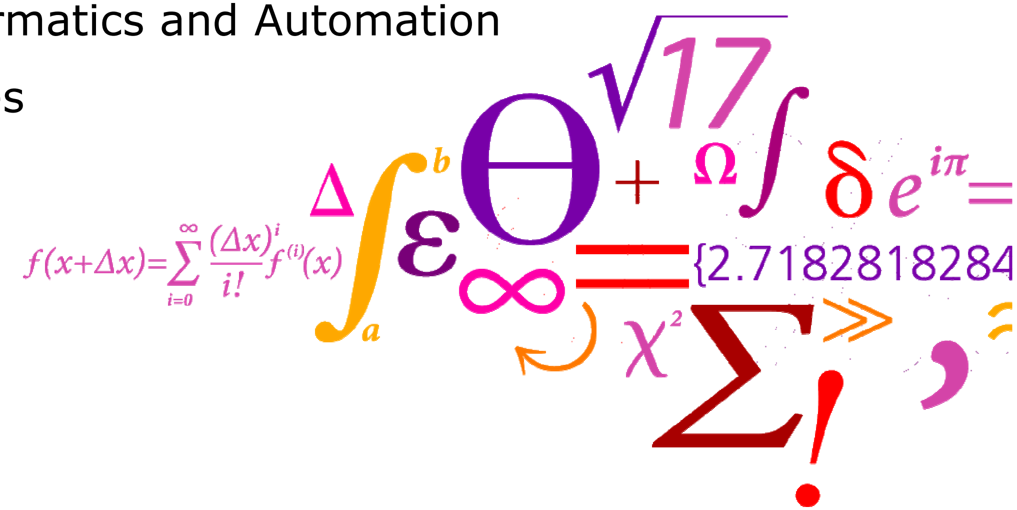
Towards encoder power consumption comparison of Distributed Video Codec and H.264/AVC

Ann Ukhanova[1], Eugeniy Belyaev[2] and Søren Forchhammer[1]

[1]Technical University of Denmark

[2]Saint-Petersburg Institute for Informatics and Automation of the Russian Academy of Sciences



DTU Fotonik
Institut for Fotonik

Outline

- Low-complexity codecs
- H.264/AVC scheme
- DVC scheme
- H.264/AVC and DVC power consumption
- Performance comparison

What is and what for is “low-complexity”?

Use cases

- Wireless sensor networks
- Another systems, where it is necessary to decrease encoding power consumption

Purpose

- Achieve longer working time
- Decoder power consumption for this system is not an important issue

Result

- Power consumption on the encoder side became one of the most important issues along with compression efficiency

The main goal

- Some investigations[1] show that low-complexity DVC outperforms other algorithms in terms of energy consumption if compared at the encoder side
- Our estimations of power consumption show that for current implementations of DVC these statements could be disputable
- It may be more efficient to use compression algorithms based on differential frame coding!

- [1]For example, A. Aaron and B. Girod, "Wyner-Ziv video coding with low-encoder complexity," Proc. Picture Coding Symposium, PCS, San Francisco, CA, December 2004. Invited paper.

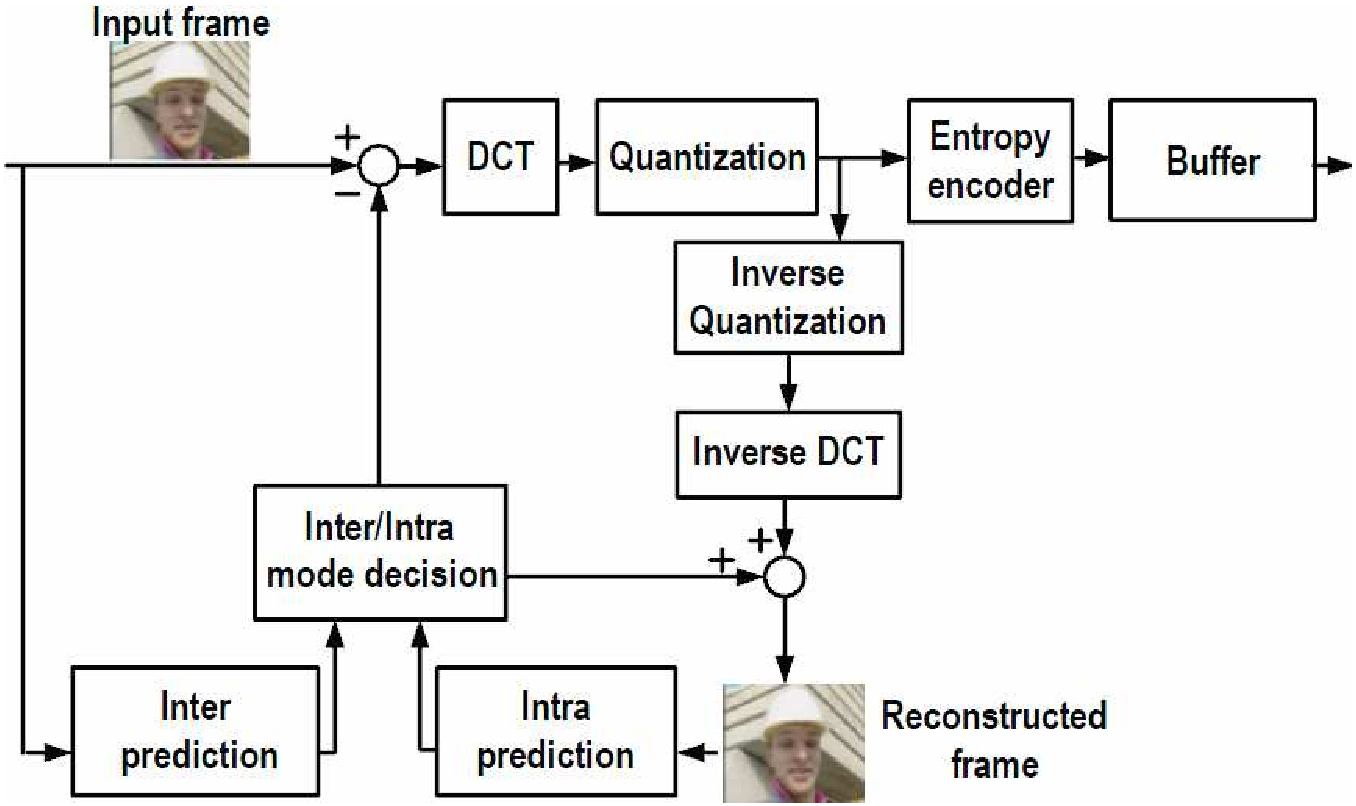
Ideas of comparison methodology

- Processing time measurements
  - *P.L. Dragotti, M. Gastpar **Distributive Source Coding. Theory, Algorithms, and Applications**, Elsevier, 2009.*

- Complexity computation in simple operations
  - *William B. Pennebaker, Joan L. Mitchell **Jpeg: Still Image Data Compression Standard,** 1992*

- Power consumption analysis
  - *Z. He, Y. Liang, L. Chen, I. Ahmad, D. Wu **Power-Rate-Distortion Analysis for Wireless Video Communication Under Energy Constraints**, IEEE Transactions on Circuits and Systems for Video Technology, Vol. 15, No. 5, 2005*

H.264/AVC encoding



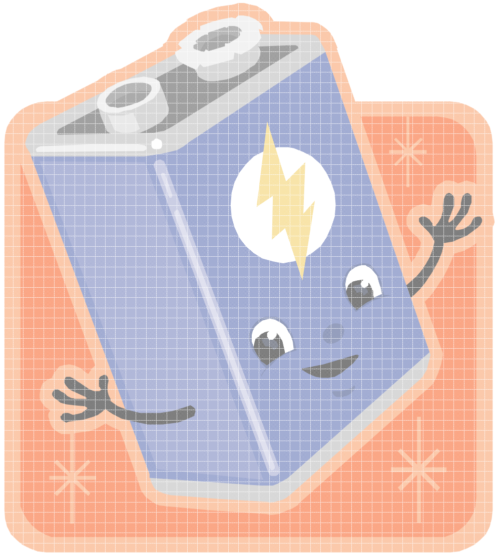
Let's define variables...

Consider that

- F_I and F_B are frame rate for I and B frames accordingly,
- W and H are frame width and height
- R_I and R_B are bit rate of compressed video stream correspondent to I and B frames

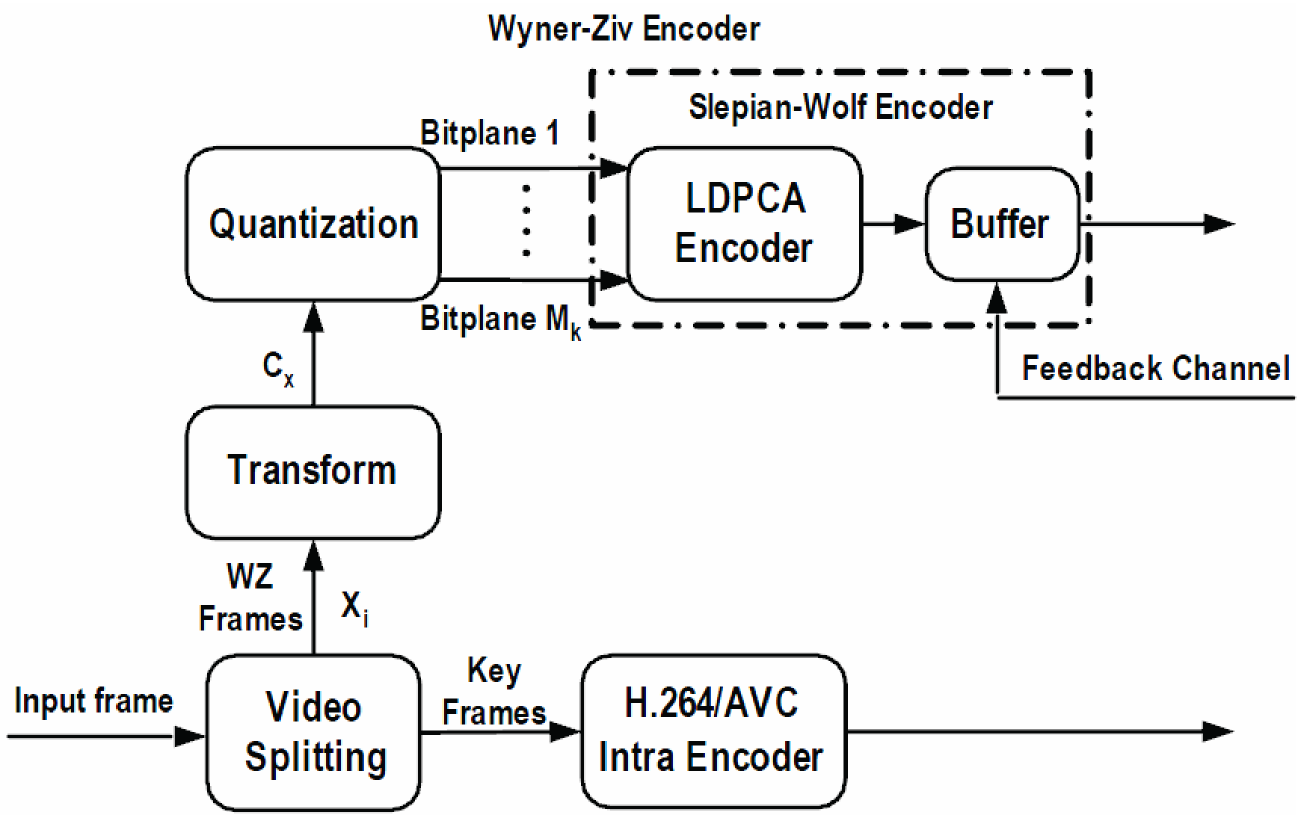
H.264/AVC power consumption[1]

$$\begin{cases} P_{h264} = f_{tran}^{h264}(F_I + F_B, W, H) + f_{CAVLC}(R_I + R_B) \\ f_{CAVLC}(R_I + R_B) = (R_I + R_B) \cdot C_{CAVLC} + P_{CAVLC}^0 \end{cases}$$



[1]F. Huang, S. Lei, "A High Performance and Low Cost Entropy Encoder for H.264 AVC Baseline Entropy Coding", ICCCAS, 2008.

DVC encoding

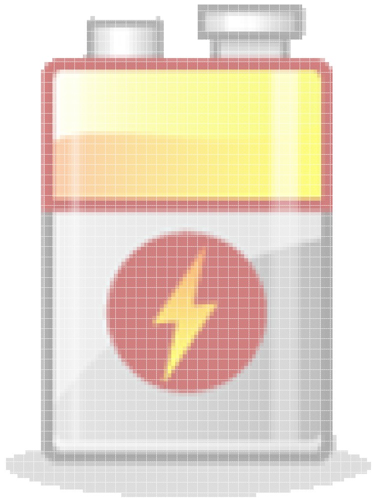


Variables reminder

- F_I and F_B are frame rate for I and B frames accordingly
- F_W – frame rate for Wyner-Ziv frames
- W and H are frame width and height
- R_I and R_B are bit rate of compressed video stream correspondent to I and B frames
- R_W - bit rate of compressed video stream for Wyner-Ziv frames

DVC power consumption[1]

$$\begin{cases} P_{DVC} = f_{tran}^{h264}(F_I, W, H) + f_{CAVLC}(R_I) + f_{tran}^{DVC}(F_W, W, H) + f_{LDPC}(R_W) \\ f_{LDPC}(R_W) = R_W \cdot C_{LDPC} + P_{LDPC}^{0} \end{cases}$$



[1]R. Swamy, S. Bates and T. L. Brandon, "Architectures for ASIC implementations of low-density parity-check convolutional encoders and decoders",ISCAS (5), pp. 4513-4516, 2005.

H.264/AVC and DVC power consumption (1)

$$\begin{cases} P_{h264} = f_{tran}^{h264}(F_I + F_B, W, H) + f_{CAVLC}(R_I + R_B) \\ f_{CAVLC}(R_I + R_B) = (R_I + R_B) \cdot C_{CAVLC} + P_{CAVLC}^0 \end{cases}$$

&

$$\begin{cases} P_{DVC} = f_{tran}^{h264}(F_I, W, H) + f_{CAVLC}(R_I) + f_{tran}^{DVC}(F_W, W, H) + f_{LDPC}(R_W) \\ f_{LDPC}(R_W) = R_W \cdot C_{LDPC} + P_{LDPC}^0 \end{cases}$$

$$f_{tran}^{h264}(F_I + F_B, W, H) \approx f_{tran}^{h264}(F_I, W, H) + f_{tran}^{dvc}(F_W, W, H)$$

H.264/AVC and DVC power consumption (2)

Let us take a typical rate-distortion function

$$d(i) = \sigma^2 \cdot \exp\left(-\frac{r(i)}{\beta(i)}\right), \text{where}$$

$d(i)$ and $r(i)$ are distortion and number of bits for frame i,

σ is the variance of the video data,

$\beta(i)$ is encoder efficiency for given video source complexity

$$\Delta P = P_{h264} - P_{dvc} \approx f_{CAVLC}(R_B) - f_{LDPC}(R_W) \approx R_B \cdot C_{CAVLC} - R_W \cdot C_{LDPC} =$$

$$= \frac{F}{N} \sum_{i=0}^{N-1} (r_B(i) \cdot C_{CAVLC} - r_W(i) \cdot C_{LDPC}) =$$

$$= \frac{F}{N} \sum_{i=0}^{N-1} \ln\left(\frac{\sigma^2}{d(i)}\right) (\beta_B(i) \cdot C_{CAVLC} - \beta_W(i) \cdot C_{LDPC})$$

Practical results

For practical experiments the JM codec v.16.2 [1] was used which is the H.264/AVC reference software and DISCOVER Codec [2] which is a reference software for distributed video coding.

Practical results were obtained for three test video sequences “foreman“, “akiyo“ and “hall monitor“, QCIF (176x144) resolution, 30 Hz.

All power consumption measurements were achieved for 0.18μ TSMC cell library

[1] H.264/AVC JM Reference Software, available on: http://iphome.hhi.de/

[2] DISCOVER codec, available on: http://discoverdvc.org/

Process of obtaining results

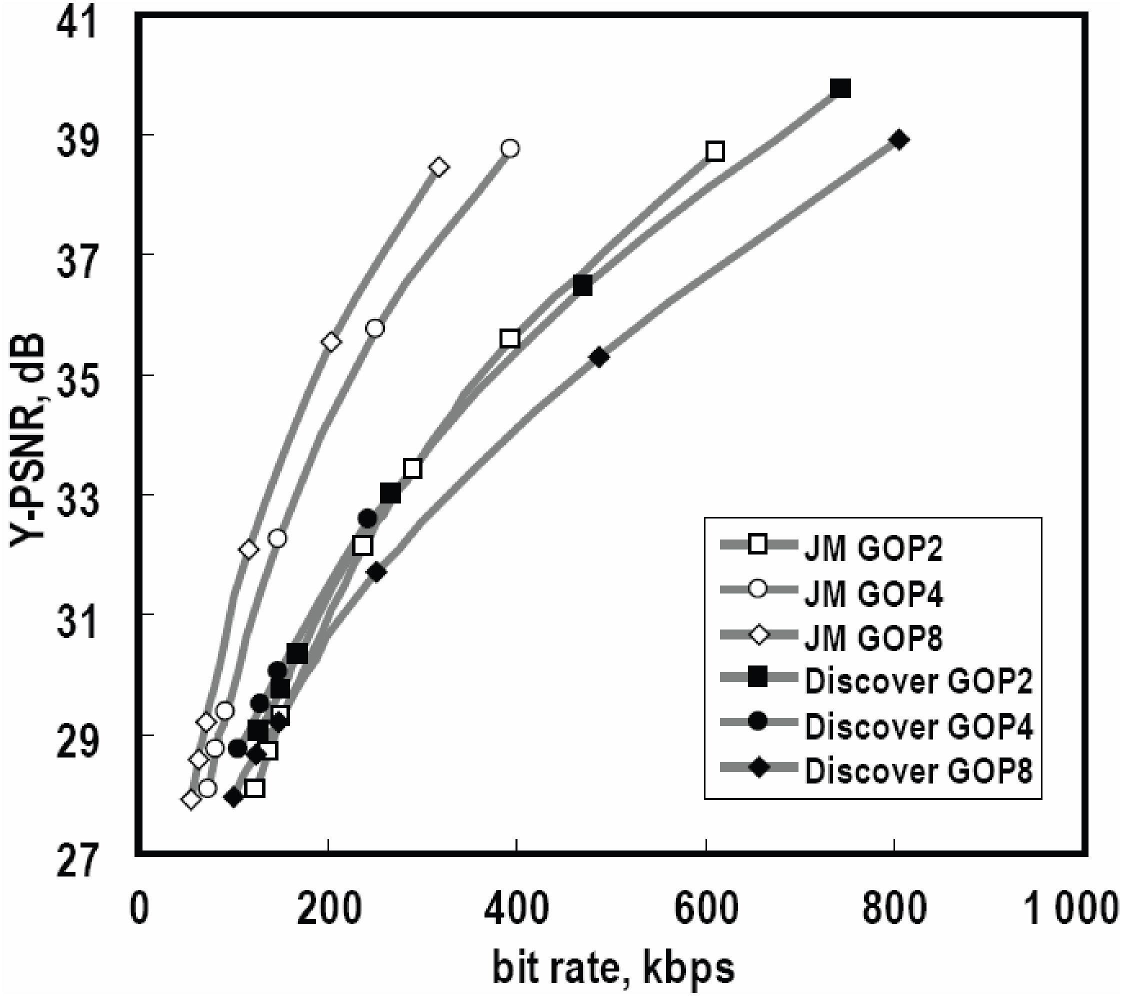
- CAVLC and LDPC power consumption from [1] and [2]
- Compress test video sequences with defined parameters
- Rate-distortion curves
- Graphs PSNR/power consumption

---

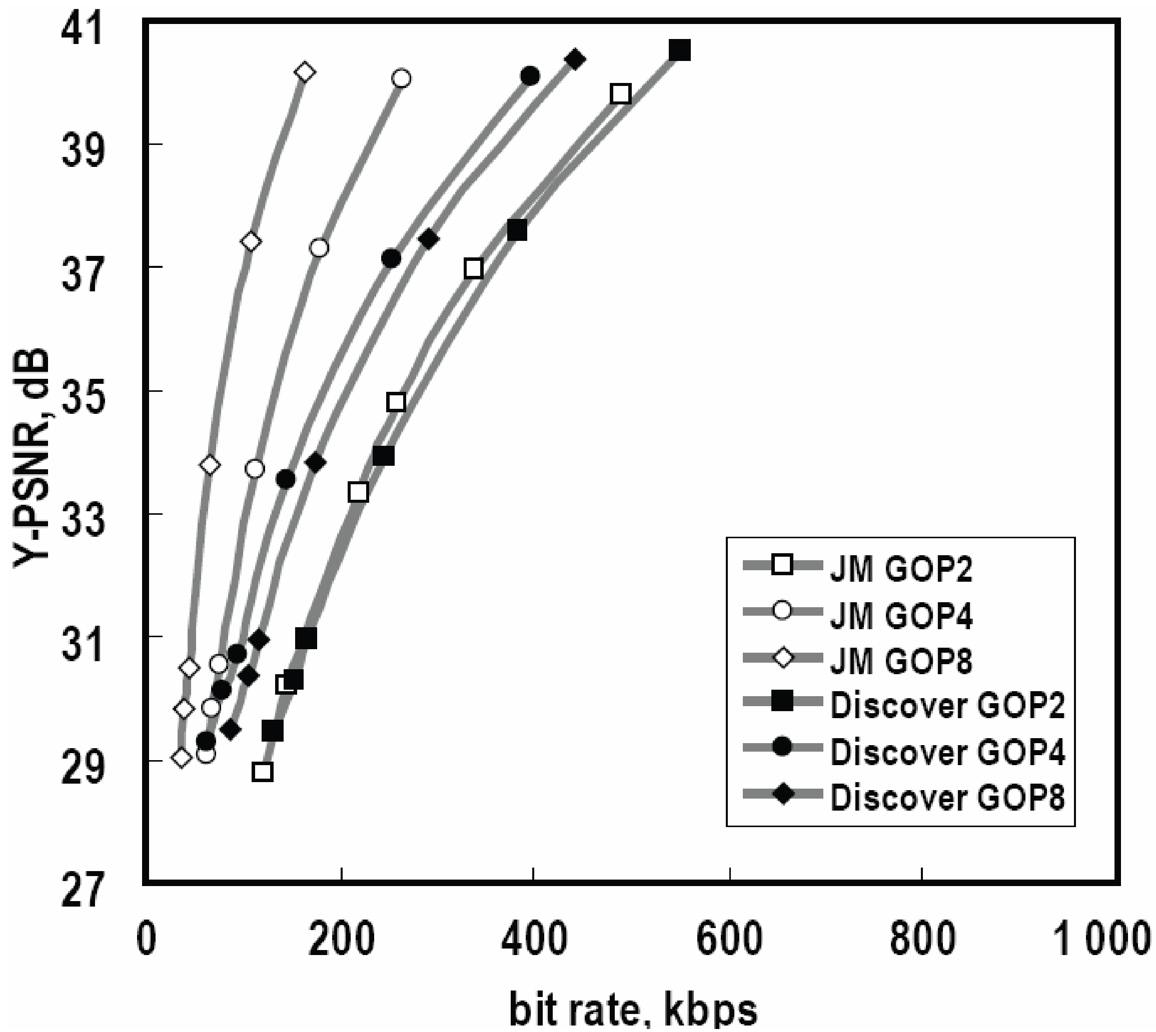
[1]F. Huang, S. Lei, "A High Performance and Low Cost Entropy Encoder for H.264 AVC Baseline Entropy Coding", ICCCAS, 2008.

[2]R. Swamy, S. Bates and T. L. Brandon, "Architectures for ASIC implementations of low-density parity-check convolutional encoders and decoders",ISCAS (5), pp. 4513-4516, 2005.

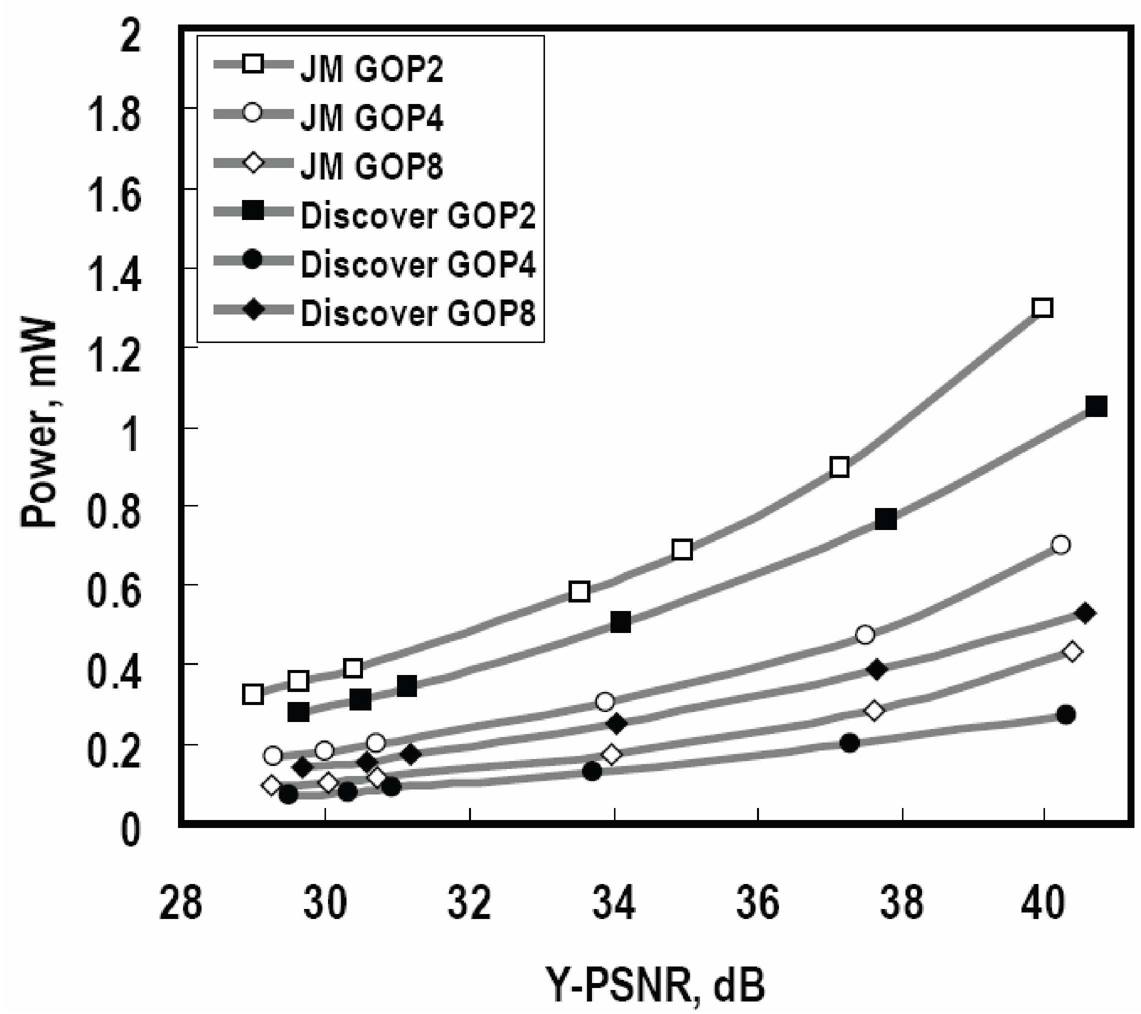
Rate-distortion performance (foreman)



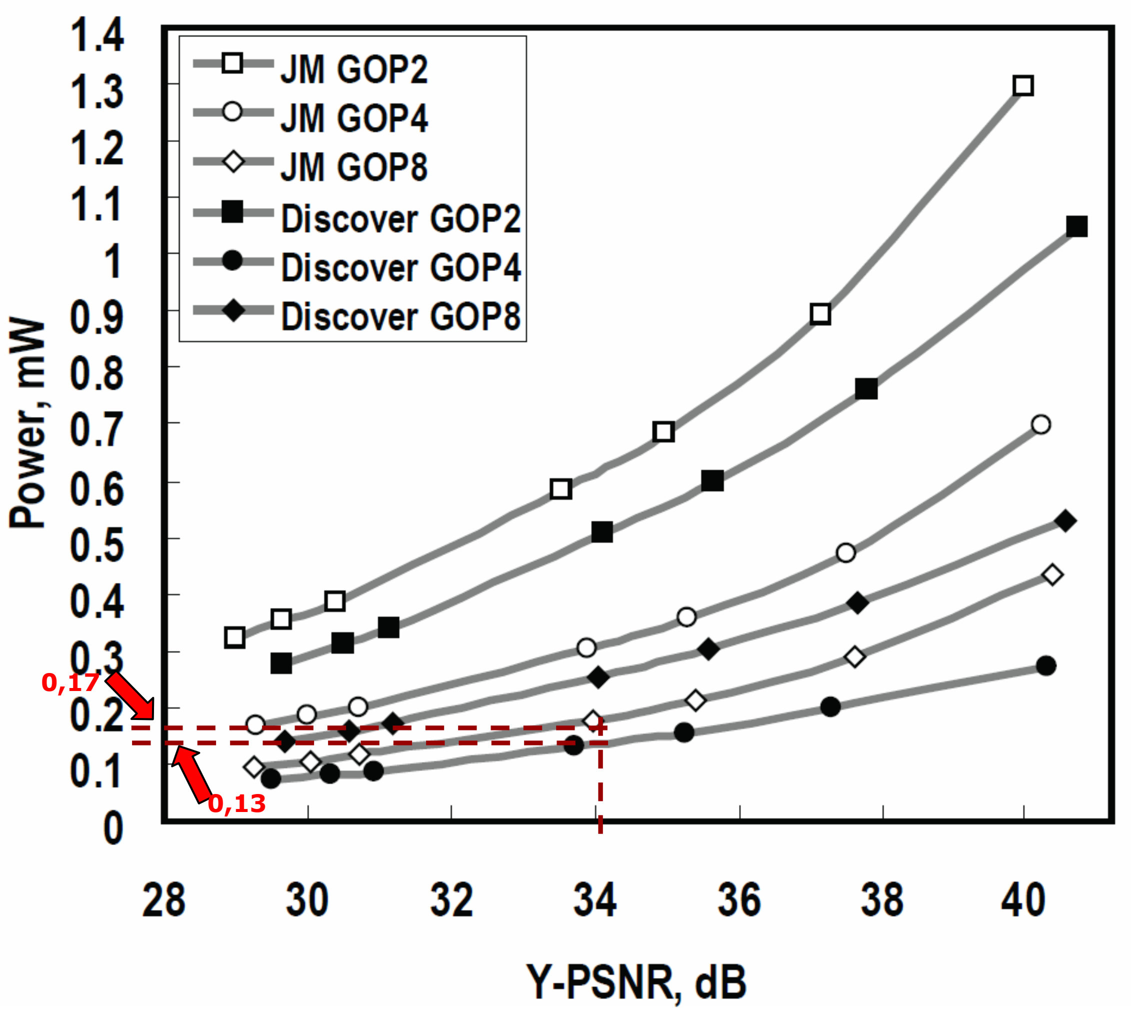
Rate-distortion performance (hall)



Relative power consumption (foreman)



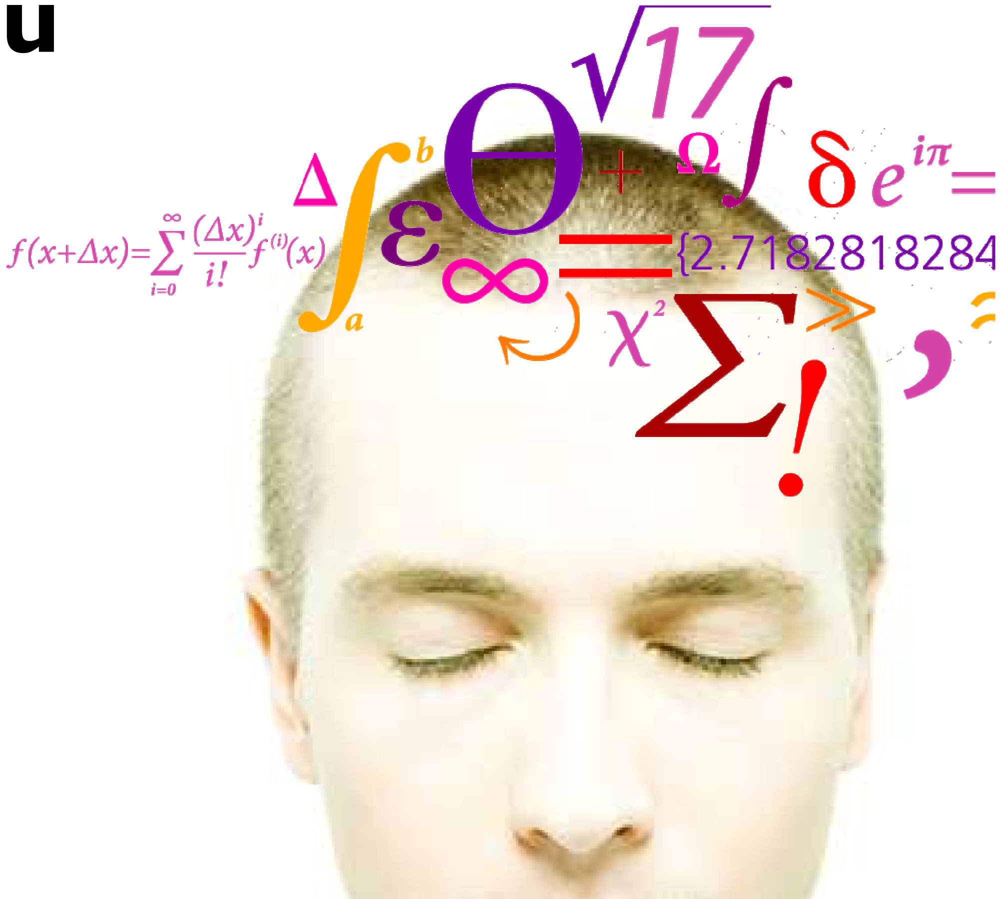
Relative power consumption (hall)



Conclusion

- DVC power consumption gain is estimated around 0.1 mW
- This is not a significant gain, taking into account the following factors
  - DVC actually utilizes 2! codecs, one of which is H.264
  - DVC decoder complexity is rather high!

Thank you



$f(x+\Delta x)=\sum_{i=0}^{\infty}\frac{(\Delta x)^i}{i!}f^{(i)}(x)$
$\sqrt{17}$
$\{2.7182818284$