Introduction

• Fixed compression methods such as DXT are becoming antiquated
• Distribution of data “density” becoming less uniform as complexity increases
  – Lighting is a good example
• Would really like to focus data where it matters most
• Wavelets can help with this
What are wavelets?

• Wavelets are mathematical basis functions formed from scaled and translated copies of a single set of functions

• The advantage is their ability to localize functions in both frequency and space
  – As compared to spherical harmonics for instance, which is just frequency

• There are many different wavelet bases with varying properties, but we focus on “2D Haar”
Why 2D (nonstandard) Haar?

• We will focus exclusively on 2D nonstandard Haar wavelets
• It’s the simplest basis set, and is orthogonal
  – Orthogonality important for lighting
  – In general this simplicity means more basis terms are required than for more sophisticated wavelets
  – You can also get blocky-like artifacts, although this is less of a problem for “smoothing” operations such as integration
• What do these wavelets look like?
2D Haar basis functions

- Three basis wavelet functions, plus ‘solid’ scaling function
- White represents +1, black represents -1
Wavelets in action

- Rescaled wavelets are used to refine data at resolution $N \times N$ down to resolution $2N \times 2N$
- This is continued until desired accuracy is obtained
- NLO performed for compression
The wavelet advantage

• Local coverage allows windowed changes
  – Compared to spherical harmonics, which have global cover
  – This means that local changes only involve local bases

• Truly variable compression
  – Can focus efforts where it counts
  – Certain parts of the data can be represented near-losslessly

• Many other advantages
  – E.G. MIP-MAP chains are not required for image decompression, etc.
Usage examples

• Real-time shader image decompression
• Very accurate lighting and GI
• Displacement maps over large areas
• Easy dynamic texture packing
• Geometry representations
• Static shadow maps
• …many more!
Talk focus

• Real-time shader image decompression
  – Real-time on the Xbox 360™ GPU
  – 500Hz+ for full-screen 720p monochrome decompression

• Global illumination
  – Real-time wavelet double product integration on the Xbox 360™ GPU
  – Methods for BRDF representation, including a sampling approach
Real-time image decompression

- Firstly, image is broken up into 16 X 16 texel blocks
  - Because we want good texture cache performance, and this helps guarantee that
  - We also want decompression performance to be independent of image size
  - Other benefits, e.g. can unroll traversal loop in microcode shader for performance gain, etc.

- Each 16 X 16 sub-block is compressed into a wavelet tree
Real-time image decompression

- We also require a texture at 1/16\textsuperscript{th} the resolution to store a scaling coefficient and an offset
  - Offset used to point to start of wavelet tree for that block
- This wavelet tree is stored breadth-first in a line texture
Real-time image decompression

- So given a \((u,v)\) coordinate, we traverse the appropriate sub-tree in the pixel shader for the final value
  - See course notes for more details and an optimized microcode shader for this process

- Several advantages to this approach
  - MIP-MAP chain not required
  - Intermediate memory not required
  - Dynamic predication works well
  - Genuine variable compression
Real-time image decompression

- Uncompressed with MIP-MAPS ~2MB @ 1578fps
Real-time image decompression

- Wavelet compressed, cut-off 0.05, 249KB @ 579fps
Real-time image decompression

- 622.23 fps
- Real-time GPU wavelet decompression
- Total memory usage: 156.93 KB
- Cut-off: 0.08125

- Wavelet compressed, cut-off 0.08, 157KB @ 622fps
Real-time image decompression

- Notice how areas of high contrast have their detail preserved, and areas of lower contrast are “smoothed out”
- Can use a different notion of importance
Real-time image decompression

1024.59 fps
Real-time GPU wavelet decompression
Total memory usage 248.35 MB
Cutoff 0.050000
Real-time image decompression

- Wavelet compression of SH coefficients
- 94-98% coefficient pruning in this case
Global Illumination

• Another good application is relighting
• Work by [NRH03] and [NRH04] in particular (see course notes)
• Double and triple product integration
• Wavelets have the ability to recover high-frequency environmental specular responses much more effectively than SH
Global Illumination

- First application: Double-product integration
- We represent both the transfer function (with cosine) and the environment as wavelet trees in a texture
- Only need to traverse intersection of both trees for sparse dot product evaluation
- So GPU traversal needs to be able to “jump over” whole branches of each tree
Global Illumination

• How to do this?
  – If a node has a child, store linear offset to sibling or ancestor (could be root)
  – Then we can jump over the whole child branch is the other tree has no children at that point
  – This saves a lot of time

• Performance is a function of the size of the intersection between both trees
  – See course notes for more details and microcode shader
Global Illumination

- Red area light on left, Grace Cathedral on right
  - Notice quality of shadow on left
Global Illumination

• One real difficulty with wavelets is rotation
  – Need to be able to freely orient BRDF around normal and reflected light direction, where normal is sampled from high-resolution normal map
  – [NRH03] stores a heavily compressed “BRDF database”
  – What about a point-sampling approach?
    • Depends on variance of BRDF over “mutual leaf”
    • Can use iterative sampling. See course notes.
Global Illumination

- Triple-product integration approximation using double-product integration and point sampling
- HEALPix a good partitioning method for this
Global Illumination

• (Videos)
Conclusion

• Wavelets are a very powerful method of data compression
• They are well suited to representing data with uneven complexity
• Even on current hardware, real-time application is possible
• “Shader tree traversal” approaches likely to become popular in future