Analyse de la vidéo

Chapitre 2.2 - Estimation du mouvement avancé

Dernière rév.:
20 janvier 2014

Plan de chapitre

Correspondance de bloc de taille fixe - Approche par primitive

On assume que **tous les pixels dans un voisinage près** défini par un **bloc** suivent un mouvement égal.

On recherche le mouvement pour chaque bloc séparément en assumant un **mouvement de translation** commun à tous les pixels du bloc.

On aura un algorithme :

- Rapide ;
- Calculé de façon indépendante aux blocs avoisinants ;
- Limité à un mouvement translationnel simple.

Correspondance de bloc de taille fixe

Pour chaque bloc m, la fonction à minimiser pour trouver le déplacement d^m associé au bloc est :

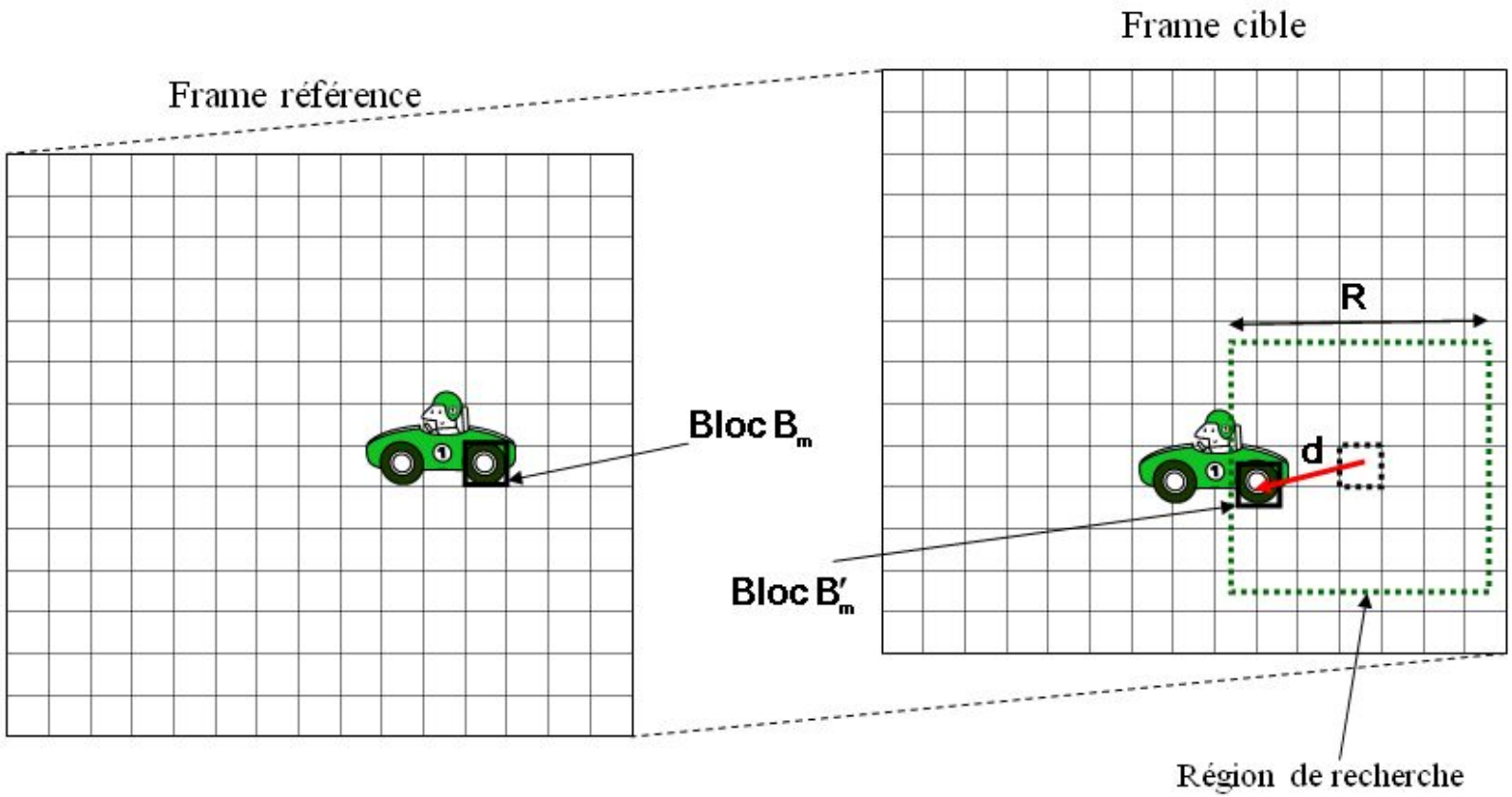
$$E_m(\mathbf{d}^m) = \sum_{(x,y)\in B_m} \left[I(x + d_x^m, y + d_y^m, t_2) - I(x, y, t_1) \right]^2 \quad (1)$$

Pour minimiser Eq.1, on peut :

- Faire une recherche exhaustive dans un voisinage ;
- Utilisez un algorithme de recherche intelligent ;
- Faire une recherche exhaustive sous-pixel.

Correspondance de bloc de taille fixe

Correspondance de bloc fixe



Correspondance de bloc de taille fixe

Sous-pixel

Les mouvements apparents ne seront pas toujours des multiples de pixel. Pour avoir une **précision sous-pixel avec une recherche exhaustive** :

- Difficulté : Un interpolation est très couteuse.
- Solution : Interpoler le image d'avance en doublant sa grosseur $\rightarrow$ Précision demi-pixel.
- Complexité : Quatre fois plus importantes qu'une recherche exhaustive normale.
- Application : On fait une recherche exhaustive normale d'abord, puis on raffine dans plus petit voisinage.

Correspondance de bloc de taille fixe

Complexité algorithmique

Supposons que la taille de la région de recherche est $(2R_0 + 1) \times (2R_0 + 1)$, R_0 étant le rayon de la région de recherche, et que la taille d'un bloc est $N \times N$, et que L et H sont la largeur et la hauteur de chaque image de la séquence.

Le nombre d'opérations pour estimer le mouvement d'un bloc est alors $N^2(2R_0 + 1)^2$. Le nombre d'opérations η requis pour estimer le mouvement de toute l'image est donné par :

$$\eta = \frac{L \cdot H}{N^2} N^2(2R_0 + 1)^2 \quad (2)$$

$$= (L \cdot H)(2R_0 + 1)^2 \quad (3)$$

Correspondance de bloc de taille fixe

Complexité algorithmique

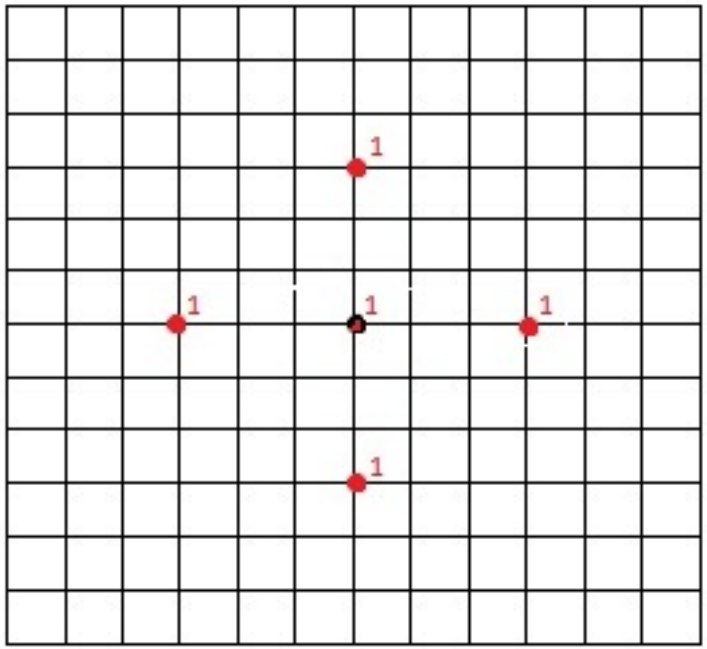
Par exemple, si $H = L = 215$ et $R_0 = 12$, alors $\eta \simeq 2.85 \cdot 10^8$ opérations, vraiment trop important pour un traitement temps réel.

On peut réduire considérablement le temps de calcul en restreignant guidant et restreignant la recherche dans la région de recherche.

Correspondance de bloc de taille fixe

RECHERCHE 2-D LOG

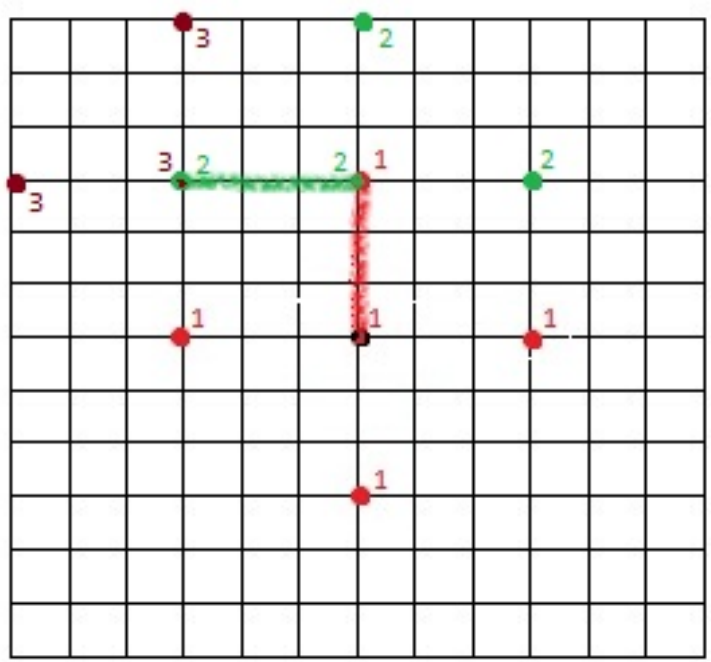
Cet algorithme commence par la position correspondant à zéro déplacement. À chaque étape de recherche, on teste 5 blocs dont les centres forment les 4 sommets d'un losange, ainsi que le centre de ce dernier. On initialise le rayon initial à la moitié du rayon R_0 de la région de recherche.



Correspondance de bloc de taille fixe

RECHERCHE 2-D LOG

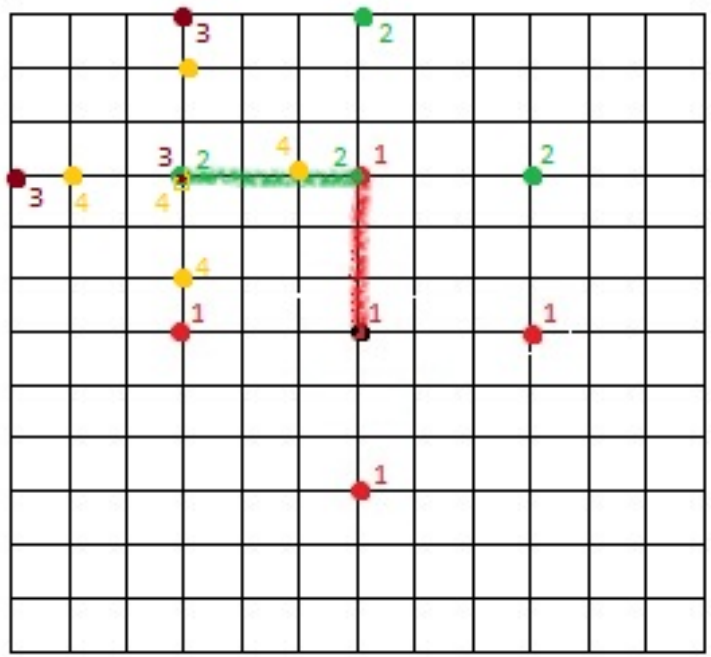
Dans la recherche suivante, on déplace le centre du losange vers le sommet du dernier losange minimisant l'erreur (1) par rapport au bloc de référence B_m.



Correspondance de bloc de taille fixe

RECHERCHE 2-D LOG

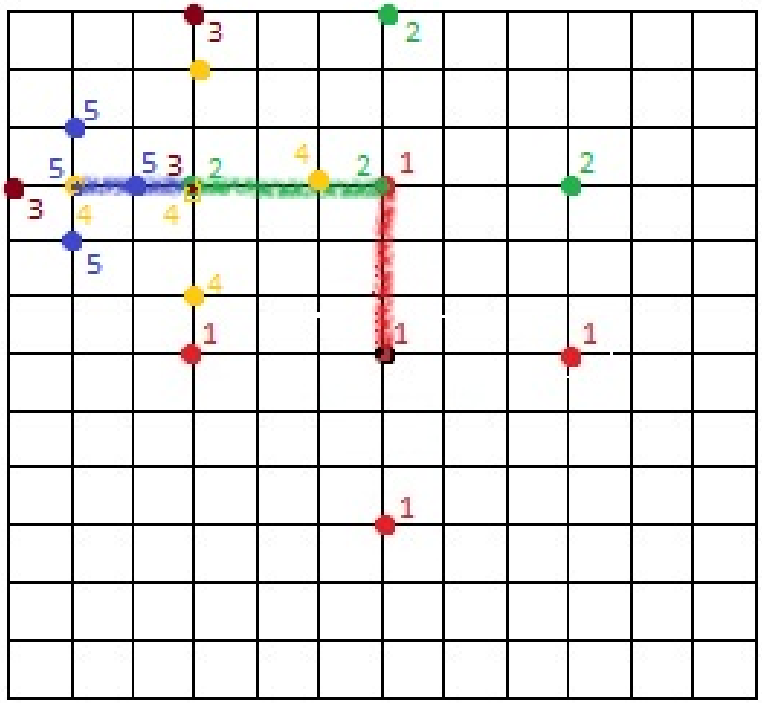
Si le meilleur bloc trouvé pour la correspondance est celui du centre du dernier losange ou la limite de la région de recherche est atteinte, on diminue le rayon de recherche de 1.

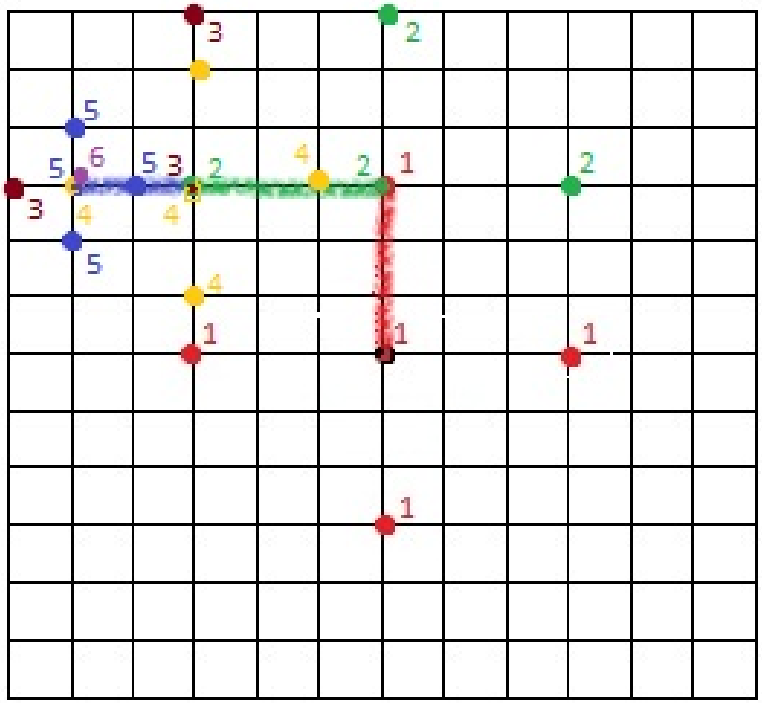


Correspondance de bloc de taille fixe

RECHERCHE 2-D LOG

On arrête lorsque le rayon de recherche est nul.





Correspondance de bloc de taille fixe

RECHERCHE 2-D LOG

Il a été démontré que la complexité de cet algorithme est logarithmique. Il est donc plus rapide que l'algorithme exhaustif.

Cependant, on ne peut pas déterminer à l'avance le nombre d'étapes nécessaires pour trouver la correspondance optimale.

Algorithme de la recherche 2D log

Entrées: $I(t)$, $I(t+1)$, R_0, i, j : Rayon de recherche initial R_0 du bloc situé à la position (i, j)

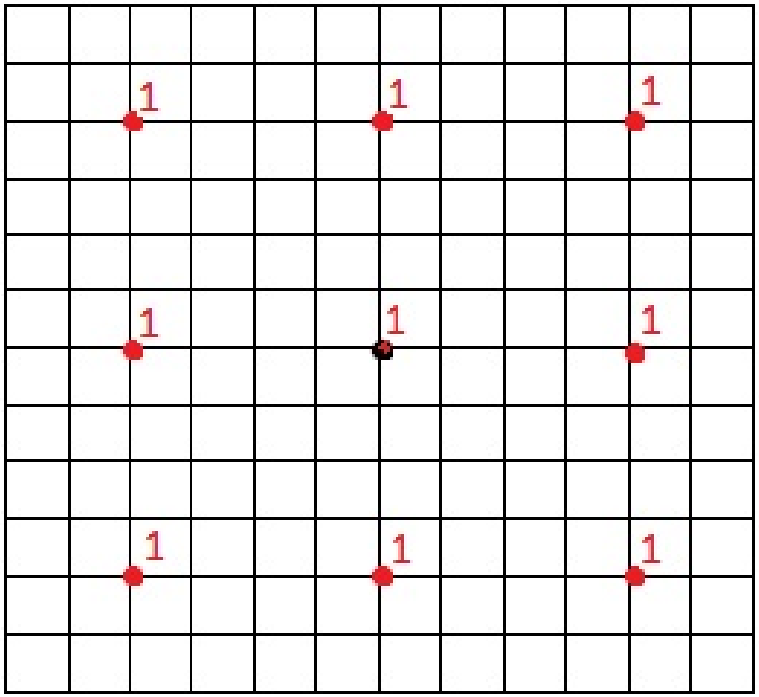
Sorties: d_x, d_y : Déplacement du bloc

$R \leftarrow R_0$, $(x, y) \leftarrow i, j$, $Bloc_m \leftarrow$ Bloc à la position (x, y) de $I(t)$;
tant que $R > 0$ **faire**
    $Bloc_c[0] \leftarrow$ Bloc à la position (x, y) de $I(t+1)$;
    $Bloc_c[1] \leftarrow$ Bloc à la position $(x, y+R)$ de $I(t+1)$;
    $Bloc_c[2] \leftarrow$ Bloc à la position $(x+R, y)$ de $I(t+1)$;
    $Bloc_c[3] \leftarrow$ Bloc à la position $(x, y-R)$ de $I(t+1)$;
    $Bloc_c[4] \leftarrow$ Bloc à la position $(x-R, y)$ de $I(t+1)$;
    pour $k \leftarrow 0$, $k <= 4$ **faire**
        $E_m[k] \leftarrow$ Calcul d'énergie par Eq.1;
    si $E_m[0]$ *est le plus petit* **alors**
        $R \leftarrow R - 1$;
    sinon si $E_m[1]$ *est le plus petit* **alors**
        $y \leftarrow y + R$;
    sinon si $E_m[2]$ *est le plus petit* **alors**
        $x \leftarrow x + R$;
    sinon si $E_m[3]$ *est le plus petit* **alors**
        $y \leftarrow y - R$;
    sinon si $E_m[4]$ *est le plus petit* **alors**
        $x \leftarrow x - R$;
retourner $(d_x, d_y) \leftarrow (i - x, j - y)$

Correspondance de bloc de taille fixe

RECHERCHE À NB D'ÉTAPES FIXES

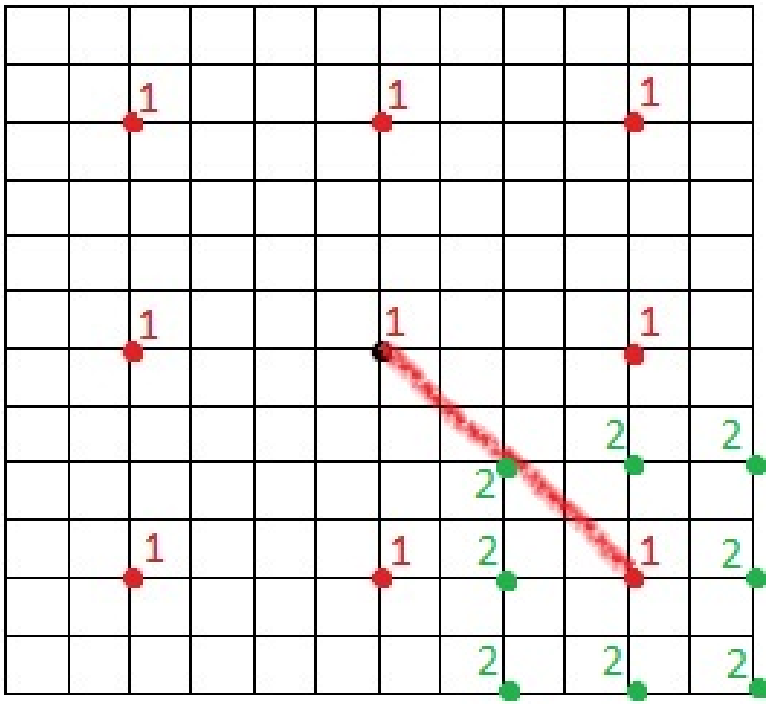
Dans cet algorithme, à chaque étape sauf la première (où on effectue 9 comparaisons), 8 blocs sont testés pour voir leur similarité par rapport au bloc de référence B_m.

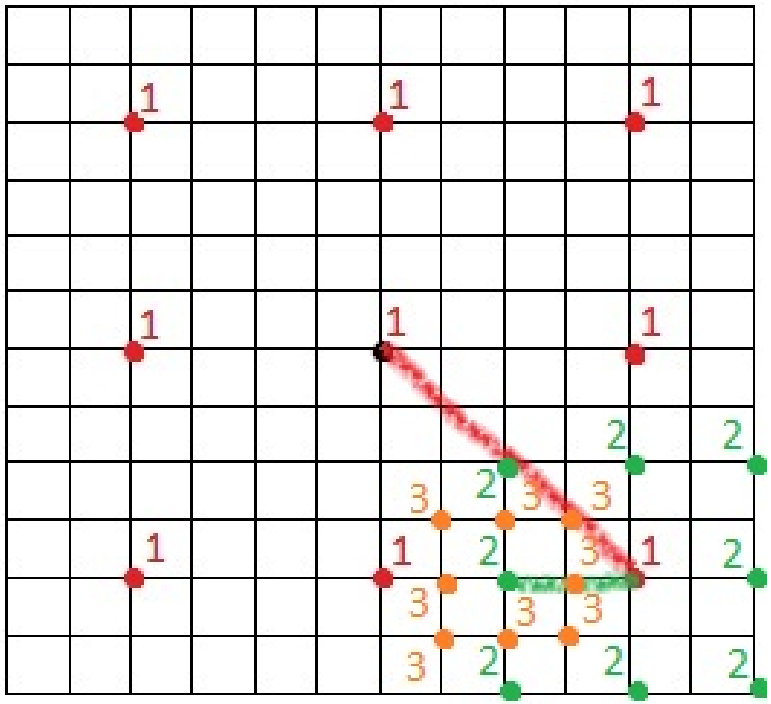


Correspondance de bloc de taille fixe

RECHERCHE À NB D'ÉTAPES FIXES

En passant d'une étape à une autre, on divise le rayon de recherche par 2, et le centre de la nouvelle région est déplacé vers le centre du bloc ayant la plus grande similarité avec le bloc de référence.





Correspondance de bloc de taille fixe

RECHERCHE À NB D'ÉTAPES FIXES

Soit R_0 le rayon de la région initiale, alors la correspondance optimale du bloc B_m est faite en $k = \lfloor \log_2(R_0 + 1) \rfloor$ étapes, où $\lfloor \cdot \rfloor$ désigne la partie entière d'un nombre.

Pour un bloc donné, le nombre total de blocs à tester avant la correspondance finale est $8k + 1$. Si par exemple, le rayon de la région de recherche est $R_0 = 32$, la méthode exhaustive requerra 4225 comparaisons de blocs, alors que l'algorithme à nombre fixe d'étapes requerra seulement 41 comparaisons.

Correspondance de bloc de taille fixe

Algorithme de calcul d'énergie des 8 blocs voisins

Entrées: $I(t), I(t+1), R_0, i, j$: Rayon de recherche initial R_0 du bloc situé à la position (i, j)

Sorties: E_m : Énergie des 8 blocs voisins

$Bloc_c[0] \leftarrow$ Bloc à la position (x, y) de $I(t+1)$;
$Bloc_c[1] \leftarrow$ Bloc à la position $(x, y+R)$ de $I(t+1)$;
$Bloc_c[2] \leftarrow$ Bloc à la position $(x+R, y)$ de $I(t+1)$;
$Bloc_c[3] \leftarrow$ Bloc à la position $(x, y-R)$ de $I(t+1)$;
$Bloc_c[4] \leftarrow$ Bloc à la position $(x-R, y)$ de $I(t+1)$;
$Bloc_c[5] \leftarrow$ Bloc à la position $(x+R, y+R)$ de $I(t+1)$;
$Bloc_c[6] \leftarrow$ Bloc à la position $(x+R, y-R)$ de $I(t+1)$;
$Bloc_c[7] \leftarrow$ Bloc à la position $(x-R, y+R)$ de $I(t+1)$;
$Bloc_c[8] \leftarrow$ Bloc à la position $(x-R, y-R)$ de $I(t+1)$;
pour $k \leftarrow 0, k <= 8$ **faire**
  $E_m[k] \leftarrow$ Calcul d'énergie par Eq.1;

retourner E_m

Algorithme de la recherche à Nb d'étapes fixe

Entrées: $I(t)$, $I(t+1)$, R_0, i, j : Rayon de recherche initial R_0 du bloc situé à la position (i, j)

Sorties: d_x, d_y : Déplacement du bloc

$R \leftarrow R_0$, $(x, y) \leftarrow i, j$, $Bloc_m \leftarrow$ Bloc à la position (x, y) de $I(t)$;
tant que $R > 0$ **faire**
    $E_m \leftarrow$ Calcul d'énergie des 8 blocs voisins (page précédente);
    si $E_m[1]$ *est le plus petit* **alors**
        $y \leftarrow y + R$;
    sinon si $E_m[2]$ *est le plus petit* **alors**
        $x \leftarrow x + R$;
    sinon si $E_m[3]$ *est le plus petit* **alors**
        $y \leftarrow y - R$;
    sinon si $E_m[4]$ *est le plus petit* **alors**
        $x \leftarrow x - R$;
    sinon si $E_m[5]$ *est le plus petit* **alors**
        $x \leftarrow x + R$, $y \leftarrow y + R$;
    sinon si $E_m[6]$ *est le plus petit* **alors**
        $x \leftarrow x + R$, $y \leftarrow y - R$;
    sinon si $E_m[7]$ *est le plus petit* **alors**
        $x \leftarrow x - R$, $y \leftarrow y + R$;
    sinon si $E_m[8]$ *est le plus petit* **alors**
        $x \leftarrow x - R$, $y \leftarrow y - R$;
    $R \leftarrow R - 1$;
retourner $(d_x, d_y) \leftarrow (i - x, j - y)$

Correspondance de bloc de taille fixe

Avantages et inconvénients

Avantages :

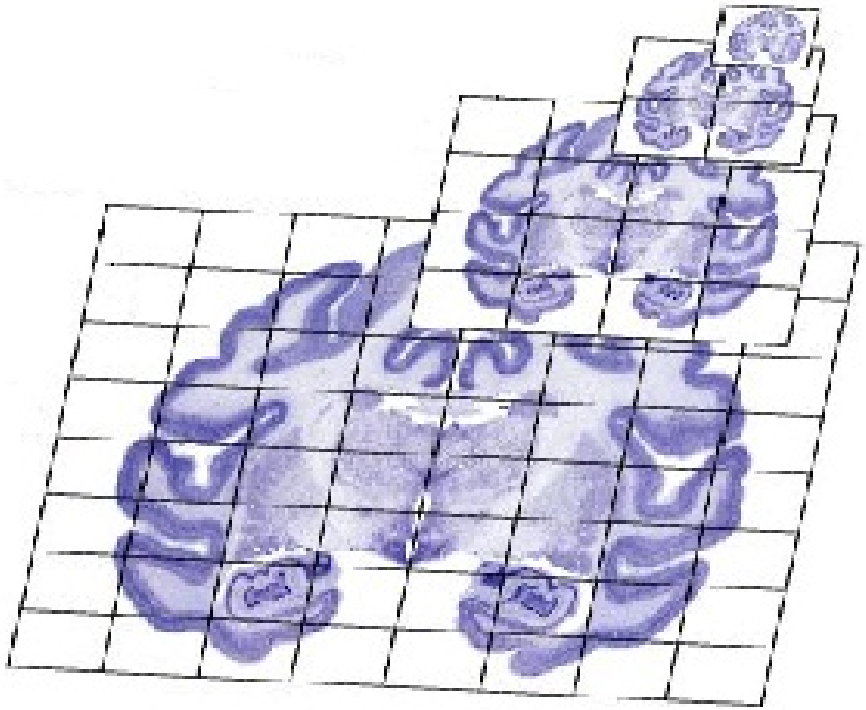
- Simplicité : Ne requiert pas de forme dérivative ;
- Convergence : Vu l'évaluation exhaustive, on s'assure que le déplacement trouvé est optimal ;
- Voisinage : On ne suppose plus nécessairement un petit voisinage (limité au rayon de recherche).

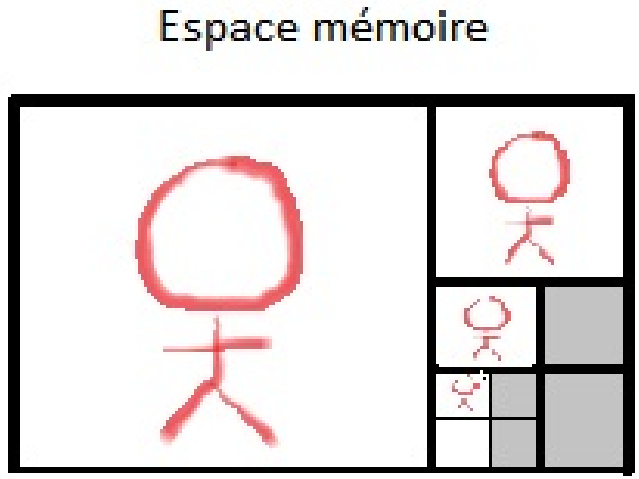
Inconvénients :

- Effet de bloc : Discontinuité aux bords des blocs ;
- Effet chaotique : Dû au "Bloc par bloc" ;

Multi-résolution

L'estimation multi-résolution du flux optique a comme principal objectif d'augmenter la robustesse d'estimation du mouvement.





Multi-résolution

En descendant à des niveaux à plus basse résolution, **on augmente l'ouverture** (un peu comme augmenter le facteur de lissage), puisqu'on mélange l'information du voisinage.

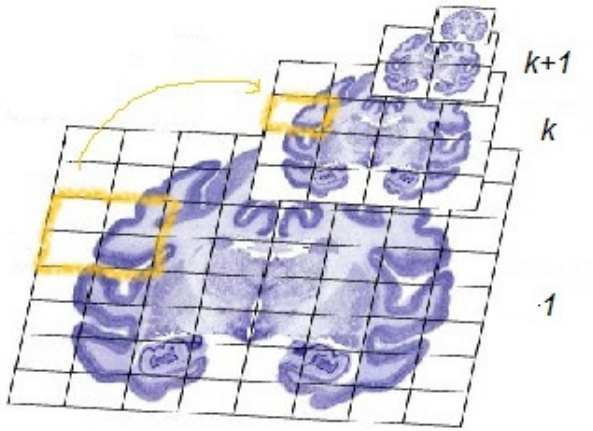
- K différentes résolutions pour chaque image (1 est la plus haute résolution (résolution originale) et K est la plus basse) ;
- $I^k(t) \rightarrow$ image de la résolution $k \in \{1, \ldots, K\}$;
- $\mathbf{d}_x^k$, $\mathbf{d}_y^k \rightarrow$ vecteur de mouvement dans la résolution k.

Multi-résolution

Cas général : Construction de la pyramide

Pour une pyramide diminuant de moitié à chaque niveau, le pixel (x, y) de I_k est formé par une moyenne des quatre pixels du niveau supérieur qui le forme, soit :

$$I_k(x,y) = \frac{I_{k-1}(2x,2y) + I_{k-1}(2x+1,2y) + I_{k-1}(2x,2y+1) + I_{k-1}(2x+1,2y+1)}{4}$$



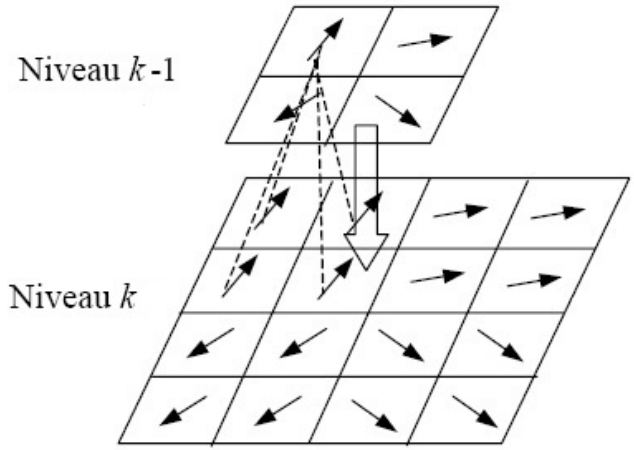
Multi-résolution

Cas général : Utilisation de la pyramide

On interpole le mouvement au niveau k (plus haute résolution), à l'aide du mouvement du niveau $k+1$ (plus basse résolution) par :

$$\tilde{\mathbf{d}}^k = \mathcal{U}(\mathbf{d}^{k+1}) \tag{4}$$

Où $\mathcal{U}$ est un opérateur d'interpolation. Dans cet exemple-ci, on applique une propagation du vecteur du mouvement du bloc associé :



Multi-résolution

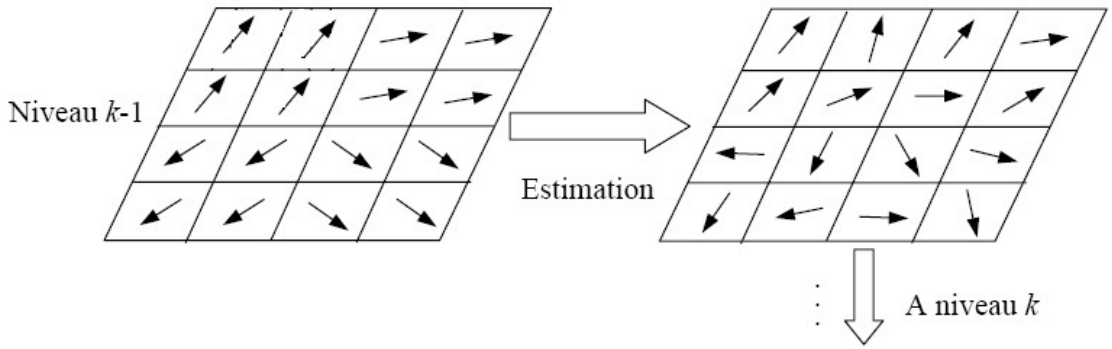
Cas général

Pour déterminer le mouvement à la résolution k, on doit minimiser la fonction suivante :

$$E(\mathbf{d}_x^k, \mathbf{d}_y^k) = \sum_{(x,y)\in I^k} \left[I^k(x + \tilde{\mathbf{d}}_x^k + \mathbf{q}_x^k, y + \tilde{\mathbf{d}}_y^k + \mathbf{q}_y^k, t_1) - I^k(x, y, t_2) \right]^2 \quad (5)$$

Où $\mathbf{q}^k = (\mathbf{q}_x^k, \mathbf{q}_y^k)^T$ est la correction qu'on rajoute au mouvement propagé de la résolution $k+1$ ($\tilde{\mathbf{d}}^k = \mathcal{U}(\mathbf{d}^{k+1})$).

$\rightarrow$ La correction $\mathbf{q}_k$ est l'estimation du flux optique à la résolution $k+1$!



Multi-résolution

Cas général

Par récursivité, on peut obtenir les relations suivantes entre les mouvements des différentes résolutions :

$$\begin{aligned}
\mathbf{d}^k &= \mathbf{q}^k + \tilde{\mathbf{d}}^k \\
&= \mathbf{q}^k + \mathcal{U}(\mathbf{d}^{k+1}) \\
&= \mathbf{q}^k + \mathcal{U}\left(\mathbf{q}^{k+1} + \mathcal{U}(\mathbf{d}^{k+2})\right) \\
&\cdot \\
&\cdot \\
&= \mathbf{q}^k + \mathcal{U}\left(\mathbf{q}^{k+1} + \mathcal{U}(\ldots\mathcal{U}(\mathbf{q}^K + \tilde{\mathbf{d}}^K))\right),
\end{aligned}$$

Où $\tilde{\mathbf{d}}^K = \mathbf{0}$.

Correspondance de blocs en multi-résolution

Application à la méthode des blocs

Avec l'approche classique de la méthode des blocs de taille fixe :

- La solution n'est pas nécessairement le minimum global de l'Eq.1, sauf si on fait une recherche exhaustive ;
- La recherche exhaustive est très longue.

L'approche multi-résolution :

- Estime d'abord grossièrement le mouvement dans une image à résolution réduite et dégradée ;
- Raffine la recherche de niveau en niveau, permettant ainsi une recherche globale orientée ("*coarse to fine*") ;
- Réduit le temps de calcul ;
- Est applicable à plusieurs algorithmes de calcul de flux optique.

Correspondance de blocs en multi-résolution

Application à la méthode des blocs

Supposons que nous avons une pyramide de K niveaux d'images dégradées :

- I_k : Image dégradée du niveau $k \in [1, K]$;
- B_k^m : Bloc de taille fixe ($N \times N$ pixels) du image $I_k(t)$ où le mouvement à l'intérieur de celui-ci est constant.
  $\rightarrow$ Nb blocs $\in I_k(t) = 4\times$ Nb blocs $\in I_{k+1}(t)$.
- $\mathbf{d}_k^m$: Vecteur de mouvement associé au bloc B_k^m

Correspondance de blocs en multi-résolution

Application à la méthode des blocs

Pour calculer les déplacements à l'aide de la multi-résolution :

1. On trouve l'ensemble des $\mathbf{d}_k^m$ pour tous les blocs B_k^m du niveau $k = K$ (le plus dégradé) ;
2. Pour changer de niveau$(k-1)$, on interpole le mouvement par :

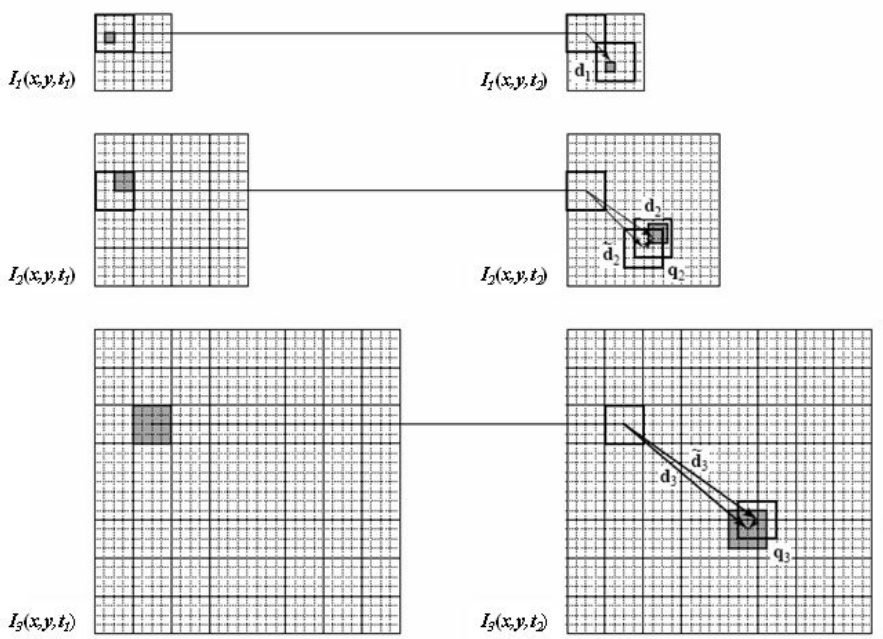
$$\begin{aligned} \tilde{\mathbf{d}}_{k-1}^m &= \mathcal{U}(\mathbf{d}_k^{\lfloor \frac{m}{2} \rfloor}) \\ &= 2\mathbf{d}_k^{\lfloor \frac{m}{2} \rfloor} \end{aligned}$$

3. Puis on apporte une correction $\mathbf{q}_k^m$, calculée par la méthode des blocs, pour avoir le déplacement réel au niveau k :

$$\mathbf{d}_{k-1}^m = \mathbf{q}_{k-1}^m + \tilde{\mathbf{d}}_{k-1}^m$$

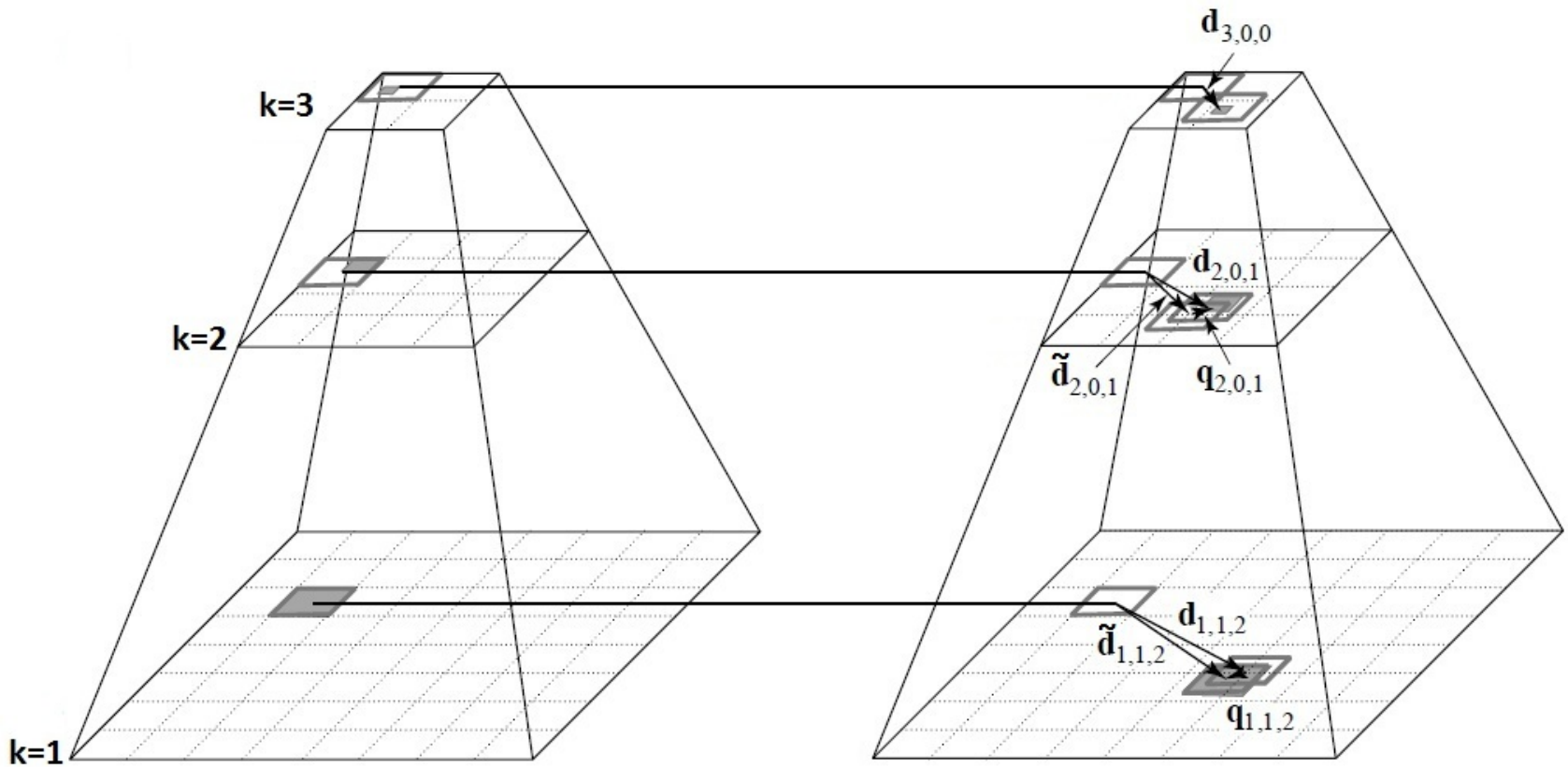
Correspondance de blocs en multi-résolution

Application à la méthode des blocs



Correspondance de blocs en multi-résolution

Application à la méthode des blocs



Correspondance de blocs en multi-résolution

Algorithmie des blocs fixes par hiérarchie

Supposons :

- R_0 : Rayon de recherche initial ;
- $M \times M$: Taille du image maximale ($k = K$) ;
- $N \times N$: Taille d'un bloc.

Alors :

- $\left(\frac{M}{2^{K-k}N}\right)^2$: Nombre de blocs au niveau k ;
- $M^2(2R_0+1)^2 \simeq 4M^2R_0^2$: Nombre d'opération pour une recherche exhaustive classique.

Donc :

- $\left(\frac{M}{2^{K-k}N}\right)^2 \left(2\frac{R_0}{2^{K-1}}+1\right)^2$: Nombre d'opération au niveau k.
- $Somme \approx \frac{1}{3 \cdot 4^{(K-2)}} 4M^2R_0^2$: Nombre d'opération total.

Correspondance de blocs en multi-résolution

Algorithme d'un bloc fixe par hiérarchie

Entrées: $\boxed{\bigtriangleup I(t), \bigtriangleup I(t+1), K, R_0}$: Pyramide de K niveaux des frames $I(t)$ et $I(t+1)$, rayon de recherche R_0

Sorties: $\boxed{d_x, d_y}$: Déplacement du bloc

$R \leftarrow \frac{R_0}{2^{K-1}}, \quad d_{0,x}, d_{0,y} \leftarrow 0, \quad i, j \leftarrow 0, \quad k \leftarrow 1;$

tant que $k \leq K$ **faire**

- $\tilde{d}_{k,x}, \tilde{d}_{k,y} \leftarrow 2d_{k+1,x}, 2d_{k+1,y}$;
- $I_k(t), I_k(t+1) \leftarrow \bigtriangleup I(t,k), \bigtriangleup I(t+1,k)$;
- $q_{k,x}, q_{k,y} \leftarrow$ Bloc fixe initialisé à $(\tilde{d}_{k,x}, \tilde{d}_{k,y})$;
- $d_{k,x}, d_{k,y} \leftarrow q_{k,x}, q_{k,y} + \tilde{d}_{k,x}, \tilde{d}_{k,y}$;
- $k \leftarrow k+1$;

retourner $d_{K,x}, d_{K,y}$

Correspondance de blocs déformables

Utilisons un modèle de mouvement plus complexe, tels les blocs déformables :

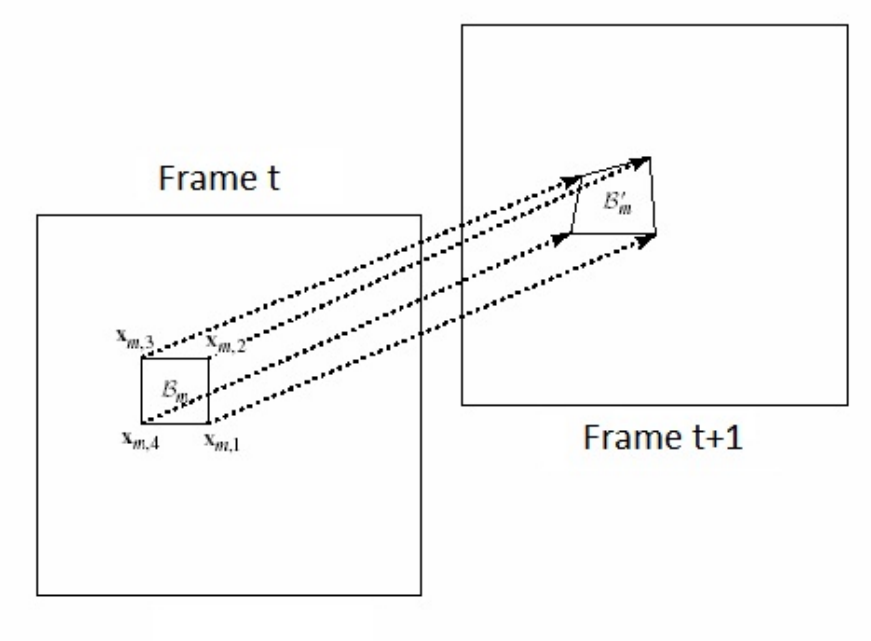
- Toujours “bloc par bloc”, alors le problème de discontinuité est toujours présent ;
- Permet la modélisation paramétrique des mouvements 3D :

Affine :

$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x + a_0 + a_1 x + a_2 y \\ y + b_0 + b_1 x + b_2 y \end{pmatrix}$$

Bilinéaire :

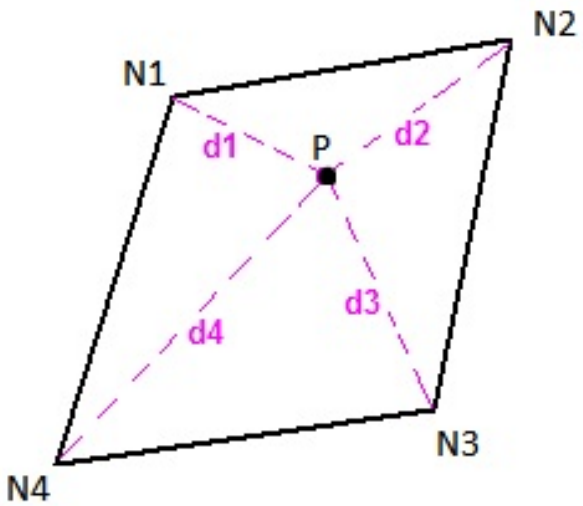
$$\begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} x + a_0 + a_1 x + a_2 y + a_3 xy \\ y + b_0 + b_1 x + b_2 y + b_3 xy \end{pmatrix}$$



Méthode des noeuds de contrôle

Pour palier au problème de mouvement constant à l'intérieur d'un bloc, on utilise la **méthode des noeuds**.

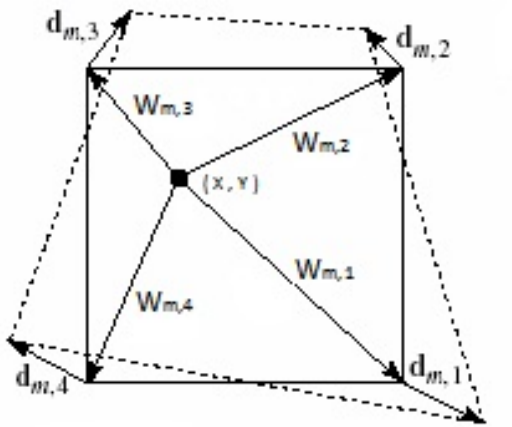
- Chaque coin d'un bloc représente un **noeud**. On interpole le mouvement à l'intérieur à partir de ces noeuds.
- On peut généraliser l'interpolation à un quadrilatère quelconque.
- On assigne à chaque noeud un certain poids w, où $\sum_i w_i = 1$.
- Cette méthode d'interpolation est appelée *inverse distance weighting*



.

Méthode par primitive de noeuds de contrôle

On peut donc finalement exprimer le déplacement d'un pixel à l'intérieur d'un quadrilatère formé par des noeuds.



Soit d_k^m le déplacement du noeud k du quadrilatère m, $d^m(x,y)$ le déplacement d'un point à l'intérieur du quadrilatère m et $w_k^m(x,y)$ le poids calculé au noeud k :

$$d^m(x,y) = \sum_{k=1}^{4} (w_k^m(x,y) * d_k^m) \tag{6}$$

Méthode par primitive de noeuds de contrôle

Pour un quadrilatère, on peut exprimer l'énergie à minimiser ainsi :

$$E_m = \sum_{(x,y)\in m} \left[I(x + \mathbf{d}_x^m, y + \mathbf{d}_y^m, t_1) - I(x, y, t_2) \right]^2 \quad (7)$$

- On utilise la méthode de noeuds de contrôle pour **ajouter de la précision sous-pixel** à l'algorithme de bloc.
- On utilise une méthode d'optimisation itérative afin de résoudre le problème.

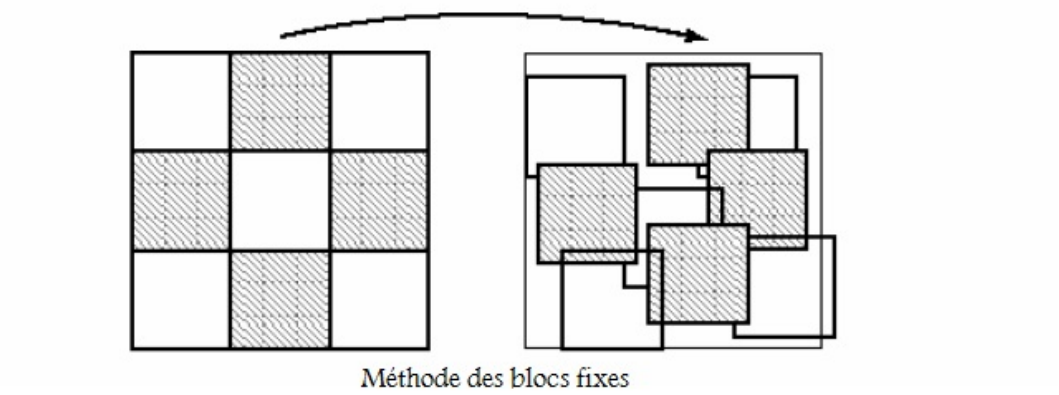
Méthode par primitive de noeuds de contrôle

Problème des blocs déformables

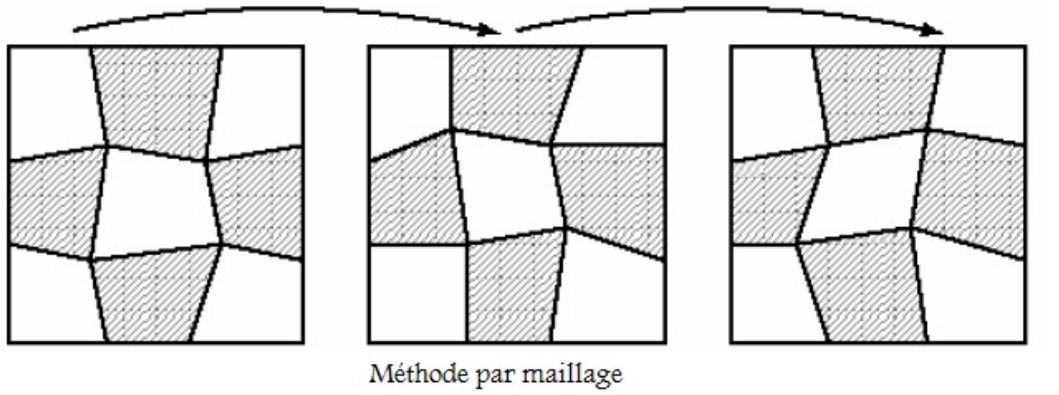
- Discontinuité aux bords des blocs (comme la méthode des blocs fixes).
  $\rightarrow$ Réglé par la méthode du maillage ;
- Complexité, mais on peut y aller en étapes :
  1. Appliquer la méthode des blocs fixes ;
  2. Comparer le résultat avec l'image suivant :
     - $\rightarrow$ Si l'erreur est faible ou inexistant, le mouvement est *translationnel* ;
     - $\rightarrow$ Si l'erreur est importante, le mouvement est *non-translationnel* :
       - On applique la méthode des noeuds sur ces blocs ;
       - Si l'erreur est encore importante, alors le mouvement n'est pas *compensable*.

Méthode par maillage

Le mouvement aux noeuds est calculé simultanément, non indépendamment d'un quadrilatère à l'autre. Cela permet d'avoir un maillage difforme sans discontinuité entre les quadrilatères :

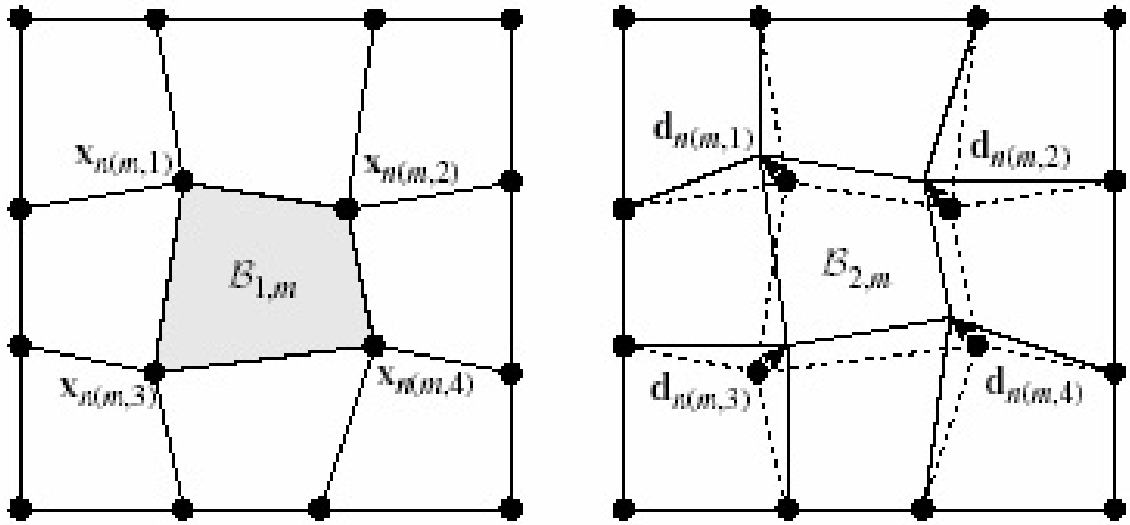


Méthode des blocs fixes



Méthode par maillage

Méthode par maillage



Maillage VS Noeuds :

- Maillage : Chaque noeud n'a qu'un seul mouvement, qui influence quatre "blocs" à la fois.
- Noeuds : Chaque noeud a quatre différents mouvements qui sont associés directement au bloc considéré.

Méthode par maillage

Comme tous les noeuds du maillage M sont déterminés en même temps, l'énergie globale à minimiser est représentée par :

$$E(d_x, d_y) = \sum_{m \in M} \sum_{(x,y) \in m} \left[I(x + \mathbf{d}_k^m, y + \mathbf{d}_k^m, t_1) - I(x, y, t_2) \right]^2 \quad (8)$$

Où l'on détermine le déplacement à l'intérieur du maillage par l'Eq.6 :

$$d^m(x, y) = \sum_{k=1}^{4} (w_k^m(x, y) * d_k^m)$$

Méthode par maillage

Optimisation et variantes

Le processus de minimisation est complexe, puisqu'un changement à un noeud du maillage influence beaucoup de vecteurs adjacents.

- On doit mettre à jour un noeud à la fois :
  - Descente de gradient (ordre 1) [Wang and Lee 1994]
  - Downhill simplex (ou autre simplex) (ordre 0)
- On peut ajouter une contrainte de qualité pour déterminer l'ordre des noeuds à optimiser (noeuds sur les contours)
- On peut ajouter une contrainte de mouvement pour ne pas déformer excessivement les objets.

Méthode par maillage

Les désavantages de la méthode de maillage :

- La recherche du déplacement des noeuds exige quand même une autre méthode d'estimation.
- Une génération de maillage adaptée à la scène est difficile à réaliser

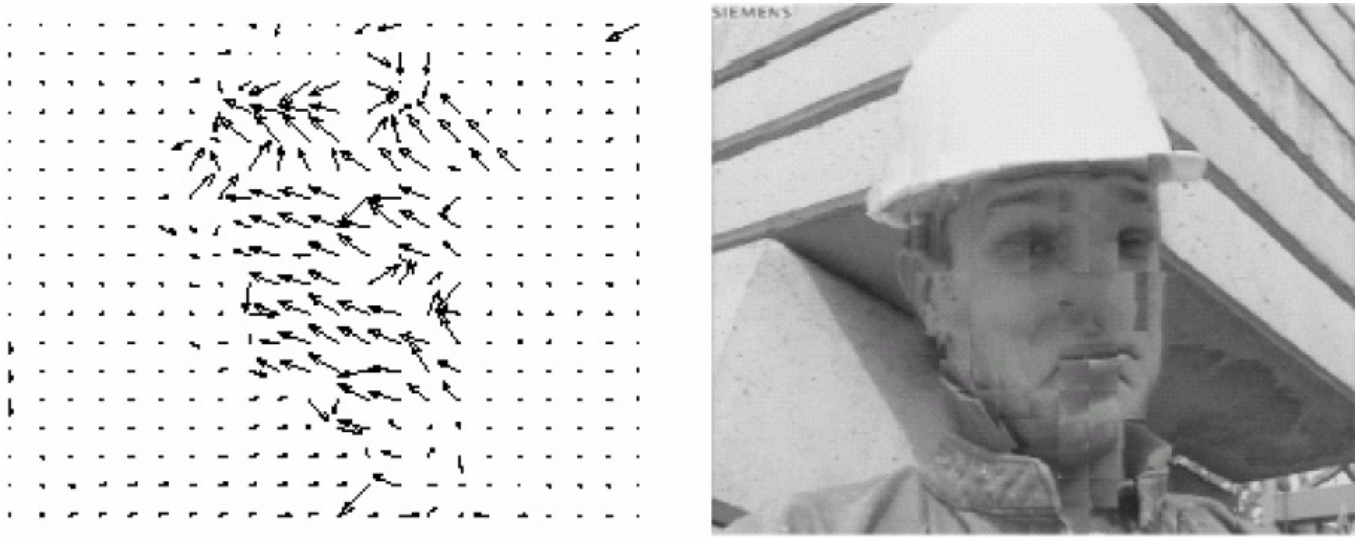
Les avantages :

- Mouvement non constant à l'intérieur des mailles (mouvement plus réaliste)
- Moins sensible aux changements d'illumination (sauf aux noeuds).
- Adapté pour les déformations non uniformes et mouvements complexes.

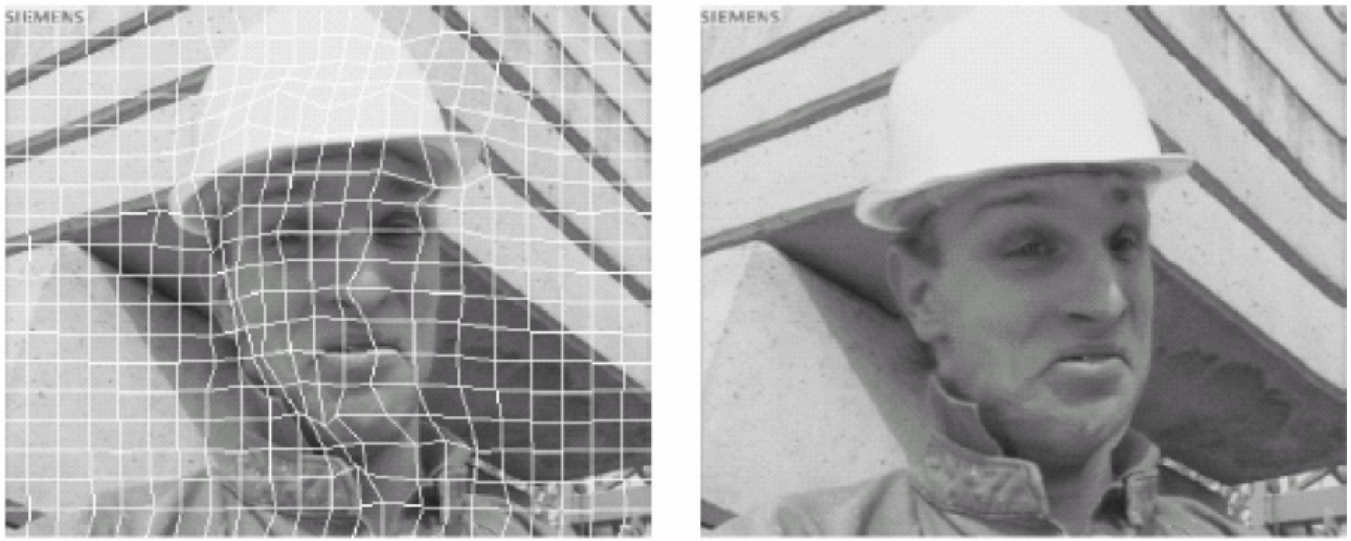
Méthode par maillage - comparée à blocs fixes

Méthode par blocs fixes

Méthode par maillage - comparée à blocs fixes



Méthode par blocs fixes



Méthode par maillage