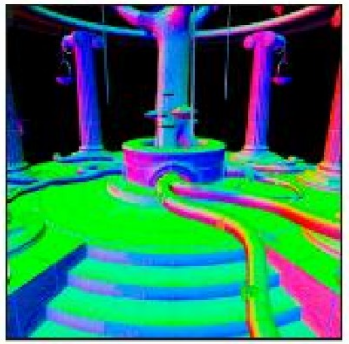
Typical G-Buffer Layout

	R	G	B	A
GB0	Normal (10:10)		Smoothness	MaterialId (2)
GB1	BaseColor			MatData(5)/Normal(3)
GB2	---------	MetalMask	Reflectance	AO
GB3	Radiosity/Emissive			

Albedo

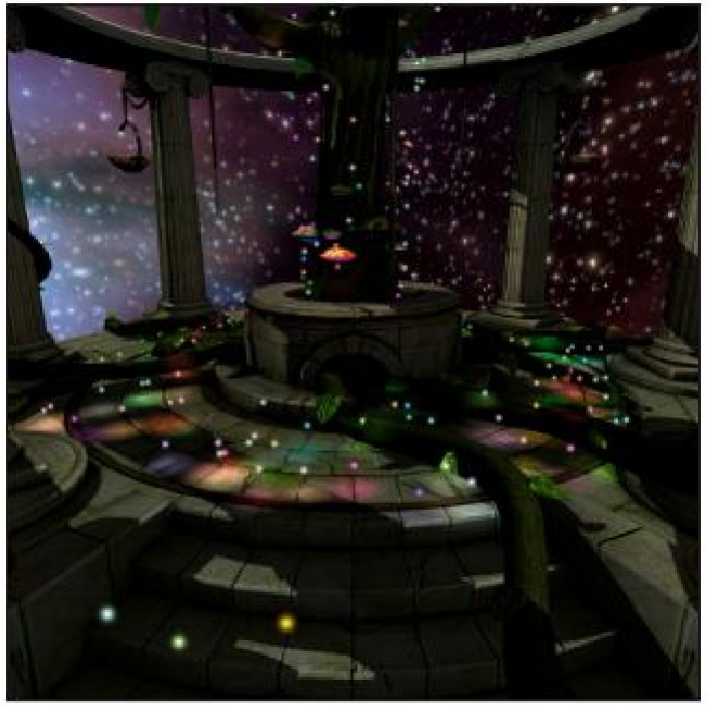
Specular

Normals



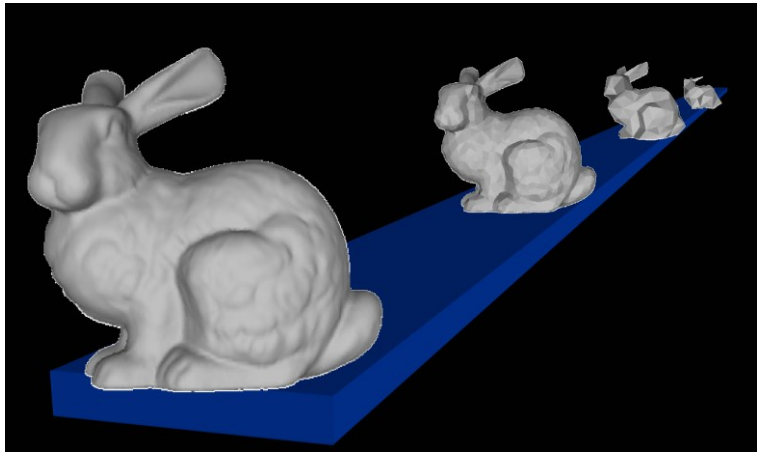
Depth

Final render

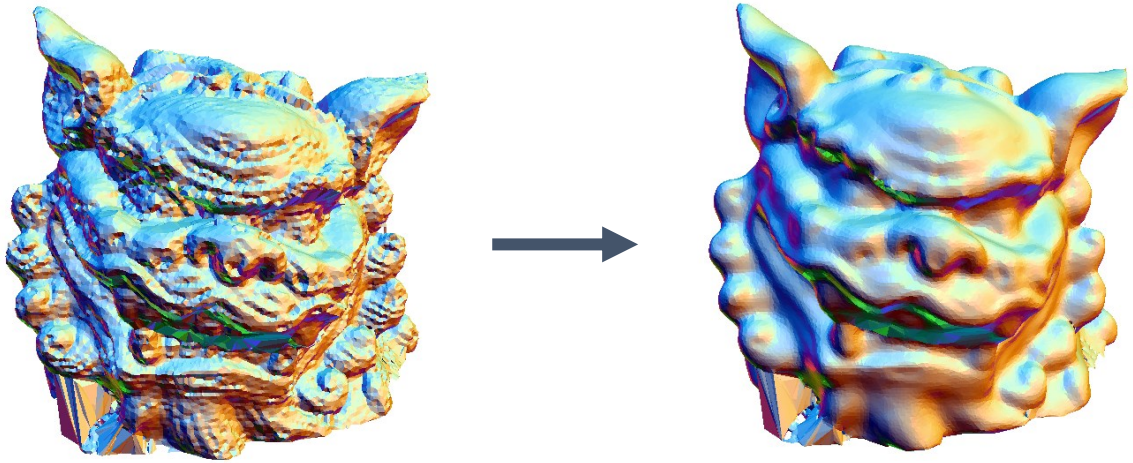


From Xi Wang, GAMES 104

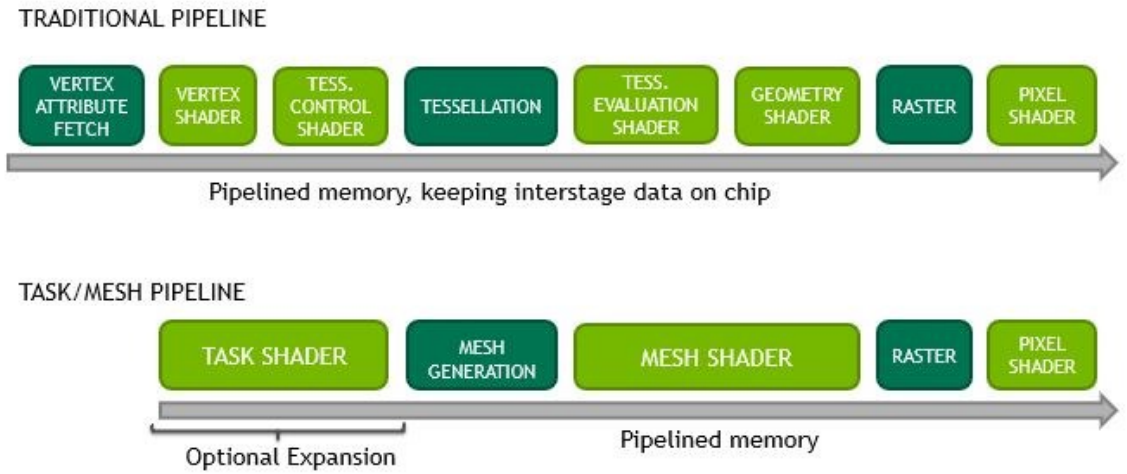
Mesh Shader



Level of Detail



Mesh Smoothing



- From vertex shader to mesh shader
- Allow for more flexible geometry processing

Image From Nvidia Blog

Dynamic Global Illumination

Image From Unreal Engine Docs

Thanks