

# Introduction to Video Compression


*Insight, Analysis, and Advice on Signal Processing Technology*



## Introduction to Video Compression (ESC-368)

Jeff Bier  
BDTI  
Oakland, California USA  
+1 (510) 451-1800  
[info@BDTI.com](mailto:info@BDTI.com)  
<http://www.BDTI.com>

© 2008 Berkeley Design Technology, Inc.



## Outline

- Motivation and scope**
- Still-image compression techniques
- Motion estimation and compensation
- Reducing artifacts
- Color conversion
- Comparing performance and efficiency
- Conclusions

© 2008 BDTI

INSIGHT • ANALYSIS • ADVICE  
ON SIGNAL PROCESSING TECHNOLOGY

2

© 2008 BDTI



## Motivation and Scope

Consumer video products increasingly rely on video compression

- DVDs, digital TV, personal video recorders, Internet video, multimedia jukeboxes, video-capable cell phones and PDAs, camcorders...

Video product developers need to understand the operation of video “codecs”

- To select codecs, processors, software modules
- To optimize software

This presentation covers:

- Operation of video codecs and post-processing
- Computational and memory demands of key codec and post-processing components



## Outline

Motivation and scope

**Still-image compression techniques**

Motion estimation and compensation

Reducing artifacts

Color conversion

Comparing performance and efficiency

Conclusions



## Still-Image Compression

### Still-image compression

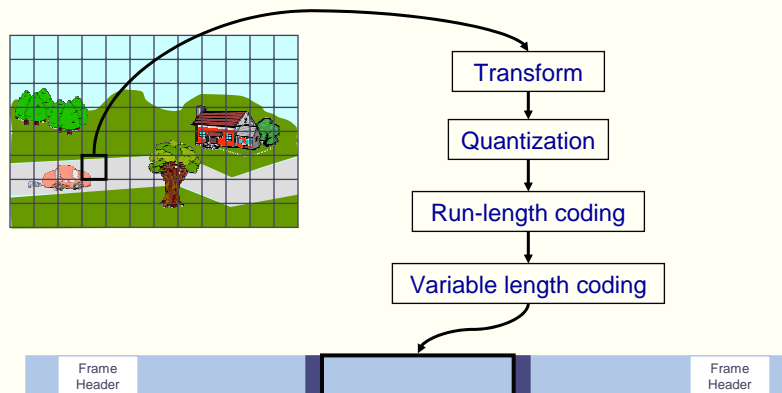
- Still-image techniques provide a basis for video compression
- Video can be compressed using still-image compression individually on each frame
- E.g., “Motion JPEG” or MJPEG

### But modern video codecs go well beyond this

- Start with still-image compression techniques
- Add motion estimation/compensation
  - Takes advantage of similarities between frames in a video sequence

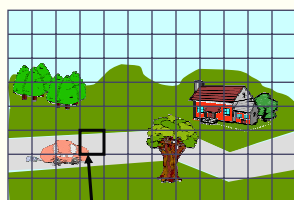


## Still-Image Compression



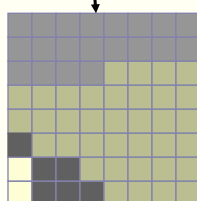
### Typical Still-Image Compression Data Flow

## Block Transform: 8x8 DCT



Y values

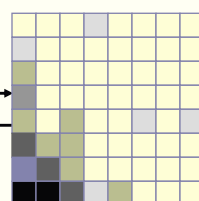
- 8x8 DCT blocks applied on Y, U, and V planes individually
- The energy is concentrated in the low frequencies
- Perceptual information also concentrated in low frequencies



Spatial domain

8x8 DCT

8x8 IDCT



Frequency domain

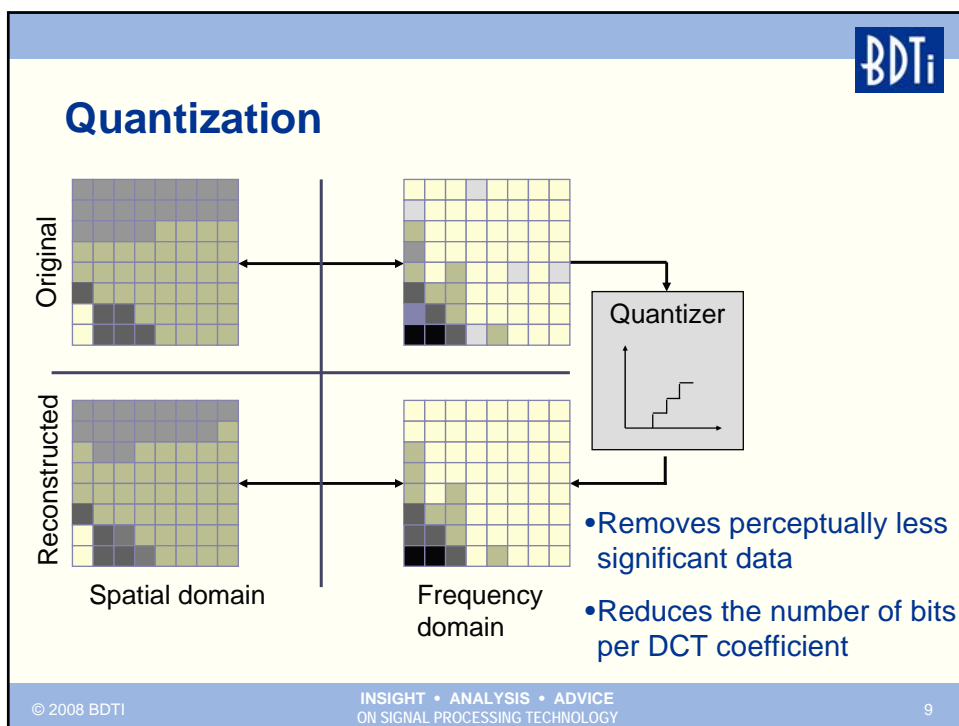
■ High energy  
□ Low energy

## Block Transform: Resource Reqt's

Compute load:

- Up to 30% of total video decoder processor cycles
- MPEG-4 CIF (352x288) @ 30 fps:
  - 71,280 DCTs/s
  - ~40 MHz on a TMS320C55x DSP
  - ~10 MHz if using TMS320C55x DCT accelerator
- Many implementation and optimization options
  - Can make compute requirements hard to predict

Memory usage: negligible



## Quantization: Resource Req't's

Quantization (encoder) and dequantization (decoder and encoder) have similar loads

Compute load:

- From 3% to about 15% of total decoder processor cycles
  - Typically near the lower end of this range
- MPEG-4 CIF (352x288) @ 30 fps:
  - ~10 MHz on a TMS320C55x DSP (estimated)

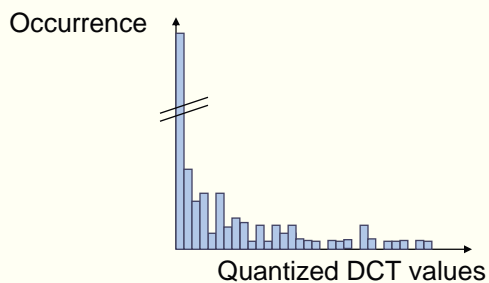
Memory usage: negligible

The BDTi logo is in the top right corner. The footer contains '© 2008 BDTI', 'INSIGHT • ANALYSIS • ADVICE ON SIGNAL PROCESSING TECHNOLOGY', and the number '10'.



## Coding Quantized DCT Coefficients

Goal: Reduce the number of bits required to transmit the quantized coefficients



Observation: Unequal distribution of quantized DCT coefficient values



## Variable Length Coding (VLC/VLD)

Allocates fewer bits to the most frequent symbols (e.g., using Huffman)

Integer number of bits per symbol

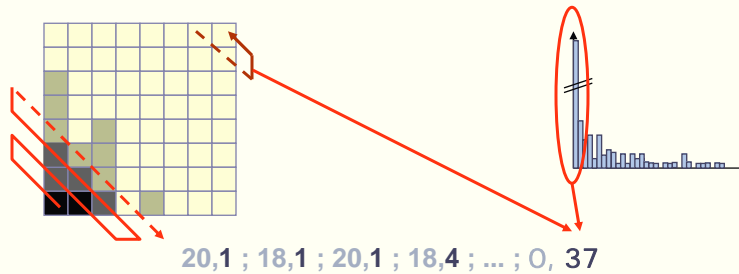
- Not the most efficient coding method
- Arithmetic coding more efficient, but expensive
- Run-length coding improves efficiency of VLC/VLD for image and video coding

<u>Symbol</u>	<u>Frequency</u>	<u>Code</u>
A	22	1
B	16	011
C	9	0101
D	7	0100
E	4	0011
F	2	0010
...	...	...

## Run-Length Coding

Encodes value and number of successive occurrences

Takes advantage of the high number of recurring zeros



20,1 ; 18,1 ; 20,1 ; 18,4 ; ... ; 0, 37

## VLC/VLD Processing Requirements

VL decoding much more computationally demanding than VL encoding

VLD compute load:

- Up to 25% of total video decoder processor cycles
- MPEG-4 CIF (352x288) @ 30 fps, 700 kbps:
  - ~15-25 MHz on a TMS320C55x DSP (estimated)
- About 11 operations per bit on average

Memory usage

- A few KB of memory for lookup tables
- More for speed optimizations

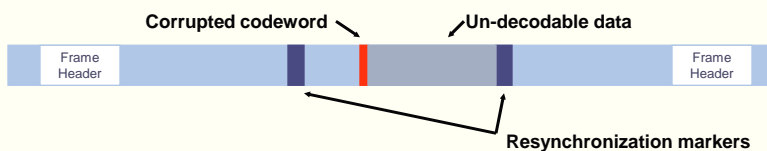
## Resynchronization Markers

Without markers, a single bit error in the coded bitstream prevents decoding of the rest of the frame

- Size of a corrupted variable-length code word is unknown
- Therefore, the start of the next code word (and all following code words) is unknown

Resynchronization markers help the decoder recover from bitstream errors

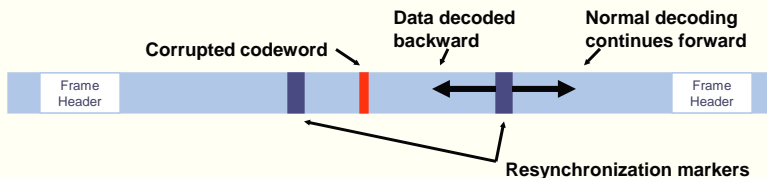
- Provide a known bit pattern interspersed throughout the bitstream
- In case of an error, decoder searches for next marker, then continues decoding



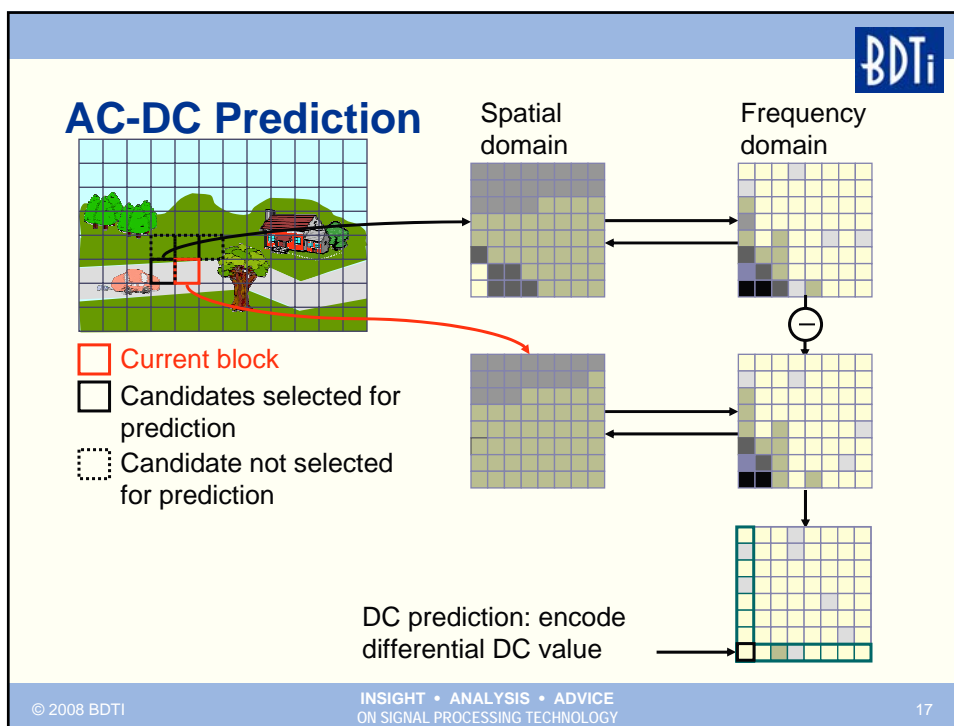
## Reversible VLC

Reversible VLC, in conjunction with resynchronization markers, further assists error recovery

- Code words can be decoded forward and backward
- In case of error, data can be decoded backward from the next resynchronization marker
- More data is recovered compared to resynchronization markers alone







**AC-DC Prediction**

AC-DC prediction cannot be used in conjunction with motion compensation

- **Used mostly for compressing a single image**
- DC prediction used in JPEG

AC-DC prediction often uses simple filters to predict each coefficient value from one or more adjacent blocks

© 2008 BDTI

INSIGHT • ANALYSIS • ADVICE  
ON SIGNAL PROCESSING TECHNOLOGY

18



## AC-DC Prediction: Processing Reqt's

### Compute load:

- DC prediction has negligible load
- AC-DC prediction used in about 8% of frames in typical video
  - Negligible average load (~2% of processor cycles in decoder)
  - Substantial per-frame load (~20-30% of cycles required to decode a frame that uses AC-DC prediction)

### Memory usage:

- Under 2 KB for CIF (352x288) resolution
  - But more memory (up to 10 KB) can result in faster code
- May be overlapped with other memory structures not in use during prediction



## Outline

Motivation and scope

Still-image compression techniques

**Motion estimation and compensation**

Reducing artifacts

Color conversion

Comparing performance and efficiency

Conclusions



## Motion Estimation and Compensation

Still image compression ignores the correlation between frames of video

- JPEG achieves ~10:1 compression ratio
- Wavelet transform-based image coding reaches compression ratios up to ~30:1

Adding motion estimation and compensation results in much higher compression ratios

- Good video quality at compression ratios as high as ~200:1



## Motion Estimation and Compensation

Requires at least one “reference frame”

- Reference frame must be encoded before the current frame
- But, reference frame can be a future frame in the display sequence

Three kinds of frames: I, P, and B

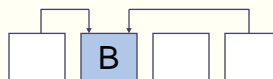
- I frame is encoded as a still image and doesn't depend on any reference frame



- P frame depends on previously displayed reference frame

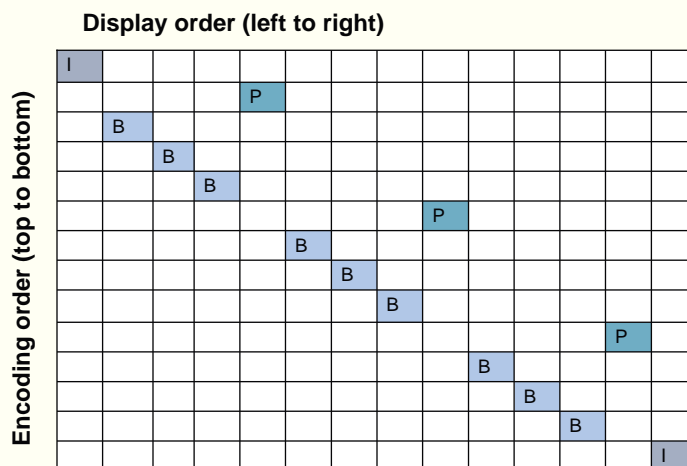


- B frame depends on previous and future reference frames





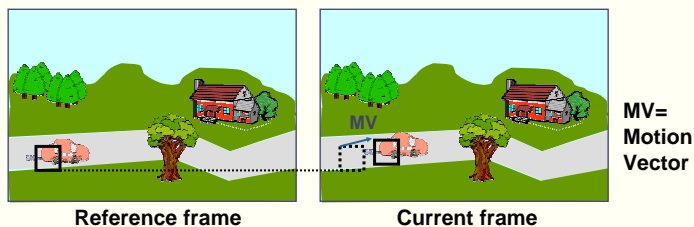
## Typical Sequence of I, P, B frames

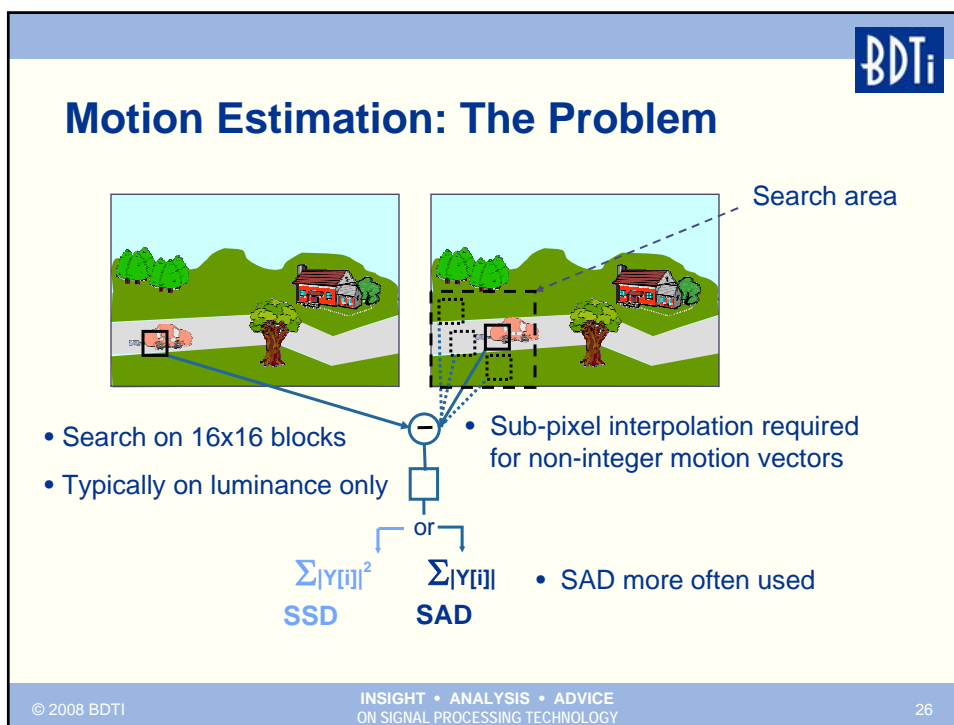
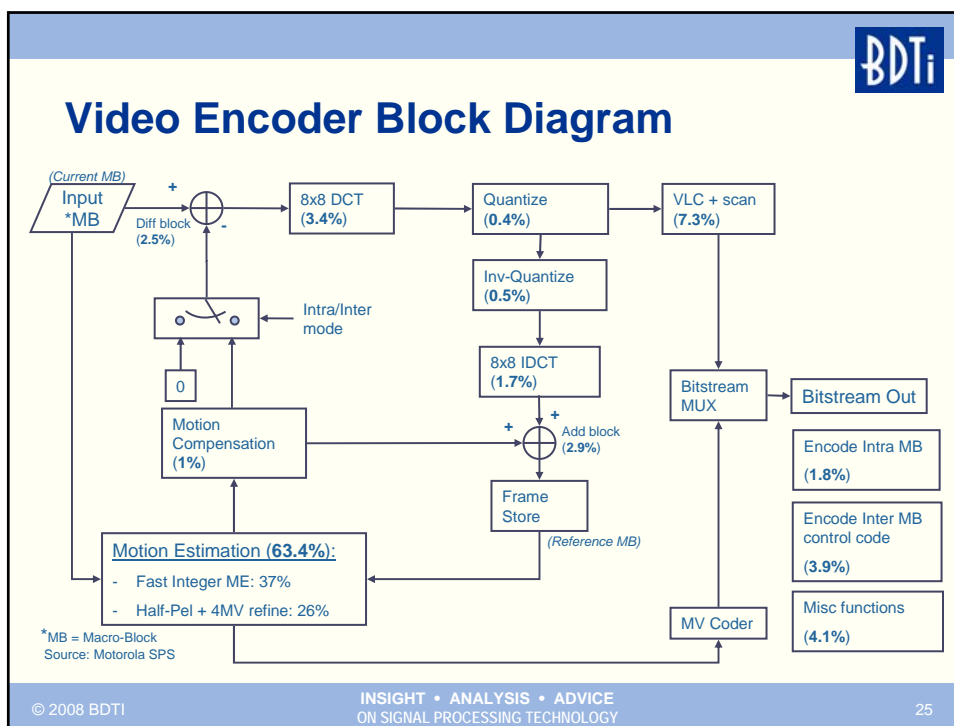



## Motion Estimation

Predict the contents of each macroblock based on motion relative to reference frame



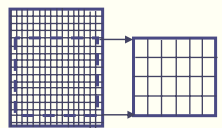
- Search reference frame for a 16x16 region that matches the macroblock
- Encode motion vectors
- Encode difference between predicted and actual macroblock pixels








## Efficient Motion Vector Search




- Exhaustive search is impractical
- Evaluate only promising candidate motion vectors
  - Don't have to find absolute best match
    - Trade video quality for computational load
  - Many methods in use
    - Often proprietary
    - Refine candidate vector selection in stages
    - Predict candidate vectors from surrounding macroblocks and/or previous frames



**Motion vector search approach is a key differentiator between video encoder implementations**

© 2008 BDTIINSIGHT • ANALYSIS • ADVICE  
ON SIGNAL PROCESSING TECHNOLOGY27



## Motion Estimation: Processing Reqt's

**Compute load**

- Most demanding task in video compression
  - Up to 80% of total encoder processor cycles
  - Many search methods exist, with varying requirements
  - May vary with video program content
  - Makes encoder computational demand several times greater than that of the decoder
  - Dominated by SAD computation

**Memory use**

- Motion estimation requires reference frame buffers
  - Frame buffers dominate memory requirements of encoder
    - E.g., 152,064 bytes per frame @ CIF (352x288) resolution
- High memory bandwidth required

© 2008 BDTIINSIGHT • ANALYSIS • ADVICE  
ON SIGNAL PROCESSING TECHNOLOGY28



## Motion Compensation: Processing Reqt's

Motion compensation copies pixels from reference frame to predict current macroblock

- Requires interpolation for non-integer motion vector values

Compute load

- Varies with video program content
- Can require from 5% to 40% of total decoder processor cycles
- MPEG-4 CIF (352x288) @ 30 fps:
  - Roughly 15-25 MHz on a TMS320C55x DSP (estimated)

Memory usage

- Requires reference frame buffers
  - Frame buffers dominate decoder memory requirements
- Good memory bandwidth is desirable, but less critical compared to motion estimation



## Outline

Motivation and scope

Still-image compression techniques

Motion estimation and compensation

**Reducing artifacts**

Color conversion

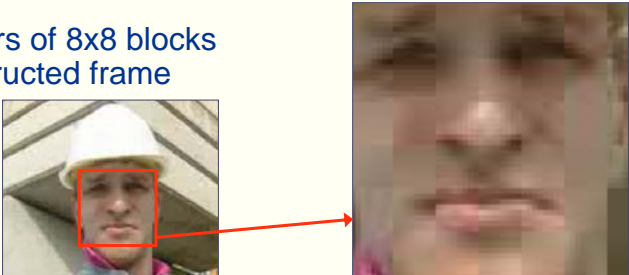
Comparing performance and efficiency

Conclusions

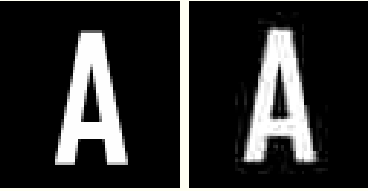
**BDTi**

## Artifacts: Blocking and Ringing

**Blocking:** Borders of 8x8 blocks visible in reconstructed frame



**Ringing:** Distortions near edges of image features



Original image

Reconstructed image (with ringing Artifacts)

© 2008 BDTI      INSIGHT • ANALYSIS • ADVICE ON SIGNAL PROCESSING TECHNOLOGY      31

**BDTi**

## Deblocking and Deringing Filters

Low-pass filters are used to smooth the image where artifacts occur

**Deblocking:**

- Low-pass filter the pixels at borders of 8x8 blocks
- One-dimensional filter applied perpendicular to 8x8 block borders

**Deringing:**

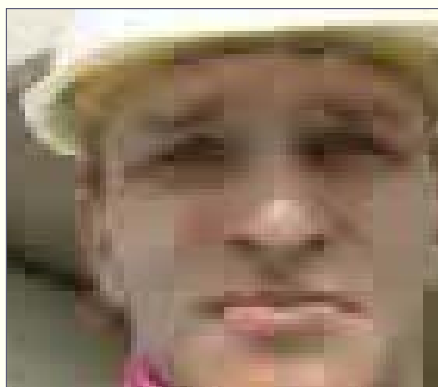
- Detect edges of image features
- Adaptively apply 2D filter to smooth out areas near edges
- Little or no filtering applied to edge pixels in order to avoid blurring

© 2008 BDTI      INSIGHT • ANALYSIS • ADVICE ON SIGNAL PROCESSING TECHNOLOGY      32

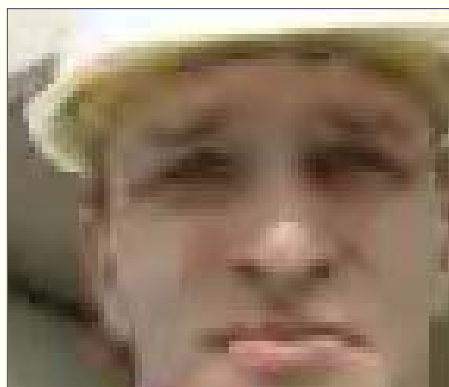




## Artifact Reduction: Deblocking



Original MPEG Still Frame



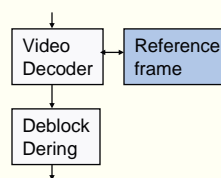
Horizontally & Vertically  
Deblocked Still Frame



## Post-processing vs. In-loop

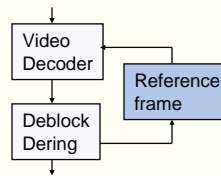
Deblocking/deringing often applied after the decoder (post-processing)

- Reference frames are not filtered
- Developers free to select best filters for the application or not filter at all



Deblocking/deringing can be incorporated in the compression algorithm (in-loop filtering)

- Reference frames are filtered
- Same filters must be applied in encoder and decoder
- Better image quality at very low bit-rates





### Artifact Reduction: Processing Reqt's

Deblocking and deringing filters can require more processor cycles than the video decoder

- Example: MPEG-4 Simple Profile, Level 1 (176x144, 15 fps) decoding requires 14 MIPS on ARM's ARM9E for a relatively complex video sequence
- With deblocking and deringing added, load increases to 39 MIPS
  - Nearly 3x increase compared to MPEG-4 decoding alone!

Post-processing may require an additional frame buffer



### Outline

Motivation and scope

Still-image compression techniques

Motion estimation and compensation

Reducing artifacts

**Color conversion**

Comparing performance and efficiency

Conclusions



## Color Space Conversion

### Need for color conversion

- Capture and display video equipment: RGB...
- ...while codecs use YUV

### Computational demand

- 12 operations per pixel → 36 million operations/second for CIF (352x288) @ 30 fps
  - About 36 MHz on a TMS320C55x DSP
  - Not including interpolation of chrominance planes
- Roughly 1/3 to 2/3 as many processor cycles as the video decoder



## Outline

Motivation and scope

Still-image compression techniques

Motion estimation and compensation

Reducing artifacts

Color conversion

**Comparing performance and efficiency**

Conclusions



## Challenges in Evaluating Solutions

Validate the production-readiness of the solution

Measuring performance fairly

- Algorithm configurations
- Test streams
- Test conditions
- Performance metrics

Vendor data often provides little help

- Apples-to-apples comparisons are difficult
- Varying and often unrealistic configurations, test streams, test conditions, and metrics



## BDTI's Approach

Standardization:

- Operating points (codec parameters)
- Test streams
- Metrics
- Instrumentation guidelines

Independent verification of:

- Functionality
- Performance
- Metrics used for comparisons



## BDTI H.264 Decoder Solution Benchmarking

Primary operating point:

- Baseline profile
- D1 resolution (720 × 480)
- 30 frames per second
- 1.5 Mbit/second test stream (*proprietary*)

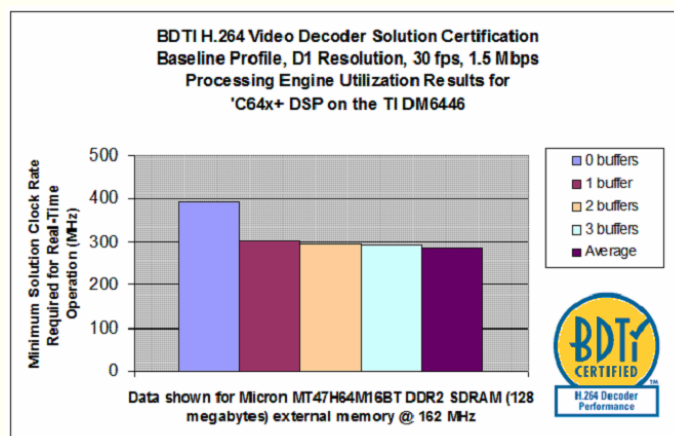
Other “secondary” operating points are characterized to provide a complete performance picture

Metrics:

- CPU utilization
- External memory bandwidth utilization
- Energy consumption (mJ/frame)
- Die area or cost (mm<sup>2</sup> or \$)
- Program and data memory use (Mbytes)



## Example Results: H.264 on TI TMS320DM6446



For more results, see [www.BDTI.com](http://www.BDTI.com)



## Outline

Motivation and scope

Still-image compression techniques

Motion estimation and compensation

Reducing artifacts

Color conversion

Comparing performance and efficiency

**Conclusions**



## Conclusions

Understanding the computational and memory requirements of video compression is critical but challenging

- Application design choices are driven by computational and memory requirements
  - Algorithm selection, processor selection, software optimization
  - Video processing often stresses processing resources
- But video applications combine many different signal-processing techniques
  - Transforms, prediction, quantization, entropy coding, image filtering, etc.
- And there is large variation in computational and memory requirements among different applications
  - E.g., digital camcorder has vastly different requirements from a video-enabled cell phone, even when using the same compression standard



## Conclusions, cont.

### Understanding computational load

- Computational load of encoder is several times greater than that of decoder due to motion estimation
- Computational load proportional to frame size and rate for most functions
  - Note: VLD computational load is proportional to bit rate
- Post-processing steps—deblocking, deringing, color space conversion—add considerably to the computational load

### Computational load can be difficult to predict

- Many different approaches to motion estimation
- Computational load of some tasks can fluctuate wildly depending on video program content
  - E.g., motion compensation



## Conclusions, cont.

### Understanding memory requirements:

- Memory requirements dominated by frame buffers
  - A decoder that supports only I and P frames requires two frame buffers (current and reference)
  - A decoder that supports I, P, and B frames requires three buffers (current and two reference)
  - Deblocking/deringing/color conversion may require an additional buffer
- Program memory, tables, other data comprise a non-negligible portion of memory use
  - But this portion is still several times smaller than frame buffers

### High memory use often necessitates off-chip memory

- Off-chip memory bandwidth can be a performance bottleneck



## Conclusions, cont.

Video compression used in many products

- DVDs, digital TV, personal video recorders, Internet video, multimedia jukeboxes, video-capable cell phones and PDAs, camcorders...

Different products have different needs

- Wide range of frame sizes and rates, video quality, bit rates, post-processing options, etc.
- Result in wide range of computational and memory requirements

Need to understand the operation of video codecs

- To understand computational and memory requirements
- To select codecs, processors, software modules
- To optimize software

Need to compare performance and efficiency of competing solutions

- Requires a thoughtful approach: algorithm variants, test data, metrics, ...



## For More (Free!) Information...

[www.BDTI.com](http://www.BDTI.com)

[www.INSIDEdsp.com](http://www.INSIDEdsp.com)

*Inside DSP* newsletter

benchmark scores for dozens of processors, FPGAs, video solutions, etc.

*Pocket Guide to Processors for DSP*

- Basic stats on over 40 processors

Articles, white papers, and presentation slides

- Processor architectures and performance
- Signal processing applications
- Signal processing software optimization

*comp.dsp* FAQ

