The Next Generation of Open Video Codecs

Dr. Timothy B. Terriberry
Outline

- Introduction
- Features
- Technology
- Conclusion
Introduction

- Codecs typically compete for several markets
  - Streaming
    - Low resolution/bit-rates
    - Strict buffering/rate/latency requirements
  - Broadcast
    - Fidelity, throughput, latency more important than raw compression
  - Desktop
    - Compression is King
    - High computational complexity
Introduction

- MPEG includes many features targeting all of these; most users don’t care about most features (shape-adaptive coding, object-based coding, global motion compensation, etc.)

- Which users should we be pursuing?

- What features are most important to those users?
Outline

- Introduction
- **Features**
- Technology
- Conclusion
Error Recovery

- Theora’s main application, and an important application of Dirac, is low-bitrate streaming
- Yet there are no explicit or implicit provisions for dealing with packet loss
  - The Ogg unit of error correction and recovery is the page, yet the API deals with only complete packets
    - Packets (especially key frames) spanning multiple pages will be completely lost if any page is lost
  - Dirac removes buffered reference frames by explicitly retiring them
    - If a packet is lost, the wrong frames could be retired
Error Recovery: Tools

- Data Partitioning (into multiple packets)
  - Include enough information to decode part of a frame when some packets are lost
    - 0.5 to 2 dB loss in quality, depending on sequence/bitrate
- Unequal Error Protection
  - Simple: Send header (and motion) data twice
  - Or complex: Use channel codes with varying amounts of redundancy
    - Wireless errors tend to be bursty, and software layers discard complete network packets, limiting the utility of these complex approaches
Error Recovery: Tools

- Recovering from uncorrectable errors
  - INTRA frames: too expensive, large latency
- Periodic INTRA refresh (used by MPEG)
  - Refresh individual blocks (not whole frame) with INTRA data to spread out cost (improve latency)
- Auto-Regressive Predictor (used by Speex)
  - Weight motion compensated predictor by 0.8-0.9
  - Less than 5% error after 14 (0.8) to 29 (0.9) frames
  - Adds 20% to 200% overhead, depending on motion
Interlacing

- Primary challenge with interlacing is patents
- Theora has no interlacing support
- Dirac supports interlacing, but must decide at the sequence level if fields are coded together or separately
  - Choice depends on amount of motion, which can vary from frame-to-frame, or even within a frame
- Real sequences can switch from interlace to progressive, change parity, encode 24 fps content at 60 fields p.s. with 3:2 pulldown, etc.
Interlacing

- Picture-level adaptive frame/field coding performs as well as the best single choice for short sequences
  - Typically 20-40% fewer bits than the other choice
- Performs significantly better than any single choice on longer, hybrid sequences (up to 1 dB or more)
- Block-level adaptive frame/field coding gives another 0.25-0.8 dB gain
Outline

- Introduction
- Features
- Technology
- Conclusion
Transform/Subband Coding

- 3 main approaches
  - DCT
    - Used by most popular codecs, including Theora
  - Wavelets
    - Used by Dirac
    - Power comes from extracting large scale correlations, but Dirac limits the transform depth
    - Relatively computationally expensive
  - Overlapped Transforms (Tran et al.)
    - Coding gain as good as wavelets
    - Much less computationally expensive (40% fewer muls)
Overlapped Transforms

- Similar to the MDCT used in mp3/Vorbis
  - Completely eliminates blocking artifacts
- Pre-filter → DCT → code → iDCT → Post-filter
  - Pre-filter “decorrelates” block edges, introducing artificial blockiness cheaply represented by the DCT
  - Post filter similar to current post-processing filters, but invertible, e.g., it is the inverse of the pre-filter
- Wavelets slightly better on smooth input (0.5 dB)
- Much better on textured or low-resolution (less smooth) input (up to 1.5 dB)
Hybrid Transforms

- Wavelets expensive because of cache coherency

- However, DC subband requires 1/256 the data (for 16x16 blocks), which often fits in cache
  - A hybrid approach could use an overlapped transform on the full frame, followed by wavelets (or wavelet packets, joint adaptive space-frequency bases, etc.) on the DC subband
  - DC subband not as smooth as full resolution image, cheaper Haar basis should perform better than 9/7
Motion Compensation

- How does motion compensation interact with overlapped transforms?
  - For an \( N \times 2N \) transform, Tran searches for \( 2N \times 2N \) blocks: 4 times the memory bandwidth
  - Dan Miller had a better idea: Compensate for motion as normal, then pre-filter the predicted image
    - Much cheaper than Tran’s approach
    - Allows arbitrary motion compensation approaches
    - Multiple reference frames more complex
      - Dirac’s approach (2 references for whole image) still easy
      - H.264’s approach (n references, often for small areas) not so much
Blocking Artifacts

- Wavelets and OTs have no inherent blocking, therefore neither should MC

- Two main approaches:
  - OBMC: Overlapped Block Motion Compensation (blend multiple predicted images)
    - Cost: 3-15 multiplies, 4 memory accesses/pixel
    - Better for uncertain motion, easier to estimate
  - CGI: Control Grid Interpolation (blend multiple motion vectors to make a single prediction)
    - Cost: 3 multiplies, 1 memory access/pixel
    - Preserves more fine details, very difficult to estimate
Adaptive Methods

• Switching between OBMC and CGI should give in the neighborhood of 0.5 dB over either alone
  – Based on preliminary work by Heising et al. in a wavelet coder

• But switching introduces blocking artifacts

• My own recent research
  – Label the *edges* of each block with an interpolation type
  – Use interpolation formulas that smoothly approach the correct behavior on each edge
Adaptive Methods

- **VVVV**  
  \[ I(w_0 \vec{m}_0 + w_1 \vec{m}_1 + w_2 \vec{m}_2 + w_3 \vec{m}_3) \]

- **BVVV**  
  \[ I((w_0 + w_1) \vec{m}_0 + w_2 \vec{m}_2 + w_3 \vec{m}_3) \cdot s_0 + \]
  \[ I((w_0 + w_1) \vec{m}_1 + w_2 \vec{m}_2 + w_3 \vec{m}_3) \cdot s_1 + \]
  \[ I(w_0 \vec{m}_0 + w_1 \vec{m}_1 + w_2 \vec{m}_2 + w_3 \vec{m}_3) \cdot (s_2 + s_3) \]

- **BVBV**  
  \[ I((w_0 + w_1) \vec{m}_0 + (w_2 + w_3) \vec{m}_3) \cdot (s_0 + s_3) + \]
  \[ I((w_0 + w_1) \vec{m}_1 + (w_2 + w_3) \vec{m}_2) \cdot (s_1 + s_2) \]

- **VVBB**  
  \[ I((1-w_1) \vec{m}_0 + w_1 \vec{m}_1) \cdot s_0 + I(w_1 \vec{m}_1 + (1-w_1) \vec{m}_2) \cdot s_2 + \]
  \[ I(w_0 \vec{m}_0 + w_1 \vec{m}_1 + w_2 \vec{m}_2 + w_3 \vec{m}_3) \cdot s_1 + I(\vec{m}_3) \cdot s_3 \]

- **VBBB**  
  \[ I((1-w_1) \vec{m}_0 + w_1 \vec{m}_1) \cdot s_0 + I(\vec{m}_2) \cdot s_2 + \]
  \[ I(w_0 \vec{m}_0 + (1-w_0) \vec{m}_1) \cdot s_1 + I(\vec{m}_3) \cdot s_3 \]

- **BBBB**  
  \[ I(\vec{m}_0) \cdot s_0 + I(\vec{m}_1) \cdot s_1 + I(\vec{m}_2) \cdot s_2 + I(\vec{m}_3) \cdot s_3 \]

\( w_i \): bilinear vector weights  
\( s_j \): bilinear image weights
Adaptive Subdivision

- Sends more motion vectors in areas that need it (near object boundaries, etc.)
  - Not as good as content-adaptive meshes (arbitrary shape), but vastly simpler to compute
- Large gains (0.9 to 1.7 dB) at low bitrates
- Blocking artifacts
  - Dirac subdivides to smallest level and copies MVs
    - Lots of setup overhead for smaller blocks
    - Redundant computations for adjacent blocks with same MV
    - Only works for OBMC, not CGI
Adaptive Subdivision

- Slight modifications to the previous formulas allow artifact-free subdivision in a 4-8 mesh
  - Adjacent blocks differ only by 1 level of subdivision
  - Fine-grained control (MV rate doubles each level)
  - Efficient R-D optimization methods (Balmelli 2001)
    - Developed for compressing triangle mesh/terrain data
  - Larger interpolation kernels, less setup overhead, fewer redundant calculations
Arithmetic Encoding

- Main patent has expired
- Other patents on fast implementations and hardware implementations still apply
  - These can be coded around
- Multiply-free technique of Stuiver and Moffat (1998) believed to be unpatented
  - Allows the usage of ordinary frequency counts for probability modeling
  - Not restricted to binary alphabets
  - Can be adapted to range coding (byte-oriented) instead of being bit-oriented
Context Modeling

• Trade-offs for complex models
  – Makes stats closer to stationary, zero-order
  – Context dilution
    • Overhead of $O(\log n)$ bits per context means lots of small $n$’s can perform much worse than one large one
  – Memory overhead
    • Binary contexts need $O(m)$ words to code an $m$ bit symbol (but have more contexts)
    • Larger alphabets need $O(2^m)$ words

• Xiph approach: Put it in the header
Context Modeling

- Enables a unique feature: zero-probability symbols

  - Bitstream can contain features which the encoder can disable for no bitrate penalty
    - Block-level quantizers, their range, prediction modes, etc.
    - Allowing several codebooks allows arbitrary features to be enabled or disabled on a frame-by-frame basis

  - Features can be added to the bitstream without breaking backwards-compatibility of old streams
    - Simply add a new symbol to a context, which will be assumed to have zero probability in existing files
Outline

- Introduction
- Features
- Technology
- Conclusion
Conclusion

• Most of these ideas are still in the experimental stage
  - No way to know now how well they will really work (though we can guess)
• Lots of other simple ideas also beneficial
  - Zig-zag scan starting from the end, not beginning
  - Color channels uncorrelated in INTRA blocks, but not uncorrelated in INTER blocks
• With a dedicated developer, could have running code in 6 months; without one: years