COM3503/4503/6503: 3D Computer Graphics

Lecture 13: Texture: part 3

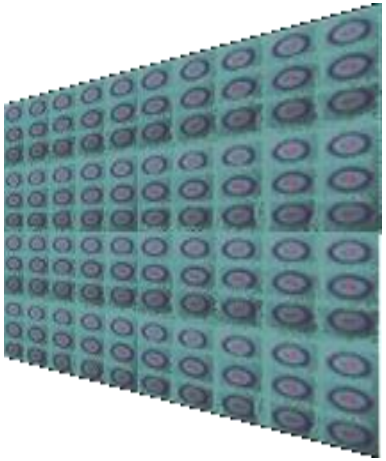
http://commons.wikimedia.org/wiki/File:Liquorice_allsorts_2010.jpg
By Ali K (Own work) [CC-BY-SA-3.0 (http://creativecommons.org/licenses/by-sa/3.0) or GFDL (http://www.gnu.org/copyleft/fdl.html)], via Wikimedia Commons

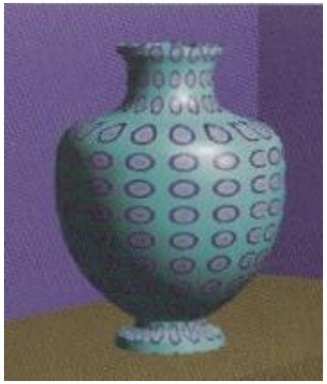
Dr. Steve Maddock
Room G011, Regent Court
s.maddock@sheffield.ac.uk

1. Introduction

- 2D texture colour/diffuse mapping pastes a bitmap picture (texture) onto the surface of an object

- There are a range of approaches for pasting the 2D texture on the surface of a 3D object

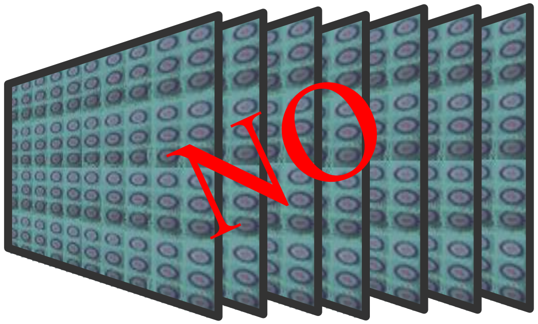




1. Introduction

- 3D texture mapping places an object inside a 3D pattern

- The 3D pattern is NOT represented as an array of 2D textures
  - This would be memory inefficient
  - Many values would remain unused



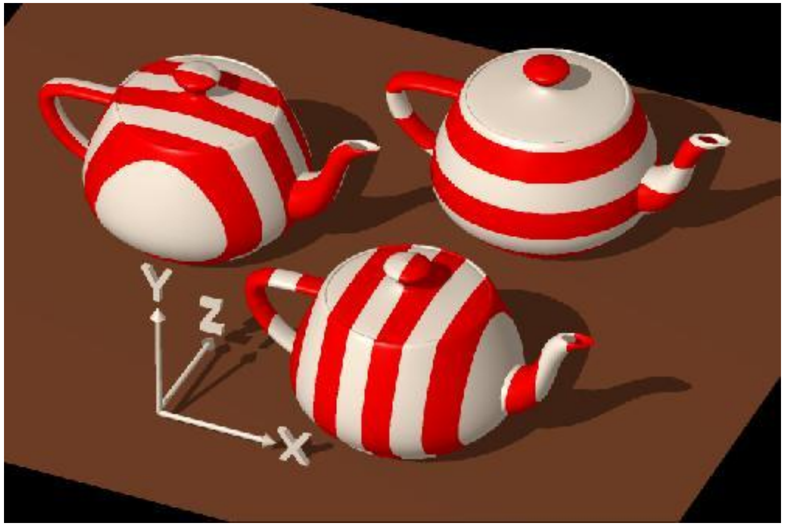
- Instead, the pattern is usually represented as a procedure…

2. 3D texture mapping ('Solid texture') [Perlin,85; Peachey,85]

- For 3D texture mapping, all that is required to texture a point is a 3D coordinate
- A texture function f(x,y,z) is evaluated for each 3D coordinate (x_w,y_w,z_w)
- Example function:

```
if trunc(z) is even {
  colour of point is red
}
else {
  colour of point is white
}
```

- Example: z = 2.4 → red
- Example: z = 1.9 → white

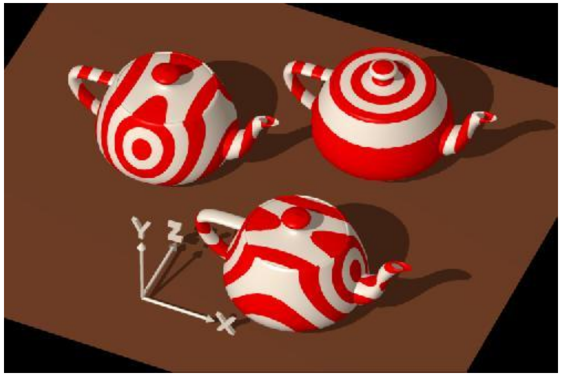


Left teapot:
  if trunc(z) is even then red; else white
Middle teapot:
  if trunc(x) is even then red; else white
Right teapot:
  if trunc(y) is even then red; else white

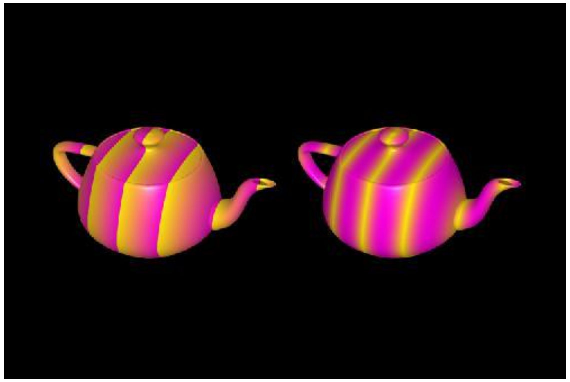
The SIGGRAPH 97
Education Slide set

2. 3D texture mapping

- Left teapot →
  - use x and y to compute radial distance from axis through teapot centre
  - If trunc(result) is even then red; else white



- More complex functions can be used:
  - Left: *mod(x,a)/a*.
  - Right: *(sin(x)+1)/2*

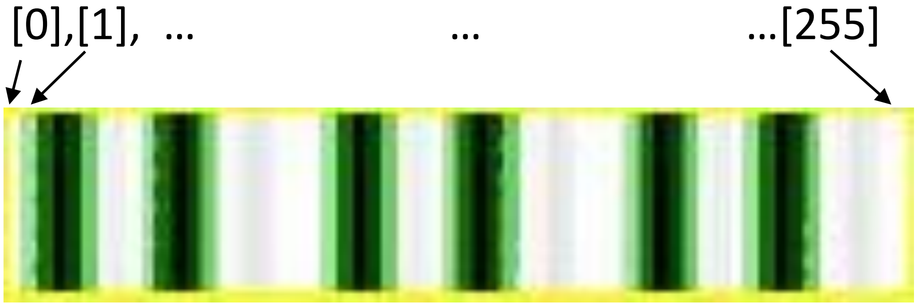


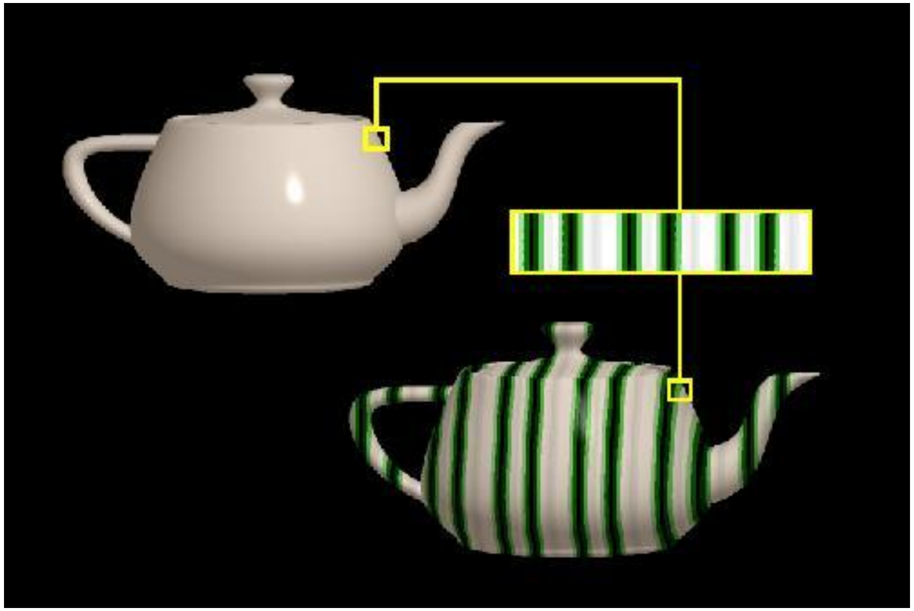
The SIGGRAPH 97
Education Slide set

2. 3D texture mapping

- Here a colour table is used
- Function:

- the fractional part of the x coordinate (0.0 .. 0.99) multiplied by number of elements in colour table (256)
- process repeats

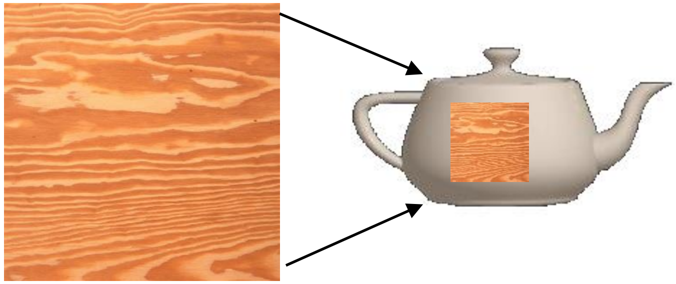


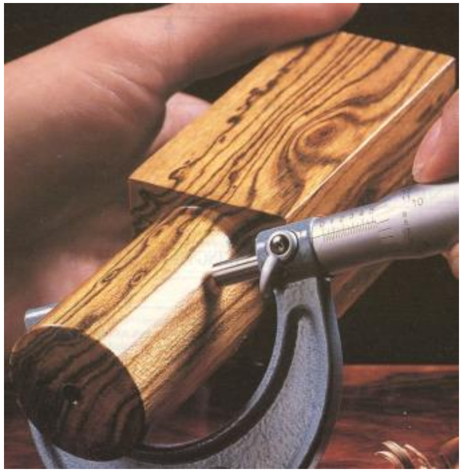


The SIGGRAPH 97 Education Slide set

2.1 Example: wood

- We want to model the appearance of wood
- 2D texture mapping:
  - Take a photo of a wood surface and paste it on to a model
  - Limited to using the photograph (or photographs)
- 3D texture mapping (solid texture)
  - Create a function that produces the pattern of colour variation
  - A good function can produce endless patterns



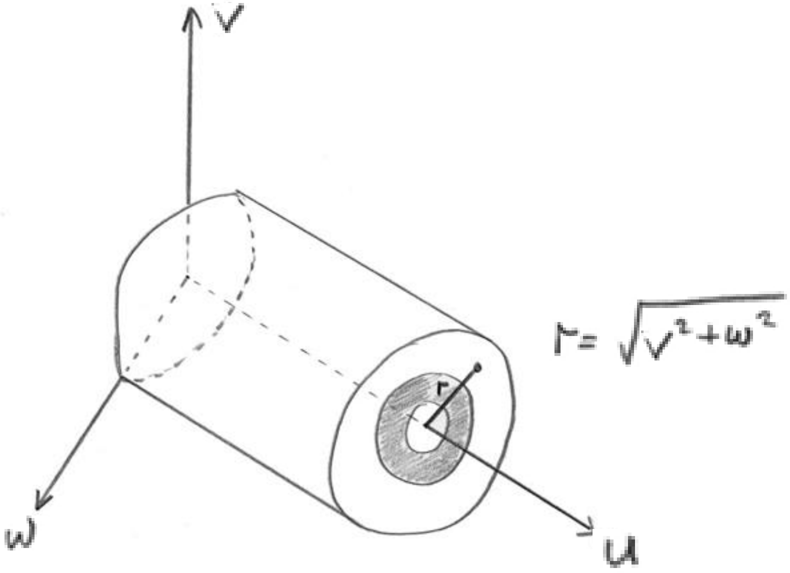


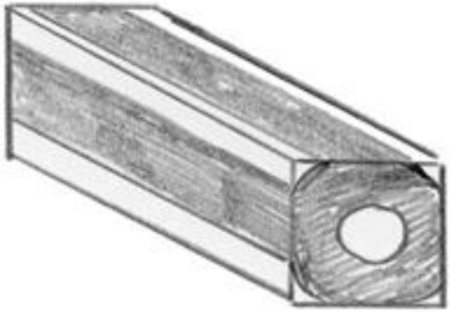
2.1 Example: wood

- Start by defining a function based on a cylinder

- In u, v, w space,
  if trunc(r) is even then dark brown
  else light brown

- Map x,y,z of object to the u,v,w space the texture function is defined in
  - A scale function may be required if the x,y,z space is large in comparison to the u,v,w space



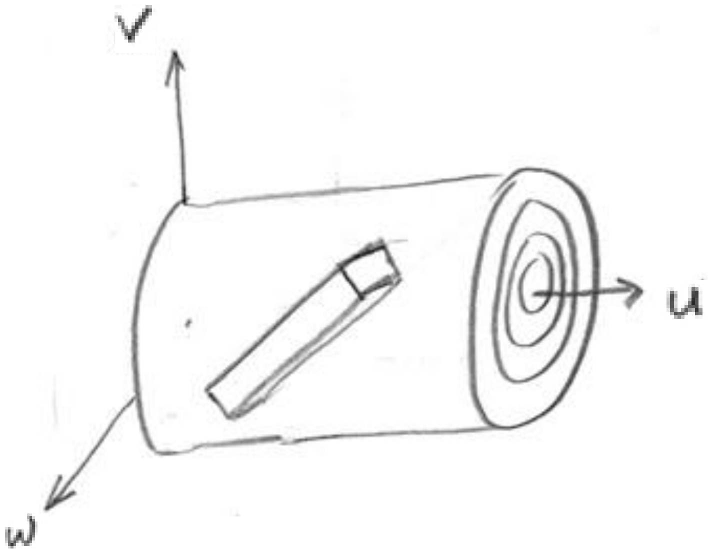


$(u,v,w) \leftarrow (x_w, y_w, z_w)$

2.1 Example: wood

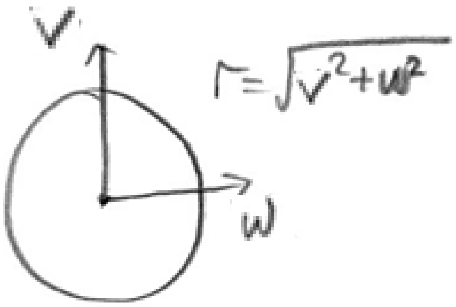
- We can remove the regularity of the function in a number of ways:

- Tilt (i.e. rotate) the (x, y, z) coordinate before using in the function
  - (u, v, w) ← Tilt (x, y, z)

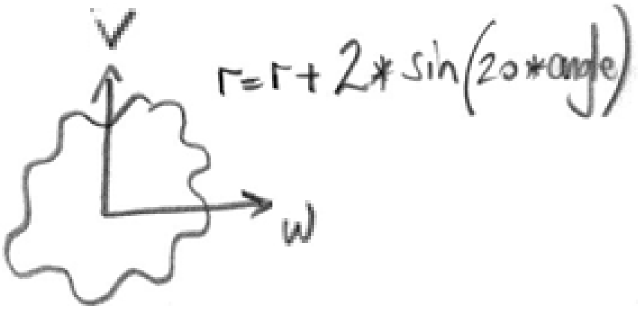


2.1 Example: wood

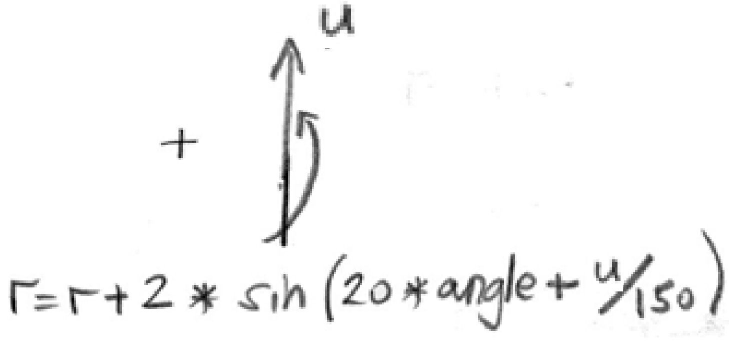
- We can remove the regularity of the function in a number of ways:
- Alter the calculation of r:



- E.g. by adding a sine wave ⇒

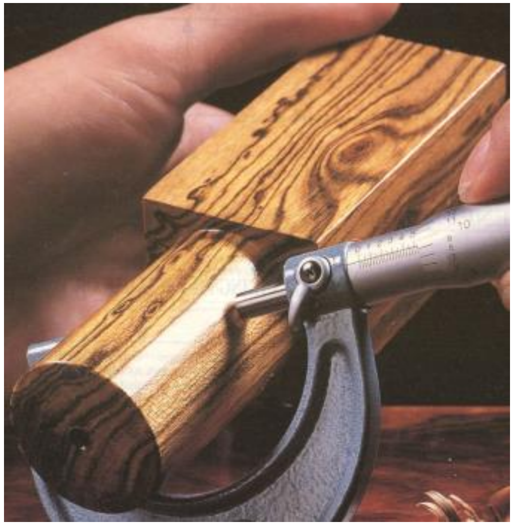


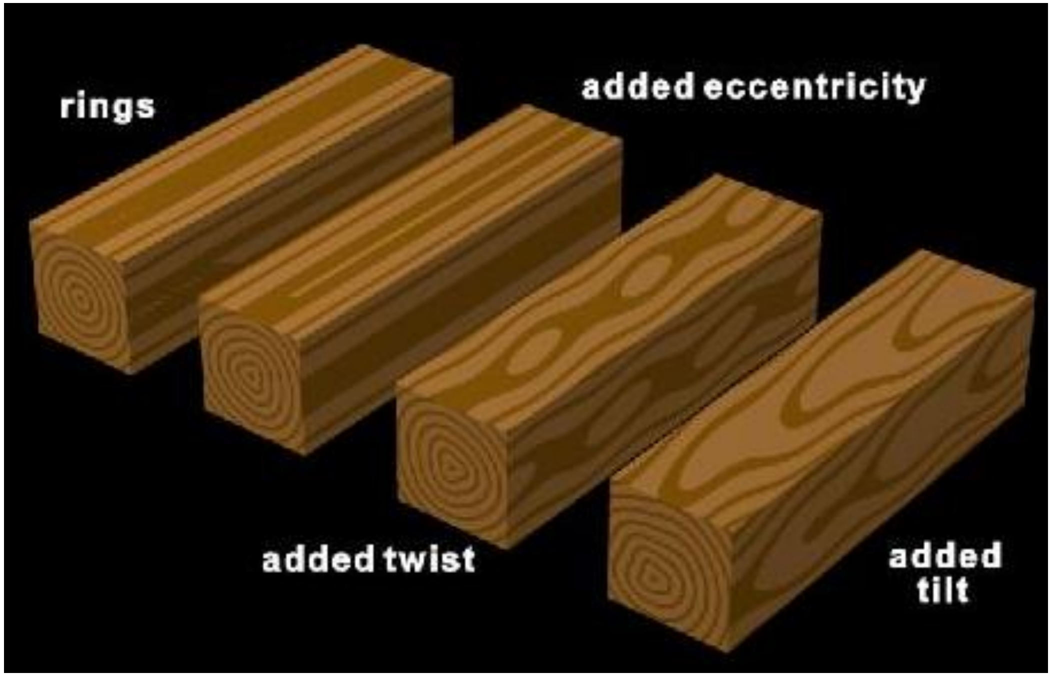
- E.g. by adding a twist with respect to the u axis ⇒



2.1 Example: wood

- These are all combined to give the final result
- However, the result is still too regular in appearance

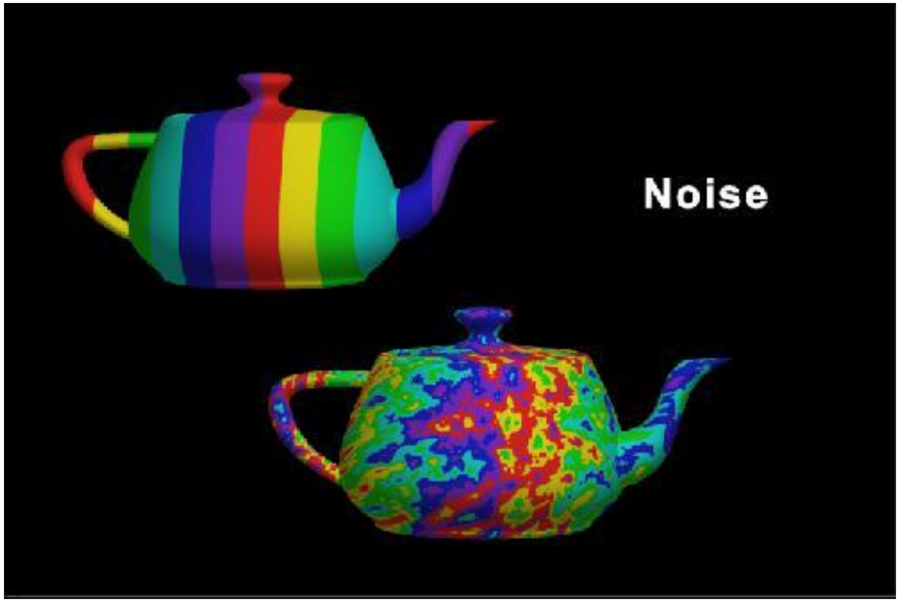




The SIGGRAPH 97
Education Slide set

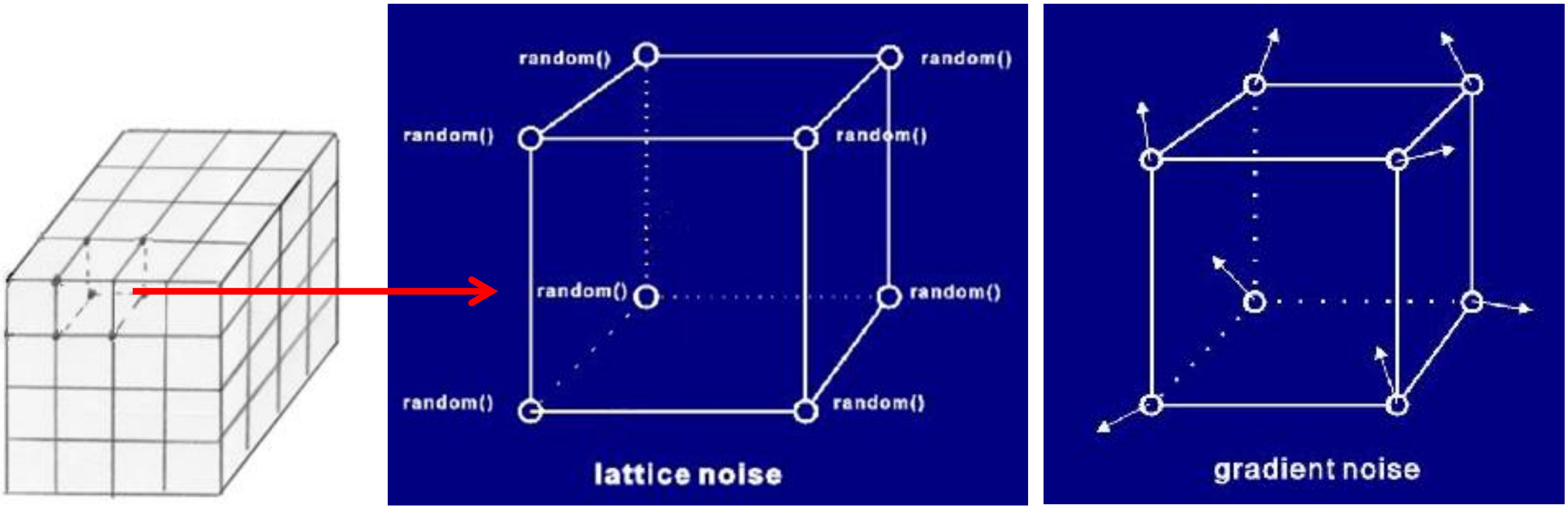
2.2 Adding noise

- To remove the regular appearance, we can add randomness (i.e. noise) to the function
- But, needs to be controllable. Need to be able to:
  - Intuitively control the amount of noise to give different effects
  - Use same values every time the object is drawn



2.2 Adding noise

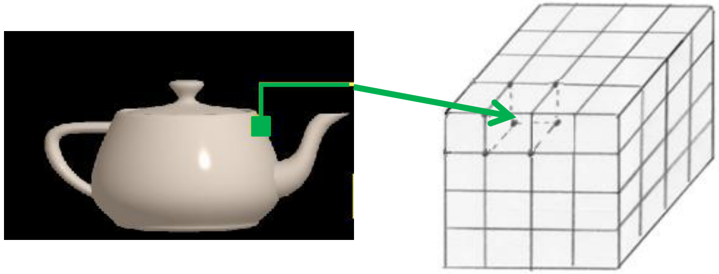
- The noise is generated once and stored in a low resolution grid (u, v, w)
- Lattice noise: random value at each point in a 3D grid
- Gradient noise: random unit vector for each lattice point

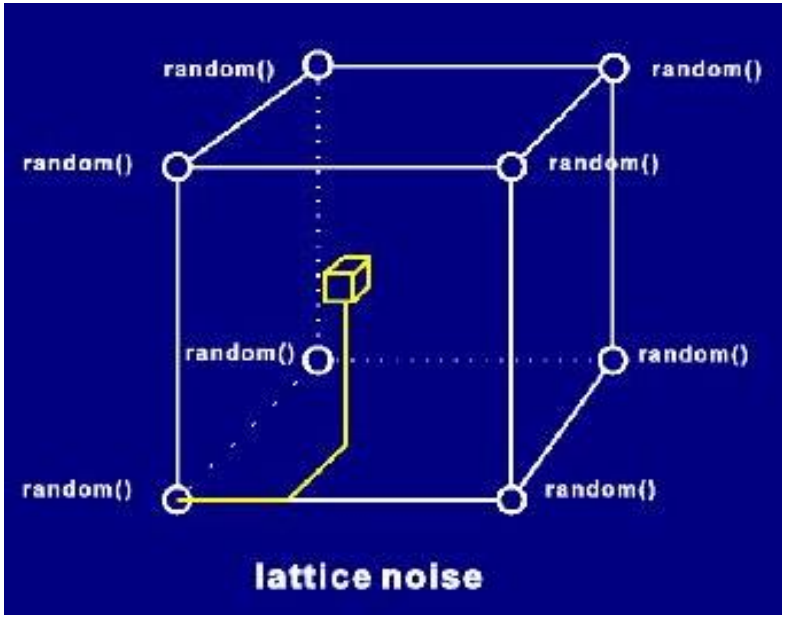


The SIGGRAPH 97
Education Slide set

2.2 Adding noise

- Given a floating point (x,y,z) object position, this is mapped to a floating point (u,v,w)
- The noise value **n(u,v,w)** is calculated using interpolation from surrounding grid points
- n(u,v,w) can be used to alter f(u,v,w) in the procedural texture function
  - The (u,v,w) spaces in each may need mapping between, e.g. scaling
- Each time an object is rendered, (x,y,z) will generate the same (u,v,w) and thus the same n(u,v,w)
- Thus for n(x,y,z):
  - map(x,y,z) to (u,v,w)
  - return n(u,v,w)



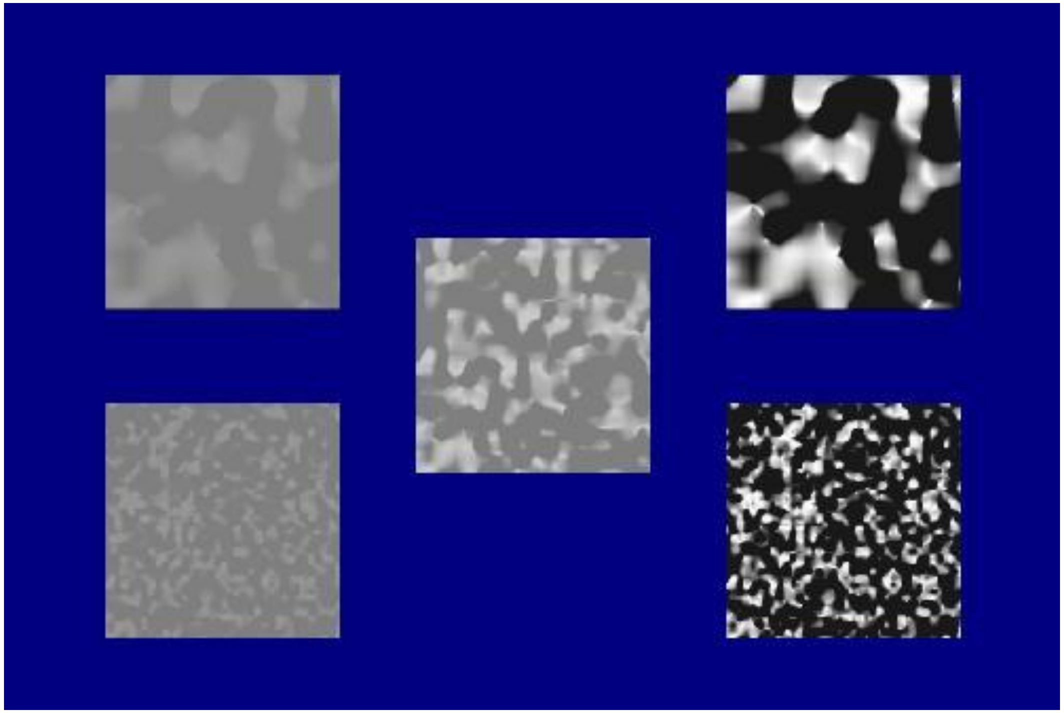


2.2 Adding noise

- For more control, noise(x,y,z) can be rewritten as

  *noise(f*x, f*y, f*z) * a*

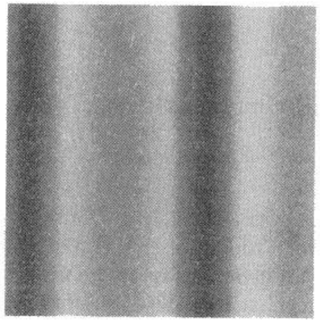
  where *f* controls the frequency and *a* controls the amplitude



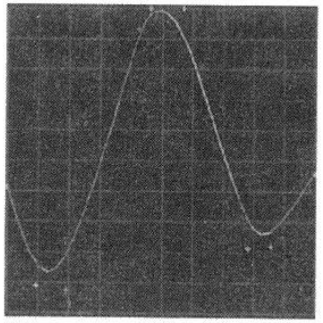
2D slice of a noise grid, The SIGGRAPH 97 Education Slide set

2.3 Example: marble

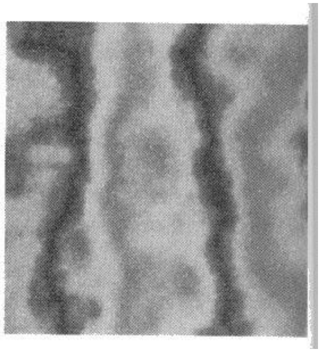
- A colour spline (b) is mapped to shades of grey (a)
- When turbulence is added, we get (c). In 3D, we get (d)
- marble(x) = marble_colour(sin(x+turbulence(x)));



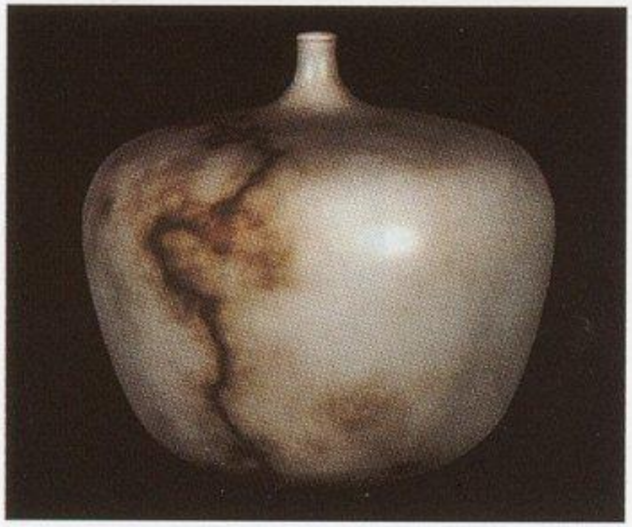
(a)



(b)



(c)

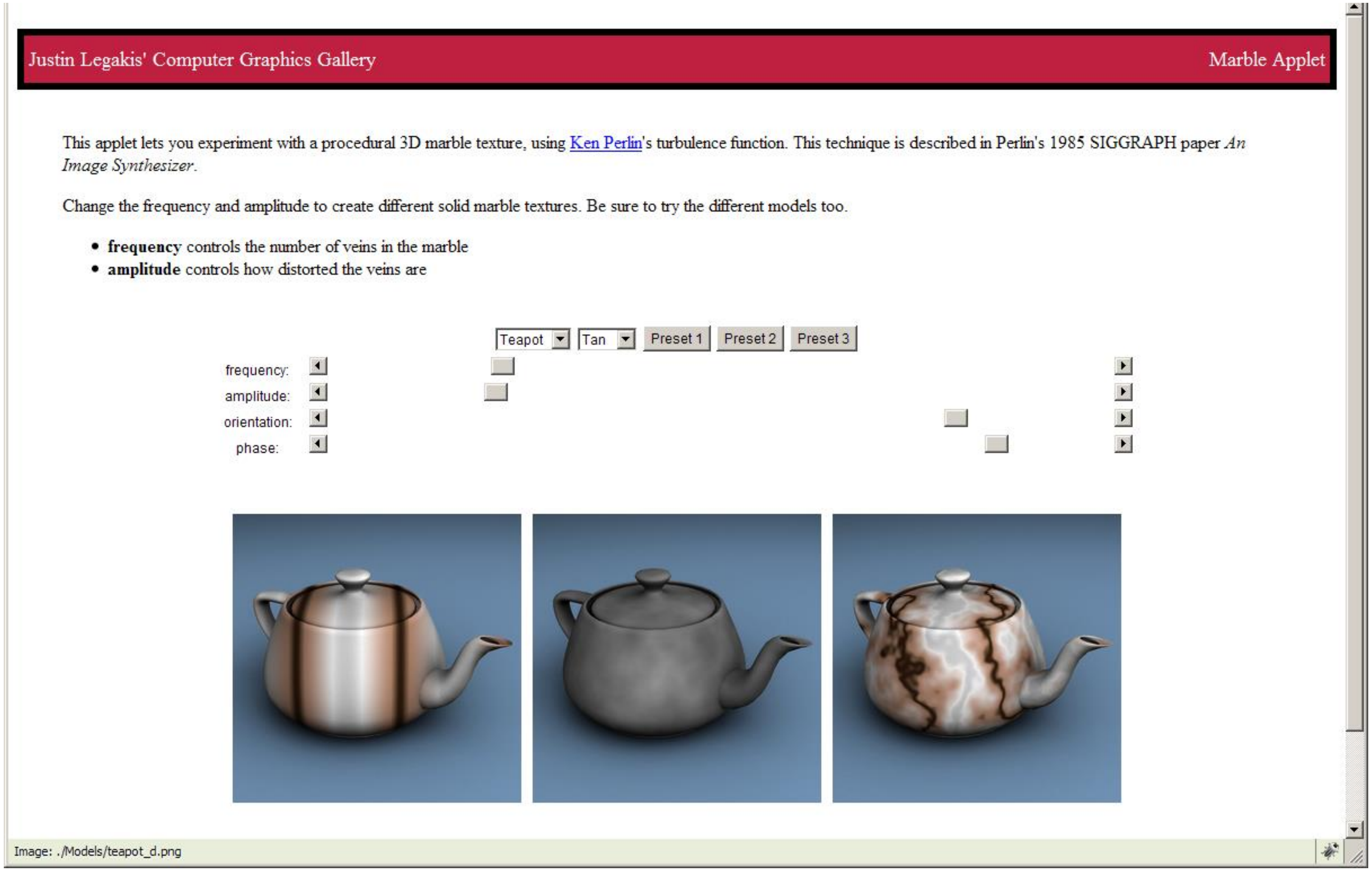


(d)

Figure 8.21 Imitating marble – the classic example of three-dimensional procedural texture.

Watt, 00

http://legakis.net/justin/MarbleApplet/



Justin Legakis' Computer Graphics Gallery

Marble Applet

This applet lets you experiment with a procedural 3D marble texture, using Ken Perlin's turbulence function. This technique is described in Perlin's 1985 SIGGRAPH paper *An Image Synthesizer*.

Change the frequency and amplitude to create different solid marble textures. Be sure to try the different models too.

- **frequency** controls the number of veins in the marble
- **amplitude** controls how distorted the veins are

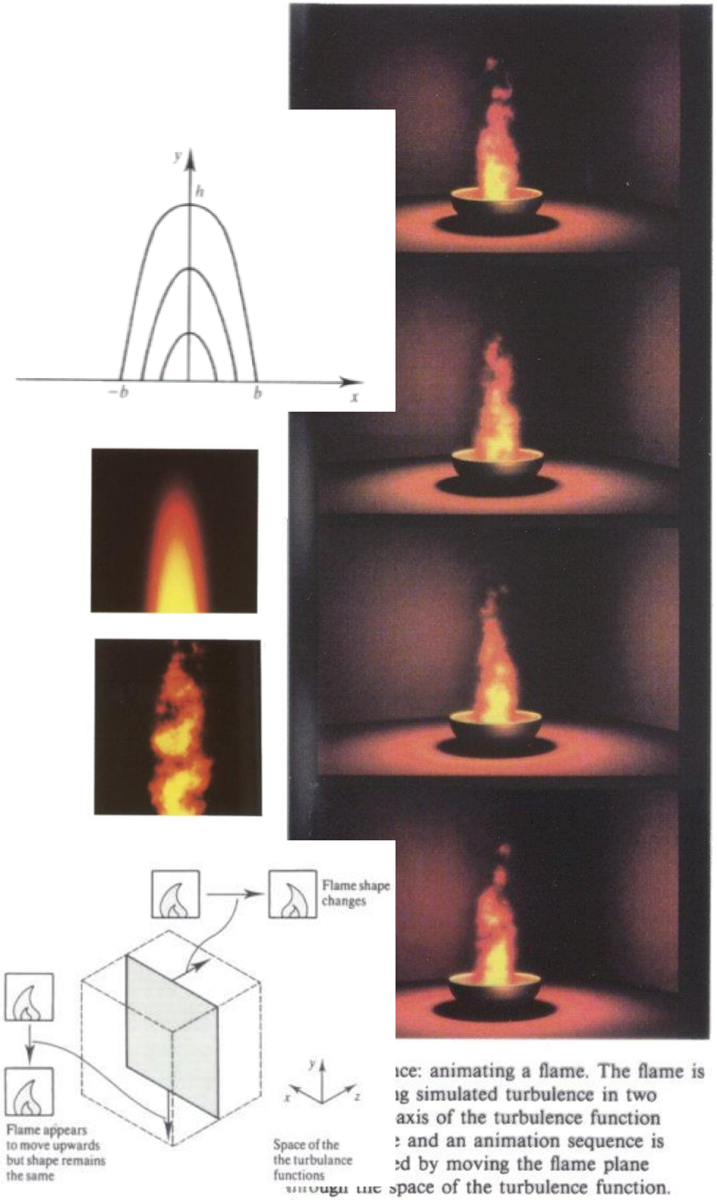
Image: ./Models/teapot_d.png

2.4 Noise and time

- A noise function (i.e. a noise grid) can also be used in conjunction with one or more 2D texture maps

- Example:Texture map of flame
  - or procedural model of flame's colours
- Use the noise function to add turbulence to the texture
  - u,v noise plane alters the texture
  - w axis acts as time
  - wary w over time will change the noise profile over time

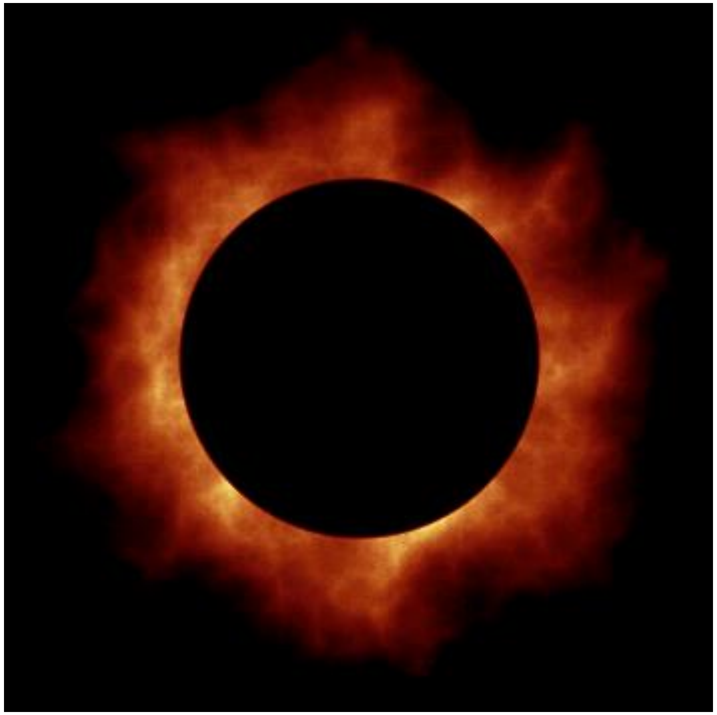
- (Alternative: play animation of real flame on a billboard polygon)

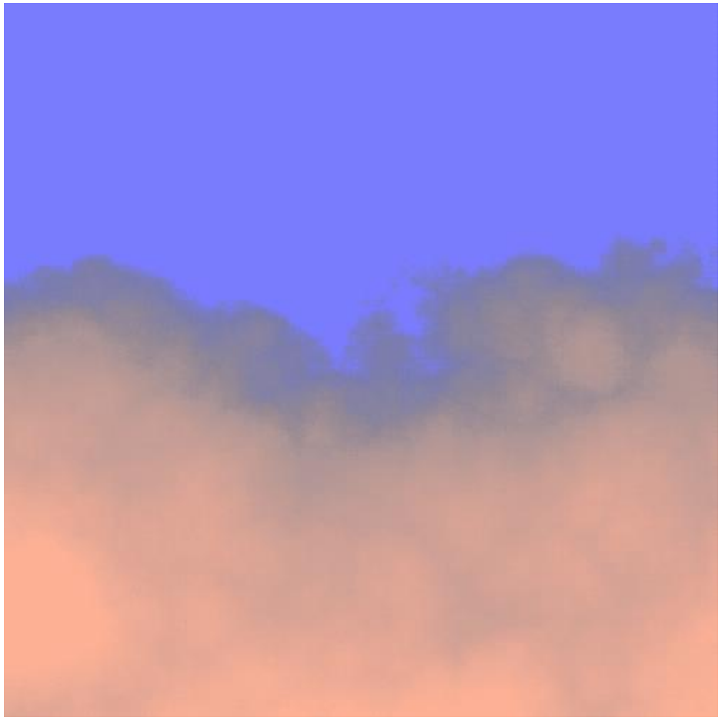

y
h
−b
b
x
Flame shape changes
Flame appears to move upwards but shape remains the same
y
x
z
Space of the the turbulance functions

ce: animating a flame. The flame is
g simulated turbulence in two
axis of the turbulence function
and an animation sequence is
d by moving the flame plane
through the space of the turbulence function.

Watt & Watt, 92

2.4 Noise and time

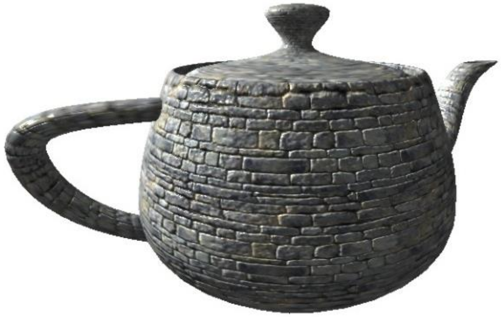




Ken Perlin

2.5 3D texture mapping versus 2D texture mapping

- 2D texture mapping
  - A 2D texture map can be compressed or stretched depending on the mapping, especially if the object is deforming
  - 2D texture mapping is awkward for non-trivial topologies
  - Advantage: easy to produce lots of 2D textures, e.g. photographs
- 3D texture mapping
  - Based on a function f(x,y,z), so can be generated to any level of detail for any topology
  - Extra work required if object is deforming
  - Can be difficult to design complex colour patterns – lots of trial and error



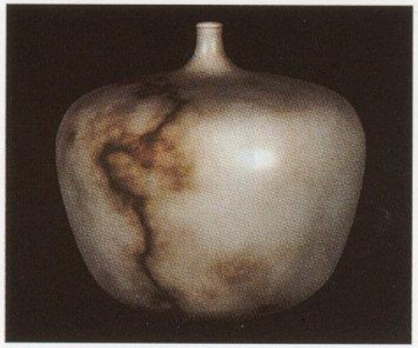
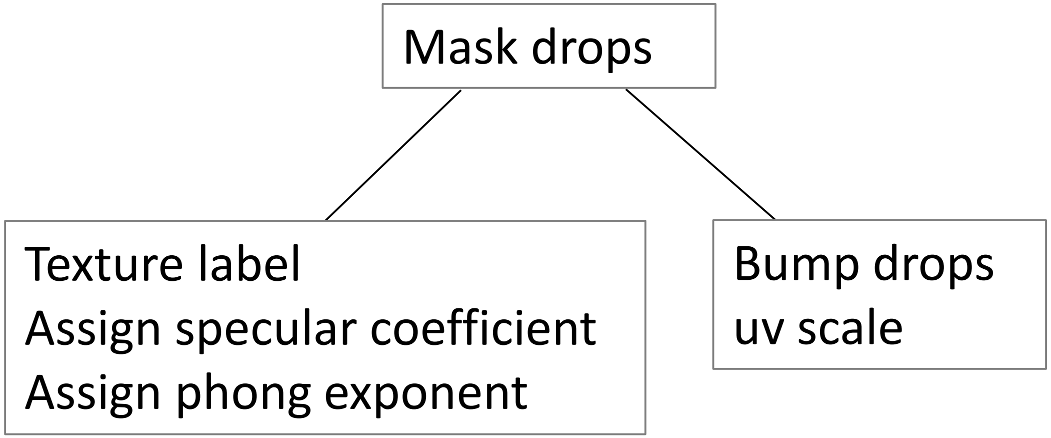


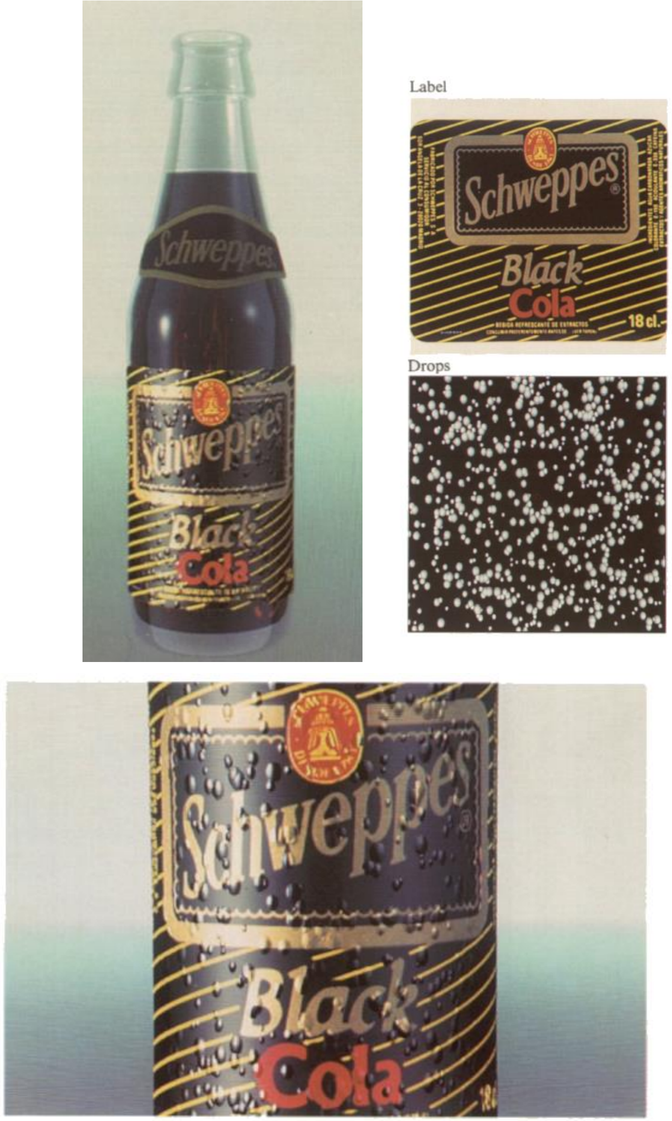
Figure 8.21 Imitating marble – the classic example of three-dimensional procedural texture.

Watt, 00

3. Combining texture maps

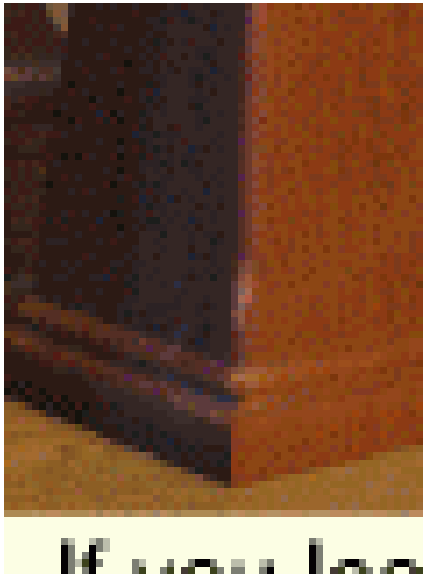
- Texture maps can be combined to produce more complex effects
- Based on the idea of shade trees
  - (Cook, 84), shading language, RenderMan (Pixar)

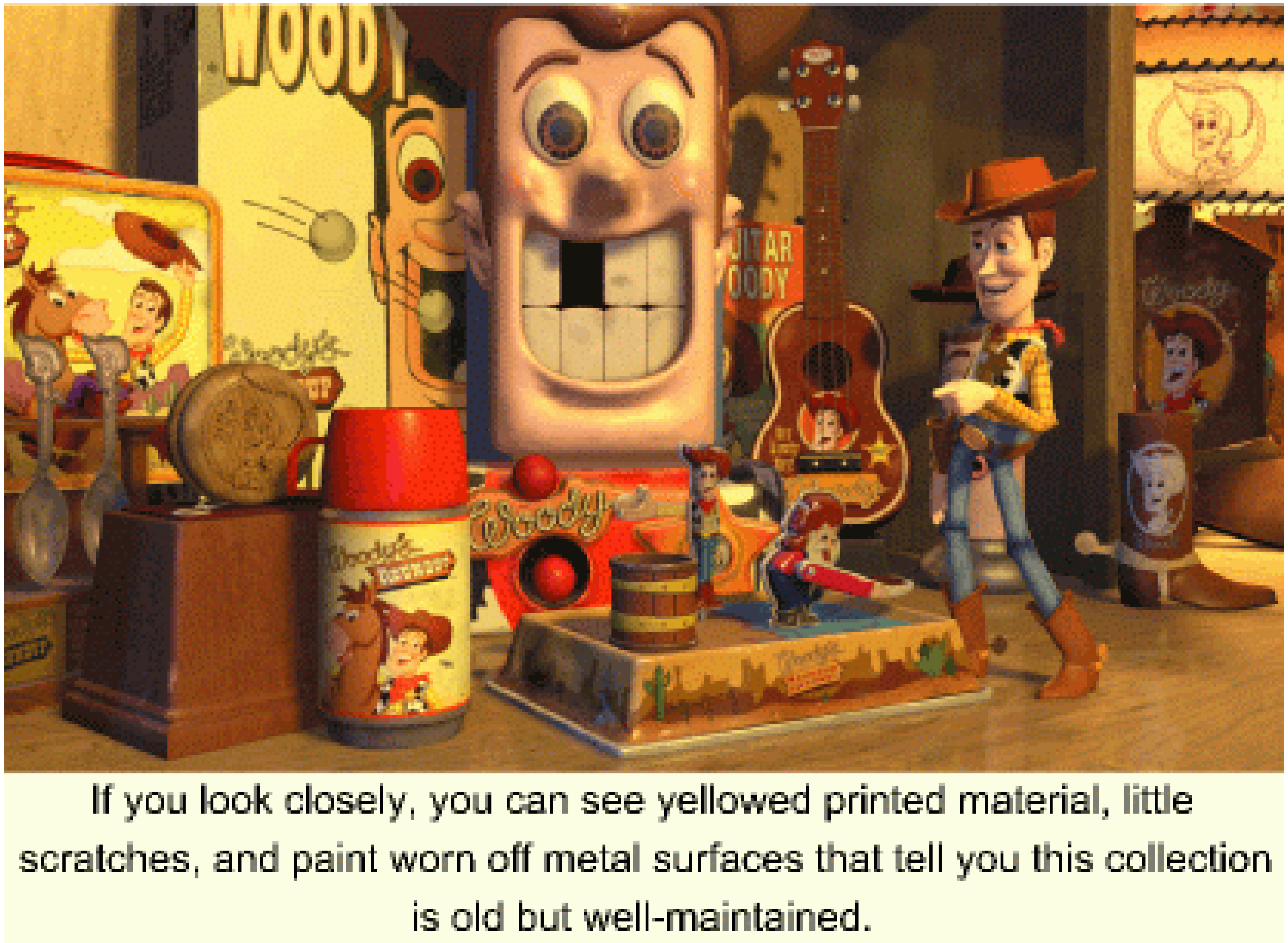




Watt & Watt, 92

4. Textures for ‘dirt’



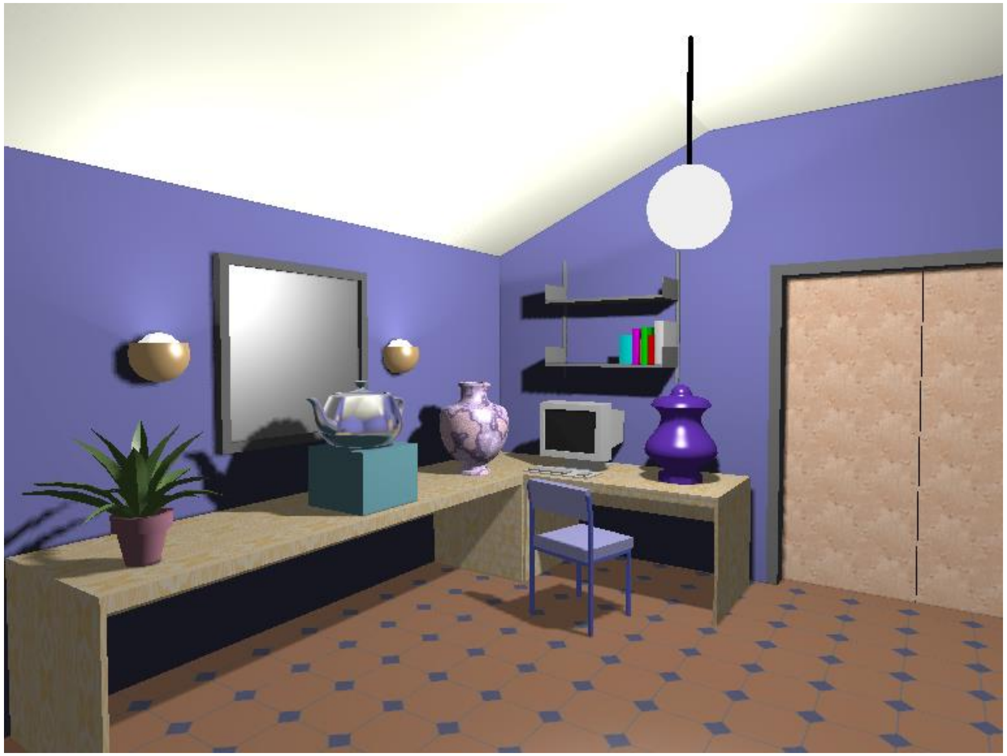


If you look closely, you can see yellowed printed material, little scratches, and paint worn off metal surfaces that tell you this collection is old but well-maintained.

Toy Story 2, Pixar, 99

5. Summary

- Most common *texture mapping* approach is to apply 2D texture maps to polygon mesh objects – texture colour mapping
- *Other approaches*:
  - environment mapping
  - bump mapping
  - normal mapping
  - displacement mapping
  - 3D texture mapping
- *Texture trees (based on shade trees)*: Organise combination of texturing effects



Watt, 00