Introduction to Video Encoding

INF5063

Håvard Espeland

15. October 2010

History of MPEG

- Motion Picture Experts Group
  - MPEG1 work started in 1988, published by ISO in 1993
    - Part 1 Systems, Part 2 Video, Part 3 Audio, Part 4 Compliance Testing and Part 5 Software Simulation
    - MP2 and MP3 Audio
  - MPEG2 was published in 1996 together with ITU
    - 11 parts in total
    - Part 2 Video also known as H.262, used on DVD
    - Part 3 Audio (improved MP3) and Part 7 Audio (AAC)

History of MPEG

- MPEG4 was introduced in 1996
  - 27 parts in total, all known as MPEG4
  - Part 2 Visual (H.263): Simple Profile, Advanced Simple Profile (ASP)
    - Colloquially called MPEG4 (until recently)
    - Widely used in broadcasting, teleconferencing
    - Compresses much better than MPEG2
    - Often referred to by the name of the encoder (DivX, Xvid)
  - Part 3 Audio: AAC+, CELP, and more.

MPEG4

- Part 10 (H.264): Advanced Video Coding (AVC)
  - Introduced in 2003, and is still the most efficient (standardised) codec available
  - Up to about twice the compression of MPEG4 ASP
  - Used by Bluray, Rikstv, Youtube, and many others
  - Also referred to by the name of a specific encoder (x264)
  - 17 profiles, including Multiview High Profile (MVC) for stereo (3D) video, and Scalable High Profile (SVC) for adaptive streaming and trick-play functionality.
  - Covered by roughly 1500 patents worldwide

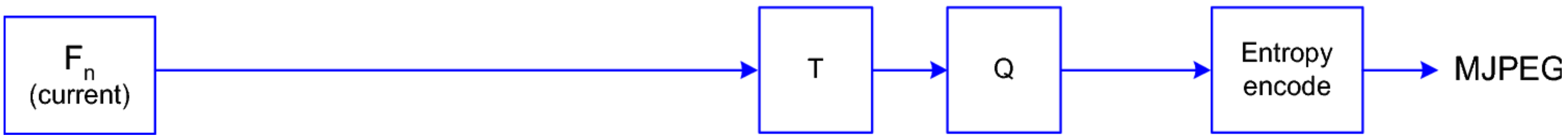
Along came Google...

- Bought On2 in February 2010
  - Released the VP8 codec specification with a royalty free license in May 2010 together with an open-source implementation (libvpx)
  - Also released a container format (webm) based on Matroska (mkv) to distribute VP8 together with Vorbis.
- VP8 is very similar to H.264, but only supports a small subset of the features
- Supported by all major browser vendors, VP8 is expected to dominate web video in the coming years.

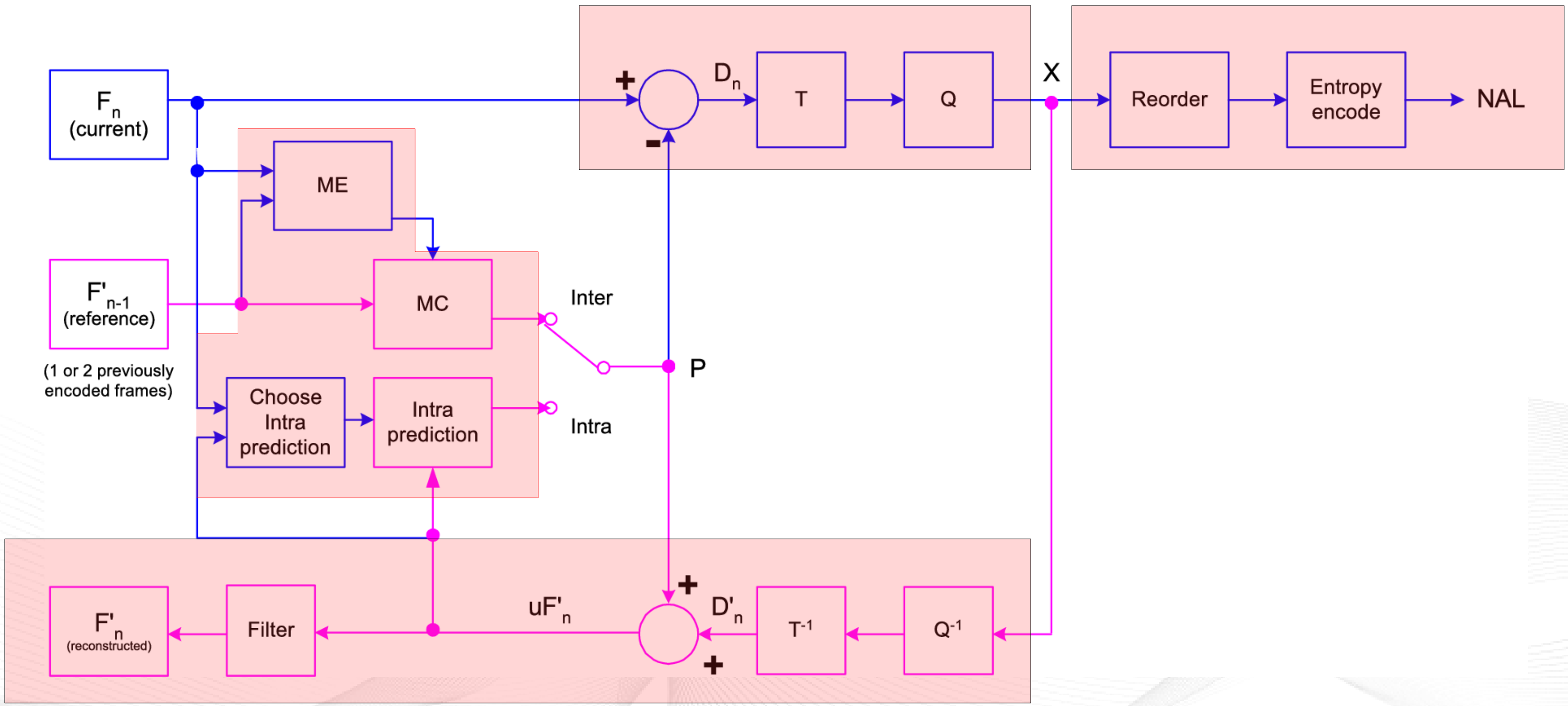
Video standards

- The latest version of the H.264 specification has more than 600 pages, costs 294 CHF and can be bought from ISO/ITU.
- The VP8 standard has only 104 pages and is available for free from http://webmproject.org
- Video standards only describe *decoding* of the bitstream. The black art of *encoding* video is left as an excercise to implementers.

MJPEG Encoder Overview

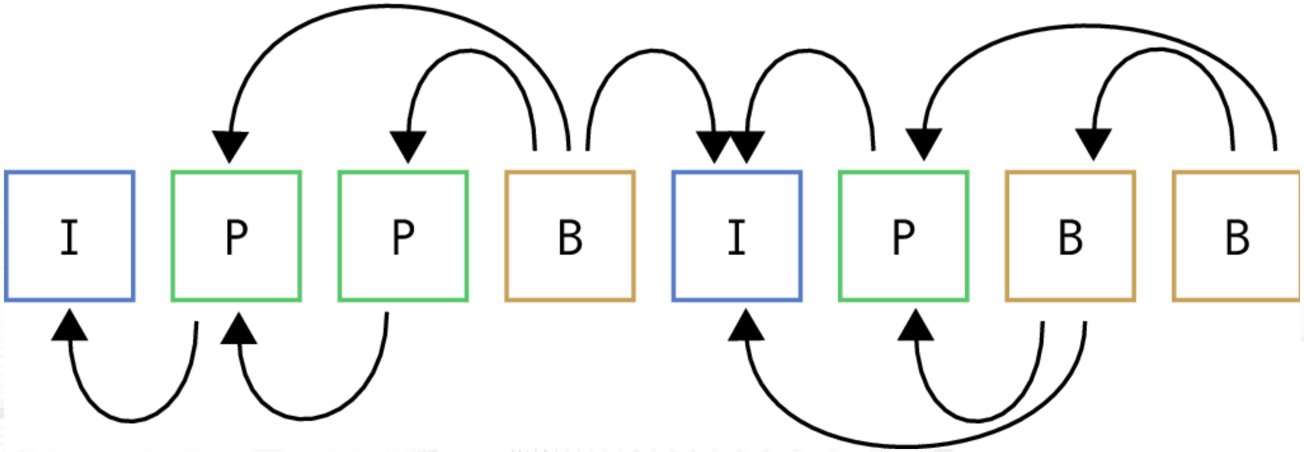


H.264 / VP8 Encoder Overview



Macroblock Types

- Macroblocks are 16x16 pixels, but can be subdivided down to 4x4 pixels. In H.264 they are of type I, P or B
  - Intra-MBs use Intra-prediction
  - Predicted MBs use Inter-prediction
  - Bi-Directional Predicted MBs use prediction both backward and forward in time



Frame types

- Traditional frame types are I-, P- and B-frames.
  - Intra-predicted frames can only use I-macroblocks.
  - Predicted frames can only use I- and P-macroblocks.
  - Bi-Directional predicted frames can use I-, P- and B-macroblocks

- VP8 does not have the concept of B-frames, but instead provides Alt-ref and Golden frames
  - Alt-ref frames are never showed to the user, and are only used for prediction

Inter-Prediction

- Predict a macroblock by reusing pixels from another frame
  - Objects tend to move around in a video, and *motion vectors* are used to compensate for this
  - H.264 allows up to 16 reference frames, while VP8 only supports 3 frames

Determining Prediction Modes

- Motion estimator tries as many modes and parameters as possible given a set of restrictions
  - The restrictions can be frame type, encoding time, heuristics, and possibly many more
- The different modes are evaluated with a cost function
- The estimator often use a two-step process, with initial coarse evaluation and refinements
- Refinements include trying every block in the area, and also using sub-pixel precision (interpolation)
- Many algorithms exist designed for different restrictions

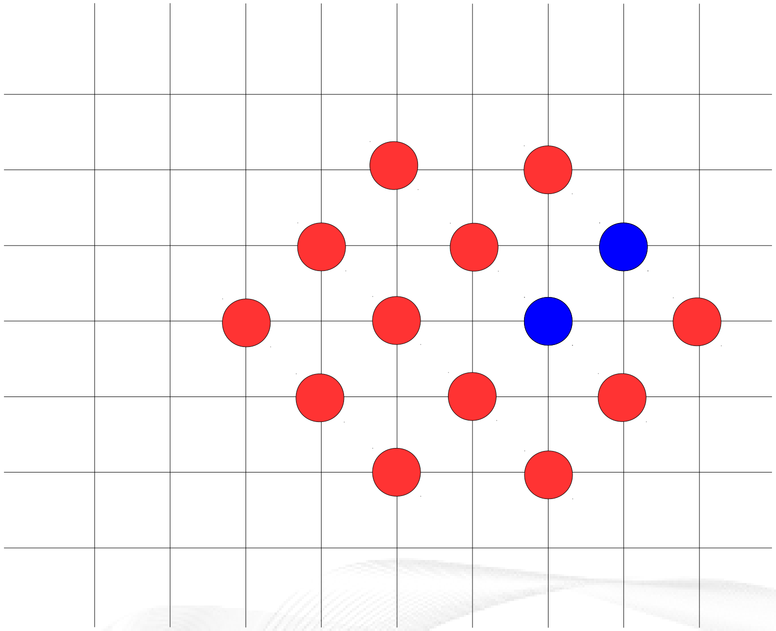
Cost Functions

- Typically Sum of Absolute Differences (SAD) or Sum of Absolute Transformed Differences (SATD)

- SATD tranforms the sum with a Hadamard transformation
  - More accurate than SAD

$$SAD = \sum_{i=0}^{M-1} \sum_{j=0}^{N-1} |I_{i,j} - T_{i,j}|$$

Diamond Motion Estimation Pattern



Motion Compensation

- When the best motion vector has been found and refined, a predicted image is generated using the motion vectors

- The reference frame can not be used directly as input to the motion compensator
  - The decoder never sees the original image. Instead, it sees a *reconstructed* image, i.e. an image that has been quantized (with loss)

- A reconstructed reference image must be used as input to motion compensation

Intra-Prediction

- Predict the pixels of a macroblock using information available within a single frame
- Typically predicts from left, top and top-left macroblock by inter- or extrapolating the border pixel's values
- Many prediction modes available, e.g. horizontal, vertical, average

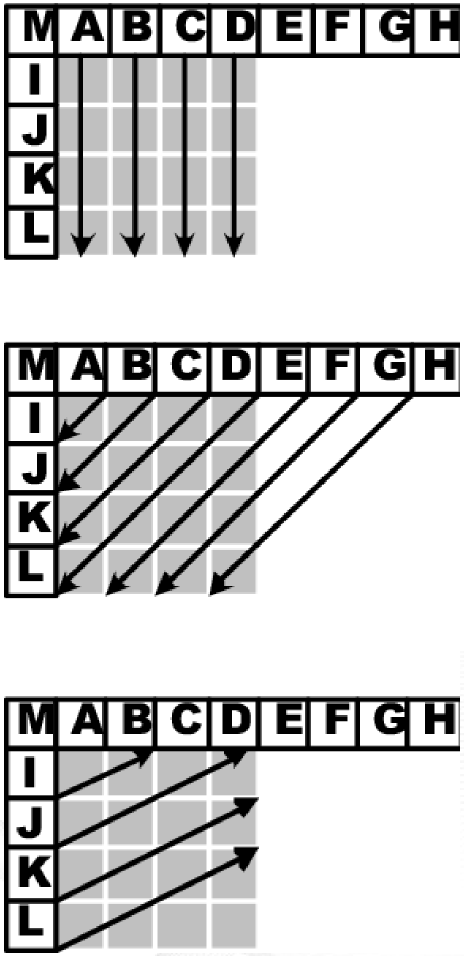
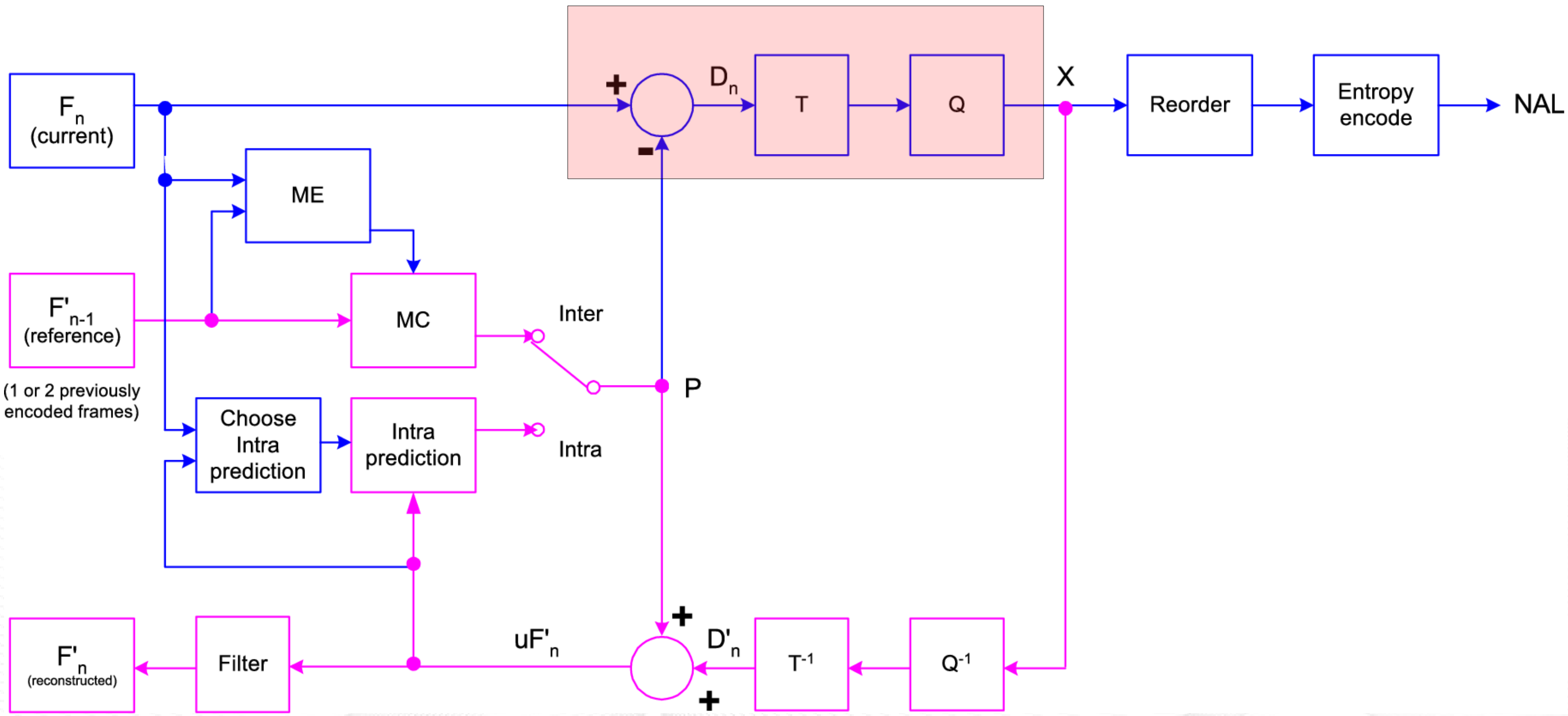


Figure from vcodex.com

H.264 / VP8 Encoder Overview



Residual Transformation

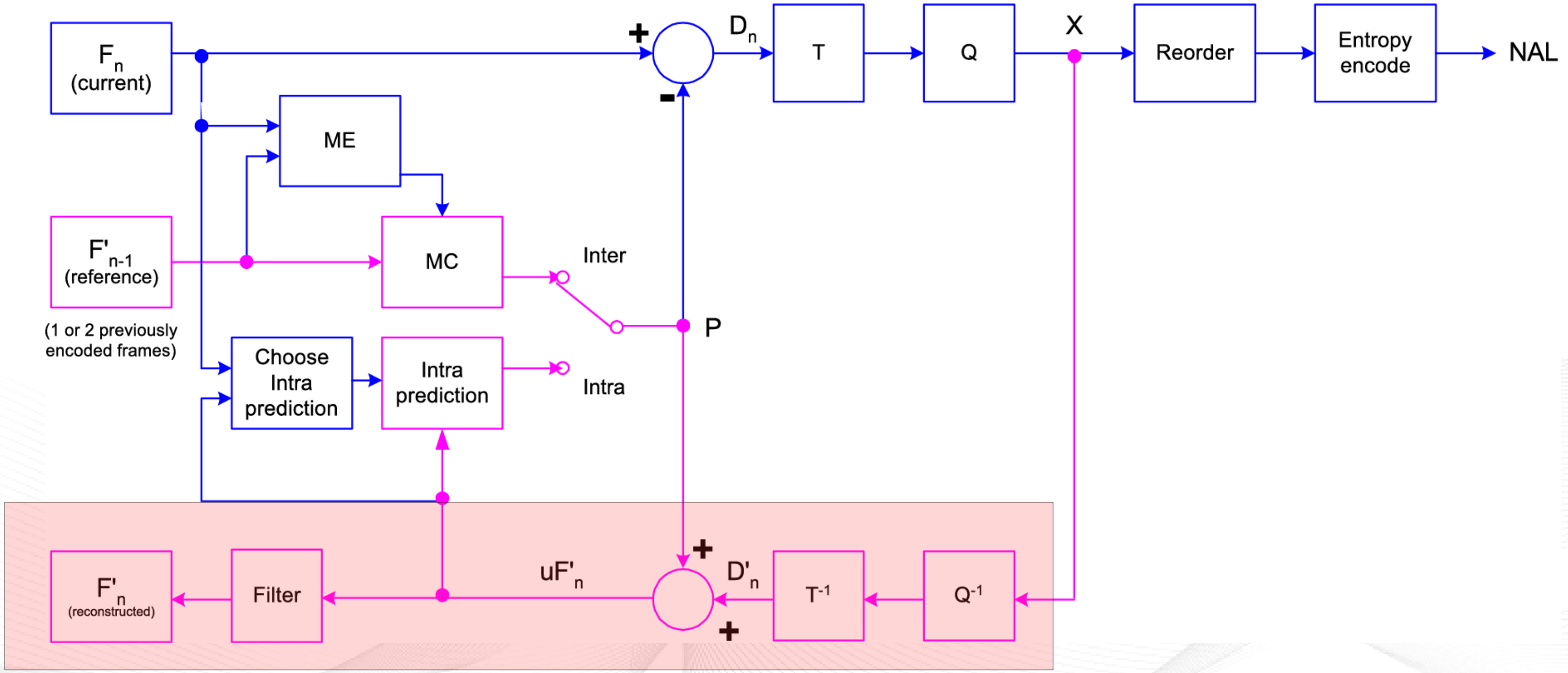
- The pixel difference between the original frame and the prediction is called residuals
- Since the residuals only express the difference from the prediction, they are much more compact than full pixel values such as in JPEG.
- Residuals are transformed using DCT (H.264) or a combination of DCT and Hadamard (VP8)
- Other alternatives exist such as Wavelets (Dirac)
- VP8 and H.264 use 4x4 DCT functions to reduce computation complexity

Quantization and Rate Control

- Divide the transformed residuals (coefficients) with a quantization matrix
  - Decimates the precision of the pixel frequencies, i.e. the lossy part of a video codec
  - The matrix is often scaled with a quantization parameter (qp) to a desired level of reduction

- Each frame is given a bit budget and qp is adjusted to match the budget

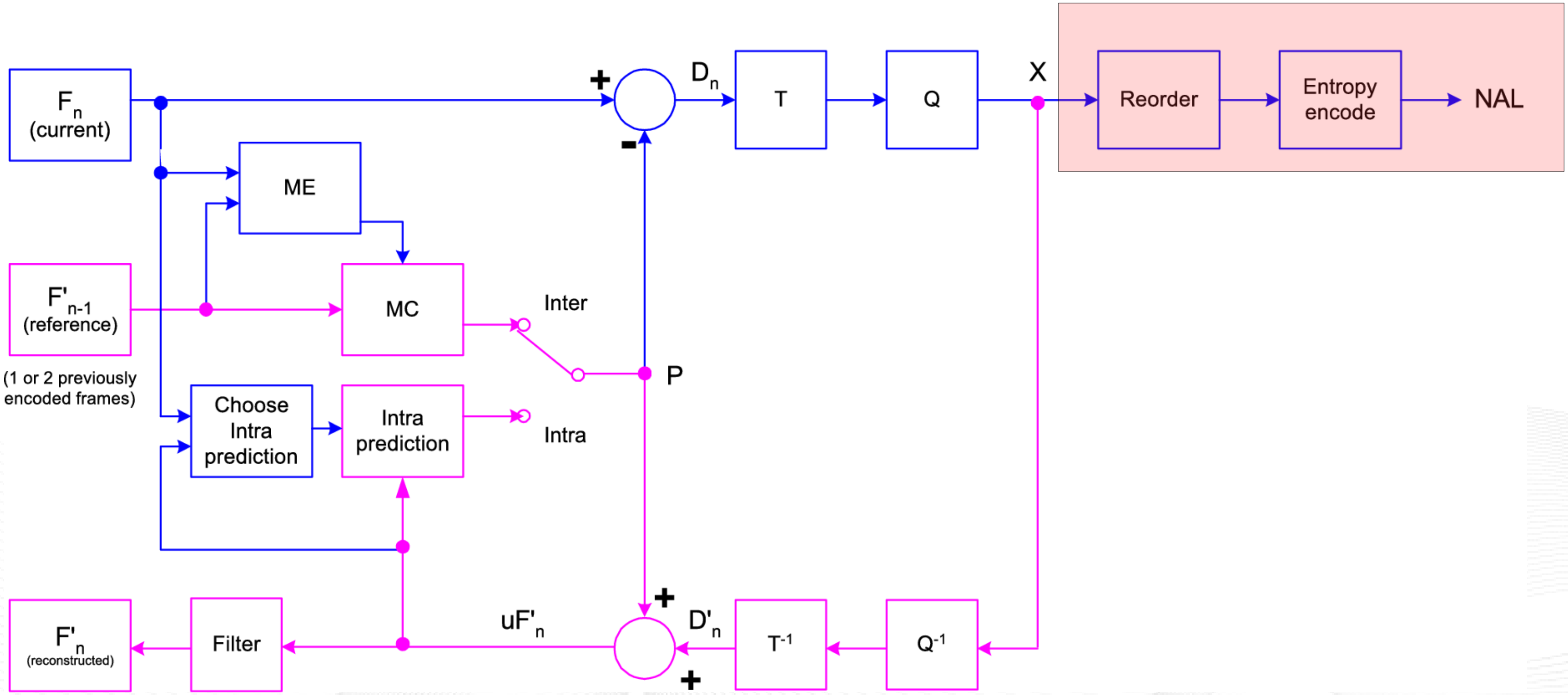
H.264 / VP8 Encoder Overview



Frame Reconstruction

- The motion compensator requires as input the same reference frame as the *decoder* will see
- De-quantize and inverse transform the residuals and add them to our predicted frame
  - The result is the same *reconstructed* frame as the decoder will receive
- To increase visual quality, H.264 and VP8 use a deblocking filter to remove hard edges from the macroblocks

H.264 / VP8 Encoder Overview



Frame Reordering

- In H.264, B-frames may reference *future* frames
- The frames are reordered in such a way that frames used for predicting other frames are decoded first
- In practice, this means that decoding order and display order of frames may differ
- VP8 does not have this concept, but can instead use alt-ref frames for prediction

Entropy Coding

- The motion vectors, intra-predictors, encoder parameters and residuals must be stored somehow
- The entropy coding process is lossless, and removes redundancy from the output bitstream
- Many alternatives exist
  - Variable Length Coding with RLE (MPEG4 ASP)
  - Arithmetic Coding (H.264 and VP8)
  - Exp-Golomb Coding (VC-1)
- The symbol probabilities change over time and are continously updated by the encoder
- The symbol probabilities may depend on the context

Variable Length Coding - Huffman

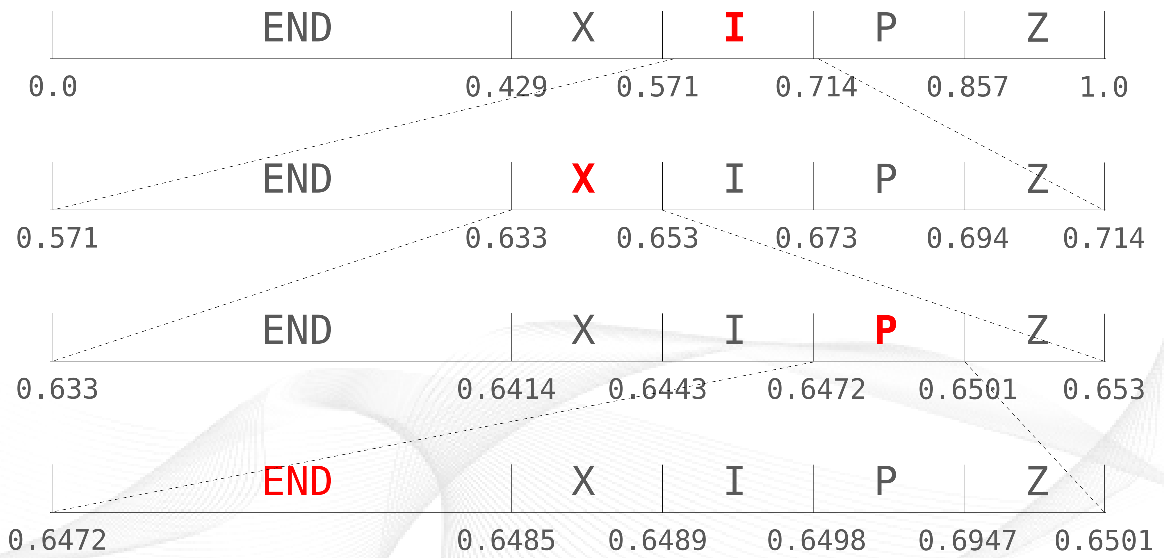
- Since the literals we want to store varies in length, VLC preprends the literal with a symbol that represents the *length* of the literal instead of storing the maximum length bits

```
To store an uint8_t v1 = 7;

bit_width(7) = 3

Output: 100 111 (6 bits)

Store uint8_t v2 = 0;

bit_width(0) = 0

Output: 00
```

Prefix Code	Bit Length
00	0
010	1
011	2
100	3
101	4
110	5
1110	6
11110	7

Arithmetic Coding

- Encodes the entire message into a single number between 0.0 and 1.0.
- Allows a symbol to be stored in less than 1 bit (!)



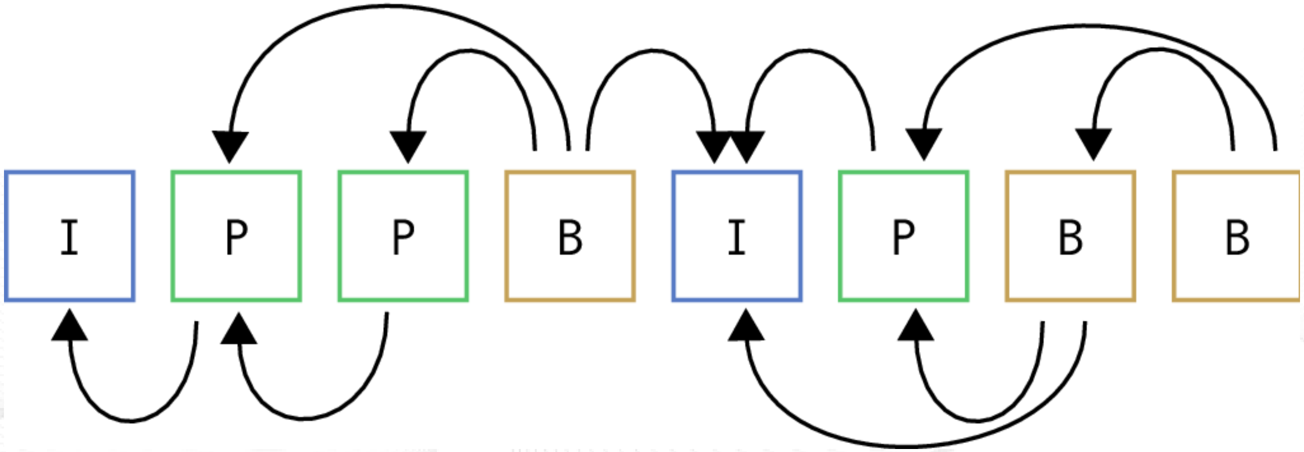
Encode IXP as a number between 0.6472 and 0.6485, e.g 0.648 = 1010001000

Parallel Encoding

- Many approaches available both for Intra and Inter prediction
- Some of these give up compression efficiency for increased parallelity.
- Pipeline-approaches do not combine well with realtime-encoders

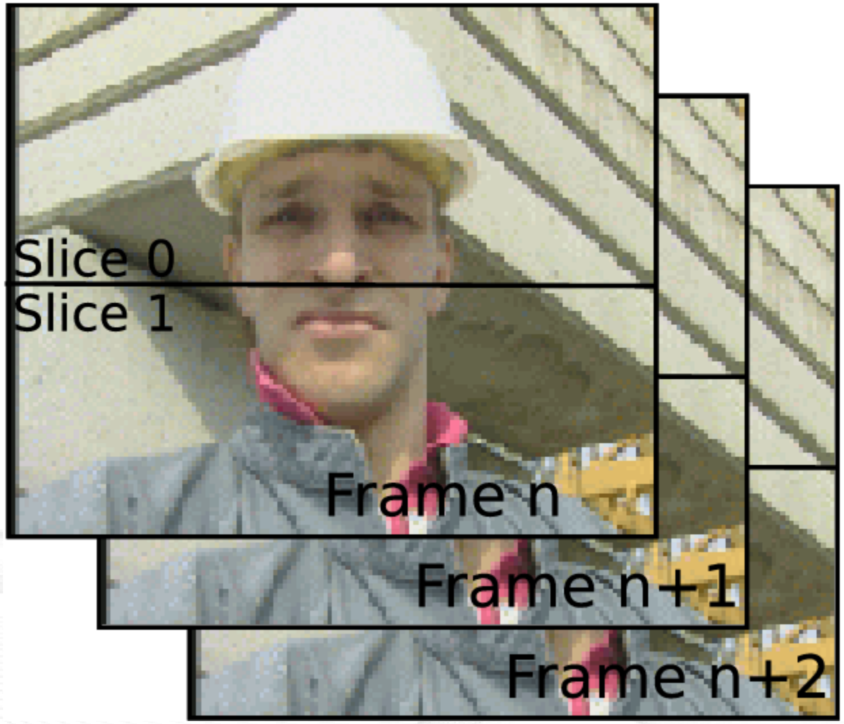
GOP-based Approach

- Delay pictures until there are multiple keyframes within the encoder pipeline
- Intra-frames do not depend on previous frames, and thus can be used as starting points for multiple encodes
- Adds a considerable pipeline to the encoder



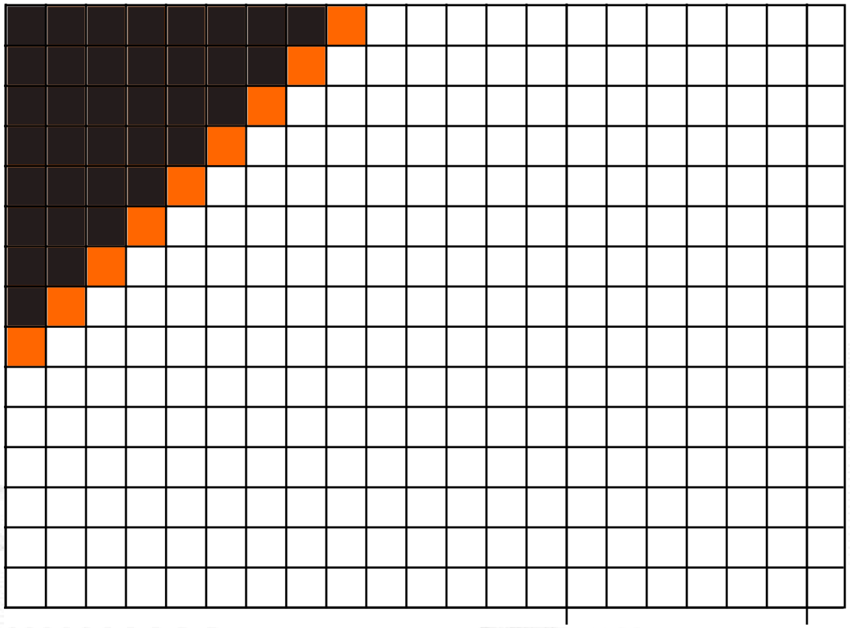
Slice-Based Approach

- Split every frame in one or more slices
- Supported by H.264
- Slices are completely independent – one may not predict across borders
- Severly hurts compression efficiency
- Typically used by the Apple H.264 compressor



Triangle Intra-Prediciton

- MBs depend on the left, top and top-left MB relative to its own position
- As soon as those dependencies are satisfied, the encoder can process in parallel without sacrificing encoder efficiency



Original-Frame Approach

- Both Intra- and Inter-prediction can use the original frame instead of the *reconstructed* frame when evaluating prediction modes

- However, when doing the actual motion compensation and intra-prediction, the encoder *must* use the reconstructed frame – if not there will be a drift between the encoder and decoder

- This approach reduces the quality of predictions

MV-Search Within a Frame

- When the reference frame has been fully reconstructed, the Motion Estimator tries for every macroblock to compensate for motion
- Since every motion vector is independent in the estimator, all MVs in a single frame can be search for in parallel
- When using optimized MV search patterns that takes advantage of the nearby block's MVs, this must be done in a manner similar to triangle intra-prediction
- Does not reduce encoding efficiency

MV-Search Across Frame Boundaries

- Reconstruct part of the frame as soon as the ME is finished with a macroblock
- When a large enough area has been reconstructed of the reference frame, the next frame's ME can start searching MVs that can only be found in this area
- The same technique can be used on multiple frames simultanously
- Requires synchronization on macroblock level
- Advanced technique with good performance
  - Does not reduce encoding efficiency
  - Adds a pipeline to the encoder – but it does not have to be deep. A few frame's delay can be acceptable for realtime encoding

Video Quality Assessment

- Evaluating video quality is a research field by itself
- Usually requires a panel of subjects that rate which version is *best*
  - Subjects have different preferences for artifacts and quality reductions
- Objective measurements give a rough number which says something about the difference between the original frames and the reconstructed frames
  - Typically SSIM or PSNR
  - A shell script for finding PSNR values of YUV frames can be found in the mplayer source tree under TOOLS/

Conclusion

- Video encoding is mainly about trying (and failing) different prediction modes limited by user-defined restrictions (resource usage)

- The “actual” encoding of the video when the parameters are known usually accounts for a small percentage of the running time

- Any (reasonable) codec can produce the desired video quality – what differs between them is the size of the output bitstream they produce