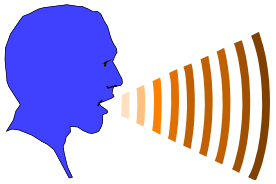
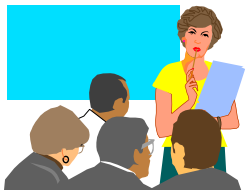



Image and Video Processing

Predictive Coding and Block-Based Hybrid Video Coding

Yao Wang
Tandon School of Engineering, New York University

Outline

- Predictive coding
  - General description
  - MMSE predictor
  - Gain of predictive coding
- Block-Based Hybrid Video Coding
  - Intra prediction
  - Inter prediction
  - Quantization of prediction error
  - Coding mode selection and rate control
  - Loop filtering
  - GoP Structure

Predictive Coding

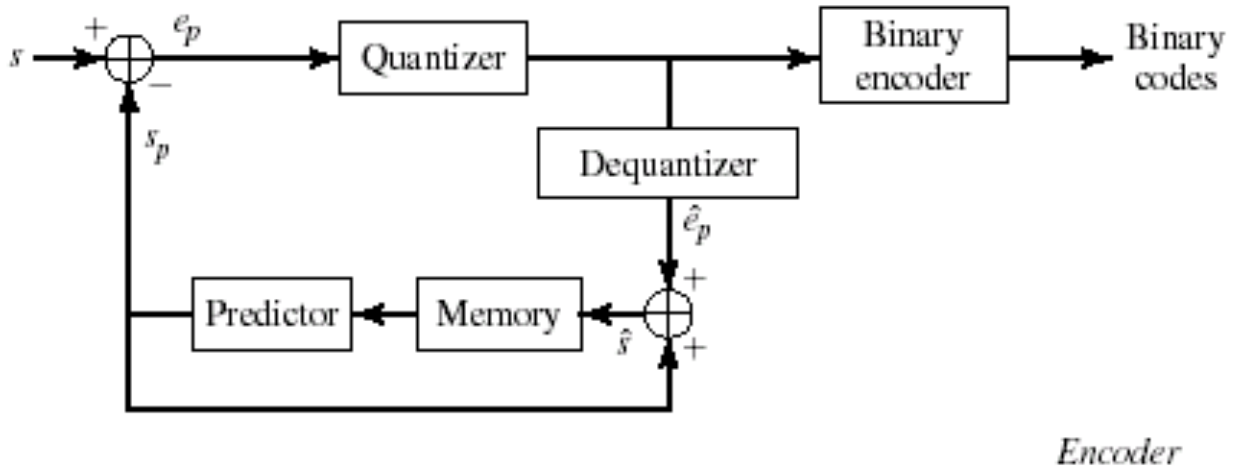
- Motivation: Predicts a sample from past samples and quantize and code the error only
- Because the prediction error typically has smaller variance than the original sample, it can be represented with a lower average bit rate
- Linear predictor: use linear combination of past coded/decoded samples (in the same frame or previous frame)
- Example: spatial prediction

A	B	C	D
E	F	G	H
I	J	K	L

$$\hat{f}_K = a f_F + b f_G + c f_H + d f_J$$

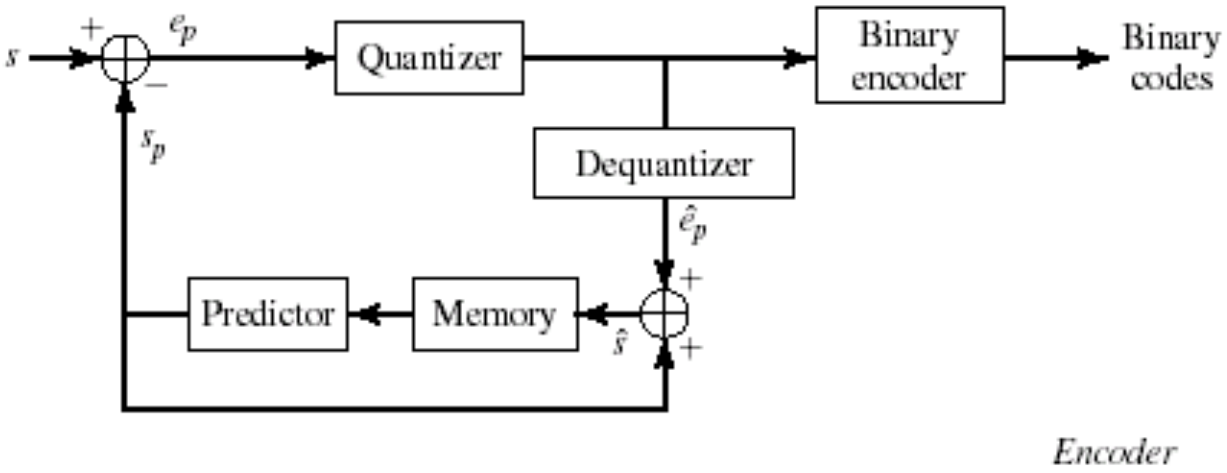
Encoder and Decoder Block Diagram

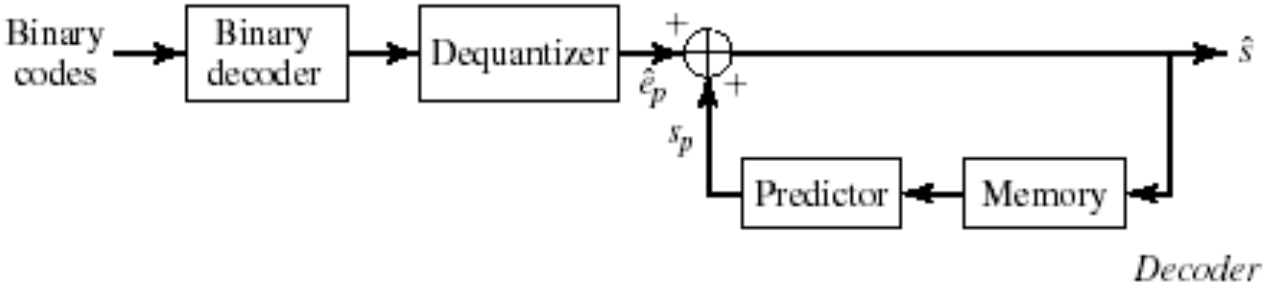
- Prediction is based on previously decoded (with quantization error) samples – closed-loop prediction to avoid encoder/decoder mismatch!



Distortion in Predictive Coder

- Reconstruction error = quantization error for the prediction error





$$e_p = s - s_p$$

$$\hat{e}_p = e_p + e_q$$

$$\hat{s} = s_p + \hat{e}_p$$

$$= s - e_p + e_p + e_q$$

$$= s + e_q$$

$$e = \hat{s} - s = e_q !$$

Optimal Predictor

- Question: what predictor should we use?
  - Minimize the bit rate for coding the prediction error
  - Because quantization error with a given bit rate depends on the variance of the signal, minimizing the quantization error = minimizing the prediction error variance (=mean square prediction error).
  - We will limit our consideration to linear predictor only

$$s_p = \sum_{k=1}^{K} a_k s_k$$

  - s_k denotes the k-th previous sample to use for prediction (k-th prediction sample)

Linear Minimal MSE Predictor

- Prediction error:

$$\sigma_p^2 = E\{|S_0 - S_p|^2\} = E\left\{\left|S_0 - \sum_{k=1}^{K} a_k S_k\right|^2\right\}.$$

- Optimal coefficients must satisfy (by setting derivative with respect to a_k to zero):

$$E\left\{\left(S_0 - \sum_{k=1}^{K} a_k S_k\right) S_l\right\} = 0, \quad l = 1, 2 \ldots, K. \qquad (*)$$

$$\sum_{k=1}^{K} a_k R(k, l) = R(0, l), \quad l = 1, 2 \ldots, K,$$

Note (*) is also known as the orthogonality principle in estimation theory

Matrix Form

- The previous equation can be rewritten as:

$$\begin{bmatrix} R(1,1) & R(2,1) & \cdots & R(K,1) \\ R(1,2) & R(2,2) & \cdots & R(K,2) \\ \cdots & \cdots & \cdots & \cdots \\ R(1,K) & R(2,K) & \cdots & R(K,K) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ \cdots \\ a_K \end{bmatrix} = \begin{bmatrix} R(0,1) \\ R(0,2) \\ \cdots \\ R(0,K) \end{bmatrix}$$

$$[\mathbf{R}]\mathbf{a} = \mathbf{r}.$$

$$\mathbf{a} = [\mathbf{R}]^{-1}\mathbf{r}.$$

$$\sigma_p^2 = E\{(\mathcal{S}_0 - \mathcal{S}_p)\mathcal{S}_0\} = R(0,0) - \sum_{k=0}^{K} a_k R(k,0)$$

$$= R(0,0) - \mathbf{r}^T\mathbf{a} = R(0,0) - \mathbf{r}^T R^{-1}\mathbf{r}.$$

- R(k,l) is the correlation between k-th and l-th prediction sample. R(k,0) is the correlation of the current sample with the k-th prediction sample.

Predictive Coding Gain

Recall: Reconstruction error for original samples = quantization error of prediction errors.
$D_{DPCM}=D_q(R)$

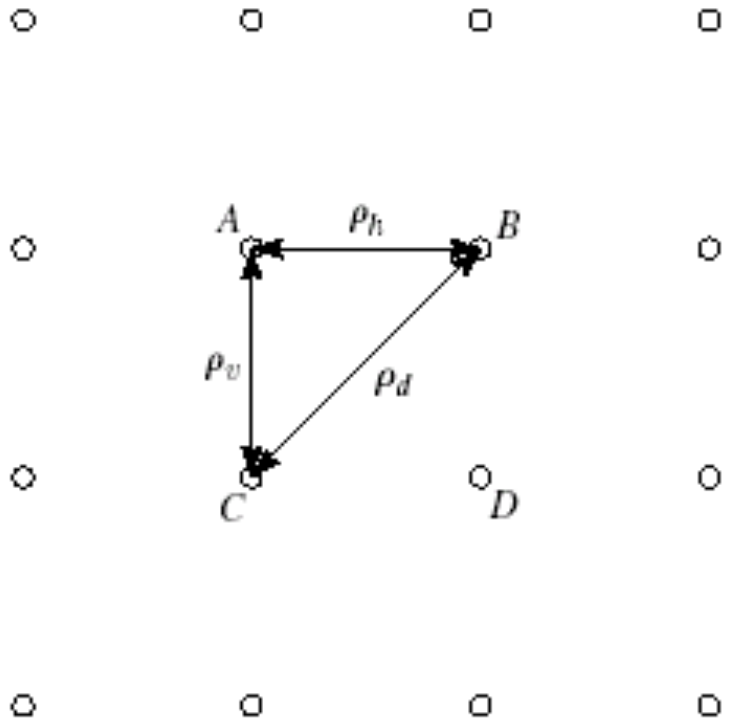
Distortion-rate for predictive coder (DPCM) $$D_{\text{DPCM}} = \epsilon_p^2 \sigma_p^2 2^{-2R}$$

Distortion-rate for coding the samples directly (PCM) $$D_{PCM} = \varepsilon_s^2 \sigma_s^2 2^{-2R}$$

Coding gain of DPCM: $$G_{\text{DPCM}} = \frac{D_{\text{PCM}}}{D_{\text{DPCM}}} = \frac{\epsilon_s^2}{\epsilon_p^2}\frac{\sigma_s^2}{\sigma_p^2}$$

PCM: Pulse coded modulation (quantize original sample directly)
DPCM: Differential PCM (quantize the difference of original sample and predicted sample)

Example



$$\hat{D} = a_1 C + a_2 B + a_3 A$$

Example Continued

$$\begin{bmatrix} R(C,C) & R(C,B) & R(C,A) \\ R(B,C) & R(B,B) & R(B,A) \\ R(A,C) & R(A,B) & R(A,A) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} R(D,C) \\ R(D,B) \\ R(D,A) \end{bmatrix}$$

$$\begin{bmatrix} 1 & \rho_d & \rho_v \\ \rho_d & 1 & \rho_h \\ \rho_v & \rho_h & 1 \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \rho_h \\ \rho_v \\ \rho_d \end{bmatrix}.$$

In the special case of $\rho_h = \rho_v = \rho$, $\rho_d = \rho^2$, the optimal predictor is

$$\begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = \begin{bmatrix} \rho \\ \rho \\ -\rho^2 \end{bmatrix}.$$

The MSE of this predictor, using Equation (9.2.10), is

$$\sigma_p^2 = R(0,0) - [R(0,1) \quad R(0,2) \quad R(0,3)] \begin{bmatrix} a_1 \\ a_2 \\ a_3 \end{bmatrix} = (1-\rho^2)^2 \sigma_s^2.$$

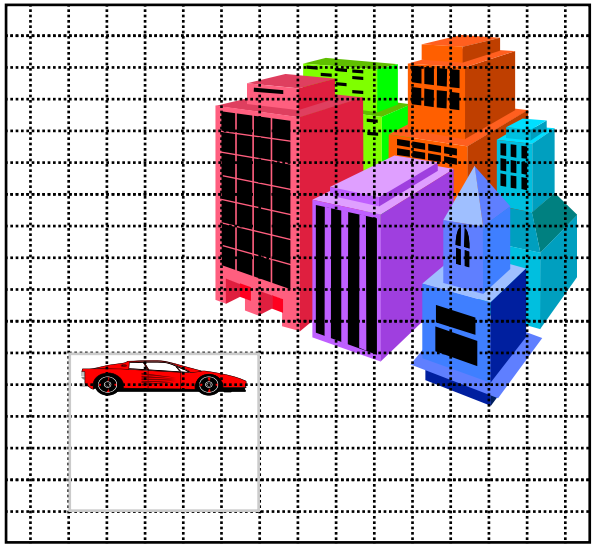
$$G_{\text{DPCM}} = \frac{\sigma_s^2}{\sigma_p^2} = \frac{1}{(1-\rho^2)^2}$$

(DPCM is better than TC for this case!)

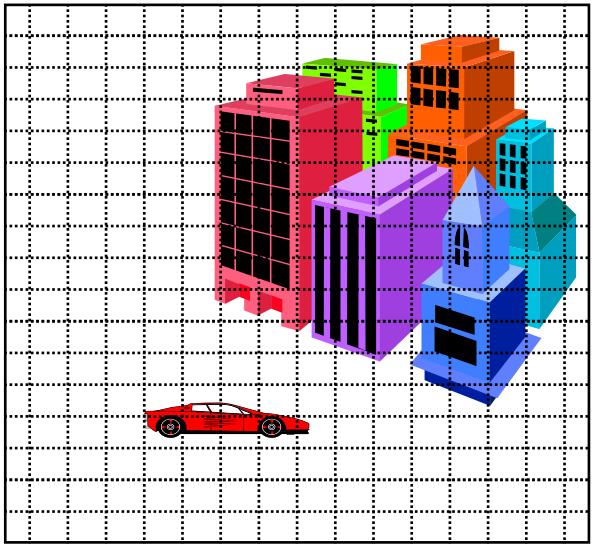
Predictive Coding for Image and Video

- For images: we can predict a current sample from previously coded samples.
  - In JPEG: predictive coding is used to code the DC coefficient of a block, which is the mean of the block. The current block DC is predicted from the previous block DC.
  - Adaptive directional prediction is used for coding I-frames in video coding. (non-linear predictor)
- For video: we apply prediction both among pixels in the same frame (intra-prediction or spatial prediction), and also among pixels in adjacent frames (inter-prediction or temporal prediction)
  - Both spatial and temporal predictor are adaptive (hence non-linear!)

Characteristics of Typical Videos



Frame *t-1*



Frame *t*

Adjacent frames are similar and changes are due to object or camera motion

Temporal Prediction

- No Motion Compensation (zero motion):
  - Work well in stationary regions

$$\hat{f}(t,m,n) = f(t-1,m,n)$$

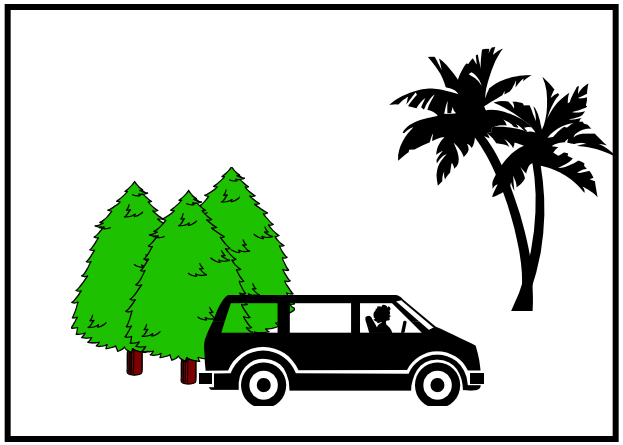
- Uni-directional Motion Compensation:
  - Does not work well for uncovered regions by object motion

$$\hat{f}(t,m,n) = f(t-1,m-d_x,n-d_y)$$

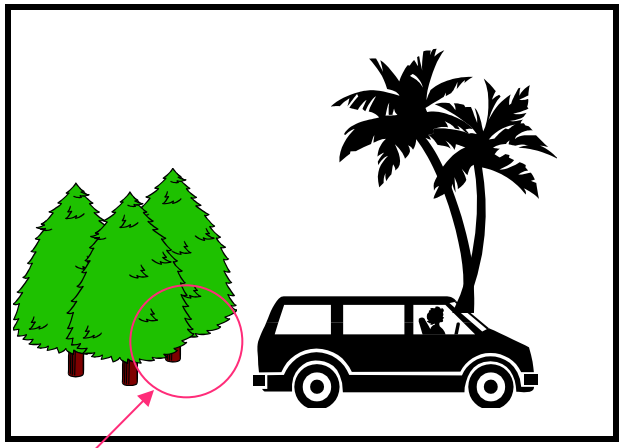
- Bi-directional Motion Compensation
  - Can handle better uncovered regions

$$\hat{f}(t,m,n) = w_b f(t-1,m-d_{b,x},n-d_{b,y}) + w_f f(t+1,m-d_{f,x},n-d_{f,y})$$

Uni-Directional Temporal Prediction



Past frame

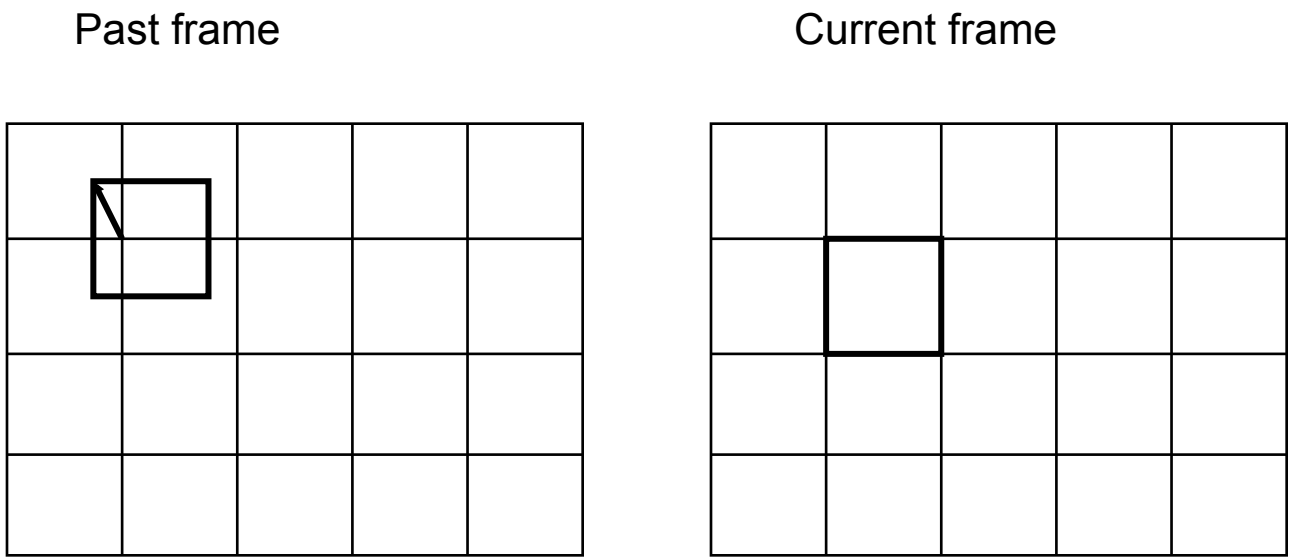


Current frame

All objects **except** this area have already been sent to decoder in "past frame"

From Amy Reibman

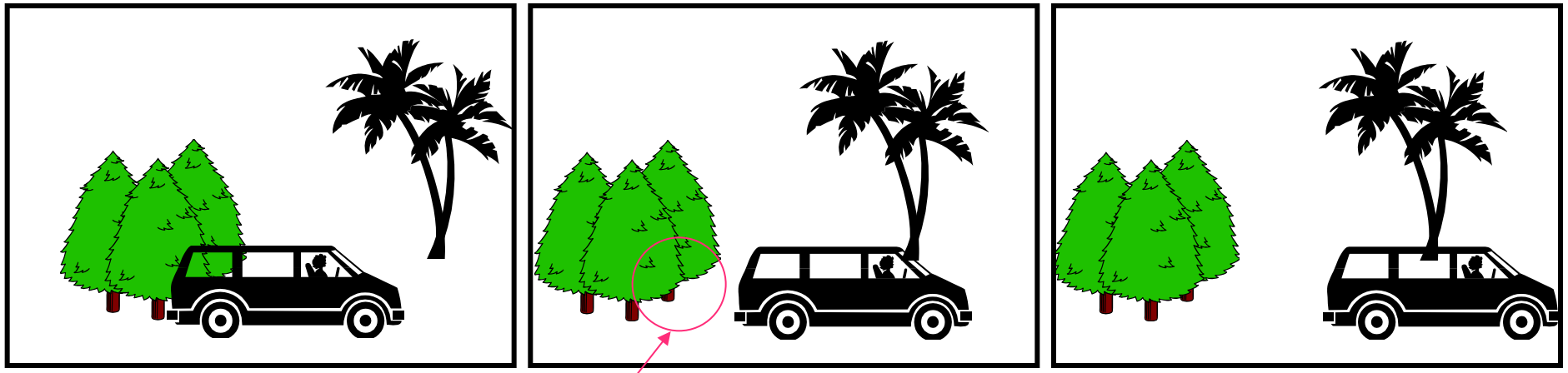
Motion Compensated Prediction



- Assumes rigid bodies move translationally; uniform illumination; no occlusion, no uncovered objects
- Big win: Improves compression by factor of 5-10

From Amy Reibman

Bi-directional Prediction



Past frame

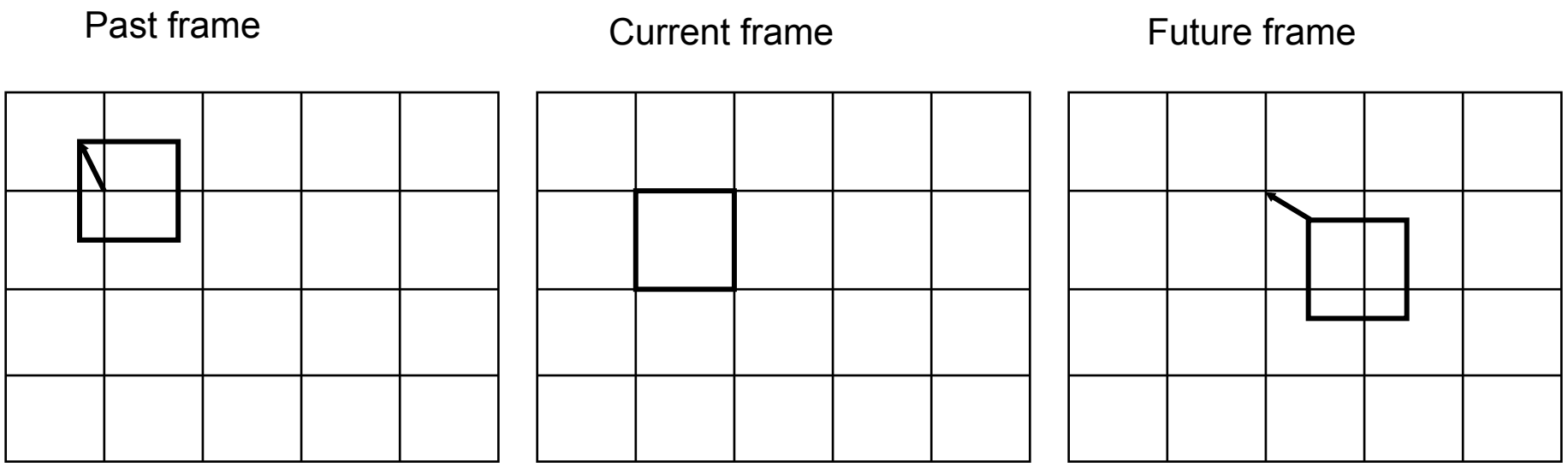
Current frame

Future frame

This area can now be predicted using “future frame”

From Amy Reibman

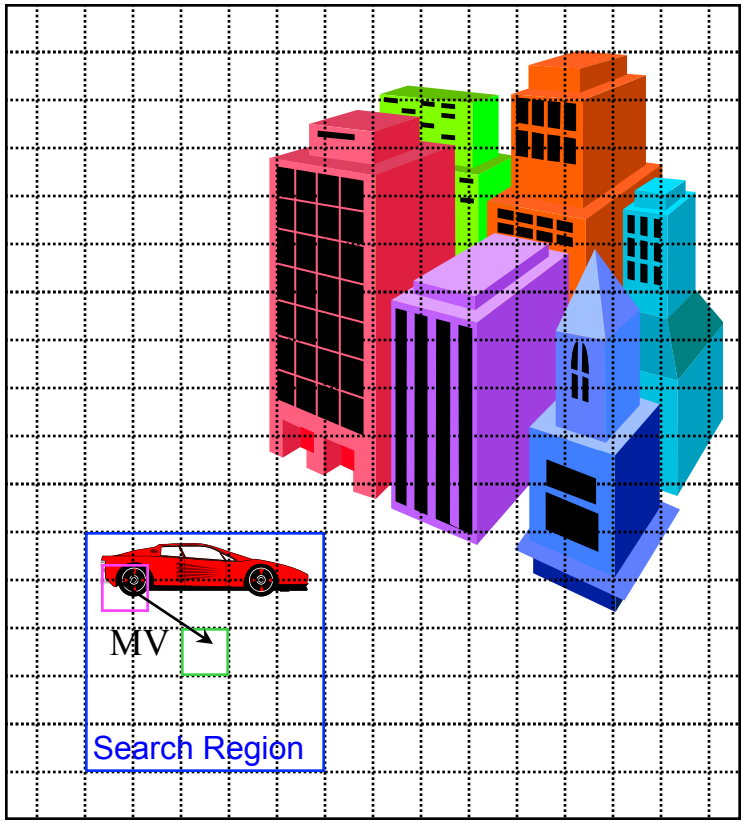
Motion Compensated Bidirectional Prediction



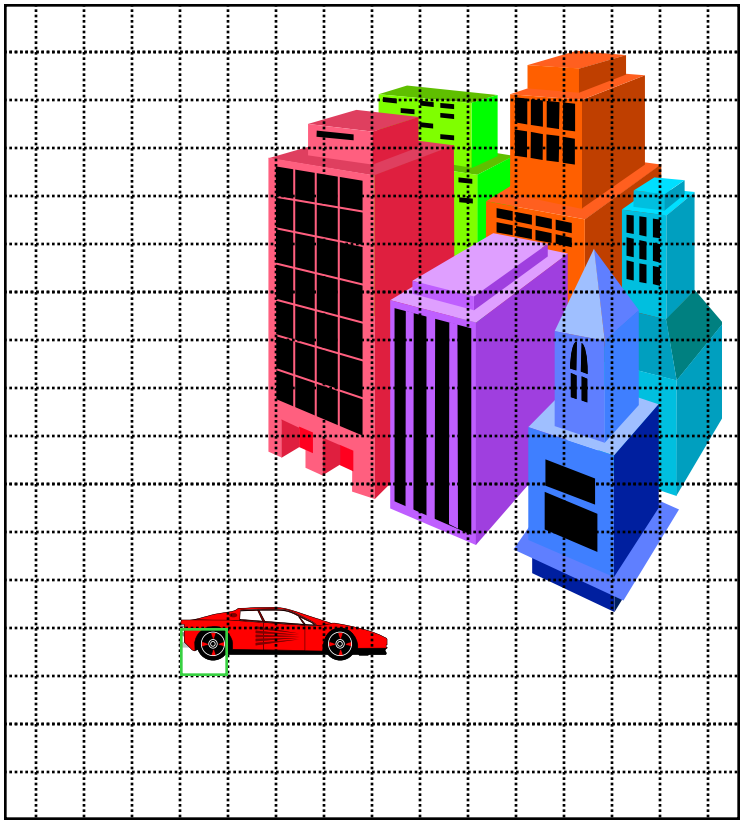
- Code past frame and future frame first, future predicted from past
- Then code the current frame using bi-directional prediction
- Helps when there is occlusion or uncovered objects
- Vector into the future need not be the same as vector into the past

From Amy Reibman

Block Matching Algorithm for Motion Estimation



Reference Frame, target frame



Predicted frame, anchor frame

Reference frame can be before or after the predicted frame in the original frame order!

Reference frame must be coded before the predicted frame!

Fractional-pel step-size MV search is often used to yield more accurate prediction

Multiple Reference Frame Temporal Prediction

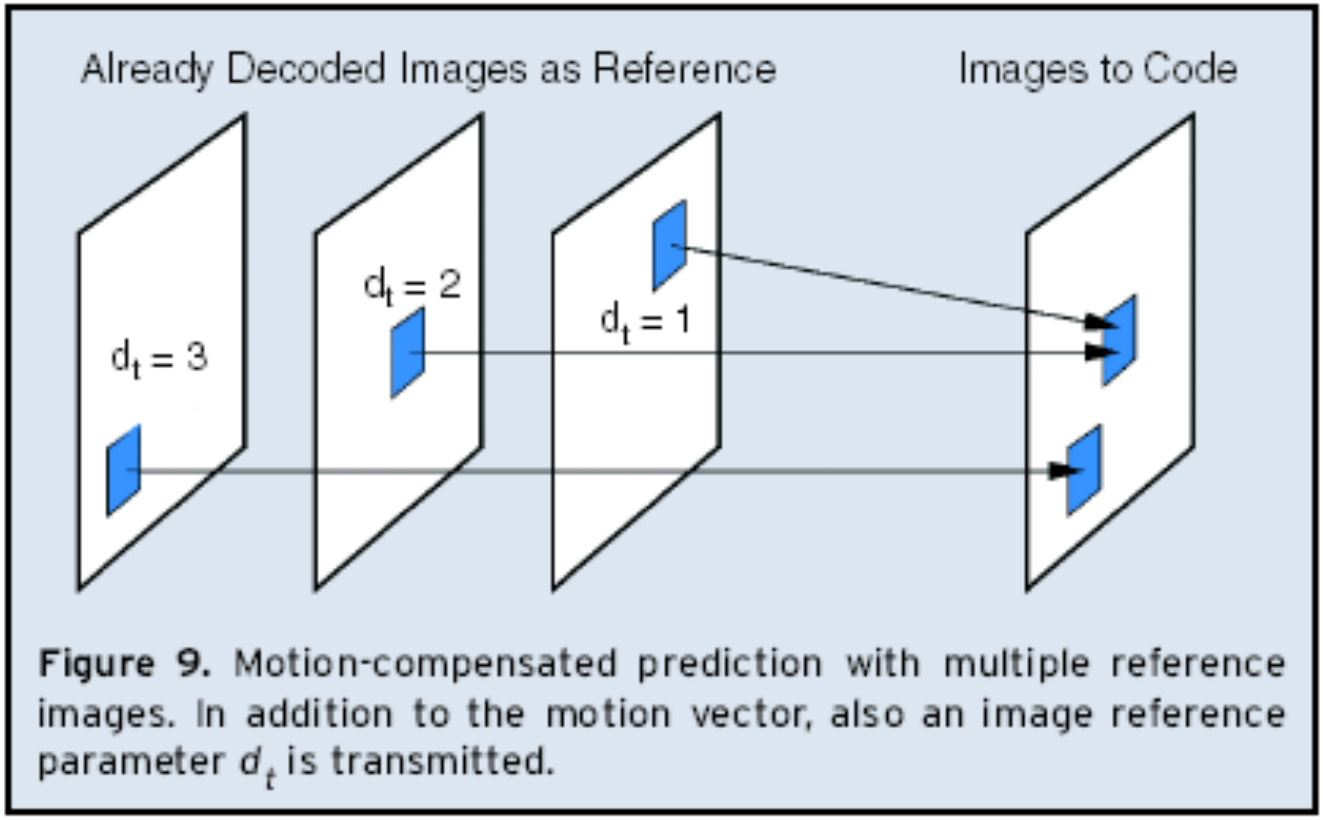


Figure 9. Motion-compensated prediction with multiple reference images. In addition to the motion vector, also an image reference parameter d_t is transmitted.

One may choose the best prediction among all frames, or use a weighted average of the predictions from all reference frames.

Gain of Uni-Directional Prediction

- Uni-directional motion compensated prediction

$$\hat{f}(t,m,n) = f(t-1, m-d_x, n-d_y)$$

Assume the signal is stationary with variance σ_p^2 and

the correlation coefficient between $f(t,m,n)$ and $f(t-1,m-d_x,n-d_y)$ is ρ

$$\begin{aligned}\sigma_p^2 &= E\left\{\left(f(t,m,n) - f(t-1,m-d_x,n-d_y)\right)^2\right\} \\ &= E\left\{f^2(t,m,n) + f^2(t-1,m-d_x,n-d_y) - 2f(t,m,n)f(t-1,m-d_x,n-d_y)\right\} \\ &= 2\sigma_s^2(1-\rho)\end{aligned}$$

Gain over coding a pixel directly $G_P = \dfrac{\sigma_s^2}{\sigma_p^2} = \dfrac{1}{2(1-\rho)}$

ρ is typically in the range of (0.9, 1)

Gain of Bi-Directional Prediction

- Bi-directional motion compensated prediction

$$\hat{f}(t,m,n) = w_b f(t-1, m-d_{b,x}, n-d_{b,y}) + w_f f(t+1, m-d_{f,x}, n-d_{f,y})$$

Assume the signal is stationary with variance σ_s^2 and the correlation coefficient between $f_0 = f(t,m,n)$ and $f_1 = (t\text{-}1, m\text{-}d_{b,x}, n-d_{b,y})$ and that between $f(t,m,n)$ and $f_2 = f(t+1, m\text{-}d_{f,x}, n-d_{f,y})$ are both ρ, the correlation coefficient between f_1 and f_2 is ρ^2. Further consider the special case of $w_b = w_f = 1/2$

$$\sigma_p^2 = E\left\{\left(f_0 - \frac{1}{2}f_1 - \frac{1}{2}f_2\right)^2\right\} = E\left\{f_0^2 + \frac{1}{4}f_1^2 + \frac{1}{4}f_2^2 - f_0 f_1 - f_0 f_2 + \frac{1}{2}f_1 f_2\right\}$$

$$= \sigma_s^2\left(1 + \frac{1}{4} + \frac{1}{4} - \rho - \rho + \frac{1}{2}\rho^2\right) = \sigma_s^2\left(\frac{3}{2} - 2\rho + \frac{1}{2}\rho^2\right) = \sigma_s^2\frac{(1-\rho)(3-\rho)}{2}$$

Gain over coding a pixel directly $G_B = \dfrac{\sigma_s^2}{\sigma_p^2} = \dfrac{2}{(1-\rho)(3-\rho)}$

$\dfrac{G_B}{G_P} = \dfrac{4}{(3-\rho)}$ is close to 2 since ρ is close to 1.

Spatial Prediction

- General idea:
  - A pixel in the new block is predicted from previously coded pixels in the same frame
  - What neighbors to use?
  - What weighting coefficients to use?
- Content-adaptive prediction
  - No edges: use all neighbors
  - With edges: use neighbors along the same direction
  - The best possible prediction pattern can be chosen from a set of candidates, similar to search for best matching block for inter-prediction
    - H.264 has many possible intra-prediction pattern

H.264 Intra Prediction

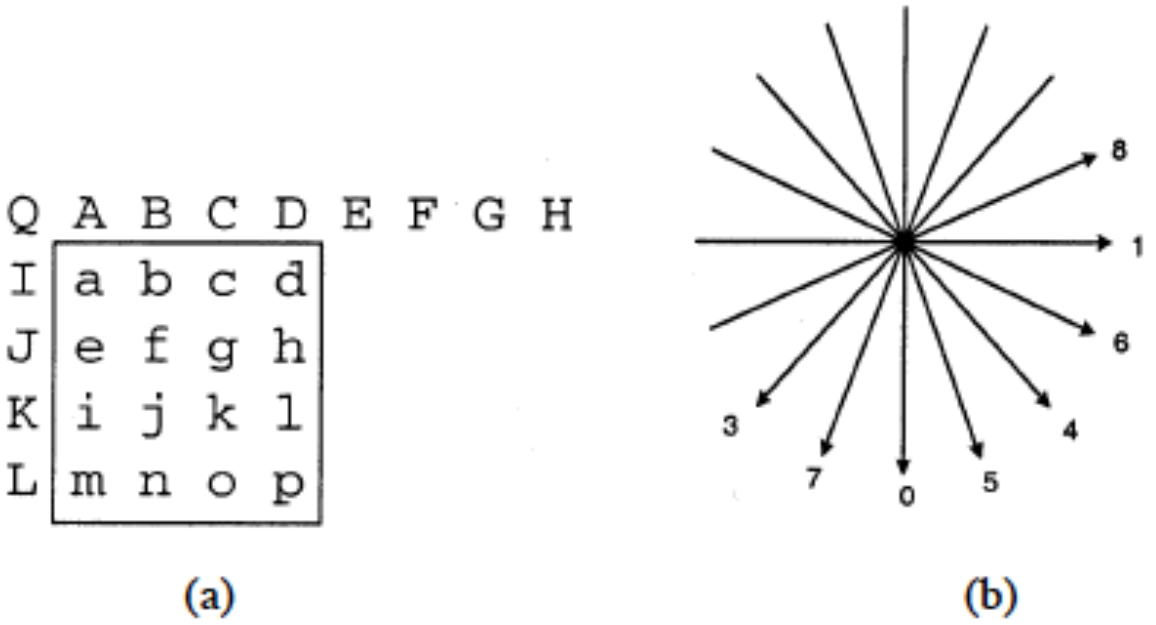


Fig. 10. (a) Intra_4×4 prediction is conducted for samples a-p of a block usin samples A-Q. (b) Eight "prediction directions" for Intra_4×4 prediction.

Intra-Prediction modes in H.264

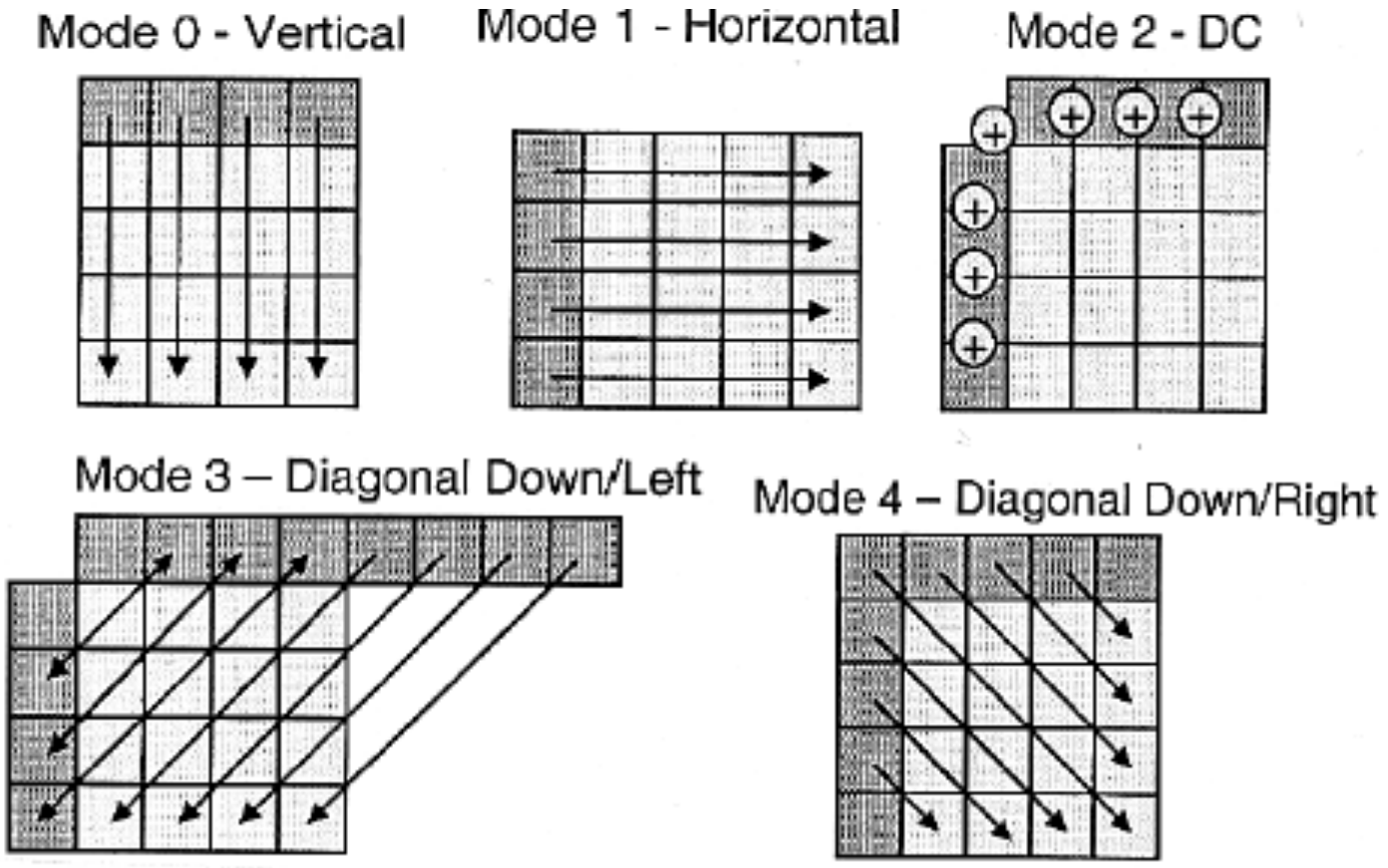


Fig. 11. Five of the nine Intra_4×4 prediction modes.

Coding of Prediction Error Blocks

- Error blocks typically still have spatial correlation
- To exploit this correlation:
  - Vector quantization
  - Transform coding
- Vector quantization
  - Can effectively exploit the typical error patterns due to motion estimation error
  - Computationally expensive, requires training
- Transform coding
  - Can work with a larger block under the same complexity constraint
  - Which transform to use?
  - DCT is typically used

Transform Coding of Error Blocks

- Theory: Karhunen Loeve Transform is best possible block-based transform
- Problems with theory:
  - Finding an accurate model (covariance matrix) of the source is difficult
  - Model and KLT change over time and in different regions
  - Decoder and encoder need to use same KLT
  - Implementation complexity: a full matrix multiply is necessary to implement KLT
- Practice: Discrete Cosine Transform
  - When the the inter-pixel correlation approaches one, the KLT approaches the DCT

Transform Coding: What block size?

- Theory: Larger transform blocks (using more pixels) are more efficient
- Problem with theory:
  - Hard to get an accurate model of the correlation of distant pixels
  - In the limit as the inter-pixel correlation approaches one, the KLT approaches the DCT; however, the inter-pixel correlation of distant pixels is not close to one
- Practice:
  - Small block transforms – usually 8x8 pixels, although in more recent systems we can use 4x4 blocks or 16x16 blocks

From Amy Reibman

Key Idea in Video Compression

- Divide a frame into non-overlapping blocks
- Predict each block using different modes (intra-, unidirection-inter, bidirectional-inter)
- Choose the best prediction mode (the one leading to least prediction error)
- Quantize and code prediction error using the DCT method
- Code (losslessly) the mode and motion info
- Prediction errors have smaller energy than the original pixel values and can be coded with fewer bits
- Work on each macroblock (MB) (16x16 pixels) independently for reduced complexity
  - Motion compensation done at the MB level (although an MB may be split into smaller blocks predicted using different motion vectors
  - DCT coding of error at the block level (8x8 pixels or 4x4 pixels)
- Hybrid coding: predictive coding+transform coding

Macroblock Coding in Intra-Mode

Find the best intra-prediction among all intra-prediction modes, determine the prediction error (intra-prediction may be skipped)

↓

DCT transform each prediction error block

↓

Quantize the DCT coefficients with properly chosen quantization matrices

↓

The quantized DCT coefficients are zig-zag ordered and run-length coded

Macroblock Coding in P-Mode

Estimate one MV for each macroblock (16x16)

↓

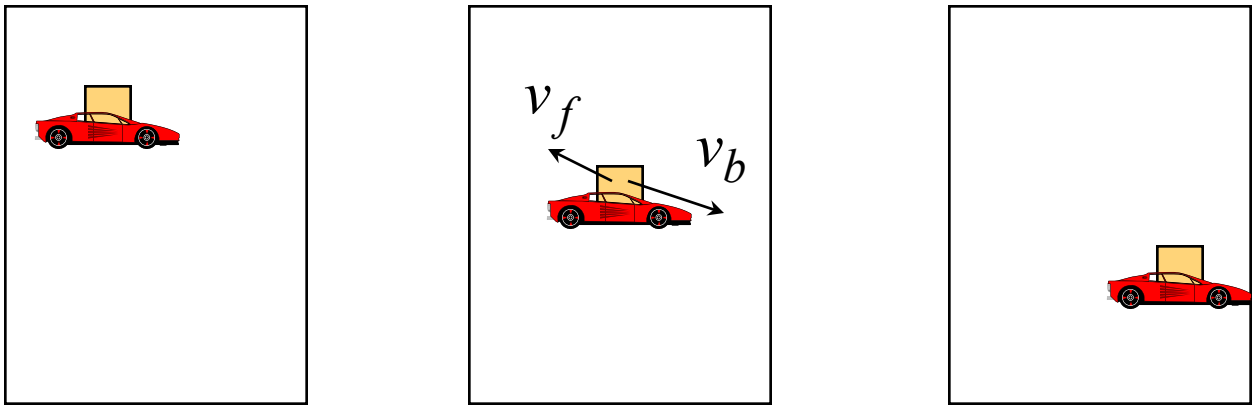
Depending on the motion compensation error, determine the coding mode (intra, inter-with-no-MC, inter-with-MC, etc.)

↓

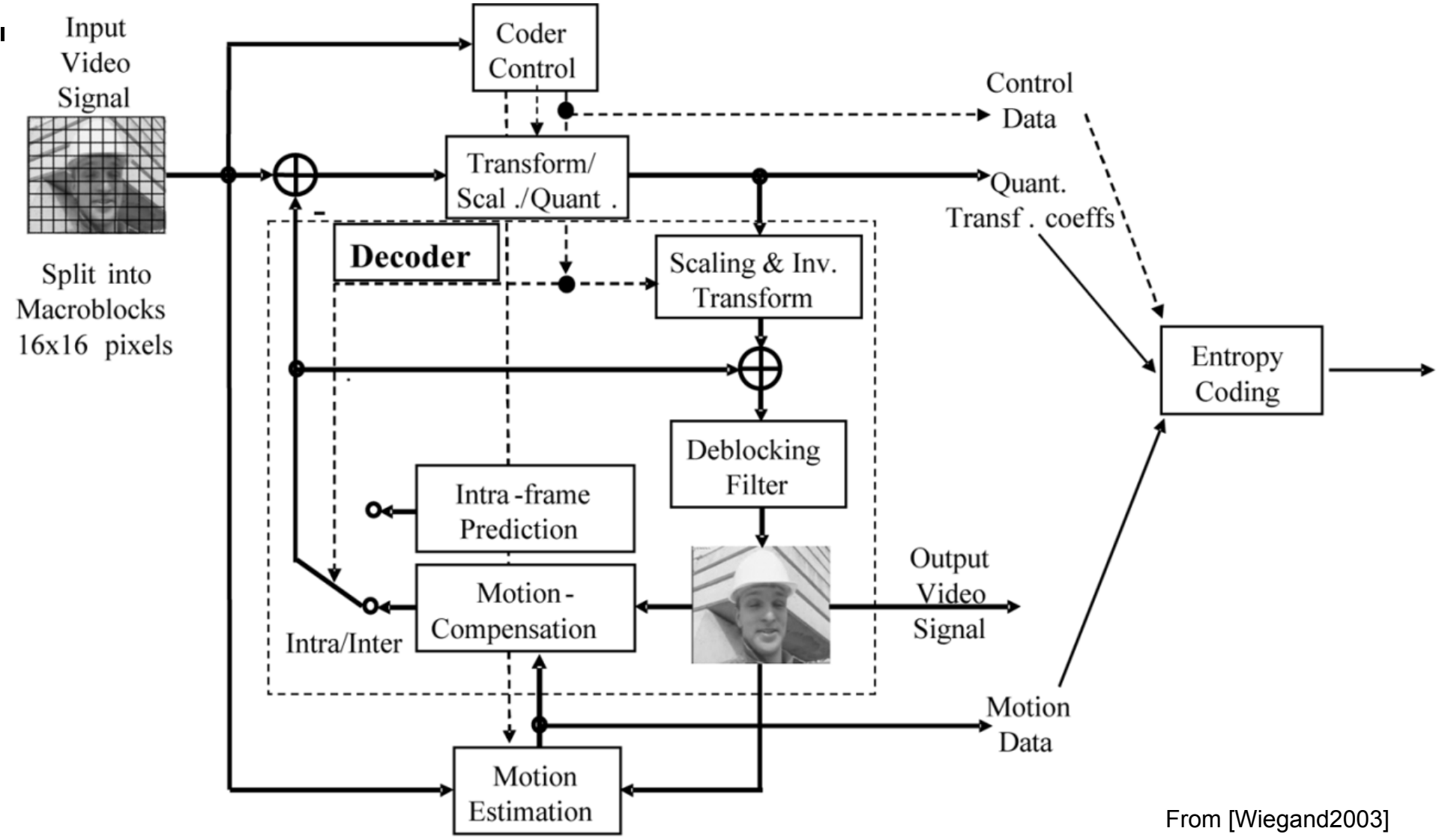
The original values or intra-prediction error or motion compensation errors in each of blocks (8x8) are DCT transformed, quantized, zig-zag/alternate scanned, and run-length coded (Huffman or arithmetic coding)

Macroblock Coding in B-Mode

- Same as for the P-mode, except a macroblock can be predicted from a previous picture, a following one, or both.

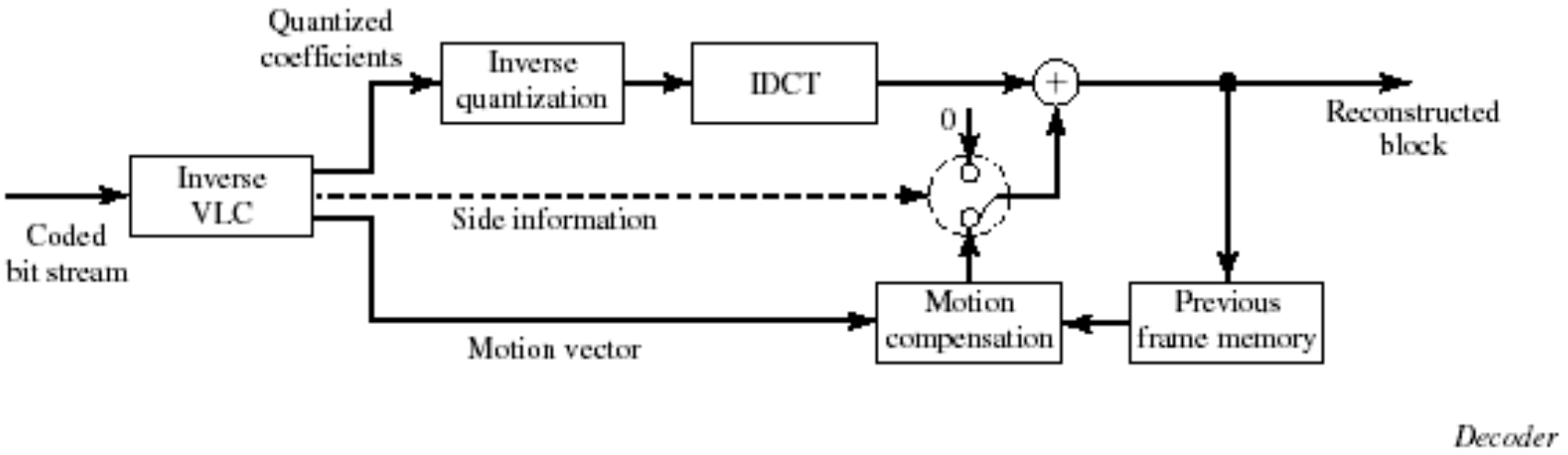


General Block Diagram



From [Wiegand2003]

Decoder Block Diagram



Should add the branch for intra-prediction!

Coding of Motion Vector

- Typically we predict the MV for a current block from MV of previously coded blocks and code the prediction error using entropy coding
  - Ex: using median MV of the top left, top, and left blocks
  - Prediction error is typically small, and has a Laplacian distribution (reduced entropy than original MV!)
- We may use a special flag to indicate the case when MV=0 (a special mode)

Rate-distortion Optimized Motion Estimation

- In EBMA: we find MV to minimize the prediction error

$$\mathbf{v}^* = \arg\min\{D_p(\mathbf{v})\}, \quad D_p(\mathbf{v}) = \sum_{(x,y)\in B} \left|\psi_2(x,y) - \psi_1(x+v_x, y+v_y)\right|^p$$

- Why do we want to minimize prediction error?
  - # required bits for coding the error is proportional to mean square of prediction error
  - $D_q = \varepsilon^2 \sigma_p^2 2^{-2R_p}$ (with MSE optimized quantizer design)

    or more generally

$$D_q = \varepsilon^2 \sigma_p^2 2^{-\alpha R_p} \text{ or } R_p = \frac{1}{\alpha}\log_2 \frac{\varepsilon^2 \sigma_p^2}{D_q}$$

- But we also need to code the MV!
  - A small error may be associated with a MV that rarely occur and hence require more bits to code!
- RD optimized motion estimation

$$\mathbf{v}^* = \arg\min\{D_p(\mathbf{v}) + \lambda R(\mathbf{v})\}$$

Coding Mode Selection

- Which mode to use for a block?
  - First use RD optimal motion estimation to determine best MV for both unidirectional and bi-directional inter mode, and RD optimal intra-mode decision to determine best intra-mode
  - Then choose between P-mode, B-mode and I-mode
- Rate-distortion optimized mode selection

$$m^* = \arg\min\{D_q(m) + \lambda R(m)\}$$

λ: Lagrangian multiplier, depending on expected quantization distortion or QP

$D_q(m)$: final reconstruction error with mode m =quantization error of prediction error

$R(m)$: total bits for mode m, including bits for coding the mode info, MV, and prediction error

RDO mode selection: Coding a block with all candidates modes and taking the mode that yields the least cost.

Fast mode selection: Using some simple computation to limit the candidate set, estimate the bits instead of running actual encoder,…

How to Choose the Lagrange Parameter?

$L(m) = D_q(m) + \lambda R(m)$

Think of m as a continuous variable, m* should be chosen so that

$$\frac{\partial L}{\partial m} = \frac{\partial D}{\partial m} + \lambda \frac{\partial R}{\partial m} = 0 \quad \text{or} \quad \lambda = -\frac{\partial D}{\partial m} \Big/ \frac{\partial R}{\partial m} = -\frac{\partial D}{\partial R}$$

λ = distortion-rate slope at the operating rate or distortion

Distortion depends on quantization error!

Assume uniform quantization using a uniform quantizer with stepsize=q

When q is small, we can assume $D(q) = q^2/12$.

Furthermore, $D(R) = b2^{-aR}$ or R= -a $\log_2\left(D/b\right)$ = -a $\log_2\left(q^2/12b\right)$

$$\frac{\partial D}{\partial q} = q/6, \frac{\partial R}{\partial q} = -2a/q \ln 2 \Rightarrow \lambda = -\frac{\partial D}{\partial q} \Big/ \frac{\partial R}{\partial q} = cq^2$$

λ should be proportional to q^2!

- Practical Solution: Determine c experimentally!
- T. Wiegand and B. Girod, "LAGRANGE MULTIPLIER SELECTION IN HYBRID VIDEO CODER CONTROL," Proc. ICIP2001

Variable Block Size Motion Estimation and Mode Decision

- For improved accuracy, starting with a maximum block size, we may partition it to smaller blocks, and allow different prediction modes (or MV / intra direction) in each sub-block.
- We pick the partition (and mode for each subblock) that yields minimal Lagrange cost for the entire block.

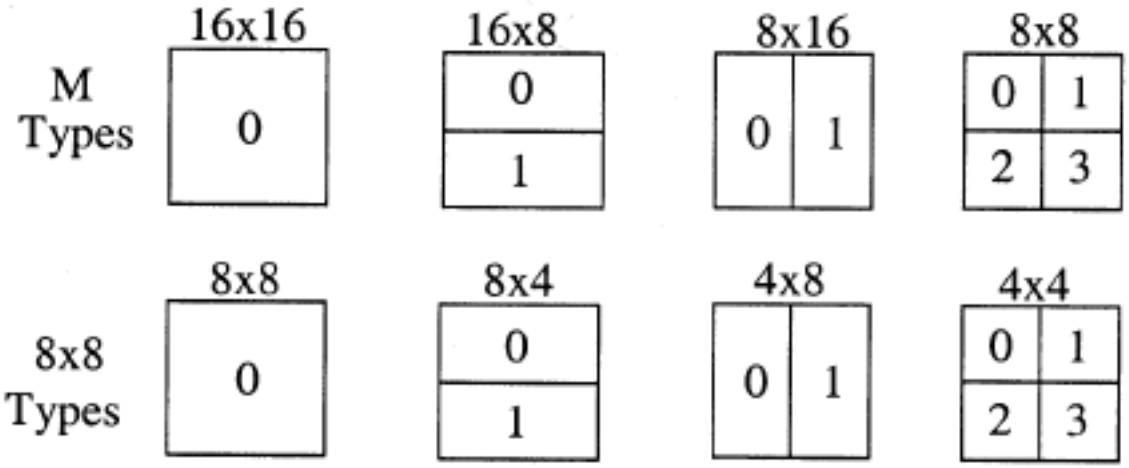
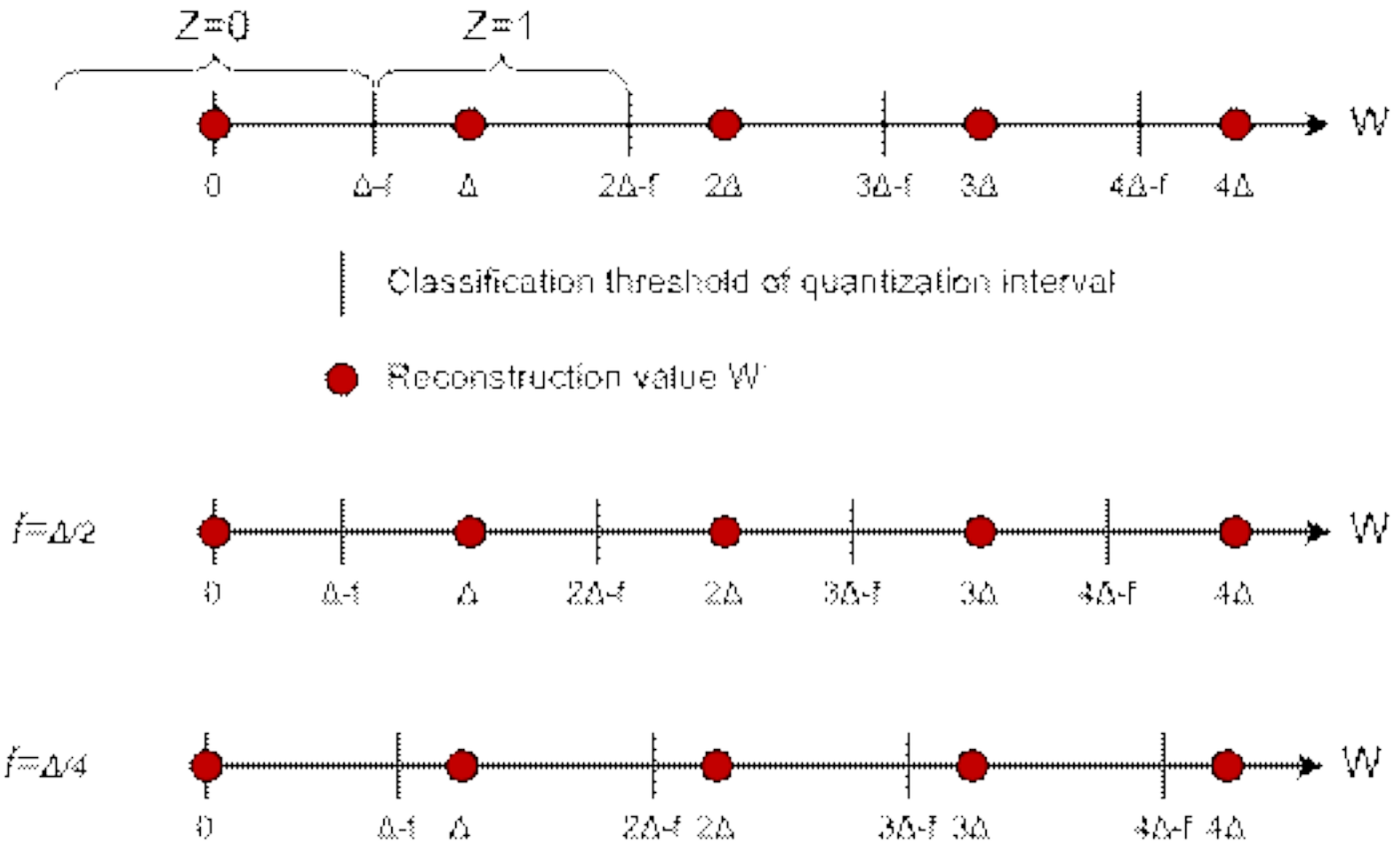


Fig. 12. Segmentations of the macroblock for motion compensation. Top: segmentation of macroblocks, bottom: segmentation of 8×8 partitions.

From [Wiegand2003]

Deadzone Quantizer for Transform Coefficients



From http://www.h265.net/2009/06/quantization-techniques-in-jmkta-part-2.html

Why deadzone quantizer?

- Recall the "centroid condition" – Reconstruction level = centroid (conditional mean) of a partition interval
- For non-uniform symmetric pdf, it is expected that the reconstruction level would not be in the middle of an interval
- Transform coefficients of prediction error can be modeled well by Laplacian distribution

$$p(w)=\frac{\lambda}{2}\exp\{-\lambda|w|\}$$

Centroid in the interval $(k\Delta-f,(k+1)\Delta-f)$:

$$W_k=\int_{k\Delta-f}^{(k+1)\Delta-f} wp(w)dw \Big/ \int_{k\Delta-f}^{(k+1)\Delta-f} p(w)dw$$

$$=k\Delta-f+\frac{1}{\lambda}-\frac{\Delta}{e^{\lambda\Delta}-1}$$

Let $W_k=k\Delta$, we get

$$f^*=\frac{1}{\lambda}-\frac{\Delta}{e^{\lambda\Delta}-1}$$

For Laplacian source, by choosing the particular partition structure of deadzone quantizer and set "f" in this way, the reconstruction level in each interval minimizes the quantization error of that interval!

For H.264: f=Δ/3 for intra modes, f=Δ/6 for Inter modes (with more peaky distribution or larger λ)

$$p(w)=\frac{\lambda}{2}\exp\left\{-\lambda|w|\right\}$$

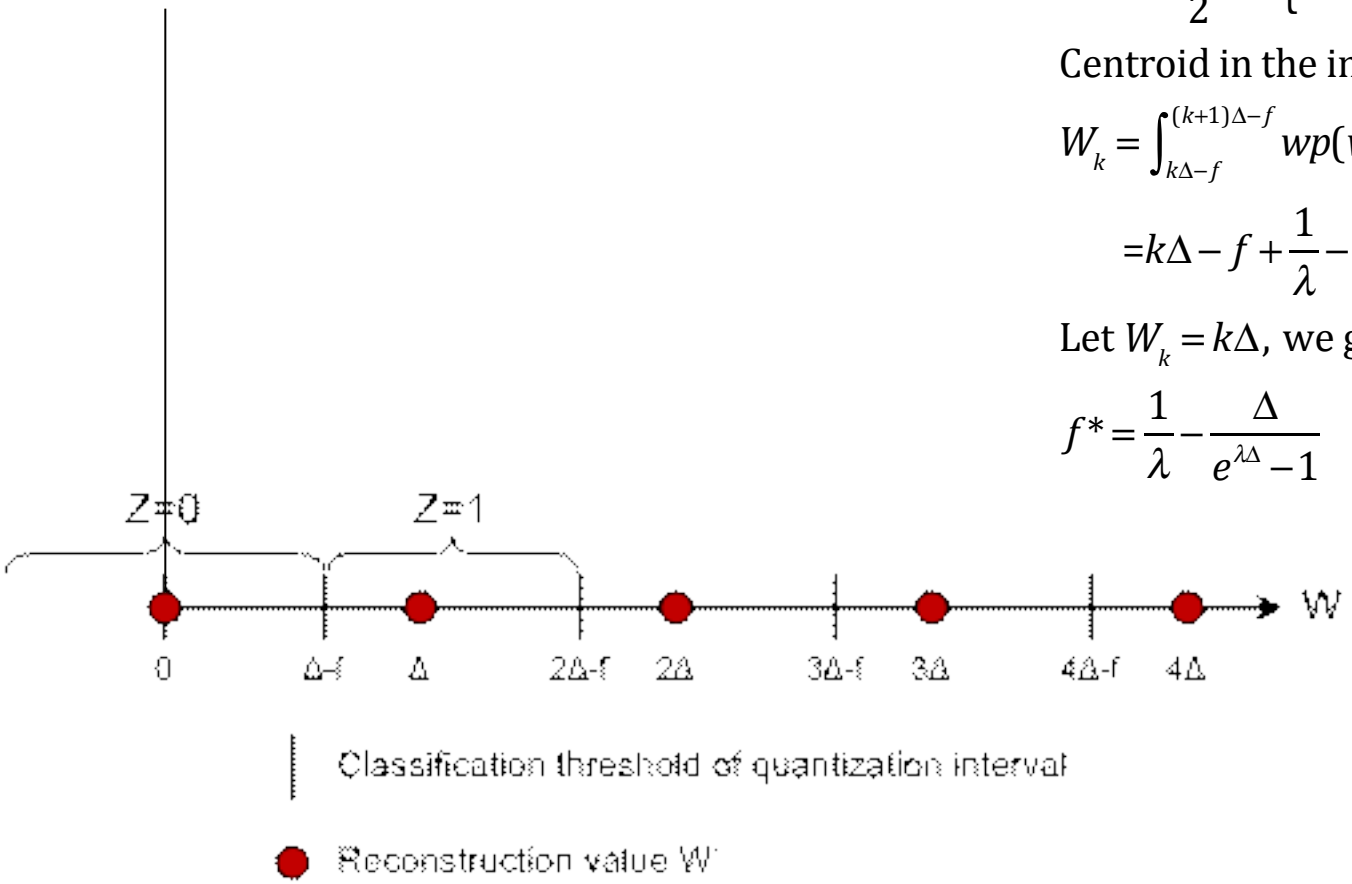
Centroid in the interval $(k\Delta-f,(k+1)\Delta-f)$:

$$W_k=\int_{k\Delta-f}^{(k+1)\Delta-f} wp(w)dw \Big/ \int_{k\Delta-f}^{(k+1)\Delta-f} p(w)dw$$

$$=k\Delta-f+\frac{1}{\lambda}-\frac{\Delta}{e^{\lambda\Delta}-1}$$

Let $W_k=k\Delta$, we get

$$f^*=\frac{1}{\lambda}-\frac{\Delta}{e^{\lambda\Delta}-1}$$



Rate Control: Why

- The coding method necessarily yields variable bit rate
- Active areas (with complex motion and/or complex texture) are hard to predict and requires more bits under the same QP
- Rate control is necessary when the video is to be sent over a constant bit rate (CBR) channel, where the rate when averaged over a short period should be constant
- The fluctuation within the period can be smoothed by a buffer at the encoder output
  - Encoded bits (variable rates) are put into a buffer, and then drained at a constant rate
  - The encoder parameter (QP, frame rate) need to be adjusted so that the buffer does not overflow or underflow

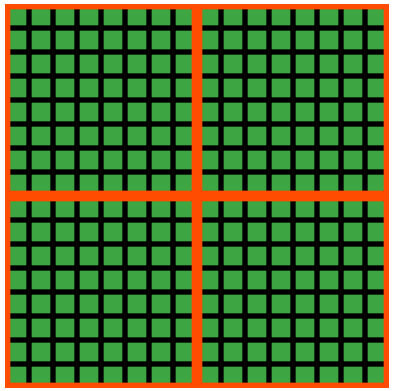
Rate Control: How

- General ideas:
  - Step 1) Determine the target rate at the frame or GOB level, based on the current buffer fullness
  - Step 2) Satisfy the target rate by varying frame rate (skip frames when necessary) and QP
    - Determination of QP requires an accurate model relating rate with Q (quantization stepsize)
    - Model used in MPEG2: R ~ A/Q + B/Q^2
- A very complex problem in practice

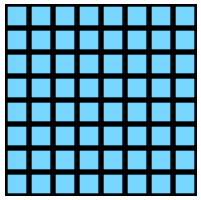
In-Loop Filtering (Deblocking)

- Errors in previously reconstructed frames (mainly blocking artifacts) accumulate over time with motion compensated temporal prediction
  - Reduce prediction accuracy
  - Increase bit rate for coding new frames
- Loop filtering:
  - Filter the reference frame before using it for prediction
  - Must be done in the same way both at the encoder and decoder (in-loop, not postprocessing outside the encoder)
  - Can be embedded in the motion compensation loop
    - Half-pel motion compensation
    - OBMC
  - Explicit deblocking filtering: removing blocking artifacts after decoding each frame
- Loop filtering can significantly improve coding efficiency
- Simple fixed filters lead to blurring!
- Complex adaptive deblocking filtering is used in H.264/HEVC

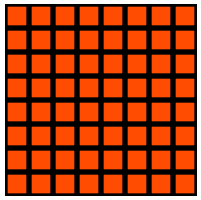
MB Structure in 4:2:0 Color Format



4 8x8 Y blocks



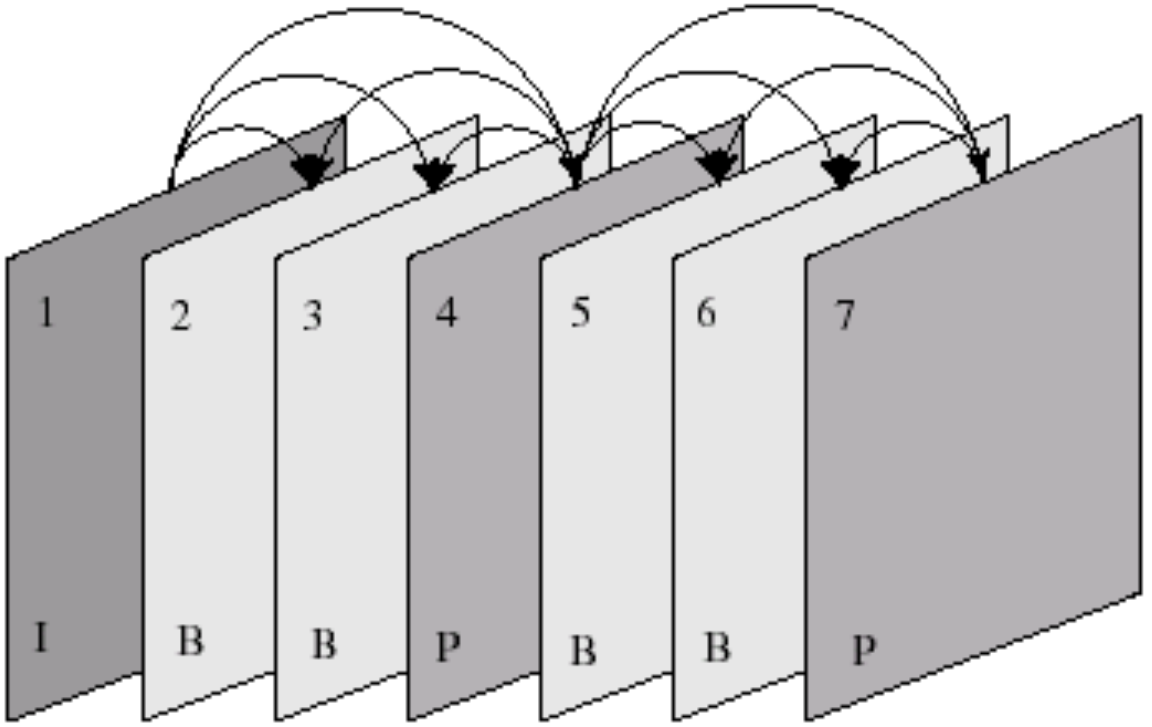
1 8x8 Cb blocks



1 8x8 Cr blocks

Typically we use the 16x16 or 8x8 Y blocks to determine the best MV and intra-prediction direction and apply the resulting MV/intra-direction to both Y and Cb and Cr.

Group of Picture Structure

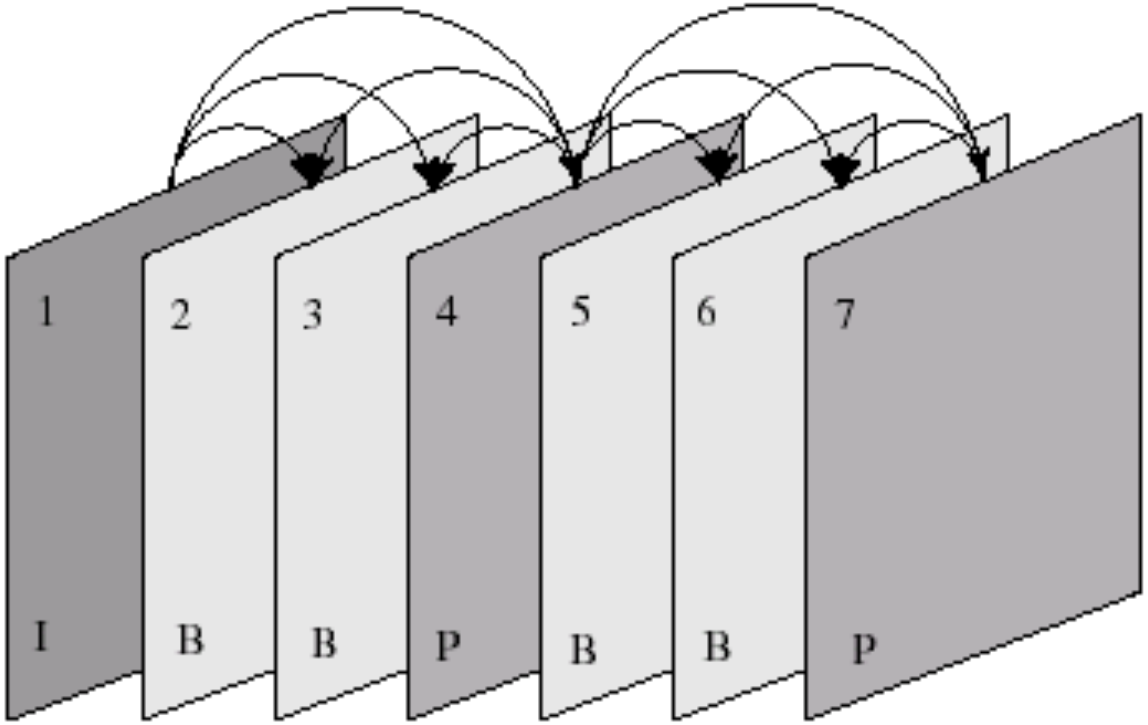


Encoding order: 1 4 2 3 7 5 6

Group-of-picture (GoP) structure

- I-frames coded without reference to other frames
  - Only intra prediction allowed
  - To enable random access (AKA channel change), fast forward, stopping error propagation
- P-frames coded with reference to previous frames
  - Uni-directional inter-prediction or intra mode
- B-frames coded with reference to previous and future frames
  - Bi-directional or uni-directional inter-prediction or intra mode
  - Highest coding efficiency
  - Requires more computation and extra delay!
  - Enable frame skip at receiver (temporal scalability)
- *Typically*, an I-frame every 1-2 sec.
- *Typically*, two B frames between each P frame
  - Compromise between compression and delay

Delay due to B-Frames



Encoding order: 1 4 2 3 7 5 6

Encoding delay=time when a frame is encoded-time when a frame arrives
First B-frame: 2 * frame interval + encoding time for 1 P and 1 B
Second B-frame: 1 * frame interval + encoding time for 1 P and 1 B
B-frame is usually not used in real-time applications (video phone/conferencing, gaming, virtual desktop, etc.)

Pseudo Code for Coding an I-frame

```
%Assume: f: current frame to be coded; N: block size for prediction, q: quantization stepsize
Function [fQ]=IframeCoding(f,q)
for (x0=1:N:height, y0=1:N:width) %for every NxN block B at (x0,y0) in f2
    B=f(x0:x0+N-1,y0:y0+N-1);
    [BI, intramode,errI]=intraPred(B,f,x0,y0);%Find best intra-prediction for B, based on previously coded
    pixels in this frame, BI is the predicted block, intramode is the chosen intra mode, errI is the prediction
    error (e.g. sum of absolute difference)
    BP=BI;
    Bits=BinaryCodingModeIFrame(intramode); %entropy coding to generate binary bits for representing the
    chosen mode and associated side information: intramode
    AppendBits(Bits); %append these bits to an existing bit stream
    BE=B-BP; %form prediction error block to be coded using the best prediction
    %transform coding of BE, assume transform is to be done at smaller block size than prediction block
    For every subblock BE_i in BE
        T=dct2(BE_i)
        TQI=quantize(T,q); %generating quantizer indices using quantization stepsize q, e.g. TQI=floor(T
        +q/2)/q
        Bits=BinaryCodingCoef(TQI); %entropy coding to generate binary bits for quantization indices
        AppendBits(Bits);
        TQ=dequantize(TQI,q); %generate quantized values from quantizer indices, e.g. TQ=TQI*q;
        BEQ_i=idct2(TQ); %reconstruct the subblock from quantized DCT coefficients
    BEQ=Assemble(BEQ_1,BEQ_2,…) %put all quantized subblocks to a larger block
    fQ(x0:x0+N-1,y0:y0+N-1)=BP+BEQ;
end; end;
```

Pseudo Code for Coding a P-frame

Assume: f1: a previous frame (already coded and decoded); f2: current frame

```
Function [f2Q]=PframeCoding(f2,f1,q)
for (x0=1:N:height, y0=1:N:width) %for every NxN block B at (x0,y0) in f2
        B=f2(x0:x0+N-1,y0:y0+N-1);
        [BI, intramode,errI]=intraPred(B,f2,x0,y0);%Find best intra-prediction for B, based on previously coded
        pixels in this frame
        [BP1,MV,errP]=EBMA(B,f2,x0,y0,f1); %Find best prediction block BP1 for B in f1, and corresponding MV
        "MV" and matching error "errP"
        [BP,mode]=modeDecisionPframe(errI,errP,intramode, MV,B,BI,BP1)
        %Choose mode based on errI,errP as well as MV,intramode,
        %In the simplest case where rates are not considered, simply compare errI and errP and choose the one
        with smaller error. If (errI<=errP) mode="intra", BP=BI, else mode="interP", BP=BP1;
        % More generally, the coder may find the bits needed and the corresponding quantization error to code in
        the intra mode, and in the inter mode and choose the one with the smallest Lagrangian cost

        Bits=BinaryCodingModePFrame(mode,intramode, MV); %entropy coding to generate binary bits for
        representing the chosen mode and associated side information: intramode or MV
        AppendBits(Bits); %append these bits to existing bit stream
        BE=B-BP; %form prediction error block to be coded using the best prediction

        %Transform coding of BE, Same as in IframeCoding( )
        …..
         f2Q(x0:x0+N-1,y0:y0+N-1)=BP+BEQ;ß
end; end;
```

Pseudo Code for Coding a B-frame

Assume: f1: a previous frame (already coded and decoded); f2: current frame; f3: a future frame (already coded and decoded), N: block size for prediction

```
Function [f2Q]=BframeCoding(f2,f1,f3,q)
for (x0=1:N:height, y0=1:N:width) %for every NxN block B at (x0,y0) in f2
        B=f2(x0:x0+N-1,y0:y0+N-1);
        [BI, intramode,errI]=intraPred(B,f2,x0,y0);%Find best intra-prediction for B, based on previously coded pixels in this frame
        [BP1,MV1,errP1]=EBMA(B,f2,x0,y0,f1); %Find best prediction block BP1 for B in f1, and corresponding MV "MV1"
        [BP2,MV2,errP2]=EBMA(B,f2,x0,y0,f3);%Find best prediction block BP2 for B in f2, and corresponding MV "MV2"
        BB=(BP1+BP2)/2; % for bi-direcitional prediction.
        errB=sum(sum(abs(B-BB)));
        [BP,mode]=modeDecisionBframe(errI,errP1,errB,intramode, MV1,MV2,B,BI,BP1,BB)
        %Choose mode based on errI,errP1,errB as well as MV1,MV2,intramode, BP is the chosen prediction.
        %For example, it can choose the one giving the least prediction error
        Bits=BinaryCodingModeBFrame(mode,intramode, MV1,MV2); %entropy coding to generate binary bits for representing the chosen mode and associated side information: intramode or MV1 and/or MV2
        AppendBits(Bits); %append these bits to existing bit stream
        BE=B-BP; %form prediction error block to be coded using the best prediction
        %Same as IframeCoding( ) for transform, quantized and code BE to generate BEQ
        …..
        f2Q(x0:x0+N-1,y0:y0+N-1)=BP+BEQ;
end; end;
```

Pseudo Code for Coding a GoP

Consider coding the following frames f1,f2,f3,f4,f5,f6,f7,f8,f9,f10, in IBBPBBPBBI … structure
Note: encoding order: f1 (I), f4(P), f2(B), f3(B), f7(P), f5(B), f6(B), …

```
[f1Q]=IframeCoding(f1,q);
[f4Q]=PframeCoding(f4,f1Q,q); %note: use f1Q for prediction!
[f2Q]=BframeCoding(f2,f1Q,f4Q,q); %node: Use f1Q and f4Q for prediction!
[f3Q]=BframeCoding(f3,f1Q,f4Q,q);

[f7Q]=PframeCoding(f7,f4Q,q);
[f5Q]=BframeCoding(f5,f4Q,f7Q,q);
[f6Q]=BframeCoding(f6,f4Q,f7Q,q);

[f10Q]=IframeCoding(f10,q);
[f8Q]=BframeCoding(f8,f7Q,f10Q,q);
[f9Q]=BframeCoding(f9,f7Q,f10Q,q)
…
```

Recommended Readings

- Reading assignment: [Wang2002] Secs. 9.2, 9.3 (sec. 9.3.2 on OBMC optional)
- Optional Reading:
  - T. Wiegand et al., “Overview of the H.264/AVC video coding standard,”, IEEE Trans. Circuits and Systems for Video Technology, special issue on H.264, July 2003.
  - T. Wiegand and B. Girod, “LAGRANGE MULTIPLIER SELECTION IN HYBRID VIDEO CODER CONTROL,” Proc. ICIP2001
  - Deadzone quantization: http://www.h265.net/2009/06/quantization-techniques-in-jmkta-part-2.html
  - http://www.h265.net/2009/06/quantization-techniques-in-jmkta-part-3.html
  - Sullivan, Gary J. "Efficient scalar quantization of exponential and Laplacian random variables." Information Theory, IEEE Transactions on 42.5 (1996): 1365-1374.
  - Sullivan, Gary J., and Shijun Sun. "On dead-zone plus uniform threshold scalar quantization." Visual Communications and Image Processing 2005. International Society for Optics and Photonics, 2005.

Written Homework (1)

1. Prob. 9.7 in [Wang2002]

2. Consider a coding method that codes two frames as a group. Frame n is coded directly (as an I-frame), and frame n+1 is predicted from frame n and the error is coded (i.e. as a P-frame). Let a pixel in frame n be denoted by *f1*, and its corresponding pixel in frame n+1 be denoted by *f2* . f1 is directly quantized and coded, f2 is predicted from f1 using fp=f1, and the prediction error *e= f2 –fp* is quantized and coded. (for this problem, for simplicity, assume that your prediction is based on the original f1, not the quantized f1). Assume that each pixel has zero mean and variance σ_s^2 and that the correlation coefficient between two corresponding pixels in two adjacent frames is ρ. Furthermore, assume the rate-distortion function for coding a single variable (original pixel or the prediction error) can be expressed as $D(R) = \varepsilon^2 \sigma^2 2^{-2R}$ where σ^2 is the variance of the variable being coded. (a) Determine the variance of the prediction error. (b) Suppose we want the average bit rate to be R (bits/pixel). How many bits you should use for *f1* and *e* respectively, to minimize the mean square error (MSE) of the reconstructed pixels? (You could assume the allocated bits can be non-integer or even negative). What would be the corresponding minimal MSE? (c) How does this method compare with coding each frame directly (as an I-frame)? Specifically, when the bit rate is the same, which one gives lower reconstruction error ?

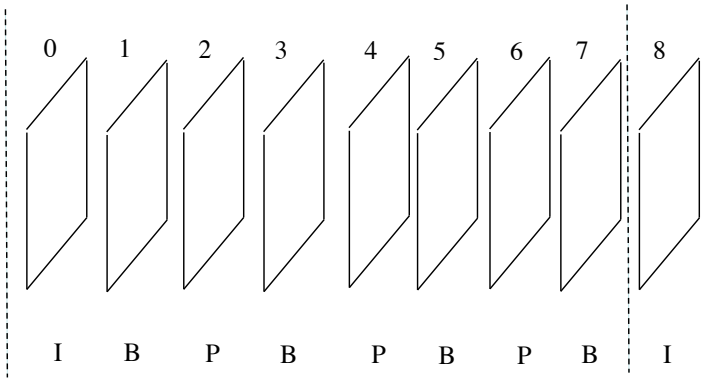
Written Homework (2)

3. Answer the following questions about I, P, B frames:

a) In video coding, a frame is often coded as either an I-frame, a P-frame, or a B-frame. Rank these modes in terms of coding efficiency and complexity, respectively. What are some of the difficulty caused by using P-frame and B-frame, in spite of their efficiency? (List one difficulty for each).

b) A video coder often divides successive frames into separate Groups of Pictures (GOP), and each GOP contains an I-frame, some P-frames and some B-frames. For the GOP structure illustrated below, what is the encoding order? What is the decoding order?

c) Assume the video frame rate is 30 frame/sec, and that encoding each frame takes 10 ms. What is the delay in millisecond at the encoder (time difference between when a frame arrives at the encoder and the time the frame is encoded) for coding an I, P, and B frames, respectively? Suppose transmitting each frame takes 100 ms, and decoding each frame takes 10 ms. What is the minimal playback delay (the time difference between when the first frame arrives at the encoder and the time the first frame should be displayed) the decoder should set, to ensure smooth display of decoded video?



Computer Assignment

1. Write a MATLAB code that implements a basic form of the block-based hybrid video coder for coding a P-frame. For simplicity, consider only intra-prediction (using only the first 3 intra prediction modes shown in Slide ?), and unidirectional inter-prediction (with either integer or half-pel accuracy EBMA, with a search range of +/-24). You program should do the following for each 16x16 block: i) find the best intra-prediction mode and the corresponding error block and its variance; ii) find the best MV for inter-prediction and the corresponding error block and its variance; iii) Choose the prediction block whose prediction error has the smaller variance; iv) Perform 8x8 DCT on each of the 4 8x8 subblocks of the 16x16 prediction error block of the chosen mode; v) Quantize all the DCT coefficients with the same quantization stepsize q; vi) Count how many non-zero coefficients you have after quantization, vii) Reconstruct the error block by performing inverse DCT on quantized DCT coefficients; viii) Reconstruct the original block by adding the reconstructed error block into the best prediction block. Instead of developing a real entropy coder, we will use the total number of non-zero DCT coefficients as an estimate of the bit rate and ignore the bits needed to code the side information (mode info, motion vector, etc.). Your program should determine the PSNR of the reconstructed image (compared to the original image) and the total number of non-zero quantized DCT coefficients K, for a given quantization stepsize q. Apply your program to two frames of a video for several different q with a large range and determine the corresponding PSNR and K for each q. Plot PSNR vs. K as your approximate PSNR vs rate curve. Note: if your video has very little motion, you may want to select two frames that are several frames apart).
2. (Optional) Develop a MATLAB code for coding a sequence of frames, with a GOP structure of IBPBPBP. For I-frame, you should use intra-prediction only. For P-frame, you choose between intra and uni-directional inter only. For B-frame, you choose between intra, uni-directional inter, and bi-directional inter prediction. Record the average PSNR and number of non-zero DCT coefficients K for each frame type (I, P, and B) for the same q. Repeat for different q. Plot PSNR vs. K curves for different frame types as separate curves. You should observe that to achieve the same PSNR, I frame will require the highest K, followed by P, and then by B.