Formation et Analyse d'Images

James L. Crowley

ENSIMAG 3 Premier Bimestre 2007/2008

Séance 9 7 décembre 2007

Analyse des Séquences d'Images - Le Suivi

Plan de la Séance :

Suivi des voisinages d'images par Inter-Correlation.....2
Comparaisons de Voisinages par SSD.....3
Détection par SSD et par Inter Corrélation 5
Voisinage en tant que vecteur :.....6
Relation entre SSD et NCC :.....7
Normalisation de l'Énergie.....7
Suivi par l'inter corrélation : la zone d'intérêt.....8
Suivi des régions ("blobs") avec un filtre de Kalman.....9
L'architecture d'un système de suivi robuste.....9
Les Blobs Gaussiens.....9
Détection par couleur (rappelle).....10
Calcul des moments :.....10
Composantes principales.....11
Estimation Baysienne Robuste.....13

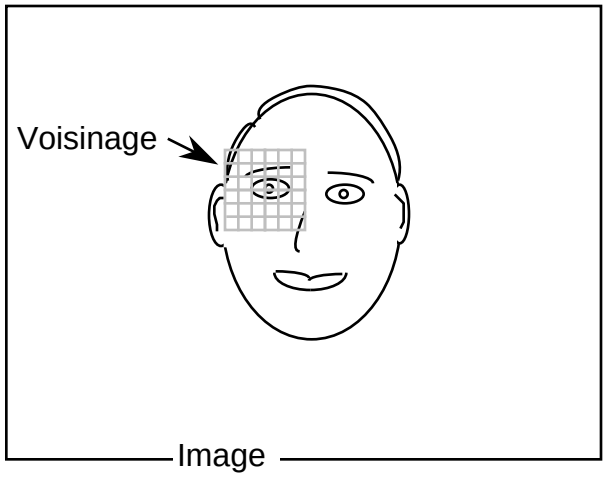
Suivi des voisinages d'images par Inter-Correlation.

Considérer la cas ou $\vec{X}$ est composées des échantillon du signal (d'image).

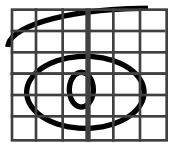
Dans ce cas, $\vec{X}$, est une "voisinage" ou une "fenetre".

Exemple : Considére l'image d'une téte.
Supposons que nous avons besoins faire une système de siuvi des direction de regardes des yeux.

Une approche pour le suivi consiste de faire de prend une imagette comme motif pour de l'apparence.



Exemple d'une motif :

$$\vec{X} = \{ x_1, x_2 \ldots x_n \} =$$ 

$\vec{X}$ sert de "prototype" pour une forme.

Soit une image, P(i, j), le probleme est de trouver le pixel (i_o, j_o) à lesquel il y a un oeuil.

Il nous faut une mésure de similitude.

Comment comparer deux vecteurs? Minimize la difference carrée.

Comparaisons de Voisinages par SSD

La similitude entre une motif $\vec{X}$ et une voisinage $\vec{Y}$ $P(i_o, j_o)$
est fourni par la SSD ("Sum of Squared Differences")

$$Sim(\vec{X},\vec{Y}) = \sum_{m=0}^{M} \sum_{n=0}^{N} (Y(m, n) - X(m, n))^2$$

Pour $\vec{Y}$ $P(i_o, j_o)$ ceci peut être écrit comme :

$$Sim(\vec{X}_d, P(i_o, j_o)) = \sum_{m=0}^{M} \sum_{n=0}^{N} (p(i_o+m,j_o+n) - X_d(m, n))^2$$

Pour trouver la position d'une ouil :

$$\min_{i_o\, j_o} \{ \sum_{m=0}^{M} \sum_{n=0}^{N} (p(i_o+m, j_o+n) - X(m, n))^2 \}$$

Pour trouver la la direction de regarde : :

$$\min_{d} \{ \sum_{m=0}^{M} \sum_{n=0}^{N} (p(i_o+m, j_o+n) - X(m, n))^2 \}$$

Mais $(A - B)^2 = (A^2 - 2AB + B^2)$

Donc, on peut écrire

$$Sim(\vec{X}_d, P(i_o, j_o)) = \sum_{m=0}^{M} \sum_{n=0}^{N} (p(i_o+m,j_o+n) - X_d(m, n))^2$$

$$= \sum_{m=0}^{M} \sum_{n=0}^{N} p(i_o+m,j_o+n)^2 + \sum_{m=0}^{M} \sum_{n=0}^{N} X_d(i_o+m,j_o+n)^2$$

$$- 2 \sum_{m=0}^{M} \sum_{n=0}^{N} p(i_o+m,j_o+n) X_d(m, n)$$

Mais :

1) $\sum_{m=0}^{M} \sum_{n=0}^{N} X_d(m,n)^2$ est la "energie" du signale du fenetre .

On peut la normaliser à unité pour chacun des prototypes une fois pour tout.

2) $\sum_{m=0}^{M} \sum_{n=0}^{N} p(i_o+m,j_o+n)^2$ est la "energie" du voisinage de l'image.

il est independant du direction de regarde "d".

Donc, si $\vec{X}_d$ est normalisé, le direction qu'on cherche vas maximisé :

$$Cor(\vec{X}_d, P(i_o, j_o)) = \sum_{m=0}^{M} \sum_{n=0}^{N} p(i_o+m,j_o+n)\, X_d(m, n)$$

$Cor(\vec{X}_d, P(i_o, j_o))$ est la correlation entre une masque $\vec{X}_d$ est un voisinage $P(i_o, j_o)$

Leçon : 1) On peut utiliser les voisinage comme les vecteurs caracteristiques.
2) On peut mésurer les statistiques de ces vecteurs.

Détection par SSD et par Inter Corrélation

Pour un bruit additif, une norme Euclidienne est la méthode "optimale" (min probabilité d'erreur)pour détecter le voisinage d'une image qui ressemble à un motif.

Hypothèse : bruit additif Gaussien.
Pas de rotation dans l'image (2D)
Pas de rotation dans l'espace 3D
Pas de changement d'échelle.

La norme Euclidienne est connue comme la SSD ("Sum of Squared Distances").
Il s'agit d'une opération efficace et précise, mais fragile.

Définition :

Soit un motif X(m, n) pour m [0, M–1], n [0, N–1].
Soit une image P(i, j) pour i [0, I–1], j [0, J–1]. (M << I, N << J)

Placer X(m, n)à chaque position possible (i, j)et
mesurer la distance Euclidienne entre X et les MN pixels de P à (i, j).

$$SSD(i, j) = \| X(m,n) - P(i+m, j+n) \|^2$$

$$= \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} (P(i+m, j+n) - X(m, n))^2$$

Si les pixels de P à la position (i, j)ressemblent à X, la distance est nulle.
Sinon, la position ayant le maximum de vraisemblance est celle correspondant au minimum de la fonction SSD.

$$\min_{i\ j} \{ \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} (P(i+m, j+n) - X(m, n))^2 \}$$

Voisinage en tant que vecteur :

Les pixels X(m, n)peuvent être vus comme un vecteur $\vec{X} = X_k$ ou $k = nM+m$.
Les pixels de chaque voisinage (i, j)de P peuvent aussi être vus comme un vecteur.

$$\vec{P} = P_k \text{ ou } k = (j+n)I+m+i$$

L'opération SSD est la norme de la différence de ces deux vecteurs :

$$SSD(i, j) = \| \vec{X} - \vec{P} \|^2 = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} (p(i+m, j+n) - X(m, n))^2 \}$$

Une autre méthode de comparaison est le produit scalaire
(CC pour "Cross Correlation"):

$$CC(i, j) = < \vec{X}, \vec{P} > = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} P(i+m,j+n)\, X(m, n)$$

Si les vecteurs $\vec{X}$ et $\vec{P}$ ont une longueur unitaire, le produit scalaire est un cosinus de l'angle entre les vecteurs.

$$X_u(m, n) = \frac{\vec{X}}{\|\vec{X}\|} = \frac{X(m, n)}{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} X(m, n)^2}$$

$$P_u(m, n) = \frac{\vec{P}}{\|\vec{P}\|} = \frac{P(i+m, j+n)}{\sum_{m=0}^{M-1} \sum_{n=0}^{N-1} P(i+m, j+n)^2}$$

On obtient un inter corrélation "normalisée" par l'énergie (NCC):

$$NCC(i, j) = < \vec{X}_u, \vec{P}_u > = \sum_{m=0}^{M-1} \sum_{n=0}^{N-1} \frac{P(i+m,j+n)}{\|\vec{P}\|} \frac{X(m, n)}{\|\vec{X}\|}$$

Le NCC est le cosinus entre $\vec{X}_u$ $\vec{P}_u$. Sa valeur est entre –1 et 1.

Relation entre SSD et NCC :

Pour $\vec{X}_u$ et $\vec{P}_u$ le minimum d'une SSD vaut le maximum d'une NCC.

On note que pour $\vec{X}_u, \vec{P}_u$

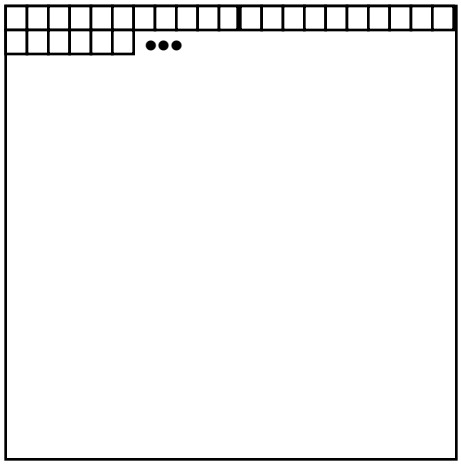
$$\| \vec{X}_u - \vec{P}_u \|^2 = \| \vec{X}_u \|^2 - 2 \| \vec{X}_u \vec{P}_u \| + \| \vec{P}_u \|^2$$

$$1 - 2 <\vec{X}_u, \vec{P}_u> + 1 = 2 - < \vec{X}_u, \vec{P}_u >$$

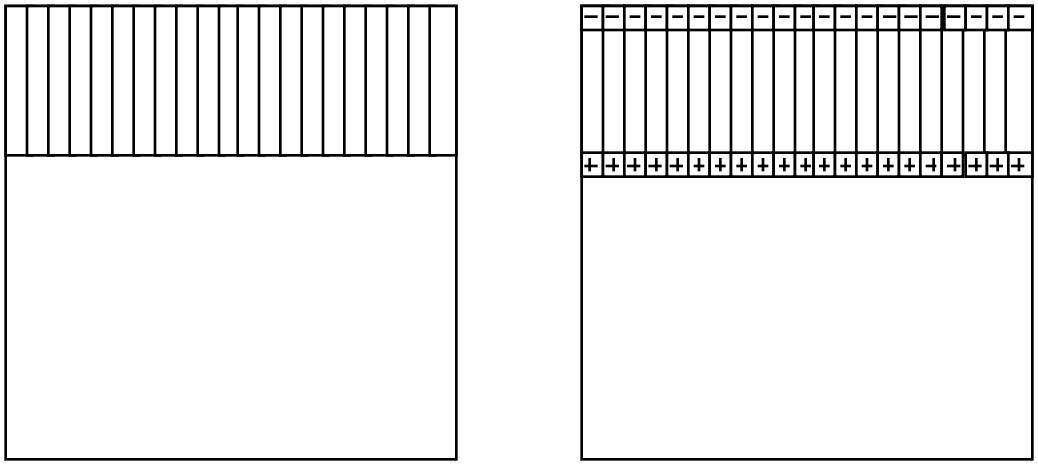
Normalisation de l'Énergie.

Il existe un algorithme rapide pour le calcul de la norme de chaque voisinage de P.

1) Calculer le carré de chaque pixel : $P^2(i, j) = P(i, j) \cdot P(i,j)$

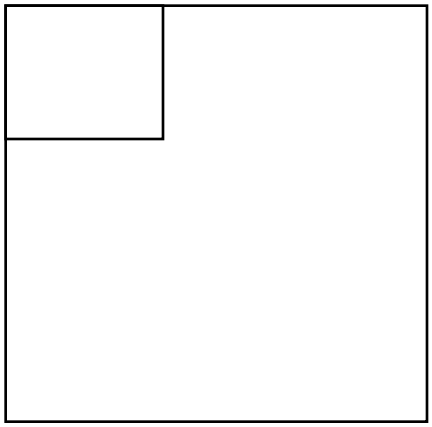


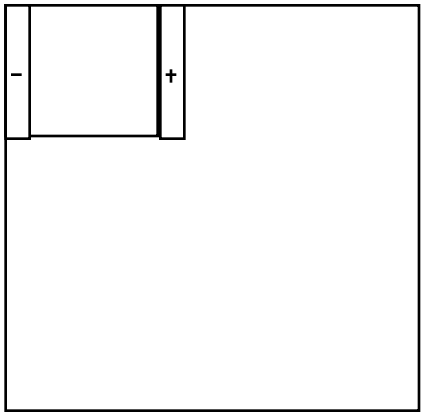
2) Pour chaque ligne, calculer $S_N(i,j) = \sum_{k=0}^{N} P^2(i, j+k)$



Premier Somme Somme Incrémentale

3) Calculer $S_{MN}(i,j) = \sum_{k=0}^{M} S_N(i+k, j)$





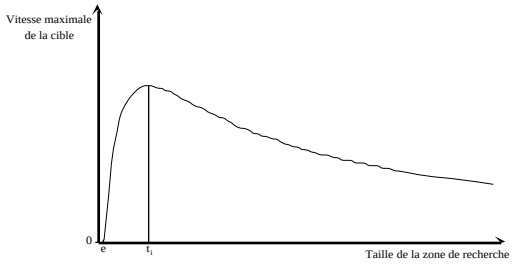
Suivi par l'inter corrélation : la zone d'intérêt

("Région of Interest" ou ROI):

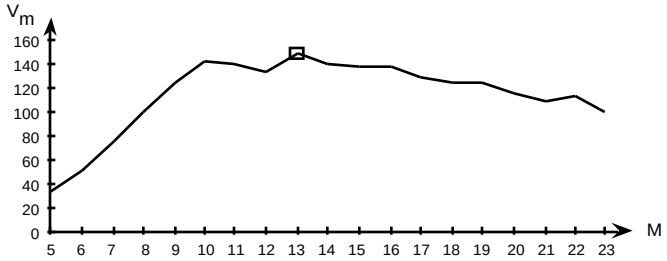
Pour chaque position (i,j) recherchée, il faut MN additions et MN multiplications.
On peut gagner en temps de calcul en limitant le nombre de positions testées.

Pour le suivi d'un objet, la taille de la zone de recherche est déterminée par le déplacement maximum possible depuis la dernière image.
Ceci se mesure en pixels/Δt, où Δt est le temps entre images.

Si on limite la recherche à une ROI, on peut diminuer le temps entre images, Δt.
Si on réduit Δt, on peut réduire la zone de recherche. Ceci entraîne une réduction supplémentaire dans la zone de recherche, etc.



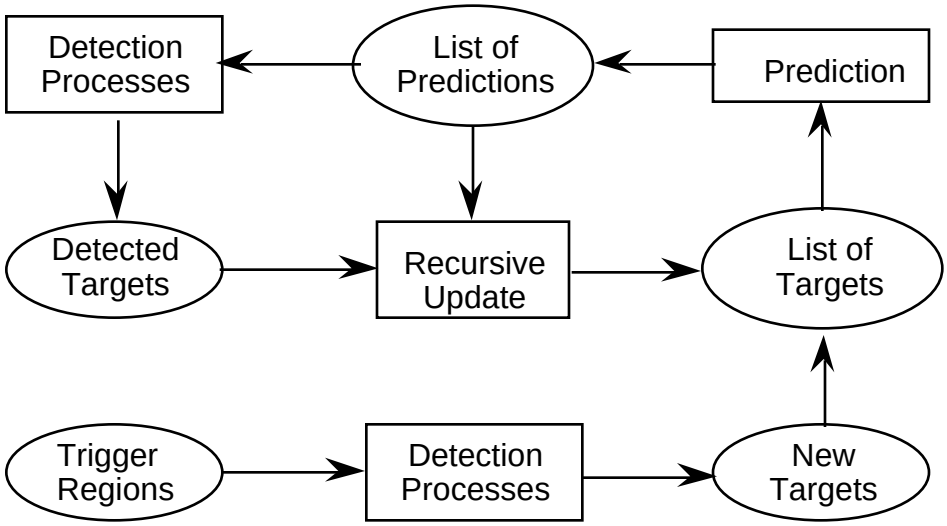
Courbe Théorique (d'après Bérard 93):



Courbe Expérimentale (Quadra 700)

Suivi des régions ("blobs") avec un filtre de Kalman

L'architecture d'un système de suivi robuste



Les Blobs Gaussiens

Les entités suivies sont une ensemble connexes de pixels sont de cibles, parfois appelées des "blobs".

On peut décrire une blob par un vecteur de caractéristiques "invariantes" à l'orientation grâce aux "moments" d'est pixels sorties d'une procédure de détection appliqué à une "ROI" (Region of Interest).

Le ROI est une rectangle englobante représentée par quatre paramètres : (u, l, b, r)

u - "up" - le premier ligne du ROI.
l - "left" - la première colonne du ROI.
b - "bottom" - le dernier ligne du ROI
r - "right" - La dernière colonne du ROI

Les procédures de détection fréquentent utilisé sont :

1) Ratio d'Histogramme de couleurs.
2) Différences d'image avec un fond adaptatif.
3) Différence d'image temporelle
4) Détection probabiliste calculée avec les caractéristiques locale (e.g. Dérivées de Gaussiens).

Détection par couleur (rappelle)

Pour chaque pixel $\vec{C}(i, j)$ $p(\text{objet} \mid C) = p(C \mid \text{objet}) \frac{p(\text{objet})}{p(C)}$

Soit M images de I x J pixels. Ceci fait N = I x J x M Pixels.
Soit h(r, v), l'histogramme de tous les N pixels.
Soit $h_o(r, v)$, l'histogramme des N_o pixels de l'objet "o".

$$p(\text{objet}) = \frac{M_o}{M}$$

$$p(\vec{C}) = \frac{1}{M} \; h(\vec{C})$$

$$p(\vec{C} \mid \text{objet}) = \frac{1}{M_o} \; h_o(\vec{C})$$

$$p(\text{objet} \mid \vec{C}) = p(\vec{C} \mid \text{objet}) \frac{p(\text{objet})}{p((\vec{C}))} = \frac{1}{M_o} \; h_o(\vec{C}) \frac{\frac{M_o}{M}}{\frac{1}{M} h((\vec{C}))} = \frac{h_o(\vec{C})}{h(\vec{C})}$$

Calcul des moments :

Soit w(i, j), des valeurs issues d'un algorithme de détection dans une ROI de taille N x M

Somme des Pixels : $S = \sum_{i=1}^{M} \sum_{j=1}^{N} w(i, j)$

Premiers moments :

$$\mu_i = \frac{1}{S} \sum_{i=1}^{M} \sum_{j=1}^{N} w(i, j) \cdot i \qquad \mu_j = \frac{1}{S} \sum_{i=1}^{M} \sum_{j=1}^{N} w(i, j) \cdot j$$

Le premier moment est le centre de gravité de la forme :

Deuxième moment :

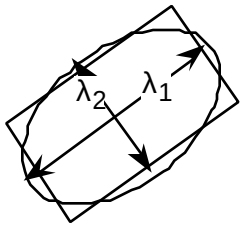
$$\sigma_i^2 = \frac{1}{S} \sum_{i=1}^{M} \sum_{j=1}^{N} (w(i, j)) \cdot (i - \mu_i)^2 \qquad \sigma_j^2 = \frac{1}{S} \sum_{i=1}^{M} \sum_{j=1}^{N} w(i, j) \cdot (j - \mu_j)^2$$

$$\sigma_{ji}^2 = \frac{1}{S} \sum_{i=1}^{M} \sum_{j=1}^{N} w(i, j) \cdot (i - \mu_i)(j - \mu_j)$$

Ceci permet de définir les "axes", majeur, λ_1 et mineur, λ_2, de la forme par analyse des composantes principales du deuxième moment

$$\mathbf{C} \triangleq \begin{bmatrix} \sigma_i^2 & \sigma_{ij}^2 \\ \sigma_{ij}^2 & \sigma_j^2 \end{bmatrix}$$

Composantes principales



Les deuxièmes moments sont "invariants" à l'orientation

Les axes sont calculés par une analyse en composantes principales de la matrice C. Il s'agit de trouver une rotation, Φ, dans l'espace de caractéristiques $\Phi\,\mathbf{C}\,\Phi^T = \Lambda$ telles que Λ soit diagonale.

$$\Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \quad \text{tel que } \lambda_1 > \lambda_2 \qquad \Phi = \begin{bmatrix} \text{Cos}(\theta) & \text{Sin}(\theta) \\ -\text{Sin}(\theta) & \text{Cos}(\theta) \end{bmatrix}$$

$$\Phi\,\mathbf{C}\,\Phi^T = \Lambda = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \qquad \Phi^T\,\Phi = \mathbf{I} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

$$\Phi\,\mathbf{C}\,\Phi^T\,\Phi = \Phi\,\mathbf{C}_P = \Lambda\,\Phi = \begin{bmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{bmatrix} \begin{bmatrix} \text{Cos}(\theta) & \text{Sin}(\theta) \\ -\text{Sin}(\theta) & \text{Cos}(\theta) \end{bmatrix}$$

Les lignes du Φ sont des vecteurs propres du **C.**

La longueur des axes majeurs et mineur est les valeurs propres de la matrice **C**.

θ est l'orientation de l'axe "majeur" et λ_1 / λ_2 est le rapport entre la longueur et la largeur.

λ_1 / λ_2 est une caractéristique invariante de la taille et de l'orientation.

$$\hat{X}_t = (x, y, w, h, a)$$

où

x, y : Centre de Gravité.
h : "height" - valeur du premier vecteur propre
w : "width" - valeur du deuxieme vecteur propre
a : angle - orientation du premier vecteur propre

Confiance :

$CF_t \quad [0, CF_{max}]$

$CF = S / NM \cdot D$

S = somme de valeurs de détection
NM : Surface du ROI
D : Valeur maximale d'une détection

e.g. D = 1 pour la probabilité.
D = 512 pour une différence d'image.

Précision de la position

$$\mathbf{P} = C = \begin{pmatrix} \hat{\sigma}_{xx}^2 & \hat{\sigma}_{xy}^2 \\ \hat{\sigma}_{xy}^2 & \hat{\sigma}_{yy}^2 \end{pmatrix}$$

La précision englobe la taille ET les erreurs d'estimation.

Pour chaque "blob" a chaque instante, on maintient :

$\hat{X}_t, \mathbf{P}_t, CF_t.$

$$\hat{X}_t, \mathbf{P}_t, CF := F\{ \hat{X}_{t-1}, \mathbf{P}_{t-1}, CF, \vec{Y}_d, \mathbf{P}_d, CF_d\}$$

Prédiction : generation du ROI.

On ajoute une estimation de l'erreur du aux evolution du cible : σ_e

left = $x - 2\sigma_X + \sigma_e$
right = $x + 2\sigma_X + \sigma_e$
top = $y - 2\sigma_Y + \sigma_e$
bottom = $x + 2\sigma_Y + \sigma_e$

Estimation Baysienne Robuste

L'estimation robuste utilise une Gaussien comme "fenêtre" afin de favoriser l'influence de pixels proche de la prédiction. L'idée est de multiplier les pixels de détection par une Gaussien.

Parce que le produit de deux Gaussien est aussi une Gaussien, les moments sont artificiellement réduits par ce produit.Pour éviter cela on accroît la taille de la fenêtre par 2.

$$V = 2\,\mathbf{P}_t^*$$

On utilise V pour calculer le ROI (u, l, b, r).

Ensuite on calcule l'algorithme de détection dans la fenêtre.
(différence de fond, ratio d'histogramme de couleur, différence d'images, etc.….

On évalue une porte de validation :

$$G(x, y; X_t, V) = e^{-\frac{1}{2}\begin{pmatrix} x & y \end{pmatrix}\begin{pmatrix} \hat{\sigma}_{xx} & \hat{\sigma}_{xy} \\ \hat{\sigma}_{xy} & \hat{\sigma}_{yy} \end{pmatrix}\begin{pmatrix} x \\ y \end{pmatrix}}$$

$$CF_d = \frac{1}{N}\sum_{y=top}^{bottom}\sum_{x=left}^{right} d(x, y)\, G(x, y; X_t, V)$$

La nouvelle position estimée est $\hat{Y}_d$

$$\hat{Y}_d = \begin{pmatrix} \mu_x \\ \mu_y \end{pmatrix}$$

ou

$$\mu_x = \frac{1}{N}\sum_{y=top}^{bottom}\sum_{x=left}^{right} d(x, y) \cdot x \cdot G(x, y; X_t, V)$$

$$\mu_y = \frac{1}{N}\sum_{y=top}^{bottom}\sum_{x=left}^{right} d(x, y) \cdot y \cdot G(x, y; X_t, V)$$

La taille du cible est, $\mathbf{P}_d$, est

$$\mathbf{P}_d = \begin{pmatrix} \hat{\sigma}_{xx}^2 & \hat{\sigma}_{xy}^2 \\ \hat{\sigma}_{xy}^2 & \hat{\sigma}_{yy}^2 \end{pmatrix}$$

ou

$$\sigma_{xx}^2 = \frac{1}{N} \sum_{y=top}^{bottom} \sum_{x=left}^{right} d(x, y) \cdot (x-\mu_x)^2 \cdot G(x, y; X_t, V)$$

$$\sigma_{yy}^2 = \frac{1}{N} \sum_{y=top}^{bottom} \sum_{x=left}^{right} d(x, y) \cdot (y-\mu_y)^2 \cdot G(x, y; X_t, V)$$

$$\sigma_{xy}^2 = \frac{1}{N} \sum_{y=top}^{bottom} \sum_{x=left}^{right} d(x, y) \cdot (x-\mu_x) \cdot (y-\mu_y) \cdot G(x, y; X_t, V)$$

Les autres etaps sont :

- Initialisation: zones de détection
- Prédiction-verification
- élimination d'un cible
- Split-Merge