Estimation du mouvement entre paires d'images – Méthodes de flux optique
TP Master MISS, Module FSR-MED

Patrick Clarysse & Simon Rit

Git repository : https://github.com/PatrickClarysse/MISS_Motion_Estimation.git
Binder : https://github.com/PatrickClarysse/MISS_Motion_Estimation.git
Consigne : Un compte-rendu de TP vous est demandé. Il consiste en le code dûment commenté.

1. Contexte

Le contexte de l'étude concerne l'estimation de la transformation entre 2 images d'une même scène où un mouvement de faible amplitude s'est produit (de l'ordre du pixel). Dans ce cas, la grande majorité des méthodes de la littérature s'appuient sur l'hypothèse d'invariance du niveau de gris au cours du mouvement qui peut se traduire formellement par :

$$I(\mathbf{x},t)=I(\mathbf{x}+\mathbf{dx},t+dt)$$

où **x** représente la position d'un pixel de l'image, **dx** son déplacement (inconnu) et $I(\mathbf{x},t)$ l'intensité en **x** à l'instant t.

Un développement limité à l'ordre 1 conduit à l'équation de contrainte du mouvement (ECM):

$$I_x u + I_y v + I_t = \mathbf{v}.\nabla I + I_t = 0$$

$$\text{où } I_z = \frac{\partial I}{\partial z},\ \mathbf{v}=(u,v)^T \text{ la vitesse recherchée}$$

Cette équation unique comporte 2 inconnues (les 2 composantes de la vitesse) et l'on peut montrer qu'elle ne permet de déterminer que la composante de la vitesse dans la direction du gradient spatial de l'intensité ∇I. Il faut donc trouver les moyens d'intégrer une équation supplémentaire. Une proposition est d'imposer que les pixels voisins soient animés d'un mouvement proche. C'est ce qu'ont proposé B. Horn et B. Schunck dans leur article fondateur de 1981 .

2. La méthode de Horn et Schunck

La formulation de Horn et Schunck est donnée par l'équation intégrale suivante :

$$E(\mathbf{v})=\iint_\Omega (\mathbf{v}.\nabla I + I_t)^2 + \alpha^2\left(\|\nabla u\|^2 + \|\nabla v\|^2\right) d\Omega$$

où Ω représente le domaine de l'image et α le paramètre de régularisation.
La minimisation de cette équation repose sur les équations d'Euler Lagrange et conduit au système différentiel de 2 équations à 2 inconnues:

$$\begin{cases} I_x^2 u + I_x I_y v + I_x I_t - \alpha^2 \Delta u = 0 \\ I_y^2 v + I_x I_y u + I_y I_t - \alpha^2 \Delta v = 0 \end{cases} \qquad (1)$$

$$\text{où } \Delta u = \nabla^2 u \text{ est le Laplacien de } u$$

La résolution numérique requiert la discrétisation des différents termes : dérivées spatiales et temporelles et Laplacien. Horn et Schunck proposent les approximations numériques suivantes des dérivées:

$$I_x = \frac{1}{4}\begin{pmatrix} I_{i,j+1,t} - I_{i,j,t} + I_{i+1,j+1,t} - I_{i+1,j,t} \\ + I_{i,j+1,t+1} - I_{i,j,t+1} + I_{i+1,j+1,t+1} - I_{i+1,j,t+1} \end{pmatrix}$$

$$I_y = \frac{1}{4}\begin{pmatrix} I_{i+1,j,t} - I_{i,j,t} + I_{i+1,j+1,t} - I_{i,j+1,t} \\ + I_{i+1,j,t+1} - I_{i,j,t+1} + I_{i+1,j+1,t+1} - I_{i,j+1,t+1} \end{pmatrix}$$

$$I_t = \frac{1}{4}\begin{pmatrix} I_{i,j,t+1} - I_{i,j,t} + I_{i+1,j,t+1} - I_{i+1,j,t} \\ + I_{i,j+1,t+1} - I_{i,j+1,t} + I_{i+1,j+1,t+1} - I_{i+1,j+1,t} \end{pmatrix}$$

où les indices *i, j, t* correspondent respectivement aux indices de ligne, colonne et temps. Et l'approximation suivante du Laplacien (ici donnée pour la composante *u*, la formulation de la composante *v* étant identique) :

$$\nabla^2 u \approx 3\left(\bar{u}_{i,j,t} - u_{i,j,t}\right)$$
$$\bar{u}_{i,j,t} = \frac{1}{6}\left(u_{i-1,j,t} + u_{i,j+1,t} + u_{i+1,j,t} + u_{i,j-1,t}\right) + \frac{1}{12}\left(u_{i-1,j-1,t} + u_{i-1,j+1,t} + u_{i+1,j+1,t} + u_{i+1,j-1,t}\right)$$

Ces approximations se mettent facilement en œuvre par des convolutions avec des masques.
L'intégration de ces approximations dans le système d'équations différentielles (1) conduit au système suivant en chaque pixel:

$$\begin{cases} \left(3\alpha^2 + I_x^2 + I_y^2\right)u = \left(3\alpha^2 + I_y^2\right)\bar{u} - I_x I_y \bar{v} - I_x I_t \\ \left(3\alpha^2 + I_x^2 + I_y^2\right)v = \left(3\alpha^2 + I_x^2\right)\bar{v} - I_x I_y \bar{u} - I_y I_t \end{cases} \quad (2)$$

La résolution par une méthode de type Gauss-Jordan est couteuse (Grande matrice creuse). Les auteurs proposent donc une méthode itérative de Gauss-Seidel qui se traduit par les 2 équations suivantes :

$$\begin{aligned} u^{n+1} &= \bar{u}^n - I_x\left[I_x\bar{u}^n + I_y\bar{v}^n + I_t\right] \Big/ \left(3\alpha^2 + I_x^2 + I_y^2\right) \\ v^{n+1} &= \bar{v}^n - I_y\left[I_x\bar{u}^n + I_y\bar{v}^n + I_t\right] \Big/ \left(3\alpha^2 + I_x^2 + I_y^2\right) \end{aligned} \quad (3)$$

3. Travail proposé

L'objectif est d'implanter la méthode de Horn et Schunck en Python (Jupyter notebook). Le Jupyter notebook *MotionEstimationTemplate* vous est fourni avec le squelette de la méthode de Horn & Schunck. Il s'agit de le compléter avec la mise en œuvre des équations (3).

a. Implantation des opérateurs numériques

- Mettre en place les opérateurs de dérivées spatiales I_x, I_y et temporelle I_t à partir d'une opération de convolution (fonctions Python *signal.convolve2d*). Préciser les masques utilisés et vérifier le bon comportement de ces opérateurs.
- Implanter l'opérateur de moyennage local de *u* et *v* de la même façon. Préciser les masques utilisés.

b. Calcul des composantes

- Compléter le schéma itératif pour le calcul des composantes *u* et *v*. Mettre en place les fonctionnalités de vérification du bon déroulement des calculs : affichage des composantes, de l'évolution de la minimisation au cours des itérations.

c. Expérimentation

- Expérimenter l'algorithme sur les paires d'images rubiks_cube et taxi.
- Déterminer les effets des paramètres de l'algorithme.

d. Multi-résolution

- Mettre en place une pyramide de 3 résolutions pour la paire d'images.
- Définir et implémenter un opérateur de projection du champ de vitesse du niveau courant au niveau de résolution plus élevé.
- Utiliser cet opérateur pour implémenter une version multi-résolution de l'algorithme de Horn & Schunck.
- Comparer aux résultats précédents.

References

B. K. P. Horn and B. Schunck, "Determining optical flow," *Artificial Intelligence,* vol. 17, pp. 185-203, 1981.