

# Computer Graphics 2019

## 15. A ReVIEW

Hongxin Zhang

State Key Lab of CAD&CG, Zhejiang University

2020-01-07

# Main content

- Rendering ?
- Frame Buffer
- Display algorithm
  - basic rasterization methods
  - Real-time: Z-buffer => Ray casting ...
  - illumination and shading
  - Photo realistic: Ray tracing, Radiosity

# Main content

- Modeling
  - scene graph (transforms)
  - mesh representation
  - curves and surfaces
- Viewing

# A walkthrough of computer graphics

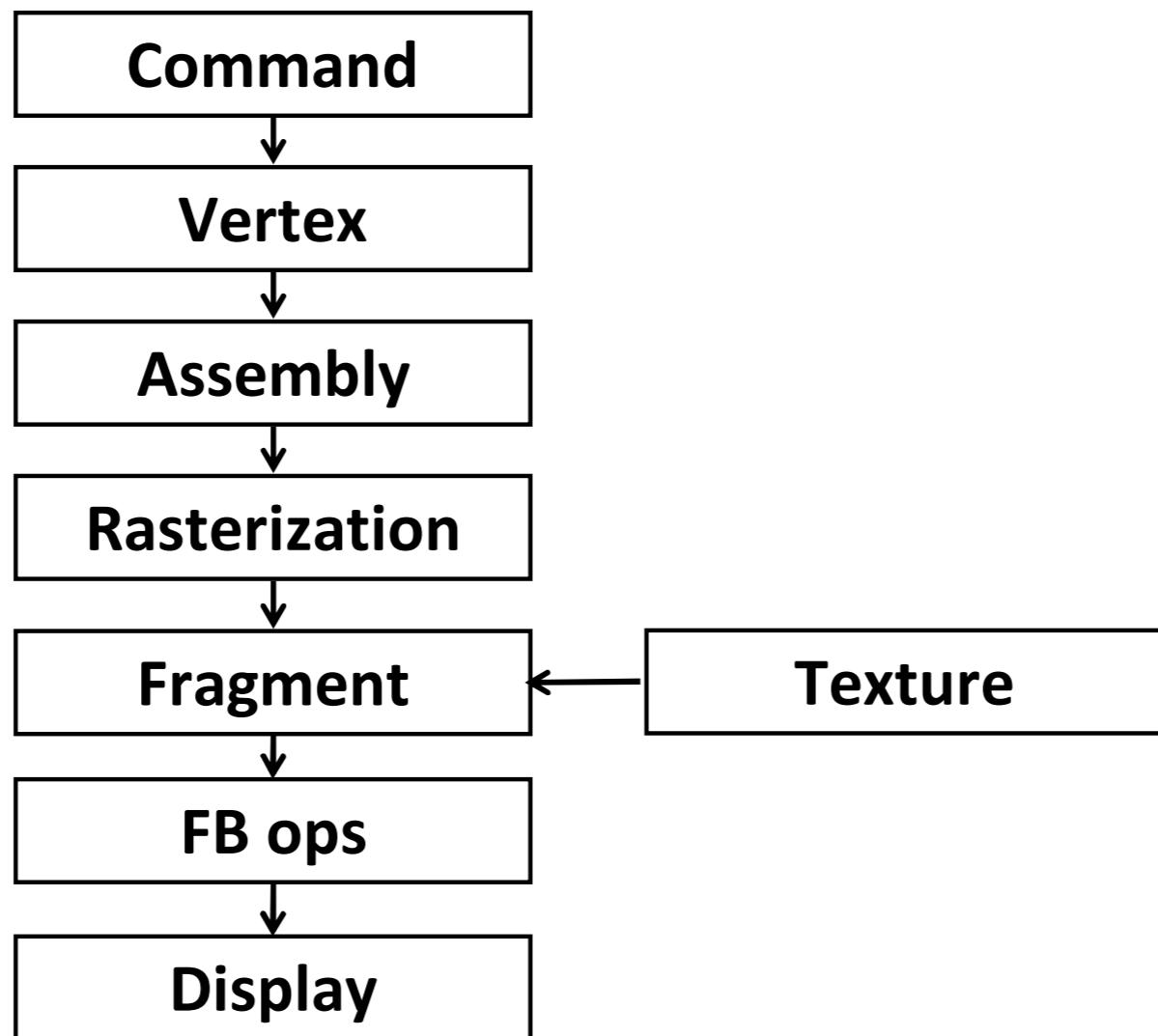
<http://www.pling.org.uk/cs/cgv.html>

# Computer Graphics and Visualization

Following pages are mainly from  
Pat Hanrahan's CG course

[http://graphics.stanford.edu/courses/cs148-10-fall/  
lectures/programmable.pdf](http://graphics.stanford.edu/courses/cs148-10-fall/lectures/programmable.pdf)

# a fixed graphics pipeline



# **Application**

---

**Simulation**

**Input event handlers**

**Modify data structures**

**Database traversal**

**Primitive generation**

**Graphics library utility functions (glu\*)**

# Command

---

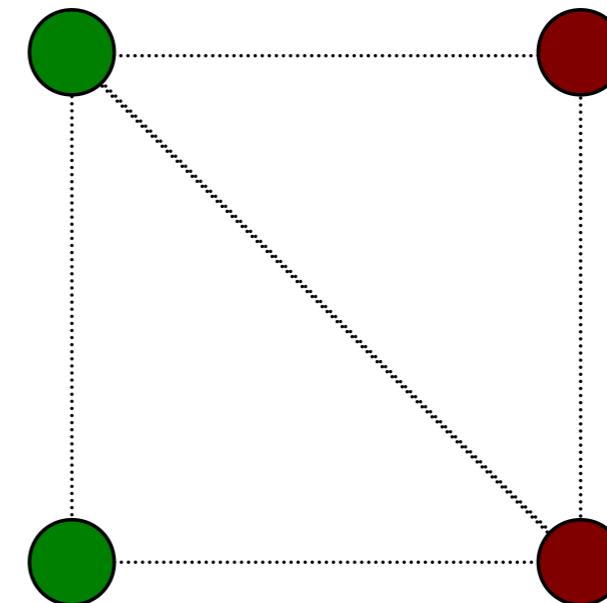
Command queue

Command interpretation

Unpack and perform format conversion

Maintain graphics state

```
glLoadIdentity( );
glMultMatrix( T );
glBegin( GL_TRIANGLE_STRIP );
glColor3f ( 0.0, 0.5, 0.0 );
glVertex3f( 0.0, 0.0, 0.0 );
glColor3f ( 0.5, 0.0, 0.0 );
glVertex3f( 1.0, 0.0, 0.0 );
glColor3f ( 0.0, 0.5, 0.0 );
glVertex3f( 0.0, 1.0, 0.0 );
glColor3f ( 0.5, 0.0, 0.0 );
glVertex3f( 1.0, 1.0, 0.0 );
...
glEnd( );
```



# **Vertex (per-vertex)**

---

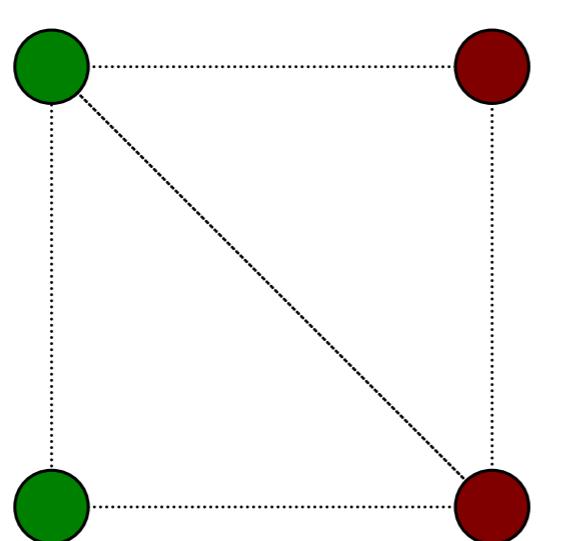
**Vertex transformation**

**Normal transformation**

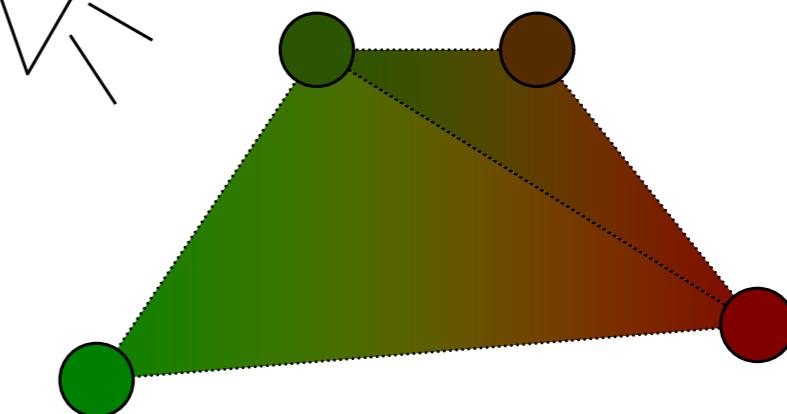
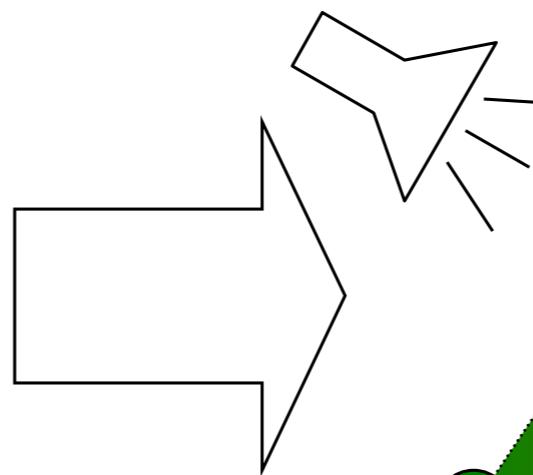
**Texture coordinate generation**

**Texture coordinate transformation**

**Lighting (light sources and surface reflection)**



**Object-space triangles**



**Screen-space lit triangles**

# Primitive Assembly

---

Combine transformed/lit vertices into primitives

- 1 vert -> point
- 2 verts -> line
- 3 verts -> triangle

Clipping

Perspective projection

Transform to window coordinates (viewport)

Determine orientation (CW/CCW)

Back-face cull

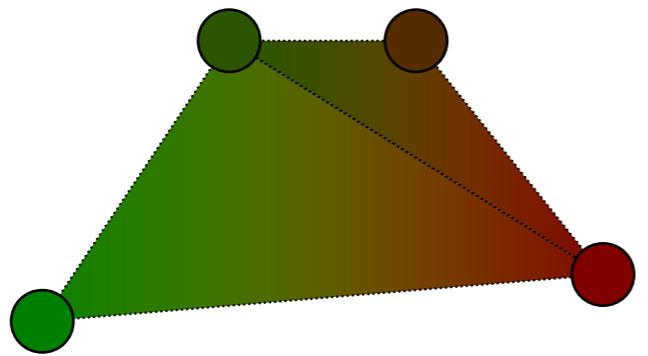
# Rasterization

---

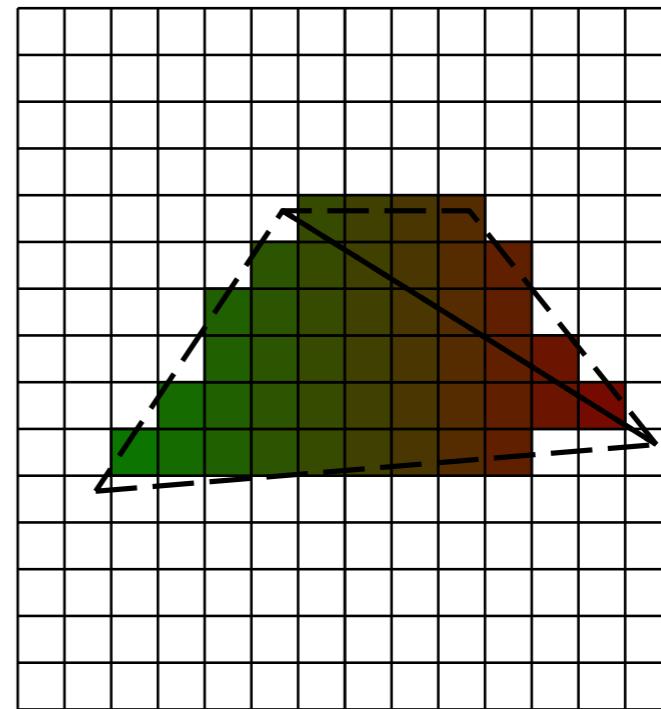
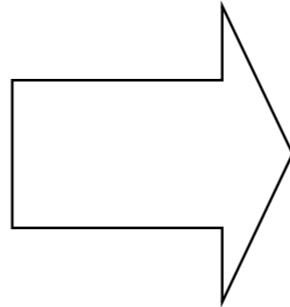
**Setup (per-triangle)**

**Sampling (triangle = {fragments})**

**Interpolation (interpolate colors and coordinates)**



**Screen-space triangles**



**Fragments**

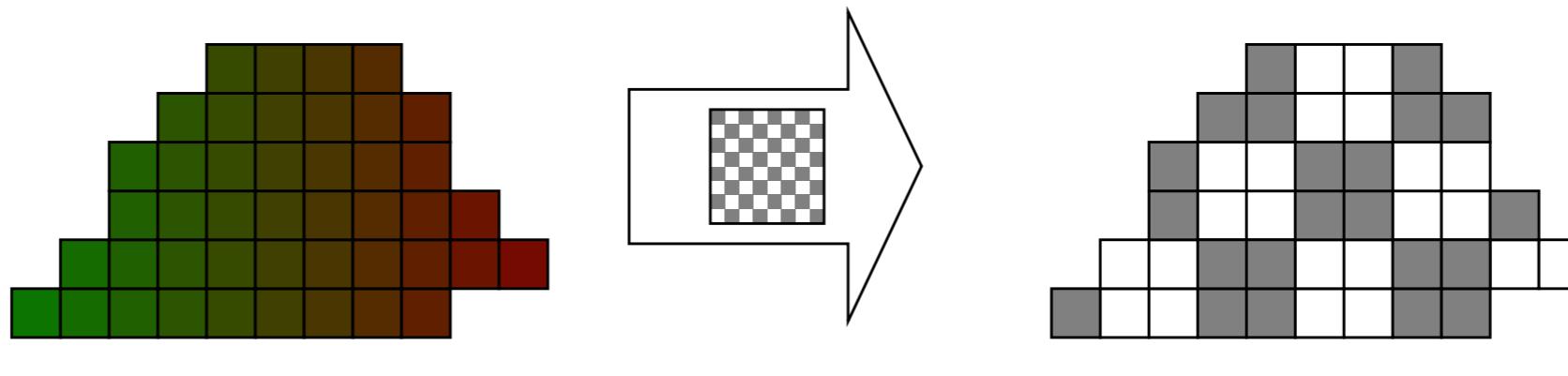
# Texture

---

*Textures are arrays indexed by floats (Sampler)*

Texture address calculation

Texture interpolation and filtering



Fragments

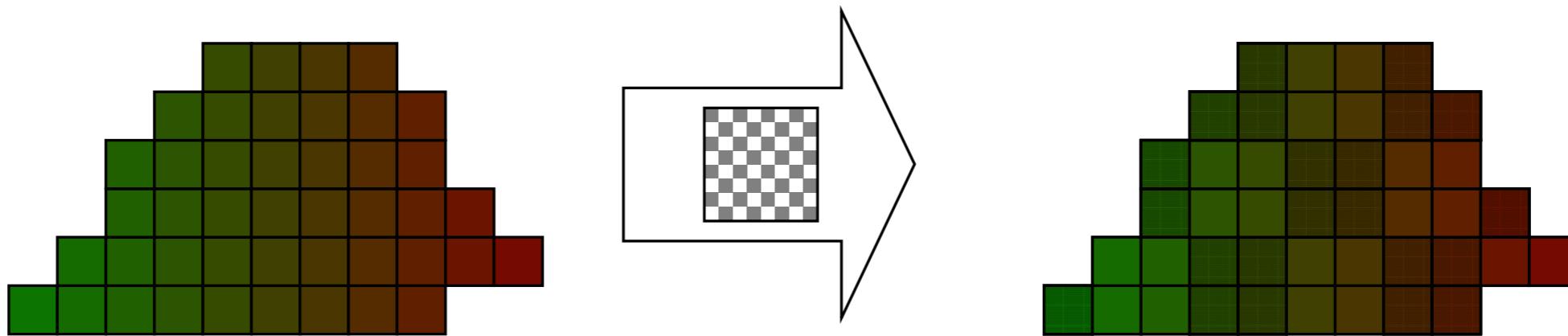
Texture Fragments

# Fragment

---

Combine texture sampler outputs

Per-fragment shading



Fragments

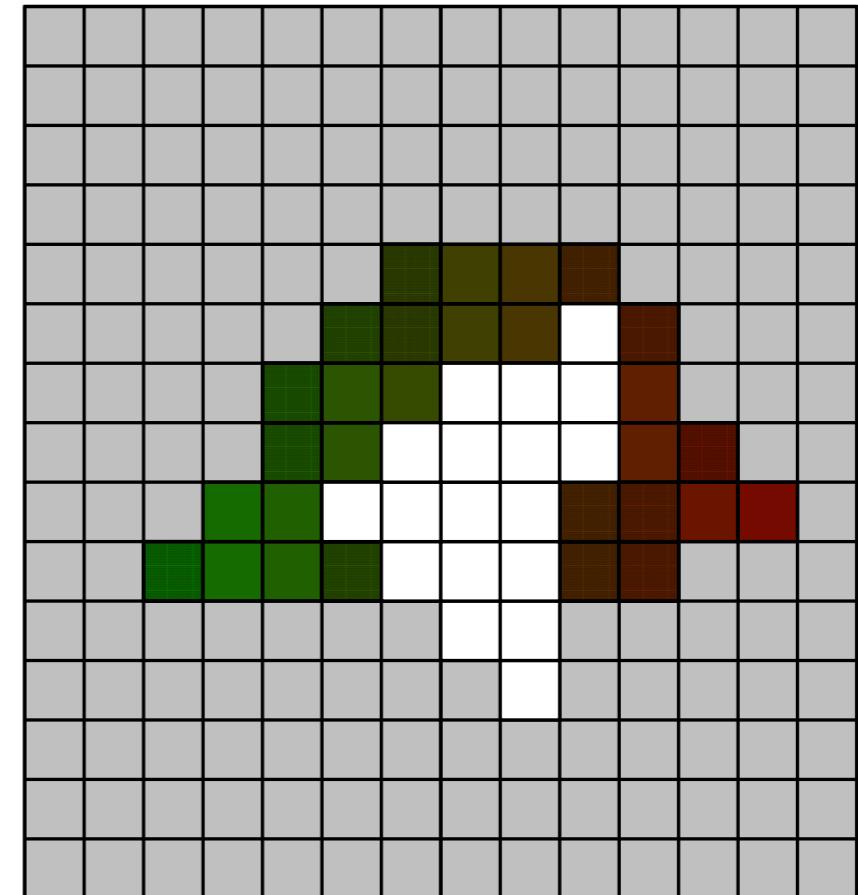
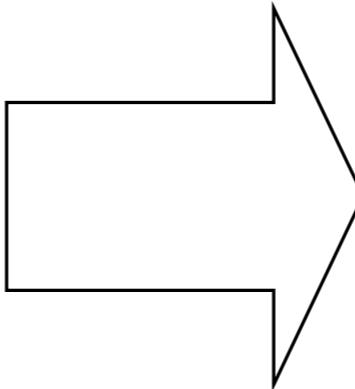
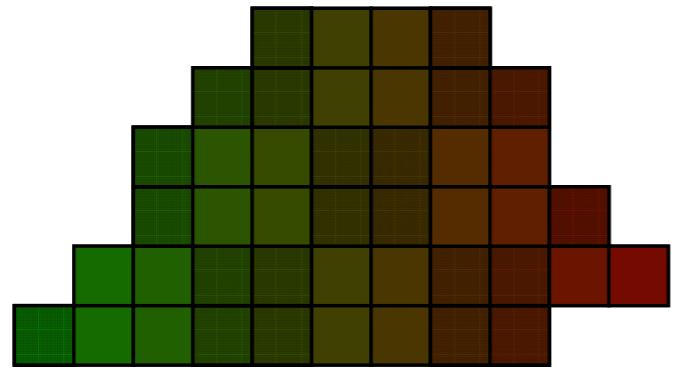
Textured Fragments

# Framebuffer Operations

---

**Owner, scissor, depth, alpha and stencil tests**

**Blending or compositing**



**Textured Fragments**

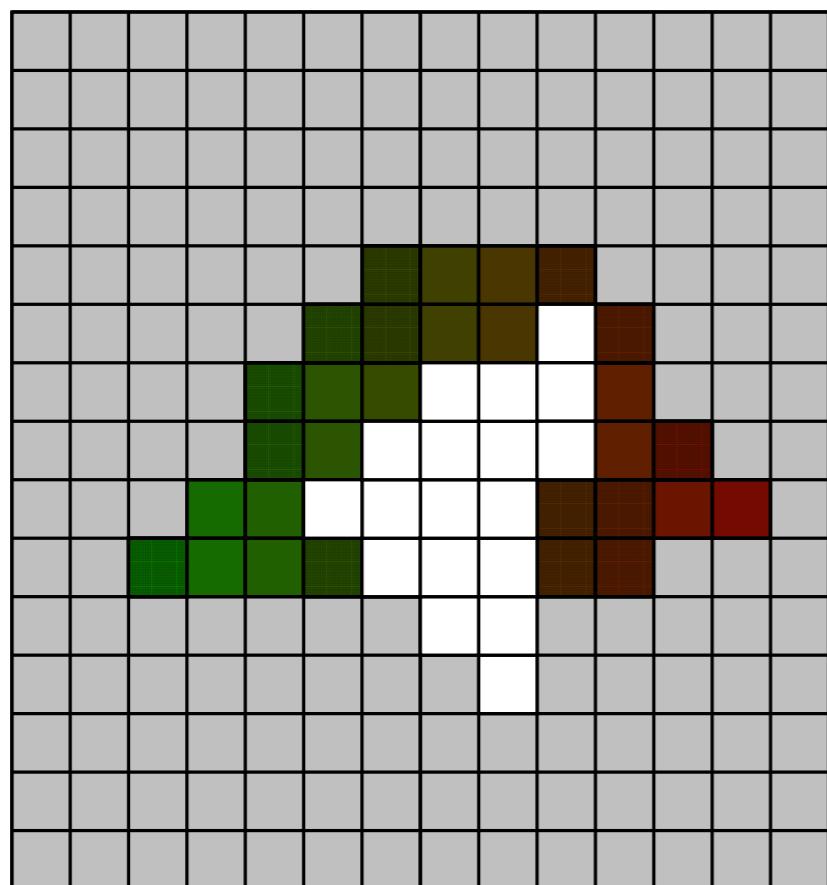
**Framebuffer Pixels**

# Display

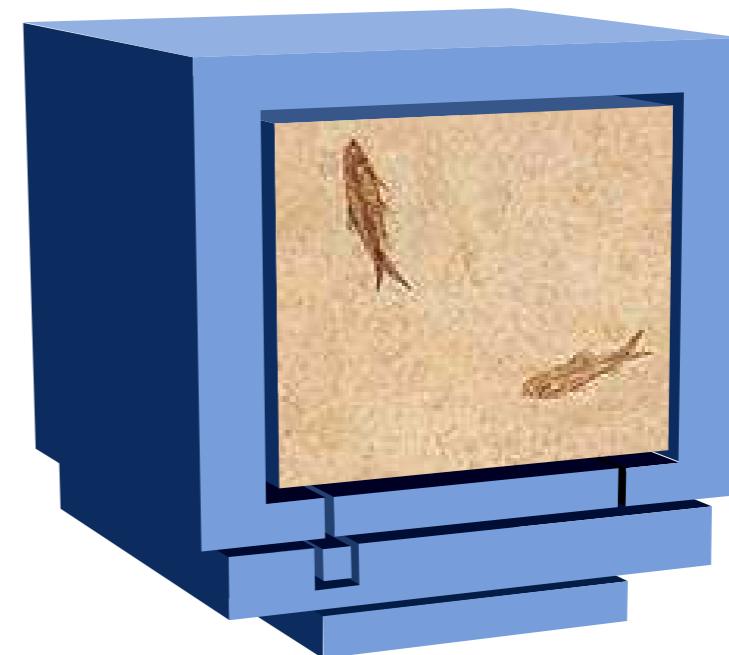
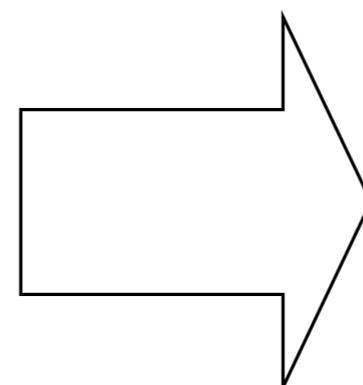
---

**Gamma correction**

**Analog to digital conversion**

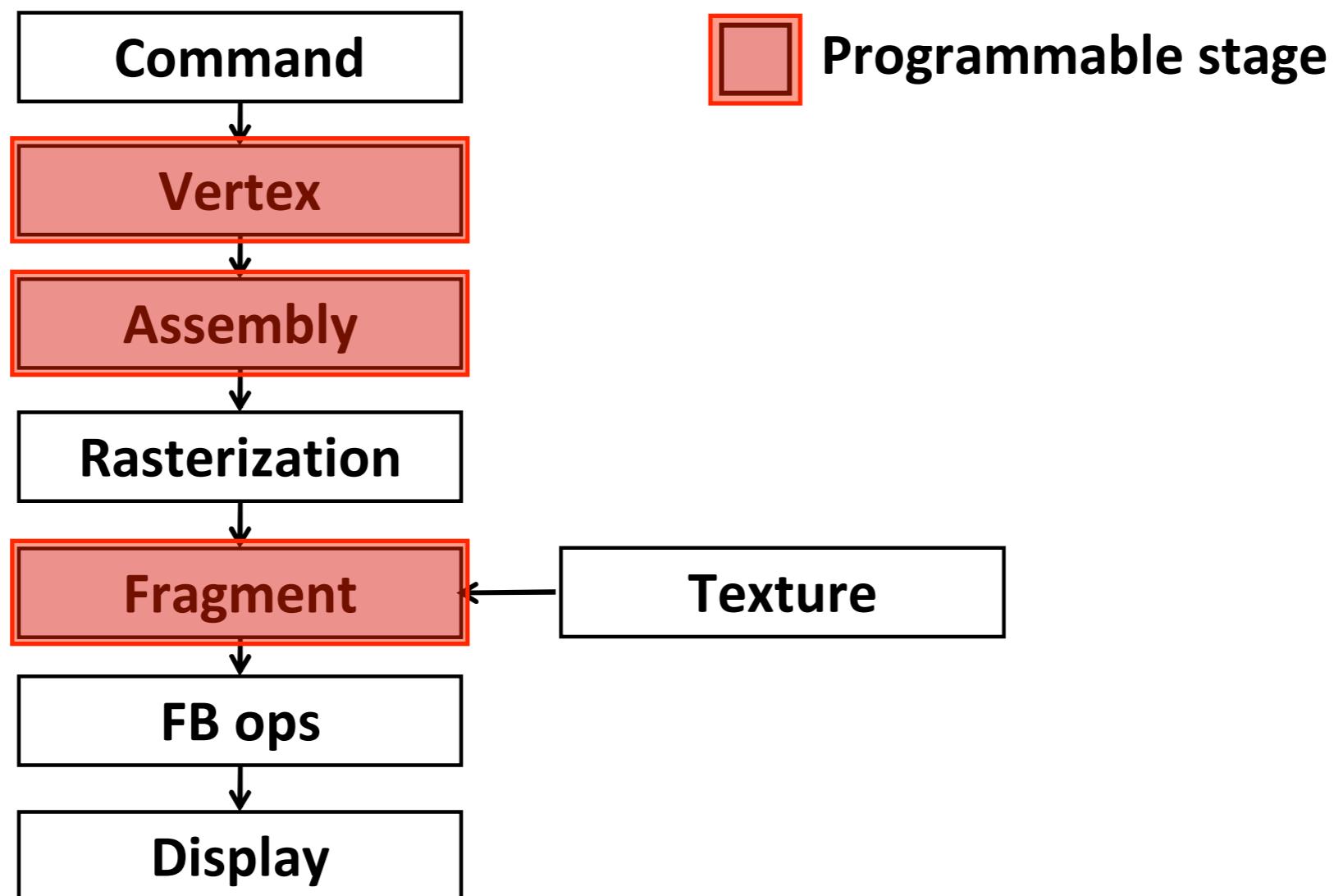


**Framebuffer Pixels**

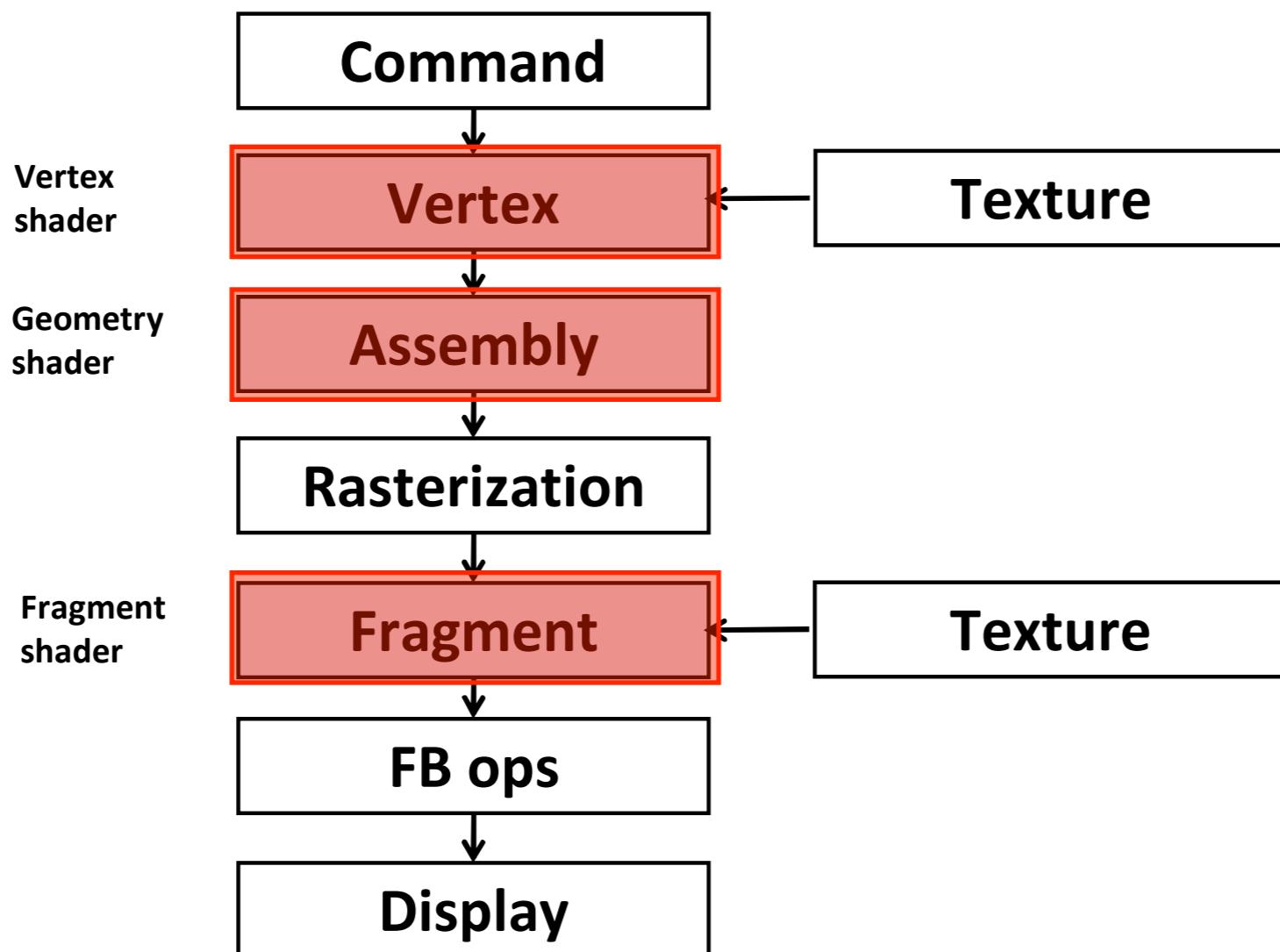


**Light**

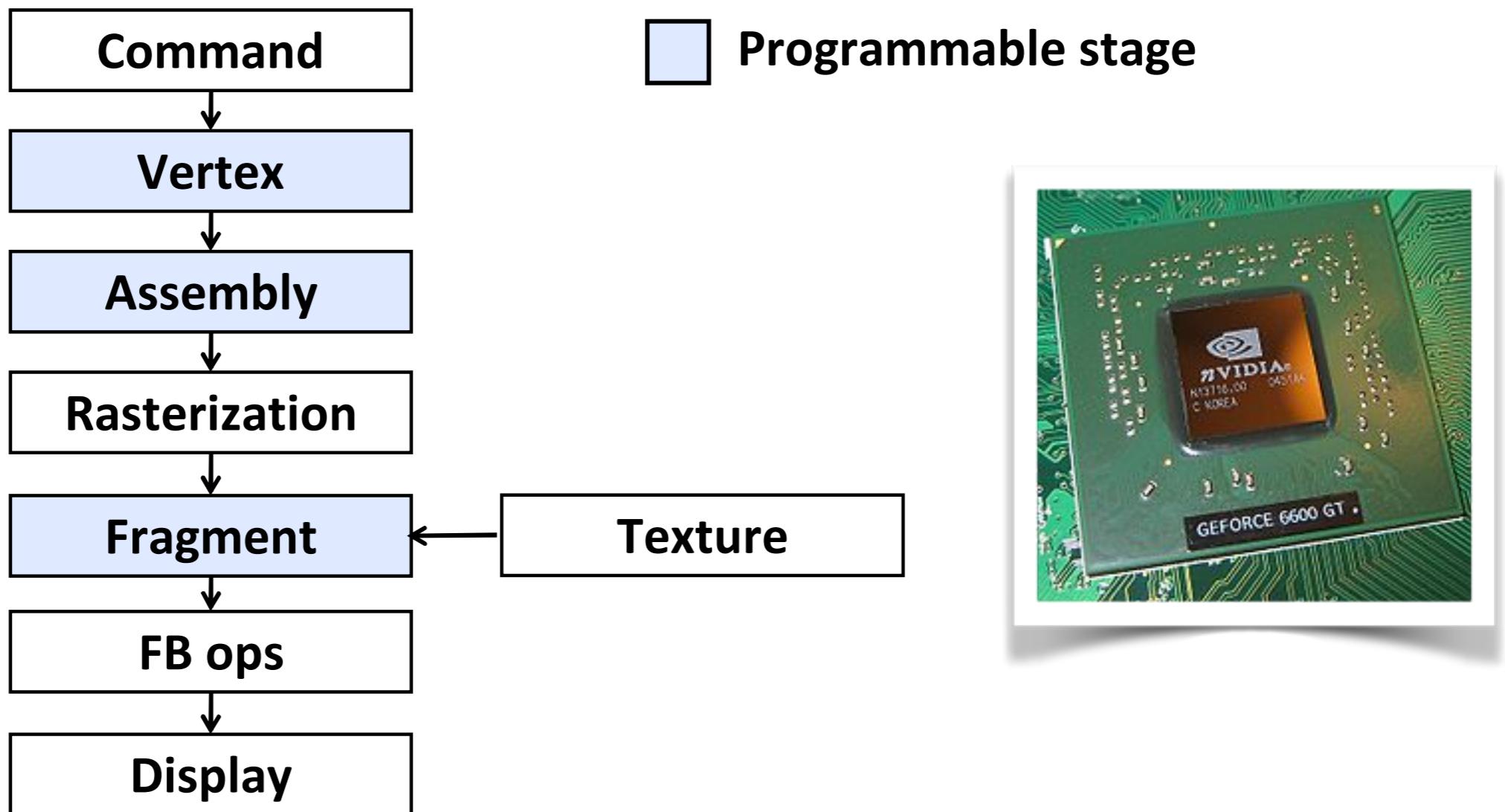
# Programmable graphics pipeline



# Programmable graphics pipeline

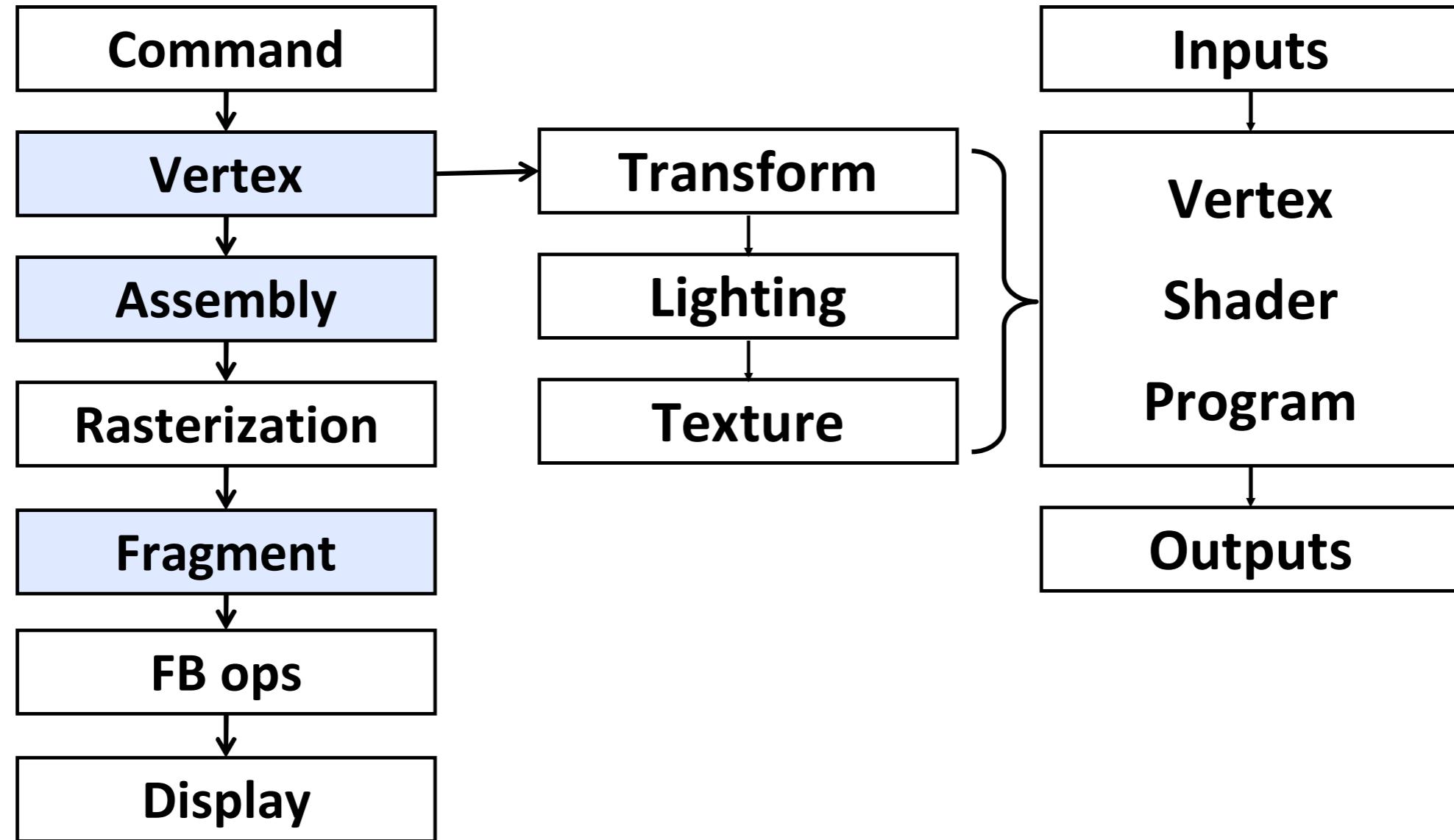


# Programmable graphics pipeline



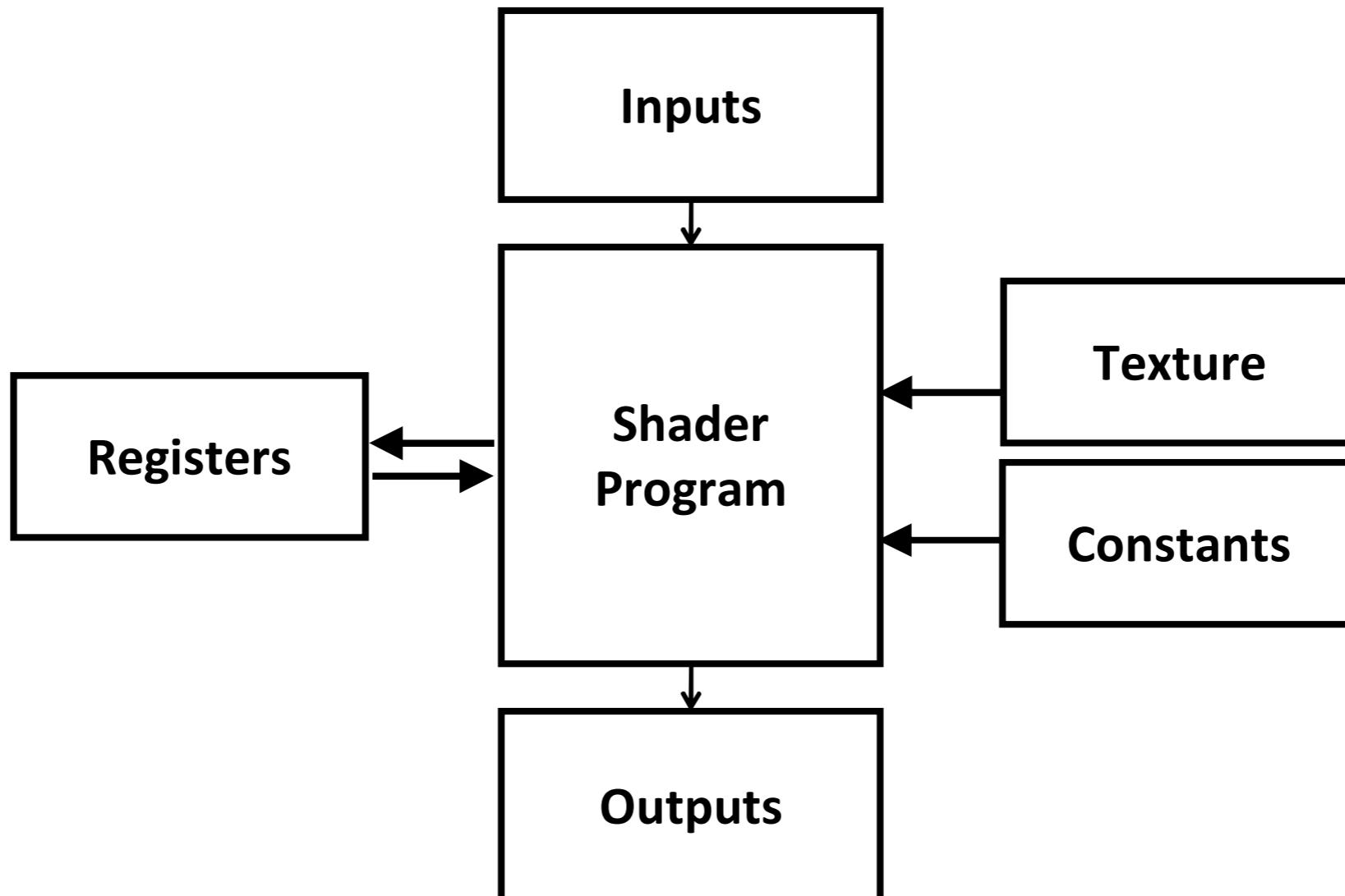
# Programmable Graphics Pipeline

---



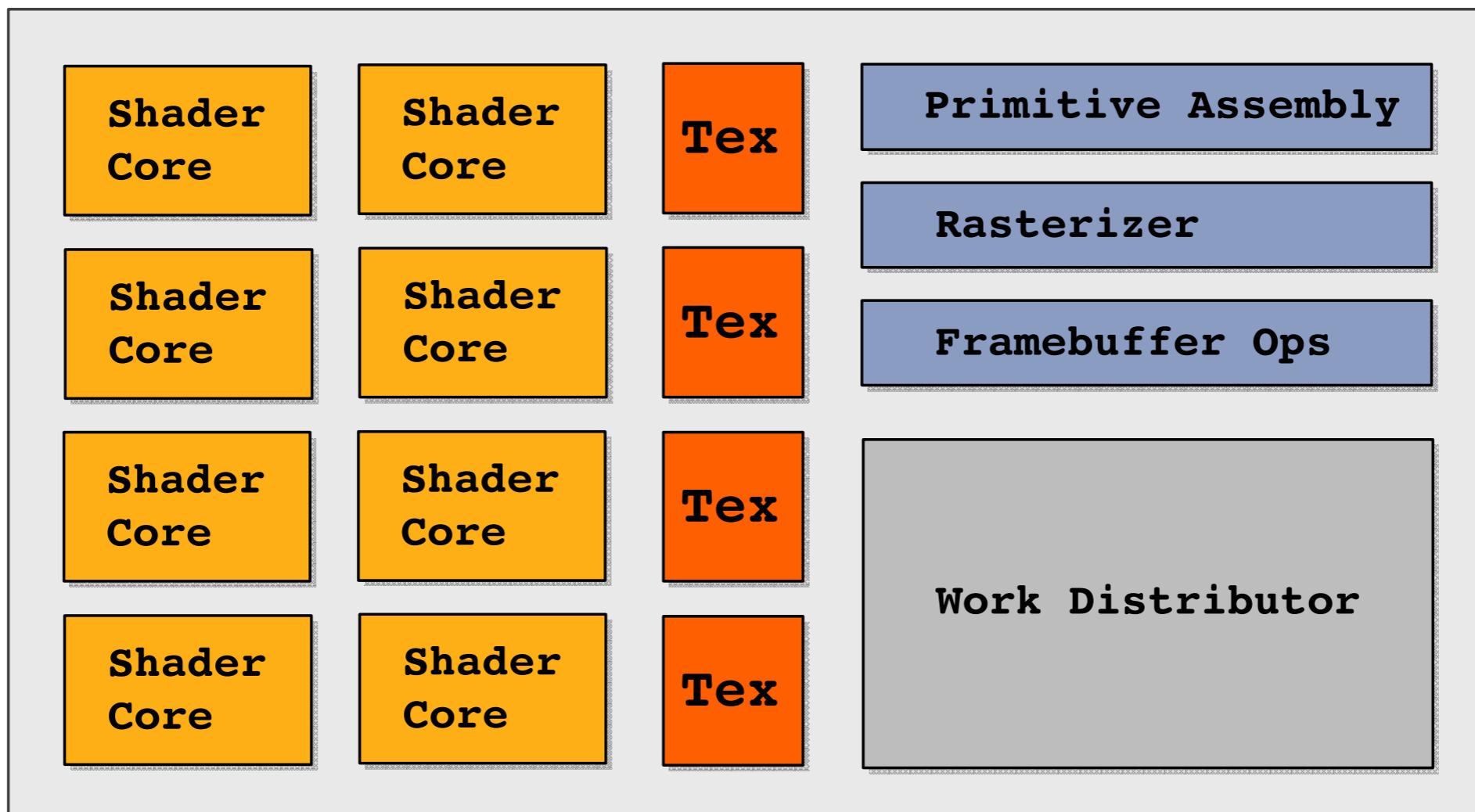
# Shader Program Architecture

---



# What's in a GPU?

---



# What's in a GPU?



NVIDIA GF100  
(GeForce GTX 480)



AMD Cypress  
(Radeon HD 5870)

# **GLSL**

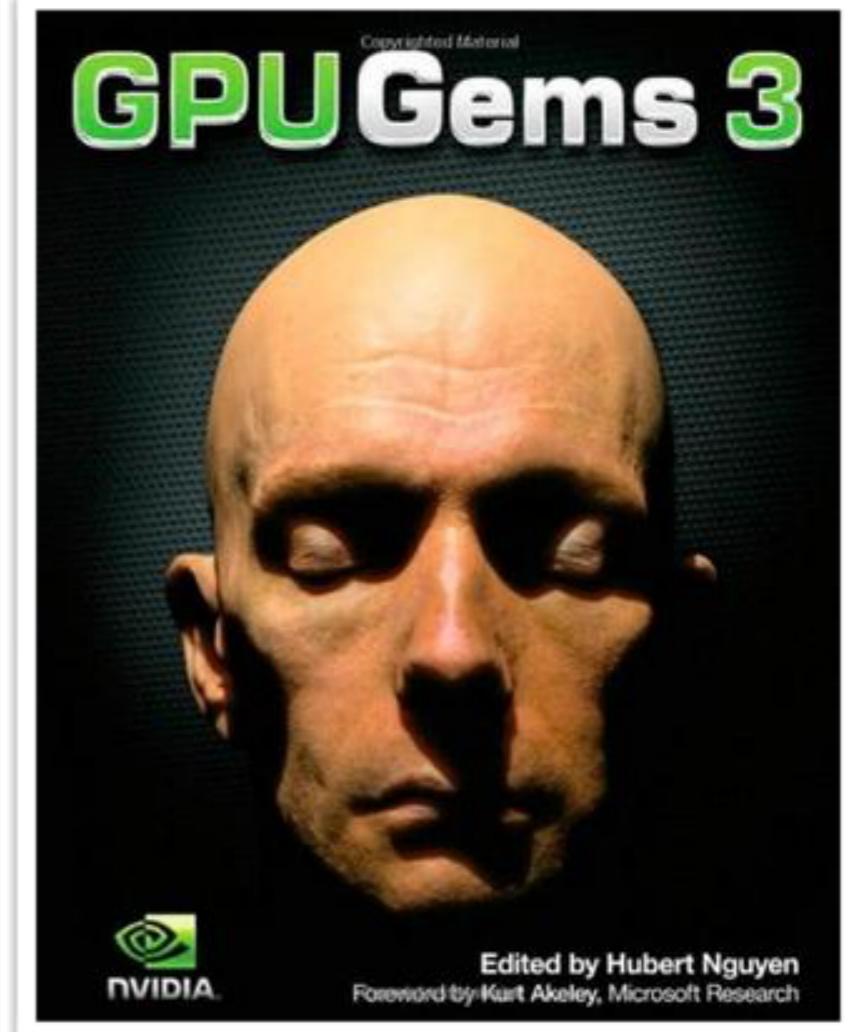
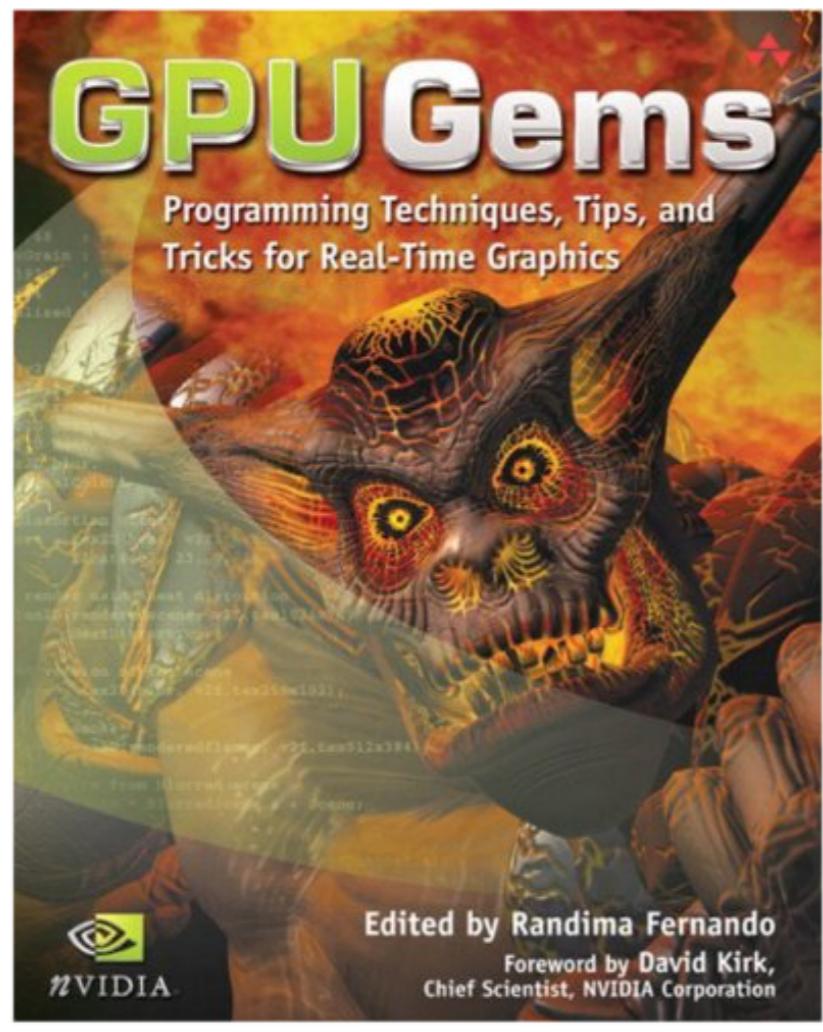
# Simple Vertex and Fragment Shaders

---

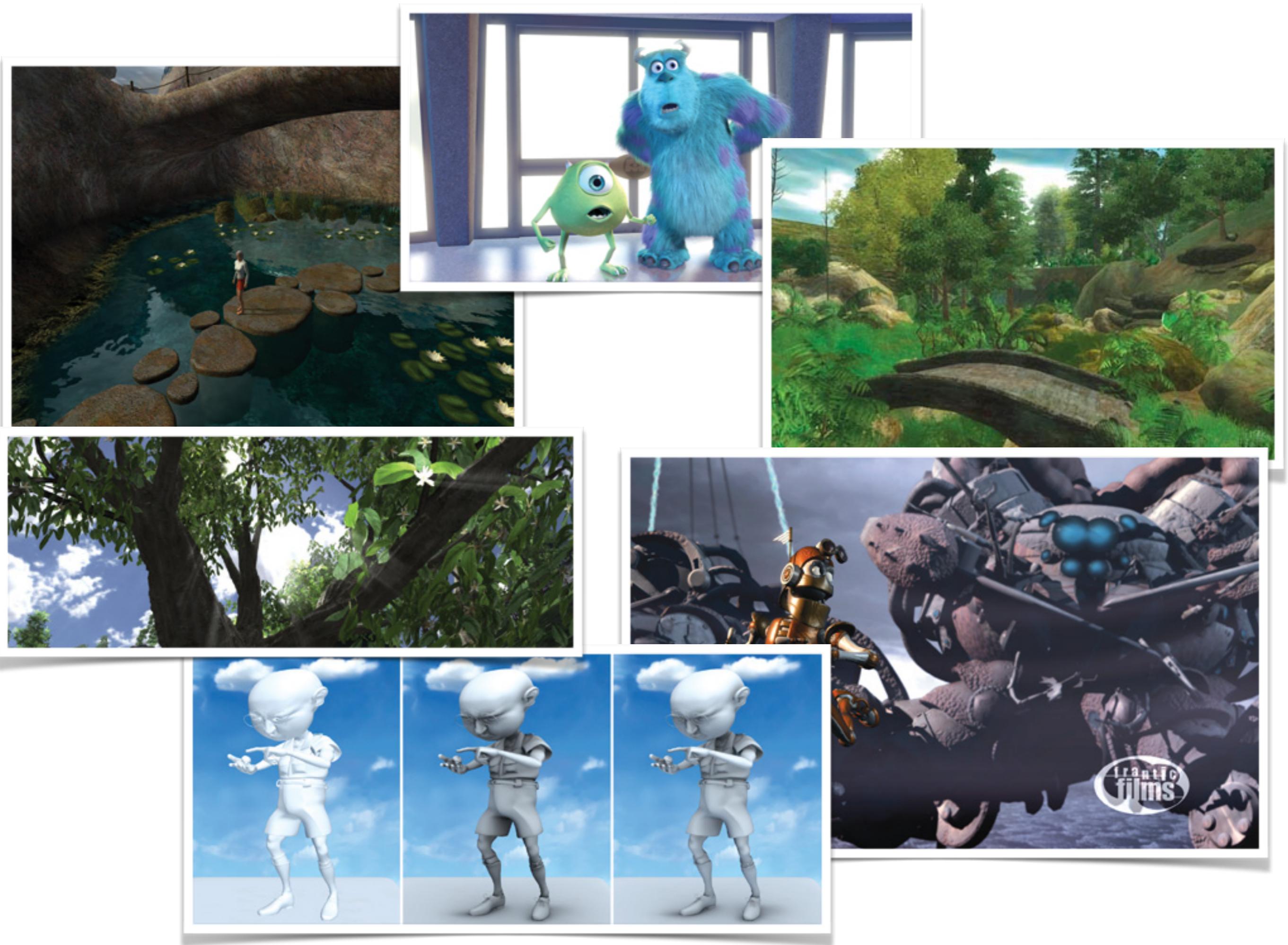
```
// simple.vert
void main()
{
    gl_Position =
        gl_ModelViewMatrix *
            gl_ProjectionMatrix * gl_Vertex;
    gl_Normal = gl_NormalMatrix * gl_Normal;
    gl_FrontColor = gl_Color;
    gl_BackColor = gl_Color;
}
// simple.frag
void main()
{
    gl_FragColor = gl_Color
}
```

# DEMO: WebGL

[http://learningwebgl.com/blog/?page\\_id=1217](http://learningwebgl.com/blog/?page_id=1217)



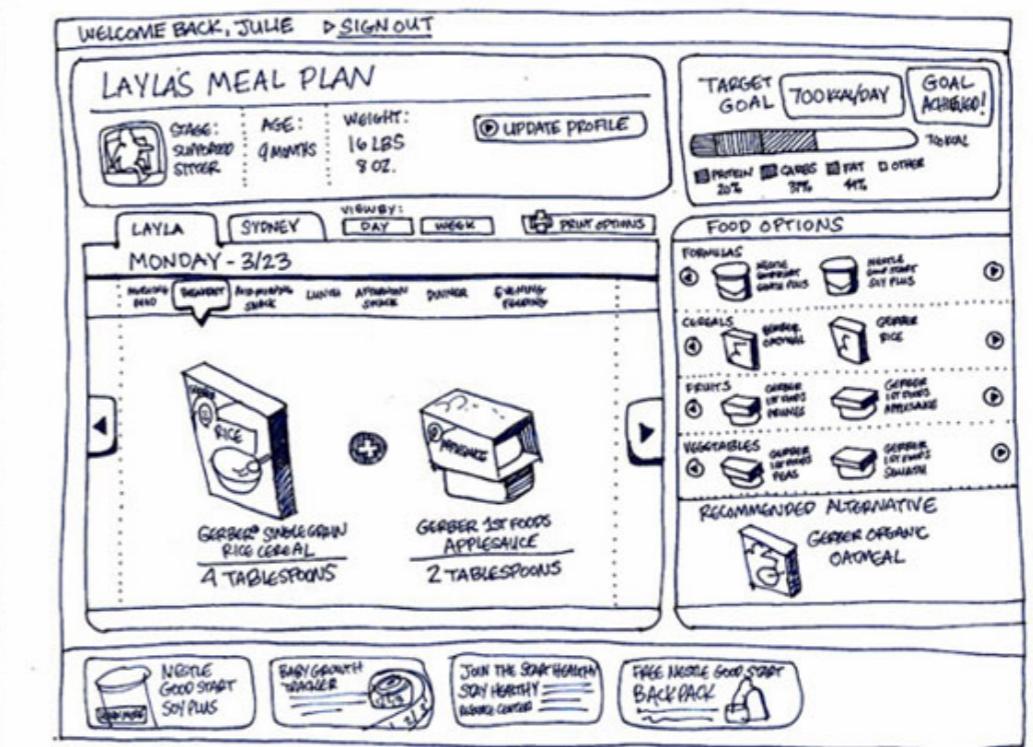
[https://developer.nvidia.com/gpugems/GPUGems/gpugems\\_pref01.html](https://developer.nvidia.com/gpugems/GPUGems/gpugems_pref01.html)



**Things does not covered  
yet in the course**

# User Interface

- pick
- feedback
- special ui devices

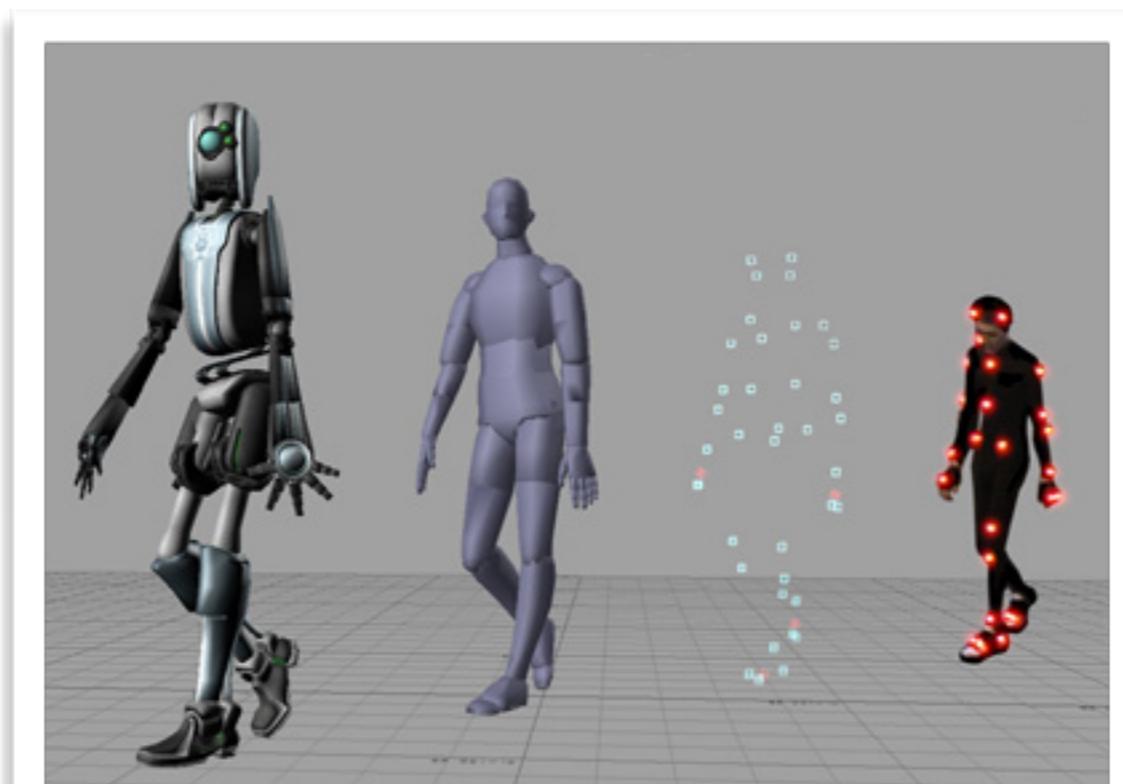


# Text

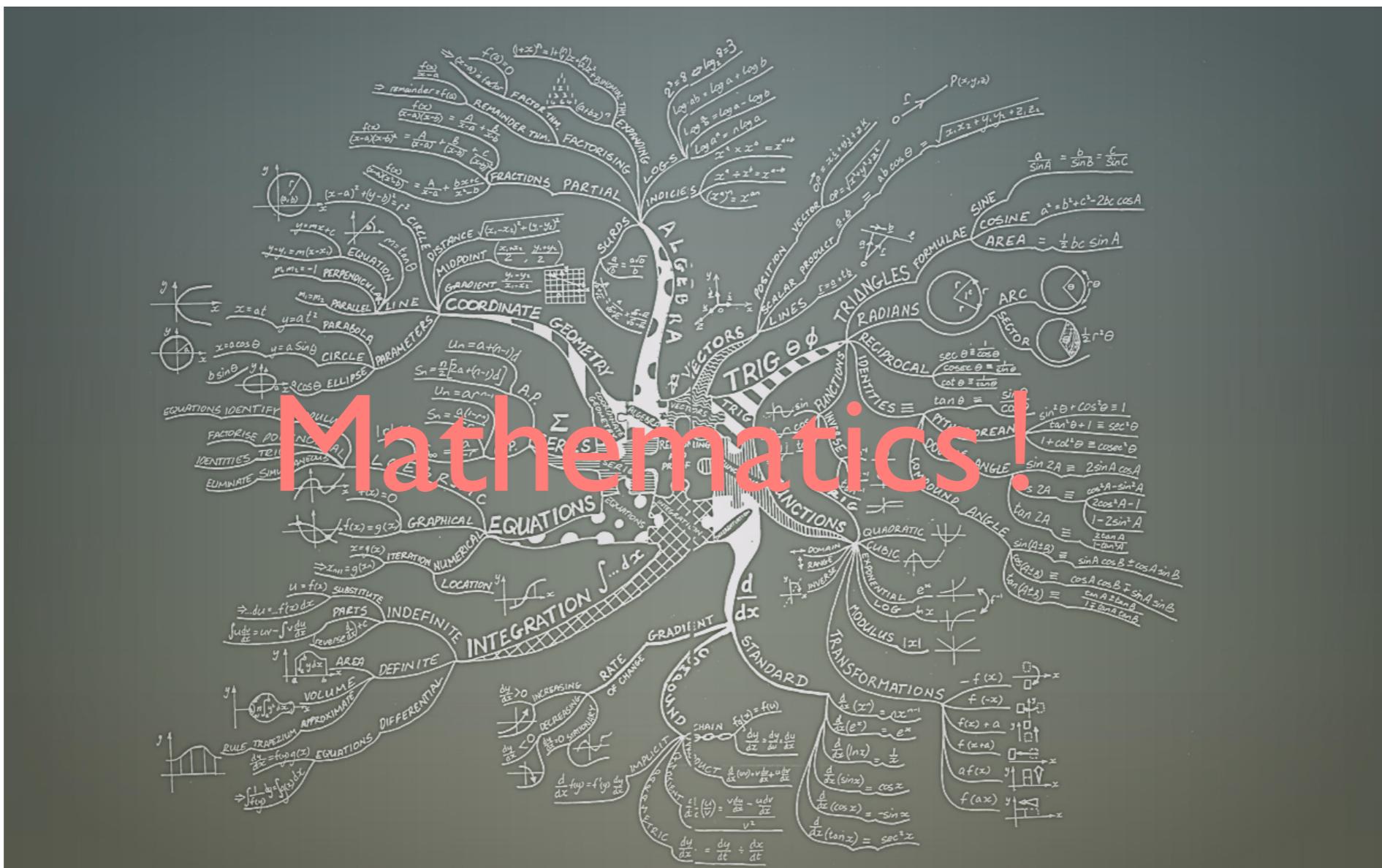
- The beautiful text once was a hot topic in computer graphics
  - font representation
  - antialias
  - encoding
  - ...

Peter met a duck  
“What kind of bird are you if you can’t fly?” said he.  
Duck replied,  
“What kind of bird are you  
if you can’t swim?”  
and laughing, dove into the pond

# Computer Animation



# One more thing ...





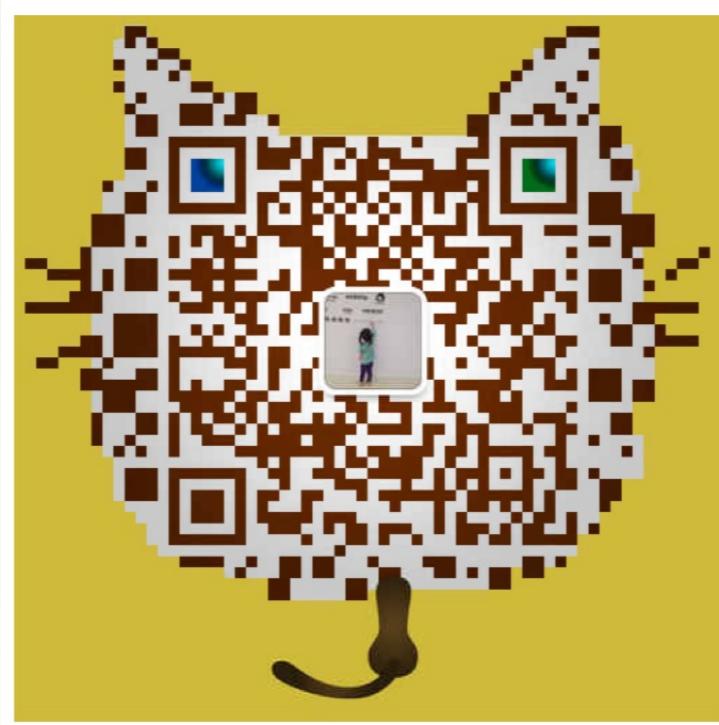
天真，热爱，实践，批判

This is not an end but a start

# 联系

- ❖ 电话：13958011790
- ❖ 主页：<http://www.cad.zju.edu.cn/home/zhx/>
- ❖ 邮件：[zhx@cad.zju.edu.cn](mailto:zhx@cad.zju.edu.cn)
- ❖ 地址：浙江大学紫金港校区蒙民伟楼517室

@浙大张宏鑫



微博

微信



微信公众号