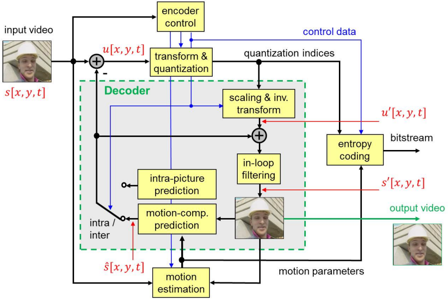
Source Coding and Compression

Heiko Schwarz



Contact:

Dr.-Ing. Heiko Schwarz
heiko.schwarz@hhi.fraunhofer.de

Outline

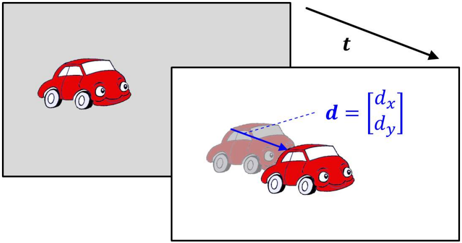
Part I: Source Coding Fundamentals

- Probability, Random Variables and Random Processes
- Lossless Source Coding
- Rate-Distortion Theory
- Quantization
- Predictive Coding
- Transform Coding

Part II: Application in Image and Video Coding

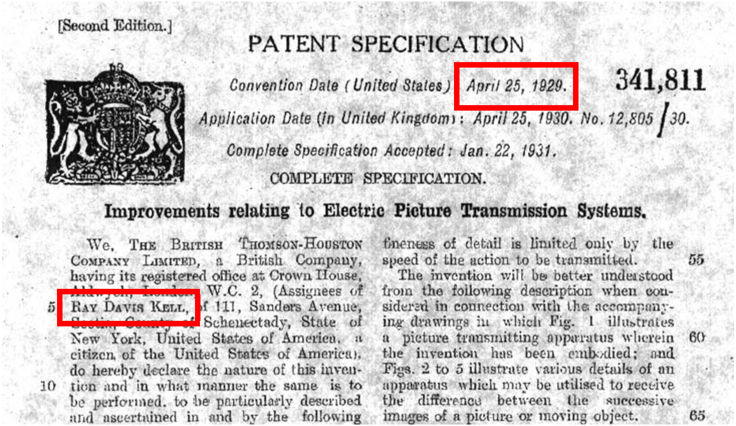
- Still Image Coding / Intra-Picture Coding
- **Hybrid Video Coding (From MPEG-2 Video to H.265/HEVC)**
  - Motion-compensated Prediction & Hybrid Video Coding
  - Encoder Control
  - Video Coding Standards

Motion-Compensated Prediction and Hybrid Video Coding



Similarities between Successive Pictures in a Video Sequence

General Idea of Hybrid Video Coding



[Second Edition.]

PATENT SPECIFICATION

Convention Date (United States): April 25, 1929. **341,811**

Application Date (in United Kingdom): April 25, 1930. No. 12,805 / 30.

Complete Specification Accepted: Jan. 22, 1931.

COMPLETE SPECIFICATION.

Improvements relating to Electric Picture Transmission Systems.

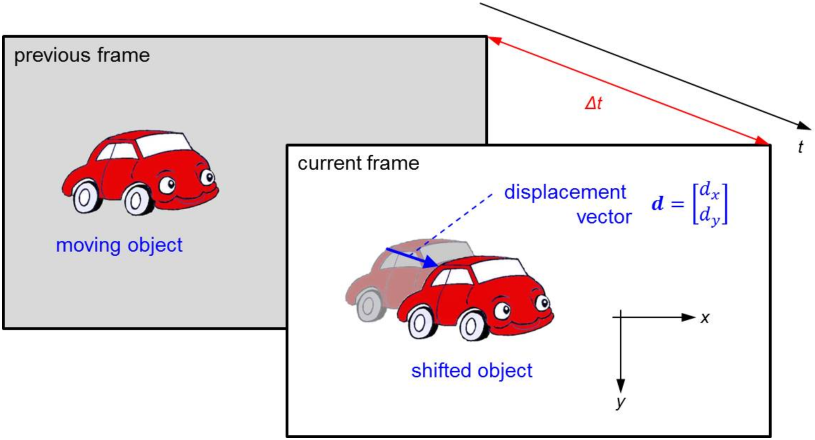
We, The British Thomson-Houston Company Limited, a British Company, having its registered office at Crown House, Aldwych, London, W.C. 2, (Assignees of Ray Davis Kell, of 111, Sanders Avenue, Scotia, County of Schenectady, State of New York, United States of America, a citizen of the United States of America), do hereby declare the nature of this invention and in what manner the same is to be performed, to be particularly described and ascertained in and by the following

fineness of detail is limited only by the speed of the action to be transmitted.

The invention will be better understood from the following description when considered in connection with the accompanying drawings in which Fig. 1 illustrates a picture transmitting apparatus wherein the invention has been embodied; and Figs. 2 to 5 illustrate various details of an apparatus which may be utilised to receive the difference between the successive images of a picture or moving object.

**"It has been customary in the past to transmit successive complete images of the transmitted picture. [· · ·]
In accordance with this invention, this difficulty is avoided by transmitting only the difference between successive images of the object."**

Principle of Motion-Compensated Prediction



Prediction for luma signal $s[x, y, t]$ within the moving object:

$$\hat{s}[x, y, t] = s'(x - d_x, y - d_y, t - \Delta t) \quad (612)$$

Motion-Compensated Hybrid Video Coding

Hybrid video coding

- Combination of two techniques:
  - Motion-compensated prediction
  - Transform coding of prediction error
- All ITU-T and ISO/IEC video coding standards follow that principle

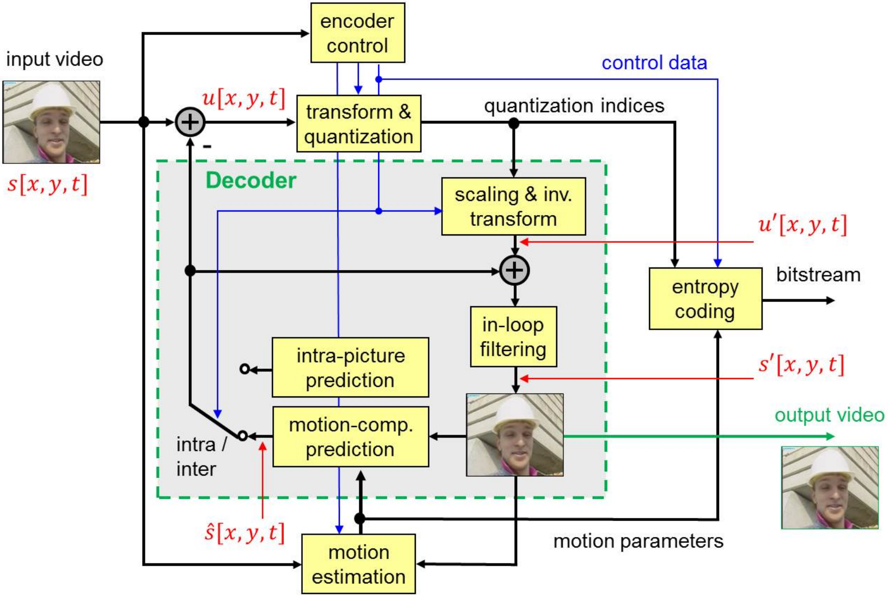
Motion-compensated prediction

- Key technique for video coding
- Substantial bit rate reduction compared to intra-picture coding

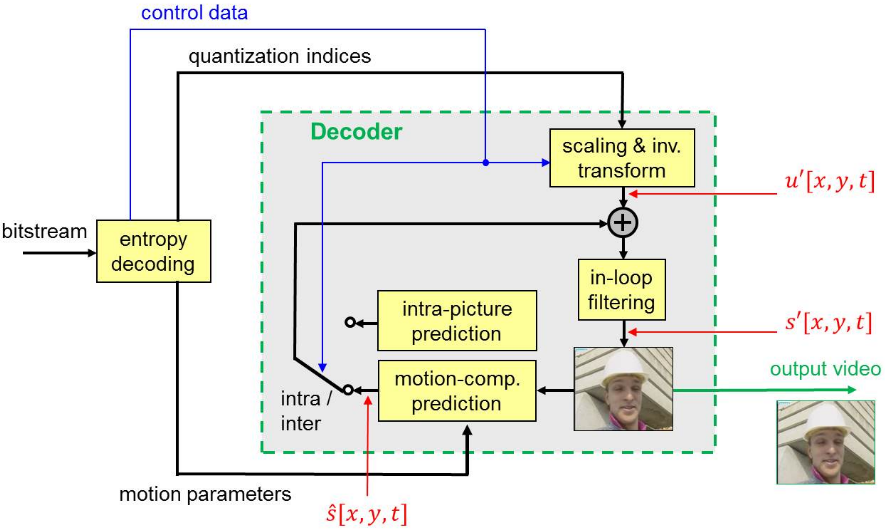
Practical hybrid video coding

- Not all parts of a picture can be efficiently predicted from a reference picture
  - Not all changes between pictures are caused by motion
  - Some parts can be occluded in reference picture
  - Complicated motion cannot be well compensated by used motion model
- For some parts, motion-compensated prediction could reduce coding efficiency
- Practical hybrid video coders allow to switch between motion-compensated prediction and intra-picture prediction

Structure of a Motion-Compensated Hybrid Video Encoder



Structure of a Motion-Compensated Hybrid Video Decoder

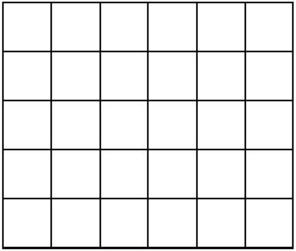


Example for Motion-Compensated Prediction

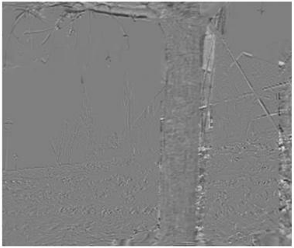
previous rec. frame $s'[x,y,t-1]$

current frame $s[x,y,t]$

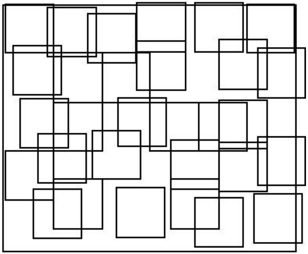
partitioning of current frame



prediction signal $\hat{s}[x,y,t]$ with motion vectors



prediction error signal $u[x,y,t] = s[x,y,t] - \hat{s}[x,y,t]$



location of reference blocks in previous frame

Design of Motion-Compensated Hybrid Codecs

Accuracy of motion parameters

- Full-sample or sub-sample accurate motion vectors (or motion parameters)
- For sub-sample accuracy, an interpolation filter is required

Motion models for describing the motion inside a region

- Simplest model: Translational motion $\Longrightarrow$ Used in all video coding standards
- Higher-order parametric motion models (e.g., affine motion model)

Selection of regions with constant motion (using same motion model)

- In principle, regions can have arbitrary shape $\Longrightarrow$ Need to transmit partitioning
- In today's coding standards: Square or rectangular blocks (fixed or variable size)

Selection of reference picture

- Always use previously coded/decoded picture
- Select one picture out of a set of previously coded/decoded pictures

Number of motion hypotheses

- Predict a region in a current frame using a single motion hypothesis, i.e., one reference picture with one motion vector (or motion parameter set)
- Weighted prediction of multiple motion hypotheses

Theoretical Performance Analysis of Hybrid Video Coding

Goal of analysis

- Approximate analysis of efficiency of motion-compensated video coding
- Approximate analysis of impact of displacement vector accuracy

Models for performance analysis

- Very simple signal model
- Consider coding at high bit rates
- Assume r-d optimal intra-picture coding using Gaussian model
- No consideration of bit rate for motion parameters

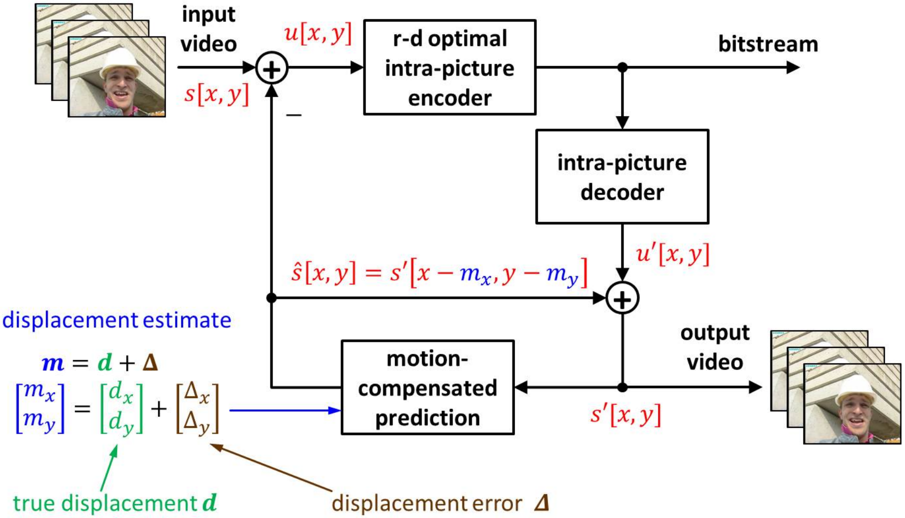
Further reading (papers include extended models)

[1] B. Girod, "The Efficiency of Motion-Compensating Prediction for Hybrid Coding of Video Sequences," *IEEE Journal on Selected Areas in Communications*, vol. SAC-5, no. 7, pp. 1140-1154, Aug. 1987.

[2] B. Girod, "Motion-Compensating Prediction with Fractional-Pel Accuracy," *IEEE Trans. on Communications*, vol. 41, no. 4, pp. 604-612, Apr. 1993.

[3] B. Girod, "Efficiency Analysis of Multihypothesis Motion-Compensated Prediction for Video Coding," *IEEE Trans. on Image Processing*, vol. 9, no. 2, pp. 173-183, Feb. 2000.

Model for Performance Analysis of Hybrid Video Coding



Model for Temporal Dependencies in Video Sequences

Continuous signal model

- Displaced signal with additive white noise

$$s_t(x,y) = s_{t-1}(x-d_x, y-d_y) + n^*(x,y)$$

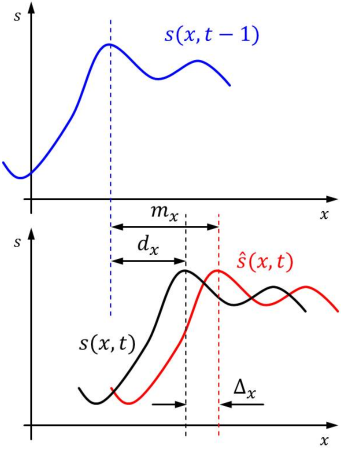
- Motion-compensated prediction

$$\hat{s}_t(x,y) = s'_{t-1}(x-m_x, y-m_y)$$

- High-rate approx.: $s'_{t-1}(x,y) = s_{t-1}(x,y)$

$$\begin{aligned}\hat{s}_t(x,y) &= s_{t-1}(x-d_x-\Delta_x, y-d_y-\Delta_y)\\ &= s_t(x-\Delta_x, y-\Delta_y) - n(x,y)\end{aligned}$$

with $n(x,y) = n^*(x-\Delta_x, y-\Delta_y)$ being the shifted noise term (same statistical properties)



- Resulting prediction error signal for current frame (omitting time index t)

$$\begin{aligned}u(x,y) &= s(x,y) - \hat{s}(x,y) = s(x,y) - s(x-\Delta_x, y-\Delta_y) + n(x,y)\\ &= s(x,y) * \big(\delta(x,y) - \delta(x-\Delta_x, y-\Delta_y)\big) + n(x,y) \qquad (613)\end{aligned}$$

Approximation of Rate-Distortion Functions

Gaussian model for signal $s(x,y)$ and prediction error $u(x,y)$

- Signal $s(x,y)$: Gaussian model provides reasonable approximation
- Prediction error $u(x,y)$: Gaussian model provides upper bound for r-d function
- Remember: Rate-distortion function $R(D)$ for 1-d Gaussian process $s(x)$ with power spectral density $\Phi_{ss}(\omega)$ is given by parametric formulation

$$D(\theta) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \min\left(\theta, \Phi_{ss}(\omega)\right) \, \mathrm{d}\omega \qquad (614)$$

$$R(\theta) = \frac{1}{2\pi} \int_{-\pi}^{\pi} \max\left(0, \frac{1}{2}\log_2 \frac{\Phi_{ss}(\omega)}{\theta}\right) \, \mathrm{d}\omega \qquad (615)$$

- Extension to 2-d signal $s(x,y)$ with power spectral density $\Phi_{ss}(\omega_x, \omega_y)$

$$D(\theta) = \frac{1}{4\pi^2} \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \min\left(\theta, \Phi_{ss}(\omega_x, \omega_y)\right) \, \mathrm{d}\omega_x \, \mathrm{d}\omega_y \qquad (616)$$

$$R(\theta) = \frac{1}{4\pi^2} \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \max\left(0, \frac{1}{2}\log_2 \frac{\Phi_{ss}(\omega_x, \omega_y)}{\theta}\right) \, \mathrm{d}\omega_x \, \mathrm{d}\omega_y \qquad (617)$$

Conditional Power Spectral Density of Prediction Error

Power spectral density for given displacement error $\boldsymbol{\Delta} = [\Delta_x, \Delta_y]^{\mathrm{T}}$

- Prediction error signal

$$\begin{aligned} u(x,y) &= s(x,y) * \big(\delta(x,y) - \delta(x-\Delta_x, y-\Delta_y)\big) + n(x,y) \\ &= s(x,y) * f(x,y) + n(x,y) = v(x,y) + n(x,y) \end{aligned} \quad (618)$$

- Use vector notation $\boldsymbol{\omega} = [\omega_x, \omega_y]^{\mathrm{T}}$ and $\boldsymbol{\Delta} = [\Delta_x, \Delta_y]^{\mathrm{T}}$
- Power spectral density $\Phi_{uu}(\boldsymbol{\omega} \,|\, \boldsymbol{\Delta})$ for given displacement error $\boldsymbol{\Delta}$

$$\begin{aligned} \Phi_{uu}(\boldsymbol{\omega} \,|\, \boldsymbol{\Delta}) &= \Phi_{vv}(\boldsymbol{\omega} \,|\, \boldsymbol{\Delta}) + \Phi_{nn}(\boldsymbol{\omega}) \\ &= \Phi_{ss}(\boldsymbol{\omega}) \cdot F(\boldsymbol{\omega}) \cdot F^*(\boldsymbol{\omega}) + \Phi_{nn}(\boldsymbol{\omega}) \\ &= \Phi_{ss}(\boldsymbol{\omega}) \cdot \left(1 - e^{-j\boldsymbol{\omega}\boldsymbol{\Delta}^{\mathrm{T}}}\right)\left(1 - e^{j\boldsymbol{\omega}\boldsymbol{\Delta}^{\mathrm{T}}}\right) + \Phi_{nn}(\boldsymbol{\omega}) \\ &= 2 \cdot \Phi_{ss}(\boldsymbol{\omega}) \cdot \left(1 - \frac{e^{-j\boldsymbol{\omega}\boldsymbol{\Delta}^{\mathrm{T}}} + e^{j\boldsymbol{\omega}\boldsymbol{\Delta}^{\mathrm{T}}}}{2}\right) + \Phi_{nn}(\boldsymbol{\omega}) \\ &= 2 \cdot \Phi_{ss}(\boldsymbol{\omega}) \cdot \big(1 - \cos(\boldsymbol{\omega}\boldsymbol{\Delta})\big) + \Phi_{nn}(\boldsymbol{\omega}) \\ &= 2 \cdot \Phi_{ss}(\boldsymbol{\omega}) \cdot \left(1 - \Re\left\{e^{-j\boldsymbol{\omega}\boldsymbol{\Delta}}\right\}\right) + \Phi_{nn}(\boldsymbol{\omega}) \end{aligned} \quad (619)$$

Power Spectral Density of Prediction Error

Power spectral density $\Phi_{uu}(\boldsymbol{\omega})$ depends on displacement error pdf

- Power spectral density $\Phi_{uu}(\boldsymbol{\omega})$ of prediction error

$$\begin{aligned}\Phi_{uu}(\boldsymbol{\omega}) &= E\{\Phi_{uu}(\boldsymbol{\omega}\,|\,\boldsymbol{\Delta})\} \\ &= 2\cdot\Phi_{ss}(\boldsymbol{\omega})\cdot\left(1-\Re\left\{E\left\{e^{-j\boldsymbol{\omega}\boldsymbol{\Delta}}\right\}\right\}\right)+\Phi_{nn}(\boldsymbol{\omega}) \\ &= 2\cdot\Phi_{ss}(\boldsymbol{\omega})\cdot(1-\Re\{P(\boldsymbol{\omega})\})+\Phi_{nn}(\boldsymbol{\omega})\end{aligned} \quad (620)$$

- The spectrum $P(\boldsymbol{\omega})$ is the Fourier transform of the probability density function $f_{\boldsymbol{\Delta}}(\boldsymbol{\Delta})$ for the displacement error

$$\begin{aligned}P(\boldsymbol{\omega}) &= E\left\{e^{-j\boldsymbol{\omega}\boldsymbol{\Delta}}\right\} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{\boldsymbol{\Delta}}(\boldsymbol{\Delta})\, e^{-j\boldsymbol{\omega}\boldsymbol{\Delta}}\,\mathrm{d}\boldsymbol{\Delta} \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} f_{\boldsymbol{\Delta}}(\Delta_x,\Delta_y)\, e^{-j(\omega_x\Delta_x+\omega_y\Delta_y)}\,\mathrm{d}\Delta_x\,\mathrm{d}\Delta_y\end{aligned} \quad (621)$$

Model for Distribution of Displacement Error

Motion estimate with given maximum accuracy

- Maximum displacement error is given by motion vector accuracy

$$\Delta_{\max} = 2^{-(1+\beta)} \qquad (622)$$

with
$\beta = 0$: Integer-sample accurate motion vectors
$\beta = 1$: Half-sample accurate motion vectors
$\beta = 2$: Quarter-sample accurate motion vectors

- Displacement error components are uniformly distributed inside $[-\Delta_{\max}, \Delta_{\max}]$

$$f_{\boldsymbol{\Delta}}(\Delta_x, \Delta_y) = \begin{cases} \frac{1}{4}\,\Delta_{\max}^{-2} & : \; |\Delta_x| \le \Delta_{\max} \text{ and } |\Delta_y| \le \Delta_{\max} \\ 0 & : \; \text{otherwise} \end{cases} \qquad (623)$$

- Resulting spectrum $P(\omega_x, \omega_y)$

$$\begin{aligned} P(\omega_x, \omega_y) &= \frac{1}{4 \cdot \Delta_{\max}^2} \int_{-\Delta_{\max}}^{\Delta_{\max}} \int_{-\Delta_{\max}}^{\Delta_{\max}} e^{-j\omega_x \Delta_x} \cdot e^{-j\omega_y \Delta_y} \, \mathrm{d}\Delta_x \, \mathrm{d}\Delta_y \\ &= \operatorname{sinc}\big(\Delta_{\max} \cdot \omega_x\big) \cdot \operatorname{sinc}\big(\Delta_{\max} \cdot \omega_y\big) \\ &= \operatorname{sinc}\big(2^{-(1+\beta)} \cdot \omega_x\big) \cdot \operatorname{sinc}\big(2^{-(1+\beta)} \cdot \omega_y\big) \end{aligned} \qquad (624)$$

Model for Image Signal

Model based on assumption of autocorrelation function $R_{ss}(\Delta_x, \Delta_y)$

- Isotropic autocorrelation function (for typical image signals: $\varrho \approx 0.9$)

$$R_{ss}(\Delta_x, \Delta_y) = E\{s(x,y) \cdot s(x-\Delta_x, y-\Delta_y)\} = \sigma_s^2 \cdot \varrho^{\sqrt{\Delta_x^2 + \Delta_y^2}} \qquad (625)$$

- Power spectral density $\Phi_{ss}(\omega_x, \omega_y)$ for image signal

$$\begin{aligned}\Phi_{ss}(\omega_x, \omega_y) &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} R_{ss}(\Delta_x, \Delta_y) \cdot e^{-j(\omega_x \Delta_x + \omega_y \Delta_y)} \, \mathrm{d}\Delta_x \, \mathrm{d}\Delta_y \\ &= K \cdot \sigma_s^2 \cdot \left(1 + \frac{\omega_x^2 + \omega_y^2}{(\ln \varrho)^2}\right)^{-\frac{3}{2}} \qquad (626)\end{aligned}$$

- Consider band-limited image signal (sampled at Nyquist rate)

$$\Phi_{ss}(\omega_x, \omega_y) = \begin{cases} K \cdot \sigma_s^2 \cdot \left(1 + \frac{\omega_x^2 + \omega_y^2}{(\ln \varrho)^2}\right)^{-\frac{3}{2}} & : \quad |\omega_x| \le \pi \text{ and } |\omega_y| \le \pi \\ 0 & : \quad \text{otherwise} \end{cases} \qquad (627)$$

where the constant K has to be determined by

$$\sigma_s^2 = \frac{1}{4\pi^2} \int_{-\pi}^{\pi}\int_{-\pi}^{\pi} \Phi_{ss}(\omega_x, \omega_y) \, \mathrm{d}\omega_x \, \mathrm{d}\omega_y \qquad (628)$$

Complete Model for Power Spectral Density of Prediction Error

Apply models for displacement error pdf and image signal

- Remember

$$\Phi_{uu}(\omega_x,\omega_y) = 2\cdot\Phi_{ss}(\omega_x,\omega_y)\cdot\left(1-\Re\{P(\omega_x,\omega_y)\}\right)+\Phi_{nn}(\omega_x,\omega_y) \qquad (629)$$

- Assume constant noise $n(x,y)$ inside $[-\pi,\pi]\times[-\pi,\pi]$

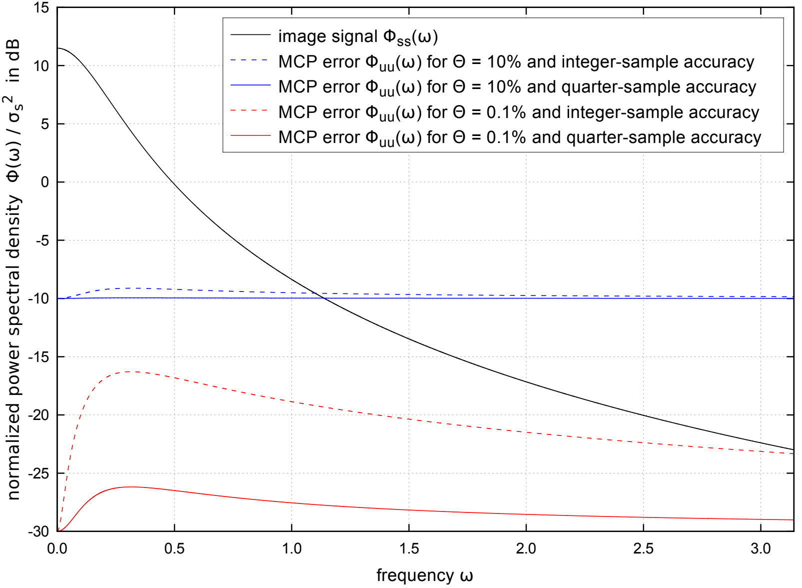
$$\Phi_{nn}(\omega_x,\omega_y) = \sigma_n^2 = \Theta\cdot\sigma_s^2 \qquad (630)$$

  with Θ being the ratio of noise and signal power

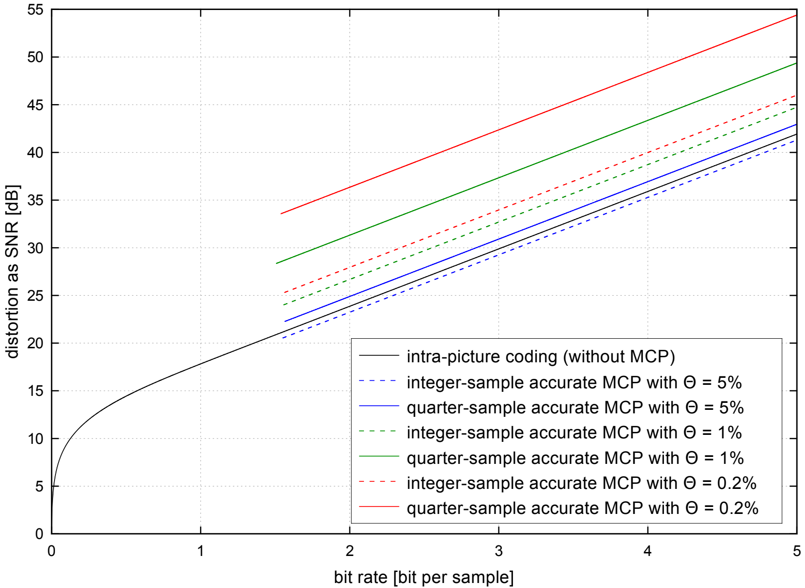
- Inserting the models for $\Phi_{ss}(\omega_x,\omega_y)$, $\Phi_{nn}(\omega_x,\omega_y)$ and $P(\omega_x,\omega_y)$ yields the normalized power spectral density inside interval $[-\pi,\pi]\times[-\pi,\pi]$

$$\frac{\Phi_{uu}(\omega_x,\omega_y)}{\sigma_s^2} = K\cdot\left(1+\frac{\omega_x^2+\omega_y^2}{(\ln\varrho)^2}\right)^{-\frac{3}{2}}\cdot \left(1-\text{sinc}(2^{-(1+\beta)}\cdot\omega_x)\cdot\text{sinc}(2^{-(1+\beta)}\cdot\omega_x)\right)+\Theta \qquad (631)$$

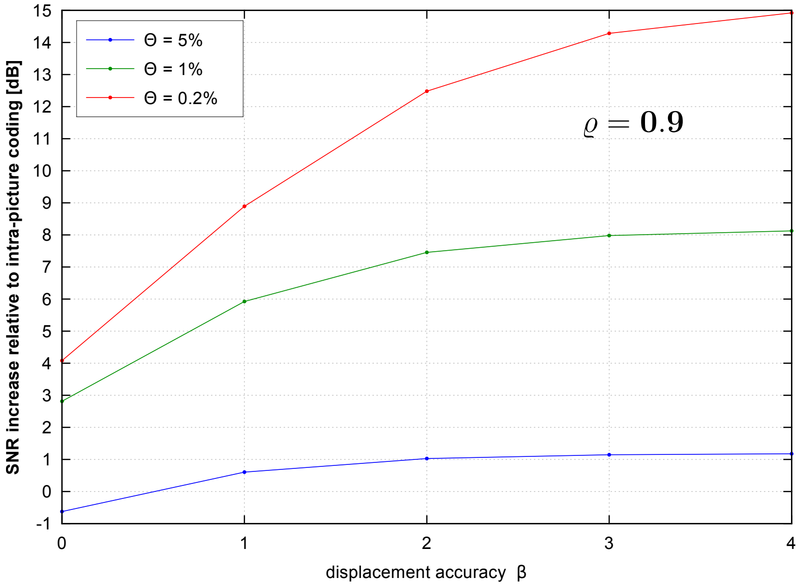
Power Spectral Densities for 1-D Signal with $\varrho = 0.8$



High-Rate Performance of MCP for 2-d Signals with $\varrho = 0.9$



Impact of Displacement Accuracy and Noise at High Rates



Interpretation of Theoretical Results

Theoretical analysis showed

- Motion-compensated prediction typically improves coding efficiency
- Efficiency of motion-compensated prediction increases with increasing accuracy of displacement vectors
- Accuracy increase is mainly useful for video signals with low noise

Motion-compensated prediction in practice

- Bit rate required for transmitting displacement vectors increases with increasing displacement vector accuracy
  - $\Longrightarrow$ There is an "optimal" displacement vector accuracy for a given noise level
  - $\Longrightarrow$ For typical sequences, an accuracy higher than quarter-sample displacements does not provide noticeable coding efficiency gains
    (for low noise data: eight-sample accuracy can provide gain)
- Interpolation filters are required for sub-sample accurate MCP
  - $\Longrightarrow$ Interpolation filters have a significant impact on coding efficiency
  - $\Longrightarrow$ More accurate interpolation filters require higher complexity

Displacement Vector Accuracy in Video Coding Standards

H.262 | MPEG-2 Video, H.263 and MPEG-4 Visual (Version 1)

- Half-sample accurate displacement vectors
- Prediction signal at half-sample positions is obtained by bi-linear interpolation

Advanced Simple profile of MPEG-4 Visual

- Quarter-sample accurate displacement vectors
- Prediction signal at half-sample positions: Separable 8-tap FIR filter
- Prediction signal at quarter-sample positions: Bi-linear interpolation of integer- and half-sample positions

H.264 | MPEG-4 AVC

- Quarter-sample accurate displacement vectors
- Prediction signal at half-sample positions: Separable 6-tap FIR filter
- Prediction signal at quarter-sample positions: Averaging of two integer- and half-sample positions

H.265 | MPEG-H HEVC

- Quarter-sample accurate displacement vectors
- Prediction signal at half- and quarter sample positions: Separable 8- and 7-tap FIR filter (depending on sub-sample shift)

Experimental Analysis of MCP and Displacement Accuracy

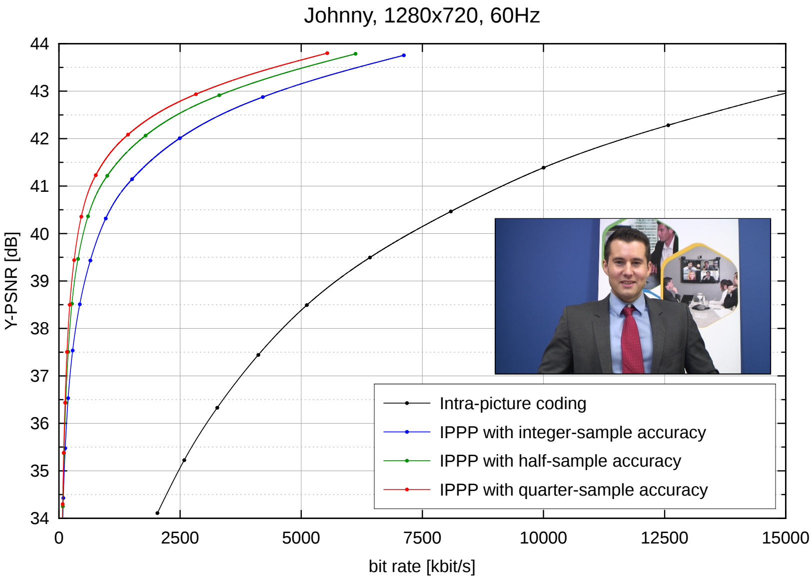
Comparison of different codecs

- HEVC intra-only coding
- HEVC inter coding (quarter-sample accuracy)
- modified HEVC with half-sample accuracy (adapted motion vector coding)
- modified HEVC with integer-sample accuracy (adapted motion vector coding)

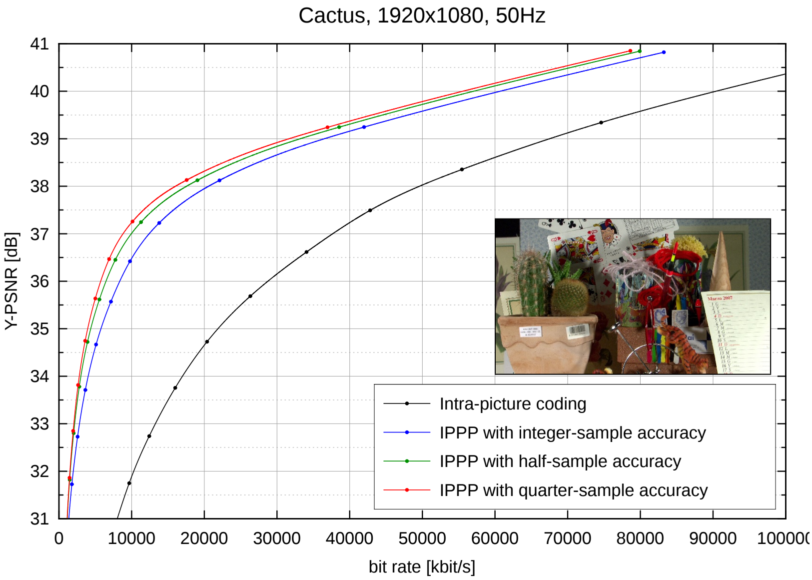
Configuration

- 12 test sequences
  - 6 sequences in 720p with video conferencing content
  - 6 sequences in 1080p with entertainment content
- Only the first picture is intra-picture coded (except for intra-only coding)
- Picture are coded in display order (IPPP coding structure)
- Previous picture is used as reference picture
- Intra blocks are allowed in inter-picture coded frames
- Fixed quantization parameter
- All codecs are based on same HEVC encoder version
- Lagrangian encoder control

Efficiency of Motion-Compensated Prediction – "Johnny"



Efficiency of Motion-Compensated Prediction – “Cactus”



Efficiency of Motion-Compensated Prediction – Summary

Comparison of HEVC intra-picture coding and HEVC-based motion-compensated coding with different motion vector accuracy

- Bit-rate saving at a PSNR value is obtained by interpolating the r-d curves
- Average bit-rate savings are obtained by averaging the savings for 100 PSNR values
- Average bit-rate savings for all sequences are summarized below

	average bit rate savings relative to ...		
codec version	**half-sample**	**integer-sample**	**intra-picture**
quarter-sample	**10.3 %**	**29.9 %**	**77.7 %**
half-sample		**22.8 %**	**75.6 %**
integer-sample			**69.7 %**

Motion Models for Motion-Compensated Prediction

Translational motion in image plane

- Motion of a region is described by 2-d displacement vector $\mathbf{d} = [d_x, d_y]^{\mathrm{T}}$

$$\boldsymbol{d}(x, y) = \boldsymbol{d} = \text{const} \tag{632}$$

- Used in all video coding standards of ITU-T and ISO/IEC
- Can only describe small amount of "real motion"

Higher order motion models

- Motion in image plane is caused by motion in 3-d space
- Assuming reasonable restrictions for the motion in 3-d space (e.g. rigid body motion), motion in image plane can be described by a parametric model

$$\boldsymbol{d}(x, y) = f(\boldsymbol{a}, x, y) \qquad \text{with } \boldsymbol{a} \text{ being a constant parameter vector} \tag{633}$$

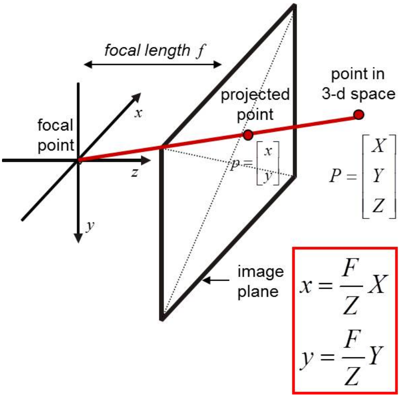
- Advantage of higher order motion models
  $\Longrightarrow$ Better approximation of "real motion" than translational model
- Disadvantages of higher order motion models
  $\Longrightarrow$ Increased bit rate for transmitting motion parameters
  $\Longrightarrow$ Increased complexity and decreased robustness of motion estimation

Models for Projection of 3-D Space onto Image Plane

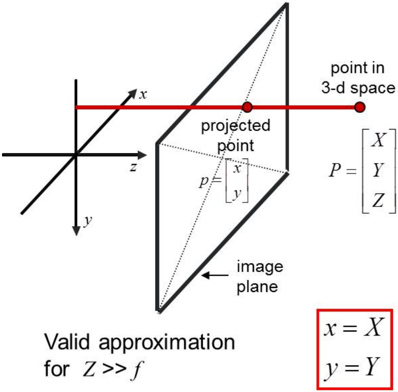
Projection of 3-d world onto 2-d image plane by camera lens

- **Perspective projection**: Neglecting image distortions and blurring
- **Orthographic projection**: All rays are parallel to image plane (valid for $Z \gg f$)

perspective projection model



orthographic projection model



Perspective Motion Model

Perspective model for motion in image plane

- Mathematical model for motion field $\boldsymbol{d} = [d_x, d_y]^{\mathrm{T}}$ inside a region

$$d_x(x,y) = \frac{a_0 + a_1 \cdot x + a_2 \cdot y}{1 + c_1 \cdot x + c_2 \cdot y} \qquad (634)$$

$$d_y(x,y) = \frac{b_0 + b_1 \cdot x + b_2 \cdot y}{1 + c_1 \cdot x + c_2 \cdot y} \qquad (635)$$

- Assumptions / restrictions:
  - Rotation and scaling of rigid body in 3-d space, but no translation
  - Translation, rotation and scaling of a plane in 3-d space
- Advantage:
  - Corresponds to perspective projection model
- Disadvantage:
  - Hyperbolic model is non-linear with respect to motion parameters
  - Difficult to estimate

Orthographic Motion Models

Motion models with different degrees of freedom

- **Translational model**: Translation parallel to image plane (or in image plane)

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} a_0 \\ b_0 \end{bmatrix} \tag{636}$$

- **4-parameter model**: Translation, zoom and rotation in image plane

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 \cdot x + a_2 \cdot y \\ b_0 - a_2 \cdot x + a_1 \cdot y \end{bmatrix} \tag{637}$$

- **Affine model**: Translation and rotation of a plane in 3-d space

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 \cdot x + a_2 \cdot y \\ b_0 + b_1 \cdot x + b_2 \cdot y \end{bmatrix} \tag{638}$$

- **Parabolic model**: Includes approximation of perspective distortions

$$\begin{bmatrix} d_x \\ d_y \end{bmatrix} = \begin{bmatrix} a_0 + a_1 \cdot x + a_2 \cdot y + a_3 \cdot x^2 + a_4 \cdot y^2 + a_5 \cdot xy \\ b_0 + b_1 \cdot x + b_2 \cdot y + b_3 \cdot x^2 + b_4 \cdot y^2 + b_5 \cdot xy \end{bmatrix} \tag{639}$$

- Orthographic models are linear with respect to parameters (easier to estimate)
- Can be interpreted as Taylor expansions of perspective motion model

Illustration of Impact of Affine Parameters

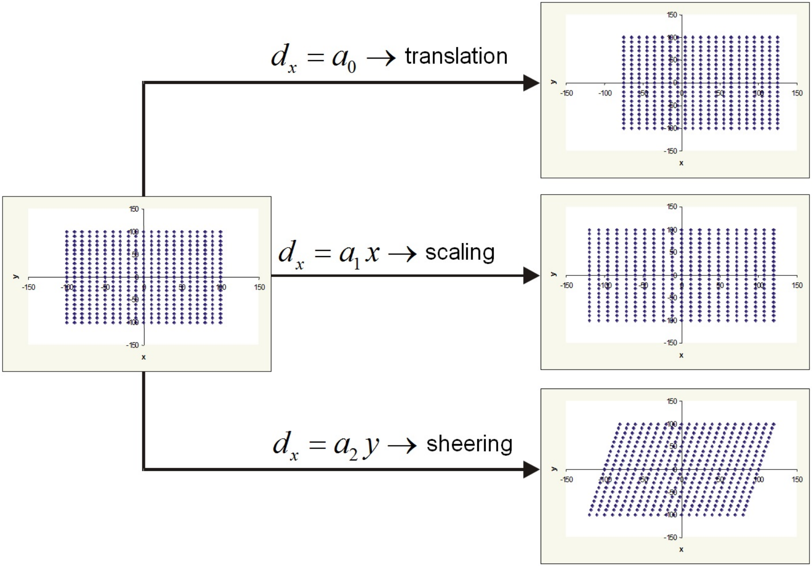


Illustration of Impact of Parabolic Parameters

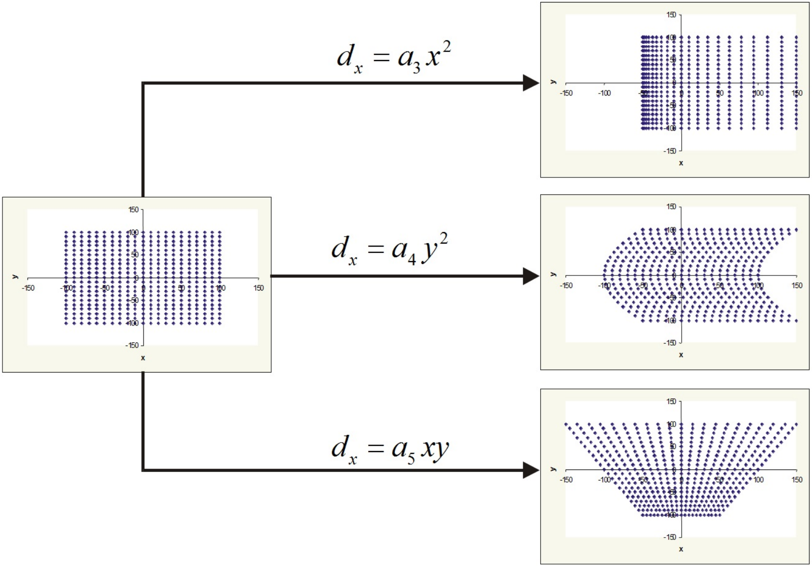
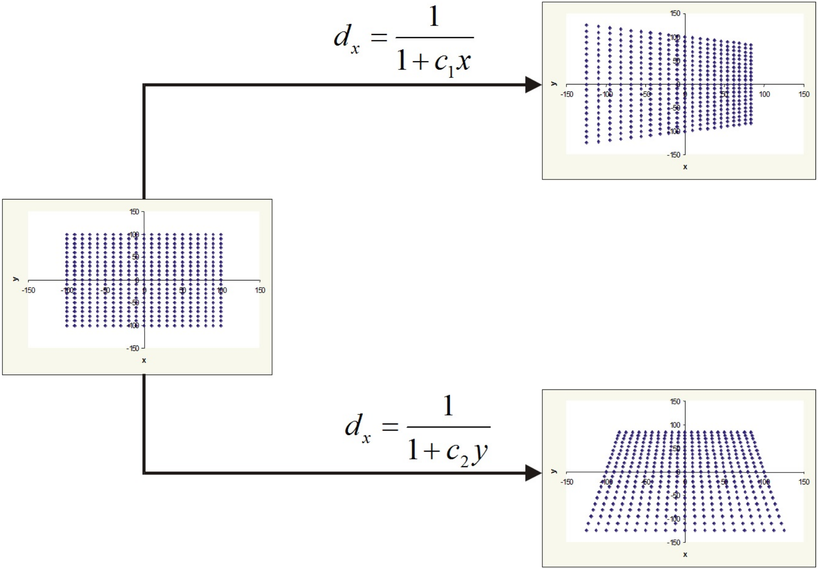


Illustration of Impact of Perspective Parameters



$$d_x = \frac{1}{1 + c_1 x}$$

$$d_x = \frac{1}{1 + c_2 y}$$

Usage of Higher Order Motion Models for Video Coding

Video coding standards of ITU-T and ISO/IEC

- All standards use simple translational motion model
- Exception: MPEG-4 Visual supports also affine and perspective model for global motion compensation (single motion model for background)
    - Difficult to estimate suitable motion parameters for background
    - Rarely used in practice

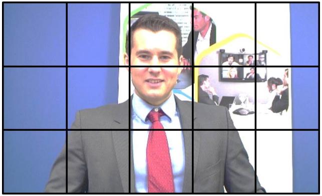
Scientific papers

- Higher-order motion models for background motion (or large regions)
- Higher-order motion models for block motion
- Example [Lakshman, et al, 2010]
    - Switching between translational and affine motion model on block basis
    - On average, bit rate savings of 1-2% compared to translational mode
    - Maximum bit rate savings of about 4%

Picture Partitioning for Motion-Compensated Prediction

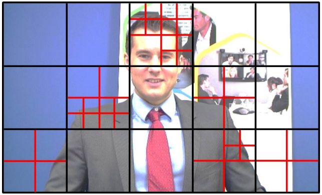
Partitioning into fixed-size square blocks

- Partitioning does not need to be transmitted
- Low flexibility
- H.262 | MPEG-2 Video:
  - One motion vector per 16×16 macroblocks



Partitioning into variable-size square blocks

- Partitioning has to be transmitted
- Simple approach: Quadtree partitioning
- Increased flexibility
- H.263 and MPEG-4 Visual:
  - 16×16 or 8×8 blocks for MCP
- H.264 | MPEG-4 AVC:
  - 16×16 to 4×4 blocks + non-square blocks
- H.265 | MPEG-H HEVC:
  - 64×64 to 8×8 blocks + non-square blocks

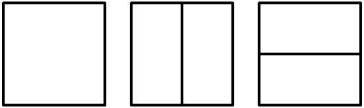


Non-Square Blocks for Motion-Compensated Prediction

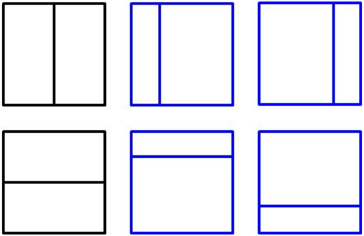
Partitioning into variable-size rectangular blocks

- Typically combined with quadtree-based partitioning into square blocks
- Square blocks can also be partitioned into 2 rectangular blocks
- Flexibility is further increased (side information rate is also increased)
- H.264 | MPEG-4 AVC:
  - Symmetric vertical and horizontal subdivision (for 16×16 and 8×8 blocks)
- H.265 | MPEG-H HEVC:
  - Symmetric and asymmetric subdivisions (for 64×64 to 8×8/16×16 blocks)

Partitioning into non-square blocks for H.264/AVC (for 16×16 and 8×8)



Partitioning into non-square blocks for H.265/HEVC (for 64×64 to 8×8/16×16)



Impact of Block Sizes for Motion-Compensated Prediction

Framework for analysis

- HEVC codec with partially disabling certain block sizes
- IPPP coding structure with quarter-sample accurate motion vectors
- Previous frame is used as reference frame

Side effect of restricting block sizes for MCP

- Impact on applicable transform sizes
- Transforms are not applied across coding unit boundaries in HEVC

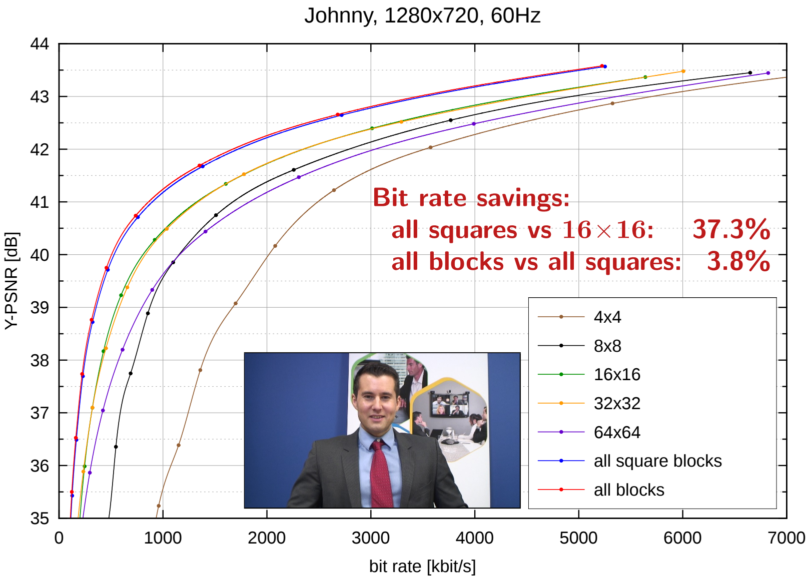
Experiment 1: Exclude effect of different transform sizes

- The same 4×4 transform coding is applied for all coding units, independently of the block sizes used for motion-compensated prediction

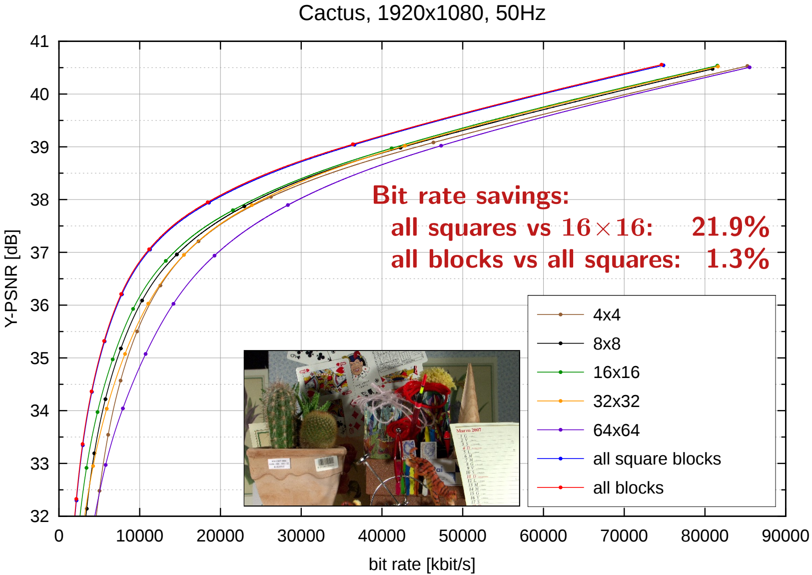
Experiment 2: Combined effect of block sizes for prediction and transform coding

- No restriction of transform sizes beyond that given by HEVC syntax

Block Sizes for MCP (4×4 Transform) – Sequence "Johnny"



Block Sizes for MCP (4×4 Transform) – Sequence “Cactus”

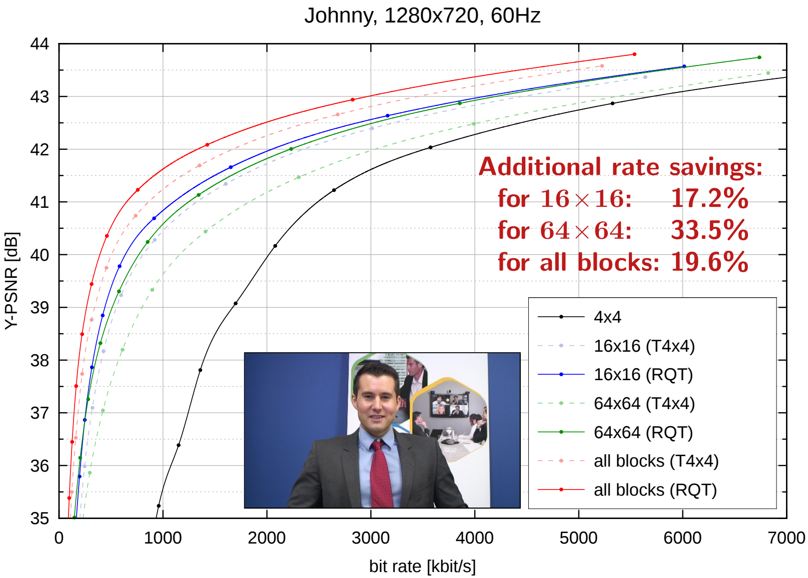


Bit rate savings:

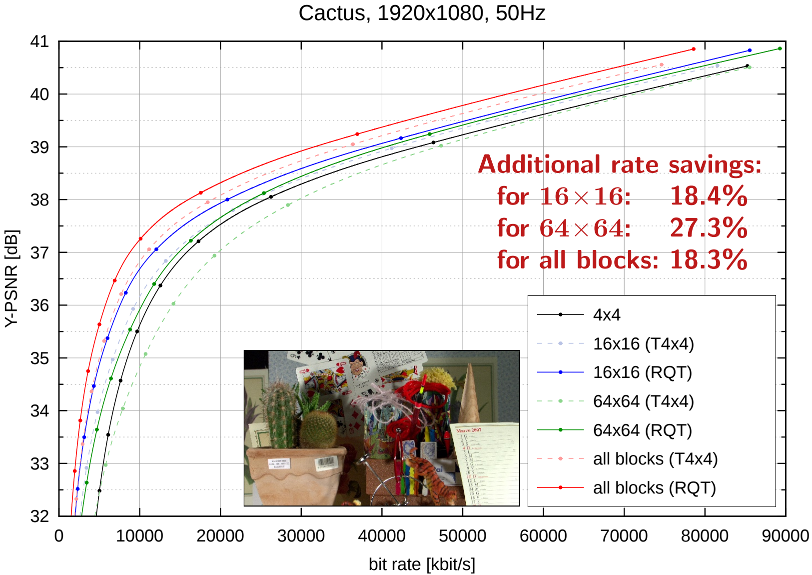
all squares vs 16×16: 21.9%

all blocks vs all squares: 1.3%

Without Restricting Transform Coding – Sequence “Johnny”



Without Restricting Transform Coding – Sequence “Cactus”



Additional rate savings:

for 16×16: 18.4%

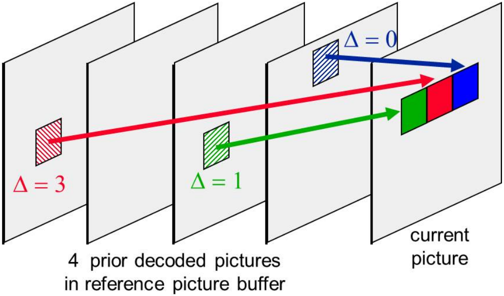
for 64×64: 27.3%

for all blocks: 18.3%

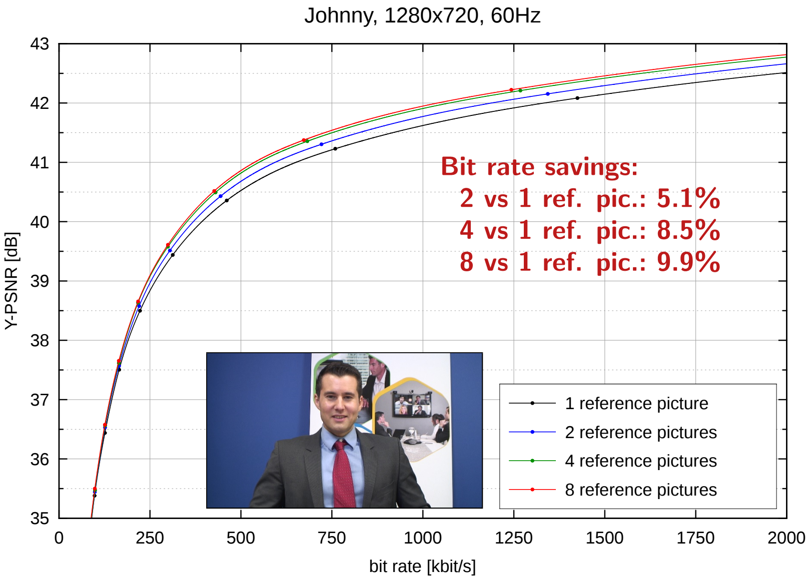
Selection of Reference Picture for MCP

Multiple reference pictures

- Motion-compensated prediction is not restricted to use previous decoded picture
- Multiple decoded pictures can be hold in a reference picture buffer
- Employed reference picture is indicated by coding an index Δ
- Side information rate is increased, but prediction may be improved
- Can exploit effects such as scene cuts, aliasing and uncovered background
- Concept is supported in H.263++, H.264/AVC and H.265/HEVC



Number of Reference Pictures – Sequence “Johnny”



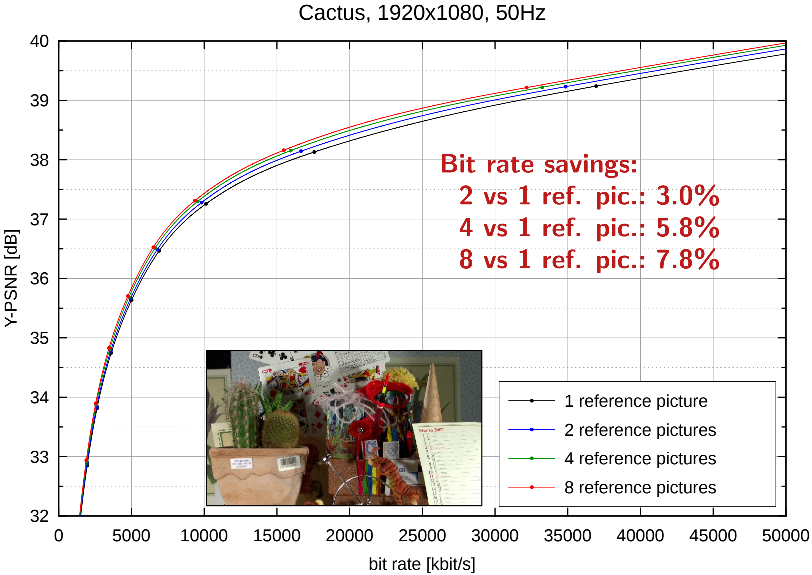
Bit rate savings:

2 vs 1 ref. pic.: 5.1%

4 vs 1 ref. pic.: 8.5%

8 vs 1 ref. pic.: 9.9%

Number of Reference Pictures – Sequence “Cactus”

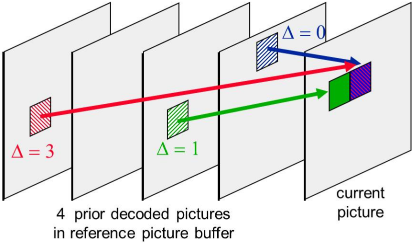


Bit rate savings:
2 vs 1 ref. pic.: 3.0%
4 vs 1 ref. pic.: 5.8%
8 vs 1 ref. pic.: 7.8%

Number of Motion Hypotheses

Multi-hypotheses motion-compensated prediction

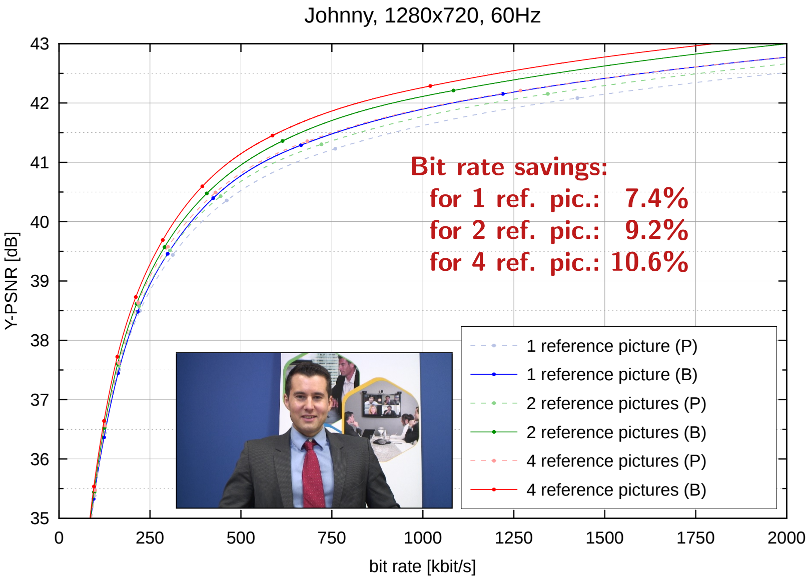
- Average (or weighted average) of multiple motion-compensated prediction signals
- Typically combined with multiple reference pictures
- Side information rate is increased, but prediction may be improved



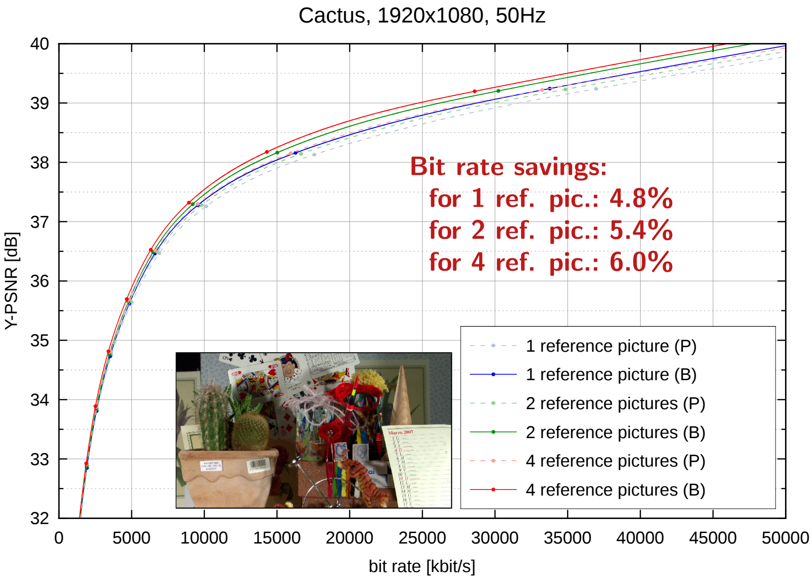
Multi-hypotheses prediction in video coding standards

- Restricted to two motion hypotheses
- Block-based switching between 1 and 2 hypotheses
- MPEG-2, MPEG-4: Restricted two "bi-directional" prediction
- H.263++, H.264/AVC, H.265/HEVC: Generalized B slices

Two Motion Hypotheses (Bi-Prediction) – Sequence “Johnny”



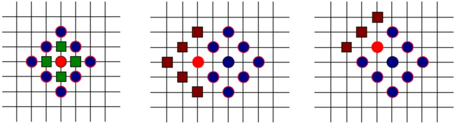
Two Motion Hypotheses (Bi-Prediction) – Sequence "Cactus"



Summary on Motion-Comp. Prediction & Hybrid Video Coding

- Motion-compensated prediction: Exploiting similarities between pictures
- Hybrid video coding
  - Motion-compensated prediction with transform coding of prediction error
  - Concept used in all ITU-T and ISO/IEC video coding standards
- Theoretical analysis of MCP using simple models
  - Motion-compensated prediction improves coding efficiency
  - Sub-sample accurate motion vectors improve coding efficiency
- Design aspects for motion-compensated prediction
  - Accuracy of displacement vector: Integer-, half, quarter-sample accuracy
  - Motion models: Translational, affine, perspective
  - Regions with constant motion: Fixed and variable block sizes
  - Selection of reference picture: Multiple reference pictures
  - Number of motion hypotheses: Bi-prediction
- Motion-compensated prediction in newest standard H.265/HEVC
  - Variable block sizes from 64×64 to $8\times4/4\times8$
  - Translational motion with quarter-sample accurate vectors
  - Multiple reference pictures and up to 2 motion hypotheses

Encoder Control



$$\min \quad D + \lambda \cdot R$$

Review of Lagrangian Encoder Control

Optimal bitstream for given set of constraints (bit rate, delay, etc.)

- With $\mathcal{B}_c$ being the set of *conforming* bitstreams $\boldsymbol{b}$ that fulfill all given constraints, the *optimal* bitstream is given by

$$\boldsymbol{b}^* = \arg\min_{\boldsymbol{b}\in\mathcal{B}_c} D\big(\boldsymbol{s}, \boldsymbol{s'}(\boldsymbol{b})\big) \tag{640}$$

  where $\boldsymbol{s}$ and $\boldsymbol{s'}$ are the original and reconstructed video, respectively
- The optimization is not feasible due to huge parameter space
  $\Longrightarrow$ Split into smaller optimization problems by partially ignoring dependencies

Lagrangian encoder control

- Consider coding of set of samples $\boldsymbol{s}_k$ (e.g. picture, macroblock) and optimized with respect to coding parameters $\boldsymbol{p}_k$ (e.g. coding modes, motion vectors)

$$\min_{\boldsymbol{p}_k} D\big(\boldsymbol{s}_k, \boldsymbol{s'}_k(\boldsymbol{p}_k)\big) \qquad \text{subject to} \qquad R(\boldsymbol{p}_k) \le R_c \tag{641}$$

- Reformulate constraint optimization problem as unconstrained problem

$$\min_{\boldsymbol{p}_k} D\big(\boldsymbol{s}_k, \boldsymbol{s'}_k(\boldsymbol{p}_k)\big) + \lambda \cdot R(\boldsymbol{p}_k) \tag{642}$$

Determination of Coding Parameters for Subsets

Determination of coding parameters for smaller units

- Consider partition of $\boldsymbol{s}_k$ into smaller subsets $\boldsymbol{s}_{k,i}$ (e.g. smaller blocks)
- For **independent coding decisions** and **additive distortion measures**, we have

$$\sum_i \left(\min_{\boldsymbol{p}_{k,i}} D\big(\boldsymbol{s}_{k,i}, \boldsymbol{s}'_{k,i}(\boldsymbol{p}_{k,i})\big) + \lambda \cdot R(\boldsymbol{p}_{k,i}) \right) \quad (643)$$

$\Longrightarrow$ **Independent selection of coding parameters** $\boldsymbol{p}_{k,i}$ for each subset

Coding decisions in image and video coding

- Coding decisions are typically not independent (e.g. due to prediction)
- For practical applicability: Partially ignore dependencies
  - $\Longrightarrow$ Consider past decisions (correct predictors for samples and coding parameters)
  - $\Longrightarrow$ Ignore impact on future decisions
- Typically used distortion measures

$$D\big(\boldsymbol{s}, \boldsymbol{s}'\big) = \sum_i \left| s_i - s'_i \right|^\beta \quad (644)$$

with $\beta = 1$: Sum of absolute differences (SAD)
$\beta = 2$: Sum of squared differences (SSD)

Application of Lagrange Optimization in Video Coding

Quantization of the transform coefficients of a block

- Select vector $\boldsymbol{q}$ of transform coefficient levels according to

$$\boldsymbol{q}^* = \arg\min_{\boldsymbol{q}} \; D(\boldsymbol{q}) + \lambda \cdot R(\boldsymbol{q}) \tag{645}$$

with $D(\boldsymbol{q})$: SSD distortion for choosing transform coefficient level vector $\boldsymbol{q}$
$R(\boldsymbol{q})$: Number of bits required for representing $\boldsymbol{q}$

$\Longrightarrow$ Rate-distortion optimized quantization (as discussed for run-level coding)
$\Longrightarrow$ Discussed algorithm considers dependencies inside a block
$\Longrightarrow$ Can be adapted to other coding schemes for transform coefficient levels

Mode decision (e.g. macroblock mode, intra prediction mode, block partitioning)

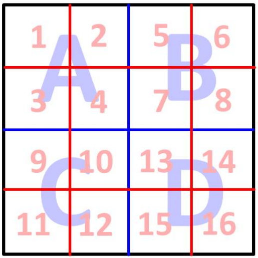
- Select coding mode c for a block

$$c^* = \arg\min_{c} \; D(c) + \lambda \cdot R(c) \tag{646}$$

with $D(c)$: SSD distortion for choosing coding mode c for the block
$R(c)$: Number of bits for block when coded with mode c

$\Longrightarrow$ Considers quantization for both distortion D and rate R
$\Longrightarrow$ If applicable, coding parameters for sub-blocks have to determined in advanced (e.g., for tree-based partitioning)

Mode Decision for Hierarchical Block Structures



Exhaustive evaluation of all partitionings

- Evaluate blocks in *depth-first order*
- Example: Two quadtree levels for a 16×16 block
  (1) Select best partitioning for first 8×8 block A
  (2) Select best partitioning for second 8×8 block B
  (3) Select best partitioning for third 8×8 block C
  (4) Select best partitioning for fourth 8×8 block D
  (5) Choose between 16×16 block and sub-division

  Note: Predictors are set according to prior decisions

Fast mode decision strategies for Hierarchical Structures

- Terminate decision process as soon as a "quality criterion" is met
  - Distortion (or Lagrangian cost) less than a threshold
  - Number of significant transform coefficients less than a threshold
- Example: Top-down approach
  (1) Evaluate 16×16 block and stop if quality criterion is met
  (2) Evaluate first 8×8 block and check quality criterion
  $\Longrightarrow$ Check 4×4 partitioning only if quality criterion is not met
  (3) Proceed with next 8×8 block, etc.

Lagrange Optimization for Choosing Motion Vectors

Straightforward application of Lagrange optimization

- Treat each motion vector $\boldsymbol{m} = [m_x, m_y]^{\mathrm{T}}$ out of a considered set $\mathcal{M}$ of motion vectors as a coding mode and apply mode decision process

$$\boldsymbol{m}^* = \arg\min_{\boldsymbol{m}\in\mathcal{M}} \; D(\boldsymbol{m}) + \lambda_m \cdot R(\boldsymbol{m}) \tag{647}$$

with $D(\boldsymbol{m})$: SSD distortion for choosing motion vector $\boldsymbol{m}$ for the block
$R(\boldsymbol{m})$: Number of bits for block when coded using motion vector $\boldsymbol{m}$

$\Longrightarrow$ Considering quantization for each possible motion vector is way too complex

Practical rate-distortion optimization for motion vector selection

- Assume that transmitted residual is equal to zero (important case in practice)
- Use the Lagrange minimization with less complex cost measure
  - $R(\boldsymbol{m})$ is the rate for coding only the motion vector $\boldsymbol{m}$
  - $D(\boldsymbol{m})$ is the distortion between original and prediction signal $D(\boldsymbol{s}, \hat{\boldsymbol{s}})$
- As distortion measure, the SAD distortion is typically used in practice

$\Longrightarrow$ The Lagrange parameter is different than for mode decision and quantization

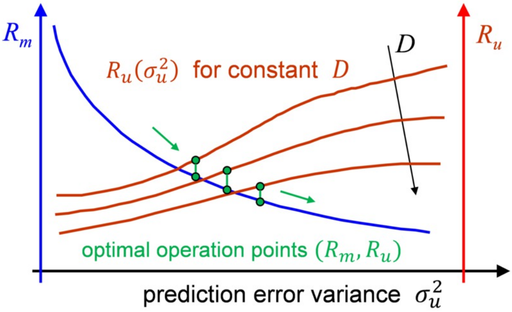
$\Longrightarrow$ The choice $\lambda_m = \sqrt{\lambda}$ is typically used in this case

Importance of Rate-Constrained Decisions for Modes & Motion

Distortion D of reconstructed signal is influence by two factors

- Side information rate R_m: Increasing R_m improves prediction and reduces distortion
- Rate for residual signal R_u: Increasing R_u reduces distortion
- Optimal rate allocation: Minimization of $J = D(R_m, R_u) + \lambda \cdot (R_m + R_u)$

$$\left.\begin{array}{lll} \frac{\partial}{\partial R_u} J = 0 & \Longrightarrow & \frac{\partial}{\partial R_u} D = -\lambda \\ \frac{\partial}{\partial R_m} J = 0 & \Longrightarrow & \frac{\partial}{\partial R_m} D = -\lambda \end{array}\right\} \Longrightarrow \boxed{\frac{\partial}{\partial R_m} D = \frac{\partial}{\partial R_u} D} \qquad (648)$$



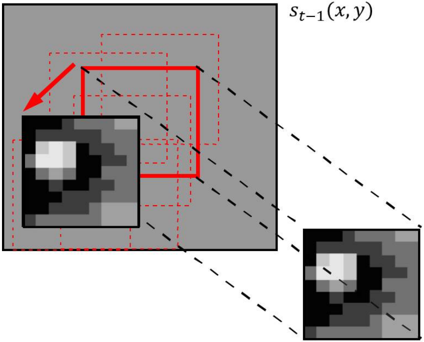
Prediction error variance σ_u^2 can be influenced by

- Number of intra pred. modes
- Block sizes for intra prediction
- Block sizes for motion comp.
- Accuracy of motion vectors
- Number of reference pictures
- Number of motion hypotheses
- Choice of motion model

Estimation of Translational Motion: Block Matching

Principle of block matching

- Subdivide current frame into blocks
- Determine one displacement vector for each block
- Find best match in reference frame by minimizing Lagrange cost $D + \lambda \cdot R$



Distortion measures for block matching

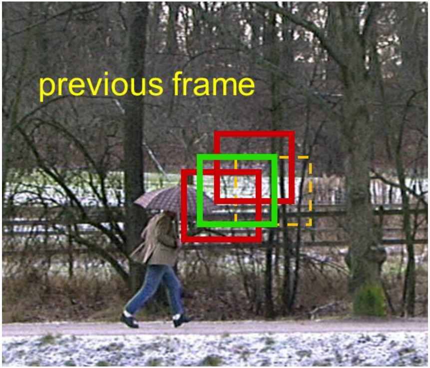
- Typically: SAD distortion
- Alternative measures:
  - SSD distortion
  - SAD in transform domain
  - cross correlation

Difficulty in determination of displacement parameters by block matching

- It is not feasible to evaluate all "possible motion vectors" (there are too many)

$\Longrightarrow$ Require intelligent search strategies (testing only most likely motion vectors)

Illustration of the Block Matching Algorithm

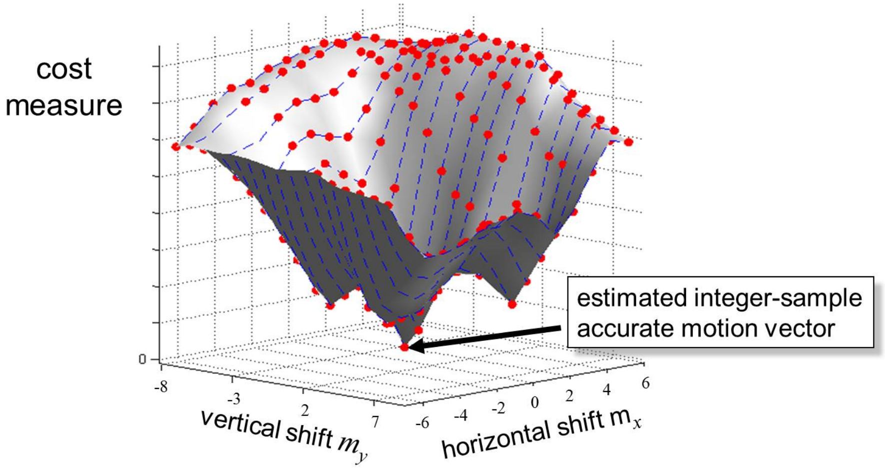


The measurement window is compared with **different shifted blocks** in the reference frame and the **best match** is determined

The considered block of samples in the current frame is selected as a measurement window

Cost Measures Values inside a Search Window

Example: Cost measure values inside a search window



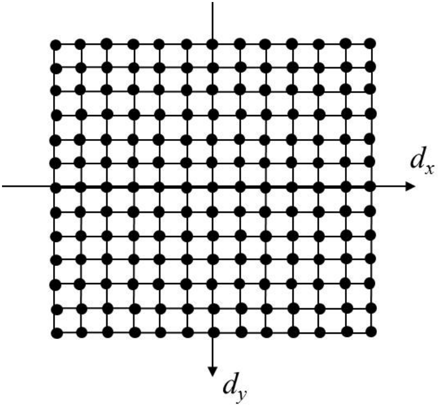
Search Strategies: Exhaustive Search

Exhaustive search

- Evaluate all possible motion vectors (displacements) inside a rectangular search window
- Computationally very complex
- Highly regular, parallelizable

Selection of search window

- Often centered around zero motion vector
- Can also be centered around motion vector predictor
- Size can be adapted during encoding of a picture
- Size can be increased under certain circumstances



Search Strategies: Methods for Complexity Reduction

Complexity of block matching

- Evaluation of **complex error measure** for **many candidates**

Two approaches for reducing encoder complexity

Complexity of error measure

- Fast approximations
- Early termination
- Exclusion of candidates

Number of search candidates

- Skip unlikely areas in search
- Adaptively increase or decrease distance between search candidates

Combine both approaches

- Choose starting point and search order that maximizes likelihood for efficient approximations, early terminations and excluding candidates

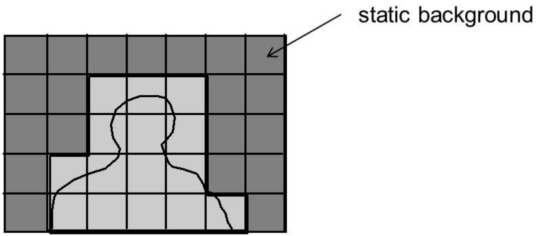
Search Strategies: Fast Approximations

Basic approach: Stop search if match is "good enough"

- Distortion measure D is less than a threshold
- Lagrange cost $D + \lambda \cdot R$ is less than a threshold

Practical method in video conferencing (static background)

- Evaluate zero vector and stop search if the match is good enough



Search Strategies: Early Termination

Compare partial cost measures

- Partial distortion measure D_K for block size $B_x \times B_y$, with $K = 1 \ldots B_y$

$$D_K(m_x, m_y) = \sum_{y=0}^{K-1} \sum_{x=0}^{B_x-1} \left| s_t(x,y) - s'_{t-1}(x - m_x, y - m_y) \right|^{\beta} \qquad (649)$$

- Compare partial cost measure with previously determined minimum cost $J_{\min}$
- Early termination without loss

$$\text{Stop if:} \qquad D_K(m_x, m_y) + \lambda_m \cdot R(m_x, m_y) \geq J_{\min} \qquad (650)$$

- Early termination with possible loss (but higher speedup)

$$\text{Stop if:} \qquad D_K(m_x, m_y) + \lambda_m \cdot R(m_x, m_y) \geq \alpha(K) \cdot J_{\min} \qquad (651)$$

Example for weighting function: $a(K) = \sqrt{\dfrac{K}{B_y}}$

Search Strategies: Early Exclusion of Candidates

Speed-up for block comparison

- Triangle inequality for samples in a block $\mathcal{B}$ (here, for SAD)

$$\sum_{k \in \mathcal{B}} \left| s_k - \hat{s}_k \right| \geq \left| \sum_{k \in \mathcal{B}} (s_k - \hat{s}_k) \right| = \left| \left(\sum_{k \in \mathcal{B}} s_k \right) - \left(\sum_{k \in \mathcal{B}} \hat{s}_k \right) \right| \qquad (652)$$

- Basic strategy
  1. Compute sum of samples values for all block locations in reference frame (sliding window average can be calculated in a very easy way)
  2. Compute sum of samples values for current block
  3. Omit complete distortion calculation if difference between sums of samples values yields larger cost measure than previous minimum

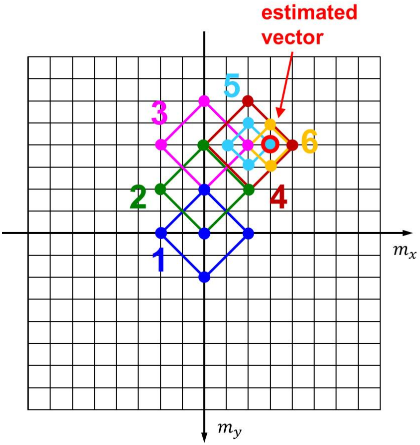
- Combination with variable size prediction blocks (H.264/AVC, H.265/HEVC)
  - Start with computation of sample sums for the smallest supported block size
  - Sums for larger blocks are obtained by adding up the sums for smaller blocks

$\Longrightarrow$ Increases speed-up for nested block sizes as found in modern video codecs

Search Strategies: 2-D Logarithmic Search

2-d logarithmic search [Jain, Jain, 1981]

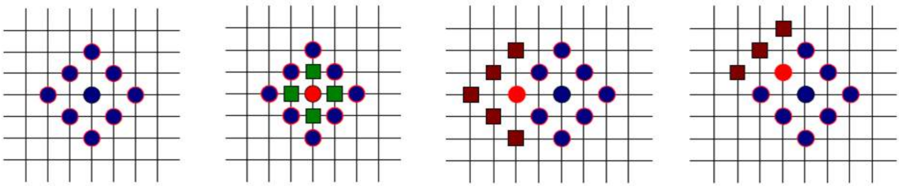
- Iterative comparison of the cost measures at 5 points (corners and center) of a diamond-shaped pattern
- Move pattern so that pattern in centered around best match
  - No more than 3 new candidates
- Logarithmic refinement of search pattern (4 new candidates) if
  - Best match is in center of pattern
  - Or best match is at the border of the search range
- Motion search is terminated if
  - Best match is in center of pattern
  - And smallest pattern size is used



Search Strategies: Diamond Search

Diamond search [Li, Zeng, Liou, 1994] and [Zhu, Ma, 1997]

- Iterative search with 9 points of a diamond pattern
- Similar search strategy as 2-d logarithmic search



Start with large diamond pattern at motion vector (0,0) or at a predicted vector

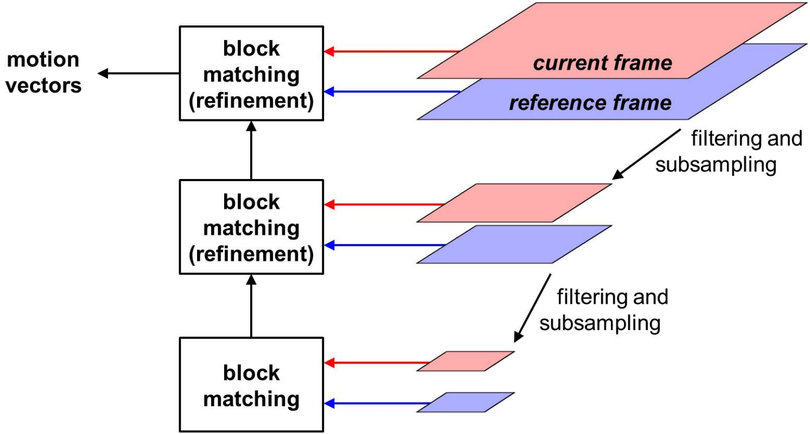
If best match is in the center of a large diamond, proceed with a smaller diamond

If best match does not lie in the center of the diamond pattern, center next diamond pattern at the best match

Search Strategies: Hierarchical Block Matching

Hierarchical block matching

- Start with dyadically downsampled pictures
- Refine motion vectors from one hierarchy level to the next



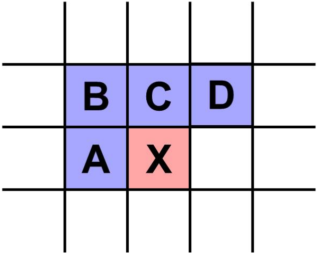
Search Strategies: Choosing of Start Point

Non-adaptive choices of start point

- Use motion vector (0,0) as start point of motion search
  - $\Longrightarrow$ Suitable for applications like video conferencing
  - $\Longrightarrow$ Problematic if large motions occur in video sequence
- Use motion vector predictor as start point for motion search
  - $\Longrightarrow$ Typically results in faster termination of motion search

Adaptive choice of start point

- General idea: Motion of a block is similar to at least one of the neighboring blocks
- First evaluate the motion vectors of the already estimated neighboring blocks
  - Example: Blocks A, B, C and D
  - Candidates can also include a temporally predicted motion vector
- Choose best match among the candidates as start point of the motion search



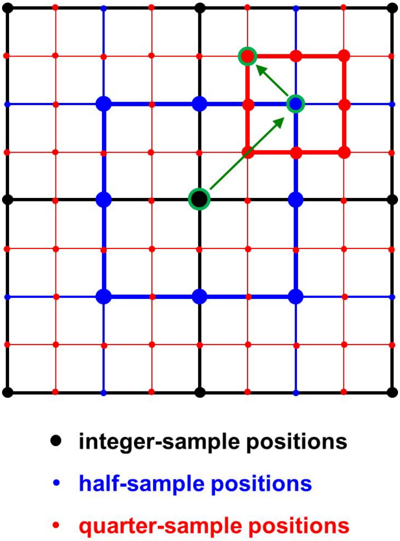
Estimation of Sub-Sample Accurate Motion Vectors

Sub-sample accurate motion vectors

- Motion vectors are often not restricted to integer-sample accurate displacements
- Typical sub-sample accuracies: Half- and quarter-sample

Estimation of sub-sample shifts

- Typical: Iterative sub-sample refinement using best integer-sample displacement
  - Test 8 half-sample candidates around best integer-sample match
  - Test 8 quarter-sample candidates around best half-sample match
- Requires interpolation of sample values at sub-sample locations



Estimation of Higher-Order Motion Parameters

Linear approximation of prediction error signal

- Interpolated reconstructed reference picture is denoted by $s'_{\mathrm{ref}}(x,y)$
- Assume: Estimate $\hat{\boldsymbol{d}} = [\hat{d}_x, \hat{d}_y]^{\mathrm{T}}$ of displacement vector $\boldsymbol{d} = [d_x, d_y]^{\mathrm{T}}$ is given
- Assume: Displacement errors $\Delta d_x = d_x - \hat{d}_x$ and $\Delta d_y = d_y - \hat{d}_y$ are small
- Prediction error for a sample location (x,y) can be approximated by

$$\begin{aligned} u[x,y] &= s[x,y] - s'_{\mathrm{ref}}(x - d_x, y - d_y) \\ &= s[x,y] - s'_{\mathrm{ref}}(x - \hat{d}_x - \Delta d_x, y - \hat{d}_y - \Delta d_y) \\ &= s[x,y] - s'_{\mathrm{ref}}(\hat{x} - \Delta d_x, \hat{x} - \Delta d_y) \\ &\approx s[x,y] - s'_{\mathrm{ref}}(\hat{x},\hat{y}) + \frac{\partial s'_{\mathrm{ref}}}{\partial x}(\hat{x},\hat{y}) \cdot \Delta d_x + \frac{\partial s'_{\mathrm{ref}}}{\partial y}(\hat{x},\hat{y}) \cdot \Delta d_y \end{aligned} \qquad (653)$$

with

$(\hat{x},\hat{y})$: Predicted reference sample location $(x - \hat{d}_x, y - \hat{d}_y)$

$\frac{\partial s'_{\mathrm{ref}}}{\partial x}$: Gradient in x direction of interpolated reference picture $s'_{\mathrm{ref}}(x,y)$

$\frac{\partial s'_{\mathrm{ref}}}{\partial y}$: Gradient in y direction of interpolated reference picture $s'_{\mathrm{ref}}(x,y)$

Linear Parametric Motion Models

Consider motion models that are linear with respect to the motion parameters

- Displacement vector field can be expressed using a matrix multiplication

$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \boldsymbol{B}(x,y) \cdot \boldsymbol{a} \tag{654}$$

- Example: Affine motion model

$$\begin{bmatrix} d_x(x,y) \\ d_y(x,y) \end{bmatrix} = \begin{bmatrix} 1 & x & y & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & x & y \end{bmatrix} \cdot \begin{bmatrix} a_0 \\ a_1 \\ a_2 \\ a_3 \\ a_4 \\ a_5 \end{bmatrix} \tag{655}$$

- Prediction error for a location $\boldsymbol{x} = [x, y]^{\mathrm{T}}$ can be written as

$$u[\boldsymbol{x}] = s[\boldsymbol{x}] - s'_{\text{ref}}(\hat{\boldsymbol{x}}) + \frac{\partial s'_{\text{ref}}}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}) \cdot \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{\Delta a} \tag{656}$$

with

$$\frac{\partial s'_{\text{ref}}}{\partial \boldsymbol{x}} = \begin{bmatrix} \frac{\partial s'_{\text{ref}}}{\partial x} & \frac{\partial s'_{\text{ref}}}{\partial y} \end{bmatrix} \quad \text{and} \quad \hat{\boldsymbol{x}} = \boldsymbol{x} - \boldsymbol{B}(\boldsymbol{x}) \cdot \hat{\boldsymbol{a}} \tag{657}$$

Minimizing SSD Distortion for Linear Motion Models

SSD distortion using linear approximation for small displacement errors

- SSD distortion for a region $\mathcal{R}$ with unique parametric motion

$$D(\boldsymbol{\Delta a}) = \sum_{\boldsymbol{x} \in \mathcal{R}} \left(s[\boldsymbol{x}] - s'_{\text{ref}}(\hat{\boldsymbol{x}}) + \frac{\partial s'_{\text{ref}}}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}) \cdot \boldsymbol{B}(\boldsymbol{x}) \cdot \boldsymbol{\Delta a} \right)^2 \tag{658}$$

- Minimization with respect to parameter update $\boldsymbol{\Delta a}$ yields a linear equation system

$$\frac{\partial D(\boldsymbol{\Delta a})}{\partial \boldsymbol{\Delta a}} = \boldsymbol{0} \qquad \Longrightarrow \qquad \boxed{\boldsymbol{G}(\hat{\boldsymbol{a}}) \cdot \boldsymbol{\Delta a} = \boldsymbol{g}(\hat{\boldsymbol{a}})} \tag{659}$$

with the matrix $\boldsymbol{G}(\hat{\boldsymbol{a}})$ and the vector $\boldsymbol{g}(\hat{\boldsymbol{a}})$ being given by

$$\boldsymbol{G}(\hat{\boldsymbol{a}}) = \sum_{\boldsymbol{x} \in \mathcal{R}} \boldsymbol{B}(\boldsymbol{x})^{\mathrm{T}} \left(\frac{\partial s'_{\text{ref}}}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}) \right)^{\mathrm{T}} \left(\frac{\partial s'_{\text{ref}}}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}) \right) \boldsymbol{B}(\boldsymbol{x}) \tag{660}$$

$$\boldsymbol{g}(\hat{\boldsymbol{a}}) = -\sum_{\boldsymbol{x} \in \mathcal{R}} \boldsymbol{B}(\boldsymbol{x})^{\mathrm{T}} \left(\frac{\partial s'_{\text{ref}}}{\partial \boldsymbol{x}}(\hat{\boldsymbol{x}}) \right)^{\mathrm{T}} \left(s[\boldsymbol{x}] - s'_{\text{ref}}(\hat{\boldsymbol{x}}) \right) \tag{661}$$

- Can be solved by conventional methods, e.g. Gauss algorithm

Iterative Estimation of Motion Parameters

Iterative algorithm for parameter estimation of linear models and SSD distortion

1. Initialize parameter estimate $\hat{\boldsymbol{a}}$, e.g. with zero vector
2. Determine matrix $\boldsymbol{G}(\hat{\boldsymbol{a}})$ and vector $\boldsymbol{g}(\hat{\boldsymbol{a}})$
3. Determine parameter update $\boldsymbol{\Delta a}$ by solving the linear equation system

$$\boldsymbol{G}(\hat{\boldsymbol{a}}) \cdot \boldsymbol{\Delta a} = \boldsymbol{g}(\hat{\boldsymbol{a}})$$

4. Update parameter estimate

$$\hat{\boldsymbol{a}} = \hat{\boldsymbol{a}} + \boldsymbol{\Delta s}$$

5. Repeat the last three steps until algorithm converges

Difficulties

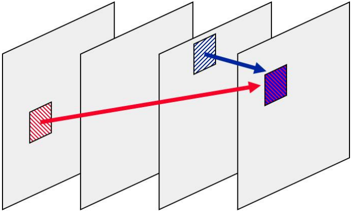
- Approximation only valid for small parameter errors $\boldsymbol{\Delta a}$

  $\Longrightarrow$ Initialize translational part with block matching result

- Aperture problem: Estimate $\hat{\boldsymbol{a}}$ has a large relative error when the smallest eigenvalue of the gradient matrix $\boldsymbol{G}$ is small

  $\Longrightarrow$ Block has to contain large gradients in both directions for a reliable estimate

Motion Estimation for Multi-Hypotheses Prediction



Motion estimation for multiple motion vectors

- Need to estimate multiple motion vectors (typically for different reference pictures) for a block in current frame
- Independent estimation is sub-optimal
- Estimation in product space is too complex

Independent estimation of motion hypotheses is not optimal

- Example: SSD distortion for bi-prediction

$$
\begin{aligned}
D_{\text{Bi}} &= \sum_{x,y} \left(s[x,y] - \frac{1}{2}\big(\hat{s}_1[x,y] + \hat{s}_2[x,y]\big) \right)^2 \\
&= \frac{1}{4} \sum_{x,y} \Big(\big(s[x,y] - \hat{s}_1[x,y]\big) + \big(s[x,y] - \hat{s}_2[x,y]\big) \Big)^2 \\
&= \frac{1}{4} D_1 + \frac{1}{4} D_2 + \frac{1}{2} \sum_{x,y} \big(s[x,y] - \hat{s}_1[x,y]\big)\big(s[x,y] - \hat{s}_2[x,y]\big) \qquad (662)
\end{aligned}
$$

$\Longrightarrow$ Minimization of D_1 and D_2 does not minimize D_{Bi}

Iterative Motion Estimation for Multi-Hypotheses Prediction

Iterative motion estimation for multiple motion hypotheses

- Example: Bi-prediction with the following assumptions
  - Motion vector for one hypothesis is given and yields prediction signal $\hat{s}_1[x, y]$
  - Want to estimate motion vector $[m_x^{(2)}, m_y^{(2)}]^{\mathrm{T}}$ for the second hypothesis
- Distortion for bi-prediction can be written as

$$
\begin{aligned}
D_{\mathrm{Bi}} &= \sum_{x,y} \left| s[x,y] - \frac{1}{2}\Big(\hat{s}_1[x,y] + s_{\mathrm{ref}}^{(2)}\big(x - m_x^{(2)}, y - m_y^{(2)}\big)\Big) \right|^{\beta} \\
&= \frac{1}{2^{\beta}} \cdot \sum_{x,y} \left| (2 \cdot s[x,y] - \hat{s}_1[x,y]) - s_{\mathrm{ref}}^{(2)}\big(x - m_x^{(2)}, y - m_y^{(2)}\big) \right|^{\beta} \\
&= \frac{1}{2^{\beta}} \cdot \sum_{x,y} \left| s^*[x,y] - s_{\mathrm{ref}}^{(2)}\big(x - m_x^{(2)}, y - m_y^{(2)}\big) \right|^{\beta} \qquad (663)
\end{aligned}
$$

$\Longrightarrow$ Conventional motion search, but with modified original signal $s^*[x, y]$

- Iterative algorithm for bi-prediction
  1. Independent estimation of first motion hypothesis
  2. Conditional estimation of second/first motion hypothesis (alternately)
  3. Repeat last step until convergence
- Algorithm can be extended to more than two hypotheses

Summary on Encoder Control

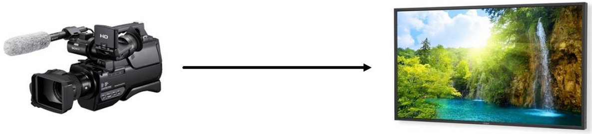
Lagrangian encoder control

- Minimization of cost function $D + \lambda \cdot R$
- In video coding: Need to partially neglect interdependencies
- Applications in video coding
    - Rate-distortion optimized quantization
    - Rate-distortion optimized mode decision
    - Rate-distortion optimized motion estimation

Motion estimation

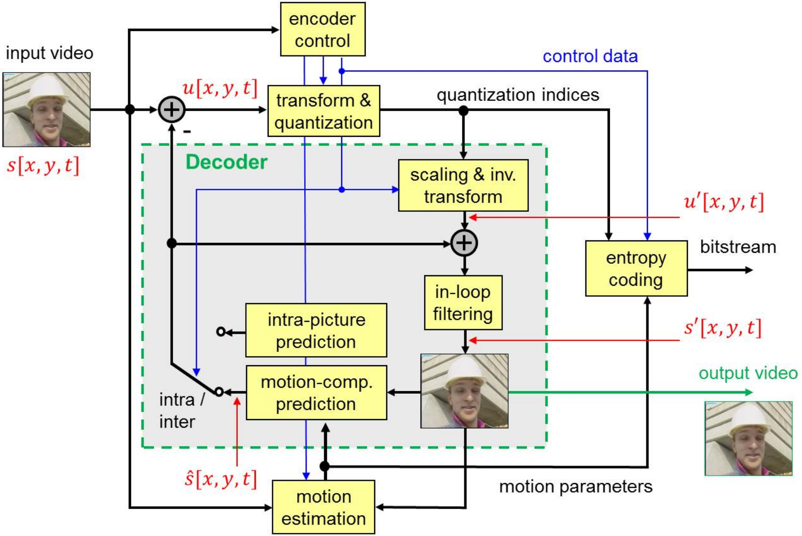
- Translational motion: Block matching
    - Early termination of distortion calculation
    - Fast search strategies
- Higher-Order motion models
    - Iterative differential motion search for linear motion models
    - Initialization with block matching result
- Motion estimation for multi-hypotheses prediction
    - Iterative motion search
    - Alternatively refinement of motion hypotheses

Video Coding Standards



Basic Coding Approach: Hybrid Video Coding

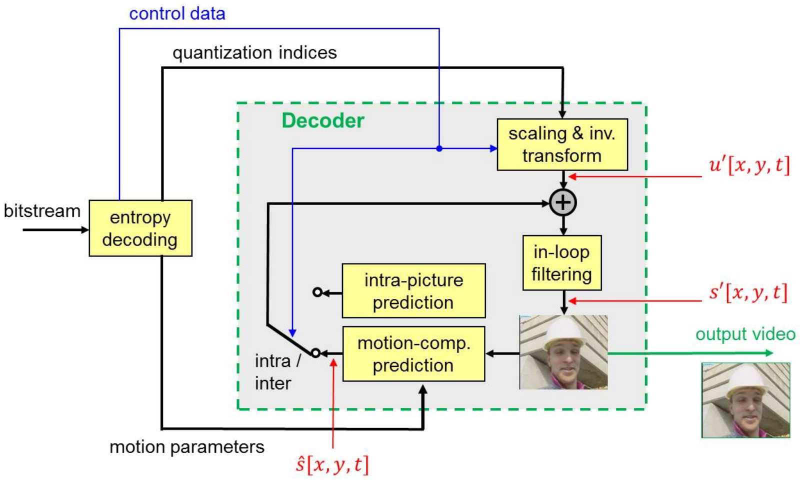
H.261, H.262/MPEG-2, H.263, MPEG-4, H.264/AVC, H.265/HEVC



Specification of Video Coding Standards

Video coding standards specify **bitstream syntax** and **decoding result**

- Enables interoperability between devices of different manufactures
- Leaves room for optimization (but does not guarantee any quality)

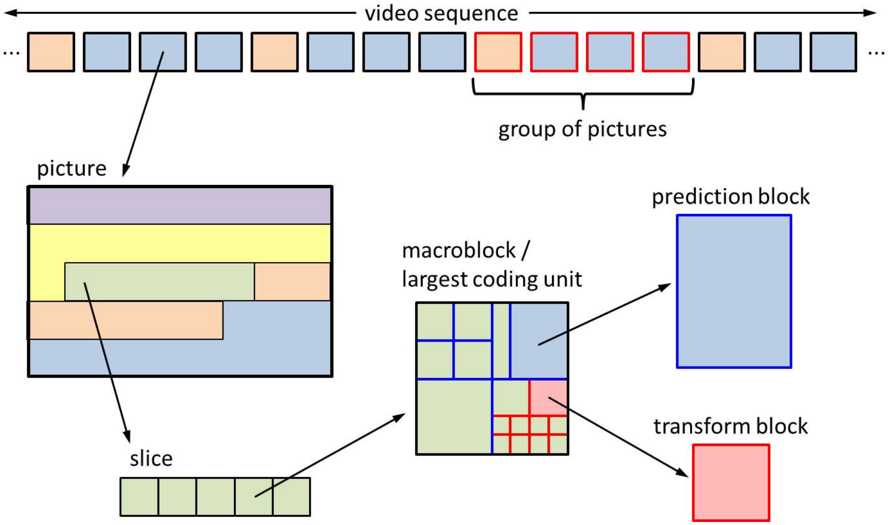


Application Areas of Video Coding Standards

digital television broadcasting	SD: 1.5 ... 6 Mbps HD: 5 ... 20 Mbps	MPEG-2, H.264/AVC
DVD-Video Blu-ray disc	5 ... 20 Mbps up to 40 Mbps	MPEG-2 MPEG-2, H.264/AVC, VC-1
Internet video streaming	100 ... 2000 kbps	H.264/AVC or proprietary codecs
video telephony video conferencing	20 ... 2000 kbps	H.263, H.264/AVC (incl. SVC extension)
video over 3G wireless networks	100 ... 500 kbps	H.263, MPEG-4, H.264/AVC

Note: H.265/HEVC is expected to be used in a significant number of application areas in near future

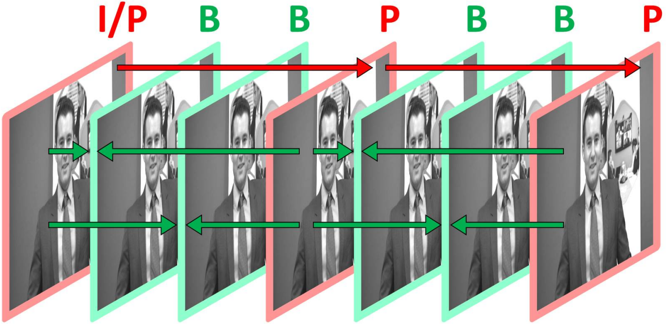
Hierarchical Bitstream Syntax



H.262 | MPEG-2 Video

(ITU-T Rec. H.262 | ISO/IEC 13818-2)

Pictures Types in H.262 | MPEG-2 Video



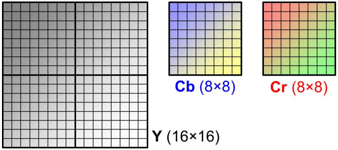
H.262 | MPEG-2 Video supports 3 pictures types:

- **I picture**: Intra-only coding (random access point)
- **P picture**: Predicted using previous I/P picture
- **B picture**: Predicted using previous I/P and next I/P picture

Picture Partitioning using Macroblocks

Partitioning of pictures into macroblocks

- Partitioning into fixed-size macroblocks
- Macroblock in 4:2:0 chroma format:
  - one 16×16 luma block
  - two 8×8 chroma blocks
- Slices: Consecutive MBs inside an MB row



For each macroblock, a coding mode can be selected

- Supported coding modes depends on picture type
- Supported modes are summarized below
  (without mentioning the modes which additionally support a quantizer change)

I picture	P picture	B picture
Intra	Intra	Intra
	P-Skip	B-Skip
	No MC, coded	Fwd, not coded
	MC, not coded	Fwd, coded
	MC, coded	Bwd, not coded
		Bwd, coded
		Interp, not coded
		Interp, coded

Macroblock Coding Modes in I and P Pictures

Intra coding mode in H.262 | MPEG-2 Video (brief review)

- Transform coding of 8×8 blocks with separable DCT and scalar quantization
- DC coefficient: Coding of difference to DC coefficient of previous block
- AC coefficients: Zig-zag scan and run-level coding with EOB symbol

Inter-picture macroblock coding modes in P pictures

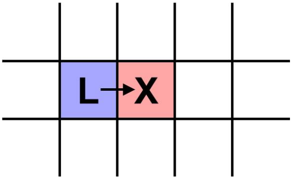
- Motion-compensated prediction using the previous I/P picture
- One motion vector for the entire macroblock
- Residual is transmitted using transform coding (similar to Intra)
  - Transform coding of 8×8 blocks using separable DCT and scalar quantization
  - Coded block pattern (VLC code indicating non-zero transform blocks)
  - Zig-zag scan and run-level coding (different table than for Intra)
- Special modes indicating that the motion vector and/or residual is zero

coding mode	motion vector	residual signal
P-Skip	inferred to be zero	inferred to be zero
No MC, coded	inferred to be zero	transmitted
MC, not coded	transmitted	inferred to be zero
MC, coded	transmitted	transmitted

Coding of Motion Vectors and Sub-Sample Interpolation

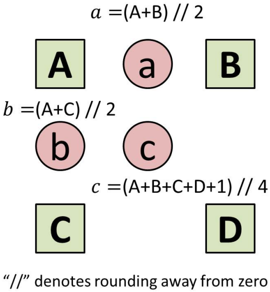
Coding of motion vectors

- Motion vector are transmitted with an accuracy of a half-luma sample
- Differential coding using the motion vector of the left macroblock as predictor
- Predictor is reset at beginning of a slice
- Motion vectors can only reference blocks inside the picture area



Sub-sample interpolation

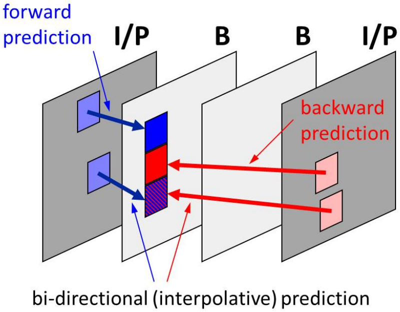
- Samples at the integer grid are directly copied from the reference frame
- Bi-linear interpolation is used for sub-sample locations
- For chroma
  - Motion vector are first rounded to half-chroma sample precision
  - Then, bi-linear interpolation is used



Motion-Compensated Prediction in B Pictures

Two reference pictures

- Previous I/P picture in display order
- Next I/P picture in display order (but preceding in coding order)



Three types of prediction

- Forward: Using previous I/P picture
- Backward: Using next I/P picture
- Bi-directional: Average of forward and backward prediction signal

Inter-picture macroblock coding modes in B pictures

- One or two motion hypotheses
- One motion vector per hypothesis
- Coding mode signals type of prediction and if residual is zero
- B-Skip: Same motion hypotheses as macroblock to the left

coding mode	prediction	resiudal
B-Skip	inferred (left)	not coded
Fwd, not coded	forward	not coded
Fwd, coded	forward	coded
Bwd, not coded	backward	not coded
Bwd, coded	backward	coded
Interp, not coded	bi-directional	not coded
Interp, coded	bi-directional	coded

ITU-Rec. H.263

Overview of Main Syntax Features in ITU-Rec. H.263

Main syntax features are similar to that of H.262 | MPEG-2 Video

- 3 pictures types: I, P and B pictures (B pictures enabled by optional Annex O)
- Macroblock coding modes: Intra & Inter (motion-compensated prediction)
- Transform coding of intra or residual signal
- 8×8 DCT and scalar quantization

Main improvements relative to H.262 | MPEG-2 Video

- 3-d run-level-last coding for transform coefficient levels
- Component-wise median prediction for motion vectors
- Annex D: Motion vectors outside picture boundaries
- Annex F: Motion-compensated prediction with 8×8 blocks
- Annex I: Prediction of intra AC coefficient and adaptive scanning
- Annex J: Deblocking filter inside motion compensation loop
- Annex U: Multiple reference pictures

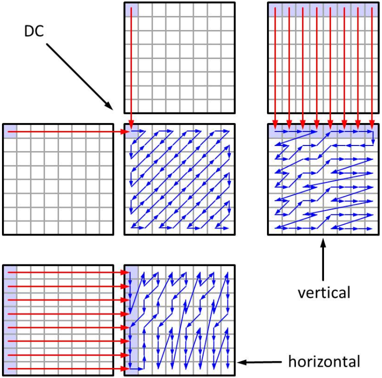
Improvements for Residual Coding and Intra Coding

Coding of the residual signal for a macroblock

- Same 8×8 DCT as in H.262 | MPEG-2 Video
- Scalar quantization (no support of quantization weighting matrices)
- **Run-level-last coding** of transform coefficient levels (instead of run-level coding)

Advanced intra coding mode (Annex I)

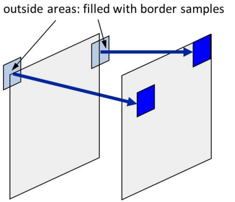
- 8×8 DCT and scalar quantization
- **Prediction of DC and AC coefficients** (signaled at macroblock level)
  - DC prediction
  - Vertical prediction
  - Horizontal prediction
- **Adaptive scanning** of transform coefficient levels
  - Determined by chosen intra prediction mode
- Run-level-last coding of transform coefficient levels



Improvements for Motion-Compensated Prediction

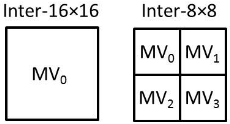
Motion vectors outside picture boundaries

- Specified in optional Annex D
- Motion vectors can reference blocks outside the picture area (not supported in MPEG-2)
- Outside areas are filled with corresponding border samples



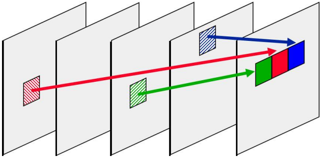
Variable block size motion compensation

- Supported in P pictures (optional Annex F)
- Inter-16×16: One motion vector per macroblock
- Inter-8×8: Four motion vectors per macroblock



Multiple reference pictures (optional Annex U)

- Transmit reference picture index in addition to motion vector
- Management of reference picture buffer
  - Sliding window operation
  - Explicit commands

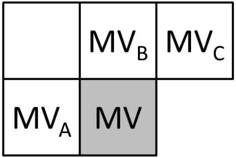


Further Improvements compared to H.262 | MPEG-2 Video

Motion vector coding

- Slices can cover multiple rows of macroblocks
- Motion vectors are predicted by the component-wise median of 3 neighboring motion vectors

$$\hat{m}_x = \text{median}\left(m_x^{(A)},\, m_x^{(B)},\, m_x^{(C)}\right)$$
$$\hat{m}_y = \text{median}\left(m_y^{(A)},\, m_y^{(B)},\, m_y^{(C)}\right)$$



- B pictures: Separate predictor for forward and backward prediction

Optional deblocking filter (Annex J)

- Deblocking filter for reducing block-edge artifacts
- Strength of smoothing filter is controlled by quantization parameter
- Improves subjective quality of current picture
- Improves "quality" of motion-compensated prediction signal of following pictures

MPEG-4 Visual

(ISO/IEC 14496-2)

Overview of Coding Tools in MPEG-4 Visual

Similar features as MPEG-2 Video or H.263

- I, P and B pictures and 16×16 macroblocks
- Intra and residual coding: 8×8 DCT, scalar quantization and run-level-last coding

Intra coding

- Prediction of transform coefficients and adaptive scanning (similar to H.263)
- DC is always predicted, prediction of AC coefficients can be selected on MB basis

Motion-compensated prediction

- Support of Inter-16×16 and Inter-8×8 mode
- Component-wise median prediction of motion vectors
- **No support of multiple reference pictures**
- **Quarter-sample accurate motion vectors** (Advanced Simple profile)
  - Half-sample interpolation: 8-tap filter
  - Quarter-sample interpolation: Bi-linear interpolation of half-sample grid
- **Global motion compensation** (rarely supported)
  - Perspective or affine motion model for background of picture
  - Prediction signal for macroblock is generated by warping

H.264 | MPEG-4 AVC

(ITU-T Rec. H.264 | ISO/IEC 14496-10)

Overview of Main Syntax Features in H.264 | MPEG-4 AVC

Commonalities with prior coding standards

- I, P and B pictures (actually I, P and B slices in H.264 | MPEG-4 AVC)
- 16×16 macroblocks supporting different coding modes
- Intra coding, uni-directional prediction or bi-prediction
- Motion vector coding with component-wise median prediction
- Transform coding with scalar quantization of residual signal

Main improvements relative to prior standards

- Decoupling of picture type, coding order and display order
- Spatial intra prediction
- Multiple reference pictures (more general than Annex U of H.263)
- More flexible partitioning of a macroblock for motion compensation
- Adaptive selection of transform size (High profile)
- Improved coding of transform coefficient levels
- Optional context-adaptive binary arithmetic coding (High profile)
- Deblocking filter (improved relative to Annex J of H.263)

Intra Coding and Residual Coding in H.264 | MPEG-4 AVC

Review of intra coding

- **Spatial intra prediction** & transform coding of residual signal
- Intra-4×4: Prediction and transform of 4×4 blocks (9 prediction modes)
- Intra-8×8: Prediction and transform of 8×8 blocks (9 prediction modes)
- Intra-16×16: Prediction of 16×16 block (4 prediction modes), transform of 4×4 blocks and second level Hadamard transform
- Intra-PCM: Direct coding of samples (fallback for high rates)

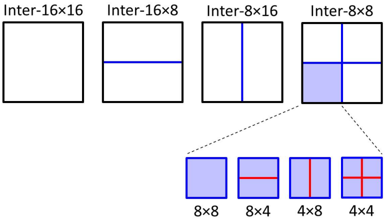
Transform coding of prediction residuals

- Transform coding using 4×4 or 8×8 transform and scalar quantization
- **Transform selection** on macroblock basis (if no MC blocks smaller than 8×8)
- Transforms are integer approximations of DCT
- **Inverse transform is specified by exact integer operations**
  - No accumulation of inverse transform mismatches
  - Encoder does not need to insert frequent intra updates
- Improved coding of transform coefficient levels (as discussed for intra)
  - Context-adaptive variable length coding (CAVLC)
  - **Context-adaptive binary arithmetic coding** (CABAC)

Improvements of Motion-Compensated Prediction

Flexible macroblock partitioning

- 4 inter coding modes with block sizes of 16×16, 16×8, 8×16 and 8×8
- For Inter-8×8 mode, sub-macroblock mode is transmitted for each 8×8 block
- Sub-macroblock mode indicates usage of 8×8, 8×4, 4×8 or 4×4 blocks



Multiple reference pictures

- Reference picture index is transmitted for each 16×16, 16×8, 8×16 or 8×8 block
- Reference picture buffer is managed by sliding window operation or explicit picture management commands (MMCO commands)
- Arbitrary construction of reference picture list using the available reference pictures

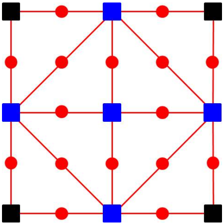
Motion-compensated prediction in B slices

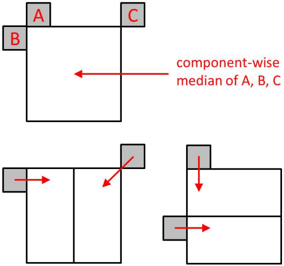
- Two reference lists (list 0 and list 1) can be arbitrarily constructed
- Prediction type (list 0, list 1 or bi-prediction) is transmitted for 16×16, 16×8, 8×16 blocks and 8×8 sub-macroblocks

Motion Vector Coding and Sub-Sample Interpolation

Motion vector accuracy and sub-sample interpolation

- Motion vector accuracy: Quarter luma sample
- Sub-sample interpolation for luma
  - Separable 6-tap filter for half-sample locations
  - Quarter-sample locations: Averaging two samples at integer and half-sample locations
- Sub-sample interpolation for chroma: Bi-linear





Coding of motion vectors

- Differential coding using a predictor
- Independent prediction per reference list
- In general: Component-wise median of the motion vectors of 3 neighboring blocks
- Some special conditions based on available motion vectors and reference picture indexes
- Special predictors: Inter-16×8 and Inter-8×16

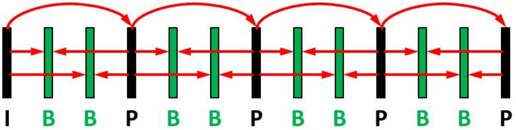
Decoupling of Picture Type, Coding and Display Order

Generalization of dependencies between pictures

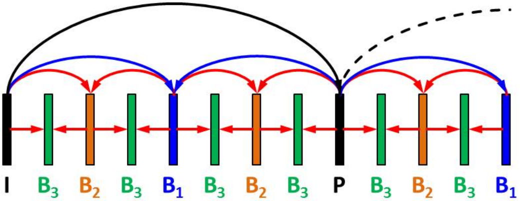
- A picture can consist of slices with different slice coding types (I, P, B)
- Each picture can be used as reference picture (as indicated in bitstream)
- Flexible coding order and construction of reference picture lists

$\Longrightarrow$ **Allows new types of prediction structures**

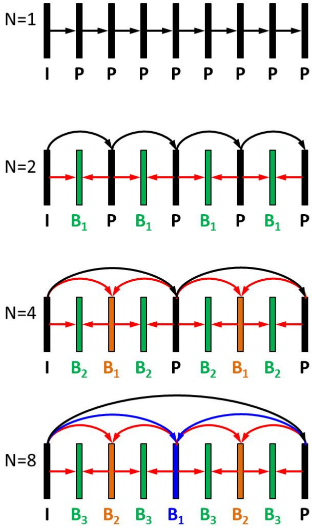
"traditional" prediction structure (2 B pictures)

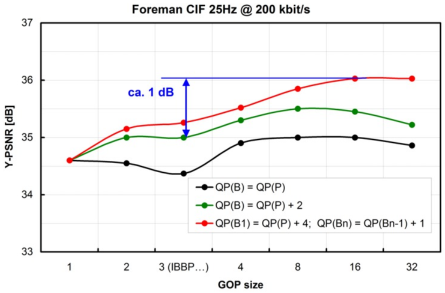


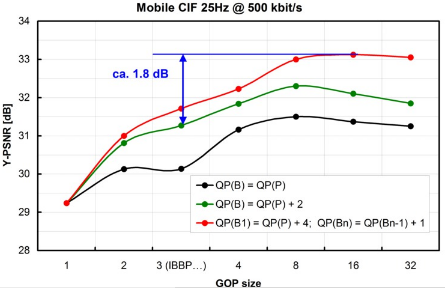
Example of a more general prediction structure:
Hierarchical B pictures



Performance of Hierarchical Prediction Structures







Subjective Quality Using Hierarchical Prediction Structures

Example: Sequence "Football" (CIF, 30Hz) at about 500 kbit/s

- Comparison of subjective quality
- Frame #206: Frame with highest QP (low PSNR) in hierarchical structure

conventional IBBP

Hierarchical B with GOP 16

Deblocking filter

Illustration of the filtering operation at block boundaries

- Filtering of $\mathbf{p_0}$ and $\mathbf{q_0}$ if all of the following conditions are fulfilled

$$\begin{aligned} |p_0 - q_0| &< \alpha(\text{QP}) \\ |p_1 - p_0| &< \beta(\text{QP}) \\ |q_1 - q_0| &< \beta(\text{QP}) \end{aligned}$$

  where

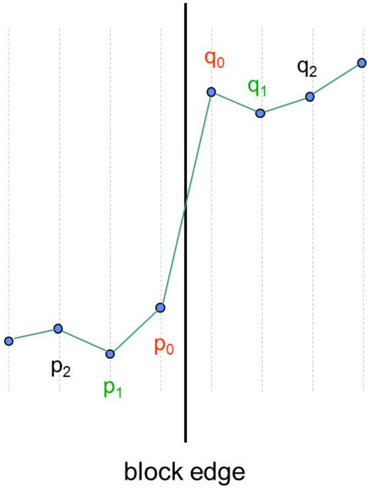
  - $\alpha(\text{QP})$ and $\beta(\text{QP})$ increase with QP
  - $\alpha(\text{QP})$ is larger than $\beta(\text{QP})$

- The sample $\mathbf{p_1}$ is additionally filtered if

$$|p_2 - p_0| < \beta(\text{QP})$$

- The sample $\mathbf{q_1}$ is additionally filtered if

$$|q_2 - q_0| < \beta(\text{QP})$$



Subjective Quality Improvement due to Deblocking Filtering

Example: Highly compressed decoded picture

without deblocking filter

with deblocking filter

H.265 | MPEG-H HEVC

(ITU-T Rec. H.265 | ISO/IEC 23008-2)

Overview of Main Syntax Features in H.265 | MPEG-H HEVC

Commonalities with H.264 | MPEG-4 AVC

- I, P and B slices and similar high-level syntax concepts
- Conceptually similar reference picture buffer management
- Conceptually similar construction of reference picture lists
- Spatial intra prediction
- Transform coding of residual with scalar quantization
- Inverse transform specification by exact integer operations
- Quarter-sample accurate motion vectors
- Deblocking filter inside motion compensation loop

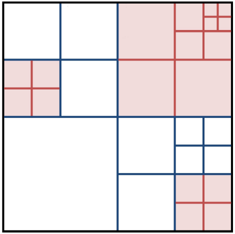
Main improvements relative to H.264 | MPEG-4 AVC

- Larger transform sizes and more flexible partitioning for transform coding
- Larger block sizes and more flexible partitioning for motion compensated prediction
- Larger number of spatial intra prediction modes
- Improved sub-sample interpolation filters
- Improved motion parameter coding
- Improved transform coefficient coding (particularly for larger transform blocks)
- Optional sample-adaptive offset filter inside motion compensation loop

Picture Partitioning, Residual and Intra Coding

Picture partitioning into coding units

- Picture partitioning into fixed-size coding tree units (CTUs) of 64×64, 32×32 or 16×16 luma samples
- Quadtree partitioning of CTUs into coding units (CUs)
- Coding units can be coded in **Intra** or **Inter** mode

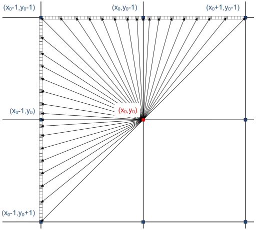


Residual coding of Inter CUs

- Quadtree partitioning of CUs into transform units (TUs)
- Transform coding of TUs with scalar quantization
- Transform sizes: 32×32, 16×16, 8×8 and 4×4
- Coding of transform coeff. levels based on 4×4 blocks
- Context-adaptive arithmetic coding (CABAC)

Coding of Intra CUs

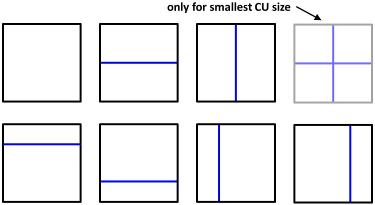
- Spatial intra prediction of TUs
- Transmission of 1 or 4 intra prediction modes per CU
- 35 intra prediction modes supported
- Same residual coding as for Inter CUs



Improvements for Motion-Compensated Prediction

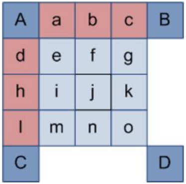
Partitioning of a CU for MCP

- Up to 8 possibilities for the partitioning of a CU into prediction units (PUs)
  - Splitting into 4 blocks only for smallest CU size
  - Asymmetric partitionings only for CUs larger than 16×16
- Selection of prediction type (list 0, list 1 or bi-prediction), reference picture(s) and motion vectors per PU



Motion vector accuracy and sample interpolation

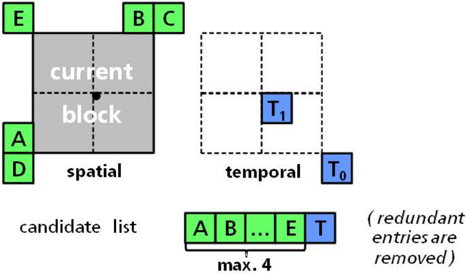
- Quarter-luma sample precision motion vectors
- Sub-sample interpolation for luma:
  - Separable 7- or 8-tap filters for all sub-sample positions
- Sub-sample interpolation for chroma:
  - Separable 4-tap filters

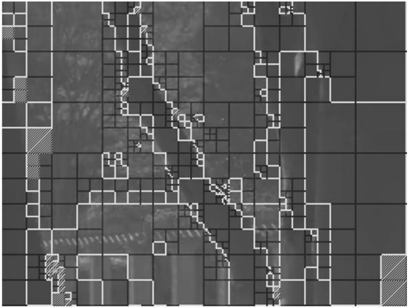


Coding of Motion Parameters: Merge Mode

Coding of prediction units in merge mode

- No transmission of prediction type, reference index or motion vector
- Prediction parameters are inferred from an already coded block
- Construction of a candidate list with up to five candidates:
  - Up to four spatially neighboring blocks
  - Up to one candidate derived from a co-located block in a reference picture
- Transmission of an index into the candidate list

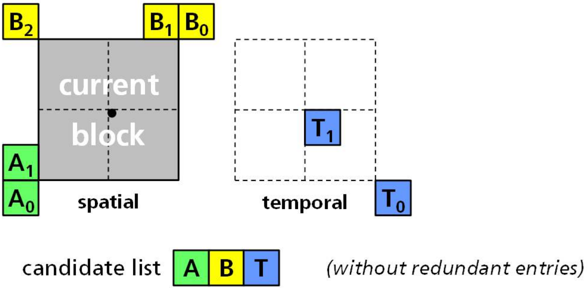




Coding of Motion Parameters: AMVP mode

Advanced motion vector prediction

- The following parameters are transmitted for PUs not coded in merge mode
  - Prediction type (list 0, list 1 or bi-prediction)
  - Per reference list: Reference index, motion vector difference, **predictor index**
- Motion vector predictor can be chosen between 3 predictors
  - Motion vector of 2 spatially neighboring blocks
  - A motion vector derived from co-located block in a reference picture



Coding Efficiency Comparison of Video Coding Standards

Coding Efficiency for Low-Delay Applications

Encoding constraints

- Bitstream characteristics suitable for low-delay applications
- Targeted application area: Video conferencing
- Constraint: Pictures are coded in display order

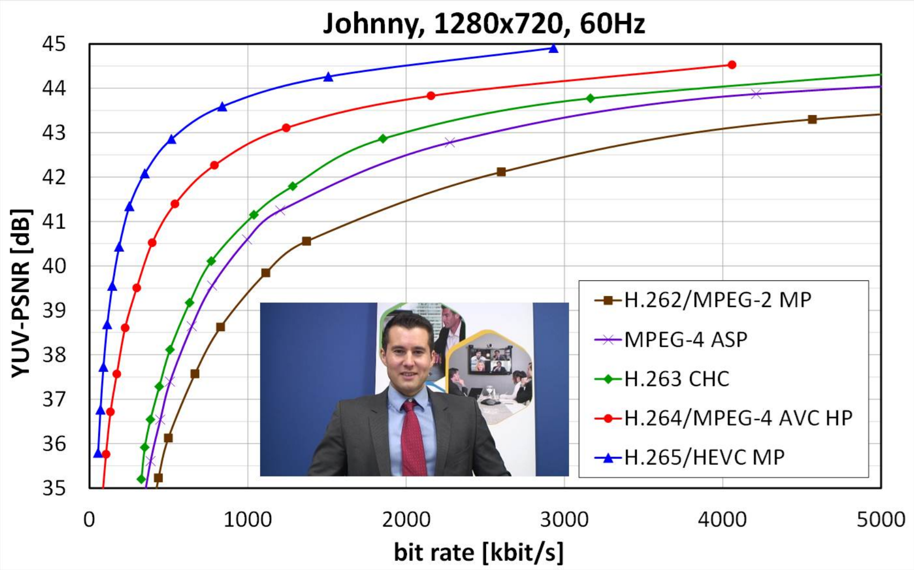
Investigated coding standards (best available configuration)

- MPEG-2 Main profile (IPPP coding structure)
- MPEG-4 Advanced Simple profile (IPPP coding structure)
- H.263 Conversational High Compression profile (IPPP coding structure)
- H.264/AVC High profile (low-delay GOP4 with P pictures)
- H.265/HEVC Main profile (low-delay GOP4 with B pictures)

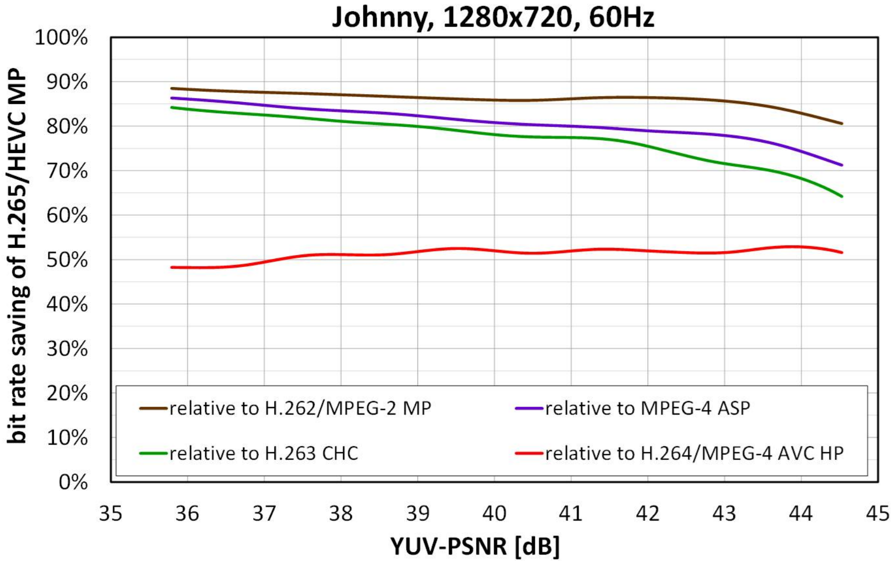
Encoder control

- Same Lagrangian encoder optimization for all encoders
- Same motion search strategy

Coding Efficiency for Low Delay: Example R-D Curves



Coding Efficiency for Low Delay: Example Rate Savings



Coding Efficiency for Low Delay: Summary

Average bit rate savings

- Averaged over covered PSNR range
- Averaged over test set of 6 video conferencing sequences

codec version	average bit rate savings relative to ...			
	H.264/AVC	H.263 CHC	MPEG-4 ASP	MPEG-2 MP
H.265/HEVC	40.3 %	67.9 %	72.3 %	80.1 %
H.264/AVC		46.8 %	54.1 %	67.0 %
H.263 CHC			13.2 %	37.4 %
MPEG-4 ASP				27.8 %

Coding Efficiency for Entertainment Applications

Encoding constraints

- Bitstream characteristics suitable for applications requiring random access
- Targeted application area: Broadcast, streaming, optical discs
- Constraint: Random access about every second (no delay constraint)

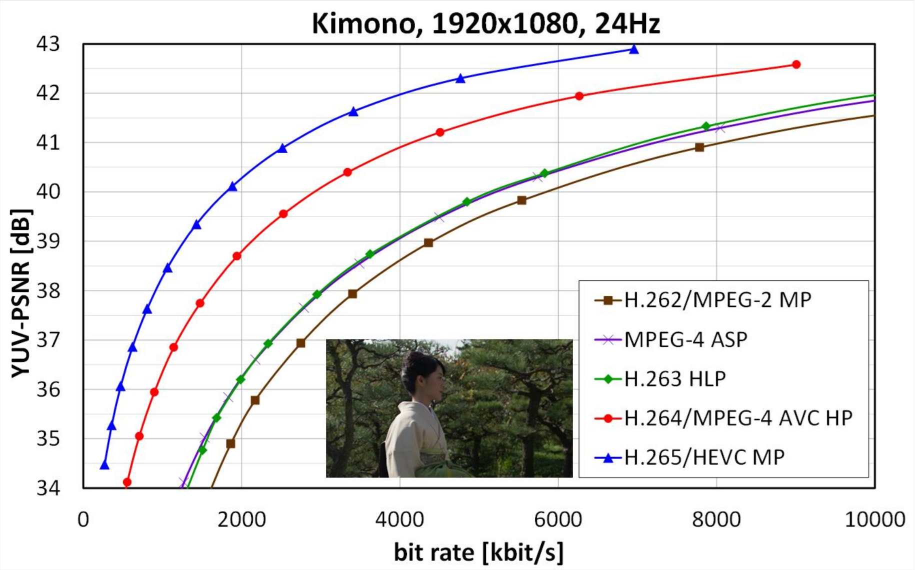
Investigated coding standards (best available configuration)

- MPEG-2 Main profile (IBBBP coding structure)
- MPEG-4 Advanced Simple profile (IBBBP coding structure)
- H.263 High Latency profile (IBBBP coding structure)
- H.264/AVC High profile (hierarchical B pictures with GOP8)
- H.265/HEVC Main profile (hierarchical B pictures with GOP8)

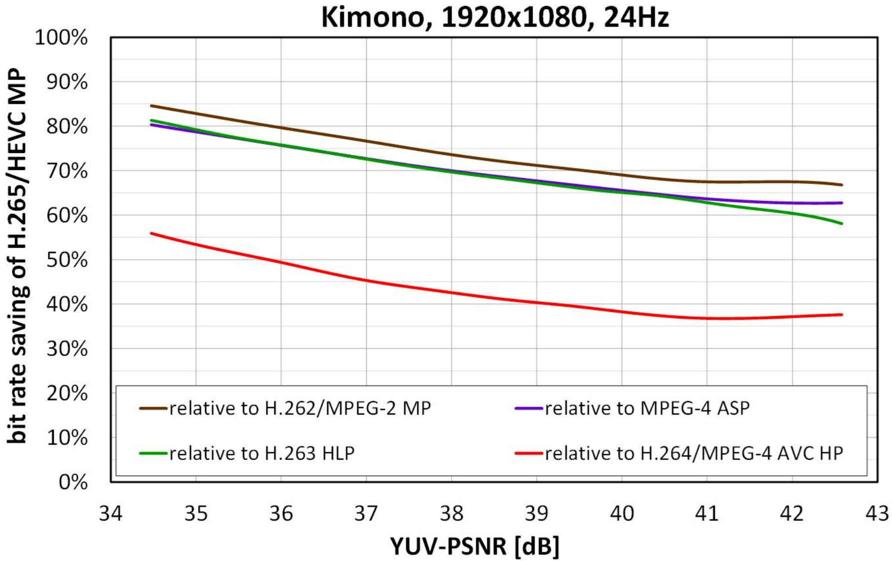
Encoder control

- Same Lagrangian encoder optimization for all encoders
- Same motion search strategy

Coding Efficiency for Random Access: Example R-D Curves



Coding Efficiency for Random Access: Example Rate Savings



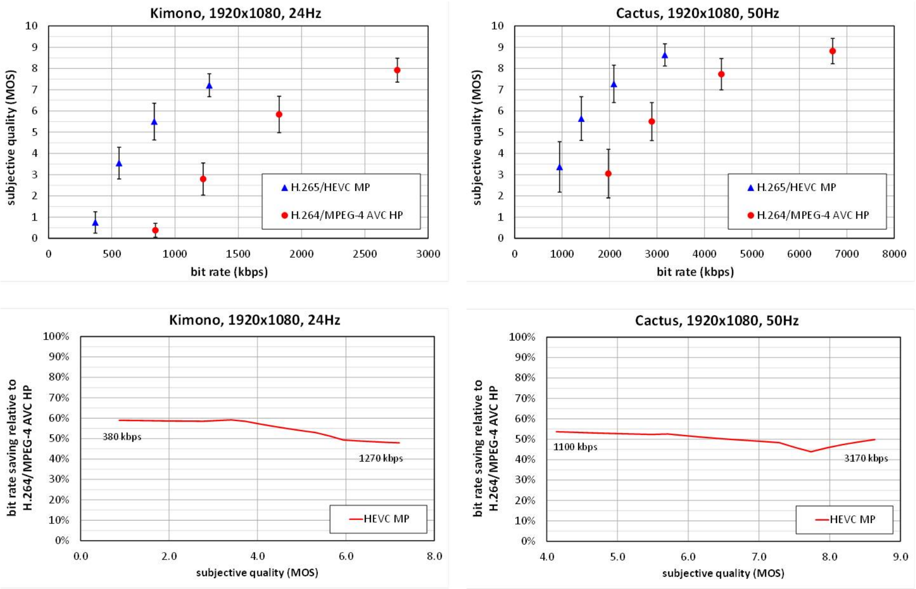
Coding Efficiency for Random Access: Summary

Average bit rate savings

- Averaged over covered PSNR range
- Averaged over test set of 9 video sequences

codec version	average bit rate savings relative to ...			
	H.264/AVC	**MPEG-4 ASP**	**H.263 HLP**	**MPEG-2 MP**
H.265/HEVC	**35.4 %**	**63.7 %**	**65.1 %**	**70.8 %**
H.264/AVC		**44.5 %**	**46.6 %**	**55.4 %**
MPEG-4 ASP			**3.9**	**19.7 %**
H.263 HLP				**16.2 %**

Subjective Comparison of H.264/AVC and H.265/HEVC



Summary on Video Coding Standards

Video coding standards

- All standards follow the hybrid video coding design
- Continuous improvement of coding efficiency
- To a large extend enabled by complexity increases

Key features for improving the coding efficiency

- Accuracy of motion vectors
- Interpolation filters
- Coding of motion vectors
- Partitioning for motion compensation, intra prediction and transform coding
- Spatial intra prediction
- Coding of transform coefficient levels
- Entropy coding
- In-loop filtering
- Generalization of supported prediction structures