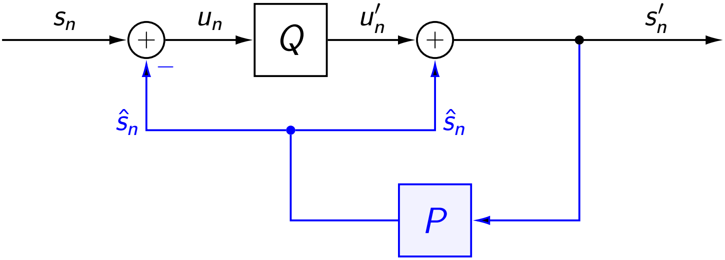
# Predictive Coding II

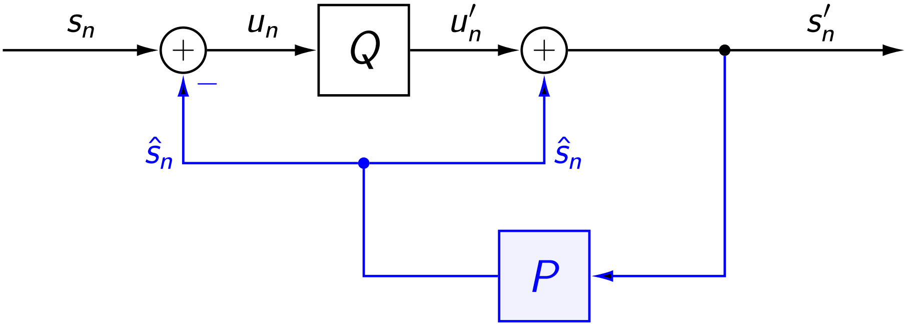


# Predictive Coding: Differential Pulse Code Modulation

## Combining prediction and quantization

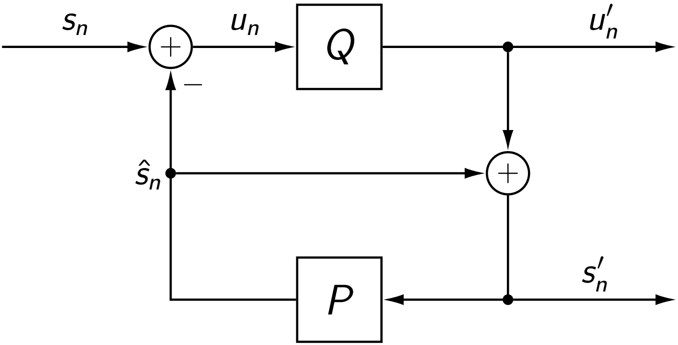
- Requires simultaneous construction of predictor at encoder and decoder

➡ Use reconstructed samples for forming predictor



# DPCM Structure

- Red-drawing yields block diagram with typical DPCM structure

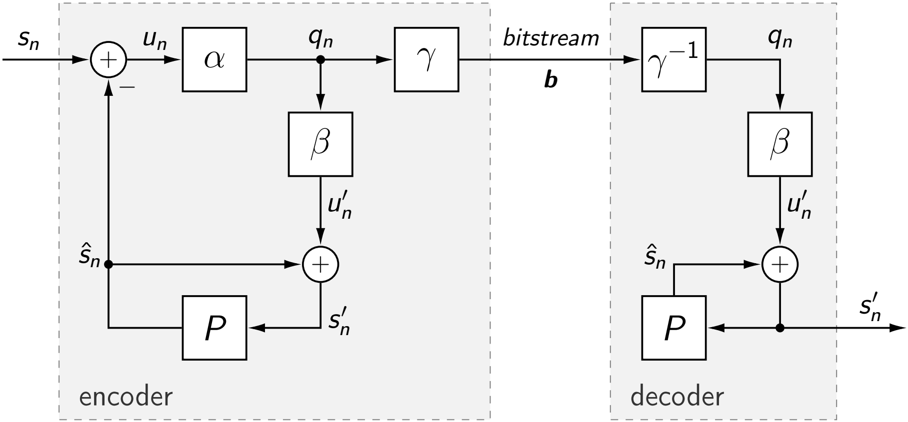



$u'_n$ — Quantized prediction residual (signal to be transmitted)

$s'_n$ — Reconstructed signal

# DPCM Encoder and Decoder

- Split quantizer into encoder mapping $\alpha$ and decoder mapping $\beta$
- Add entropy coding $\gamma$




Note: Encoder contains decoder (except entropy decoding $\gamma^{-1}$)

# DPCM: Combine Prediction and Quantization

- Consider zero-mean input signal
- Prediction $\hat{S}_n$ for a sample $S_n$ is generated by linear filtering of $K$ reconstructed samples $S'_n$ from the past

$$\hat{S}_n = \sum_{k=1}^{K} h_k \, S'_{n-k} = \sum_{k=1}^{K} h_k \, (S_{n-k} + Q_{n-k}) = \boldsymbol{h}^{\mathrm{T}} \, (\boldsymbol{S}_{n-1} + \boldsymbol{Q}_{n-1}) \qquad (1)$$

with $Q_n = S'_n - S_n$ being the quantization error signal

- Prediction error variance (for zero-mean input) is given by

$$\begin{aligned} \sigma_U^2 &= \mathrm{E}\{\, U_n^2 \,\} = \mathrm{E}\Big\{\, (S_n - \hat{S}_n)^2 \,\Big\} \\ &= \mathrm{E}\Big\{\, (S_n - \boldsymbol{h}^{\mathrm{T}} \boldsymbol{S}_{n-1} - \boldsymbol{h}^{\mathrm{T}} \boldsymbol{Q}_{n-1})^2 \,\Big\} \\ &= \mathrm{E}\{\, S_n^2 \,\} + \boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{S}_{n-1} \boldsymbol{S}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} + \boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{Q}_{n-1} \boldsymbol{Q}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} \qquad (2) \\ &\quad -2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\{\, S_n \boldsymbol{S}_{n-1} \,\} - 2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\{\, S_n \boldsymbol{Q}_{n-1} \,\} + 2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{S}_{n-1} \boldsymbol{Q}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} \end{aligned}$$

# DPCM: Combine Prediction and Quantization

- Continue: Prediction error variance

$$
\begin{aligned}
\sigma_U^2 &= \mathrm{E}\{\, S_n^2 \,\} + \boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{S}_{n-1} \boldsymbol{S}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} + \boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{Q}_{n-1} \boldsymbol{Q}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} \\
&\quad -2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\{\, S_n \boldsymbol{S}_{n-1} \,\} - 2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\{\, S_n \boldsymbol{Q}_{n-1} \,\} + 2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{S}_{n-1} \boldsymbol{Q}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} \\
&= \sigma_S^2 - 2\boldsymbol{h}^{\mathrm{T}} \boldsymbol{c}_n + \boldsymbol{h}^{\mathrm{T}} \boldsymbol{C}_K \boldsymbol{h} + \\
&\quad \boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{Q}_{n-1} \boldsymbol{Q}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h} - 2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\{\, S_n \boldsymbol{Q}_{n-1} \,\} + 2\boldsymbol{h}^{\mathrm{T}} \mathrm{E}\Big\{\, \boldsymbol{S}_{n-1} \boldsymbol{Q}_{n-1}^{\mathrm{T}} \,\Big\} \boldsymbol{h}
\end{aligned}
\tag{3}
$$

➡ Optimal prediction coefficients $\boldsymbol{h}$ depend on

- Statistical properties of input signal ($\boldsymbol{C}_K$ and $\boldsymbol{c}_n$)
- Statistical properties of quantization error signal

➡ Optimal quantizer design depends on statistical properties of prediction error

➡ Iterative numerical optimization of quantizer and predictor design

# DPCM for Gauss-Markov Source

## Estimate distortion-rate function for DPCM of Gauss-Markov Source

- Consider zero-mean Gauss-Markov process

$$S_n = Z_n + \varrho \cdot S_{n-1} \tag{4}$$

- Consider a one-tap linear prediction filter $h$
- Auto-covariance matrix and cross-covariance vector

$$\boldsymbol{C}_1 = \sigma_S^2 \qquad \text{and} \qquad \boldsymbol{c}_n = \sigma_S^2\,\varrho \tag{5}$$

- Prediction error variance

$$\begin{aligned}
\sigma_U^2 &= \sigma_S^2 - 2h\sigma_S^2\varrho + h^2\sigma_S^2 + h^2\mathrm{E}\{\,Q_{n-1}^2\,\} \\
&\qquad -2h\mathrm{E}\{\,S_nQ_{n-1}\,\} + 2h^2\mathrm{E}\{\,S_{n-1}Q_{n-1}\,\} \\
&= \sigma_S^2(1-2h\varrho+h^2) + h^2\mathrm{E}\{\,Q_{n-1}^2\,\} \\
&\qquad -2h\mathrm{E}\{\,Z_nQ_{n-1}\,\} - 2h\varrho\mathrm{E}\{\,S_{n-1}Q_{n-1}\,\} + 2h^2\mathrm{E}\{\,S_{n-1}Q_{n-1}\,\} \\
&= \sigma_S^2(1-2h\varrho+h^2) + h^2\mathrm{E}\{\,Q_{n-1}^2\,\} + 2h(h-\varrho)\mathrm{E}\{\,S_{n-1}Q_{n-1}\,\} \qquad (6)
\end{aligned}$$

# DPCM for Gauss-Markov Source

- Prediction error variance

$$\sigma_U^2 = \sigma_S^2(1 - 2h\varrho + h^2) + h^2 \mathrm{E}\{\, Q_{n-1}^2 \,\} + 2h(h-\varrho)\mathrm{E}\{\, S_{n-1} Q_{n-1} \,\} \tag{7}$$

- Consider optimal filter for high rates: $h = \varrho$

$$\sigma_U^2 = \sigma_S^2(1 - \varrho^2) + \varrho^2\, D \tag{8}$$

- Distortion-rate function of scalar quantizer

$$D(R) = \sigma_U^2 \cdot g(R) \tag{9}$$

- Resulting prediction error variance

$$\sigma_U^2 \;=\; \sigma_S^2(1 - \varrho^2) + \varrho^2\, \sigma_U^2 \cdot g(R)$$

$$\rightarrow \quad \sigma_U^2 \;=\; \sigma_S^2 \cdot \frac{1 - \varrho^2}{1 - \varrho^2\, g(R)} \tag{10}$$

# DPCM for Gauss-Markov Source

## Upper bound for distortion-rate function

- Consider filter $h = \varrho$ (optimal filter for high rates)

➡ Prediction error variance

$$\sigma_U^2 = \sigma_S^2 \cdot \frac{1-\varrho^2}{1-\varrho^2\, g(R)} \tag{11}$$

➡ Operational distortion-rate function

$$\begin{aligned} D(R) &= \sigma_U^2 \cdot g(R) \\ &= \sigma_S^2 \cdot \frac{1-\varrho^2}{1-\varrho^2\, g(R)} \cdot g(R) \end{aligned} \tag{12}$$

➡ Operational distortion-rate function at high rates

$$D(R) = \varepsilon^2 \cdot \sigma_S^2 \cdot (1-\varrho^2) \cdot 2^{-2R} \tag{13}$$

# Joint Design of Predictor and Quantizer

Initialization

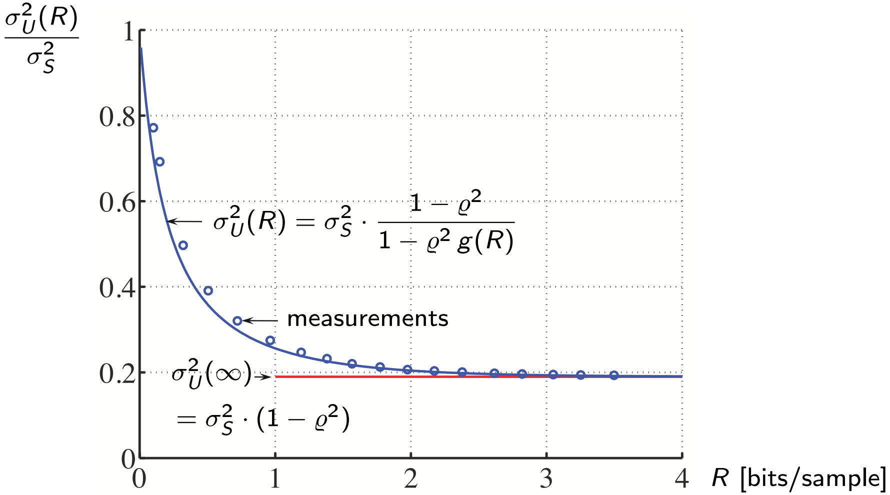
- Choose sufficiently large training set $\{s_n\}$ and a Lagrange multiplier $\lambda$
- Design optimal predictor $\boldsymbol{h}$ for $s_n$ given original samples $s_{n-1}, s_{n-2}, \cdots$

Iterative design of predictor and quantizer

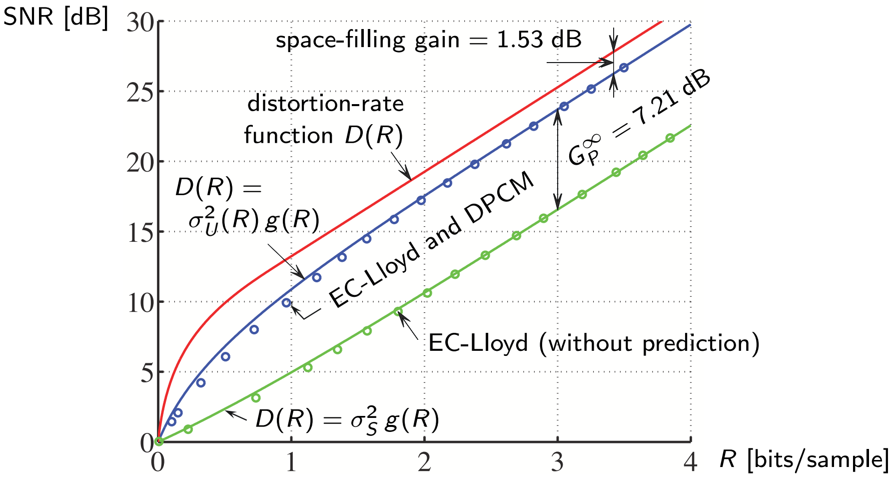
1. Design quantizer for predictor $\boldsymbol{h}$
   - a. Generate training set $\{u_n\}$ by conducting DPCM encoding of training set $\{s_n\}$ with predictor $\boldsymbol{h}$ and current quantizer $(\alpha, \beta, \gamma)$
   - b. Design quantizer $(\alpha, \beta, \gamma)$ for training set $\{u_n\}$ and given $\lambda$
   - c. Repeat previous two steps until convergence
2. Generate reconstructed samples $\{s'_n\}$ by conducting DPCM encoding of training set $\{s_n\}$ with predictor $\boldsymbol{h}$ and quantizer $(\alpha, \beta, \gamma)$
3. Design optimal predictor $\boldsymbol{h}$ for $s_n$ given reconstructed samples $s'_{n-1}, s'_{n-2}, \cdots$
4. Repeat previous three steps until convergence

# Experiment: DPCM of Gauss-Markov Source ($\varrho = 0.9$)

➡ Prediction error variance $\sigma_U^2$ depends on bit rate



# Experiment: DPCM of Gauss-Markov Source ($\varrho = 0.9$)




- ➡ High rates: Shape and memory gain are achievable
- ➡ Low rates: Worse than transform coding

# Adaptive DPCM

**Audio signals, images, videos**

- Instationary signals
- Single predictor not suitable for entire signal
- Adapt predictor (incl. observation set) during encoding/decoding

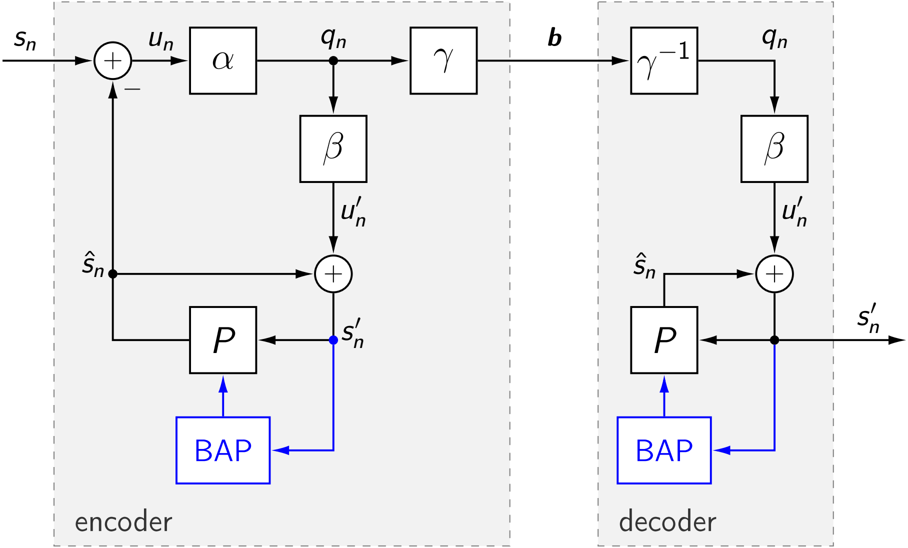
## Backward adaptation

- Simultaneously estimate predictor at encoder and decoder side
- Estimation has to be based on reconstructed samples
- No additional rate, but accuracy, complexity, error resilience issues

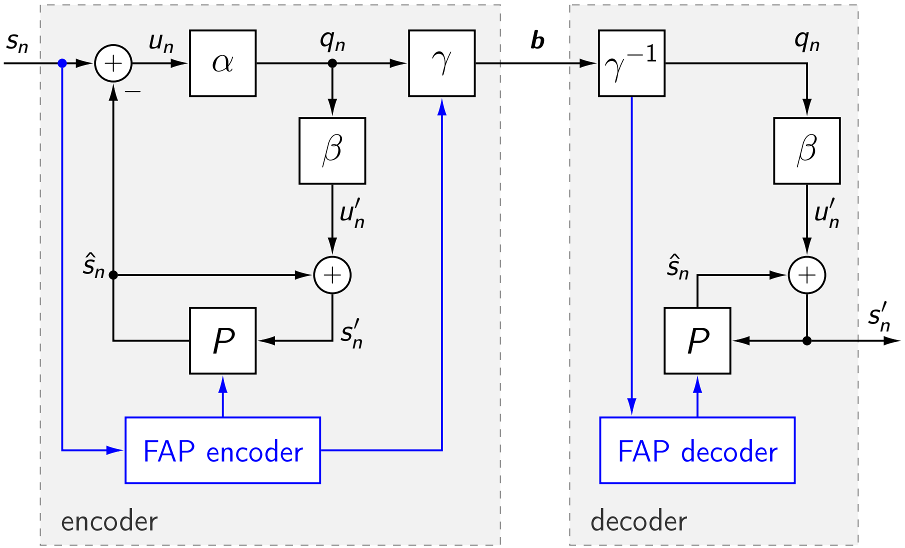
## Forward adaptation

- Analyze signal at encoder side
- Send predictor as part of the bitstream (requires additional bit rate)
- Simple method: Switched predictors

# Backward-Adaptive DPCM



# Forward-Adaptive DPCM



# Prediction in Video Coding

**Intra-picture prediction** (forward adaptive)

- Prediction of a block of samples using neighboring reconstructed samples
- Block of residual samples is typically coded using transform coding
- One of multiple available predictors is used (signaled in bitstream)

**Motion-compensated prediction** (forward adaptive)

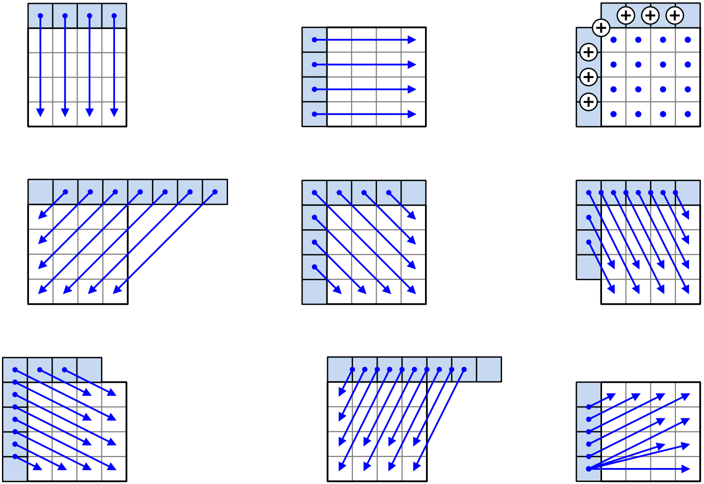
- Prediction of a block of samples using already reconstructed pictures
- Motion vector indicates observation set as well as prediction filter $\boldsymbol{h}$
- Block of residual samples is typically coded using transform coding
- Encoder: Motion search for finding suitable prediction block

**Motion vector prediction** (fixed or forward adaptive)

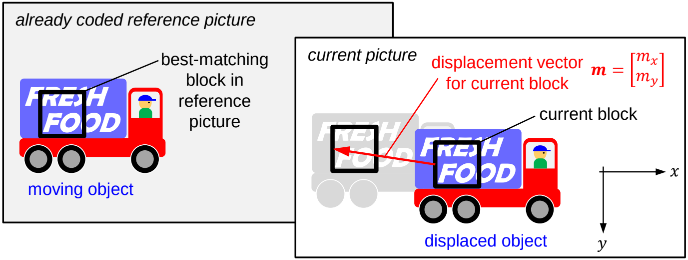
- Prediction of motion vectors before lossless coding
- Use motion vectors of neighboring blocks for predicting current motion vector
- Only difference vector is transmitted

# Intra-Picture Prediction

Example: 9 intra prediction modes in H.264/AVC



# Motion-Compensated Prediction




- Predict current block using a displaced block in an already coded picture

$$\hat{s}[x, y] = s'_{\text{ref}}[x + m_x, y + m_y]$$

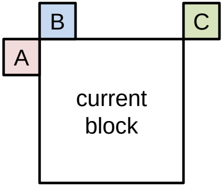
- Displacement is characterized by **displacement vector** or **motion vector**

$$\boldsymbol{m} = (\, m_x, m_y \,)^{\mathrm{T}}$$

➡ Estimate suitable motion vector in encoder $\Longrightarrow$ **Motion estimation**

➡ Transmit motion vector to decoder as part of the bitstream

# Motion Vector Prediction




## Example: Median prediction

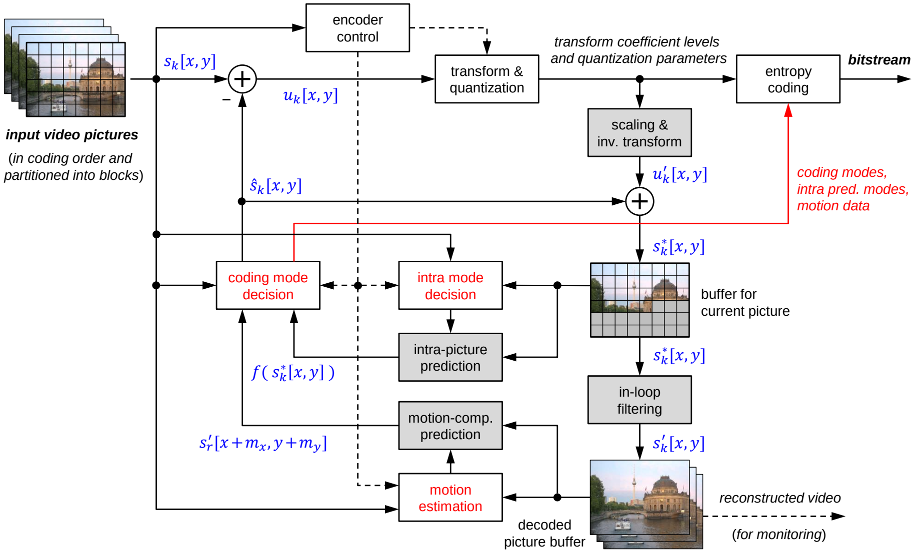
- Component-wise median of three neighboring blocks

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

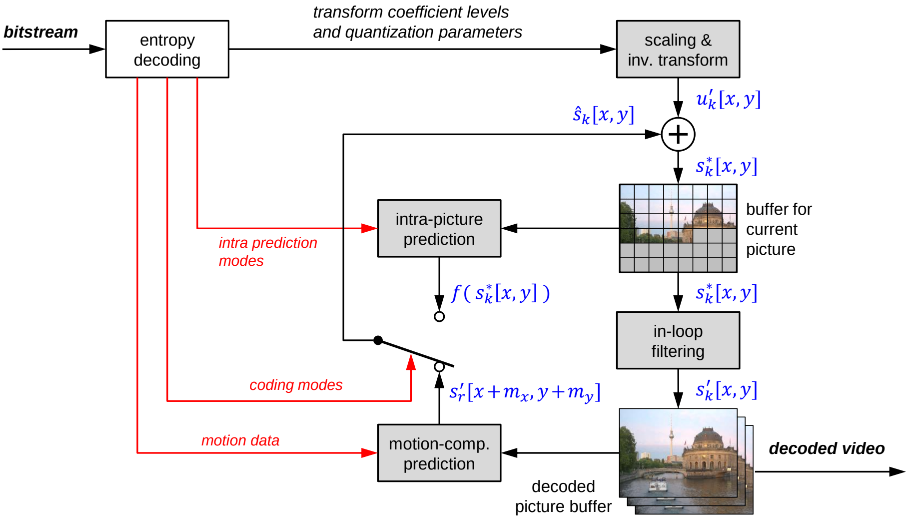
➡ Coding of prediction error (lossless)

$$\Delta m_x = m_x - \widehat{m}_x$$
$$\Delta m_y = m_y - \widehat{m}_y$$

# Hybrid Video Encoder



# Hybrid Video Decoder



# Part Summary

## Prediction

- Estimate random variable from already observed random variables
- Optimal predictor: Conditional mean

## Linear and affine prediction

- Simple and efficient structure
- Optimal predictor given by Yule-Walker equations
- AR($m$) processes: Optimal predictor has $m$ coefficients
- Optimal prediction error is orthogonal to input signal
- Non-matched predictor can increase signal variance

## Predictive quantization: DPCM

- Combination of affine prediction and ECSQ is simple and efficient
- Can exploit linear dependencies between samples
- Forward and backward adaptation