マルチメディア配信システム

- No.5 映像圧縮技術 -

渡辺　裕

Multimedia Distribution System

- No.5 Video Compression Technology -

Hiroshi Watanabe

映像画像圧縮技術

- 空間冗長性除去
  - 画像圧縮技術
- 時間冗長性除去
  - 動き補償フレーム間予測技術

Video Compression Technology

- Spatial Redundancy Reduction
  - Image Compression Technology
- Temporal Redundancy Reduction
  - Motion Compensated Interframe Prediction

動きベクトル検出

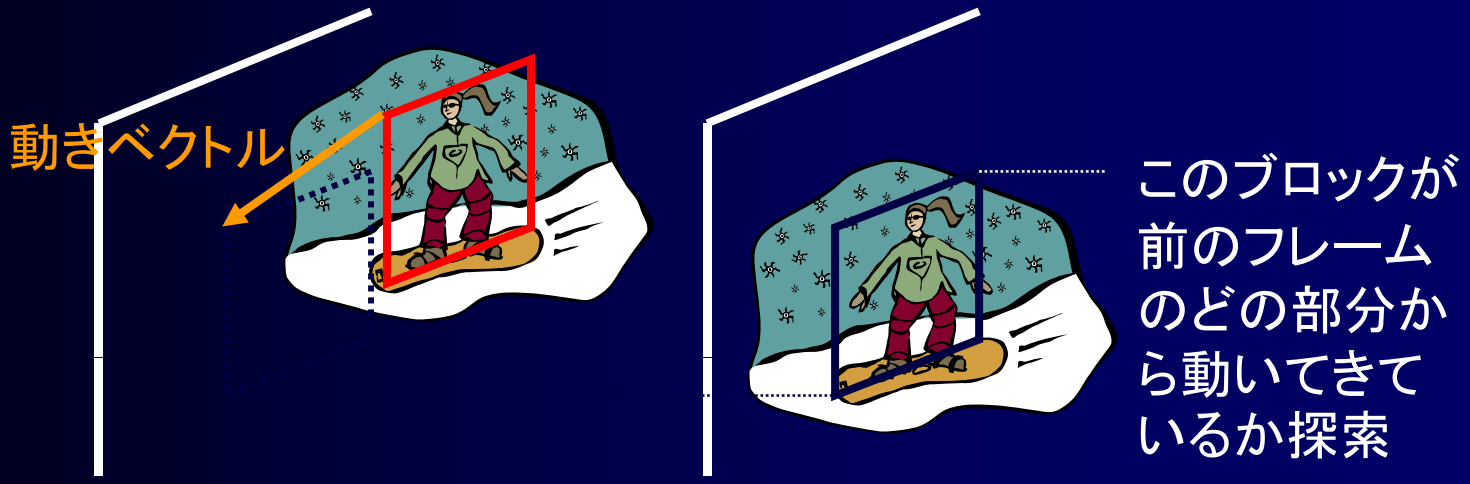
- 動きベクトル検出の目的
  - 動き補償フレーム間予測符号化
  - Shape from the motion (2次元ビデオからの3次元形状の再構成)
  - ビデオ検索用の特徴量の抽出
  - ビデオ符号化制御用の特徴量の抽出

Motion Vector Detection

- Purpose of the motion vector detection
  - Motion compensated interframe prediction
  - Shape from the motion (Reconstruction of 3-D shape from 2-D Video)
  - Feature Extraction for video search
  - Feature extraction for video coding control

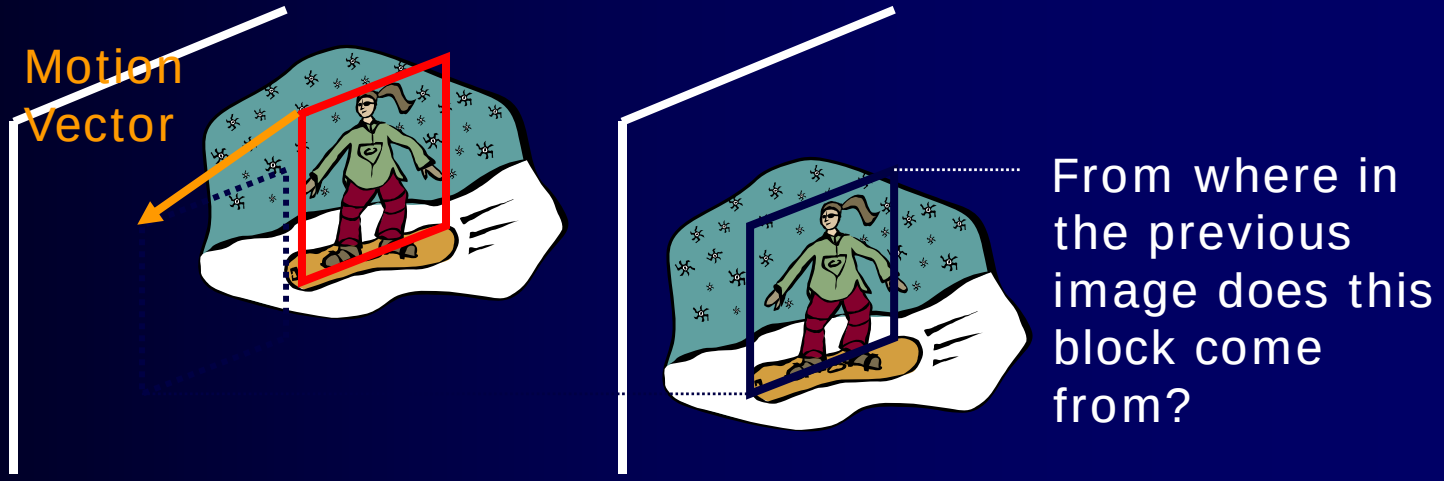
動き補償フレーム間予測

- ブロック単位の動き補償フレーム間予測
  - ブロック単位に画像を平行移動させ、次のフレームの画素を予測



Motion Compensated Interframe Prediction

- Block-based Motion Compensated Interframe Prediction
  - Predict pixels of the next frame by shifting block-based image in parallel



勾配法

- 連続するフレーム間で隣接する画素値から動きを求める手法
  - 微小区間において輝度勾配が一定であることを利用
  - コンピュータビジョンの分野ではオプティカルフローと呼ばれる (Horn and Schunk, 1981)
- 1次元の場合の算出式
  - k フレームにおける画素インデクス i の位置の画素値を $x_i(k)$ としたとき，動き量 u は次式で与えられる

$$u = \frac{x_i(k) - x_i(k-1)}{x_i(k) - x_{i-1}(k)}$$

← B

← A

Gradient Method

- Method to obtain motion from adjacent pixels between successive frames
  - Utilize the feature of constant gradient of brightness in a small section
  - Called "optical flow" in the area of computer vision (Horn and Schunk, 1981)
- Derivation for 1-dimensional case
  - Let the pixel value at index i of frame k be $x_i(k)$, motion u is given by

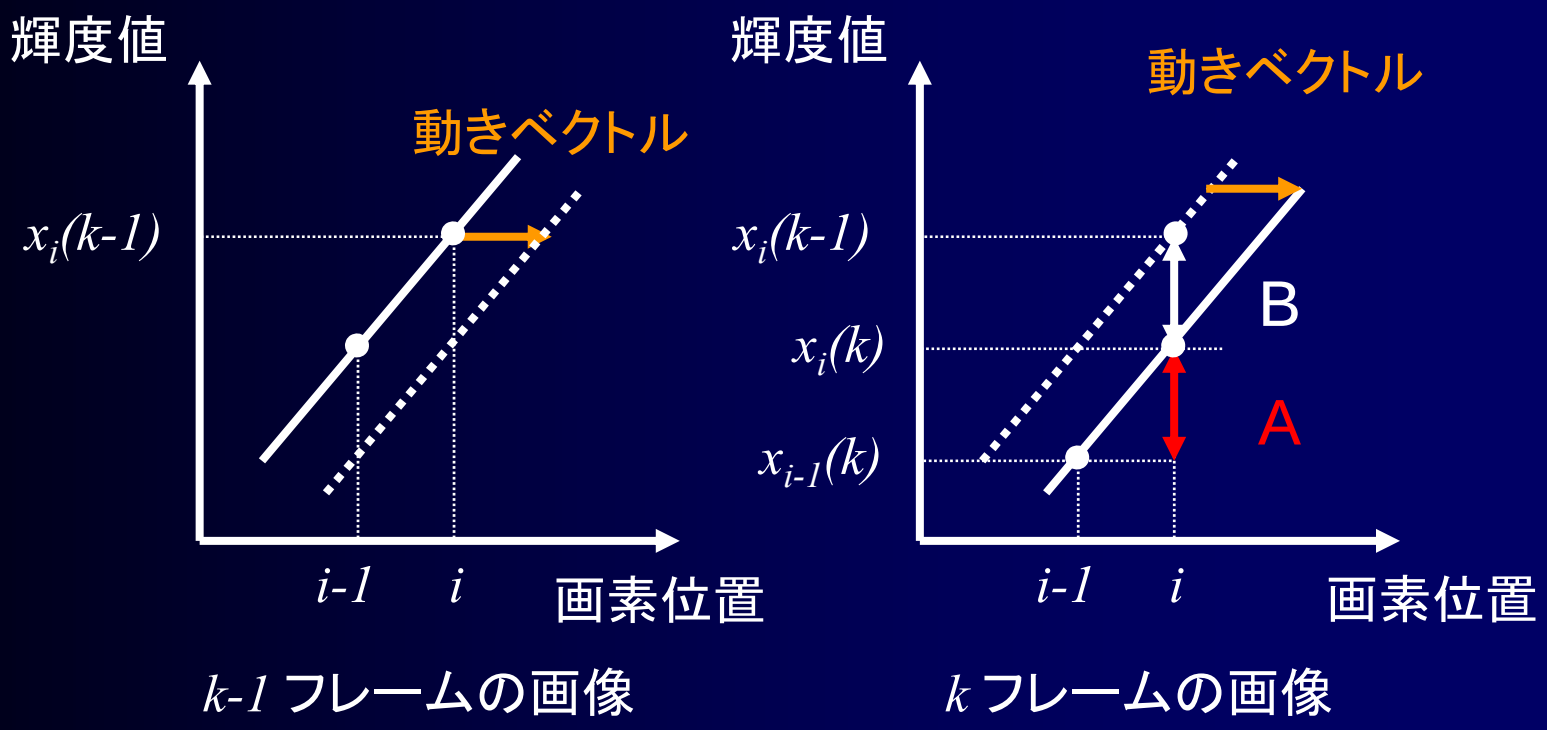
$$u = \frac{x_i(k) - x_i(k-1)}{x_i(k) - x_{i-1}(k)}$$

← B

← A

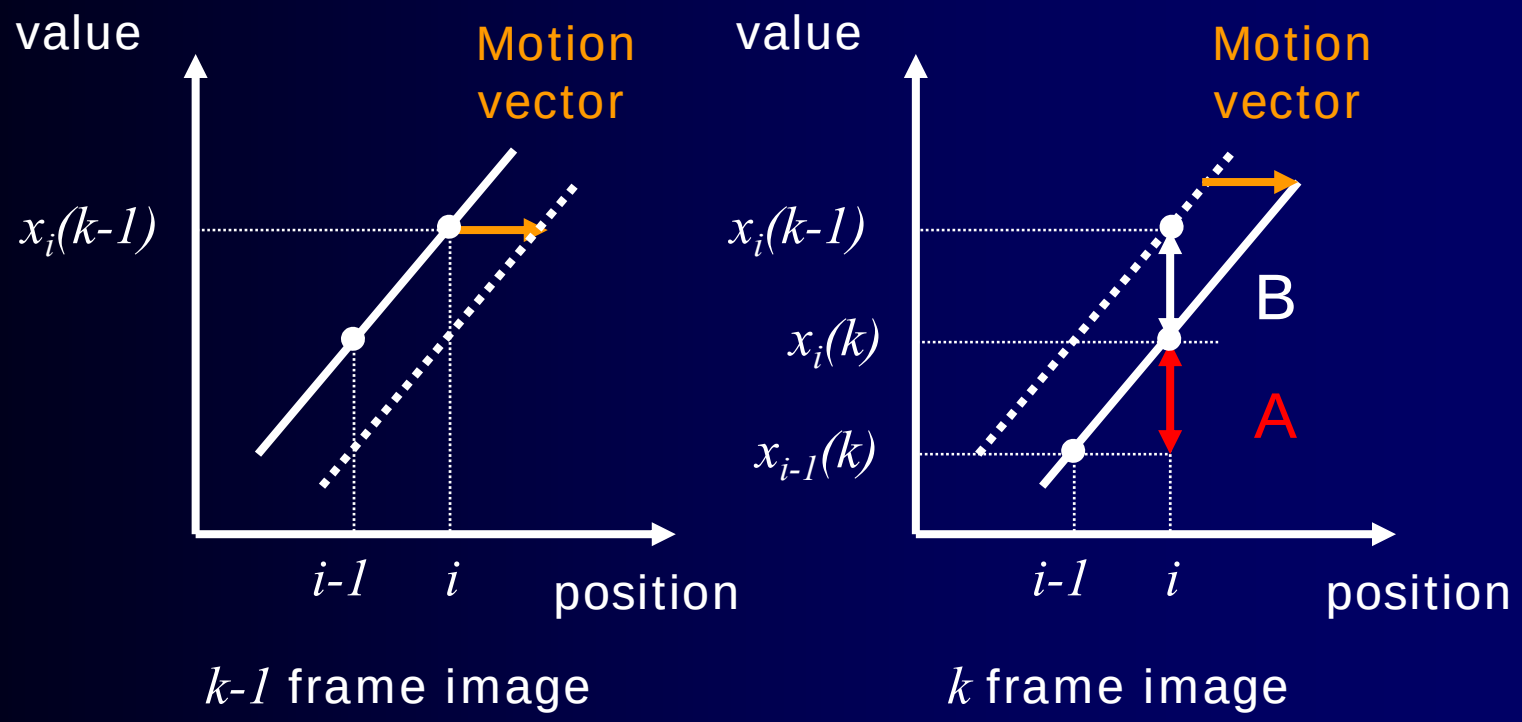
勾配法 (2)

- 動きによる輝度変化 … 動きがほぼ1画素の場合



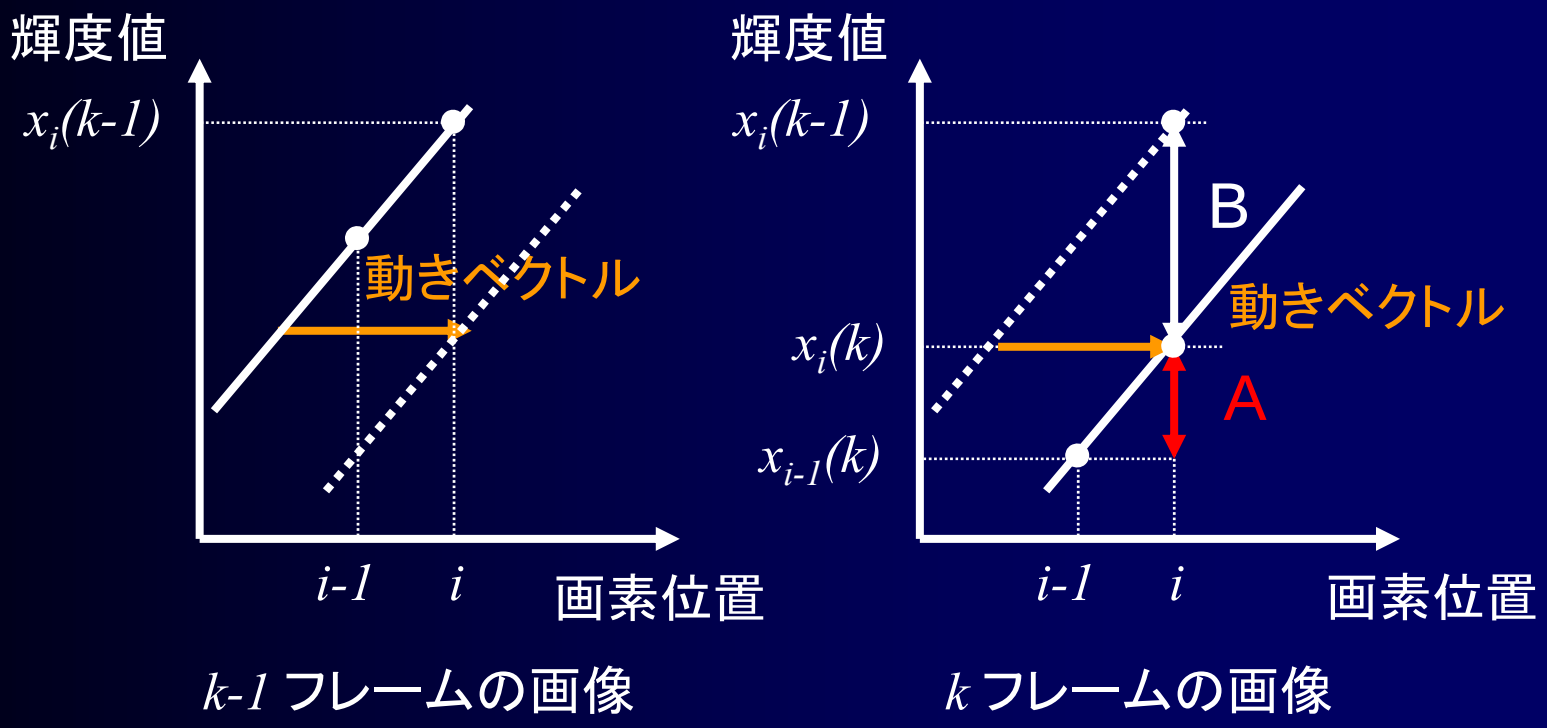
Gradient Method (2)

- Brightness change by motion … In case of motion is almost 1 pixel



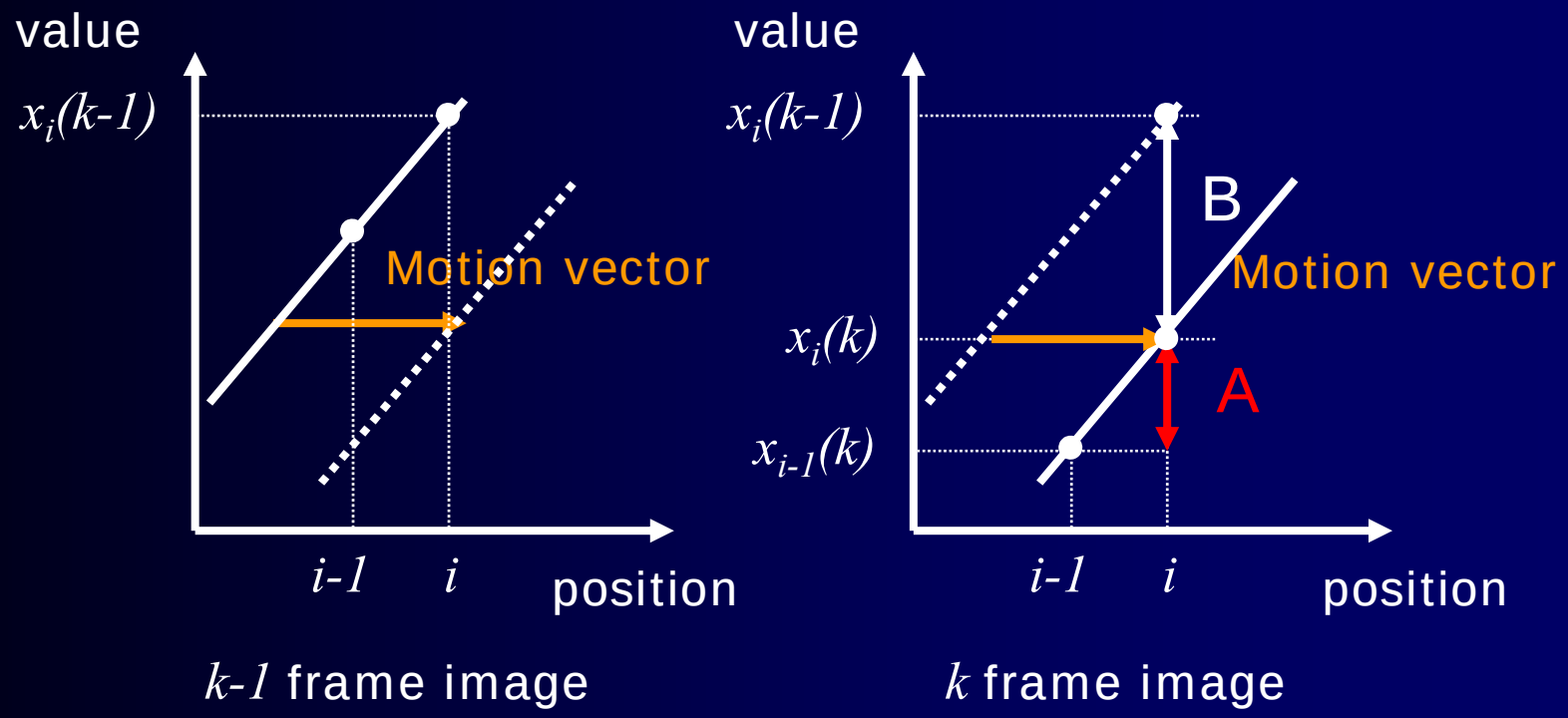
勾配法 (3)

- 動きによる輝度変化... 動きがほぼ2画素の場合



Gradient Method (3)

- Brightness change by motion … In case of motion is almost 2 pixels



勾配法（4）

- 2次元の場合の算出式
  - k フレームの画像 $x_{i,j}$ が $k+1$ フレームでは $w=(u,v)$ だけ移動して $x_{i+u,j+v}$ に変化したとき、画素値を i, j に関してテイラー展開(1次の項まで)

$$x_{i-u,j-v} = x_{i,j} - a\frac{\partial}{\partial i}\left(x_{i,j}\right) - b\frac{\partial}{\partial j}\left(x_{i,j}\right)$$

  - フレーム間差分を $d_{i,j}$ で定義する

$$d_{i,j} \equiv x_{i,j} - x_{i-u,j-v}$$

Gradient Method (4)

- Derivation for 2-dimensional case
  - When k frame image $x_{i,j}$ moves $w=(u,v)$ at $k+1$ frame, and changed to $x_{i+u,j+v}$, Taylor series expansion of pixel value in regard to position i, j (1st term) is calculated

$$x_{i-u,j-v} = x_{i,j} - a\frac{\partial}{\partial i}\left(x_{i,j}\right) - b\frac{\partial}{\partial j}\left(x_{i,j}\right)$$

  - Define interframe difference by $d_{i,j}$

$$d_{i,j} \equiv x_{i,j} - x_{i-u,j-v}$$

勾配法 (5)

- 2次元の場合の算出式
  - フレーム間差分値をテイラー展開に代入

$$a\frac{\partial}{\partial i}x_{i,j}+b\frac{\partial}{\partial j}x_{i,j}=-d_{i,j}$$

  - 水平，垂直方向の単位ベクトルを I, J としたベクトル形式

$$(Ia+Jb)(I\frac{\partial}{\partial i}+J\frac{\partial}{\partial j})x_{i,j}=-d_{i,j}$$

Gradient Method (5)

- Derivation for 2-dimensional case
  - Insert interfame difference to Taylor series expansion

$$a\frac{\partial}{\partial i}x_{i,j}+b\frac{\partial}{\partial j}x_{i,j}=-d_{i,j}$$

  - Vector form based on horizontal and vertical unit vector I, J

$$(Ia+Jb)(I\frac{\partial}{\partial i}+J\frac{\partial}{\partial j})x_{i,j}=-d_{i,j}$$

勾配法 (6)

- 2次元の場合の算出式
  - $x_{i,j}$ の勾配をグラジェントで表現すると

$$(aI + bJ)\nabla x_{i,j} = -d_{i,j}$$

  - 動きベクトルは $(u,v)=(aI, bJ)$ であるから

$$(u,v) = -\frac{d_{i,j}}{\nabla x_{i,j}}$$

Gradient Method (6)

- Derivation for 2-dimensional case
  - Slope of $x_{i,j}$ is represented by gradient

$$(aI + bJ)\nabla x_{i,j} = -d_{i,j}$$

  - Since motion vector is $(u,v)=(aI, bJ)$, we have

$$(u,v) = -\frac{d_{i,j}}{\nabla x_{i,j}}$$

勾配法の特徴

- 輝度変化の類似性を利用した検出法
- 画素単位の詳細な動きを検出
- 雑音に弱い
- 隣接画素間の差が小さいときに不安定

Characteristics of Gradient Method

- Detection method utilize similarity of brightness change
- Detect one pixel accuracy motion
- Weak against noise
- Unstable when difference of adjacent pixels value is small

ブロックマッチング法

- 画像をブロックに区切り、その領域の平均的な動きを求める手法
- 剛体の平行移動を仮定
- 動きベクトルの探索
  - k フレームの小ブロックの画像信号 $x_{i,j}(k)$ $(0 \le i \le b_i,\ 0 \le j \le b_j)$
  - 探索範囲 $S(m,n)$ ... $(-s_i \le m \le s_i,\ \ s_j \le n \le s_j)$
  - 動きベクトル: (u,v)

$$(u,v) = \arg\min_{m,n} \left(\sum_{i=0}^{b_i} \sum_{j=0}^{b_j} \left| x_{i,j}(k) - x_{i-m,j-n}(k-1) \right|^2 \right)$$

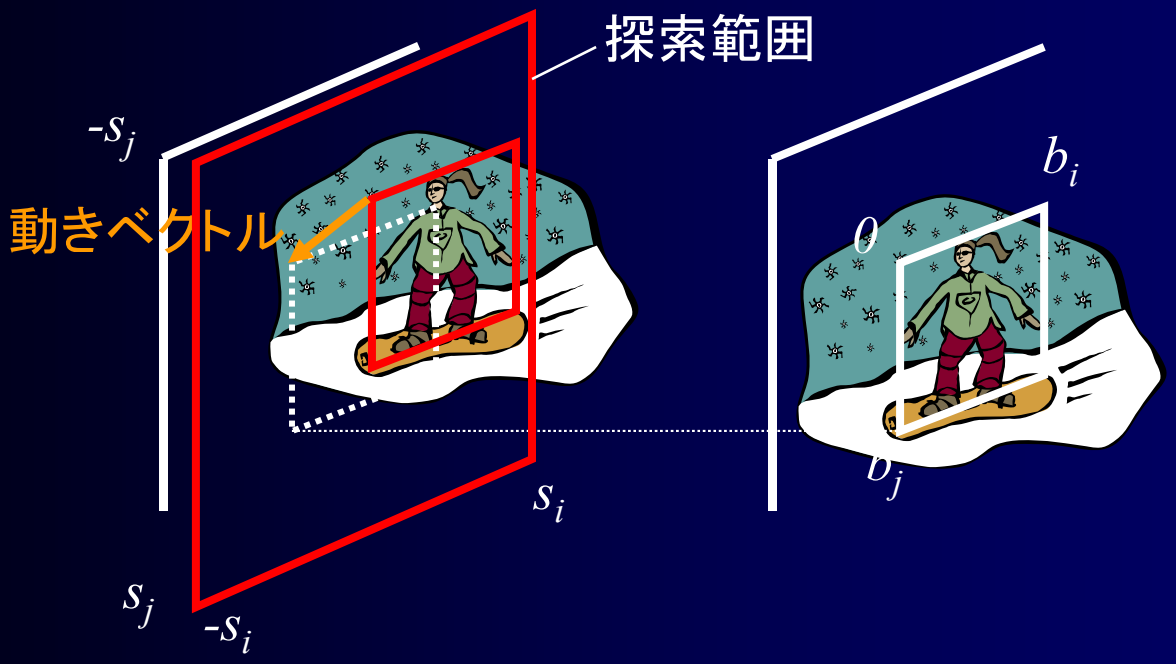
Block Matching Method

- Brake image to block and obtain average motion in the region
- Presume parallel motion of rigid object
- Motion vector search
  - Image signal at small block in k frame: $x_{i,j}(k)$ $(0 \le i \le b_i, 0 \le j \le b_j)$
  - Search range: $S(m,n) \ldots (-s_i \le m \le s_i,\ s_j \le n \le s_j)$
  - Motion vector: (u,v)

$$(u,v) = \arg\min_{m,n}\left(\sum_{i=0}^{b_i}\sum_{j=0}^{b_j} \left| x_{i,j}(k) - x_{i-m,j-n}(k-1) \right|^2 \right)$$

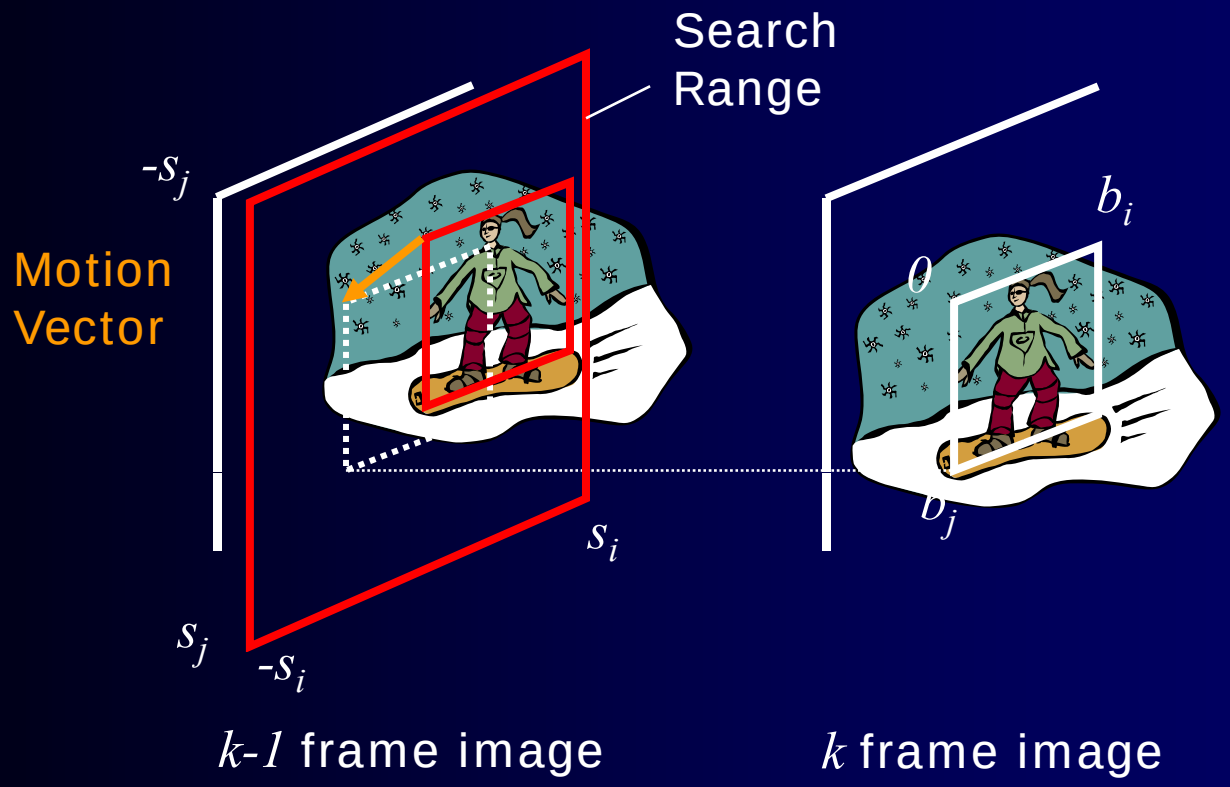
ブロックマッチング法(2)

- 全探索法



Block Matching Method(2)

- Full search method



ブロックマッチング法の特徴

- 輝度の絶対値の類似性を利用した手法
  - ブロック単位で類似性を計算するため雑音に強く安定性に優れる
- 動きのモデルが単純
  - 剛体の平行移動という仮定が崩れると正確な動き検出が不可能
- 全探索法の問題点
  - 膨大な演算量
  - 探索処理の高速化が必要

Characteristics of Block Matching Method

- Method to utilize similarity of absolute value of brightness
  - Similarity is calculated block basis, stable against noise
- Simple motion model
  - If the condition of rigid parallel motion is not hold, it is impossible to detect the right motion
- Problem of full Search method
  - Immense operational cost
  - Needs high speed search processing

ブロックマッチング法の高速化 (1)

- L1ノルムの採用

$$(u,v) = \arg\min_{x,y} \left(\sum_{i=0}^{b_i} \sum_{j=0}^{b_j} \left| x_{i,j}(k) - x_{i-x,j-y}(k-1) \right| \right)$$

  - MAD規範とも呼ばれる
    - MAD: Mean Absolute Difference

      (cf. MSE: Mean Square Error, 平均2乗誤差)

Speeding Up Block Matching Method (1)

- Employment of L1-norm

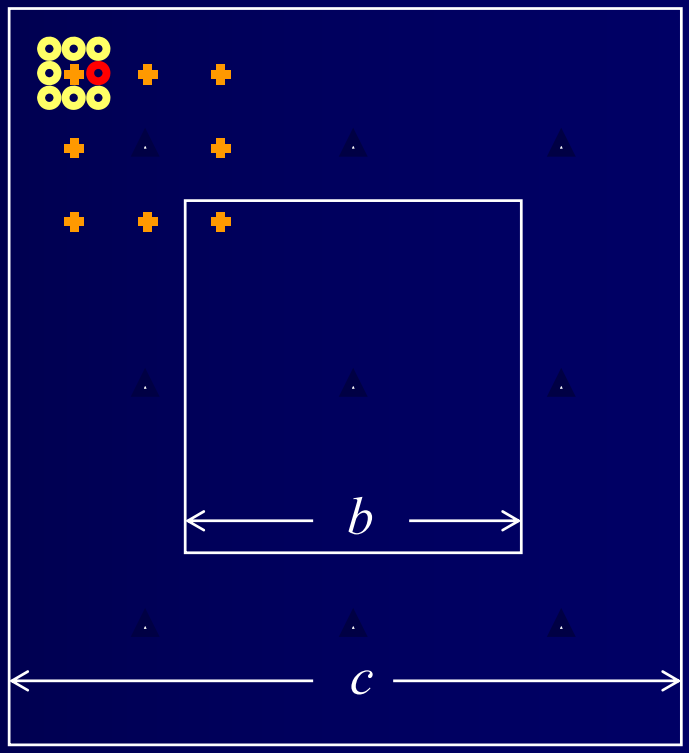
$$(u,v) = \arg\min_{x,y}\left(\sum_{i=0}^{b_i}\sum_{j=0}^{b_j}\left|x_{i,j}(k) - x_{i-x,j-y}(k-1)\right|\right)$$

  - Called MAD criterion
    - MAD: Mean Absolute Difference

      (cf. MSE: Mean Square Error)

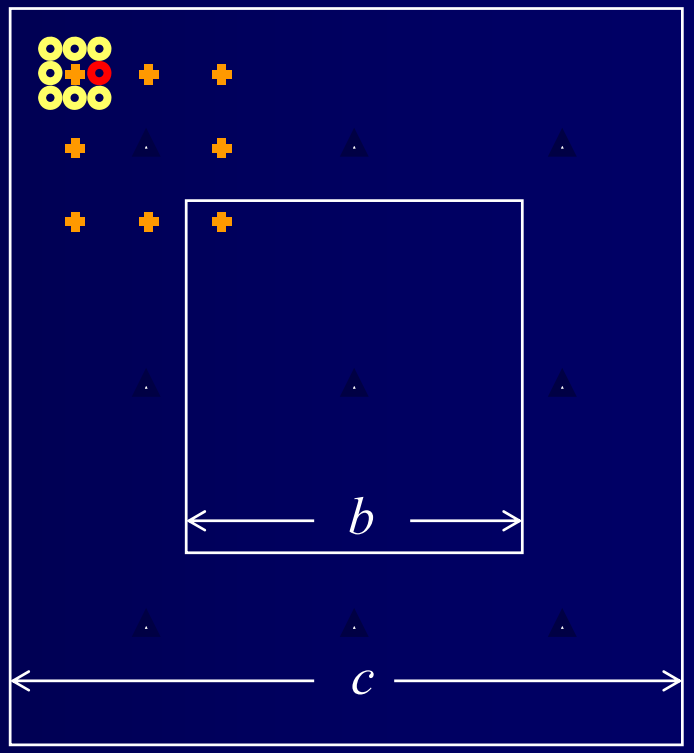
ブロックマッチング法の高速化 (2)

- 木探索法(Tree Search)
  - 第1ステップ
    - ▲ポイント9点
  - 第2ステップ
    - ＋ポイント8点
  - 第3ステップ
    - ○ポイント8点
- 演算量($c=2b$)
  - 全探索 : $b^2c^2=b^2 4b^2=4b^4$
  - 木探索 : $(9+8+8)b^2 =25b^2$



Speeding Up Block Matching Method (2)

- Tree Search Method
  - First step
    - ▲ 9 points
  - Second step
    - + 8 points
  - Third step
    - ○ 8 points
- Computational Complexity ($c=2b$)
  - Full search: $b^2c^2=b^2 4b^2=4b^4$
  - Tree search: $(9+8+8)b^2 =25b^2$



ブロックマッチング法の高速化 (3)

- 動き(0,0)の先行評価と誤差計算の打切り
- プログラミングの工夫による高速化

```
e(0,0) = d ( x(i,j,k), x(i-0,j-0,k-1) );
min.d = e(0,0);
loop x,y (-si<m<si, -sj<n<sj);
 sum=0;
 for (i=0; i<bi; i++) {
     for(j=0; j<bj; j++) {
       sum = sum + d ( x(i,j,k), x(i-m,j-n,k-1) );
       if( sum>e(0,0) ) break;
     }
 }
 if( sum<min.d ) min.d=sum;
end of loop xy;
```

Speeding Up Block Matching Method (3)

- Pre-evaluation of motion (0,0) and break off error calculation
- Speeding up by programming design

```
e(0,0) = d ( x(i,j,k), x(i-0,j-0,k-1) );
min.d = e(0,0);
loop x,y (-si<m<si, -sj<n<sj);
 sum=0;
 for (i=0; i<bi; i++) {
     for(j=0; j<bj; j++) {
       sum = sum + d ( x(i,j,k), x(i-m,j-n,k-1) );
       if( sum>e(0,0) ) break;
     }
 }
 if( sum<min.d ) min.d=sum;
end of loop xy;
```

1画素マッチング法

- 全探索法の誤差計算を分解

$$(u,v) = \arg\min_{m,n}\left(\sum_{i=0}^{b_i}\sum_{j=0}^{b_j} d_{i,j,m,n}\right)$$

$$d_{i,j,m,n} = \left|x_{i,j}(k) - x_{i-m,j-n}(k-1)\right|^2$$

- 最小誤差を与える (u,v) が求まった時点で、$d(i,j,u,v)$ の(i,j) に関する誤差分布を見れば、ブロックの中のどの部分が誤差が小さいか、1画素単位で判定可能となる。

1 Pixel Matching Method

- Decompose error calculation of full search method

$$(u,v) = \arg\min_{m,n}\left(\sum_{i=0}^{b_i}\sum_{j=0}^{b_j} d_{i,j,m,n}\right)$$

$$d_{i,j,m,n} = \left|x_{i,j}(k) - x_{i-m,j-n}(k-1)\right|^2$$

- when *(u,v)* giving the minimum error is obtained, we can detect where in block gives the small error by 1 pixel accuracy by viewing error distribution *d(i,j,u,v)* in regard to *(i,j)* .

1画素マッチング法 (2)

- 画素ごとの動きベクトルの探索
  - $d(i,j,m,n)$ の (i,j) を固定したとき、$d(i,j,m,n)$ が最小となる (m,n) を求めれば、画素ごとの動きベクトルを得る。

- ノイズによる不安定性の解決策
  - 上記の手法ではノイズや被写体の変形による影響が大きいため、最小値を与える $d(i,j,m,n)$ だけでなく、2番目、3番目の候補をメモリに格納しておき、隣接画素の候補の値に重みを付けて、(m,n) 決定に際して柔軟な判定を行う。

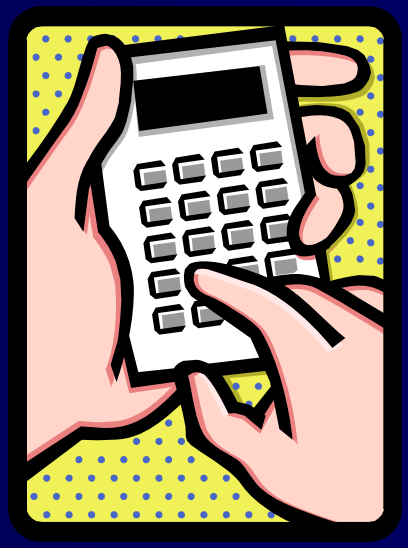
1 Pixel Matching Method (2)

- Motion vector search at pixel unit
  - We can obtain motion vector at pixel unit by obtaining (m,n) which gives minimum $d(i,j,m,n)$ for fixed (i,j)

- Remedy for instability by noise
  - Method described above are influenced by noise and shape change. Thus, soft decision by weighting neighborhood pixels to detect (m,n) is taken after storing 2nd and 3rd candidates as well as the minimum one.

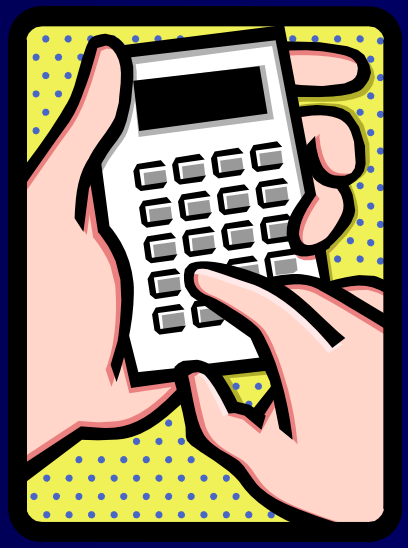
1画素マッチング法の特徴

- 動きベクトルの精度
  - 詳細なベクトル分布を算出可能
- 問題点
  - 莫大な演算量
  - 軟判定基準が不明確



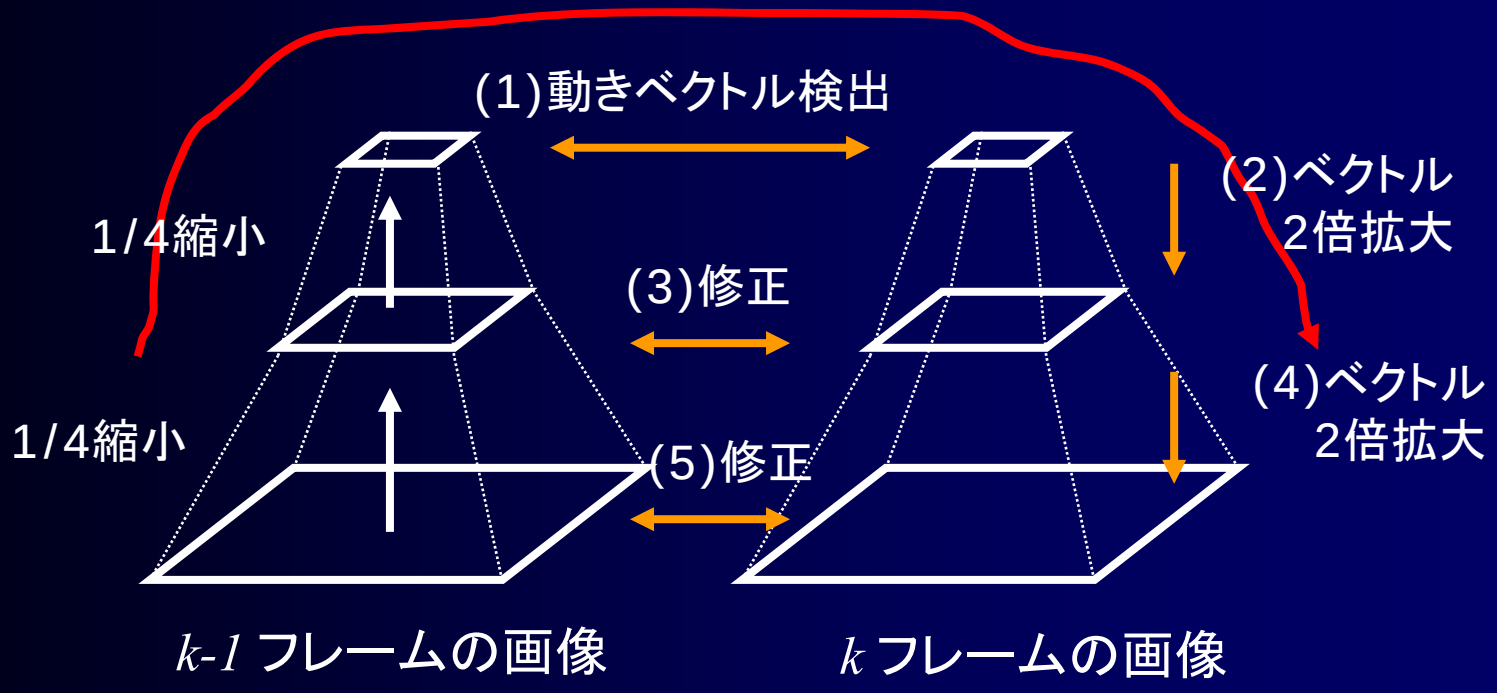
Characteristics of 1 Pixel Matching Method

- Accuracy of motion vector
  - Detailed vector distribution can be obtained
- Problem
  - Huge computational cost
  - Unclear soft decision



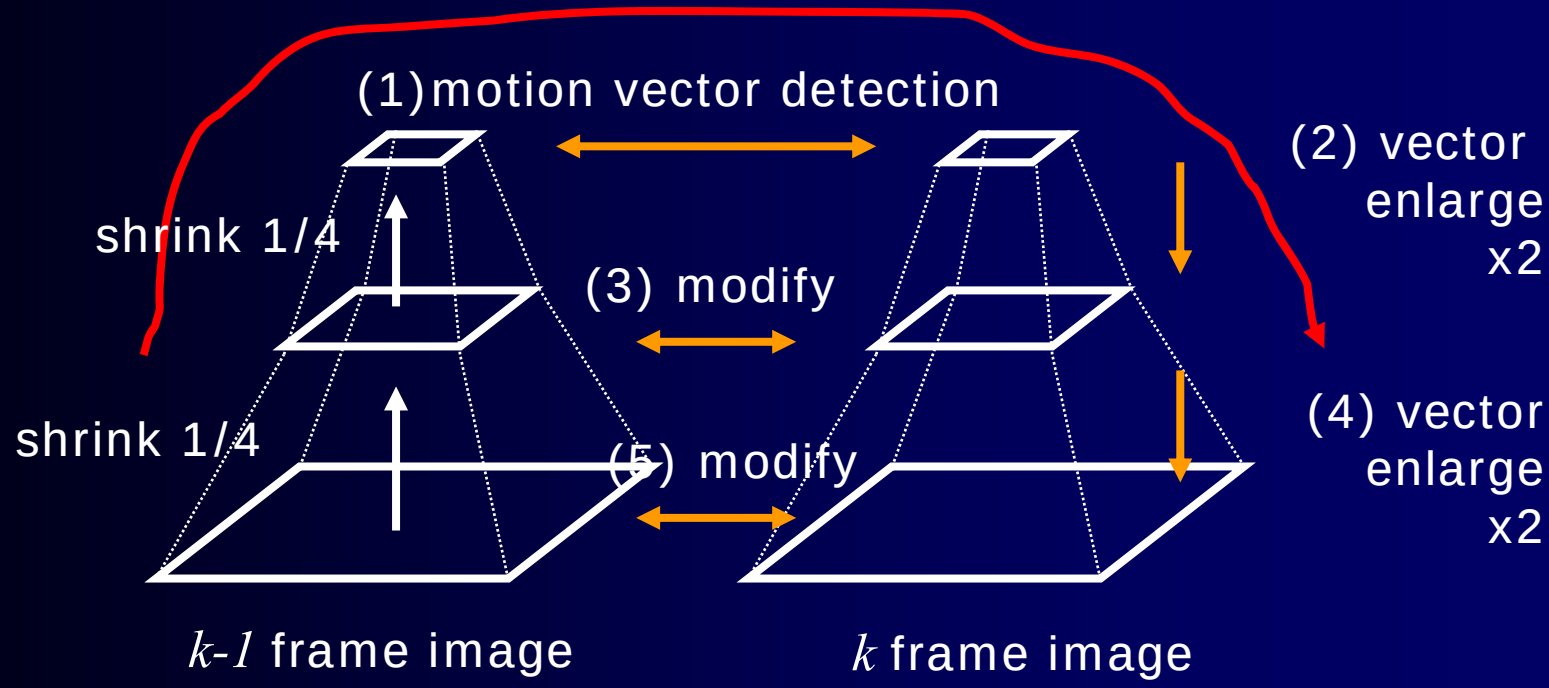
階層探索法

- 階層画像の作成と上位レイヤ（縮小画像）での動き検出



Hierarchical Search Method

- Creation of hierarchical image and motion detection at higher layer (smaller image)



階層探索法 (2)

- 動きベクトルの拡大
  - レイヤ L の画像を $x(L)$、その1/4画像を $x(L+1)$ とする。レイヤ $L+1$ での動きベクトル $(u(L+1), v(L+1))$ を拡大して、レイヤ L の動きベクトルを得る。

$$(u_p(L), v_p(L)) = 2(u(L+1), v(L+1))$$

- 拡大ベクトルの修正
  - 拡大ベクトルの周囲+/-1画素の範囲で最小値を探索

$$d = x(i, j, k, L)$$
$$- x(i - u_p(L) \pm 1, j - v_p(L) \pm 1, k - 1, L)$$

Hierarchical Search Method (2)

- Enlarge of motion vector
  - Let layer L image be $x(L)$、and ¼ shrunken image be $x(L+1)$. Obtain layer L motion vector by enlarge x2 from $L+1$ motion vector $(u(L+1), v(L+1))$

$$(u_p(L), v_p(L)) = 2(u(L+1), v(L+1))$$

- Modification of enlarged motion vector
  - Search +/-1 pixels around enlarged motion vector

$$d = x(i, j, k, L) - x(i - u_p(L) \pm 1, j - v_p(L) \pm 1, k - 1, L)$$

階層探索法の特徴

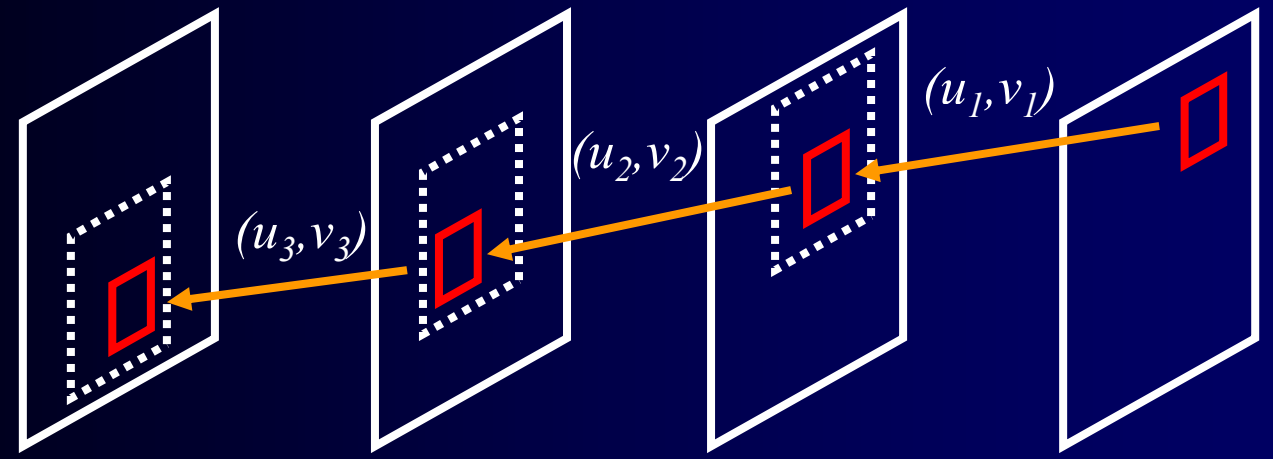
- 雑音の影響を回避でき、大局的な動きが比較的正確に得られる
- 演算量を削減可能
- 縮小画像を格納するためのメモリ領域が増大
- 上位レイヤでの検出ミスが下位に伝播

Characteristics of Hierarchical Search Method

- Can avoid noise influence, relatively correct global motion can be obtained
- Can reduce computational complexity
- Needs memory to store shrunken images
- Detection error at higher layer propagates to lower layer

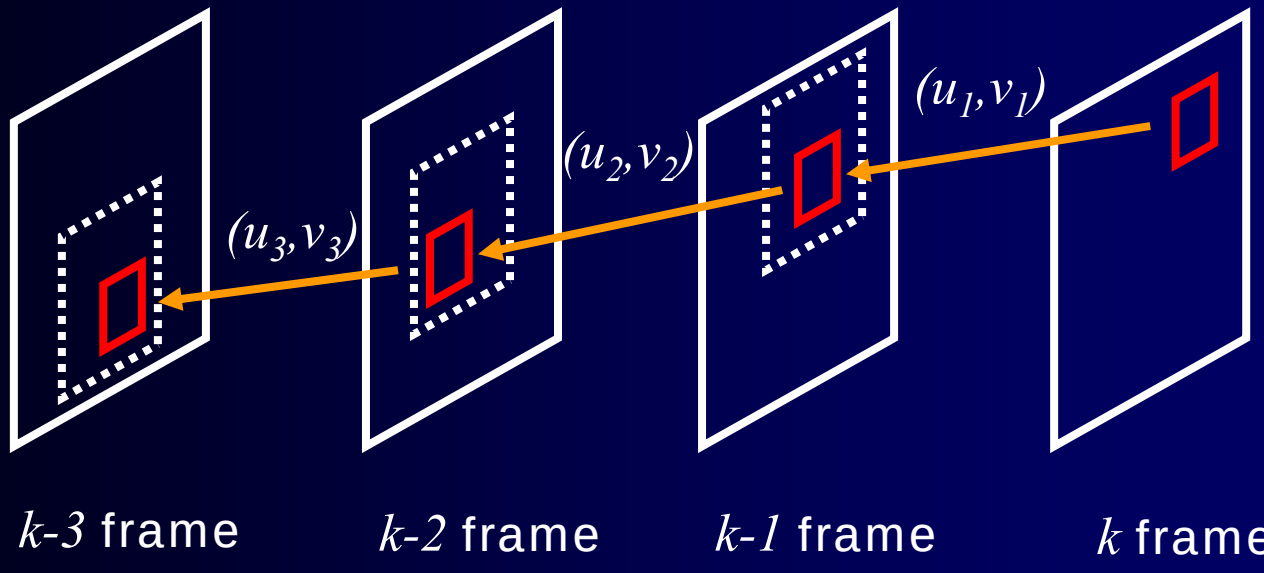
テレスコーピック探索法

- 複数フレーム先に対する動きベクトルを求める際に、直接探索するのではなく、1フレーム間隔で動きベクトルを求め、順次シフトして探索する手法



Telescopic Search Method

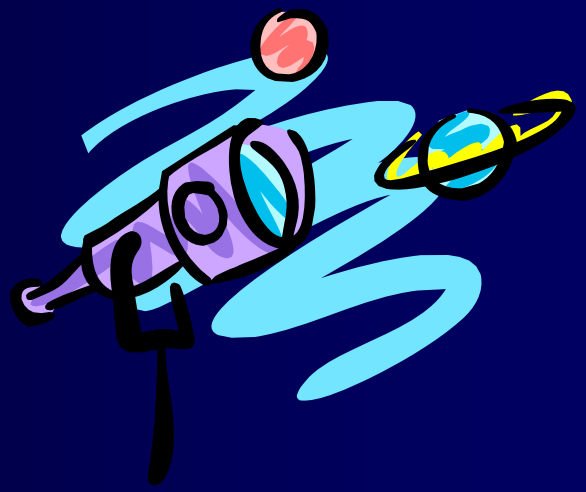
- For estimating motion vector over several frames, first obtain motion for one frame distance and then, shifting its search area based on the obtained vector frame by frame.



テレスコーピック探索法 (2)

- ベクトルの加算により、N フレーム間の動きベクトルを算出

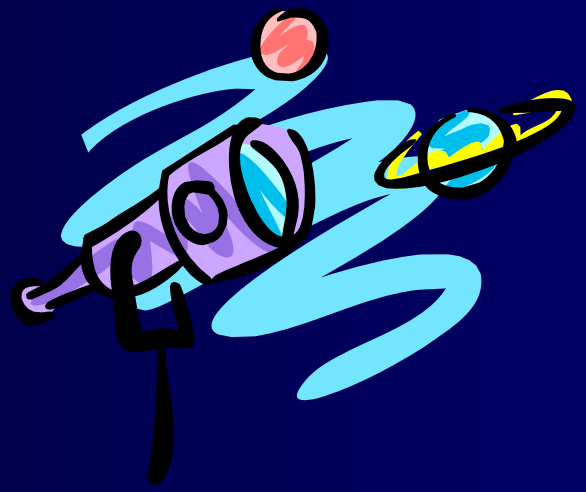
$$(u_N, v_N) = \sum_{i=1}^{N} (u_i, v_i)$$



Telescopic Search Method (2)

- By adding vectors, motion for N frame distance is obtained

$$(u_N, v_N) = \sum_{i=1}^{N} (u_i, v_i)$$



テレスコーピック探索法の特徴

- 直接Nフレーム間の動きベクトルを求めるのに比べ、探索範囲が限定できるため、演算量が少ない
- すべてのフレームをメモリに格納する必要がある
- 1段目で誤ったベクトルを選んでしまうと、誤った方向にベクトルを検索し続けることになり、正しい動きを検出できない

Characteristics of Telescopic Search Method

- Computation cost is small since search range can be limited compared with direct search between N frame distance
- Needs to store all images in the memory
- Once wrong vector is selected at the 1st stage, it continues to search in the wrong direction and cannot be recovered.

位相相関法

- フーリエ変換の位相の類似性から動きを求める手法
  - 信号 $x(n)$ の離散フーリエ変換を $X(k)$ とする

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp(-j2\pi nk / N)$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \exp(j2\pi nk / N)$$

Phase Correlation Method

- Method to detect motion from similarity of phase in Fourier transform
  - Let discrete Fourier transform of signal $x(n)$ be $X(k)$

$$X(k) = \sum_{n=0}^{N-1} x(n) \exp(-j2\pi nk / N)$$

$$x(n) = \frac{1}{N} \sum_{k=0}^{N-1} X(k) \exp(j2\pi nk / N)$$

位相相関法(2)

– 信号 $x(n)$ が m サンプルだけシフトした信号を $y(n)$ とし、その離散フーリエ変換を $Y(k)$ とする

$$
\begin{aligned}
Y(k) &= \sum_{n=0}^{N-1} y(n)\exp(-j2\pi nk/N) \\
&= \sum_{n=0}^{N-1} x(n+m)\exp(-j2\pi nk/N) \\
&= \exp(j2\pi mk/N)\sum_{s=m}^{N-1+m} x(s)\exp(-j2\pi sk/N)
\end{aligned}
$$

Phase Correlation Method(2)

– Let m sample shift signal from $x(n)$ be $y(n)$、its Fourier transform be $Y(k)$.

$$
\begin{aligned}
Y(k) &= \sum_{n=0}^{N-1} y(n)\exp(-j2\pi nk/N) \\
&= \sum_{n=0}^{N-1} x(n+m)\exp(-j2\pi nk/N) \\
&= \exp(j2\pi mk/N)\sum_{s=m}^{N-1+m} x(s)\exp(-j2\pi sk/N)
\end{aligned}
$$

位相相関法 (3)

– 信号 $y(n)$ が信号 $x(n)$ の周期関数になっていれば、$Y(k)$ は $X(k)$ と位相変化分の積で表される

$$
\begin{aligned}
Y(k) &= \exp(j2\pi mk/N)\sum_{s=m}^{N-1+m} x(s)\exp(-j2\pi sk/N) \\
&= \exp(j2\pi mk/N)\sum_{s=0}^{N-1} x(s)\exp(-j2\pi sk/N) \\
&= \exp(j2\pi mk/N)X(k)
\end{aligned}
$$

Phase Correlation Method (3)

– If signal $y(n)$ is a periodic function of signal $x(n)$, $Y(k)$ can be expressed by $X(k)$ multiplied by phase shift

$$
\begin{aligned}
Y(k) &= \exp(j2\pi mk/N)\sum_{s=m}^{N-1+m} x(s)\exp(-j2\pi sk/N) \\
&= \exp(j2\pi mk/N)\sum_{s=0}^{N-1} x(s)\exp(-j2\pi sk/N) \\
&= \exp(j2\pi mk/N)X(k)
\end{aligned}
$$

位相相関法 (4)

- 離散フーリエ変換の結果を振幅と位相の項に分離して次のように表現する

$$X(k) \equiv |X(k)| \exp(jP_x)$$
$$Y(k) \equiv |Y(k)| \exp(jP_y)$$

したがって

$$\begin{aligned} Y(k) &= |Y(k)| \exp(jP_y) \\ &= \exp(j2\pi mk/N)\,|X(k)| \exp(jP_x) \\ &= |X(k)| \exp(jP_x + j2\pi mk/N) \end{aligned}$$

Phase Correlation Method (4)

- Express discrete Fourier transform by separating amplitude and phase term.

$$X(k) \equiv |X(k)| \exp(jP_x)$$
$$Y(k) \equiv |Y(k)| \exp(jP_y)$$

Thus,

$$\begin{aligned} Y(k) &= |Y(k)| \exp(jP_y) \\ &= \exp(j2\pi mk/N)|X(k)| \exp(jP_x) \\ &= |X(k)| \exp(jP_x + j2\pi mk/N) \end{aligned}$$

位相相関法 (5)

- 離散フーリエ変換の位相差だけを逆変換する

$$
\begin{aligned}
d(n) &= F^{-1}(\exp(jP_x) - \exp(jP_y)) \\
&= \frac{1}{N}\sum_{k=0}^{N-1} \exp(-j2\pi mk/N)\exp(j2\pi nk/N) \\
&= \frac{1}{N}\sum_{k=0}^{N-1} \exp(j2\pi(n-m)k/N) \\
&= \begin{cases} 1 & n = m \\ 0 & n \neq m \end{cases}
\end{aligned}
$$

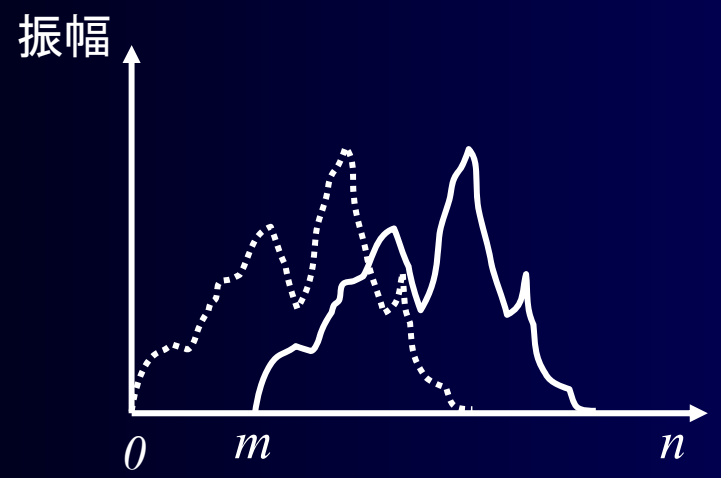
Phase Correlation Method (5)

- Only the phase difference is inverse transformed.

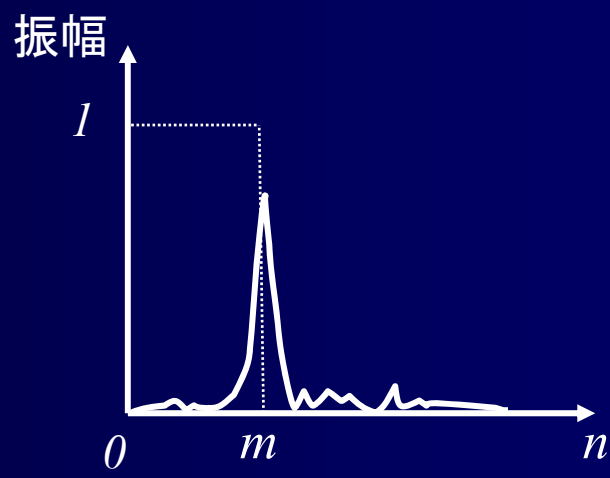
$$
\begin{aligned}
d(n) &= F^{-1}(\exp(jP_x) - \exp(jP_y)) \\
&= \frac{1}{N}\sum_{k=0}^{N-1} \exp(-j2\pi mk/N)\exp(j2\pi nk/N) \\
&= \frac{1}{N}\sum_{k=0}^{N-1} \exp(j2\pi(n-m)k/N) \\
&= \begin{cases} 1 & n = m \\ 0 & n \neq m \end{cases}
\end{aligned}
$$

位相相関法 (6)

- 信号のシフト→位相差のフーリエ逆変換はインパルス



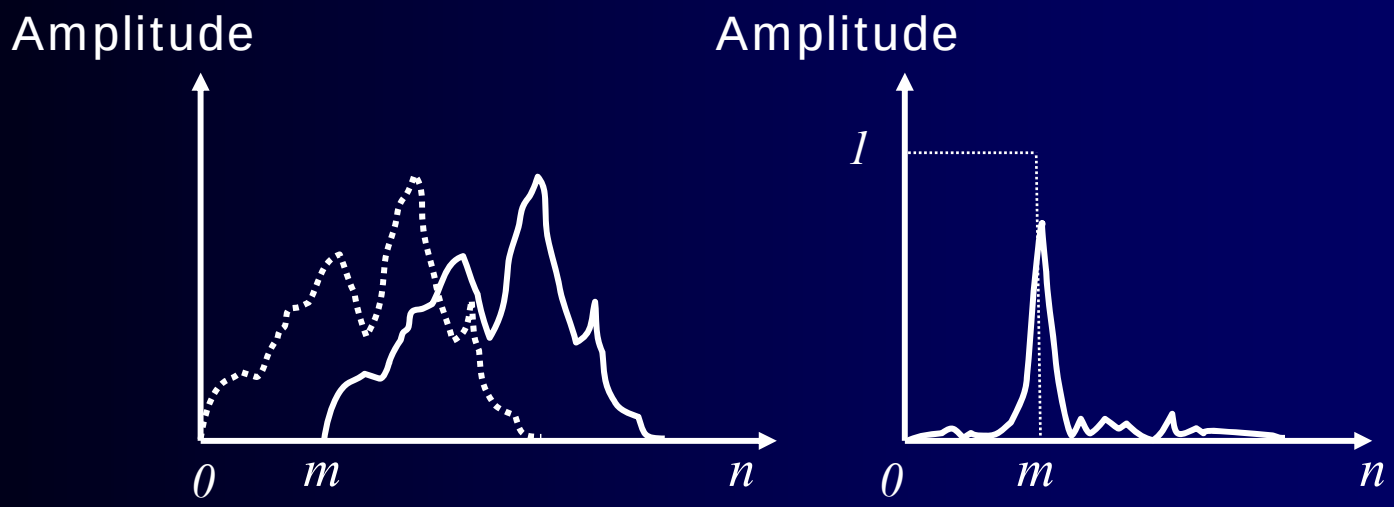
入力信号



位相差のフーリエ逆変換

Phase Correlation Method (6)

- Shifted signal → Inverse Fourier transform of phase difference is impulse



input signal

Inverse transform of phase difference

位相相関法の特徴

- 2つの信号の位相差を計算し、フーリエ逆変換することにより、動き量に対応した位置にインパルスを持つ信号が得られる
- 通常、得られた信号にはインパルスが複数個出現するため、検定が必要になる
- 1回の処理で、複数の動きを検出することができる
- 輝度変化やコントラスト変化に関係なく検出できる
- 信号の周期性の仮定が崩れると、検出精度が低下する
- 演算量が多い(特に2次元の場合）

Characteristics of Phase Correlation Method

- Signal having impulse at the location of motion by calculating inverse Fourier transform for phase difference of two signals
- Normally, some decision is necessary since multiple number of impulse appear in the calculated signal
- Multiple motion can be detected by one operation
- Motion can be detected regardless of brightness/contrast change
- Accuracy decreases when signal is not periodic
- High computational cost (especially for 2D case)

動きベクトル内挿

- 1画素単位以下の動きベクトルを検出
- Sub-pixel 動き補償に用いる
- 動きベクトルをブロック単位で検出しておき、隣接ブロックの動きベクトルを用いて画素単位でベクトルを内挿する手法

Motion Vector Interpolation

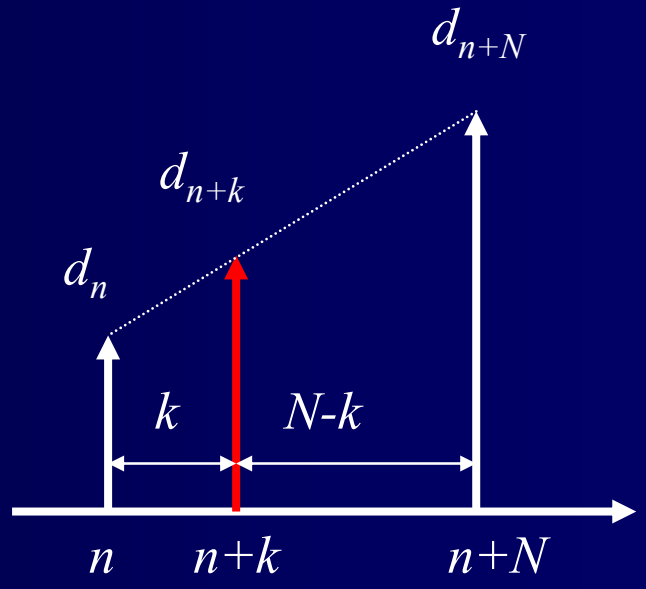
- Detect motion vector less than 1 pel accuracy
- Used for sub-pixel motion compensation
- First detect motion vector at block basis, then interpolate vector at pixel unit using neighborhood block’s motion vector

動きベクトル内挿 (2)

- 画素位置 n における動きベクトルを d_n, 画素位置 $n+N$ における動きベクトルを d_{n+N} とすると、それらの間にある画素位置 $n+k$ の動きベクトル d_{n+k} は次式で計算できる

$$d_{n+k} = \frac{N-k}{N} d_n + \frac{k}{N} d_{n+N}$$

$$= d_n + \frac{k}{N}(d_{n+N} - d_n)$$

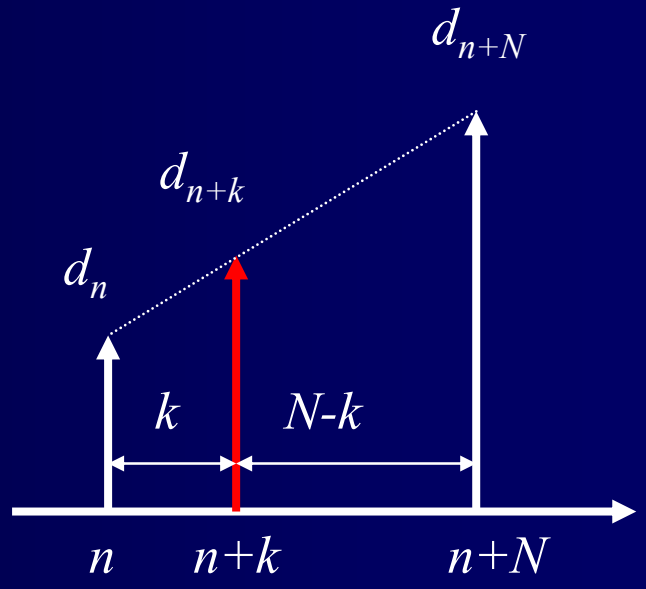


Motion Vector Interpolation (2)

- Let the motion vector at location n and $n+N$ be d_n and d_{n+N}. Motion vector d_{n+k} at location $n+k$ のbetween these points can be obtained as follows.

$$d_{n+k} = \frac{N-k}{N} d_n + \frac{k}{N} d_{n+N}$$

$$= d_n + \frac{k}{N}(d_{n+N} - d_n)$$



アフィン変換

- アフィン(Affine)変換では画素の移動を6個の射影パラメータで表す
- $x(m, n)$ の画素が $x(m', n')$ に移動する場合、座標 (m, n) の変化を以下のように表す

$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

  - 平行移動の場合、$a=d=1,\ b=c=0$ であり、動きベクトルは (e, f) で与えられる

$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} m+e \\ n+f \end{bmatrix}$$

Affine Transform

- Affine transform denotes pel's motion by 6 projection parameters
- When pels at $x(m, n)$ moves to $x(m', n')$, change of location (m, n) is written as

$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} a & b \\ c & d \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix}$$

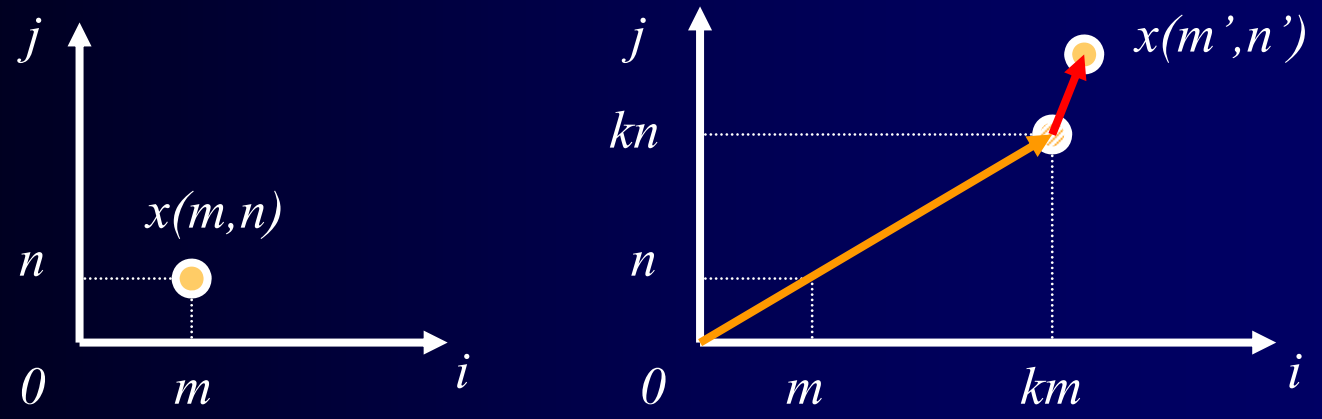
  - For parallel motion, $a=d=1, b=c=0$, and motion vectors are (e, f)

$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} m+e \\ n+f \end{bmatrix}$$

アフィン変換 (2)

- k 倍にズームアップされ、さらに平行移動が加わった場合、$a=d=k,\ b=c=0$ であり、動きベクトルは $((k-1)m+e,(k-1)n+f)$ で与えられる

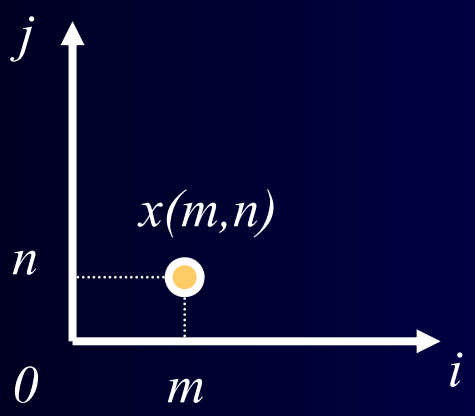
$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} km+e \\ kn+f \end{bmatrix}$$

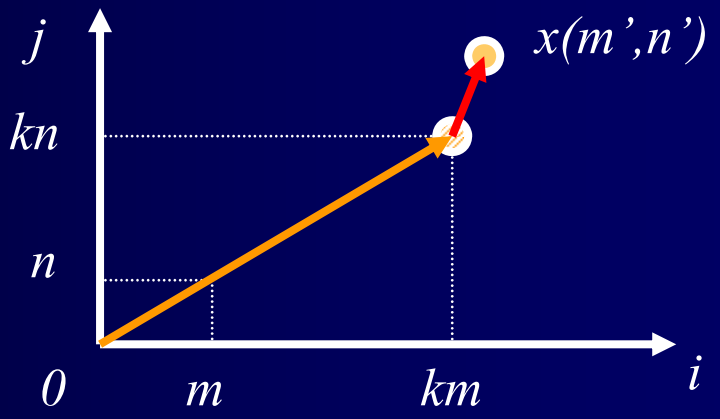


Affine Transform (2)

– For k times zoom up plus parallel motion $a=d=k$, $b=c=0$ and motion vectors are $((k-1)m+e,(k-1)n+f)$

$$\begin{bmatrix} m' \\ n' \end{bmatrix} = \begin{bmatrix} k & 0 \\ 0 & k \end{bmatrix} \begin{bmatrix} m \\ n \end{bmatrix} + \begin{bmatrix} e \\ f \end{bmatrix} = \begin{bmatrix} km+e \\ kn+f \end{bmatrix}$$





アフィン変換 (3)

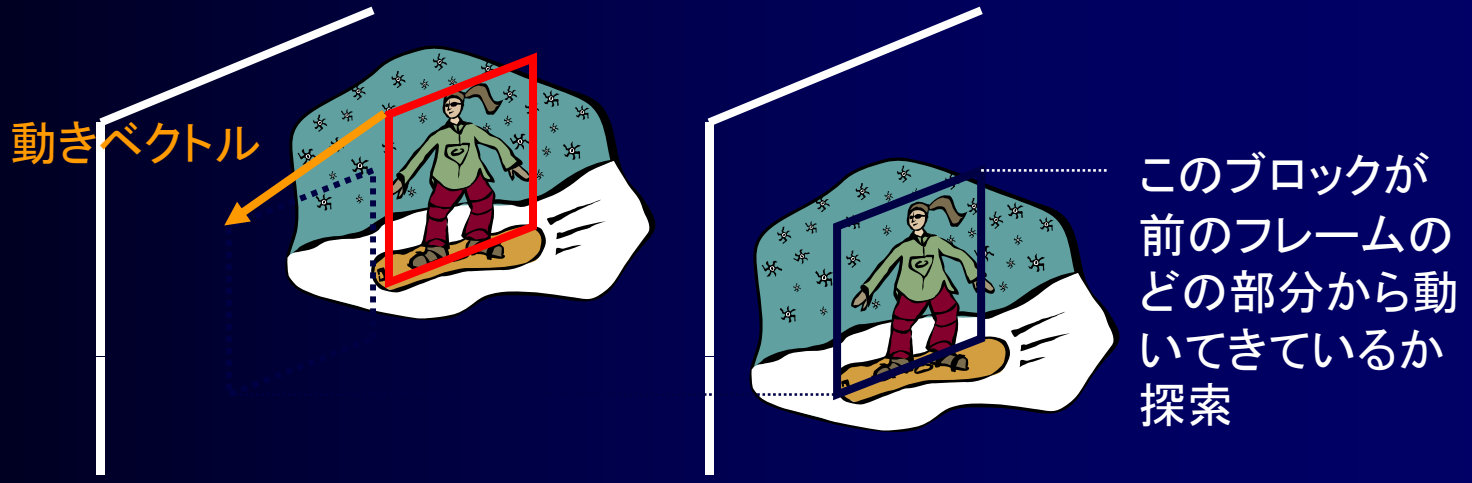
- アフィン変換を用いた動き補償の特徴
  - フレーム全体を動き補償 (グローバル動き補償, MPEG-4)
  - 6個のパラメータによりパン/ズーム補償が可能
  - グローバル動き補償 (フレーム単位) と、ローカル動き補償 (マクロブロック単位) の組み合わせにより、個別の動きにも対処

Affine Transform (3)

- Characteristics of motion compensation using affine transform
  - motion compensation to the overall frame (global motion compensation at MPEG-4)
  - panning and zooming can be represented by 6 parameters
  - combination of global motion compensation (frame basis) and local motion compensation (macroblock basis) can respond to complicated motion

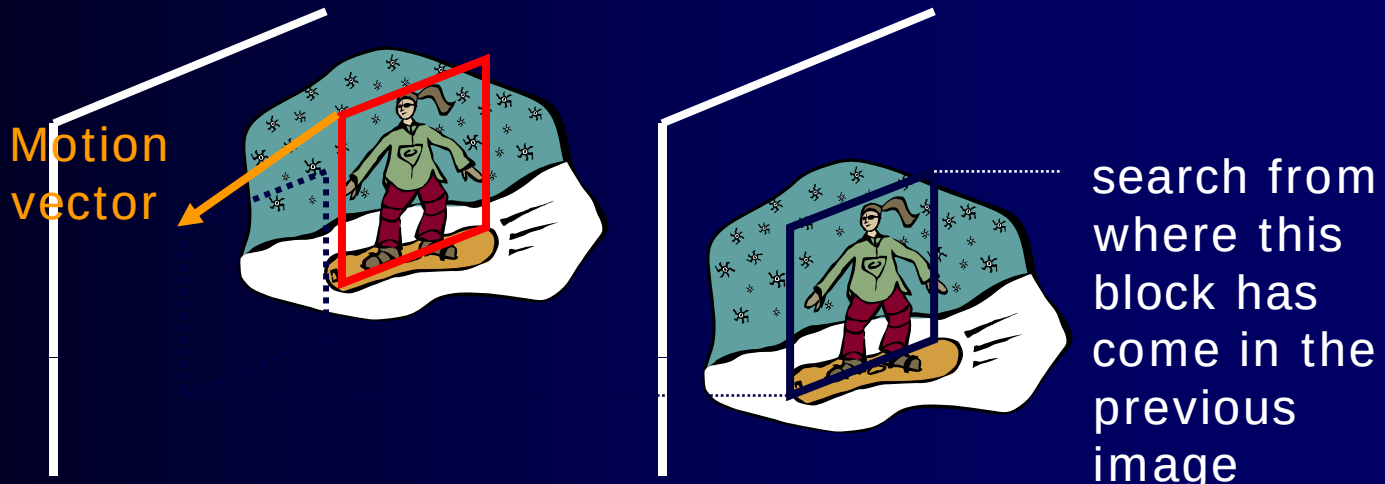
動き補償フレーム間予測

- ブロック単位の動き補償フレーム間予測
  - ブロック単位に画像を平行移動させ、次のフレームの画素を予測



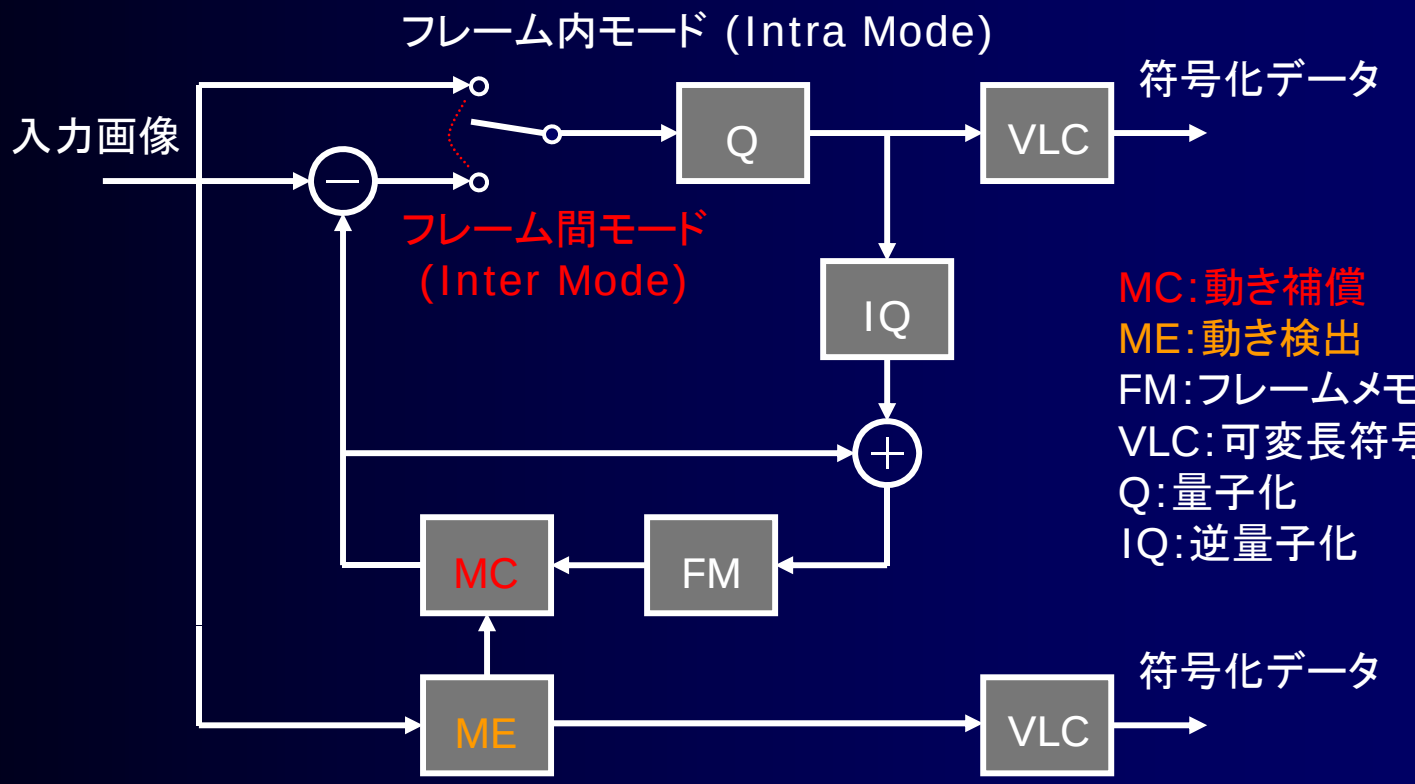
Motion Compensated Interframe Prediction

- Block basis motion compensated interframe prediction
  - Prediction for the next frame image by shifting block image



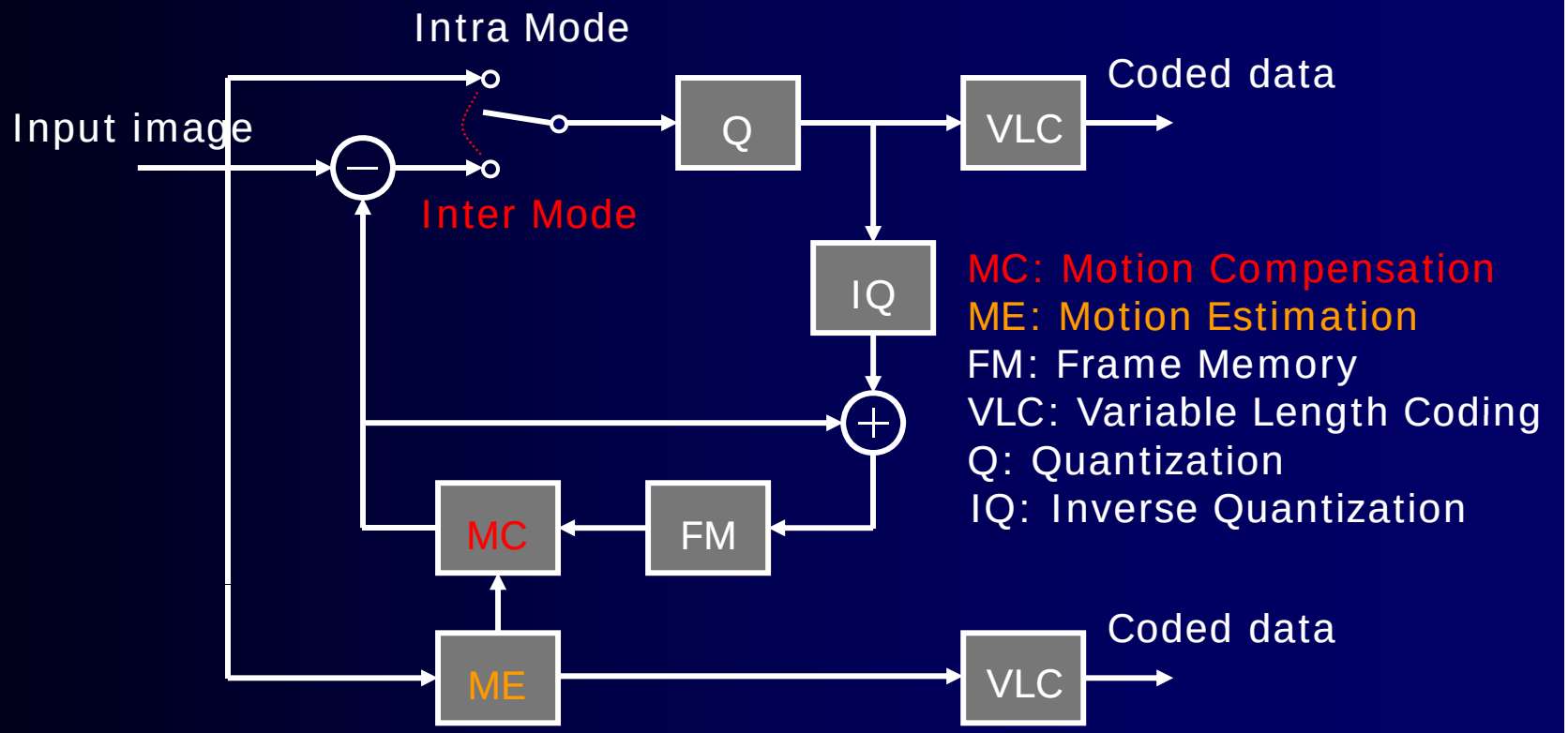
動き補償フレーム間予測 (2)

- 符号化器 (Encoder) の構成



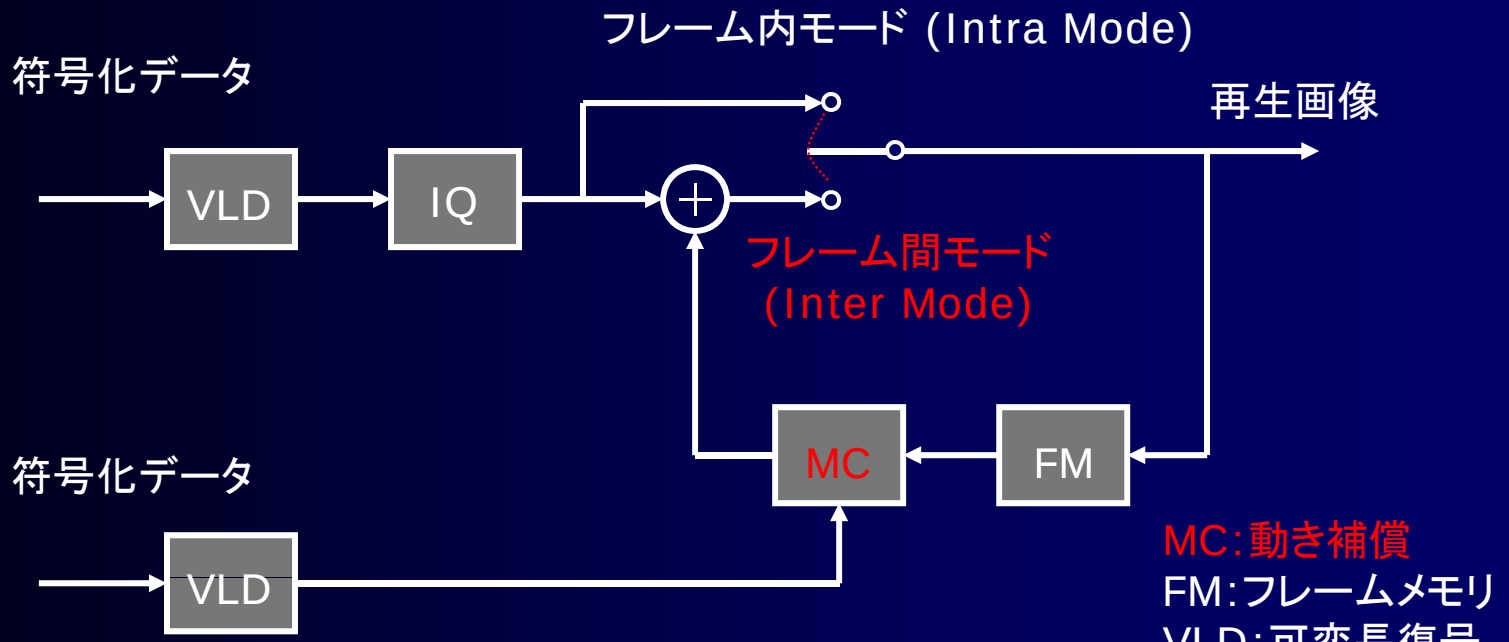
Motion Compensated Interframe Prediction (2)

- Encoder Structure



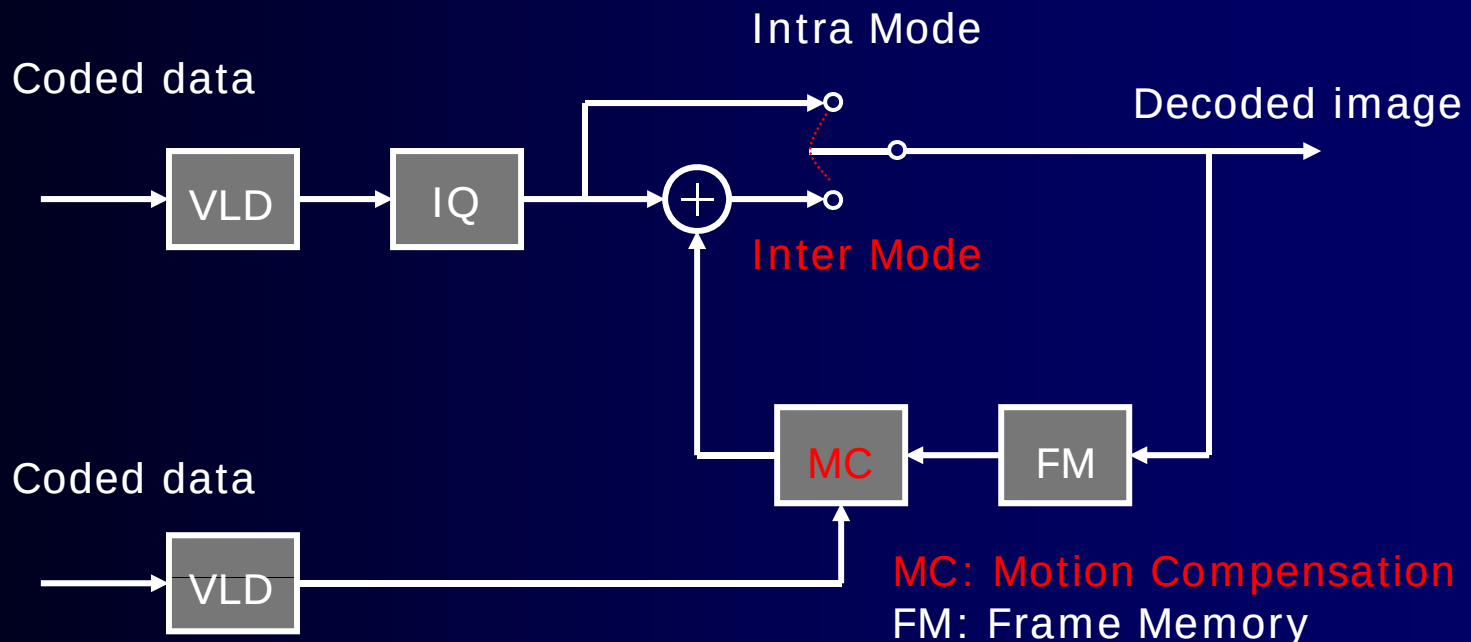
動き補償フレーム間予測 (3)

- 復号器 (Decoder) の構成



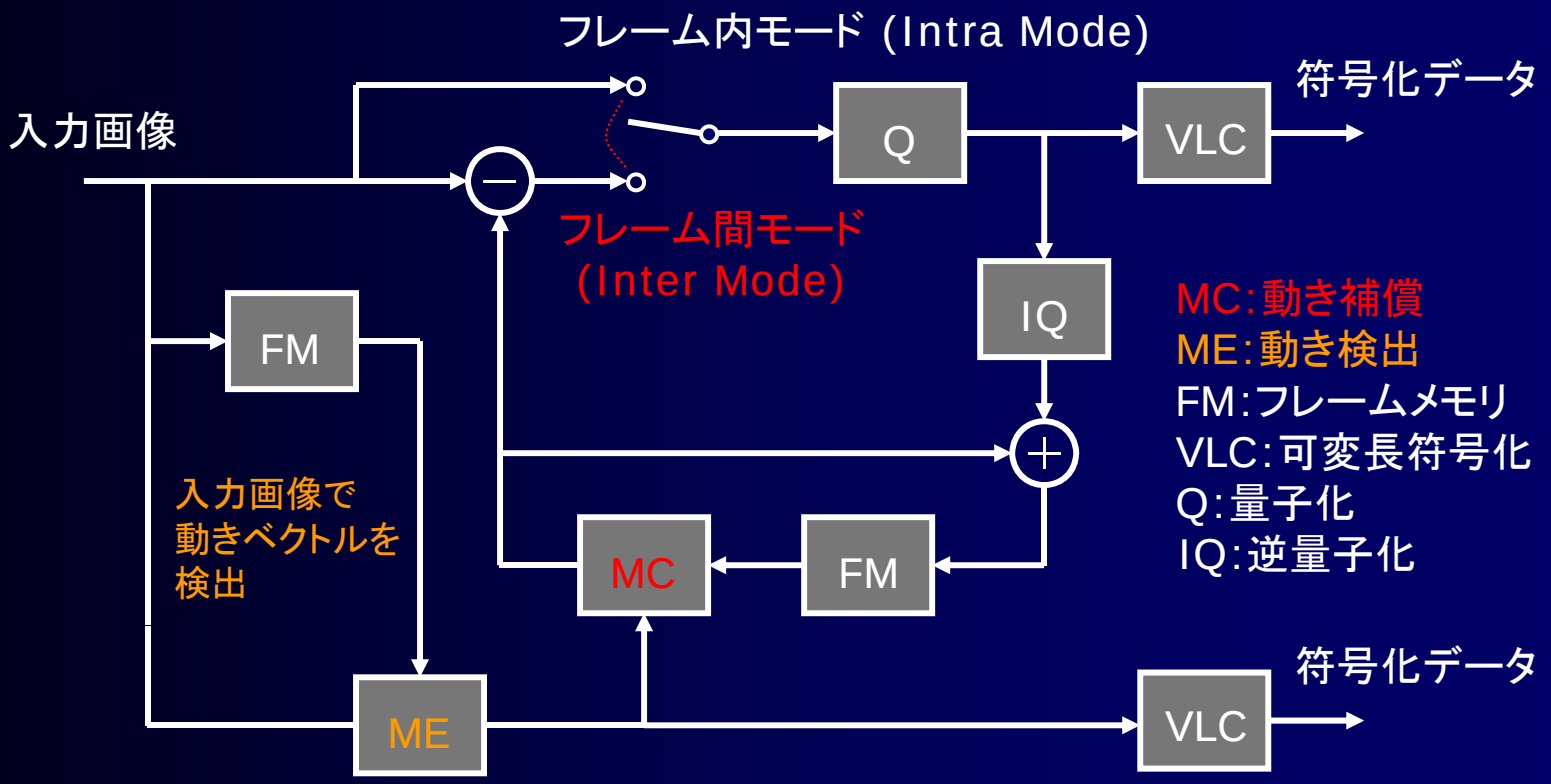
Motion Compensated Interframe Prediction (3)

- Decoder Structure



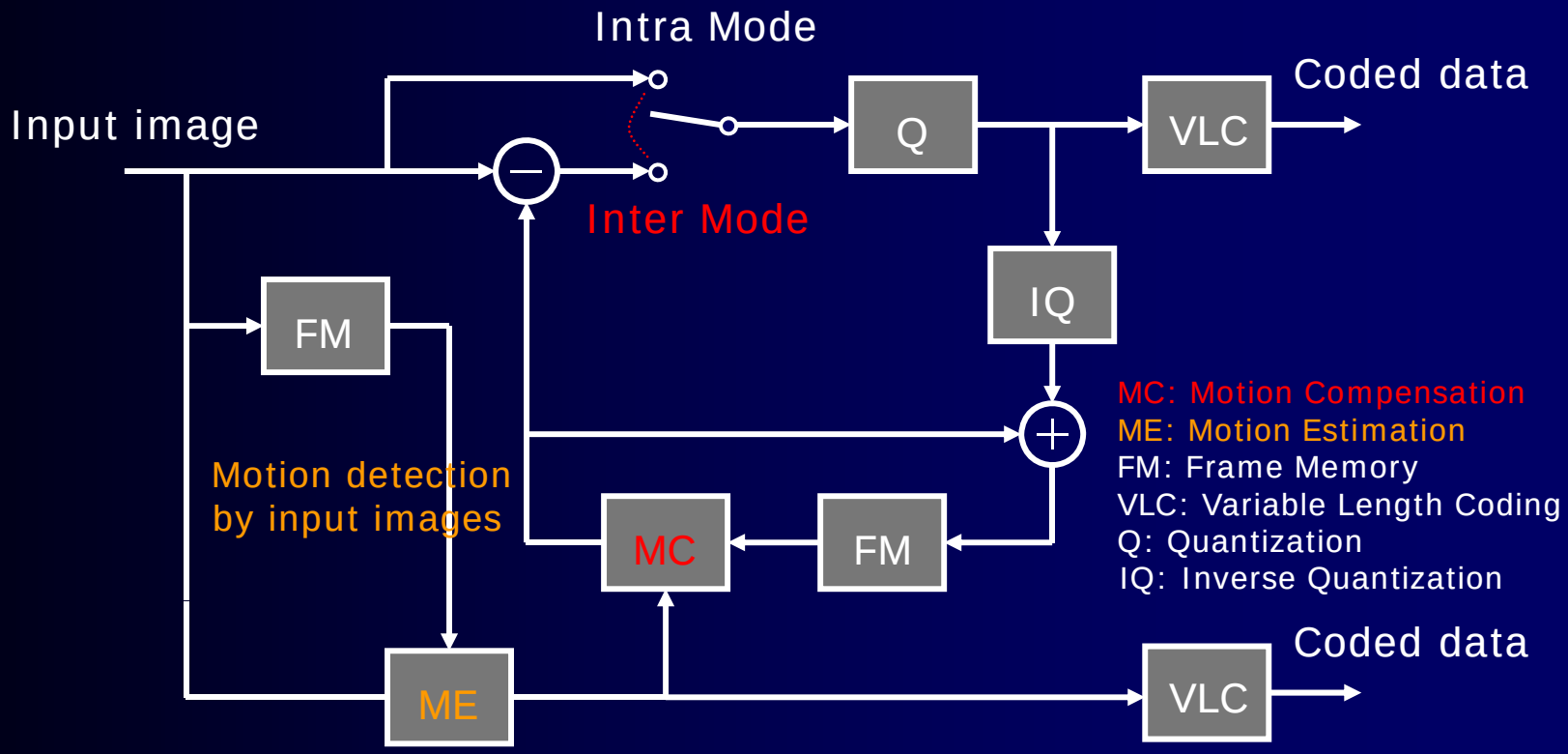
動き補償フレーム間予測 (4)

- 入力画像で動き検出する場合の符号化器の構成



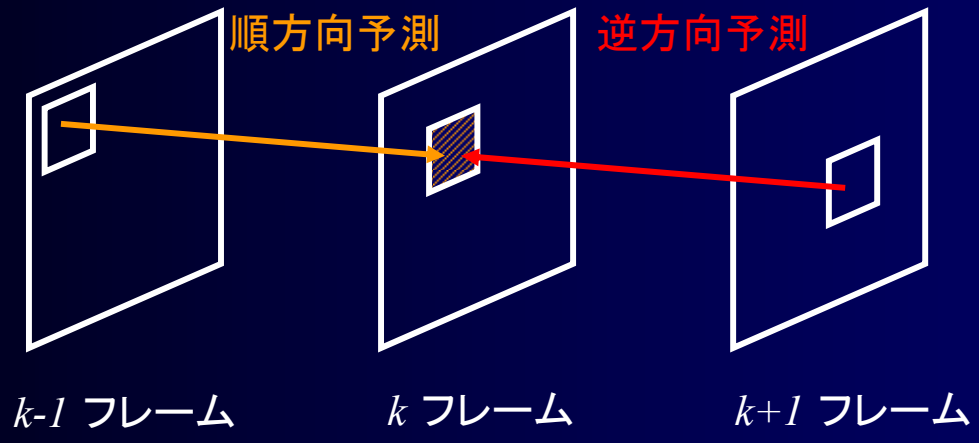
Motion Compensated Interframe Prediction (4)

- Encoder Structure when motion is detected from input image



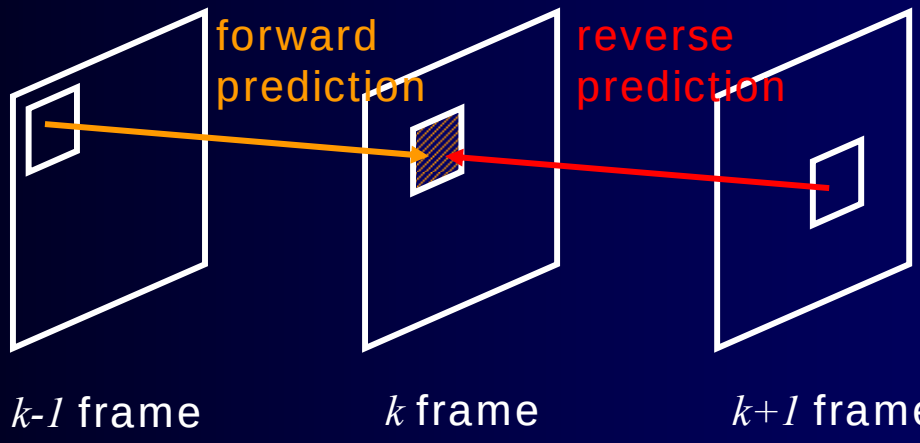
双方向予測

- 過去と未来のフレームから動き補償フレーム間予測
- 順方向の予測と逆方向の予測の組み合わせ
- Bi-directionally prediction, 両方向予測とも呼ぶ
- マクロブロック単位



Bi-directional Prediction

- Motion compensation from past and future frames
- Combination of prediction by forward and reverse prediction
- Macroblock basis



双方向予測 (2)

- 双方向の動き補償フレーム間予測は平均値で行う
- *k-1* フレームから *k* フレームへの動きベクトルを (h_f, v_f), *k+1* フレームから *k* フレームへの動きベクトルを (h_b, v_b) とすると、$x_k(m,n)$の予測画像の画素値は次式で与えられる

$$\hat{x}_k(m,n) = \frac{x_{k-1}(m+h_f, n+v_f) + x_{k+1}(m+h_b, n+v_b)}{2}$$

- 3モードの切り替え
  - 順方向予測 (Forward prediction)
  - 逆方向予測 (Backward prediction)
  - 双方向予測 (Bi-directional prediction)

Bi-directional Prediction (2)

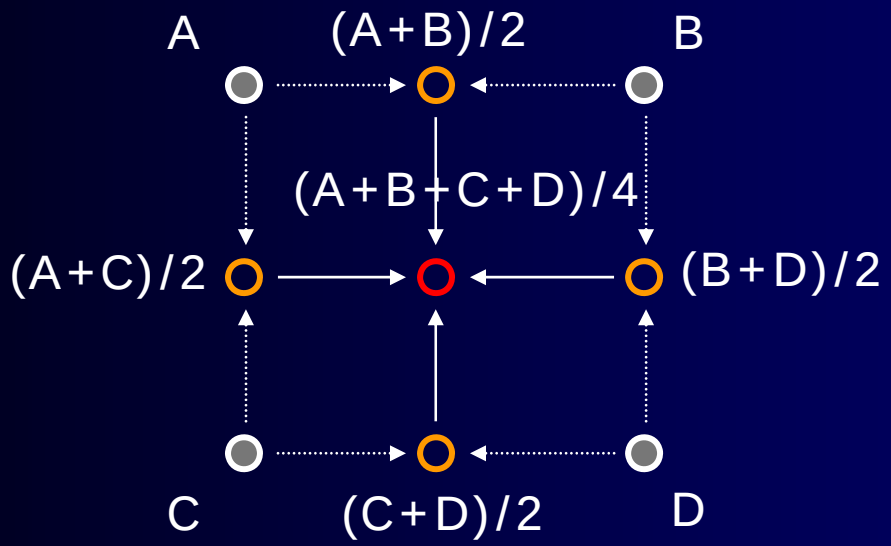
- Motion compensated bi-directional prediction employs mean
- Let motion vector from *k-1* to *k* frame and from *k+1* to *k* frame be (h_f, v_f) and (h_b, v_b), prediction value for $x_k(m,n)$ is given by

$$\hat{x}_k(m,n) = \frac{x_{k-1}(m+h_f, n+v_f) + x_{k+1}(m+h_b, n+v_b)}{2}$$

- Three modes switching
  - Forward prediction
  - Backward prediction
  - Bi-directional prediction

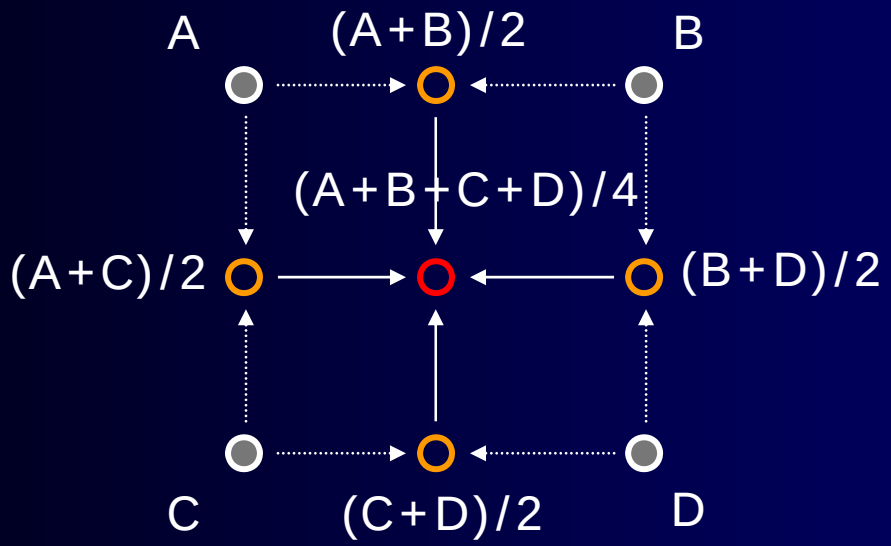
2次元半画素精度動き補償

- 0.5画素精度の動き補償 (MPEG-1)
- 水平、垂直に画素を補間



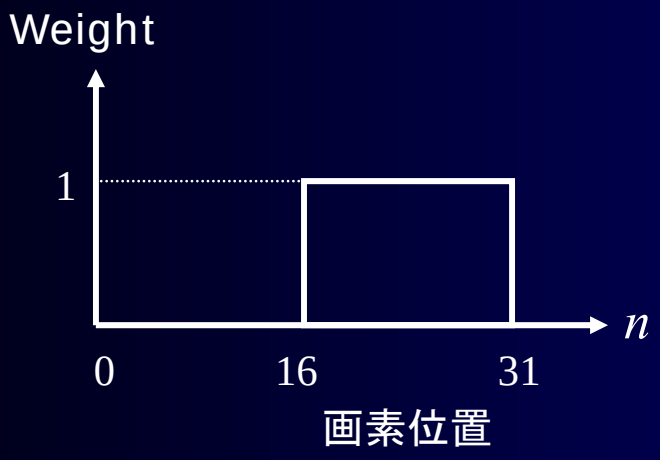
2-D Half Pixel Accuracy Motion Compensation

- 0.5 pixel accuracy motion compensation (MPEG-1)
- Interpolation pixels in horizontal and vertical direction

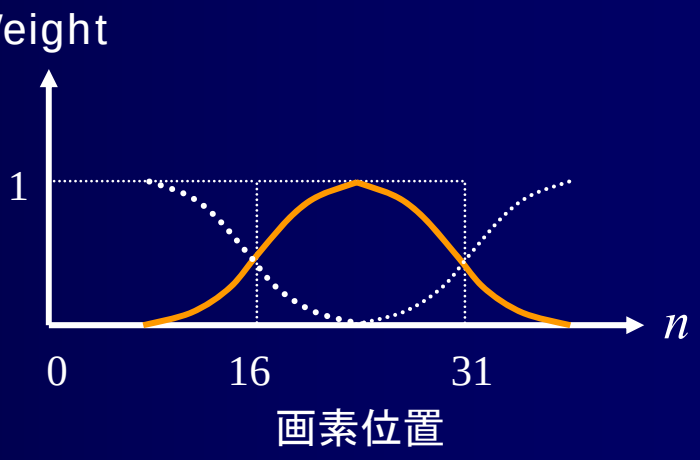


オーバーラップ動き補償

- マクロブロック単位の平行移動ではなく、隣接ブロックと重なるように、予測画素値に重みをつけて、重ね合わせる手法
- 符号化ノイズの低減に効果があり、予測効率が向上
- 低ビットレートで有効 (H.263)



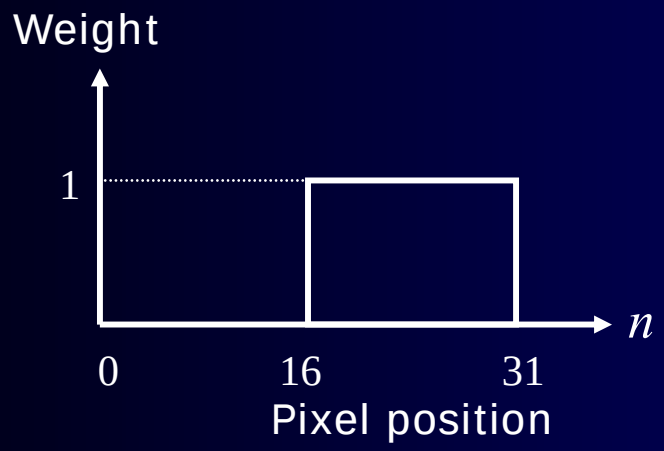
ブロック単位の動き補償



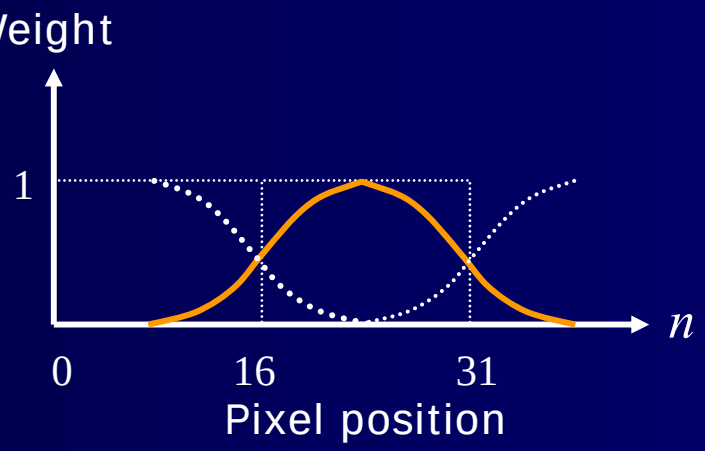
オーバーラップ動き補償

Overlapped Motion Compensation

- Unlike the conventional parallel shifting at macroblock basis, prediction block is weighted to be overlapped by data from neighboring block
- Can reduce coding noise, increase prediction efficiency
- Effective at low bit rate (H.263)



Block based motion compensation



Overlapped motion compensation

フィールド/フレーム適応動き補償

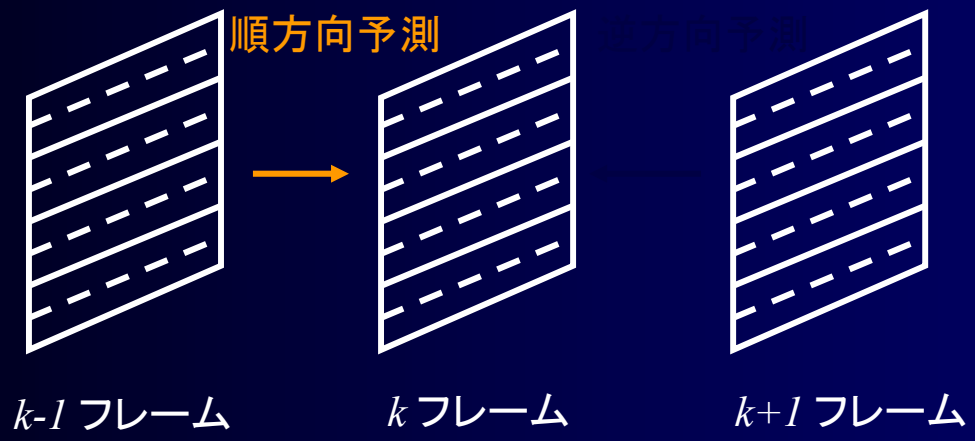
- インタレース画像の動き補償 (MPEG-2)
  - フレーム構造 (Frame structure)
    - フレーム予測 (フレームベクトルx1)
    - フィールド予測 (フィールドベクトルx2)
    - Dual-prime予測 (フィールドベクトルx1, dmv)
  - フィールド構造 (Field Structure)
    - フィールド予測 (フィールドベクトルx1)
    - フィールド16x8予測 (フィールドベクトルx2)
    - Dual-prime予測 (フィールドベクトルx1, dmv)

Field/Frame Adaptive Motion Compensation

- Motion compensation for interlaced image (MPEG-2)
  - Frame structure
    - Frame prediction (frame vector x1)
    - Field prediction (field vector x2)
    - Dual-prime prediction (field vector x1, dmv)
  - Field Structure
    - Field prediction (field vector x1)
    - Field 16x8 prediction (field vector x2)
    - Dual-prime prediction (field vector x1, dmv)

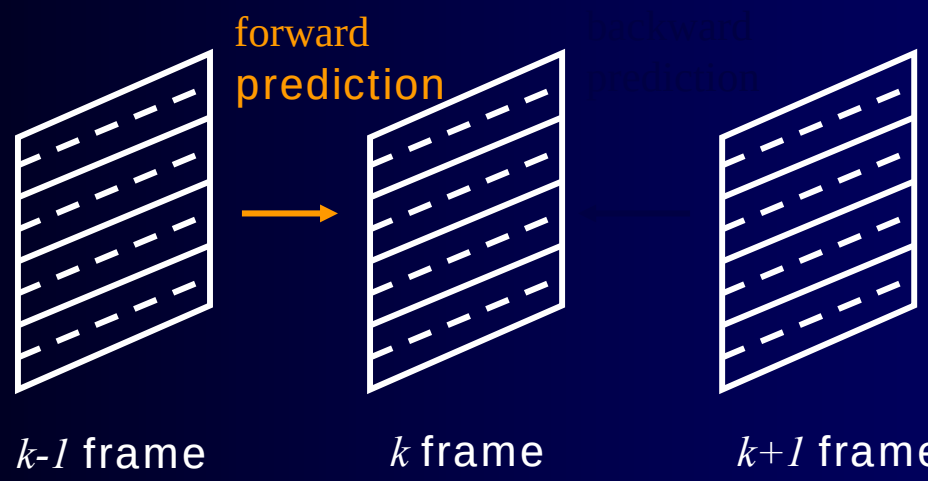
フィールド/フレーム適応動き補償 (2)

- フレーム構造 (Frame structure) における動き補償



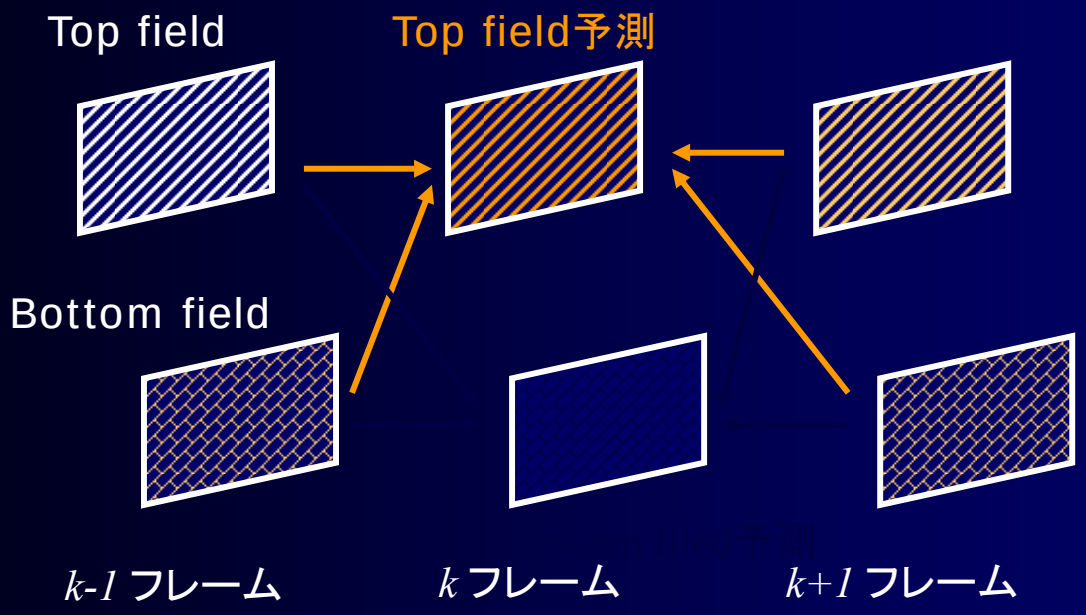
Field/Frame Adaptive Motion Compensation (2)

- Motion compensation for Frame structure



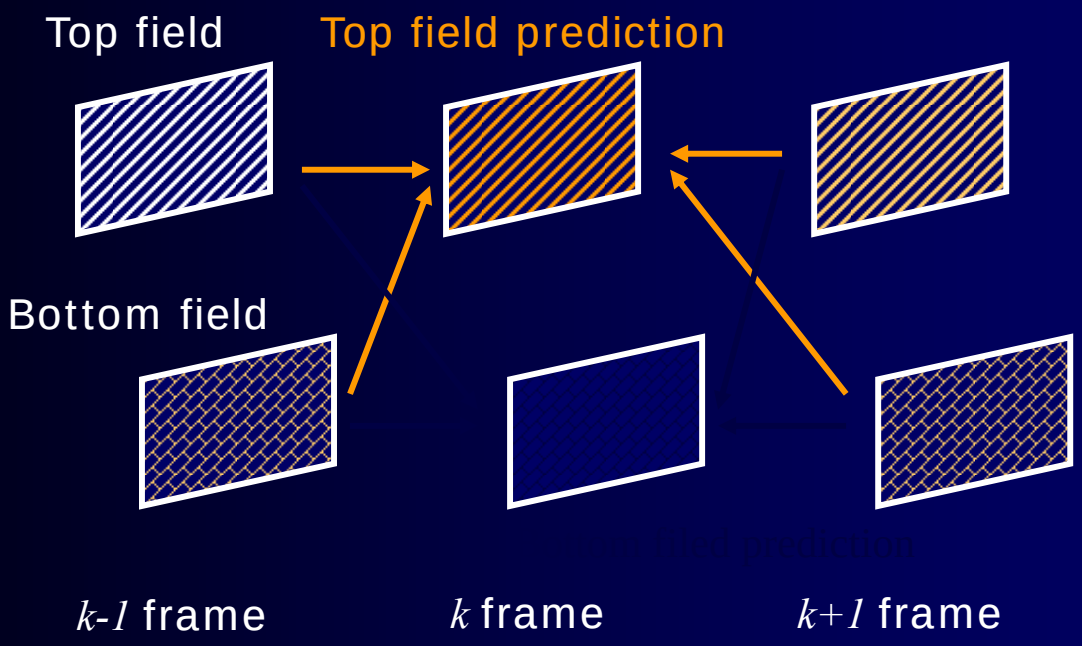
フィールド/フレーム適応動き補償 (3)

- フィールド構造 (Field structure) における動き補償



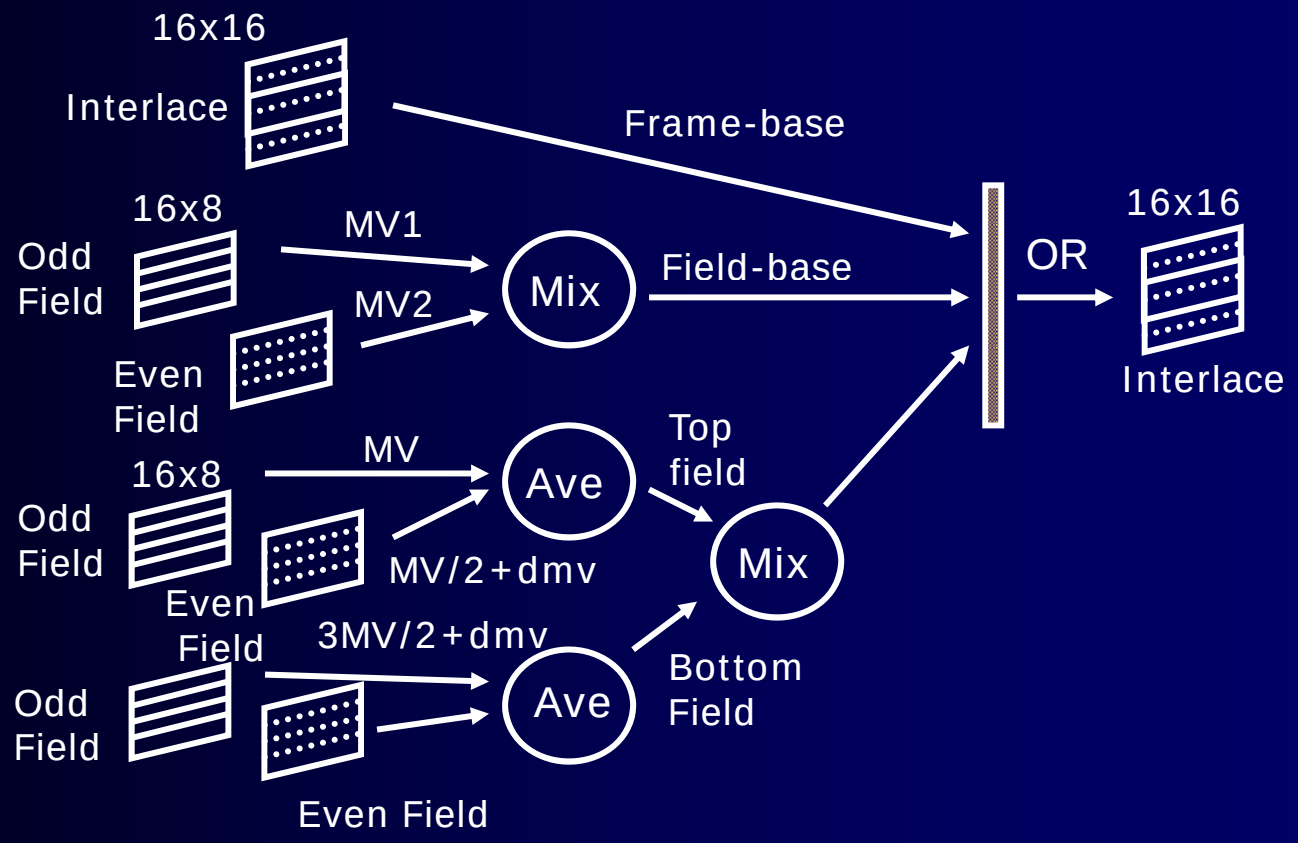
Field/Frame Adaptive Motion Compensation (3)

- Motion compensation for Field structure



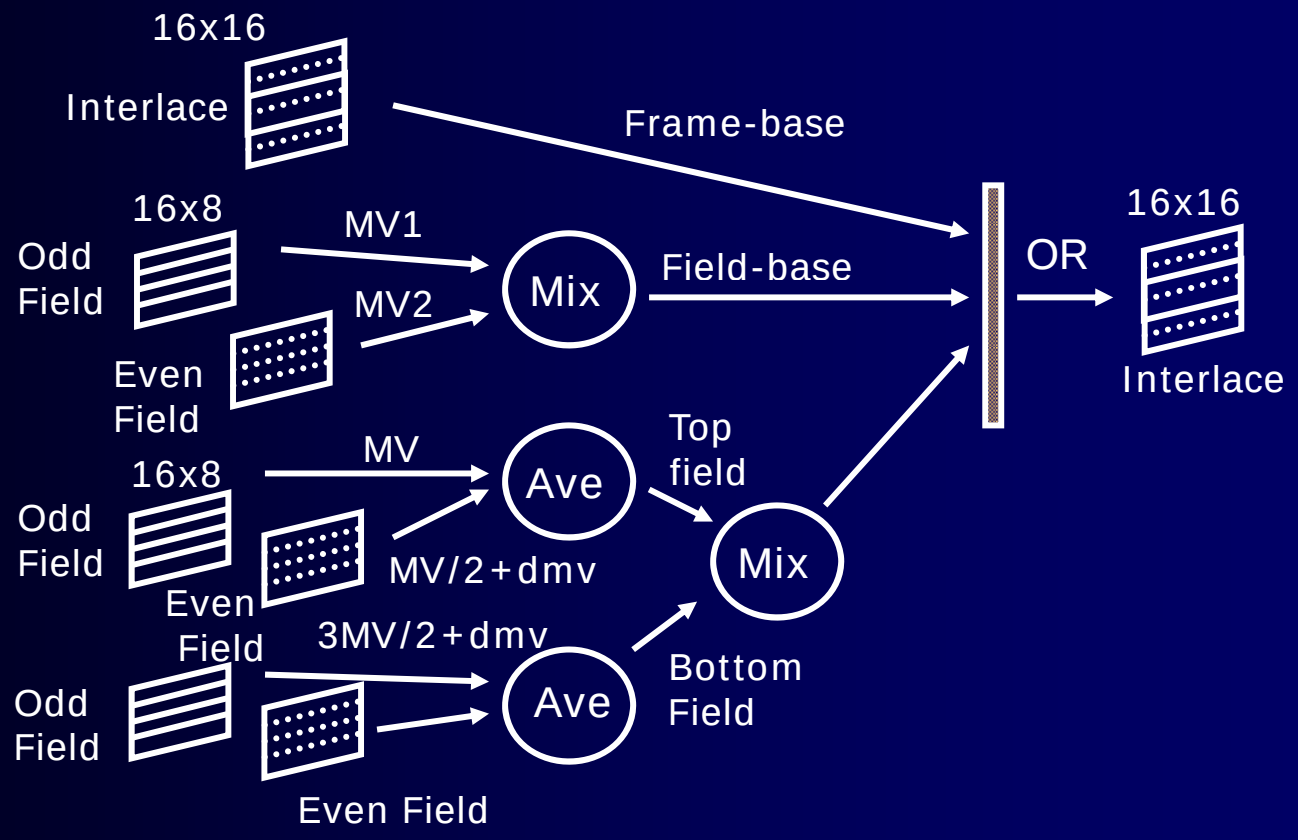
フィールド/フレーム適応動き補償（4）

- フレーム構造における予測モード



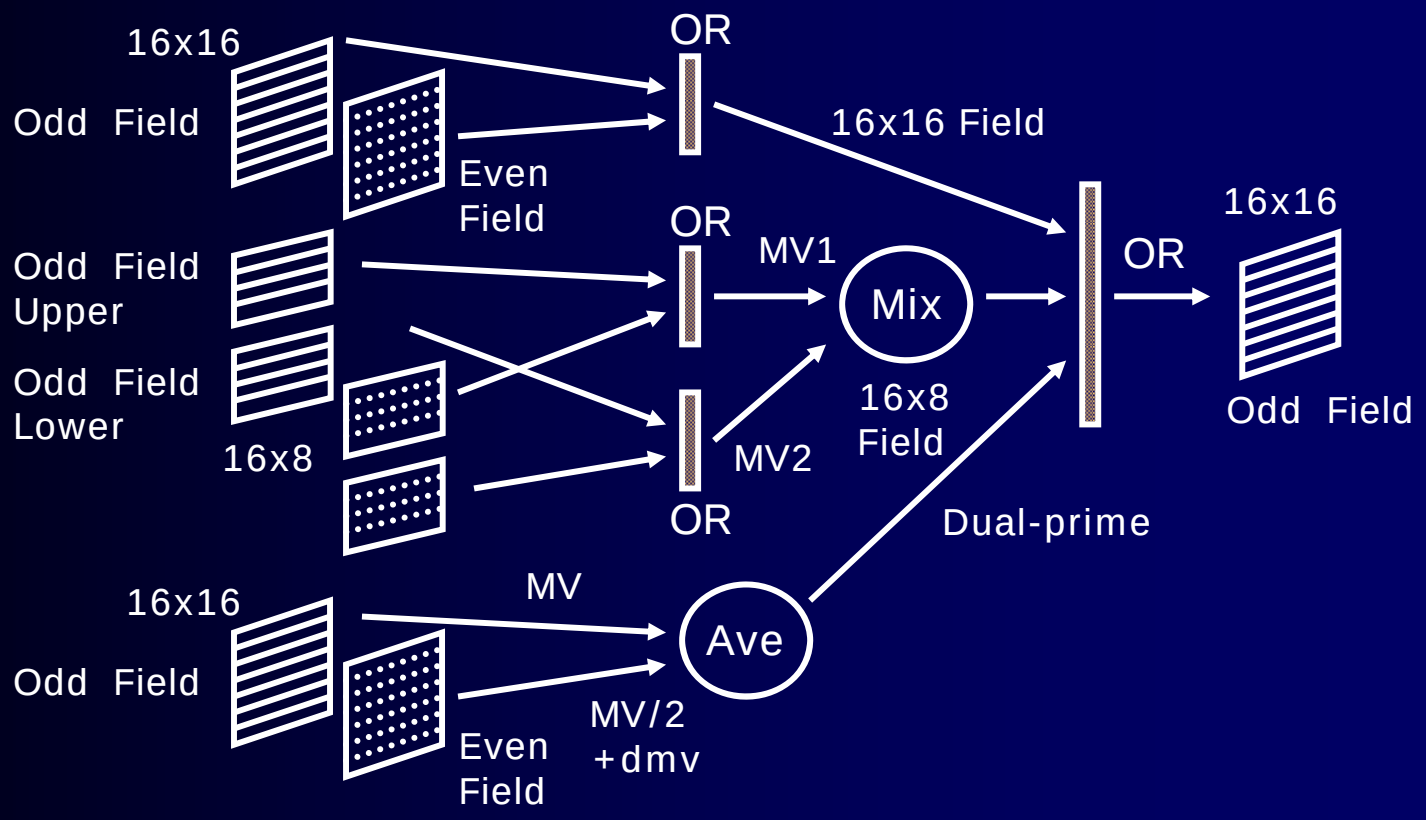
Field/Frame Adaptive Motion Compensation (4)

- Prediction mode at Frame structure



フィールド/フレーム適応動き補償（5）

- フィールド構造における予測モード



Field/Frame Adaptive Motion Compensation (5)

- Prediction mode at Field structure