



CS770/870 Spring 2017

Anti-Aliasing



Cunningham, parts of Ch 3 and Ch 9
Watt, *3D Computer Graphics*, Addison-Wesley
Siggraph Tutorial notes

Carpenter, The A-buffer, an anti-aliasing hidden surface algorithm, *Siggraph '84*

Williams, Pyramidal parametrics, *Siggraph '83*.

Wikipedia (especially for the images)

Angel, 5e: Section 8.12

Angel, 6e: Section 7.12

02/26/17

CS770/870 Spring 2017 Bergeron

1

Preview

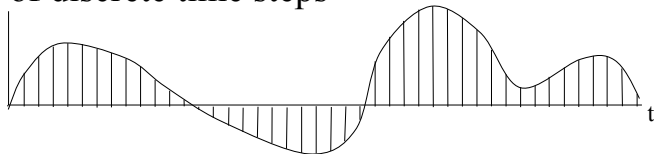
- Sampling
- Aliasing
- Antialiasing techniques
 - supersampling and post-filtering
 - non-uniform supersampling
 - pre-filtering
- Texture mapping issues
 - aliasing
 - mipmaps
- Temporal aliasing

CS770/870 Spring 2017 Bergeron

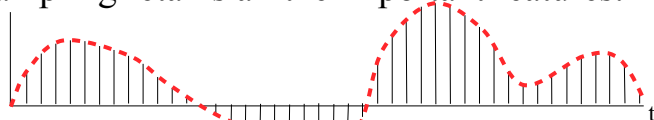
2

Sampling

- It is often necessary to *sample* a continuous signal at a series of discrete time steps



- This sampling produces a sampled signal that is a good representation of the original signal.
 - A reconstruction of a continuous signal from sampling retains all the important features:

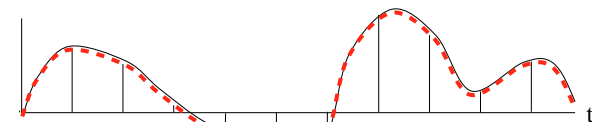


CS770/870 Spring 2017 Bergeron

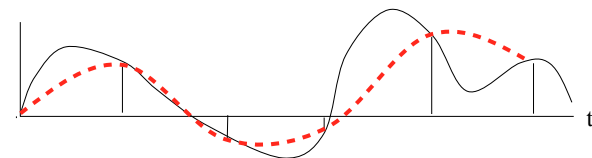
3

Poor Sampling

- Another sampling that is still ok:



- This one is not so good

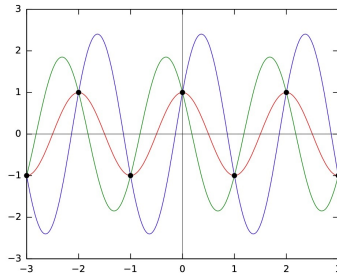


CS770/870 Spring 2017 Bergeron

4

Nyquist Limit

- Fundamental signal processing theorem:
 - An analog signal can be perfectly reconstructed from a set of samples if the sampling rate is greater than twice the highest frequency.
 - Otherwise, you get an *alias* of the signal

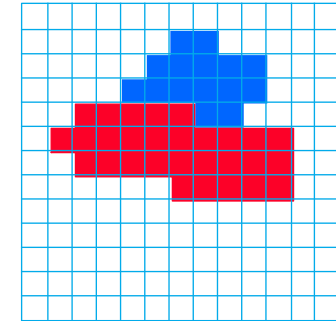
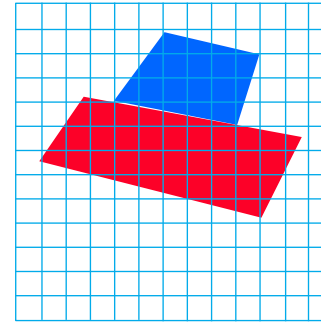


CS770/870 Spring 2017 Bergeron

5

Image Aliasing

- A pixel is a *discrete* sample of a *continuous* space
- Visual system is tuned to find discontinuities
- Results in *staircasing* or *jaggies* in static images
- Texture mapped surfaces are a particular problem



CS770/870 Spring 2017 Bergeron

6

Antialiasing techniques

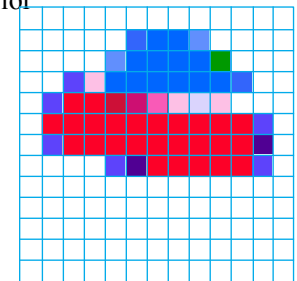
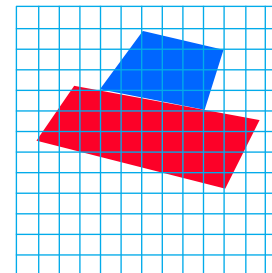
- Supersampling and post-filtering
 - create an image at a higher resolution than the final one
 - pass the higher resolution image through a filter that averages a set of high resolution pixels to get a “real” pixel
- Non-uniform sampling
 - supersampling and post-filtering are more effective if the samples are not uniformly distributed
- Pre-filtering
 - “infinite” samples per pixel (the most “correct” approach)

CS770/870 Spring 2017 Bergeron

7

Anti-Aliasing Techniques

- **Supersampling** — compute image at higher resolution and apply a convolution filter to computed pixels to get image pixel (post-filtering)
- **Image filtering (post-filtering)** — “blur” pixels after rendering
- **Scene filtering (pre-filtering)** — remove high frequencies before rendering
 - these are the only “correct” methods
 - example: compute % of each pixel covered by visible object and contribute that % of color to the pixel color

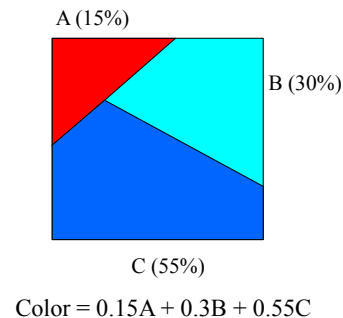


CS770/870 Spring 2017 Bergeron

8

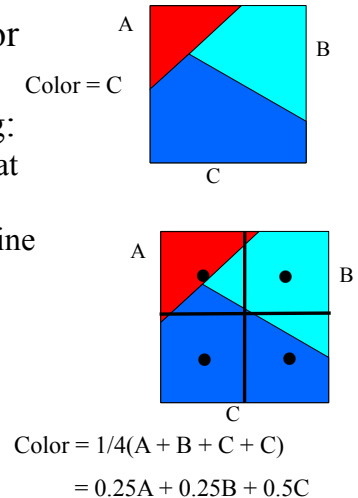
Image Aliasing Problem

- Consider a sample pixel that is partially covered by 3 objects
- What *should* the color be?
 - A weighted blend of the 3 object colors where the weights are the % of the area covered by each color
 - This “blurs” the image in areas of high frequency change; visual system sharpens the blurry edges
 - This is *pre-filtering*: remove high frequencies from image



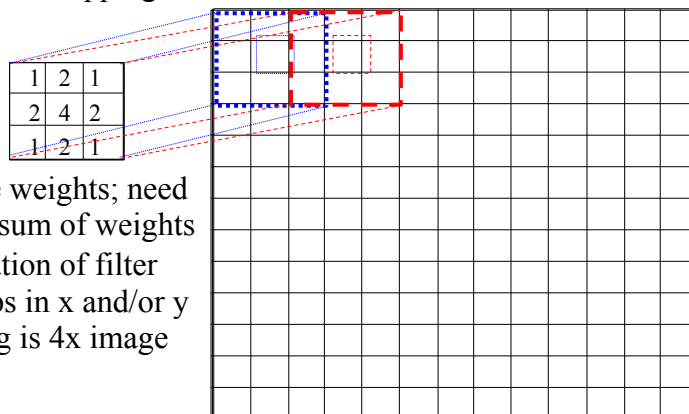
Supersampling and Post-filtering

- Simple sampling: sample color as center of pixel
- Supersampling / Post-filtering:
 - Create 4 subpixels and sample at the center of each
 - Average the 4 results to determine pixel color
 - This is a 2x2 *box* filter



Higher Order Convolution

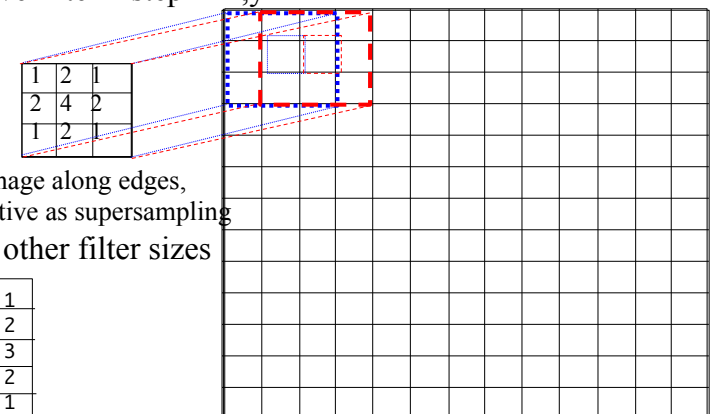
- Assume sampling resolution is 4x image resolution
- Can use 3x3 overlapping convolution filter



- Numbers are weights; need to divide by sum of weights
- Each application of filter moves 2 steps in x and/or y -- if sampling is 4x image

More Convolution

- Can apply a convolution filter to an image that is not super-sampled; move filter 1 step in x,y



- Blurs the image along edges, not as effective as supersampling
- Can also use other filter sizes

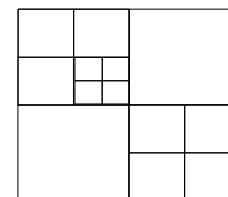
1	2	3	2	1
2	4	6	4	2
3	6	9	6	3
2	4	6	4	2
1	2	3	2	1

Non-uniform sampling

- Uniform supersampling reduces the aliasing, but has definite limitations
- Can do better with various kinds of non-uniform sampling that can be (roughly) categorized into 2 groups:
 - non-uniform subdivision
 - subdivide a display region recursively to get high sampling where needed
 - stochastic sampling
 - include a randomization aspect to positioning of samples points

Non-uniform Subdivision

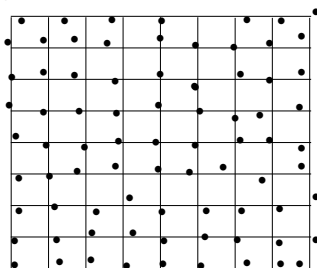
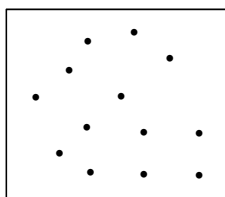
- An early extension of the first ray tracing algorithm checked for small objects whose projection onto the view plane is smaller than a pixel
 - if one is found, the pixel is divided into 4 subpixels and ray tracing continues for each of the subpixels
 - the subpixels in turn can be subdivided
- Similar *adaptive* subdivision has been proposed for many algorithms



Contribution of a color is weighted inversely based on the size of the “subpixel”.

Stochastic Sampling

- Stochastic placement of samples
 - randomly distribute sample positions
 - transforms “aliases” into “noise”
 - coordinates well with Poisson distribution of receptors in the fovea of the eye
 - we tolerate noise much better than aliasing
- Jitter
 - a form of stochastic sampling
 - start with a regular grid, but randomly perturb each point location
- Can jitter nonuniform subdivision



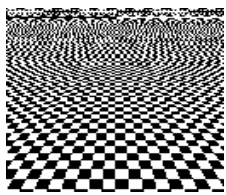
A-Buffer

- The first (1984) significant and useful pre-filtering was Carpenter’s A-Buffer, used in Pixar’s REYES system
- Extends z-Buffer / frame buffer:
 - $z[i,j] \geq 0$: frame[i,j] is color of only object that covers pixel
 - $z[i,j] < 0$: frame[i,j] is pointer to list of fragments in pixel
- A *fragment* is an approximation to the portion of a polygon’s intersection with the pixel
 - key component is a 32 bit integer representing fragment’s coverage of a 4x8 partitioning of the pixel: 1 bit per subpixel
 - Very expensive, very effective

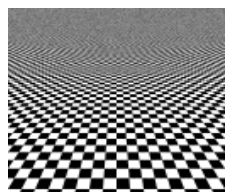
1	1	1	0	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	0	0	0	0
1	1	1	1	1	1	0	0

Texture Mapping

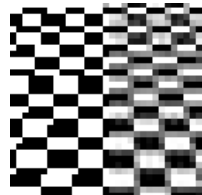
- Texture maps raise particularly challenging aliasing issues since they are often very high frequency signals
- They also create headaches when viewed at different distances



a. Aliased image



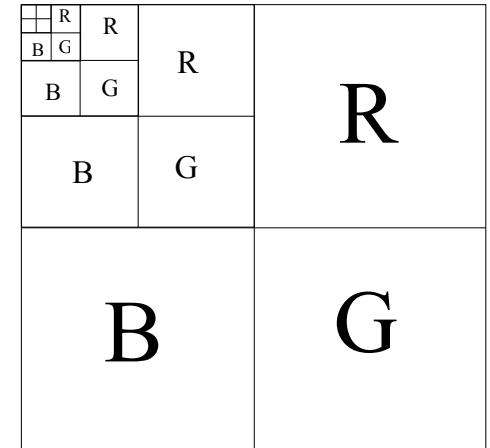
b. Anti-aliased image



c. Zoom: a left, b right

Mipmaps

- Williams (1983) developed “mip” maps (Latin: *multum in parvo*, or “many things in a small space”)
- Mipmaps are an efficient way to represent RGB textures at multiple resolutions
- All resolutions cost only 25% more space than highest resolution



Mipmap Access

- Access to mipmap is by (u,v,D) , where (u,v) represent the parametric index into the texture and D represents the *depth* which corresponds to *level of detail*.
 - the initial (u,v) are integer indexes into the highest resolution
 - once D is determined, the appropriate revised index is simply a binary shift of the original (u,v) by D bits
 - D is determined by looking at the size of the highest resolution texel in image space
 - Clever interpolation between resolution levels means can effectively use a D that is between resolutions
- Mipmaps are standard features of modern graphics cards

Temporal Aliasing

- Animation creates more aliasing issues
 - flicker*: Very small bright objects can appear and disappear
 - wheel reversal*: fast objects can move too fast for proper sampling
 - scintillation*: boundary color decisions can be slightly different from frame to frame; minor differences can lead to rippling along edges, which our visual system is very good at perceiving

Review

- Sampling
- Aliasing
- Antialiasing techniques
 - supersampling and post-filtering
 - non-uniform supersampling
 - pre-filtering
- Texture mapping issues
 - aliasing
 - mipmaps
- Temporal aliasing