

Figure 7.3
A simple experiment demonstrates functional specialization in the human brain. While viewing a colored scene, area V4 is activated (lower left). While viewing a moving scene area V5 shows the activation (lower right).¹

From S. Zeki, *Inner Vision*

work. Interesting in this regard is kinetic art, as well as Cubism. The development of the former is described in greater detail later.

← BACK

Sampling Texture Maps

When texture mapping it is rare that the screen-space sampling density matches the sampling density of the texture. Typically one of two things can occur:

Oversampling of the texture or Undersampling of the texture

In the case of oversampling we already know what to do... Interpolation (review on Antialiasing and Resampling). But, how do we handle undersampling?

← BACK

Slide 2 6.837 Fall 2002

How Bad Does it Look?

Let's take a look at what under-sampling looks like:

Notice how details in the texture, in particular the mortar between the bricks, tend to pop (disappear and reappear).

This popping is most noticeable around details (parts of the texture with a high-spatial frequency). This is indicative of aliasing (high-frequency details showing up in areas where we expect to see low frequencies).

← BACK

Slide 3 6.837 Fall 2002

We've Seen this Sort of Thing Before

Remember... Aliasing?

This was the phenomenon that occurred when we undersampled a signal. It caused certain high frequency features to appear as low frequencies.

To eliminate aliasing we had to either band limit (low pass filter) our input signal or sample it at a higher rate:

← BACK

Slide 4 6.837 Fall 2002

Spatial Filtering

In order to get the sort of images the we expect, we must *prefilter* the texture to remove the high frequencies that show up as artifacts in the final rendering. The prefiltering required in the undersampling case is basically a spatial integration over the extent of the sample.

We could perform this filtering while texture mapping (during rasterization), by keeping track of the area enclosed by sequential samples and performing the integration as required. However, this would be expensive. The most common solution to undersampling is to perform prefiltering prior to rendering.

← BACK

Slide 5 6.837 Fall 2002

MIP Mapping

MIP Mapping is one popular technique for precomputing and performing this prefiltering. MIP is an acronym for the latin phrase *multum in parvo*, which means "many in a small place". The technique was first described by Lance Williams. The basic idea is to construct a *pyramid of images* that are prefiltered and resampled at sampling frequencies that are binary fractions (1/2, 1/4, 1/8, etc) of the original image's sampling.

While rasterizing we compute the index of the decimated image that is sampled at a rate closest to the density of our desired sampling rate (rather than picking the closest one can in also interpolate between pyramid levels).

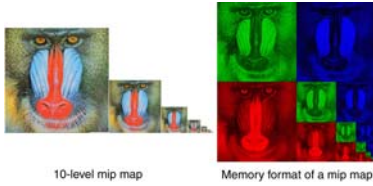
Computing this series of filtered images requires only a small fraction of additional storage over the original texture (How small of a fraction?).

← BACK

Slide 6 6.837 Fall 2002

Storing MIP Maps

One convenient method of storing a MIP map is shown below. (It also nicely illustrates the 1/3 overhead of maintaining the MIP map).



10-level mip map

Memory format of a mip map

We must make a few small modifications to our rasterizer to compute the MIP map level. Recall the equations that we derived for mapping screen-space interpolants to their 3-space equivalent.

$$u = u_1 + s(u_2 - u_1)$$

$$s = \frac{t w_2}{w_1 + t(w_2 - w_1)}$$

← BACK

Slide 7

6.837 Fall 2002

Next →

Finding the MIP level

What we'd like to find is the step size that a uniform step in screen-space causes in three-space, or, in other words how a screen-space change relates to a 3-space change. This sounds like the derivatives, (du/dt , dv/dt). They can be computed simply using the chain rule:

$$\frac{du}{dt} = \frac{du}{ds} \frac{ds}{dt} = (u_2 - u_1) \frac{w_1 w_2}{(w_1 + t(w_2 - w_1))^2}$$

$$\frac{dv}{dt} = (v_2 - v_1) \frac{w_1 w_2}{(w_1 + t(w_2 - w_1))^2}$$

Notice that the term being squared under the numerator is just the w plane equation that we are already computing. The remaining terms are constant for a given rasterization. Thus all we need to do to compute the derivative is a square the w accumulator and multiply it by a couple of constants.

Now, we know how a step in screen-space relates to a step in 3-space. So how do we translate this to an index into our MIP table?

← BACK

Slide 8

6.837 Fall 2002

Next →

MIP Indices

Actually, you have a choice of ways to translate this **gradient value** into a MIP level.

This also brings up one of the shortcomings of MIP mapping. MIP mapping assumes that both the u and v components of the texture index are undergoing a uniform scaling, while in fact the terms du/dt and dv/dt are relatively independent. Thus, we must make some sort of compromise. Two of the most common approaches are given below:

$$level = \log_2 \left(\sqrt{\left(\frac{du}{dt}\right)^2 + \left(\frac{dv}{dt}\right)^2} \right)$$

$$level = \log_2 \left(\max \left(\left| \frac{du}{dt} \right|, \left| \frac{dv}{dt} \right| \right) \right)$$



The differences between these level selection methods is illustrated by the accompanying figure.

← BACK

Slide 9

6.837 Fall 2002

Next →

Summed-Area Tables

There are other approaches to computing this prefiltering integration on the fly. One, which was introduced by Frank Crow is called a *summed-area table*. Basically, a summed-area table is a tabularized two-dimensional cumulative distribution function. Imagine having a 2-D table of numbers the cumulative distribution function could be found as shown below.

1	6	8	3
0	0	3	7
4	7	8	8
5	0	9	9

→

1	7	15	18
1	7	10	28
5	18	37	55
10	23	31	38

To find the sum of region contained in a box bounded by (x_p, y_p) and (x_r, y_r) :

$$T(x_r, y_r) - T(x_p, y_r) - T(x_r, y_p) + T(x_p, y_p)$$

This approach can be used to compute the integration of pixels that lie under a pixel by dividing the resulting sum by the area of the rectangle,

$$(y_r - y_p)(x_r - x_p)$$



With a little more work you can compute the area under any four-sided polygon (How?).

← BACK

Slide 10

6.837 Fall 2002

Next →

Summed-Area Tables

- How much storage does a summed-area table require?
- Does it require more or less work per pixel than a MIP map?
- What sort of low-pass filter does a summed-area table represent?
- What do you remember about this sort of filter?



No filtering



MIP mapping



Summed-Area Table

These nice images were originally grabbed from a link on the <http://www.gris.uni-luebingen.de> web page.

← BACK

Slide 11

6.837 Fall 2002

Next →

Anisotropic Mip-mapping

- What happens when the surface is tilted?
- Different mipmap for the 2 directions

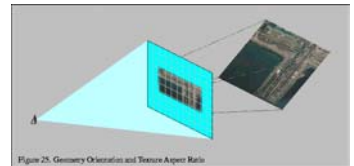


Figure 28. Geometry Orientation and Texture Aspect Ratio

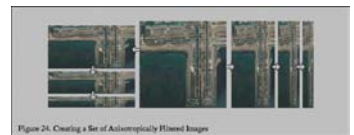


Figure 29. Creating a Set of Anisotropically Filtered Images

These nice images were originally grabbed from a link on the <http://www.esi.com/software/openal/advancedPS1/notes/notes37.html> web page.

← BACK

Slide 12

6.837 Fall 2002

Next →