Advanced Multimedia Signal Processing
(#9: H.264/AVC Deblocking & Fast Scheme)

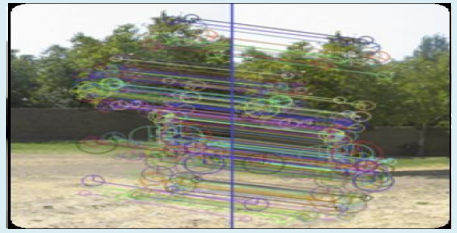
2017. Spring

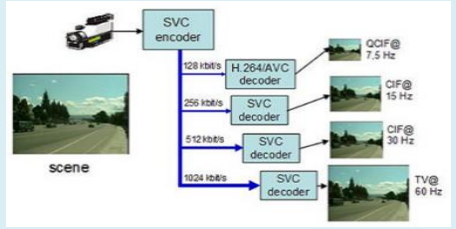
Prof. Byung-Gyu Kim

Visual Information Computing Lab. (VICL)

http://vicl.sookmyung.ac.kr

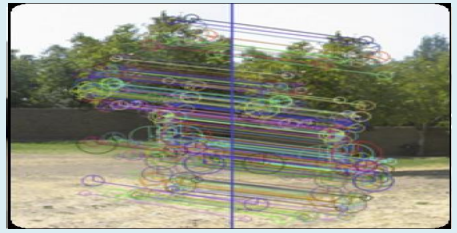
Dept. of IT Engineering, Sookmyung Women's University

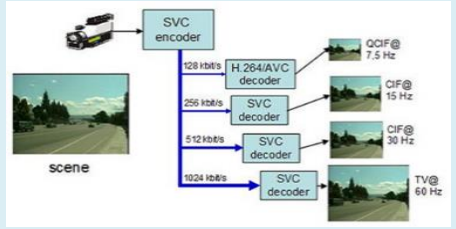




Contents

- Blocking Problem of Block-based Codec
- Filtering Techniques of H.264/AVC Video Codec
  - In-Loop Deblocking Filter
- Fast Schemes for H.264/AVC Encoder
- Summary
- Homework
  - Derivation of Motion Vector Distribution



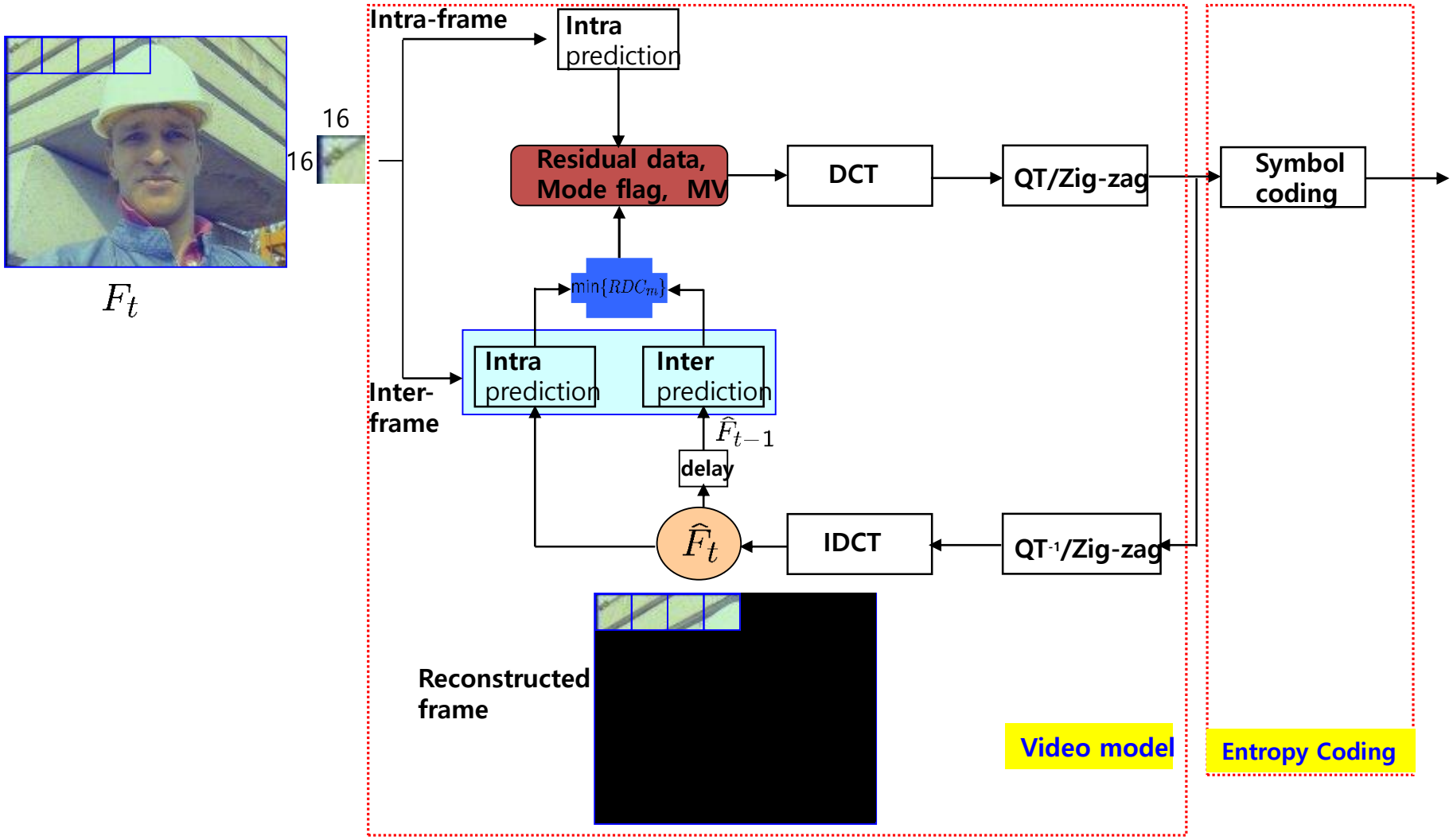


Contents

- Blocking Problem of Block-based Codec
- Filtering Techniques of H.264/AVC Video Codec
  - In-Loop Deblocking Filter
- Fast Schemes for H.264/AVC Encoder
- Summary
- Homework
  - Derivation of Motion Vector Distribution

Blocking Problem of Block-based Codec (1)

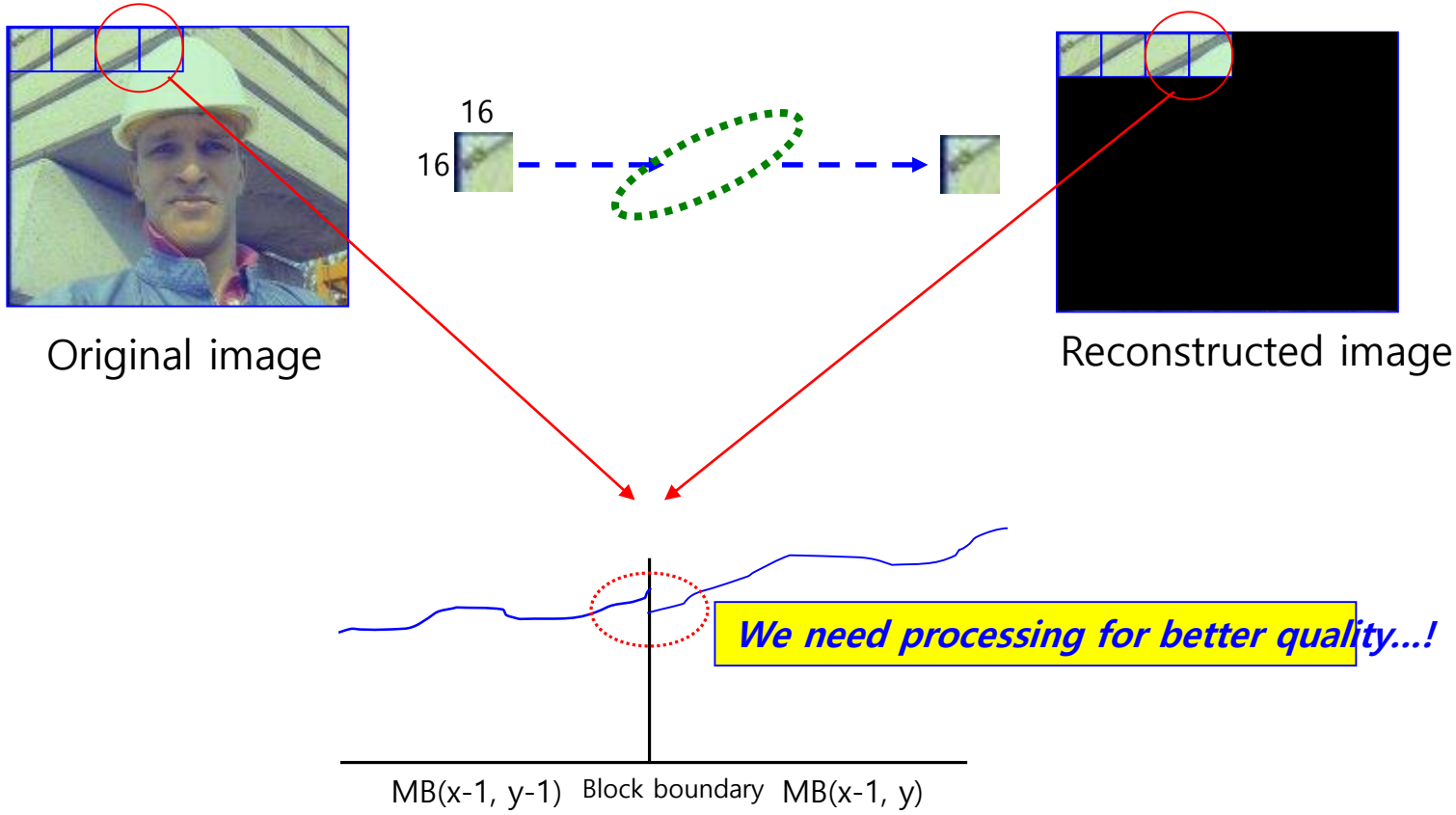
❖ Hybrid Video Coding Scheme



Blocking Problem of Block-based Codec(2)

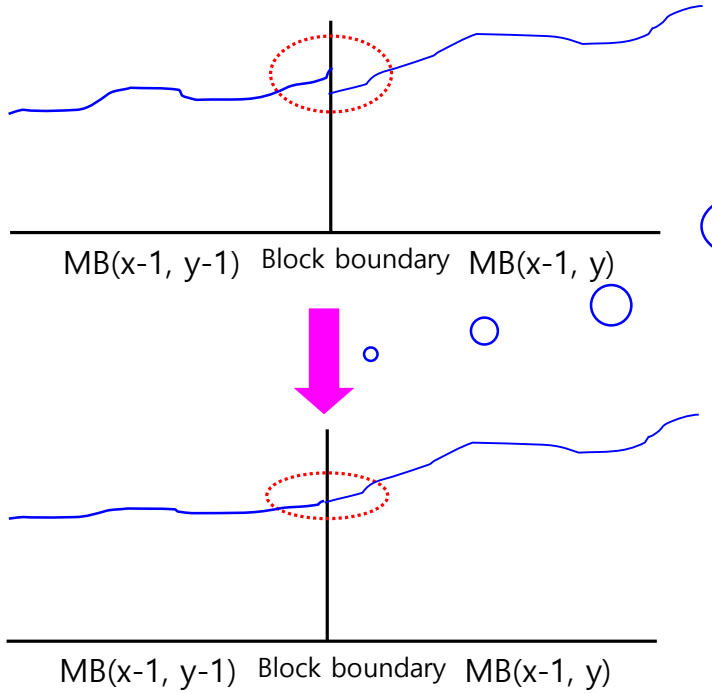
❖ Block Unit (Based) Processing

- What is problem of block unit processing in terms of image quality?



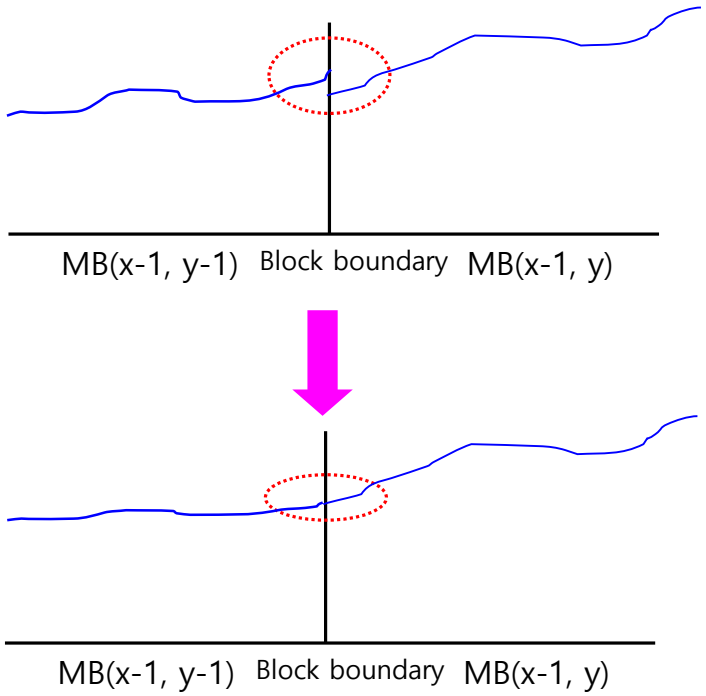
Blocking Problem of Block-based Codec(3)

- ***Which processing*** do we need..?
  - The problem is a discontinuity between the processed MBs. → ***Continuous boundary***.



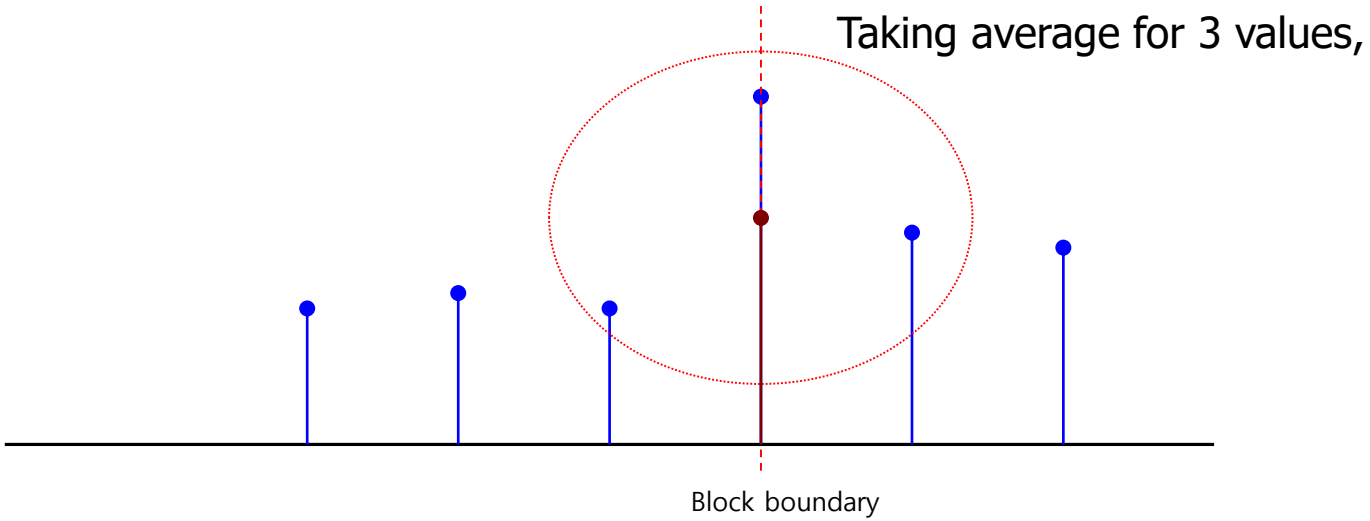
Blocking Problem of Block-based Codec (4)

- Smoothing operation
  - A kind of averaging computation on the boundary.
  - Processing to reduce the variation of the given data.



Blocking Problem of Block-based Codec(5)

- Blocking Artifact (Effect)
  - A discontinuity between the processed MBs or 4x4 blocks.
- Deblocking filter
  - Tool for smoothing operation on this discontinuity between the processed MBs or 4x4 blocks

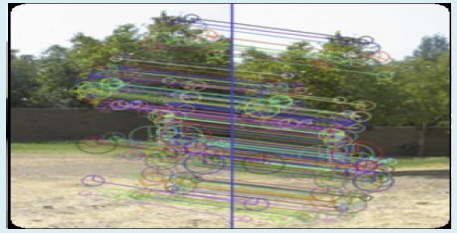


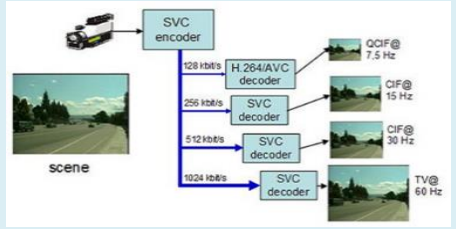
Blocking Problem of Block-based Codec(6)

- Blocking Artifact (Effect) (ex1)

Blocking Problem of Block-based Codec(7)

- Blocking Artifact (Effect) (ex2)



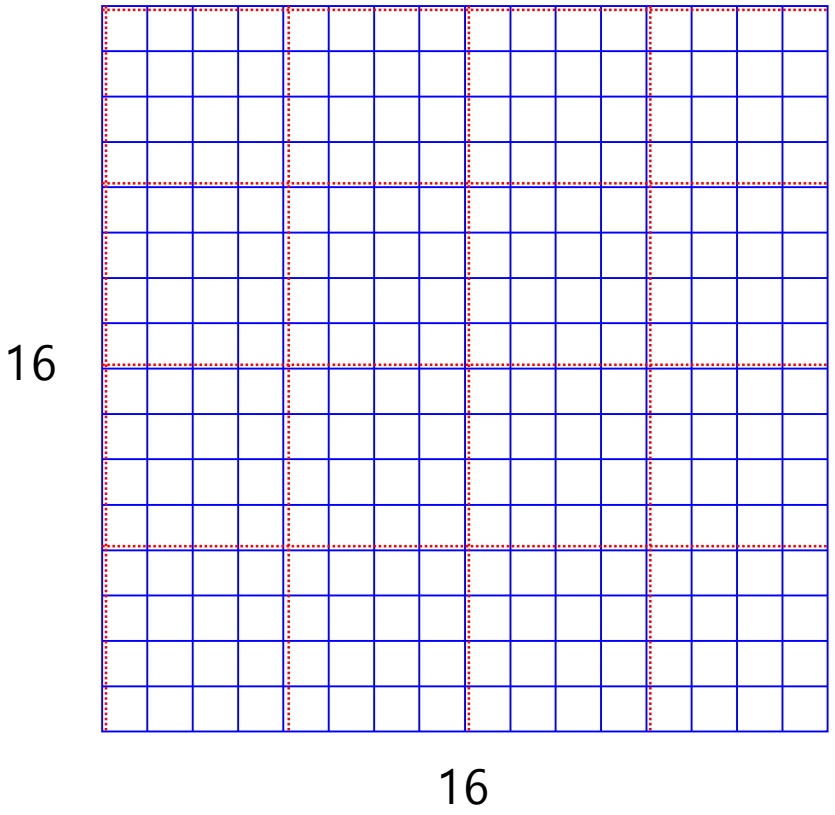


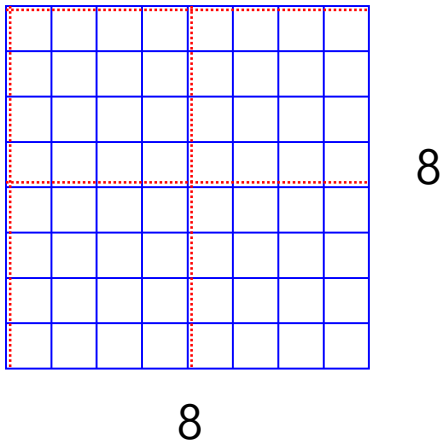
Contents

- Blocking Problem of Block-based Codec
- Filtering Techniques of H.264/AVC Video Codec
  - In-Loop Deblocking Filter
- Fast Schemes for H.264/AVC Encoder
- Summary
- Homework
  - Derivation of Motion Vector Distribution

❖ **Deblocking** Filter (jm 12.2)

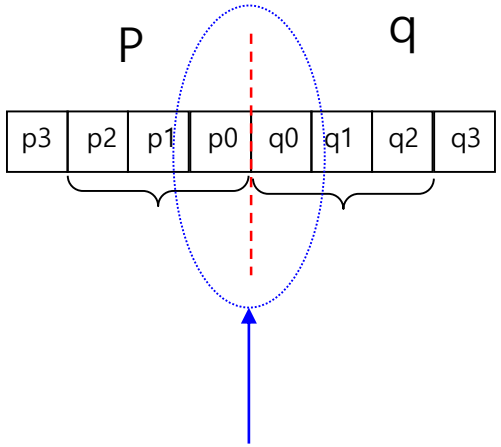
- Applied to vertical or horizontal edges of 4x4 blocks (or 8x8 blocks) in a MB.





Filtering Techniques of H.264/AVC Video Codec (2)

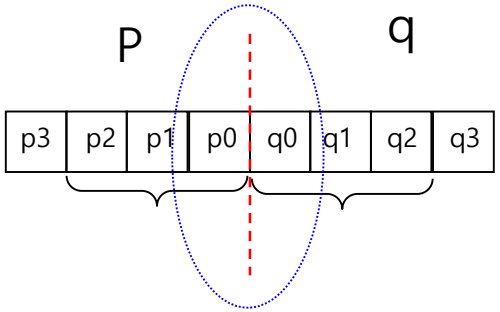
- **Target pixels for deblock filtering**



- **Some parameters for filtering**
  - Boundary Strength (bS)[0,~,4]
  - Strength of filter depends on
    - **•Quantizer.**
    - **•Gradient of image.**
    - **•Coding modes of neighboring blocks.**
    - BS=4: block p and/or q is intra and boundary is MB's boundary
    - BS=3: block p and q are intra and boundary is not MB's boundary.

Filtering Techniques of H.264/AVC Video Codec (3)

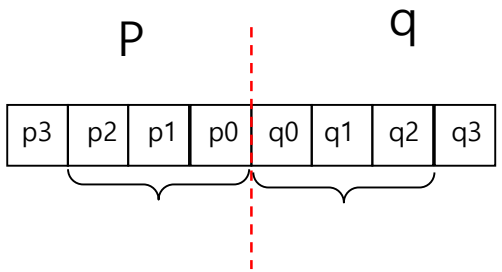
- **Target pixels for deblock filtering**



  - BS=2: neither block p nor q is intra and p and q contain coded coeffs..
  - BS=1: neither block p nor q is intra and p and q contain no coded coeffs..; p and q use different reference picture or a different number of reference pictures.
  - BS=0: Otherwise, no filtering.

Filtering Techniques of H.264/AVC Video Codec (4)

- **Filter Decision**



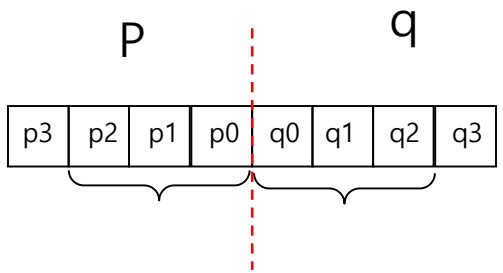
- A group of samples from the set (p2, p1, p0, q0, q1, q2) is filtered only if:
  - BS>0 and
  - $|p0-q0|<\alpha$ and $|p1-p0|<\beta$ and $|q1-q0| \leq \beta$

Defined in Standard Doc.

- Filter implementation
  - **bS** $\in \{1, 2, 3\}$
    - » Input: p1,p0, q0,q1 → 4 tap Filtering
    - » Output: p'0, q'0

Filtered output

Filtering Techniques of H.264/AVC Video Codec (5)



» **Condition (1)** $|p2-p0|<\beta$ → 4 tap Filtering

Output: p'1

» **Condition (2)** $|q2-q0|<\beta$ → 4 tap Filtering

Output: q'1

Filtered output

Filtering Techniques of H.264/AVC Video Codec (6)

– **bS=4**

» Input: p2, p1,p0, q0,q1, q2

Filtered output » Output: p'2, p'1, p'0, q'0, q'1, q'2

» **Condition (1)**

if |p2-p0|<β and |p0-q0|<round($\frac{\alpha}{4}$) and Luma

P'0= five-tap of p2, p1, p0, q0, q1
P'1= four-tap of p2, p1, p0, and q0
P'2= five-tap of p3, p2, p1, p0, and q0

else

P'0= three-tap of p1, p0, and q1

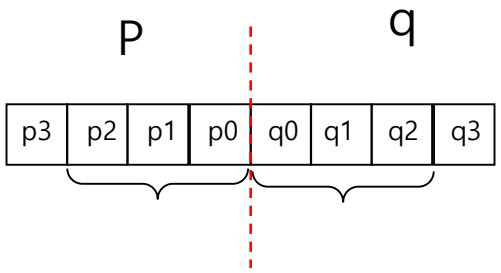
» **Condition (2)**

if |q2-q0|<β and |p0-q0|<round($\frac{\alpha}{4}$) and Luma

P'0= five-tap of q2, q1, q0, p0, p1
P'1= four-tap of q2, q1, q0, and p0
P'2= five-tap of q3, q2, q1, q0, and p0

else

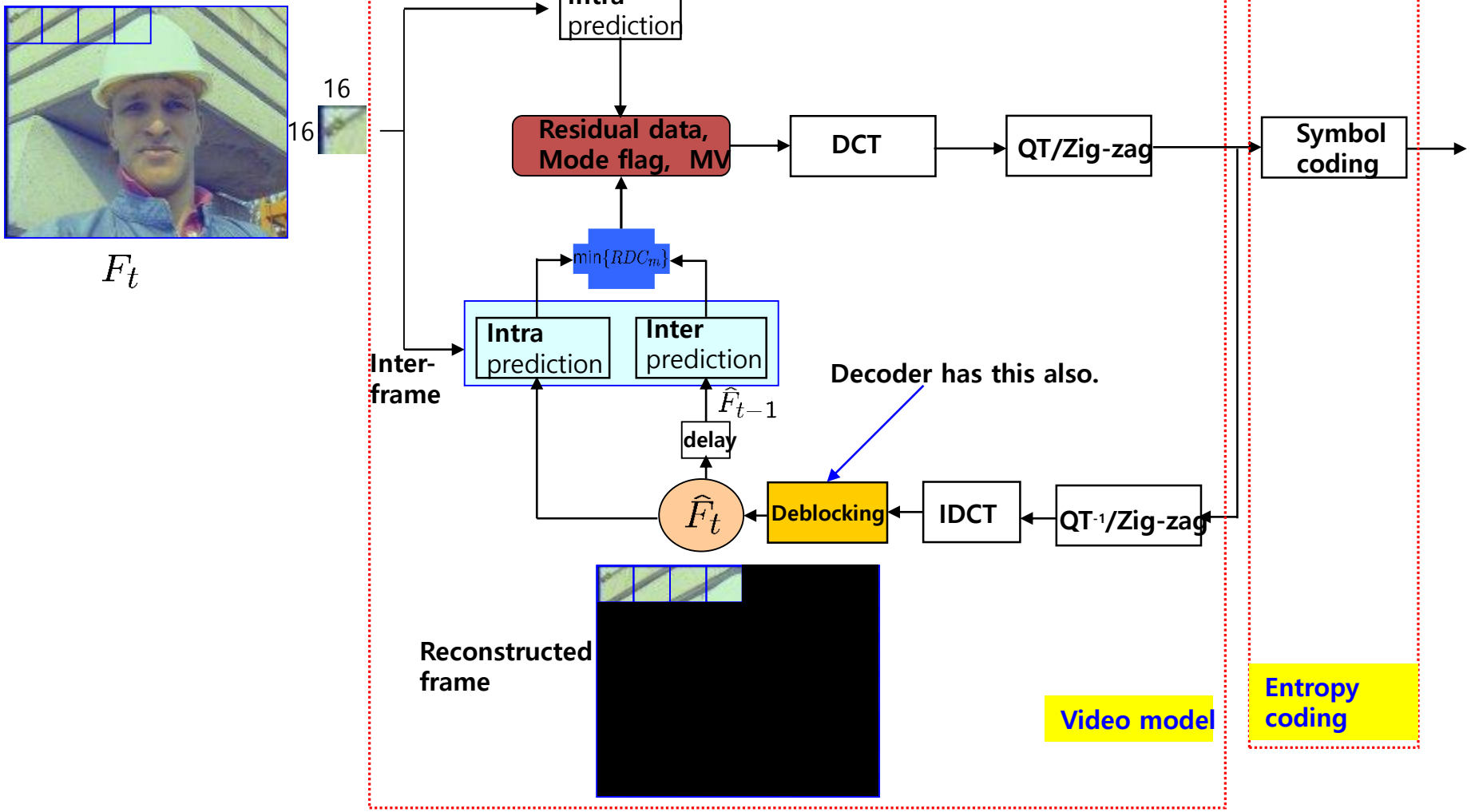
q'0= three-tap of q1, q0, and p1



❖ **Intra prediction: reconstructed MB without filtering**

Filtering Techniques of H.264/AVC Video Codec (7)

❖ Hybrid Video Coding Scheme



Deblock Filter of H.264/AVC Video Codec (1)

- Deblocking Filter: (jm 12.2)
  - "Image.c"
    - DeblockFrame (img, enc_picture->imgY, enc_picture->imgUV);

"loopFilter.c"

```
for (i=0; i < img->PicSizeInMbs; i++)
{
    DeblockMb( img, imgY, imgUV, i ) ;
}
```

Deblock Filter of H.264/AVC Video Codec (2)

- Deblocking Filter: (jm 12.2)

```
for (i=0; i < img->PicSizeInMbs; i++)
{
    DeblockMb( img, imgY, imgUV, i ) ;
}
```

```
for( dir = 0 ; dir < 2 ; dir++ ) // filter first vertical edges, followed by horizontal
{
  for( edge=0; edge<4 ; edge++ ) // first 4 vertical strips of 16 pel, then 4 horizontal
  {
                    ⋮
        GetStrength(Strength, img, MbQAddr, ~)
        EdgeLoopLuma( imgY, Strength, ~)
        EdgeLoopChroma( imgUV[0], Strength, ~)
                    ⋮
  }
}
```

Deblock Filter of H.264/AVC Video Codec (3)

- Deblocking Results (1)

Before

After

Deblock Filter of H.264/AVC Video Codec (4)

- Deblocking Results (2)

Before

After

Deblock Filter of H.264/AVC Video Codec (5)

- Deblocking Results (3)

Before

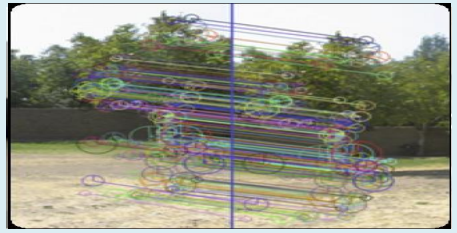
After

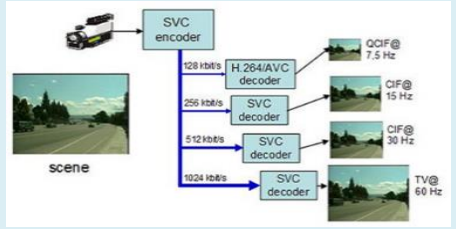
Deblock Filter of H.264/AVC Video Codec (6)

- Deblocking Results (4)

Before

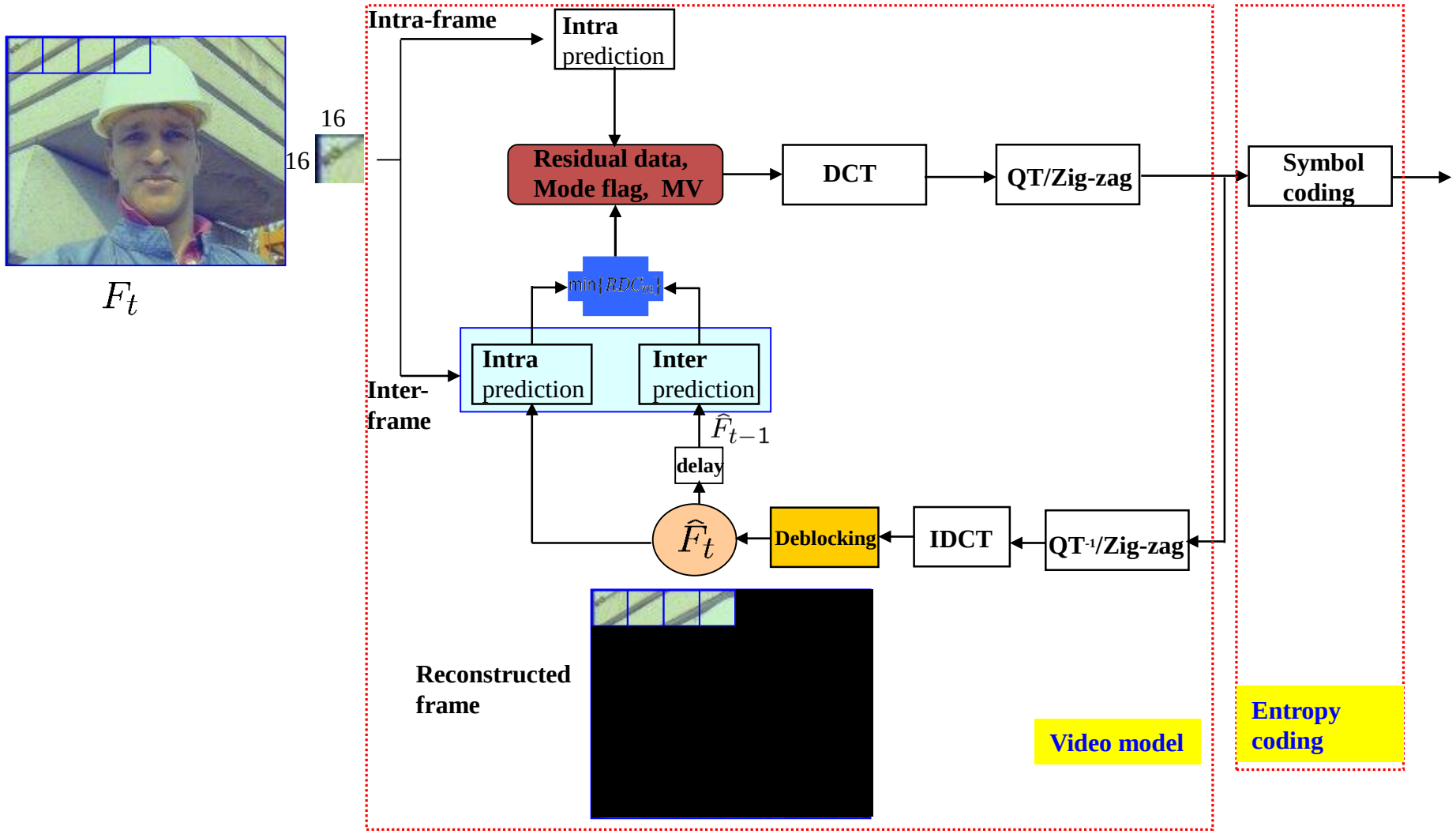
After





Contents

❖ Hybrid Video Coding Scheme



Fast Schemes for H.264/AVC Encoder (2)

❖ The Consumed Time Profile

- Actual computational complexity.

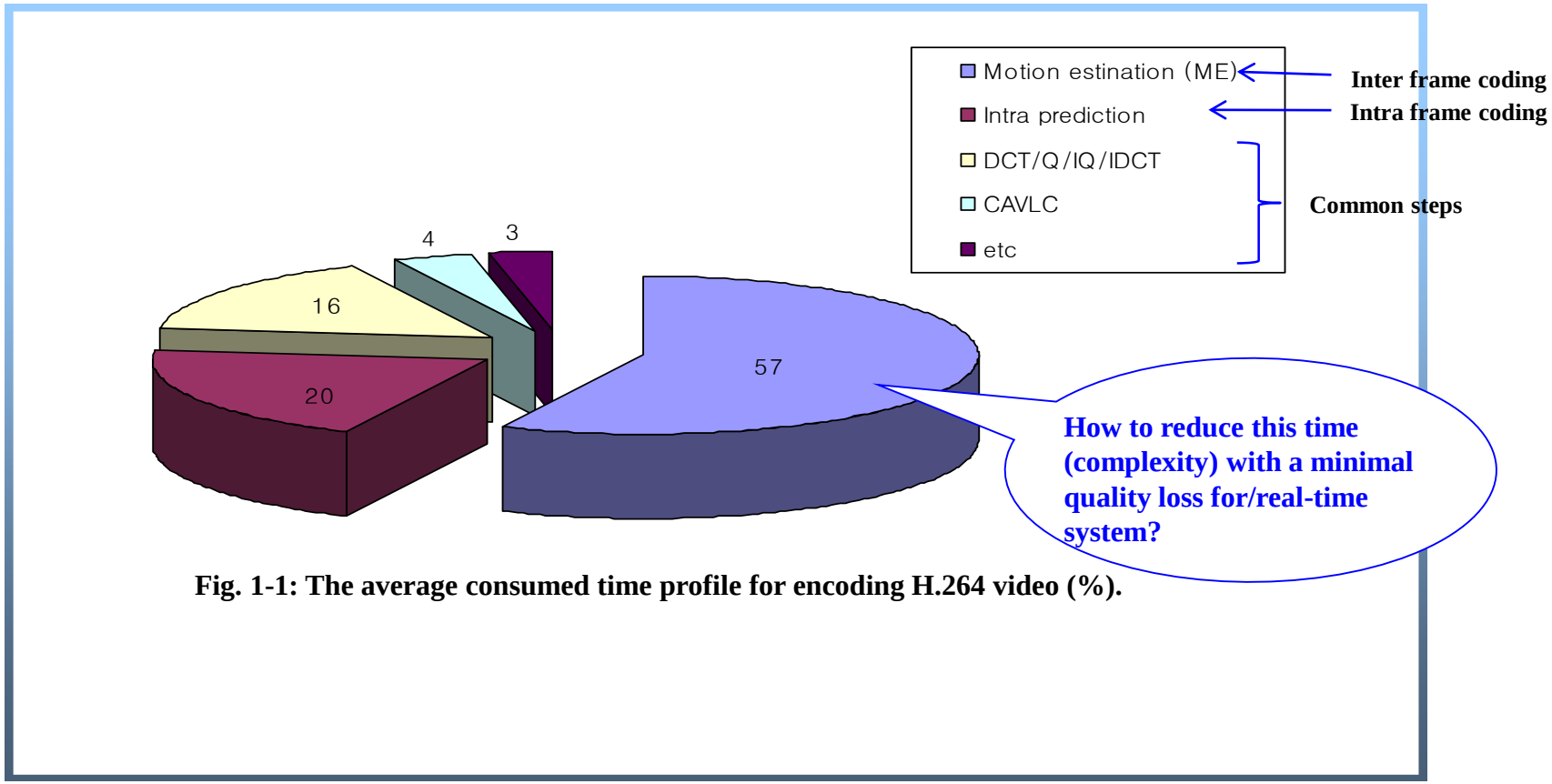


Fig. 1-1: The average consumed time profile for encoding H.264 video (%).

❖ Motion Estimation Process

- Mode type selection
- MV search process

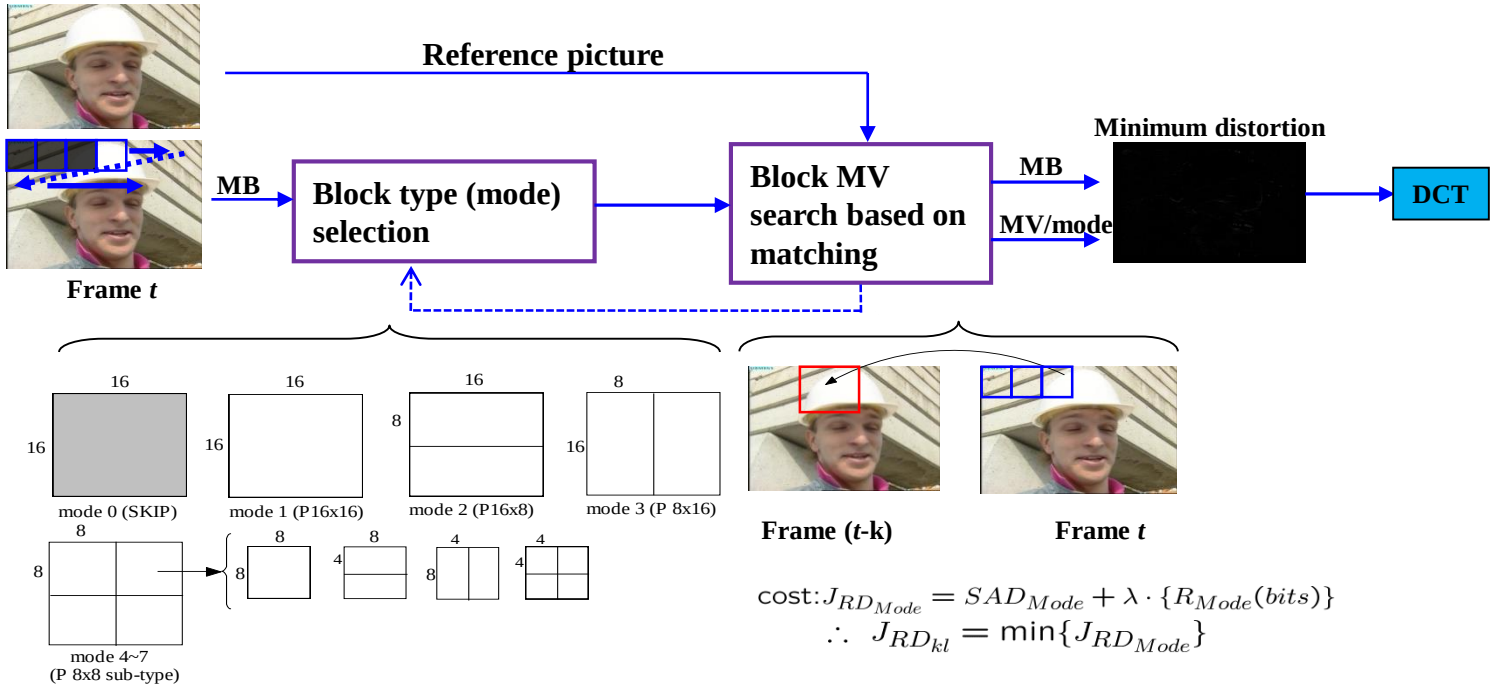


Fig. 3-1: Motion estimation: Inter prediction (H.264 and SVC) procedure with 7 variable block sizes.

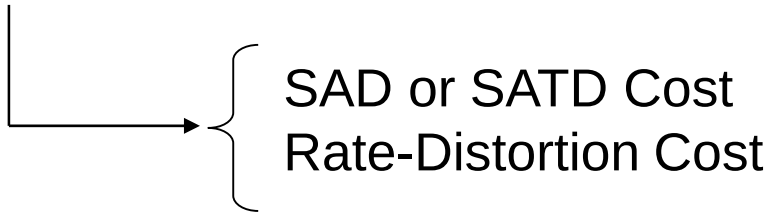
Fast Schemes for H.264/AVC Encoder (4): Mode Decision Procedure (1)

❖ Mode Decision Process

- Inter prediction: Inter mode: 7 types
- Intra prediction: Intra mode: 4 types
  - Direction modes for each block type.

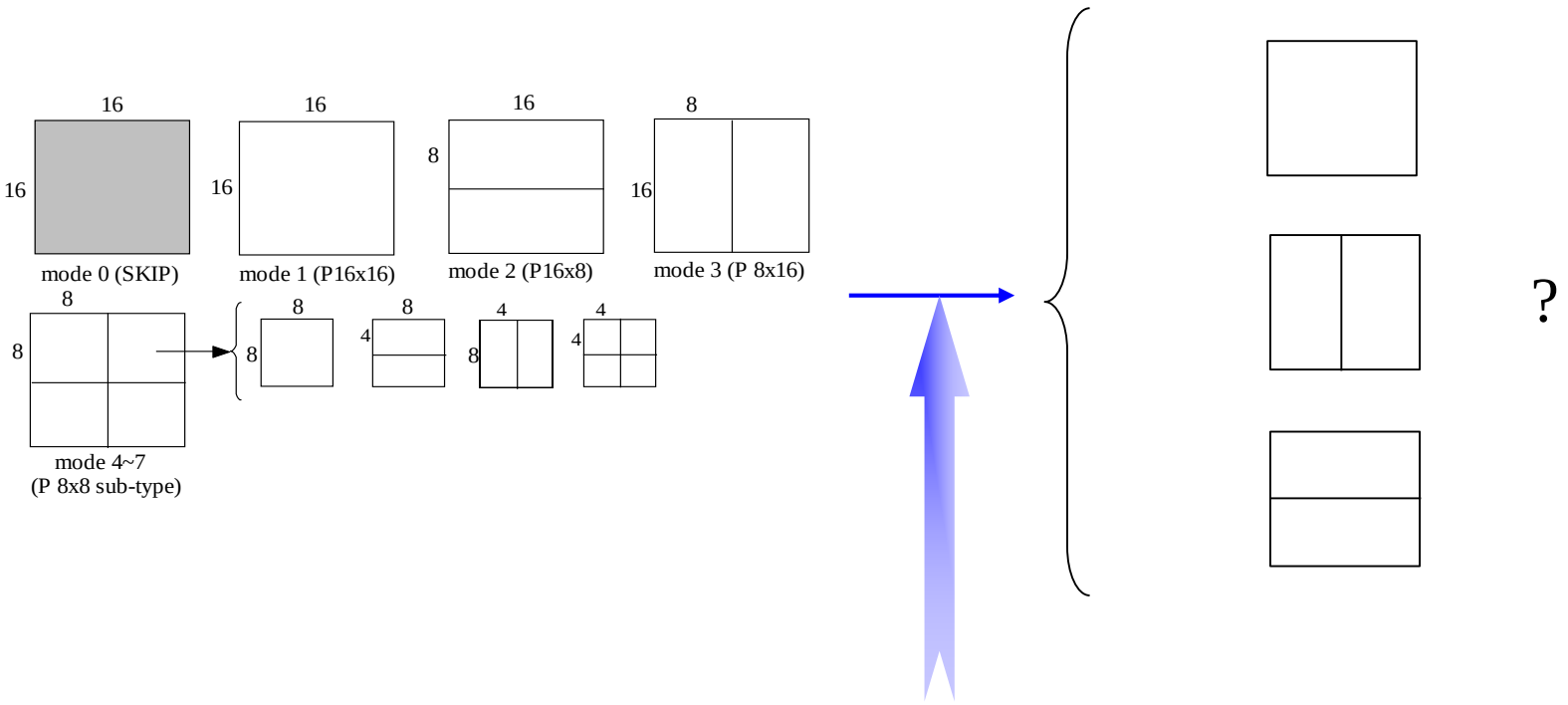
❖ What Is *Mode Decision*?

- A process that selects the best coding (block or direction types) type in terms of the given criterion.



Fast Schemes for H.264/AVC Encoder (5): Mode Decision Procedure (2)

❖ Full set of Modes → Partial Candidate Modes

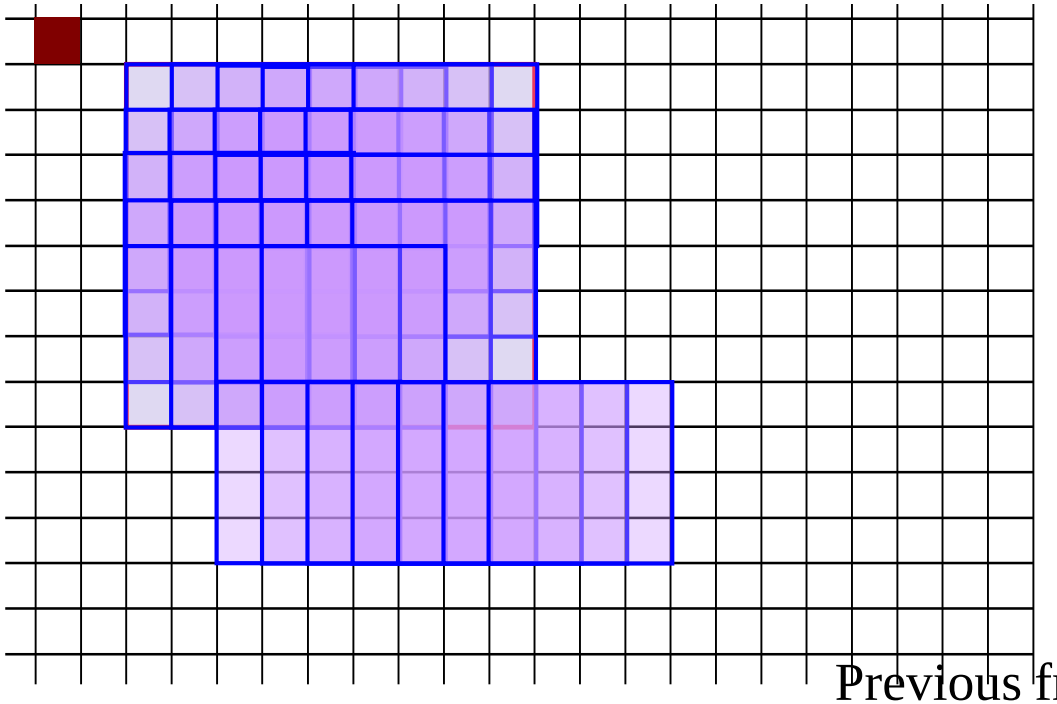


<u>Selection rule</u> with negligible loss of quality.

Fast Schemes for H.264/AVC Encoder (7): Fast MV Search Scheme

❖ Default: Full Search Method (2)

- Exhaustive search process.



Previous frame (*t*-1)

Fast Schemes for H.264/AVC Encoder (8): Fast MV Search Scheme

❖ Fast MV Search Method.

- In this time, let's think about ***how can we speed-up the search process***....!!!!

Brain storming

Fast Schemes for H.264/AVC Encoder (9): Fast MV Search Scheme

❖ How To ????

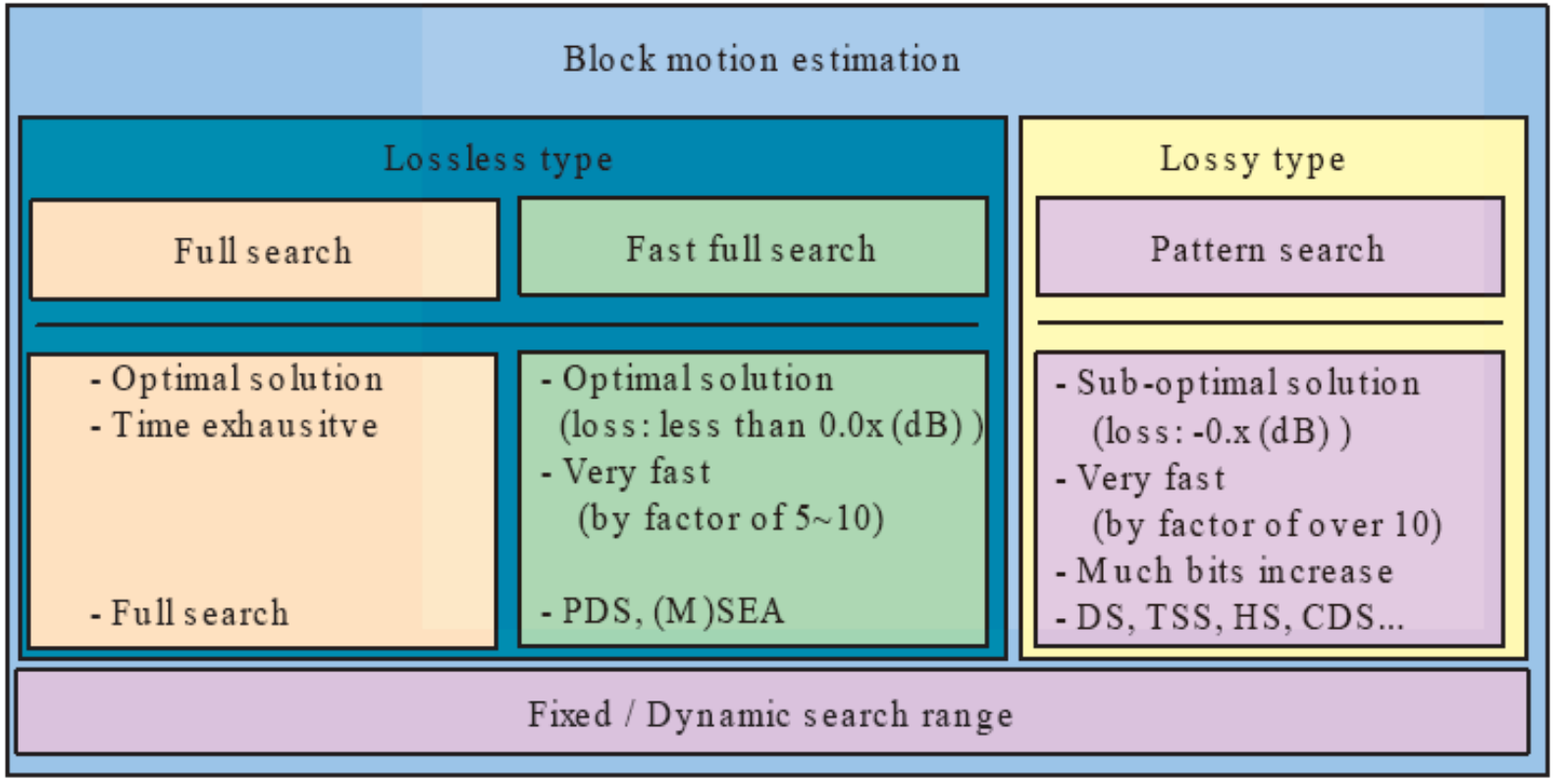
- Basically, in full search method, all pixel points are examined for finding the best matching point.
- To speed-up?
  - *Reducing the number of search points.*
  - *Reducing the operation number for each SAD computation.*
  - *Adaptive search range control, not fixed search range.*

  - ***Constraint:***
    - *No loss or very small loss of quality and bit increment*

Fast Schemes for H.264/AVC Encoder (10): Fast MV Search Scheme

❖ Class of Fast MV Search Approaches

- Block-based Motion Search.



❖ Based on ***MV Distribution***.

- Analysis of the characteristics of the motion vector distribution
  - Center-biased motion distribution.
  - Dominant vertical/horizontal shifts.

Table 1. An example for the motion vector distribution.

Radium (pel)	Tennis	Football	Caltrain	Susie	Salesman	Claire
0	0.2622	0.6196	0.0416	0.0938	0.6562	0.9076
1	0.3751	0.7297	0.5373	0.3592	0.9452	0.9702
2	0.5276	0.7983	0.8523	0.5950	0.9609	0.9870
3	0.7178	0.8641	0.9168	0.7622	0.9741	0.9932
4	0.8402	0.9042	0.9380	0.8225	0.9795	0.9950
5	0.8930	0.9329	0.9561	0.8779	0.9853	0.9957
6	0.9200	0.9483	0.9720	0.9038	0.9957	0.9964
7	0.9599	0.9658	0.9894	0.9365	0.9975	0.9973

❖ MV Distribution

- Sequence: "Foreman"
- Search Range: ± 7

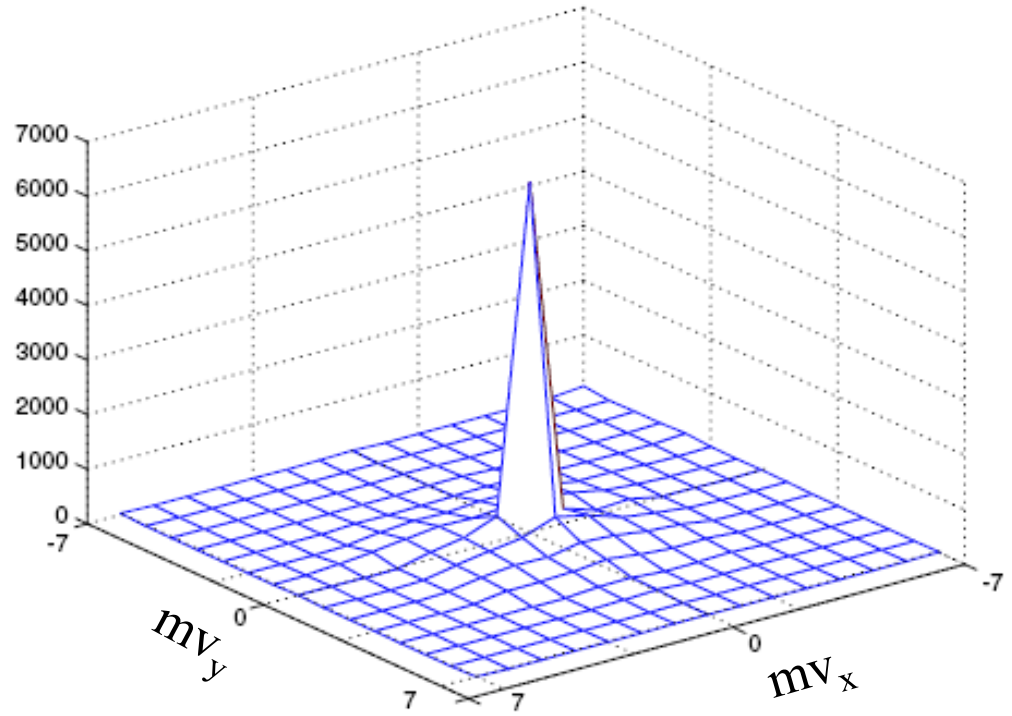


Fig 3-2. The characteristics of the motion vector distribution.

Fast Schemes for H.264/AVC Encoder (13): Design of Fast MV Search Schemes

❖ Fast MV Estimation Techniques

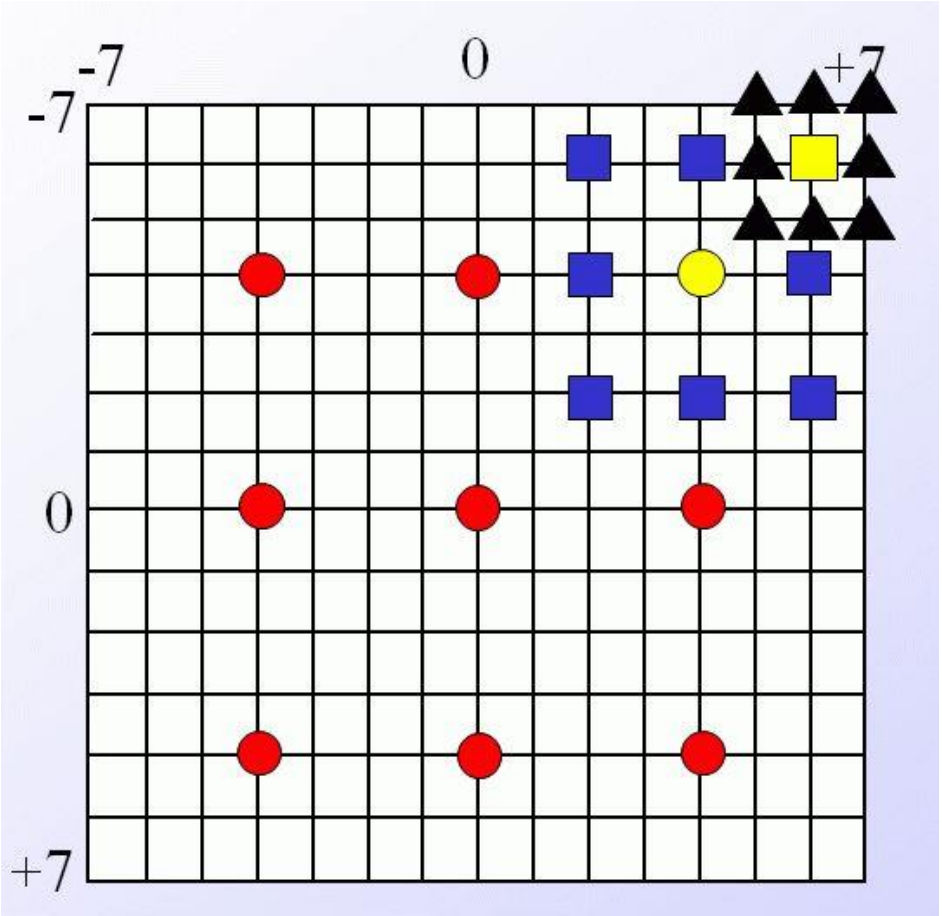
- Fixed Pattern Searches
  - Diamond pattern search
  - Cross-Diamond pattern search
  - Hexagon Search
  - Four Step Search
  - Adaptive rood pattern search
  - ….. (so many)

- Fast Full Searches
  - PDS (APDS)
  - SEA (MSEA)

  ……

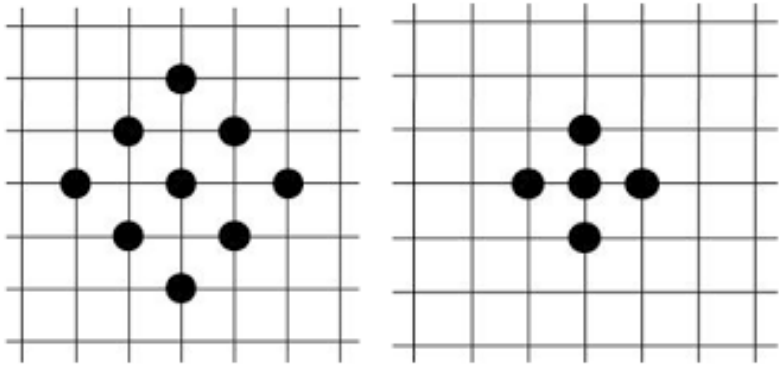
Fast Schemes for H.264/AVC Encoder (13): Pattern Searches

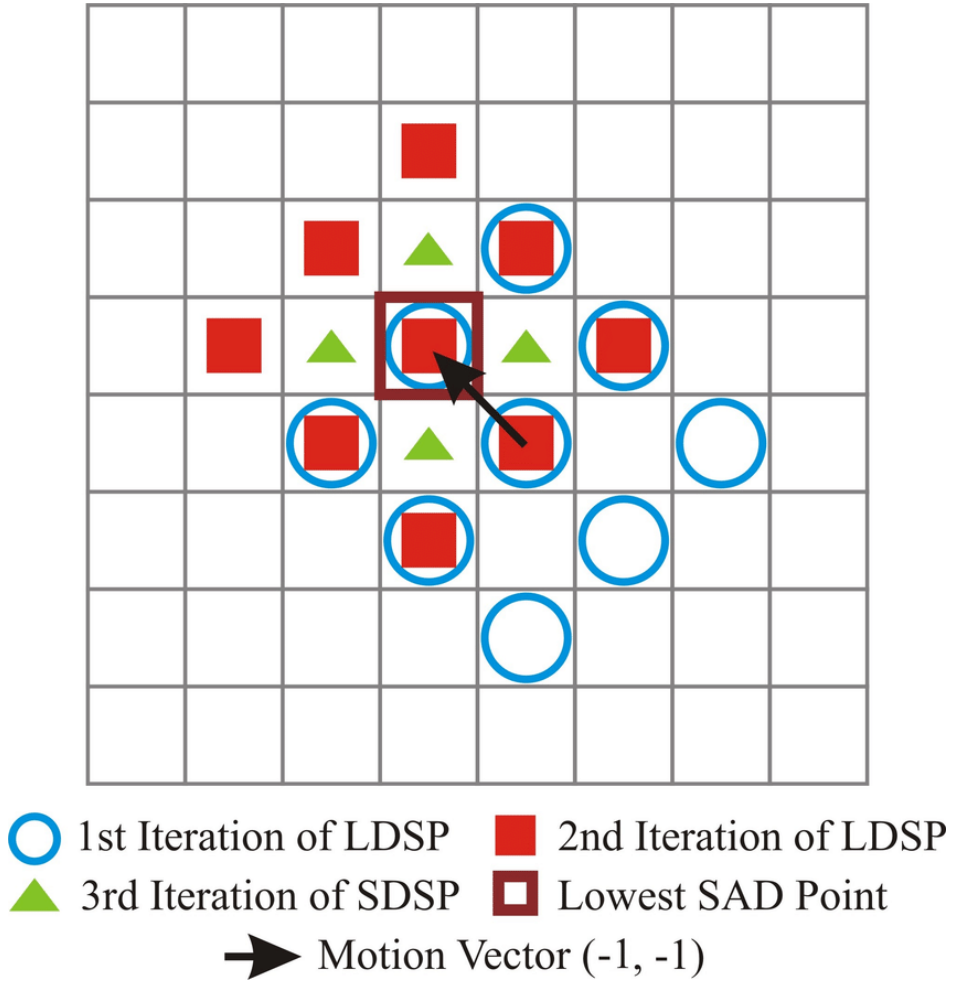
❖ Three-step Search (TSS)



Fast Schemes for H.264/AVC Encoder (14): Pattern Searches

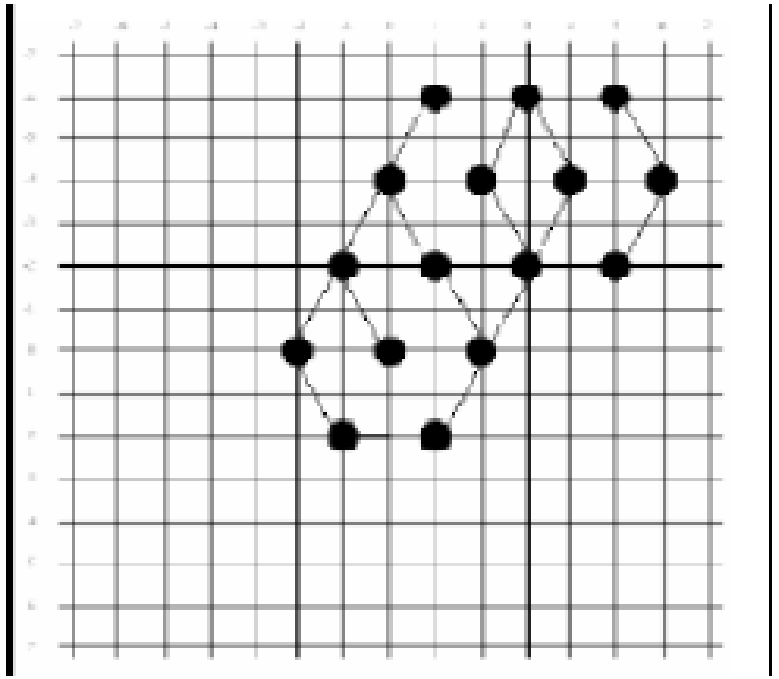
❖ Diamond Search (DS)





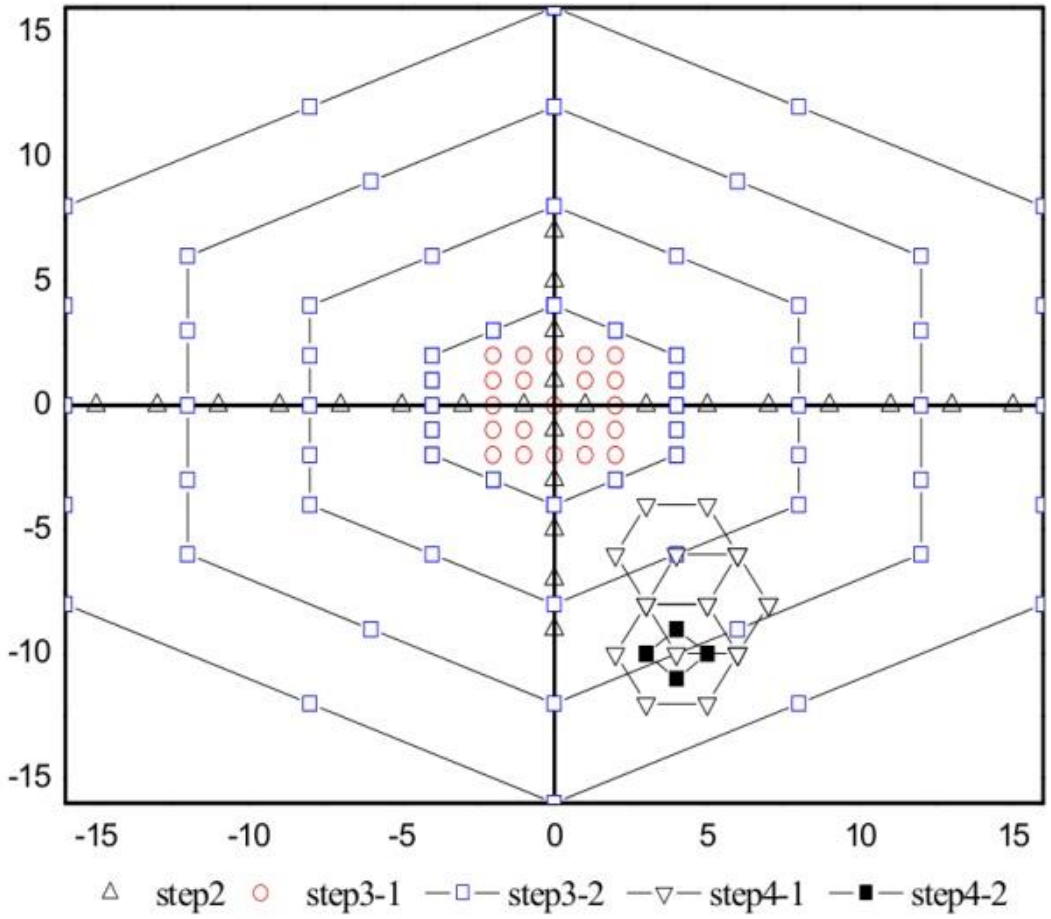
Fast Schemes for H.264/AVC Encoder (15): Pattern Searches

❖ Hexagon Search (HS)



Fast Schemes for H.264/AVC Encoder (16): Pattern Searches

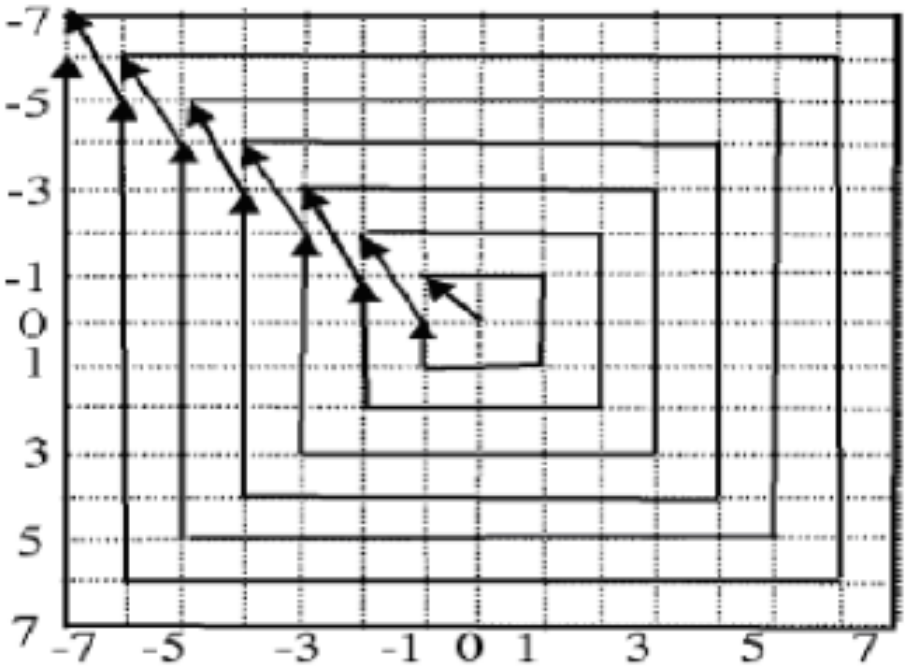
❖ UM-Hexagon Search (UM Hexagon)



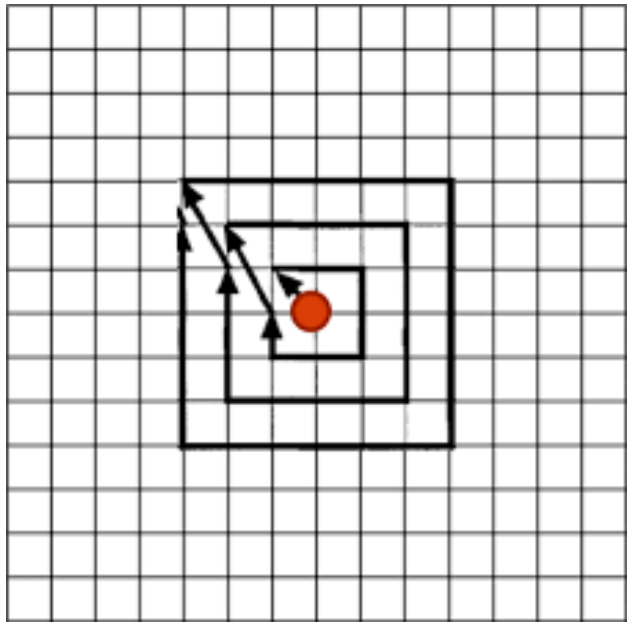
Fast Schemes for H.264/AVC Encoder (15): Fast Full Search

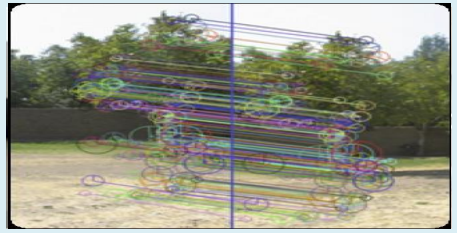
❖ Partial Distortion Search (PDS)

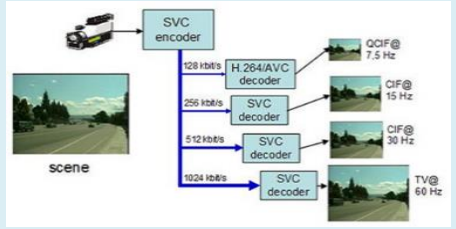
- Partial Distortion
- Spiral Search



[Spiral search]







Contents

Summary

- Deblocking Filter in H.264/AVC
  - Problem
  - In-loop Deblocking model
- Fast Schemes in H.264/AVC
  - Mode decision process
  - MV search process
  - Pattern Searches
  - Fast Full Searches

HomeWork #4(Due: ~ to May 1)

❖ Derivation of Motion Vector Distribution

- Using your ME module that has been implemented.
- Method:
  - Motion vector (MV) is 2-dim. Data. (mv_x, mv_y)
  - Count all available MV for all inter frames.
    - Normalized MV distribution.
  - Plot 2-dim. MV distribution.
    - You can use any plotting tools like Lab view and Matlab or Labview
  - Test sequences (CIF):
    - Foreman, football, bus, paris
    - 100 frames for each sequence.
- Submission: Analyzed Document

VICL
Visual Information Computing Lab

Thank you for your attention.!!!
QnA

http://vicl.sookmyung.ac.kr