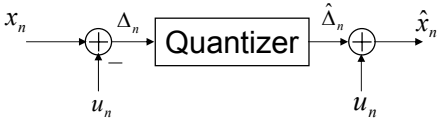
ECE 8873
Data Compression and Modeling

Lecture 8:
Predictive & Differential Coding

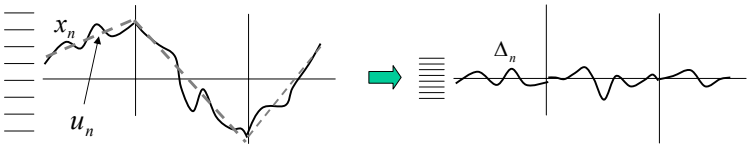
School of Electrical and Computer Engineering
Georgia Institute of Technology
Spring, 2004

Difference Quantization



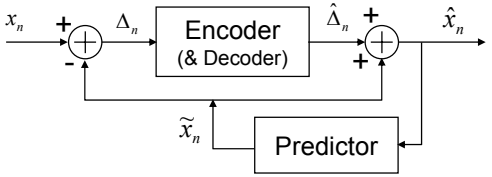
$$\Delta_n = x_n - u_n \qquad \hat{\Delta}_n = \hat{x}_n - u_n \quad \text{or} \quad \hat{x}_n = \hat{\Delta}_n + u_n$$

$$E[(X_n - \hat{X}_n)^2] = E[(\Delta_n - \hat{\Delta}_n)^2]$$



Recall the structural and random components of information. It is beneficial to treat them separately.

General Block Diagram



$$\Delta_n = x_n - \tilde{x}_n = x_n - f(\hat{x}_{n-1}, \hat{x}_{n-2}, \cdots) = x_n - f(H_n)$$

$$\hat{\Delta}_n = \beta(\alpha(\Delta_n))$$

$$\hat{x}_n = \hat{\Delta}_n + \tilde{x}_n$$

But, the notion of prediction can also be applied to probability measures – foresee change in distribution based on what is or are already observed.

Type of Predictors

- Linear $f(\hat{x}_{n-1}, \hat{x}_{n-2}, \cdots) = \sum_{i=1}^{p} a_i \hat{x}_{n-i}$
- Non-linear $f(\hat{x}_{n-1}, \hat{x}_{n-2}, \cdots)$

The simplest linear case: $\tilde{x}_n = f(\hat{x}_{n-1}, \hat{x}_{n-2}, \cdots) = \hat{x}_{n-1}$

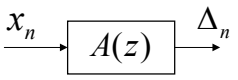
$$\Delta_n = x_n - \hat{x}_{n-1}$$

Intuition:

Assume $\hat{x}_{n-1} \approx x_{n-1}$

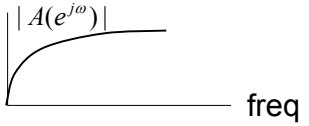
$$\Delta_n = x_n - x_{n-1}$$

$A(z)$ is high-pass.



$$A(z) = 1 - z^{-1}$$

Since most signals are low-pass type, the differentiator tends to "flatten" the original spectrum.



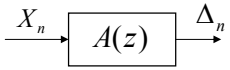
Spectral Flatness Measure

Consider a 0-mean process X_n with power spectral density $X(\omega)$

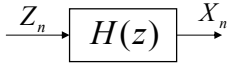
$$\sigma_X^2 = \frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)d\omega \qquad \gamma_X^2 = \frac{\exp\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi}\log X(\omega)d\omega\right\}}{\frac{1}{2\pi}\int_{-\pi}^{\pi} X(\omega)d\omega} = \frac{\eta_X^2}{\sigma_X^2}$$

γ_X^2 is called the spectral flatness measure; $0 < \gamma_X^2 \le 1$

If X_n is a white noise process, $X(\omega) = \sigma_X^2$ and $\gamma_X^2 = \frac{\sigma_X^2}{\sigma_X^2} = 1$



$$\tilde{x}_n = \sum_{i=1}^{p} a_i x_{n-i} \qquad \Delta_n = x_n - \tilde{x}_n = x_n - \sum_{i=1}^{p} a_i x_{n-i}$$



$$p \to \infty \qquad \lim_{p\to\infty} \min_{A} \sigma_\Delta^2 = \sigma_Z^2 = \eta_X^2 = \exp\left\{\frac{1}{2\pi}\int_{-\pi}^{\pi}\log X(\omega)d\omega\right\}$$

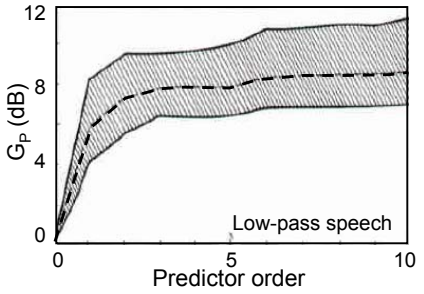
Δ_n has the "most" flattened psd. Prediction is to whiten the input psd.

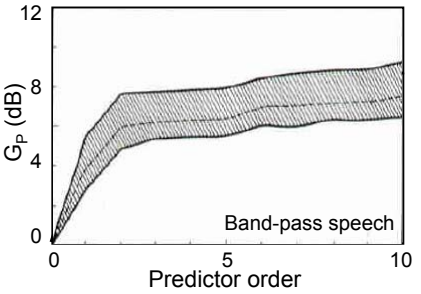
Prediction Gain

- Prediction thus reduces the signal variance.

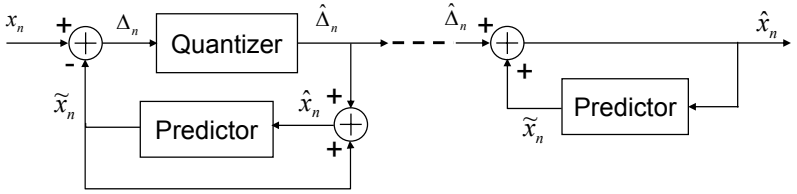
Prediction gain $G_P \equiv \frac{\sigma_X^2}{\sigma_\Delta^2}$ or $G_P(\text{dB}) \equiv 10\log_{10}\frac{\sigma_X^2}{\sigma_\Delta^2}$

$$(G_P)_{\max} \equiv \frac{\sigma_X^2}{\sigma_Z^2} = \frac{1}{\gamma_X^2}$$





Differential PCM



$$\Delta_n = x_n - \tilde{x}_n \qquad \hat{\Delta}_n = \Delta_n - q_n \qquad \hat{x}_n = \hat{\Delta}_n + \tilde{x}_n$$

$$x_n - \hat{x}_n = (x_n - \tilde{x}_n) - (\hat{x}_n - \tilde{x}_n) = \Delta_n - \hat{\Delta}_n$$

$$\tilde{x}_n = \sum_{i=1}^{p} a_i \hat{x}_{n-i}$$

The Predictor can be fixed or adaptive.

In image coding, interframe prediction has been attempted early on.

Linear Prediction

$e_n = x_n - \tilde{x}_n = x_n - \sum_{i=1}^{p} a_i x_{n-i}$ is the prediction error.

Choose mean squared error as the performance measure

$$\text{E} = E[(X_n - \tilde{X}_n)^2] = E\left[\left(X_n - \sum_{i=1}^{p} a_i X_{n-i}\right)^2\right]$$

$$\frac{\partial}{\partial a_i}\text{E} = \frac{\partial}{\partial a_i} E[(X_n - \tilde{X}_n)^2] = E\left[2(X_n - \tilde{X}_n)\frac{\partial[-\tilde{X}_n]}{\partial a_i}\right] = 0$$

$$\Rightarrow E[(X_n - \tilde{X}_n)X_{n-i}] = 0$$

$$\begin{bmatrix} r(1) \\ r(2) \\ r(3) \\ \vdots \\ r(p) \end{bmatrix} = \begin{bmatrix} r(0) & r(1) & r(2) & \cdots & r(p-1) \\ r(1) & r(0) & r(1) & \cdots & r(p-2) \\ r(2) & r(1) & r(0) & \cdots & r(p-3) \\ \cdots & & \cdots & & \cdots \\ r(p-1) & r(p-2) & r(p-3) & & r(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix}$$

The set of normal equations, Yule-Walker equations, or Wiener-Hopf equations

Solving the Normal Equations

$E^{(0)} = r(0)$ Then for $i = 1, 2, \cdots, p$ iterate:

$$k_i = \left\{ r(i) - \sum_{j=1}^{i-1} a_j^{(i-1)} r(|i-j|) \right\} \left[E^{(i-1)} \right]^{-1}$$

$$a_i^{(i)} = k_i \quad \text{and} \quad a_j^{(i)} = a_j^{(i-1)} - k_i a_{i-j}^{(i-1)}, \qquad j = 1, 2, \cdots, i-1$$

$$E^{(i)} = (1 - k_i^2) E^{(i-1)}$$

Show example for $p = 2$

- Efficient algorithms exist for solving the equations.
- k is called the reflection coefficient and is better than the predictor coefficients for coding purposes; value bounded in (-1,1) for stable all-pole filter.

Autocorrelation Function

- But speech is time-varying; how to estimate autocorrelation? Also in discrete time implementation
- Short-time spectral analysis framework:

$x_n(m) = x(n+m)$ and $e_n(m) = e(n+m)$

$$E_n = \sum_m e_n^2 = \sum_m \left(x_n(m) - \sum_{i=1}^{p} a_i x_n(m-i) \right)^2$$

- Range of summation is where we want the fitting to focus on.
- Now the "autocorrelation" is being estimated as summation rather than expectation, how does the discrepancy impact the result theoretically and practically?
- Even more importantly, how should we look at the issue of autocorrelation estimation in the context of variance reduction for COMPRESSION?

Autocorrelation & Covariance Methods

- The Autocorrelation method

Apply tapered window on data sequence to localize the analysis area

$$x_n(m) = \begin{cases} s(n+m) \bullet w(m) & 0 \le m \le N-1 \\ 0 & \text{elsewhere} \end{cases}$$

$E_n = \sum_{m=0}^{N-1+p} e_n^2$ where N is the frame length

$$r(i-k) = \sum_{m=0}^{N-1-(i-k)} x_n(m) x_n(m+i-k) = r(k-i) = r(|i-k|)$$

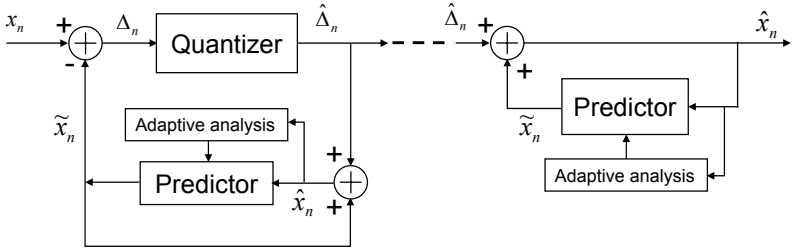
- The Covariance method

$$E_n = \sum_{m=0}^{N-1} e_n^2 \qquad r'(i,k) = \sum_{m=0}^{N-1} x_n(m-i) x_n(m-k) \quad \text{for } 1 \le i \le p, 0 \le k \le p$$

$$r'(i,k) = \sum_{m=-i}^{N-1-i} x_n(m) x_n(m+i-k) \quad \text{for } 1 \le i \le p, 0 \le k \le p$$

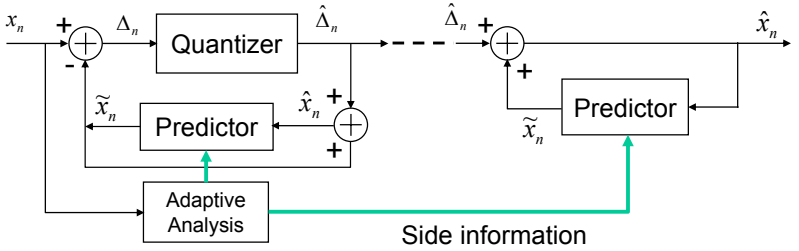
Since $r'(i,k) \neq r'(k,i)$ the system equation does not have the nice Toeplitz structure.

Adaptive Prediction – Backward (APB)



- For signals with varying characteristics, adapting the predictor accordingly will result in higher prediction gain.
- Again, the goal is to reduce the variance of Δ_n.
- Backward adaptation means using only the reconstructed values for prediction.

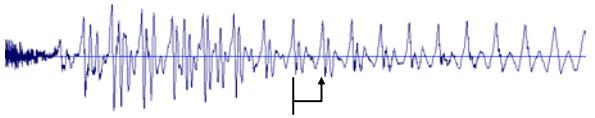
Adaptive Prediction – Forward (APF)



- Original signal is subject to buffering and analysis to produce the "optimal" predictor to reduce the variance of Δ_n .
- Buffering causes delay.
- Require transmission of the predictor specification (many possible representations) as side information so that the receiver can reproduce the signal.

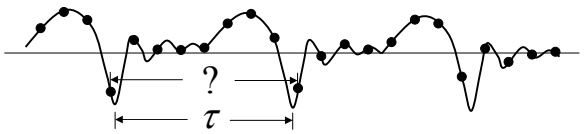
Near and Distant Sample Prediction

- Speech contains at times periodic structure due to vocal cord vibration – pitch.
- Repetition means the existence of structural information that can be extracted first to enhance the efficiency of coding.



- Try to make clear the concept of fundamental frequency, pitch, pitch prediction, discrete time implementation, pitch resolution, fundamental frequency resolution.

Pitch Prediction



$$\tilde{x}_n = \sum_{i=-p}^{p} a_i \hat{x}_{n-\tau-i}$$

τ = pitch estimate

$p = 1 \Rightarrow 3$ - tap pitch predictor

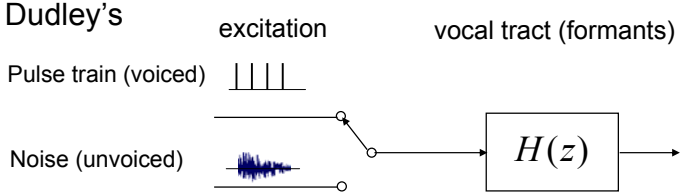
Or

$$\tilde{x}_n = \sum_{i=1}^{p} a_i \hat{x}_{n-i}$$ where p is large enough to cover a pitch period

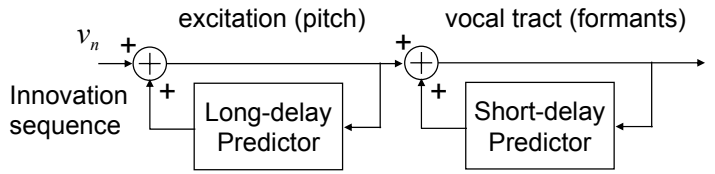
- Cannot afford to transmit large number of parameters
- Use backward prediction (original form anyway): G.728 at 16 kbps

Speech Production Models

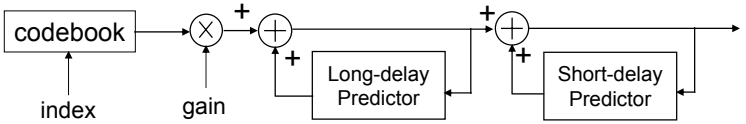
- Dudley's



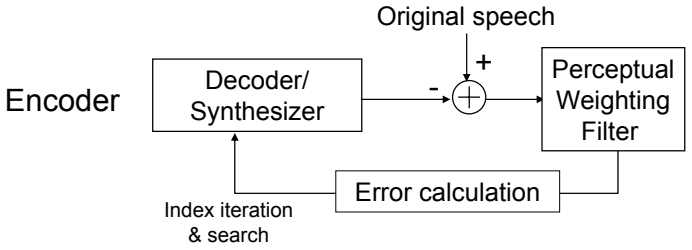
- Alternative



Coding with the Alternative Model

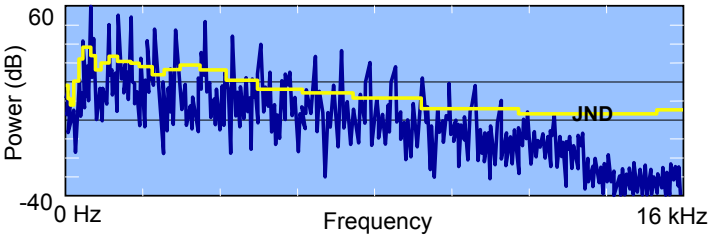


Decoder/synthesizer

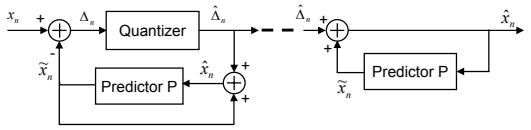


Perceptual Weighting & Post-Filtering

- Perceptual weighting – to make excitation codeword search perceptually meaningful
- Post-filtering – to make remnant coding noise less perceptible
- Both are related to noise shaping to take advantage of the auditory masking effect.



Issue of Noise Feedback



$$\tilde{x}_n = \sum_{i=1}^{p} a_i \hat{x}_{n-i}$$

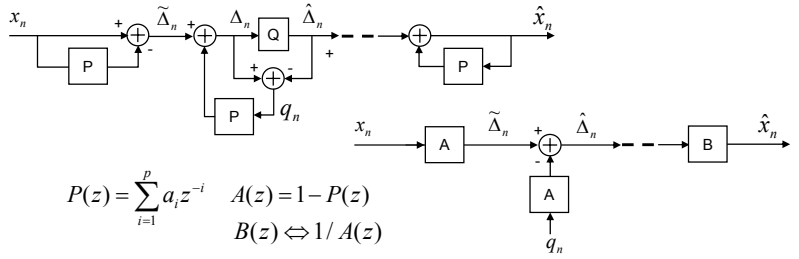
$$\hat{x}_n = \hat{\Delta}_n + \tilde{x}_n \qquad \Delta_n = x_n - \tilde{x}_n$$

$$\hat{\Delta}_n = \Delta_n - q_n$$

$$x_n - \hat{x}_n = (x_n - \tilde{x}_n) - (\hat{x}_n - \tilde{x}_n) = \Delta_n - \hat{\Delta}_n = q_n$$

Quantization noise is the same as reconstruction error

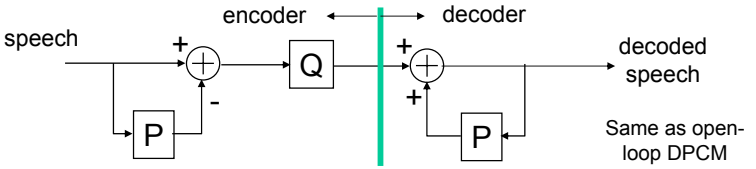
$$\Delta_n = x_n - \sum_{i=1}^{p} a_i \hat{x}_{n-i} = x_n - \sum_{i=1}^{p} a_i x_{n-i} + \sum_{i=1}^{p} a_i q_{n-i}$$



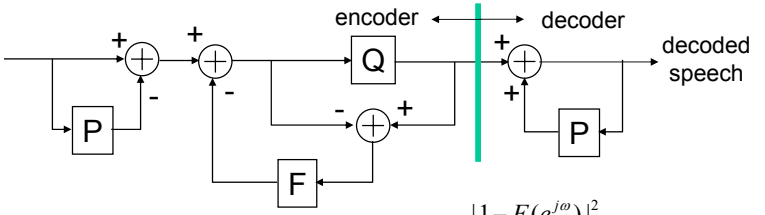
$$P(z) = \sum_{i=1}^{p} a_i z^{-i} \qquad A(z) = 1 - P(z)$$

$$B(z) \Leftrightarrow 1/A(z)$$

Noise Shaping in DPCM



Without noise shaping: $N(e^{j\omega}) \propto \dfrac{1}{|1 - P(e^{j\omega})|^2}$



With noise shaping: $N(e^{j\omega}) \propto \dfrac{|1 - F(e^{j\omega})|^2}{|1 - P(e^{j\omega})|^2}$

Noise Shaping Filter

$$F(z) = P(\alpha^{-1} z) \qquad 0 \le \alpha \le 1$$

$$\alpha = 0 \quad \Rightarrow \quad N(e^{j\omega}) \propto \frac{1}{|1 - P(e^{j\omega})|^2} \quad \text{same as open - loop DPCM}$$

$\alpha = 1 \quad \Rightarrow \quad N(e^{j\omega})$ is "flat" (as long as the noise independence assumption holds) and known as the DPCM noise power spectrum

Other ways to "manage" the noise:

- Use perceptual criterion to determine bit allocation, particularly in the transform domain, to create non-uniform level of noise across frequency.

Predictive & Differential Coding in Video

- Simple choice: Consider each frame as an image – perform 2-D prediction within each image and code the residual
- Better choice: take inter-frame dependency into account; code only the inter-frame differences (the "delta" frame)
  - As with other predictive coding schemes, errors due to imperfect transmission or storage may propagate to later frames
  - Difficulty in supporting random access to any portion of the video sequence
  - Need to consider periodic "refreshment" – thus key frames

Key Frames

- Regularly paced: code independently once every n frames, differentially in-between frames
- Synchronized with scene change, but still other structural change to consider before coding the difference frames sequentially:
  - Objects moving
  - Panning
  - Zooming

Motion prediction/compensation first.

Encoder with MCP

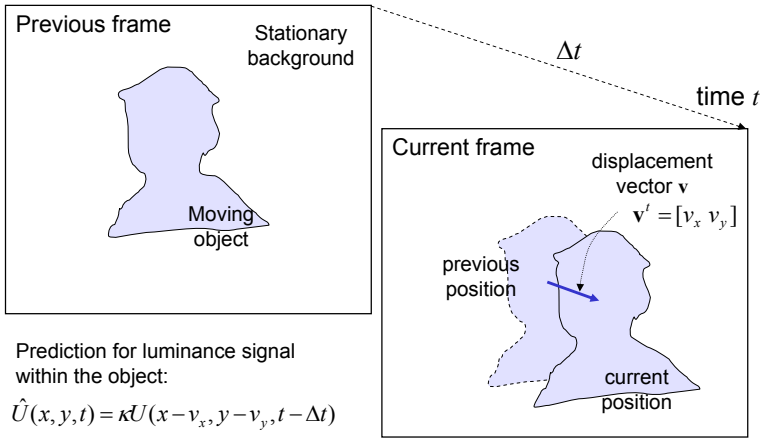
Encoder:

- Identify corresponding moving parts between consecutive frames
- Divide the frame into blocks according to these parts
- Assign a motion vector to each block
- Use the previous frame and the motion vectors to form a predicted version of the current frame
- Send the motion blocks and difference (residual)

Decoder

- Use the previous frame and the motion vectors as a predicted version of the current frame
- Decode and add back the residual

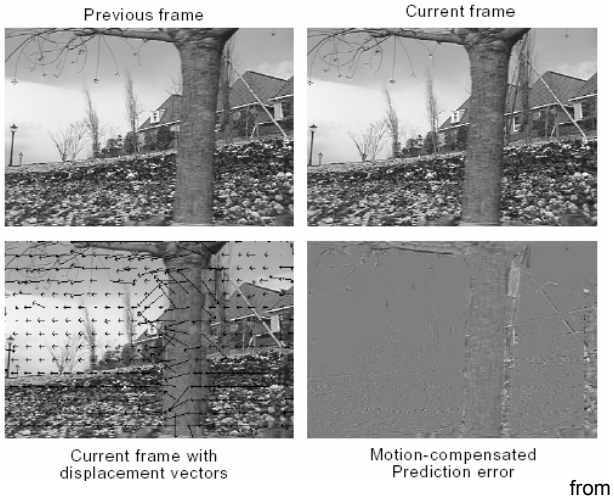
Motion Compensated Prediction



Prediction for luminance signal within the object:

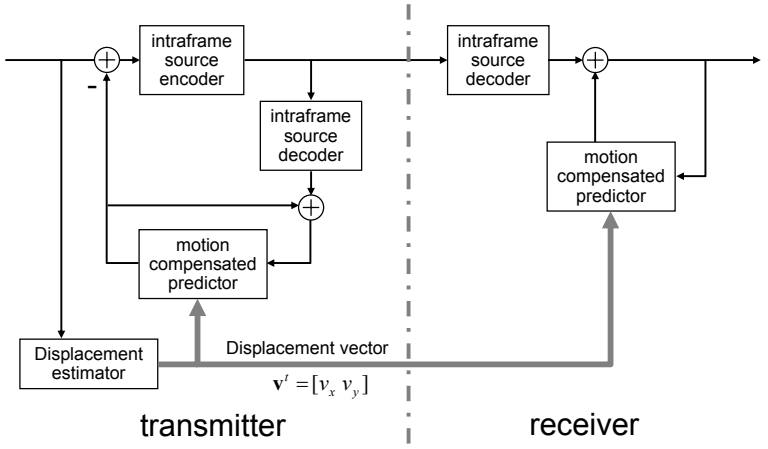
$$\hat{U}(x, y, t) = \kappa U(x - v_x, y - v_y, t - \Delta t)$$

MCP - Example



from Bernd Girod

Motion-Compensated Prediction



MPEG Video Coding