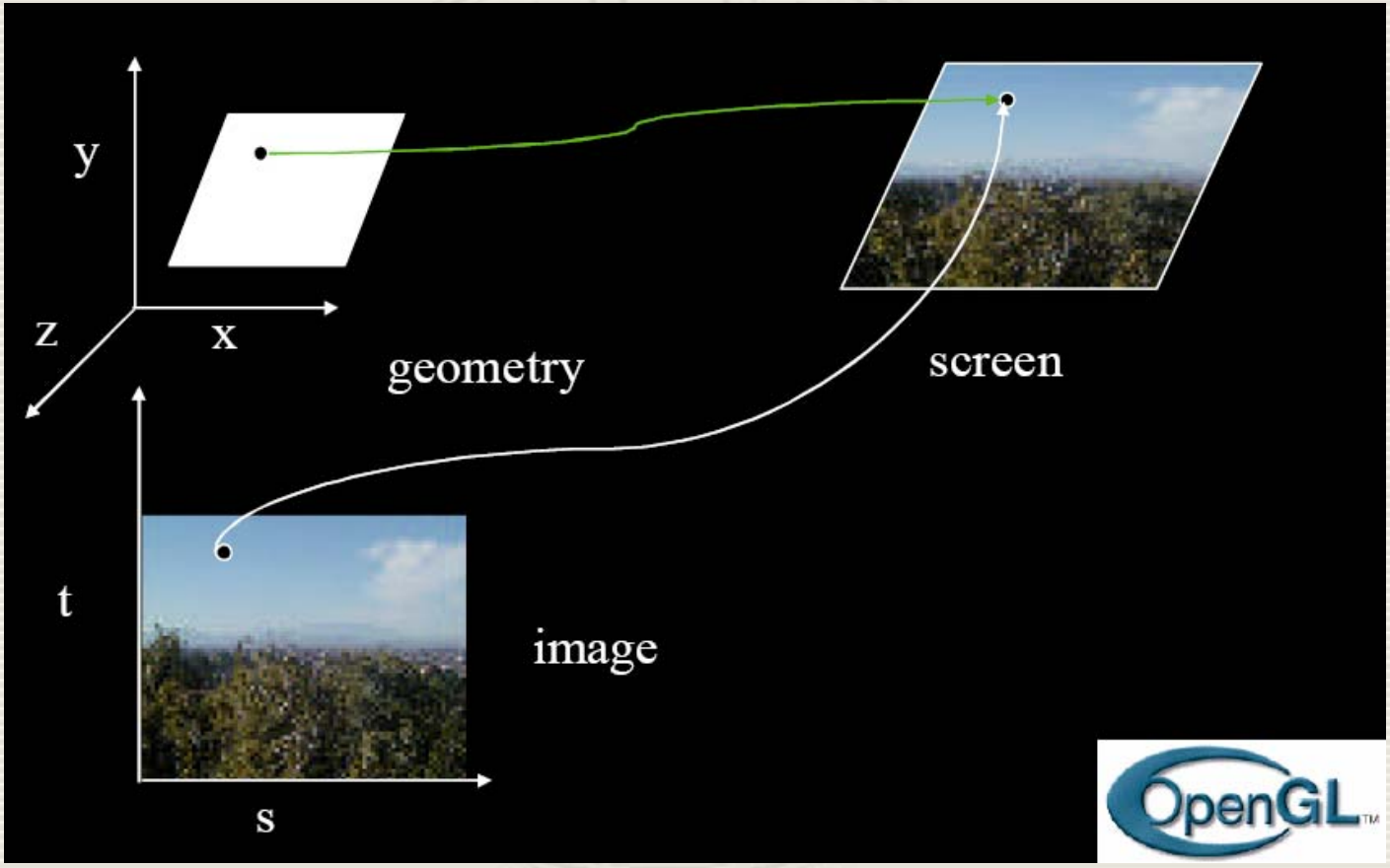
Libreria Grafica OpenGL parte III

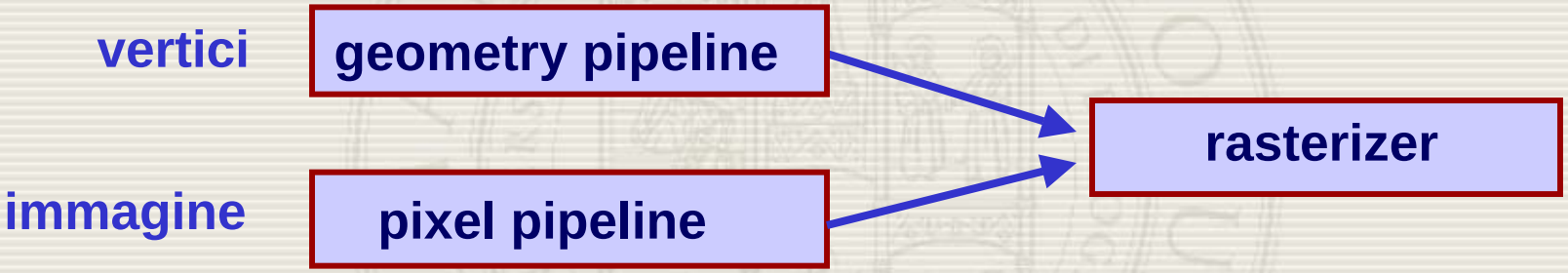
www.opengl.org

Texture mapping



Texture mapping e la pipeline OpenGL

Immagini e oggetti geometrici seguono due pipeline separate e si uniscono poi nella fase di rasterizzazione



OpenGL non fornisce funzioni per acquisire i formati immagine come JPEG, PNG, GIF, PPM, …
Il file immagine deve essere letto e decodificato per ottenere l'informazione colore, e solo allora OpenGL potrà passare l'informazione al rasterizzatore.

Applicare Texture

1. Specificare la texture
   - Leggere o generare l'immagine
   - Assegnare l'immagine alla texture RAM

2. Specificare i parametri della texture
   - Filtering, wrapping, …

3. Assegnare le coordinate texture ai vertici

Ricordiamoci di Abilitare/Disabilitare il texture mapping

glEnable(GL_TEXTURE_2D)

glDisable(GL_TEXTURE_2D)

Creare un texture object e specificare una texture per quell'oggetto

Un texture object memorizza texture data (nella texture RAM) e li rende disponibili per un loro utilizzo/riutilizzo

1. (init( )) Generare un nome per un texture object
   glGenTextures(n, &texName);

2. (init( )) Creare texture object per una texture image
   glBindTexture(GL_TEXTURE_2D, texName);

4. (init( )) Dichiarare l'array di texels :
   GLubyte my_texels[512][512];

5. (init( )) Definire o leggere una immagine in texels[ ][ ]

Creare un texture object e specificare una texture per quell'oggetto (continua)

6. (init( )) Definisce una texture image 2D da un array di texel in texture RAM:

```
glTexImage2D( target, level, components,
        w, h, border, format, type,*my_texels );
```

Es.: `glTexImage2D(GL_TEXTURE_2D, 0, 3, 512, 512, 0`
`GL_RGB, GL_UNSIGNED_BYTE, my_texels );`

- *level* = livello della mipmap, 0 se non si utilizza mipmapping;
- *components* componenti colore (RGBA) considerate;

Creare un texture object e specificare una texture per quell'oggetto (continua)

7 . (init( )) Se le dimensioni non sono una potenza di due:

```
gluScaleImage( format, w_in, h_in, type_in,
        *data_in, w_out, h_out, type_out, *data_out );
```

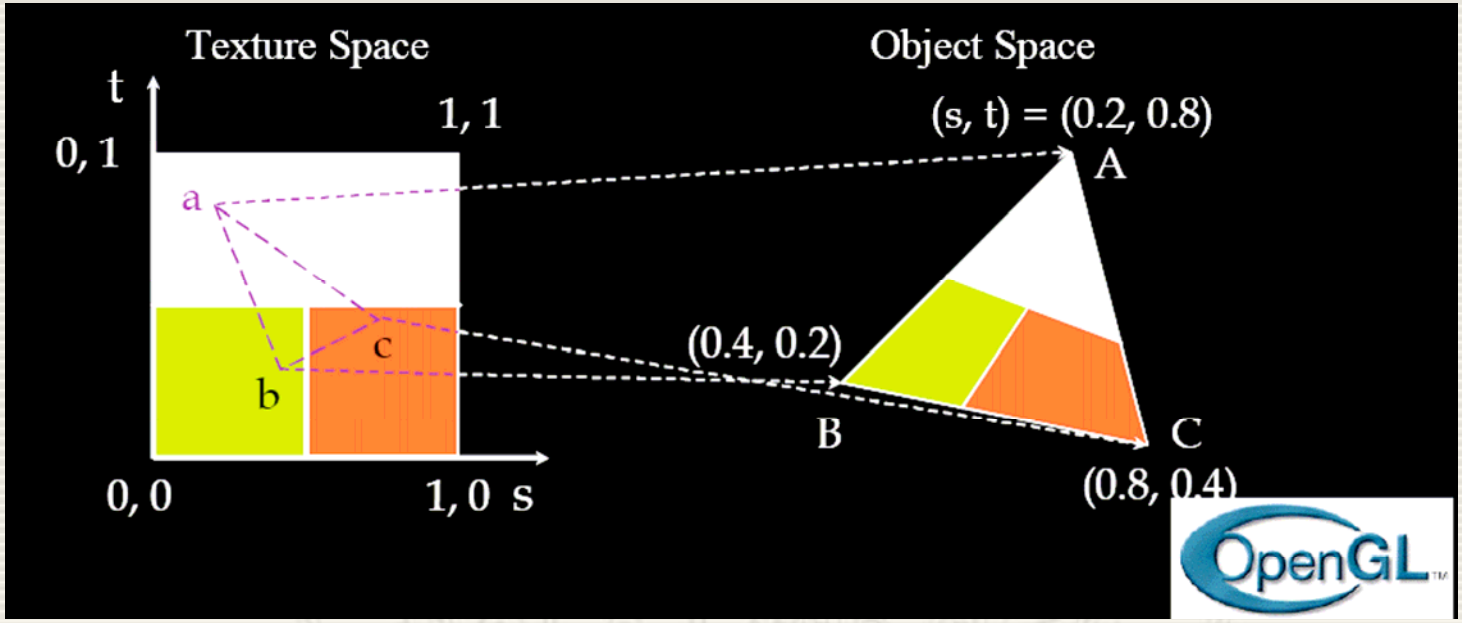
*_in sta per source image

*_out sta per destination image

8. (display( )) Seleziona la texture ogni volta che si deve rendere l'oggetto al quale la texture è associata

```
glBindTexture(GL_TEXTURE_2D, texName[i]);
```

Indicare come la texture deve essere applicata all'oggetto




```
/* associare ad ogni vertice dell'oggetto un punto della texture */
                    glBegin( );
                      glTexCoord2f(0.2,0.9);
                      glvertex3f(0.2,0.8,0.0);
                      ……….
                    glEnd( );
```

Esempio di Texture 2D

Nella figura la texture è un'immagine 256x256 che è stata mappata in un poligono rettangolare visto in prospettiva.

NOTA: texture mapping lavora solo in RGBA

```
glBindTexture(GL_TEXTURE_2D,texName);
glBegin(GL_QUAD);
  glTexCoord2f(0.0,0.0);
  glvertex3f(x1,y1,z1);
  glTexCoord2f(1.0,0.0);
  glvertex3f(x2,y2,z2);
  glTexCoord2f(0.0,1.0);
  glvertex3f(x3,y3,z3);
  glTexCoord2f(1.0,1.0);
  glvertex3f(x4,y4,z4);
glEnd( );
```

Interpolazione valori texture

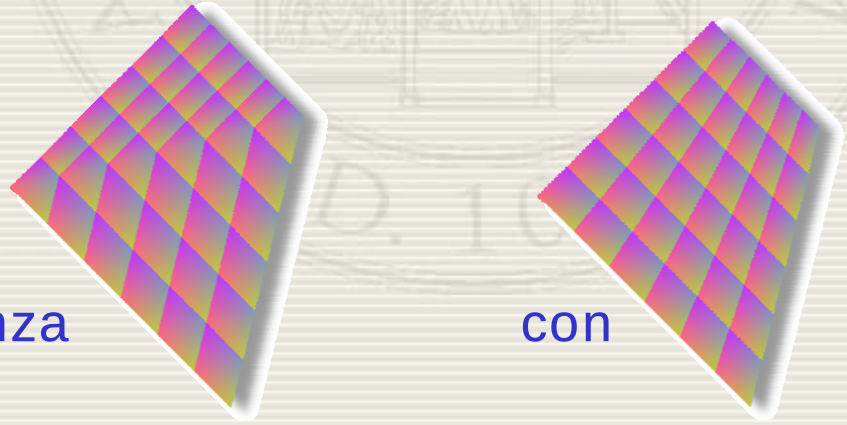
OpenGL usa interpolazione bilineare per determinare le coordinate texture dei punti interni al poligono così come per trovare i colori interni dai colori dei vertici.

Per utilizzare l'interpolazione con correzione prospettica:

```
glHint(GL_PERSPECTIVE_CORRECTION, hint);
```

dove

hint: GL_NICEST, GL_FASTEST, GL_DONT_CARE



Come applicare le texture

Generare automaticamente le coordinate texture per i vertici (two part mapping):

glEnable(GL_TEXTURE_GEN_{STRQ})

glTexGen{ifd}[v](coord, pname, param)

Genera coordinate texture basate sulla distanza dei vertici dal piano specificato.

coord = GL_S, GL_T, GL_R, GL_Q

pname = GL_TEXTURE_GEN_MODE, GL_OBJECT_PLANE, GL_EYE_PLANE

Se pname = GL_TEXTURE_GEN_MODE ***allora***

param = (Modalità di generazione)

- **GL_OBJECT_LINEAR** **(piano di riferimento in coord. mondo)**
- **GL_EYE_LINEAR** **(piano di riferimento in coord. vista)**
- **GL_SPHERE_MAP** **(environment mapping)**

Altrimenti *param* specifica i parametri per la funzione di generazione

Environment Texture mapping

```
glTexGeni(GL_S, GL_TEXTURE_GEN_MODE,
    GL_SPHERE_MAP)
glTexGeni(GL_T, GL_TEXTURE_GEN_MODE,
    GL_SPHERE_MAP)
glEnable(GL_TEXTURE_GEN_S)
glEnable(GL_TEXTURE_GEN_T)
```

Genera la proiezione di una texture su un oggetto tramite una sfera.

Modalità di applicazione delle texture

Modalità di filtri

- minification o magnification
- speciali filtri di minification mipmap

Modalità Wrap

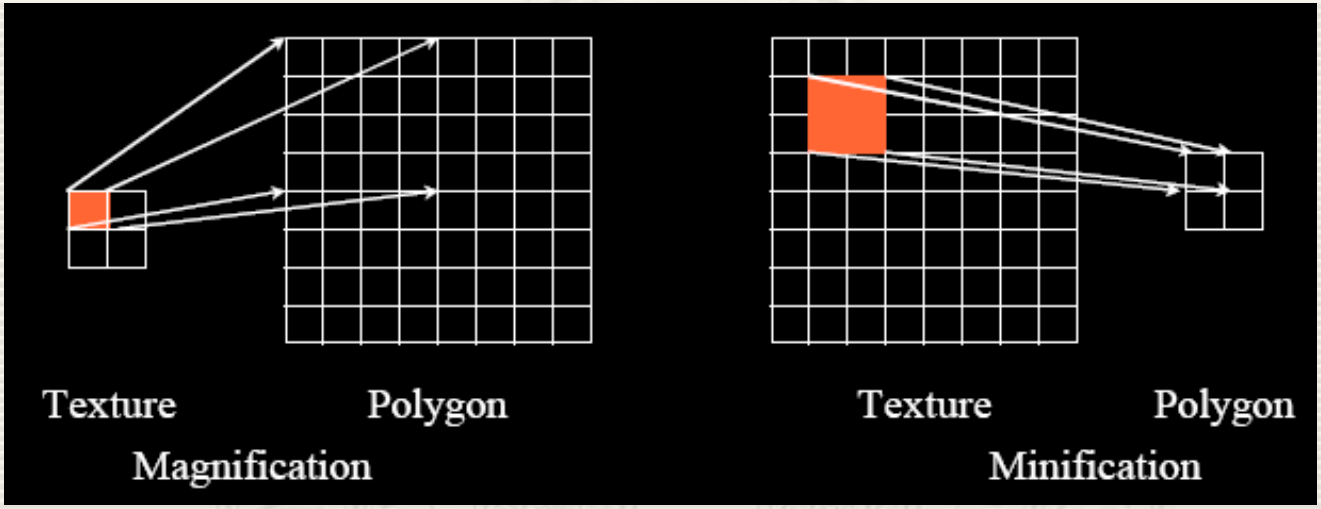
- clamping o repeating

Interazioni tra texture e shading

come combinare il colore della primitiva con il colore della texture: *blend*, *modulate* o *replace*

Modalità di filtri: anti-aliasing

glTexParameteri(GL_TEXTURE_2D, type, mode)



Magnification:
il texel è più piccolo di un pixel
Soluzione: interpolazione

Minification:
il texel è più grande di un pixel
Soluzione: media

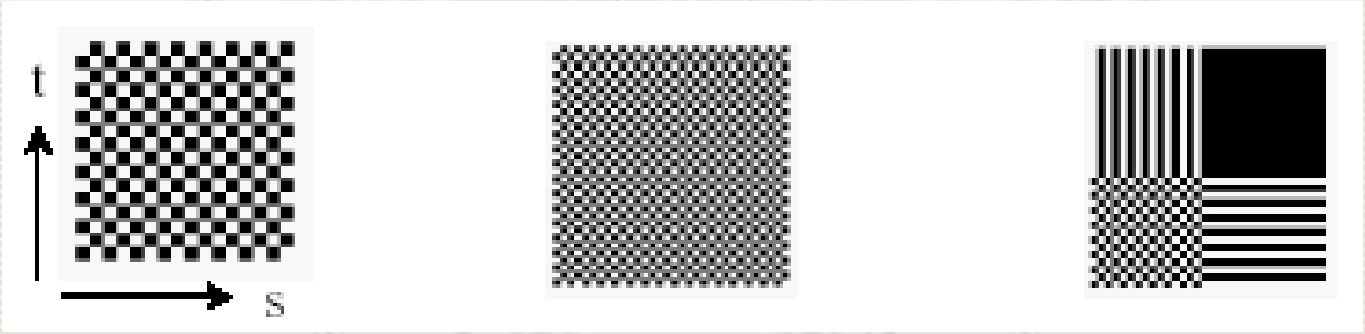
type : GL_TEXTURE_MIN_FILTER o GL_TEXTURE_MAG_FILTER.
mode: GL_NEAREST, GL_LINEAR, o modalità speciali per mipmapping

(Minification) **gluBuild2DMipmaps( );**

Modalità di Wrap

Es: Se specifichiamo valori di una texture maggiori di [0,1]

```
glTexParameteri( GL_TEXTURE_2D,
                 GL_TEXTURE_WRAP_S, GL_CLAMP )
glTexParameteri( GL_TEXTURE_2D,
                 GL_TEXTURE_WRAP_T, GL_REPEAT )
```



texture

GL_REPEAT

> [0,1]

GL_CLAMP

Valori >1.0 set 1.0
Valori < 0.0 set 0.0

Tutorial Texture
(gltutorials)

Programmi esempio