Coverage, compositing and the alpha channel + reconstruction

Dinesh K. Pai

Textbook Chapter 16,17

Several slides courtesy of M. Kim

1

Compositing?

- Example of demo reel http://vimeo.com/72617082

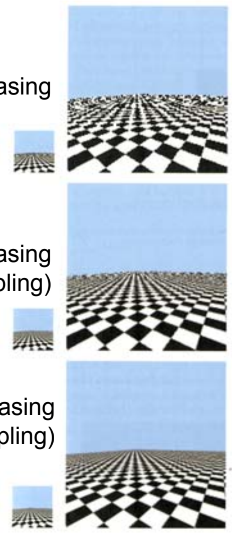
2

Recap: Aliasing and anti-aliasing

Aliasing

Anti-aliasing (multi-sampling)

Anti-aliasing (super-sampling)



These are polygons, not Textures!

← still have some problems

← good here

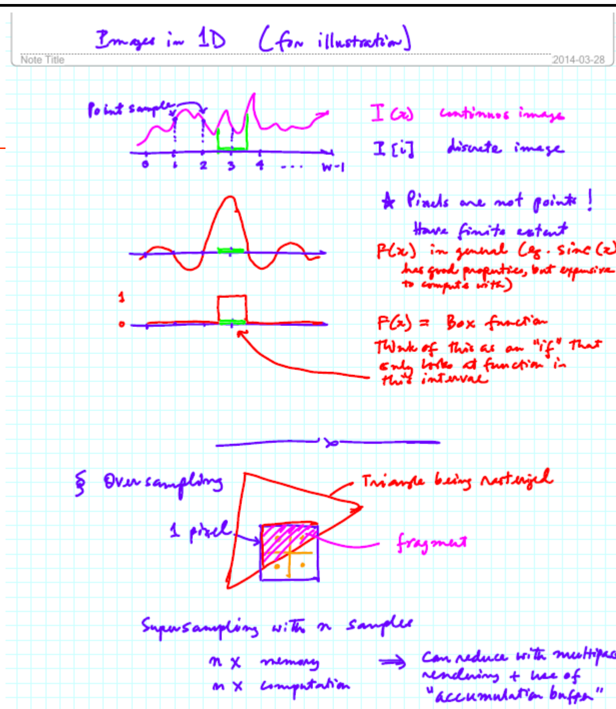
3

C³ Review: Sampling

- What causes aliasing?
  a) Too much detail per pixel
  b) Too many pixels per screen area
  c) Incorrectly loaded texture
  d) All of the above
  e) None of the above

C³ Review: Sampling

- How to avoid aliasing?
  a) Render to a display of higher resolution
  b) Calculate some average value of several texels to display for each pixel
  c) Change to a texture of higher resolution
  d) All of the above
  e) None of the above



6

Overview of Compositing

- Generalize idea of anti-aliasing to representing the "coverage" of each pixel by an object
- Essential for multi-pass rendering, requiring combination of images
- Historically, related to "matte"s in film, now done using the "alpha" channel in RGBA color images
- Importance increasing due to increasing availability of digital imagery
- Widely used: Visual Effects, "Sprites" in games, etc. Natively supported in most OS's for GUI

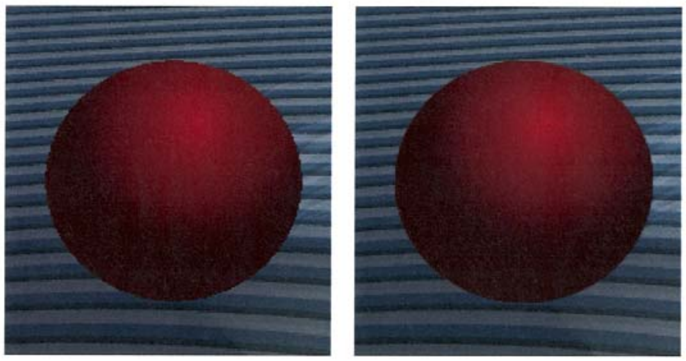
7

Image compositing

- Given two discrete images, a foreground, I^{f} , and background, I^{b}, that we want to combine into one image I^{c} .
- Simple: in composite, use foreground pixels where they are defined. Else use background pixels.
- This will give us a jagged boundary.
- Real image would have "boundary" pixels with blended colors.
- But this requires using "sub-pixel" information.

8

Image compositing



9

Alpha blending

- Associate with each pixel in each image layer, a value, $\alpha[\mathrm{i}][\mathrm{j}]$, that describes the overall *opacity* or *coverage* of the image layer at that pixel.
  - An alpha value of 1 represents a fully opaque/occupied pixel, while a value of 0 represents a fully transparent/empty one.
  - A fractional value represents a partially transparent (partially occupied) pixel.
- Alpha will be used during compositing.

10

Alpha definition

- More specifically, let $I(x,y)$ be a continuous image, and let $C(x,y)$ be a binary valued (x,y) *coverage function* over the continuous domain, with a value of 1 at any point where the image is "occupied" and 0 where it is not.
- Let us store in our discrete image the values:

$$\mathrm{I[i][j]} \leftarrow \iint_{\Omega_{i,j}} I(x,y)C(x,y)dxdy$$

$$\alpha\mathrm{[i][j]} \leftarrow \iint_{\Omega_{i,j}} C(x,y)\,dx\,dy$$

11

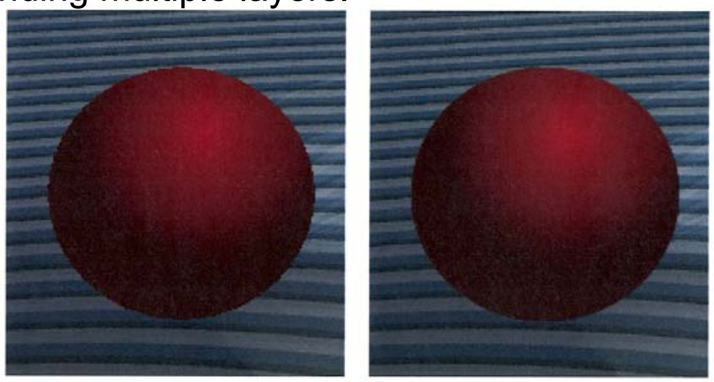
Over operation

Note: a technical term

- To compose $\mathrm{I^f[i][j]}$ over $\mathrm{I^b[i][j]}$, we compute the composite image colors, $\mathrm{I^c[i][j]}$, using
  $$\mathrm{I^c[i][j]} \leftarrow \mathrm{I^f[i][j]} + \mathrm{I^b[i][j]}\,(1 - \alpha^{\mathrm{f}}\mathrm{[i][j]})$$
  That is, the amount of observed background color at a pixel is proportional to the transparency of the foreground layer at that pixel.
- Likewise, alpha for the composite image can be computed as:
  $$\alpha^{\mathrm{c}}\mathrm{[i][j]} \leftarrow \alpha^{\mathrm{f}}\mathrm{[i][j]} + \alpha^{\mathrm{b}}\mathrm{[i][j]}\,(1 - \alpha^{\mathrm{f}}\mathrm{[i][j]})$$

Over operation

- If background is opaque, so the composite pixel is opaque.
- But we can model more general case as part of blending multiple layers.



13

Over properties

- This provides a reasonable approximation to the correctly rendered image.
- One can easily verify that the over operation is associative but not commutative. That is,

$$I^a \text{ over } (I^b \text{ over } I^c) = (I^a \text{ over } I^b) \text{ over } I^c$$

$$I^a \text{ over } I^b \neq I^b \text{ over } I^a$$

14

Chapter 17

RECONSTRUCTION

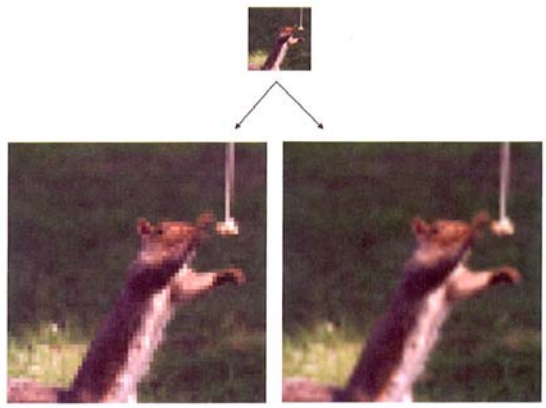
(DISCRETE → CONTINUOUS)

Reconstruction

- Given a discrete image I[i][j], how do we create a continuous image *I(x,y)*?
- Is central to resize images and to texture mapping.
  - How to get a texture colors that fall in between texels.
- This process is called *reconstruction*.
- We already know the key idea, from L24-L26: Interpolation! So we will go over this quickly.

Constant reconstruction

- The resulting continuous image is made up of little squares of constant color.
- Each pixel has an influence region of 1-by-1

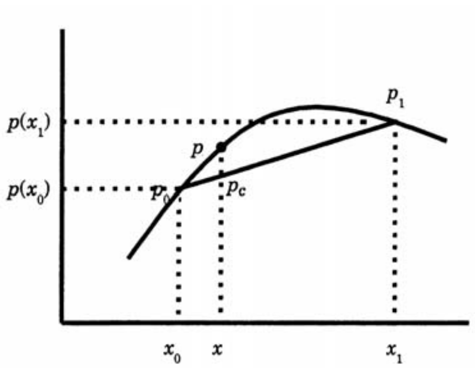


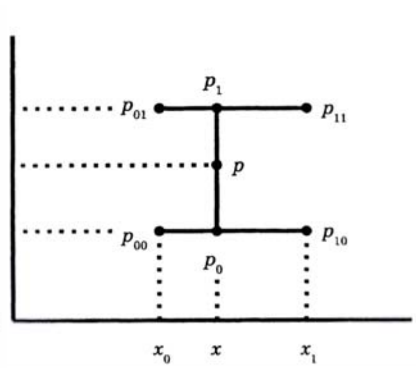
17

Linear and Bilinear interpolation

We already know how to interpolate in 1D

- Linear (1D) Bilinear (2D):





18

Bilinear reconstruction

- Can create a smoother looking reconstruction using *bilinear interpolation*.
- Bilinear interpolation is obtained by applying linear interpolation in both the horizontal and vertical directions.

```
color bilinearReconstruction(float x, float y, color
image[][]){
  int intx = (int) x;
  int inty = (int) y;
  float fracx = x - intx;
  float fracy = y - inty;

  color colorx1 = (1-fracx) * image[intx][inty] +
            (fracx) * image[intx+1][inty];
  color colorx2 = (1-fracx) * image[intx][inty+1] +
            (fracx) * image[intx+1][inty+1];
  color colorxy = (1-fracy) * colorx1 +
            (fracy) * colorx2;
  return(colorxy);
```

19

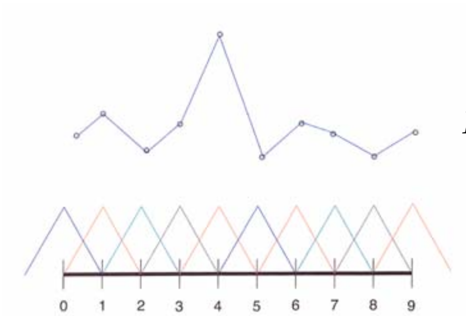
Bilinear properties

- At integer coordinates, we have *I(x,y)*=I[i][j]; the reconstructed continuous image *I* agrees with the discrete image I. => Interpolation
- In between integer coordinates, the color values are blended continuously.
- Each pixel influences, to a varying degree, each point within a 2-by-2 square region of the continuous image. => Local Support
- The horizontal/vertical ordering is irrelevant.
- Color over a square is bilinear function of (x,y).

20

Bilinear basis function

- Just as in L25,L26, we can think of interpolation as weighted combination of basis (or blending) functions B
- In 1D, we can define a univariate *hat function* $H_i(x)$



$$H_i(x) = \begin{cases} x - i + 1 & \text{for } i-1 < x < i \\ -x + i + 1 & \text{for } i < x < i+1 \\ 0 & \text{else} \end{cases}$$

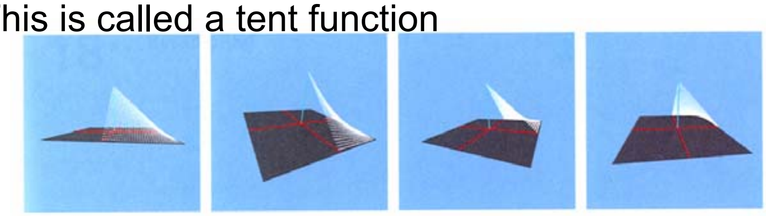
21

Bilinear basis function

- In 2D (bilinear function), let $T_{i,j}(x,y)$ be a bivariate function:

$$T_{i,j}(x,y) = H_i(x)H_j(y).$$

- This is called a tent function



- In constant reconstruction, $B_{i,j}(x,y)$ is a box function that is zero everywhere except for the unit square surrounding the coordinates (i,j), where it has constant value 1.

22