



Lite repetition

Vi kan väl...



Transformationer som 4x4-matriser

Rotation

$$\begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\theta & -\sin\theta & 0 \\ 0 & \sin\theta & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos\theta & 0 & \sin\theta & 0 \\ 0 & 1 & 0 & 0 \\ -\sin\theta & 0 & \cos\theta & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

$$\begin{bmatrix} \cos\theta & -\sin\theta & 0 & 0 \\ \sin\theta & \cos\theta & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Translation

$$\begin{bmatrix} 1 & 0 & 0 & t_x \\ 0 & 1 & 0 & t_y \\ 0 & 0 & 1 & t_z \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Skalning

$$\begin{bmatrix} s_x & 0 & 0 & 0 \\ 0 & s_y & 0 & 0 \\ 0 & 0 & s_z & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

Spegling

$$\begin{bmatrix} -1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$

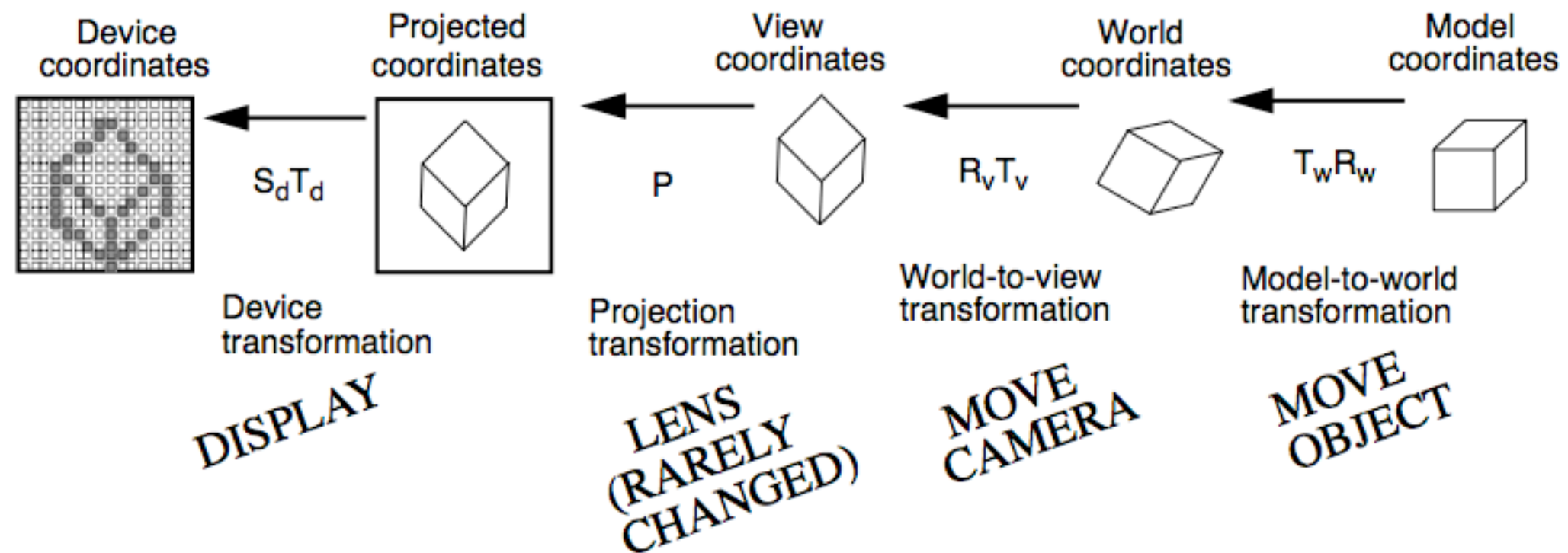
Skevning

$$\begin{bmatrix} 1 & 0 & a & 0 \\ 0 & 1 & b & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$$



Transformation pipeline

Model coordinates
World coordinates
View coordinates
Projected coordinates
Device coordinates





Mer som ni kan som ett rinnande vatten

Ljussättning med
Phongs ljusmodell

Anti-aliasing

Phong shading

Texturering

Brusgenerering

Skyboxes

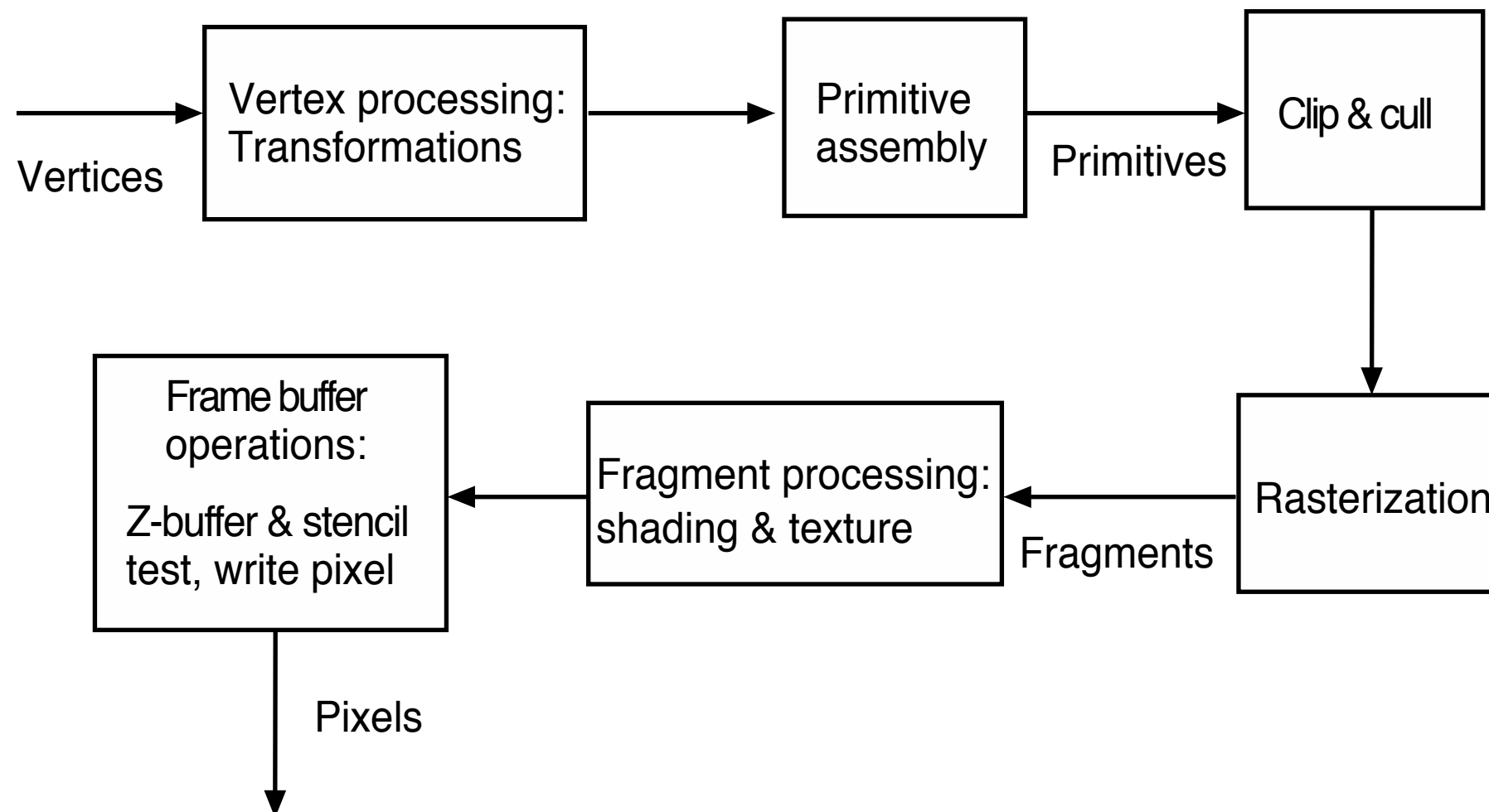
VSD

Enkel kollisions-
detektering

Transparens

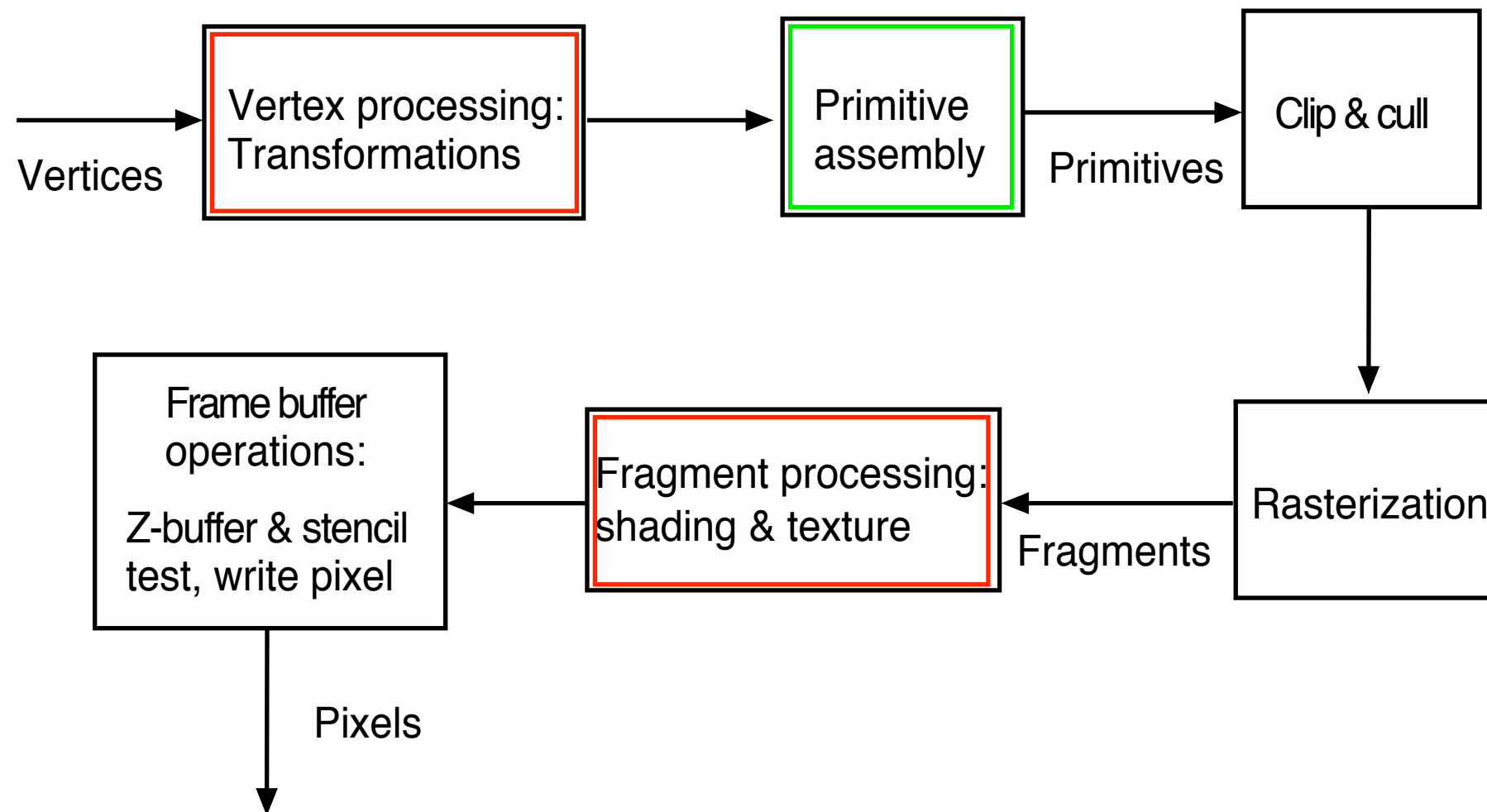


OpenGL-pipelinen





Steg som kan eller måste ha shaders

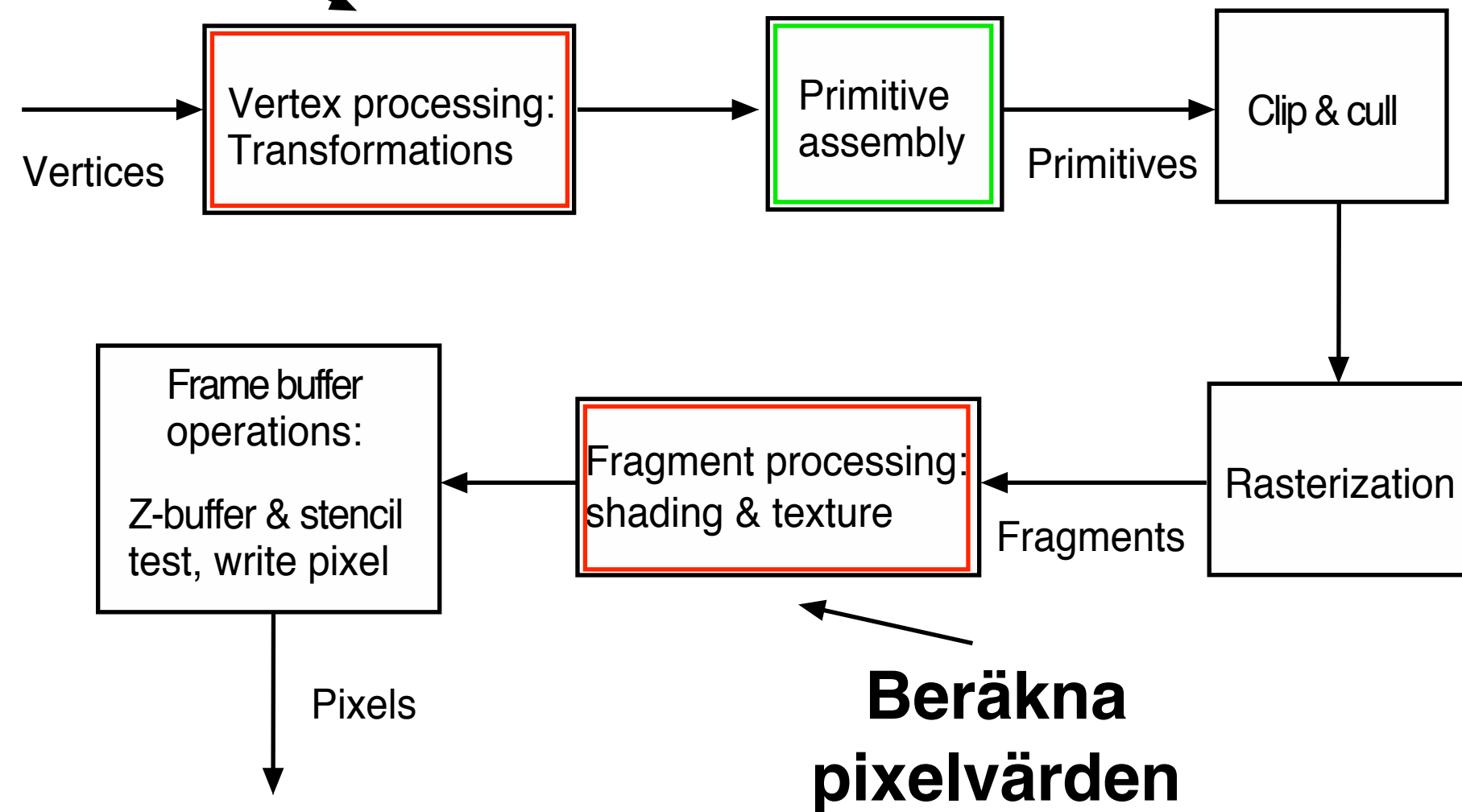




Information Coding / Computer Graphics, ISY, LiTH

Applicera transformationer

Modifiera geometri





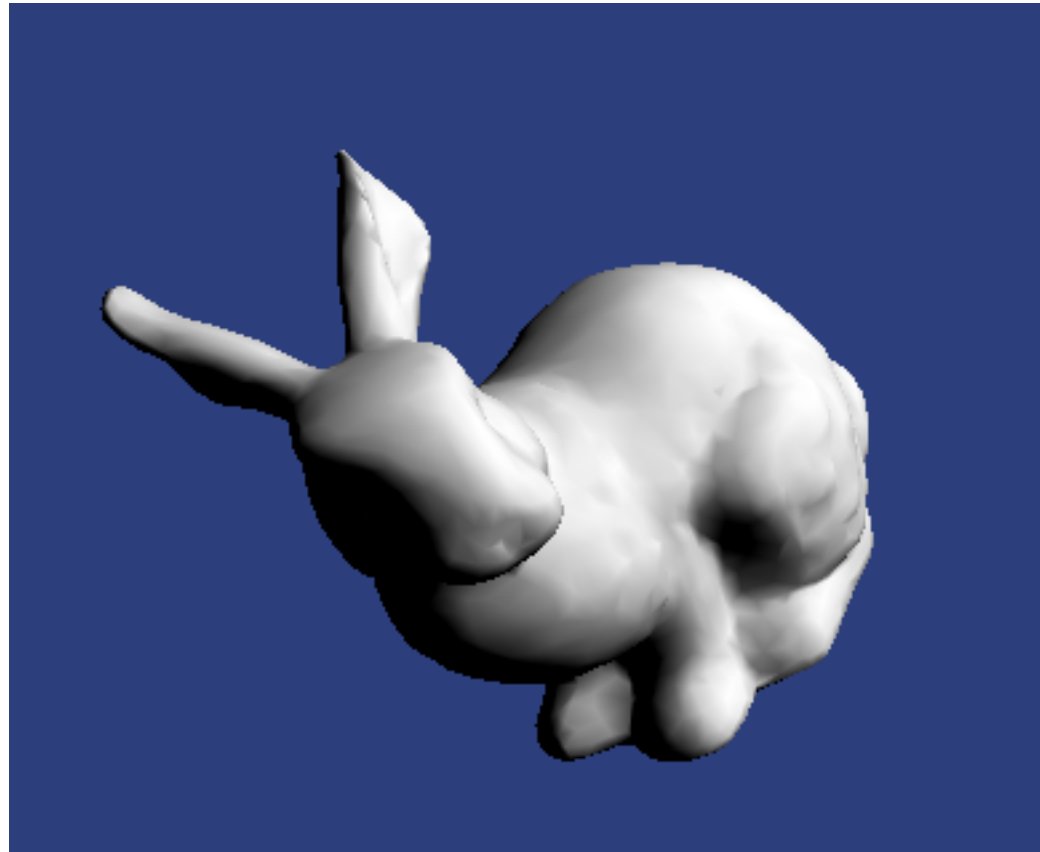
Information Coding / Computer Graphics, ISY, LiTH

Shader = beräkningskärna i OpenGL-pipelinen

- Körs när man ritar geometri
 - Körs på GPU
- Körs i massiv parallellprocessor
- Laddas, kompileras och aktiveras av huvudprogrammet (på CPU)
 - Data skickas in från huvudprogrammet



Information Coding / Computer Graphics, ISY, LiTH



Gouraud shader - vertex shader

```
#version 150
in  vec3 inPosition;
in  vec3 inNormal;
out vec3 exColor;

void main(void)
{
    const vec3 light = vec3(0.58, 0.58, 0.58);
    float shade;
    shade = dot(normalize(inNormal), light);
    shade = clamp(shade, 0, 1);
    exColor = vec3(shade);
    gl_Position = vec4(inPosition, 1.0);
}
```

Gouraud shader - fragment shader

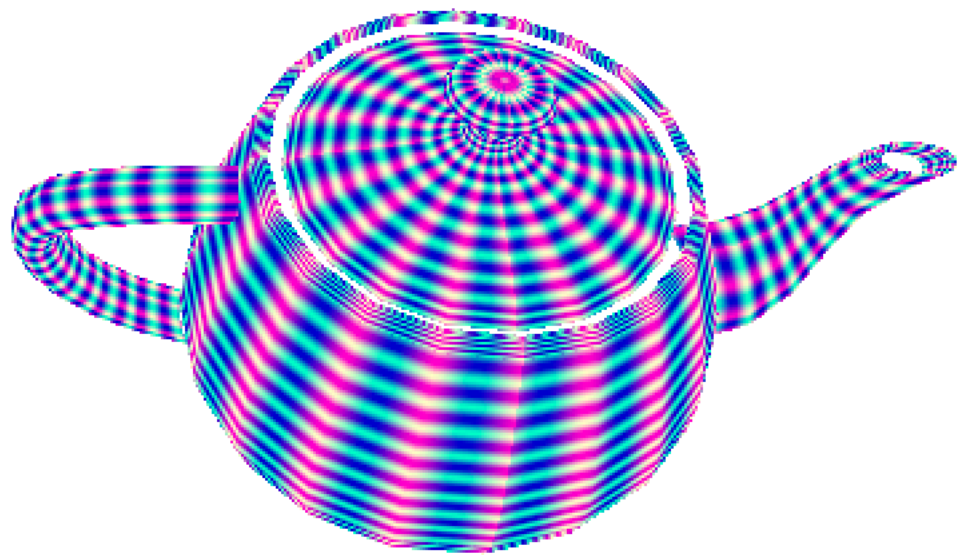
```
#version 150
in  vec3 exColor;
out vec4 outColor;
void main(void)
{
    outColor = vec4(exColor,1.0);
}
```



Information Coding / Computer Graphics, ISY, LiTH

Procedurrell textur

Ingemar's psychedelic teapot 2



```
#version 150
```

```
in vec3 inPosition;  
in vec2 inTexCoord;  
out vec2 texCoord;
```

```
uniform mat4 mdlMatrix;  
uniform mat4 camMatrix;  
uniform mat4 projMatrix;
```

```
void main(void)
```

```
{  
    texCoord = inTexCoord;  
  
    gl_Position = projMatrix * camMatrix * mdlMatrix * vec4(inPosition, 1.0);  
}
```

```
#version 150
```

```
out vec4 outColor;  
in vec2 texCoord;  
uniform float t;
```

```
void main(void)
```

```
{  
    float a = sin(texCoord.s * 30.0 + t)/2.0 + 0.5;  
    float b = sin(texCoord.t * 30.0 * (1.0+sin(t/4.0)))/2.0 + 0.5;  
    outColor = vec4(a, b, 0.8, 1.0); // inColor;  
}
```



Vanliga shadereffekter

Multitexturering

Deformationer

Procedurella texturer

Ljussättning

Men vi kommer att gå vidare med

Bumpmappning

Projektiva texturer

Skinning

Rendering till textur,
multipassrendering

Skuggrendering



Information Coding / Computer Graphics, ISY, LiTH

Har du några luckor?

Det är det labb 0 är till för att rätta till!

Helt frivillig labb för att repetera,
främst shaderprogrammering.

Frågor?



Information Coding / Computer Graphics, ISY, LiTH

Onlineresurser:

PDFs laddas upp till:

computer-graphics.se/TSBK03/files/PDF21

Filminspelningar kommer upp på annan plats allt eftersom de blir klara.



Onlinelabbar:

Thinlinc till thinlinc.edu.liu.se

```
ssh asgard1-301.ad.liu.se -X  
ssh asgard1-305.ad.liu.se -X  
ssh asgard1-306.ad.liu.se -X  
ssh asgard1-307.ad.liu.se -X  
ssh asgard1-308.ad.liu.se -X  
ssh asgard1-309.ad.liu.se -X  
ssh asgard1-310.ad.liu.se -X  
ssh asgard1-312.ad.liu.se -X
```