Textures

Outline

- Textures in image synthesis:
  - The needs: realism and complexity
  - Definition and usage
- Discrete Textures
  - Definition
  - Texture mapping
  - Textures in OpenGL
  - Filtering
- Tutorial

Realism and complexity

- To be credible, a scene must present :
  - A realistic geometric complexity:
    - Thin geometric elements
    - Thin geometric sampling
  - A realistic appearance complexity:
    - Thin color elements
    - Thin color sampling

Realism and complexity

- Unfortunately, a geometric approach is problematic:
  - A mesh composed of billions of thin triangles:
    - Is expensive to render
    - Causes aliasing artifacts due to the high number of edges generated
- We can always increase the number of triangles but not for real-time rendering
- Instead, use an image-based approach...

Definition

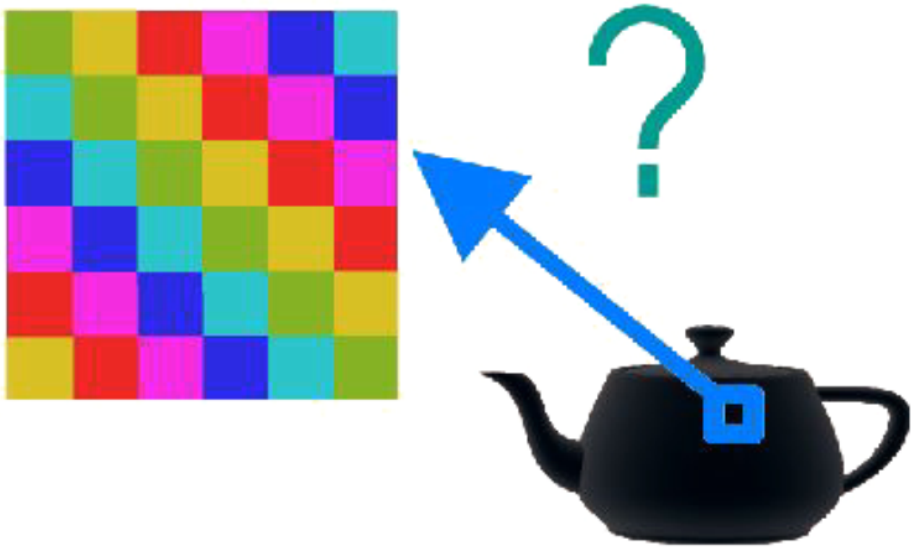
- Texture:
  - Caracterisation of the thin (high frequency) details of an object using images
- The use of textures can increase an object realism while keeping its geometry as simple as possible.
- Textures “cover” the geometry:
  - Color
  - Materials
  - Normals

Discrete Textures

- Definition:
  - Map representing the details of the appearance of an object
    - Color map
    - Normal map
  - Map representing global lighting effects:
    - Reflexions
    - Shadows
  - Dimensions:
    - 1D : linear distribution (for instance: color gradient)
    - 2D : standard picture
    - 3D : volume distribution

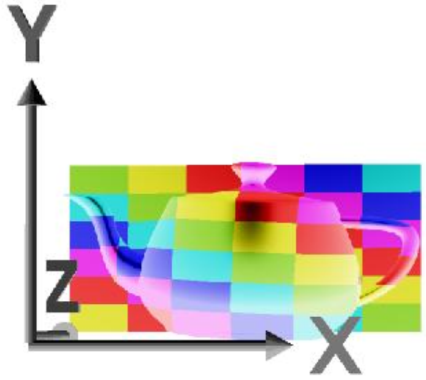
Texture mapping

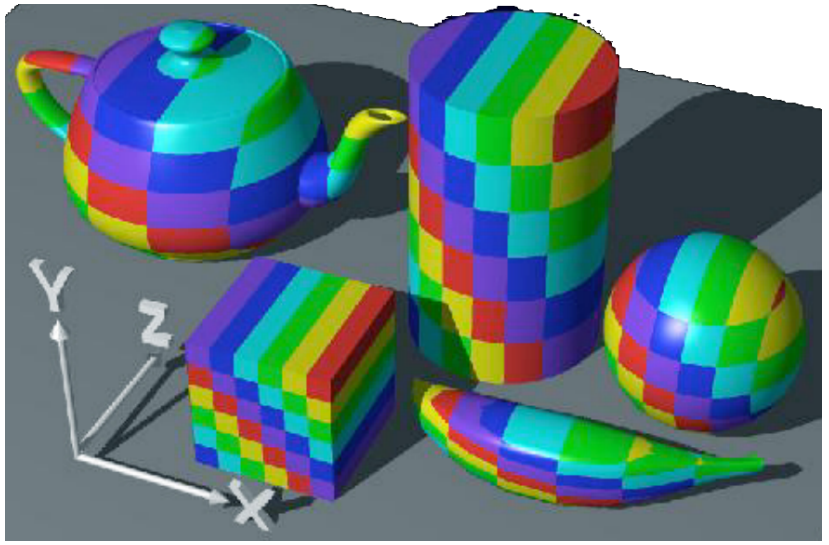
- How do we map from a 3D point in space to a point of the texture ?
  - The mapping function:
    - $f : \mathbb{R}^3 \rightarrow \mathbb{R}^2$
    - Tells how the texture “covers” the object



Mapping functions

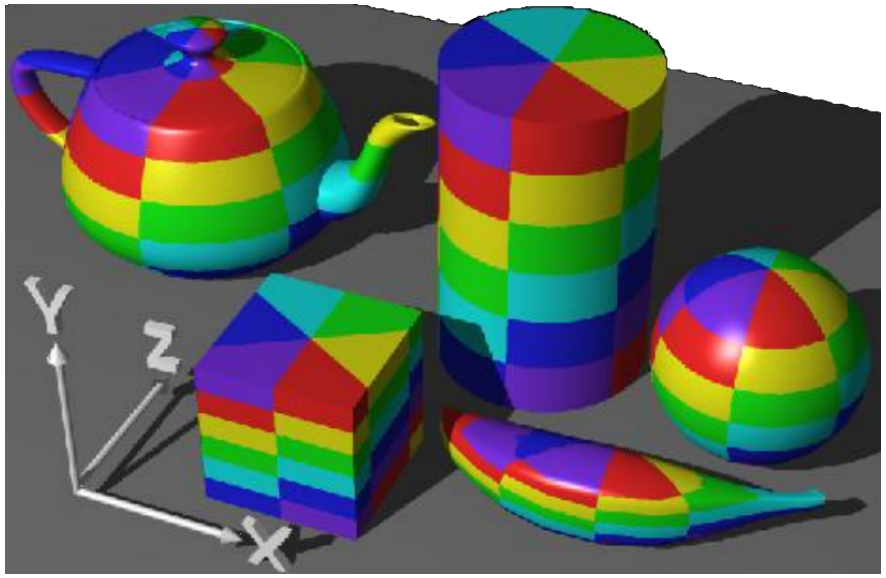
- Planar projection:
  - The points are projected along some axis





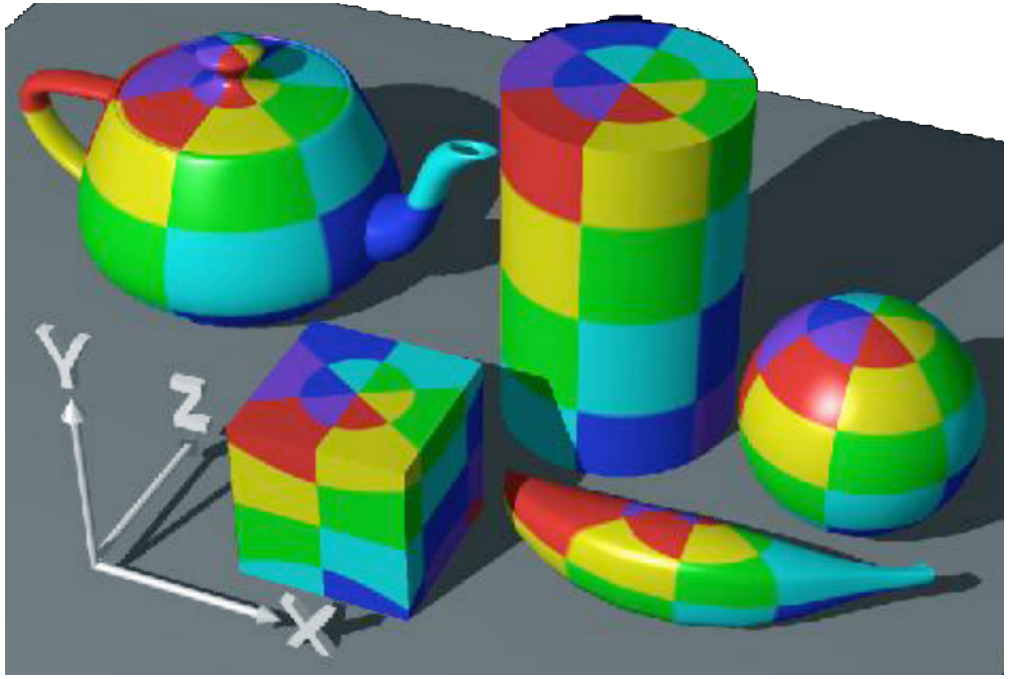
Mapping functions

- Cylindrical mapping:
  - The points are projected on a cylinder



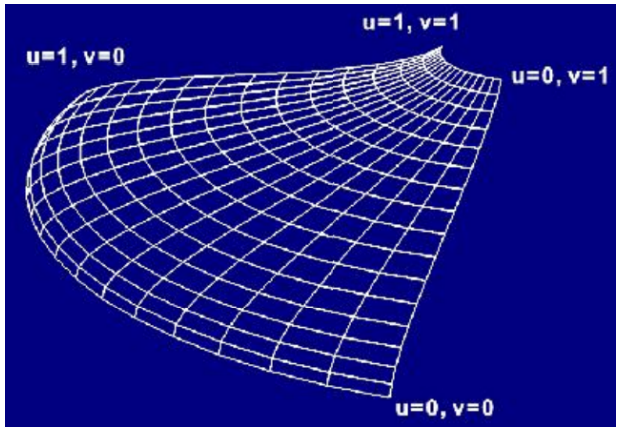
Mapping functions

- Spherical mapping:
  - The points are projected on a sphere



Mapping functions

- Parametrical mapping:
  - Explicit definition of the mapping function
  - Texture coordinates: attributes of the vertices



Textures in OpenGL

- Textures are manipulated through a **texture object**
- Functions to create or delete a texture object:
  - glGenTexture(GLsizei n, GLuint* texnames);
    - Create n new texture objects and write their ID in texnames
  - glDeleteTexture(GLsizei n, Gluint* texnames);
    - Delete previously created textures

- Setting the current texture to work on / with (state machine):
  - GlBindTexture(GLenum target, Gluint texname);
    - target is GL_TEXTURE_xD with x = 1, 2 or 3
    - texname: the ID given by glGenTexture

Textures in OpenGL

- Initializing a texture object:

  - glTexImage{1|2|3}D(GLenum target,
    GLint level, /* mipmap level */
    GLint internalFormat, /* internal format of the texture data*/
    GLsizei width, /* width of the texture in pixels */
    GLsizei height, /* height of the texture in pixels */
    GLint border, /* presence of a border */
    GLenum format, /* external format of the texture data*/
    GLenum type, /* type of the data of the input image*/
    const GLvoid* pixels); /* array of pixels*/

- Many ways of initializing a texture...

Textures in OpenGL

- Many internal formats available:
  - GL_ALPHA, GL_LUMINANCE, GL_RGB, GL_RGB16, GL_RGBA, GL_R3_G3_B2, …
- Many external formats available:
  - GL_RGB, GL_BGR, GL_BGRA, GL_LUMINANCE, GL_RED, GL_GREEN, GL_BLUE, …
- Many data types for the input image:
  - GL_UNSIGNED_BYTE, GL_BYTE, GL_INT, GL_FLOAT, ...

Textures in OpenGL

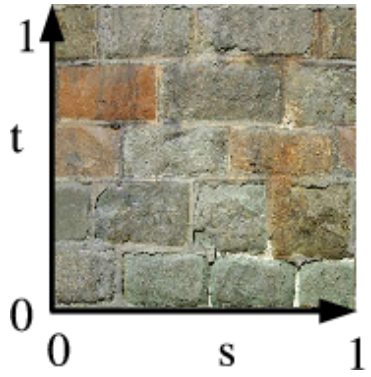
- Updating a portion of a texture:
  - glTexSubImage2D(GLenum target, GLint level, GLint xoffset, GLint yoffset, GLsizei width, GLsizei height, GLenum format, GLenum type, const GLvoid* pixels);
    - Updates the rectangle of size width x height located at position (xoffset, yoffset) in the texture
- Updating a texture is efficient:
  - No new memory allocation

Textures coordinates

- Per vertex explicit definition:
  - glTexCoord{1|2|3|4}{s|i|f|d}[v](TYPE[*] coords);
    - Define the texture coordinates that will be used by the next vertices
- Automatic generation from the vertex coordinates:
  - glTexGen{ifd}[v](enum coord, enum pname, TYPE[*] param);
- Up to 4 texture coordinates: s, t, r and q (1D, 2D, 3D, 3D homogeneous)
- Transformed by the **texture matrix:**
  - glMatrixMode(GL_TEXTURE);

Textures coordinates

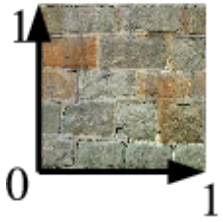
- Texture coordinates are defined in the range [0,1]
- 2D texture: coordinates (s, t) per vertex:



- What if texture coordinates are outside this range ?

Textures coordinates

- Managing coordinates outside range [0;1]:
  - glTexParameteri(GLenum target, GL_TEXTURE_WRAP_coord, GLenum param);
    - GL_CLAMP: clamp to range [0;1]
    - GL_REPEAT: use the fractional part of the texture coord.
    - GL_MIRRORED_REPEAT



GL_CLAMP

GL_REPEAT

GL_MIRRORED_REPEAT

Color blending

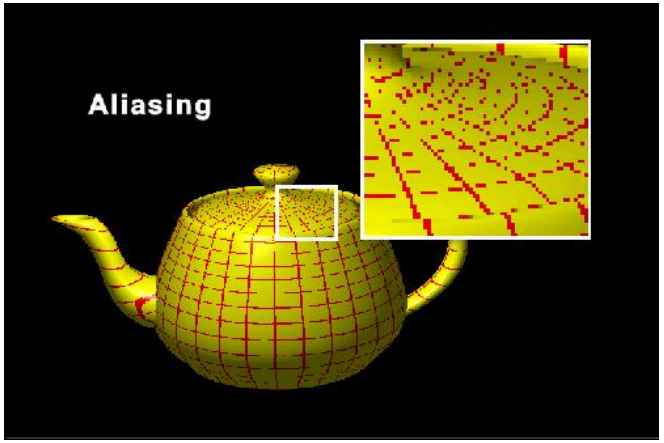
- We need to tell openGL how to compute the final color:
- Setting the blending function:

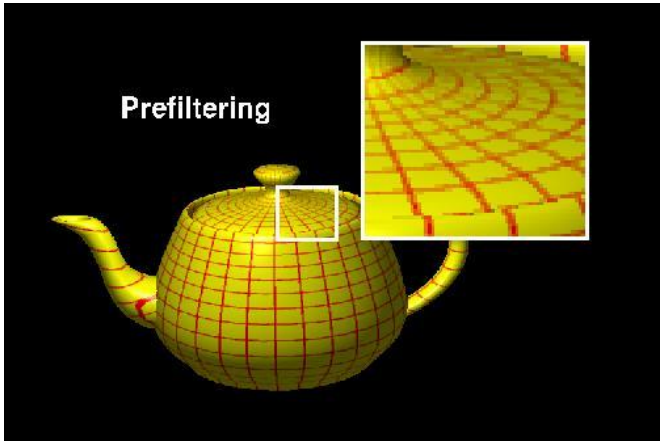
  - glTexEnv{if}[v](GL_TEXTURE_ENV,
    GL_TEXTURE_ENV_MODE,
    TYPE[*] param);

- Different blending functions available:
  - GL_MODULATE : multiplication
  - GL_REPLACE : only use texture's color
  - GL_ADD : addition
  - ...

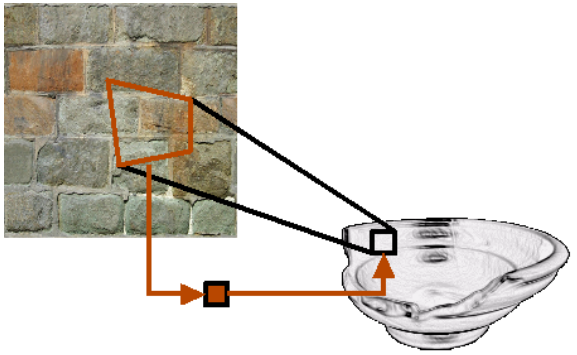
Filtering

- The texture's sampling usually do not match the one of the screen:



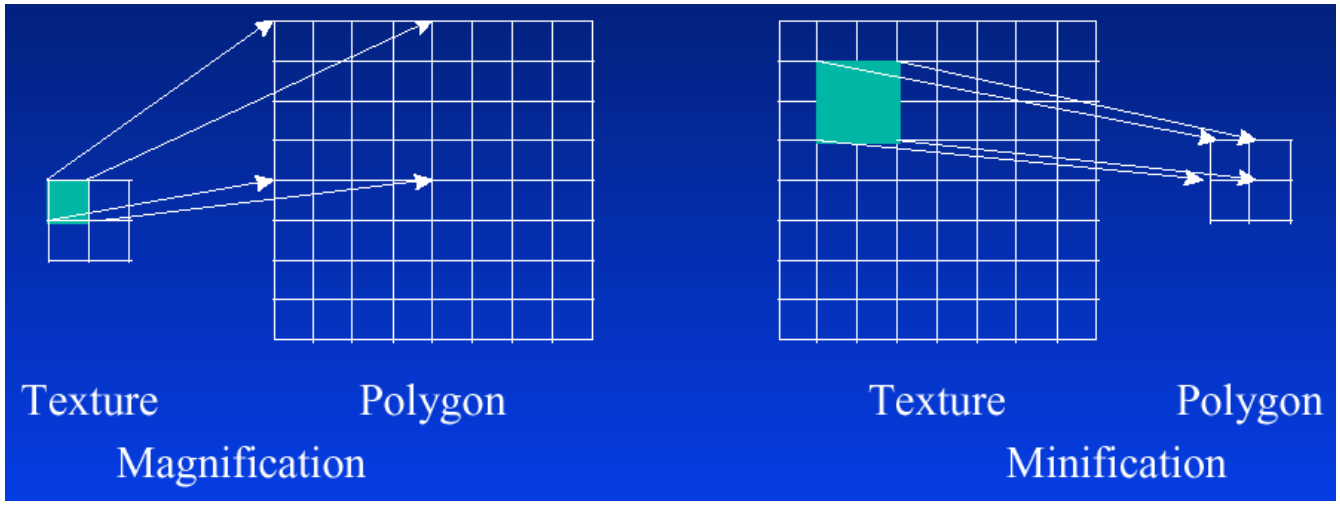


- Theoretical perfect filter:
  - Project the pixel into the texture frame and sum over the corresponding area
  - Too expensive...



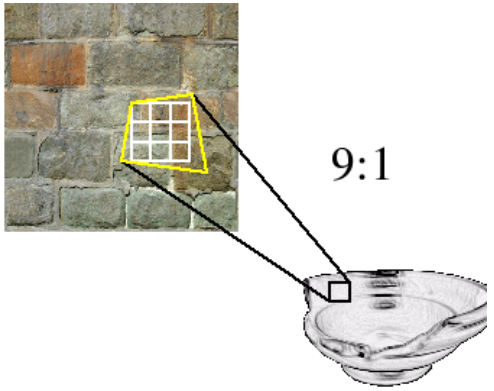
Filtering

- Setting the filtering function:
  - glTexParameteri(Glenum target, Glenum pname, Glenum param);
    - pname: the type of filtering
      GL_TEXTURE_MAG_FILTER zoom in
      GL_TEXTURE_MIN_FILTER zoom out
    - param: the filtering function to use
      GL_LINEAR linear interpolation
      GL_NEAREST nearest neighbour



Mipmap texture filtering

- Precomputation of several filtered versions of a texture
- Texture “pyramid”:

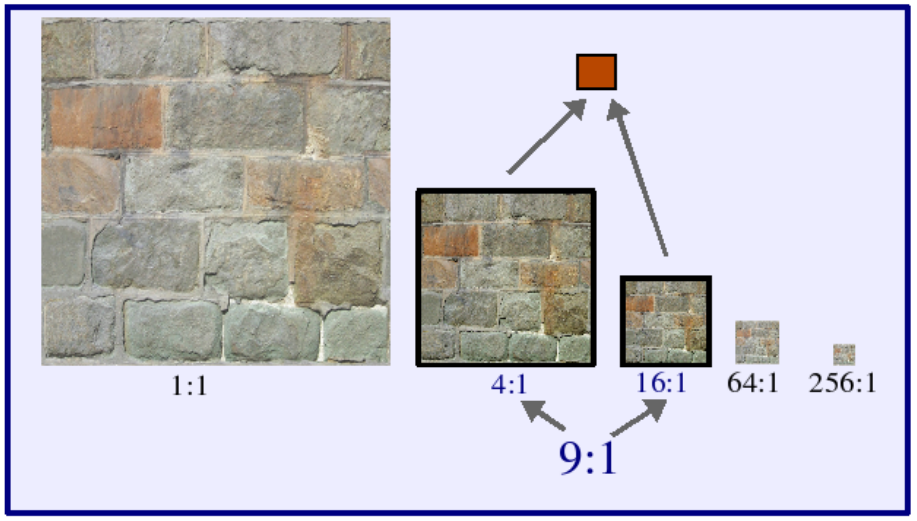


  - gluBuild2DMipmaps(enum target, int internalFormat, int h, int w, enum format, enum type, void* pixels);
    - Arguments are the same as the ones of glTexture2D(...)

- OpenGL automatically chooses the right mipmap level

Mipmap texture filtering

- New filter functions available:
  - Select the nearest mipmap level:
    GL_NEAREST_MIPMAP_NEAREST
    GL_LINEAR_MIPMAP_NEAREST
  - Interpolate between the 2 nearest mipmaps
    GL_NEAREST_MIPMAP_LINEAR
    GL_LINEAR_MIPMAP_LINEAR



Tutorial

- In this tutorial we will manipulate OpenGL's texture objects
- Create a 2D texture made of stripes. The texture's data is an array of unsigned bytes that contains a luminance value only (GL_LUMINANCE)
- Set the parameters of the texture (function glTexParameteri(...)). Set the wrap mode to GL_REPEAT for coordinates s and t. Set the filtering mode to GL_NEAREST.
- Display the texture on a fullscreen quad. Set the blending function to GL_REPLACE with: glTexEnvf(GL_TEXTURE_ENV, GL_TEXTURE_ENV_MODE, GL_REPLACE);
- Set the texture transformation matrix so as to rotate the texture 90° clockwise.
- Display the texture on a teapot. The texture coordinates of the teapot are already defined.
- Set the filtering mode to GL_LINEAR. What do you notice ?
- Rotate the texture 45° clockwise. What do you notice?