Master Sciences et Technologies
mention Sciences et Technologies
de l'Information et de la Communication

Étude du mouvement

Martin Druon

Mémoire de stage de Master 2 recherche
Spécialité Fondements et Ingénierie de l'Informatique et de l'Image

Encadrants :
Mr Augereau Bertrand
Mr Tremblais Benoit

Laboratoire Signal, Image, Communications, FRE 2731
Université de Poitiers - UFR Sciences Fondamentales et Appliquées

Poitiers, juillet 2005

Table des matières

Table des figures

Première partie

Introduction

Introduction

Cela fait déjà plusieurs années que bon nombre de travaux théoriques ont pu être appliqués au traitement du signal et des images. Avec l'évolution croissante des capacités de stockage et de calcul des micro-ordinateurs, il est désormais possible d'envisager l'étude de séquences d'images. La liste des applications relatives à ce domaine croît de jour en jour : télésurveillance, compression, mesure de vitesse, analyse du comportement, indexation vidéo, suivi de mouvement... Ces nouvelles applications amènent de nouvelles questions à la fois pratiques et théoriques. C'est pourquoi, de plus en plus de laboratoires de recherche ont entamé récemment des travaux à ce sujet.

Le processus généralement utilisé pour étudier une séquence d'images peut se résumer en quatre étapes :

- **l'extraction :** elle va permettre d'estimer le mouvement apparent contenu dans l'intégralité de la séquence ;

- **le lissage :** il a pour principal but d'éliminer le bruit lié à l'acquisition ou à la compression afin d'améliorer la robustesse de la phase d'extraction ;

- **l'analyse :** elle permet d'exprimer le mouvement de façon mathématique ;

- **la classification :** elle a pour but de reconnaître le mouvement extrait grâce à son expression analytique.

Concernant l'extraction du mouvement, beaucoup d'articles ont déjà été publiés [4], [6], [9], [5]. On trouve également des comparatifs entre les différentes méthodes [1], [3], des algorithmes de débruitage [9], [8] ou des exemples d'applications concrets [7].

Par contre, peu de publications concernant l'analyse du mouvement sont disponibles car c'est un domaine de recherche encore très récent. De plus, les articles déjà parus sont généralement axés sur un domaine d'étude très spécifique. Par exemple, certains travaux étudient uniquement le comportement du visage, d'autres celui du corps humain [2]...

L'objectif du stage est d'étudier la possibilité de caractériser tout type de mouvement comme une combinaison linéaire de polynômes issus d'une base orthonormée. Contrairement aux articles cités précédemment, la méthode que nous souhaitons mettre au point ici se veut la plus générale possible. En effet, nous ne voulons pas nous limiter à l'étude de mouvements spécifiques ni se restreindre à une seule base.

La première partie du rapport traite de la phase d'extraction du mouvement à partir d'un ensemble d'images.

La seconde partie explique comment caractériser ce mouvement en l'exprimant sous forme analytique. Comme aucune solution concrète sur le sujet n'a été trouvée, il m'a été demandé d'étudier, d'élaborer et de tester une nouvelle méthode : l'utilisation de bases polynomiales orthonormées pour la caractérisation de champs de vecteurs.

Deuxième partie

Extraction du mouvement

Cette partie concerne l'extraction du mouvement contenu dans une séquence d'images. Pour cela, nous étudions le flux vidéo afin de déterminer le mouvement associé à chaque pixel, en fonction des variations d'intensité lumineuse.

Dans le premier chapitre, nous définissons quelques notions nécessaires en traitement de vidéos, puis nous étudions l'*équation du flot optique* qui permet d'approximer le mouvement à partir des variations d'intensité lumineuse de la séquence d'images.

Le deuxième chapitre détaille trois algorithmes permettant de résoudre l'équation du flot optique afin d'extraire le mouvement d'une séquence.

Chapitre 1

De la variation d'intensité lumineuse aux champs de déplacement

1.1 Notations

1.1.1 Représentation d'une séquence

D'un point de vue théorique, une image en niveaux de gris peut être représentée par l'application :

$$\begin{array}{rrcl} \mathcal{I}: & \mathbb{R}^2 & \rightarrow & \mathbb{R} \\ & (x_1, x_2) & \mapsto & \mathcal{I}(x_1, x_2) \end{array}$$

Par extension, une séquence, qui est un ensemble d'images liées entre elles par une notion de temps, peut être définie par l'application suivante :

$$\begin{array}{rrcl} \mathcal{S}: & \mathbb{R}^3 & \rightarrow & \mathbb{R} \\ & (x_1, x_2, x_3) & \mapsto & \mathcal{S}(x_1, x_2, x_3) \end{array}$$

Afin d'améliorer la lisibilité de ce rapport, les fonctions $\mathcal{I}(x_1, x_2)$ et $\mathcal{S}(x_1, x_2, x_3)$ seront notés $\mathcal{I}$ et $\mathcal{S}$.

Dans la suite de ce rapport, le repère utilisé est celui présenté figure 1.1.

Ces deux applications $\mathcal{I}$ et $\mathcal{S}$ définissent donc une image et une séquence dans le domaine continu. Mais, d'un point de vue pratique, nous sommes

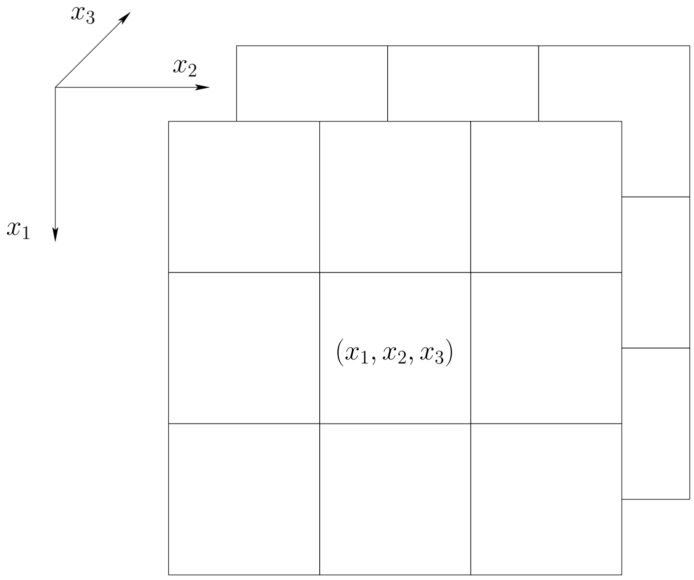
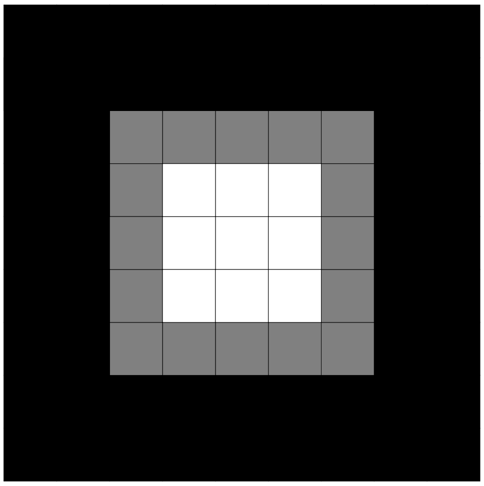


FIG. 1.1 – Repère utilisé.

obligés de travailler dans le domaine discret. Une image est donc définie par un ensemble fini de valeurs bornées. Dans notre cas, ces valeurs sont toujours comprises entre 0 (noir) et 1 (blanc) et sont stockées dans une matrice à deux dimensions (voir figure 1.2).



0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.5	0.5	0.5	0.5	0.5	0.0	0.0
0.0	0.0	0.5	1.0	1.0	1.0	0.5	0.0	0.0
0.0	0.0	0.5	1.0	1.0	1.0	0.5	0.0	0.0
0.0	0.0	0.5	1.0	1.0	1.0	0.5	0.0	0.0
0.0	0.0	0.5	0.5	0.5	0.5	0.5	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

FIG. 1.2 – Représentation matricielle d'une image.

L'application $\mathcal{S}$ permet donc de retrouver l'intensité lumineuse d'un pixel dans la séquence. Mais, pour pouvoir étudier les variations d'intensité lumineuse, il est nécessaire de définir, sur ces applications, les notions de dérivées.

1.1.2 Les dérivées

De façon formelle, la dérivée première d'une fonction $\mathcal{F}$ définie dans $\mathbb{R}^n$ est :

$$\frac{\partial \mathcal{F}}{\partial x_i} = \lim_{\Delta x_i \to 0} \frac{\mathcal{F}(x_1, \ldots, x_i + \Delta x_i, \ldots, x_n) - \mathcal{F}(x_1, \ldots, x_n)}{\Delta x_i} \tag{1.1}$$

Elle est notée : $\frac{\partial \mathcal{F}}{\partial x_i}$, $\partial_{x_i}\mathcal{F}$ ou $\mathcal{F}_{x_i}$.

De même, la dérivée seconde d'une fonction $\mathcal{F}$ définie dans $\mathbb{R}^n$ est :

$$\frac{\partial^2 \mathcal{F}}{\partial x_{i_1} \partial x_{i_2}} = \lim_{\Delta x_{i_2} \to 0} \frac{\partial x_{i_1}\mathcal{F}(x_1, \ldots, x_{i_2} + \Delta x_{i_2}, \ldots, x_n) - \partial x_{i_1}\mathcal{F}(x_1, \ldots, x_n)}{\Delta x_{i_2}} \tag{1.2}$$

Elle est notée : $\frac{\partial^2 \mathcal{F}}{\partial x_{i_1} \partial x_{i_2}}$, $\partial_{x_{i_1} x_{i_2}}\mathcal{F}$ ou $\mathcal{F}_{x_{i_1} x_{i_2}}$

Dans la pratique, nous avons vu que l'on ne dispose que de la donnée échantillonnée de $\mathcal{S}$ sur une grille discrète. Le calcul des opérateurs différentiels de l'image passe donc obligatoirement par une méthode numérique impliquant la discrétisation, c'est-à-dire l'approximation, de ces opérateurs. Dans ce rapport, nous considérons que le pas d'échantillonage spatial est de un pixel. Les dérivées peuvent alors être discrétisées ainsi :

$$\frac{\partial \mathcal{F}}{\partial x_i} = \frac{\mathcal{F}(x_1, \ldots, x_i + 1, \ldots, x_n) - \mathcal{F}(x_1, \ldots, x_i - 1, \ldots, x_n)}{2} \tag{1.3}$$

$$\begin{aligned} \frac{\partial^2 \mathcal{F}}{\partial x_{i_1} \partial x_{i_2}} = \frac{1}{4} \quad [\quad & \mathcal{F}(x_1, \ldots, x_{i_1} + 1, \ldots, x_{i_2} + 1, \ldots, x_n) \\ + \quad & \mathcal{F}(x_1, \ldots, x_{i_1} - 1, \ldots, x_{i_2} - 1, \ldots, x_n) \\ - \quad & \mathcal{F}(x_1, \ldots, x_{i_1} + 1, \ldots, x_{i_2} - 1, \ldots, x_n) \\ - \quad & \mathcal{F}(x_1, \ldots, x_{i_1} - 1, \ldots, x_{i_2} + 1, \ldots, x_n) \quad] \end{aligned} \tag{1.4}$$

1.1.3 Le gradient

De façon générale, le gradient permet d'exprimer la variation d'intensité d'un phénomène, par unité de distance. En traitement d'image, il permet notamment de retrouver le contour des objets.

Une approximation du gradient peut facilement être obtenue grâce aux dérivées premières définies précédemment :

$$\nabla \mathcal{S} = \begin{pmatrix} \mathcal{S}_{x_1} \\ \mathcal{S}_{x_2} \\ \mathcal{S}_{x_3} \end{pmatrix} \in \mathbb{R}^3 \tag{1.5}$$

1.2 Les champs de déplacement

Durant le processus d'acquisition, le mouvement réel en trois dimensions est projeté sur un plan en deux dimensions (voir figure 1.3) pour être converti en variations lumineuses. Le déplacement de ces variations est alors appelé *mouvement apparent* ou de *flot optique*. Dans la suite de ce rapport, le terme *mouvement* correspond toujours au mouvement apparent.

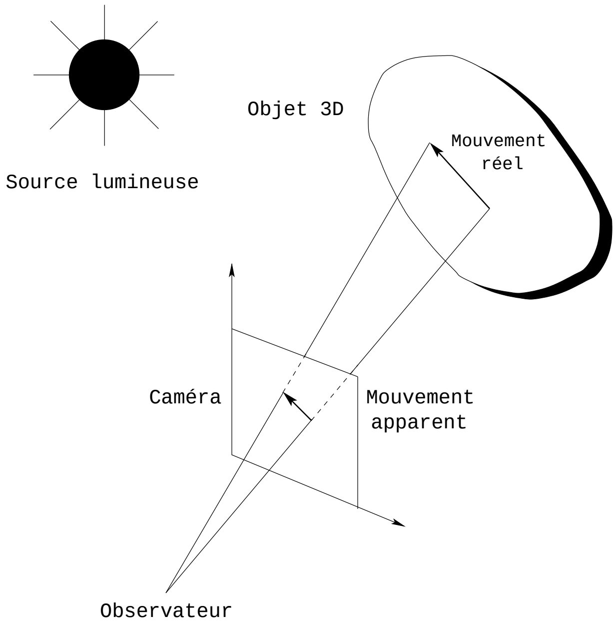


FIG. 1.3 – Correspondance entre le mouvement réel et le mouvement apparent.

L'analyse des déplacements passe donc nécessairement par l'étude de ces variations. Pour chaque pixel d'une image, on souhaite savoir à quel pixel il correspond dans l'image suivante. Cela revient à chercher le vecteur déplacement de ce pixel entre ces deux images. On va donc obtenir, pour chaque image, une grille composée d'autant de vecteurs déplacement que de pixels : c'est ce que l'on appelle un *champ de vecteurs déplacement*.

D'un point de vue mathématique, un champ de vecteurs est une application qui associe à chaque point de l'espace un vecteur. Mais ici, on suppose que l'intervalle de temps entre deux images (ici, la coordonnée x_3) est constant, ce qui est le cas dans la plupart des séquences issues de caméras standards. De ce fait, le déplacement selon cette coordonnée est lui aussi constant. On se permet donc d'ignorer cette composante et d'exprimer chaque vecteur uniquement selon les deux premières dimensions x_1 et x_2. De plus, pour éviter les calculs trop complexes, nous avons décidé, comme c'est généralement le cas, de générer un champ par image et non pas un champ pour toute la séquence. Chaque champ de vecteurs peut donc être défini ainsi :

$$\begin{array}{rrcl} \mathcal{F}: & \mathbb{R}^2 & \rightarrow & \mathbb{R}^2 \\ & (x_1, x_2) & \mapsto & (u, v) \end{array}$$

Un champ de vecteurs caractérise donc, à un instant précis, l'ensemble du mouvement entre deux images. Le mouvement d'une séquence peut donc être défini par un ensemble de champs de vecteurs.

Afin de simplifier les calculs, nous avons également décidé de représenter chaque champ de vecteurs par deux matrices de réels, l'une stockant le mouvement selon la dimension x_1, l'autre selon la dimension x_2 (voir figure 1.4).

La fonction $\mathcal{F}$ peut alors être divisée en deux sous-fonctions $\mathcal{U}$ et $\mathcal{V}$:

$$\begin{array}{rrcl} \mathcal{U}: & \mathbb{R}^2 & \rightarrow & \mathbb{R} \\ & (x_1, x_2) & \mapsto & \mathcal{U}(x_1, x_2) \end{array}$$

la fonction permettant d'obtenir le déplacement du pixel (x_1, x_2) selon la dimension x_1,

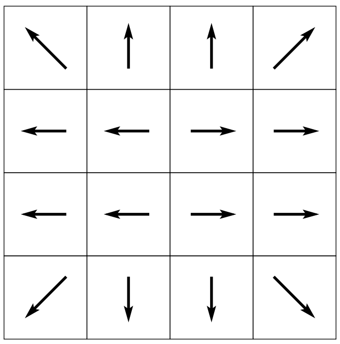
$$\begin{array}{rrcl} \mathcal{V}: & \mathbb{R}^2 & \rightarrow & \mathbb{R} \\ & (x_1, x_2) & \mapsto & \mathcal{V}(x_1, x_2) \end{array}$$

la fonction permettant d'obtenir le déplacement du pixel (x_1, x_2) selon la dimension x_2 et

$$\begin{array}{rrcl} \mathcal{F}: & \mathbb{R}^2 & \rightarrow & \mathbb{R}^2 \\ & (x_1, x_2) & \mapsto & (\mathcal{U}(x_1, x_2), \mathcal{V}(x_1, x_2)) \end{array}$$

la fonction définissant un champ de vecteurs.

Toute la difficulté réside donc dans la conversion des intensités lumineuses en vecteurs. Pour cela, on va chercher les déplacements des pixels d'une image à l'autre en espérant que les variations lumineuses correspondent au mouvement réel. La méthode utilisée ici pour résoudre ce problème se base sur *l'équation du flot optique*.



-1.0	-1.0	-1.0	-1.0
0.0	0.0	0.0	0.0
0.0	0.0	0.0	0.0
1.0	1.0	1.0	1.0

Matrice associée à la fonction U

-1.0	0.0	0.0	1.0
-1.0	-1.0	1.0	1.0
-1.0	-1.0	1.0	1.0
-1.0	0.0	0.0	1.0

Matrice associée à la fonction V

FIG. 1.4 – Représentation informatique d'un champ de vecteurs. De gauche à droite : le champ de vecteurs, la matrice associée à la fonction $\mathcal{U}$, la matrice associée à la fonction $\mathcal{V}$.

1.3 L'équation du flot optique

Le terme de flot optique désigne l'évolution des pixels d'une séquence au cours du temps. Le calcul du flot optique a donc pour but d'extraire des champs de vitesse denses à partir d'un ensemble images.

On cherche alors, pour chaque pixel de chaque image, à déterminer son déplacement, représenté par le vecteur vitesse suivant : $\vec{\nu} = \begin{pmatrix} u \\ v \end{pmatrix}$.

Si on suppose que l'intensité lumineuse est conservée sur un court intervalle de temps alors on peut écrire :

$$
\begin{aligned}
\frac{d\mathcal{S}(x_1, x_2, x_3)}{dx_3} &= 0 \\
\Leftrightarrow \frac{\mathcal{S}_{x_1} dx_1 + \mathcal{S}_{x_2} dx_2 + \mathcal{S}_{x_3} dx_3}{dx_3} &= 0 \qquad (1.6) \\
\Leftrightarrow \mathcal{S}_{x_1} \cdot \frac{d_{x_1}}{d_{x_3}} + \mathcal{S}_{x_2} \cdot \frac{d_{x_2}}{d_{x_3}} + \mathcal{S}_{x_3} \cdot \frac{d_{x_3}}{d_{x_3}} &= 0
\end{aligned}
$$

Or $\frac{d_{x_1}}{d_{x_3}}$ et $\frac{d_{x_2}}{d_{x_3}}$ correspondent respectivement au déplacement dans le temps selon la première et la deuxième composante. On peut alors écrire :

$$\mathcal{S}_{x_1} \cdot u + \mathcal{S}_{x_2} \cdot v + \mathcal{S}_{x_3} = 0 \qquad (1.7)$$

Cette équation est appelée **équation du flot optique**.

On peut aussi l'écrire sous forme vectorielle :

$$\nabla \mathcal{S} \cdot \vec{\nu} + \mathcal{S}_{x_3} = 0 \qquad (1.8)$$

On peut montrer que trouver $\vec{\nu}$ qui annule (1.8) revient à trouver $\vec{\nu}$ qui minimise la fonctionnelle :

$$\int \int_{\Omega} (\nabla \mathcal{S} \cdot \vec{\nu} + \mathcal{S}_{x_3})^2 \; dx_1 dx_2 = 0 \qquad (1.9)$$

où Ω désigne le support de l'image.

Malheureusement, cette équation ne suffit pas à résoudre le problème posé : on cherche à déterminer deux inconnues u et v à partir d'une seule équation (voir figure 1.5). À un pixel correspond alors une droite solution (voir figure 1.6). Dans ce cas, on obtient au mieux le flot normal $\vec{v_n}$. Or celui-ci ne peut être estimé localement. Ce problème est connu sous le nom de *problème d'ouverture*.

Un tel exemple est donnée figures 1.7 et 1.8 : ce dessin illustre le déplacement d'une droite de l'instant t à l'instant $t + 1$. À gauche de l'image, la

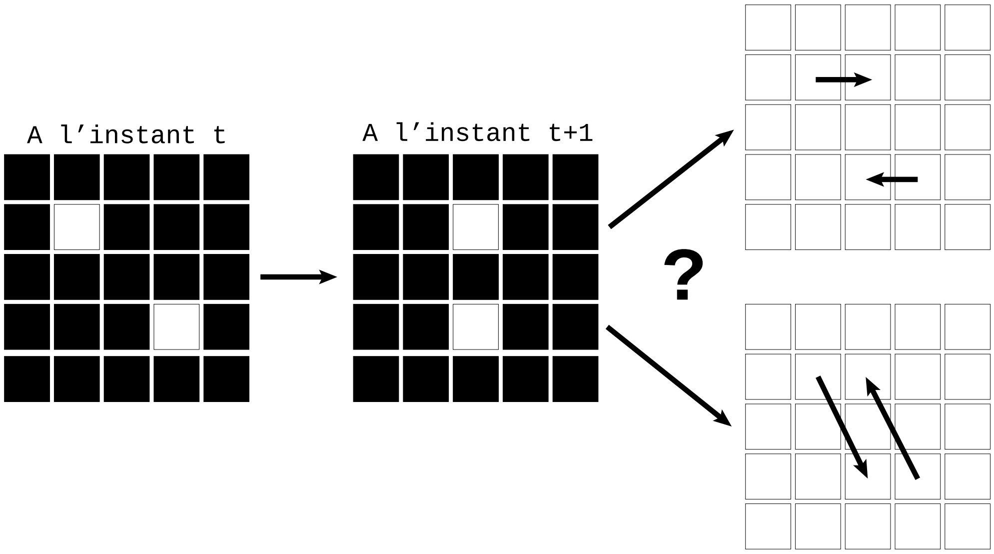


FIG. 1.5 – Une équation à deux inconnues : plusieurs solutions possibles !

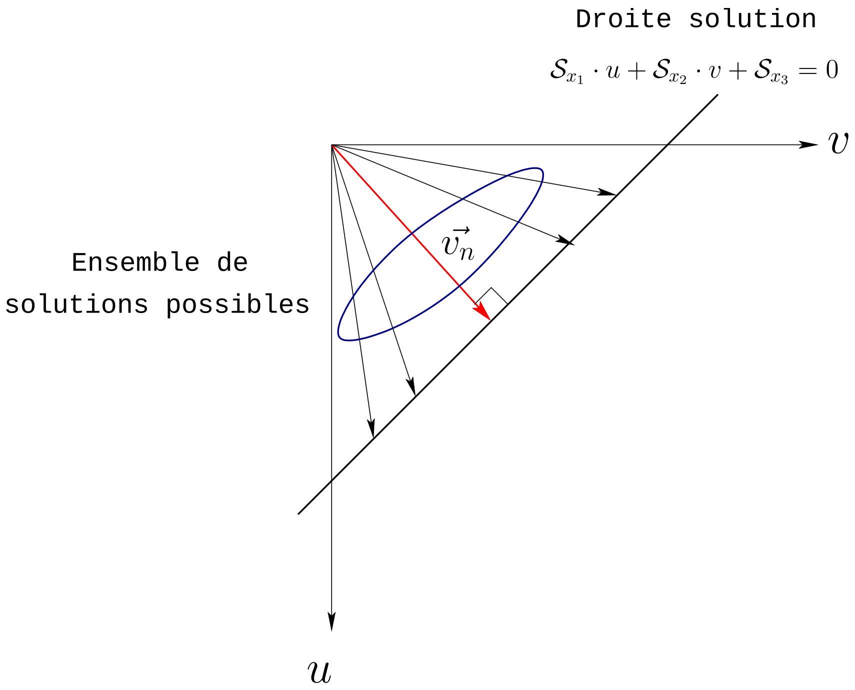


FIG. 1.6 – Ensemble de solutions possibles

fenêtre de visualisation est petite. Avec les informations dont on dispose, on peut supposer que la droite effectue un déplacement horizontal. Mais lorsque

que l'on agrandit la taille de la fenêtre (à droite de l'image) on se rend compte alors que la droite effectue un déplacement oblique.

Lorsque le problème d'ouverture est présent, seule la composante normale au contour peut être extraite :

$$\vec{v_n}(x_1, x_2, x_3) = -\frac{\mathcal{S}_{x_3}}{\|\nabla \mathcal{S}\|} \cdot \frac{\nabla \mathcal{S}}{\|\nabla \mathcal{S}\|} \tag{1.10}$$

La plupart des algorithmes d'extraction de flot optique essayent de résoudre ce problème en introduisant de nouvelles contraintes.

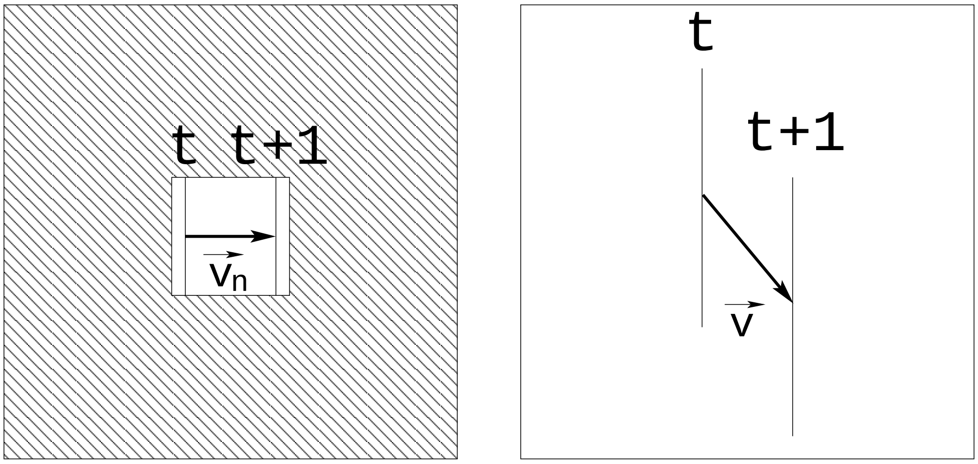


FIG. 1.7 – Exemple du problème d'ouverture. À gauche : un voisinage trop petit ne permet pas de déterminer le mouvement exact. À droite : un voisinage suffisant permet de déterminer le mouvement réel.

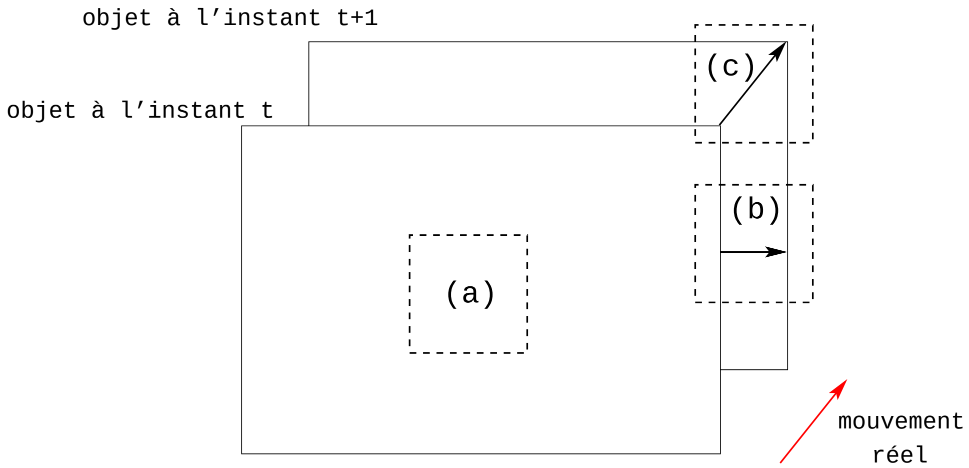


FIG. 1.8 – Les trois cas possibles lors de l'extraction du flot. (a) aucune information disponible : pas de flot. (b) problème d'ouverture : on ne peut calculer que le flot normal au contour. (c) suffisamment d'informations : le flot calculé est correct.

Chapitre 2

Étude de trois algorithmes d'extraction du mouvement

Dans ce chapitre sont présentés les trois algorithmes étudiés durant le stage. Bien que fournissant, en théorie, des résultats relativement proches, les méthodes utilisées pour résoudre le même problème initial sont totalement différentes.

Aucune comparaison n'est effectuée ici. Une étude qualitative de ces techniques, et de beaucoup d'autres, a déjà été effectuée par Barron, Fleet et Beauchemin dans [1].

2.1 Algorithme de Horn & Schunck

Horn et Schunck [4] ont été les premiers à proposer une solution permettant de résoudre le problème d'ouverture. Pour cela, ils ont introduit une seconde contrainte concernant la cohérence spatiale du champ de vecteur vitesse. Le but est alors de minimiser la fonctionnelle :

$$\int\int_{\Omega} (\nabla \mathcal{S} \cdot \vec{\nu} + \mathcal{S}_{x_3})^2 + \lambda^2 (\|\nabla u\|^2 + \|\nabla v\|^2) \cdot dx_1 dx_2 \tag{2.1}$$

avec λ un coefficient qui permet de faire un compromis entre les deux contraintes.

L'algorithme proposé est itératif et simple à mettre en œuvre. De plus, Horn et Schunck ont démontré qu'il converge vers la solution. À tout moment, on peut donc arrêter l'algorithme et affirmer que la solution dernière obtenue est au moins équivalente, sinon meilleure, que les solutions déjà calculées. Le flot est calculé, pour chaque itération n, par :

$$\begin{cases} u^{n+1} = \bar{u}^n - \frac{\mathcal{S}_{x_1}[\mathcal{S}_{x_1}\bar{u}^n + \mathcal{S}_{x_2}\bar{v}^n + \mathcal{S}_{x_3}]}{\alpha^2 + \mathcal{S}_{x_1}^2 + \mathcal{S}_{x_2}^2} \\ v^{n+1} = \bar{v}^n - \frac{\mathcal{S}_{x_2}[\mathcal{S}_{x_1}\bar{u}^n + \mathcal{S}_{x_2}\bar{v}^n + \mathcal{S}_{x_3}]}{\alpha^2 + \mathcal{S}_{x_1}^2 + \mathcal{S}_{x_2}^2} \end{cases} \tag{2.2}$$

où $\bar{u}^n$ et $\bar{v}^n$ désignent respectivement la moyenne des voisins de u et de v à l'itération n et α une constante réelle très faible utilisée pour éviter la division par zéro. Cette moyenne est calculée par :

$$\begin{cases} \bar{u}_{x_1,x_2,x_3} &= \frac{1}{6} \cdot [u_{x_1-1,x_2,x_3} + u_{x_1,x_2+1,x_3} + u_{x_1+1,x_2,x_3} + u_{x_1,x_2-1,x_3}] \\ & \quad + \frac{1}{12} \cdot [u_{x_1-1,x_2-1,x_3} + u_{x_1-1,x_2+1,x_3} + u_{x_1+1,x_2+1,x_3} + u_{x_1+1,x_2-1,x_3}] \\ \bar{v}_{x_1,x_2,x_3} &= \frac{1}{6} \cdot [v_{x_1-1,x_2,x_3} + v_{x_1,x_2+1,x_3} + v_{x_1+1,x_2,x_3} + v_{x_1,x_2-1,x_3}] \\ & \quad + \frac{1}{12} \cdot [v_{x_1-1,x_2-1,x_3} + v_{x_1-1,x_2+1,x_3} + v_{x_1+1,x_2+1,x_3} + v_{x_1+1,x_2-1,x_3}] \end{cases} \tag{2.3}$$

2.2 Algorithme de Lucas & Kanade

Lucas et Kanade [6] ont proposé une technique consistant à augmenter le nombre d'équations pour un pixel en considérant son voisinage V. Cela revient à considérer que les pixels d'un même voisinage ont le même vecteur déplacement. Pour cela, ils proposent de minimiser :

$$\sum_{\gamma \in V} W^2(\gamma) \cdot (\nabla \mathcal{S} \cdot \vec{\nu} + \mathcal{S}_{x_3})^2 \tag{2.4}$$

avec V le plus petit voisinage possible et $W(\gamma)$ une fonction de poids permettant de donner plus d'influence aux informations situées au centre qu'à la périphérie du voisinage. Le flot peut alors être obtenu en résolvant l'équation :

$$\underbrace{\begin{pmatrix} \sum W(\mathcal{S}_{x_1}^2) & \sum W(\mathcal{S}_{x_1}\mathcal{S}_{x_2}) \\ \sum W(\mathcal{S}_{x_1}\mathcal{S}_{x_2}) & \sum W(\mathcal{S}_{x_2}^2) \end{pmatrix}}_{A} \begin{pmatrix} u \\ v \end{pmatrix} = - \begin{pmatrix} \sum W(\mathcal{S}_{x_1}\mathcal{S}_{x_3}) \\ \sum W(\mathcal{S}_{x_2}\mathcal{S}_{x_3}) \end{pmatrix} \tag{2.5}$$

Si le problème d'ouverture est toujours présent, Lucas et Kanade proposent de recommencer le processus en prenant un voisinage plus important.

L'algorithme offre l'avantage d'être rapide [3] c'est pourquoi certains travaux essayent d'exploiter cette méthode en temps réel [7]. Par contre, dans certains cas, l'inversion de la matrice A est impossible ou instable.

2.3 Extraction par tenseurs de structure

Une autre méthode d'extraction du mouvement est basée sur l'étude des tenseurs de structure. Un tenseur de structure correspond simplement au produit du vecteur-gradient par son vecteur transposé. Aussi, si l'on se place dans un espace à trois dimensions alors le tenseur de structure correspondant est une matrice 3x3 réelle symétrique.

Par exemple, soit $\nabla \mathcal{S} = (\mathcal{S}_{x_1}, \mathcal{S}_{x_2}, \mathcal{S}_{x_3})^T$ le gradient. On a alors :

$$J(\nabla \mathcal{S}) = \nabla \mathcal{S} \cdot (\nabla \mathcal{S})^T = \begin{pmatrix} \mathcal{S}_{x_1}^2 & \mathcal{S}_{x_1}\mathcal{S}_{x_2} & \mathcal{S}_{x_1}\mathcal{S}_{x_3} \\ \mathcal{S}_{x_1}\mathcal{S}_{x_2} & \mathcal{S}_{x_2}^2 & \mathcal{S}_{x_2}\mathcal{S}_{x_3} \\ \mathcal{S}_{x_1}\mathcal{S}_{x_3} & \mathcal{S}_{x_2}\mathcal{S}_{x_3} & \mathcal{S}_{x_3}^2 \end{pmatrix} \tag{2.6}$$

avec J le tenseur de structure.

Au final, on a donc, pour chaque image, un tenseur par pixel. À partir de ce tenseur, on calcule ses valeurs propres λ_1, λ_2, λ_3 et ses vecteurs propres associés $\vec{v_1} = (x_1, y_1, z_1)^T$, $\vec{v_2} = (x_2, y_2, z_2)^T$, $\vec{v_3} = (x_3, y_3, z_3)^T$. L'étude de ces valeurs propres triées par ordre décroissant ($\lambda_1 > \lambda_2 > \lambda_3$) permet de déterminer le mouvement de ce pixel.

Quatre cas sont possibles :

- **Pas de structure** ($\lambda_1 \approx \lambda_2 \approx \lambda_3 \approx 0$). Il n'y a pas de variation des niveaux de gris dans le voisinage : aucun mouvement ne peut être estimé.

- **Ensemble de solutions** ($\lambda_1 > 0, \lambda_2 \approx \lambda_3 \approx 0$). L'information contenue dans le voisinage n'est pas suffisante : il existe une famille de solutions. On peut malgré tout extraire un flot normal :

$$\vec{v_n} = -\frac{z_1}{x_1^2 + y_1^2} \cdot \begin{pmatrix} x_1 \\ y_1 \end{pmatrix} \tag{2.7}$$

- **Solution unique** ($\lambda_1 > \lambda_2 > 0, \lambda_3 \approx 0$). Le voisinage contient suffisamment d'informations pour donner une direction unique et résoudre le problème d'ouverture. Le flot est obtenu par :

$$\vec{f} = \frac{1}{z_3} \cdot \begin{pmatrix} x_3 \\ y_3 \end{pmatrix} \tag{2.8}$$

– **Pas de flot cohérent** ($\lambda_1 > \lambda_2 > \lambda_3 > 0$). Il y a une discontinuité du flot : aucun mouvement cohérent ne peut être déterminé.

Cette technique permet d'extraire le flot contenu dans chaque image. Mais, afin de rendre les champs de vecteurs extraits plus "lisses", il peut être intéressant de le régulariser. Pour cela, plusieurs techniques de lissage sont possibles.

Une des plus simples est d'effectuer la convolution de chaque image par un noyau Gaussien. Cette technique, appelée *lissage homogène*, est rapide à mettre en œuvre mais ne tient pas compte du contenu de l'image. En effet, comme le lissage est le même sur toute l'image, cette méthode a tendance à rendre les contours flous.

Pour éviter ce problème, une autre technique est utilisée : le *lissage directionnel*. Comme son nom l'indique, on va tenir compte ici de la direction du gradient afin d'uniformiser les zones homogènes sans détériorer les contours. Cette méthode est généralement intégrée dans l'algorithme d'extraction par tenseur de structure. En effet, comme il est calculé à partir du gradient, ce tenseur contient déjà les informations concernant les contours. Il suffit alors, via une fonction ou une matrice de diffusion, de calculer les variations lumineuses à effectuer pour obtenir une image plus lisse. Ce processus itératif recalcule donc, à chaque itération, le gradient, le tenseur de structure, les valeurs propres, les vecteurs propres et les variations d'intensité à effectuer. De plus, le calcul des dérivées étant très sensible au bruit, on peut effectuer un lissage gaussien sur le gradient avant de calculer le tenseur de structure. Cela a pour effet de rendre la méthode plus robuste au bruit mais aussi d'élargir le domaine d'analyse et donc de limiter au maximum le problème d'ouverture.

Troisième partie

Interprétation du mouvement

Cette partie a pour but d'étudier, de façon analytique, le mouvement extrait précédemment. Pour cela, nous définissons un espace de représentation capable de caractériser ce mouvement puis nous étudions comment l'exprimer dans ce nouvel espace.

Le premier chapitre présente les notions d'espace et de base vectoriels. Il explique comment créer une base polynomiale orthonormée (qui est utilisée pour caractériser le mouvement) et définir les opérations nécessaires à sa manipulation. Il explique également comment exprimer les champs de vecteurs dans cette nouvelle base.

Le second chapitre traite de la reconnaissance. Il expose notamment comment, à partir des équations obtenues, il est possible de nommer le mouvement. Un exemple d'application montre le processus utilisé pour étudier les mouvements de tête d'une personne devant une webcam.

Le dernier chapitre présente comment, à partir des polynômes de la base, on peut simuler des mouvements synthétiques.

Chapitre 3

Modélisation du mouvement par des bases de polynômes

Nous avons vu, dans la partie précédente, qu'un champ de vecteurs peut être considéré comme une application :

$$\begin{array}{cccc} \mathcal{F}: & \mathbb{R}^2 & \rightarrow & \mathbb{R}^2 \\ & (x_1, x_2) & \mapsto & (\mathcal{U}(x_1, x_2), \mathcal{V}(x_1, x_2)) \end{array}$$

Nous cherchons ici à étudier le mouvement de façon analytique. Pour cela, la méthode proposée consiste à exprimer la fonction $\mathcal{F}$ par une combinaison linéaire de polynômes. Nous commençons donc par créer un espace vectoriel polynomial. Ensuite, nous munissons cet espace d'un produit scalaire ce qui permet de créer une base polynomiale. Finalement, nous définissons, dans cette base, les opérations permettant d'obtenir l'expression analytique recherchée.

La dernière partie de ce chapitre teste la sensibilité au bruit de la méthode mise en place.

3.1 Définition de l'espace vectoriel

Un espace vectoriel est un ensemble d'éléments muni de deux lois : l'addition, notée $+$, et la multiplication par un scalaire, notée $.$.

Soit $\mathcal{E}$ l'espace vectoriel et ϕ l'ensemble de ses éléments. Ici, comme $\mathcal{F}$ est définie sur $\mathbb{R}^2$, ϕ est composé de polynômes bivariables pouvant être définis ainsi :

$$P(x_1,x_2)=\sum_{i=0}^{I}\sum_{j=0}^{J}\ a_{i,j}\ (x_1)^i\ (x_2)^j \tag{3.1}$$

avec :

$$\begin{cases} (x_1,x_2)\in\mathbb{R}^2 \text{ les variables} \\ I\in\mathbb{N}^+ \text{ le degré selon } x_1 \\ J\in\mathbb{N}^+ \text{ le degré selon } x_2 \\ D\in\mathbb{N}^+ \text{ le degré du polynôme (si } I>J \text{ alors } D=I \text{ sinon } D=J) \\ \{a_{i,j}\}_{i\in[0;I]}^{j\in[0;J]}\in\mathbb{R}^{I*J} \text{ l'ensemble des coefficients} \end{cases}$$

Par soucis de lisibilité, comme tous les polynômes traités ici sont bivariables, ils seront notés par la la suite P et non plus $P(x_1,x_2)$.

Définissons maintenant les deux opérateurs :

$$\begin{array}{cccc} + & : & \phi^2 & \rightarrow & \phi \\ & & (P_1,P_2) & \mapsto & (P_1+P_2) \end{array}$$

$$\begin{array}{cccc} \textbf{.} & : & \mathbb{R}\text{ x }\phi & \rightarrow & \phi \\ & & (\lambda,P) & \mapsto & (\lambda\cdot P) \end{array}$$

Au final, l'ensemble ϕ muni des deux opérateurs $+$ et **.** constitue un espace vectoriel de polynômes.

3.1.1 Définition de la base

Une base est un sous-espace d'un espace vectoriel muni d'un produit scalaire. Si les éléments qui composent cette base sont orthogonaux deux à deux alors on dit qu'elle est *orthogonale*. De plus, si le produit scalaire de chacun de ces éléments par lui-même vaut un alors on dit qu'elle est *orthonormée*.

On vient de définir $\mathcal{E}$, un espace vectoriel de polynômes. Pour construire une base orthonormée il faut donc commencer par lui associer un produit scalaire. Il est noté $\langle\ |\ \rangle$ et est défini ainsi :

$$\begin{array}{cccc} \langle\ |\ \rangle & : & \phi^2 & \rightarrow & \mathbb{R} \\ & & (P_1,P_2) & \mapsto & \langle P_1 \mid P_2\rangle=\int_a^b\int_a^b\ P_1\ P_2\ \omega(x_1,x_2)\quad dx_1dx_2 \end{array}$$

avec $\omega(x_1,x_2)$ une fonction de poids choisie selon le problème considéré.

Remarque : comme une fonction de poids est toujours associée au produit scalaire, on dit qu'on effectue un produit scalaire *pondéré*. Il peut également être noté : $\langle P_1 \mid \omega(x_1, x_2) \mid P_2 \rangle$ avec $\omega(x_1, x_2)$ la fonction de poids.

À partir de ce produit scalaire, il est possible de définir la distance entre deux polynômes :

$$\|P_1 - P_2\| = \sqrt{\langle P_1 - P_2 \mid P_1 - P_2 \rangle} \tag{3.2}$$

Notre espace vectoriel $\mathcal{V}$ muni d'un produit scalaire définit donc une base $\mathcal{B}$ composée de polynômes. Pour que cette base soit orthonormée, il faut maintenant que ses éléments $\{P_1, P_2, \ldots, P_n\}$ soient définis de sorte que :

$$\begin{cases} \langle P_i \mid P_j \rangle = 0 \text{ si } i \neq j \\ \langle P_i \mid P_i \rangle = 1 \end{cases} \tag{3.3}$$

On cherche donc à créer un ensemble de polynômes orthonormés. Pour cela, nous allons utiliser la méthode de Gram-Schmidt.

3.1.2 Algorithme de Gram-Schmidt

Cet algorithme permet de rendre orthonormaux deux à deux un ensemble de polynômes divers.

Soit $\{\chi_{i,j}(x_1, x_2)\}_{i\in[0;I]}^{j\in[0;J]}$ un ensemble de polynômes quelconques.
Soit $\{\psi_{i,j}(x_1, x_2)\}_{i\in[0;I]}^{j\in[0;J]}$ l'ensemble des polynômes orthogonaux.
Soit $\{\phi_{i,j}(x_1, x_2)\}_{i\in[0;I]}^{j\in[0;J]}$ l'ensemble des polynômes orthonormaux.
Soit $\omega(x, y)$ une fonction de poids.

Dans un premier temps, on calcule, à partir d'un polynôme quelconque extrait de $\{\chi\}$, un nouveau polynôme de manière à ce qu'il soit orthogonal à tous les polynômes déjà calculés (ceux appartenant à l'ensemble $\{\psi\}$) grâce à la relation :

$$\psi_{i,j}(x_1, x_2) = a_{i,j,i-1}\phi_{i-1,j}(x_1, x_2) + \cdots + a_{i,j,1}\phi_{1,j}(x_1, x_2) + a_{i,j,0}\phi_{0,j}(x_1, x_2) + \chi_{i,j}(x_1, x_2) \tag{3.4}$$

avec $a_{i,j,k}(x_1, x_2) = -\int\int \chi_{i,j}(x_1, x_2)\ \phi_{i,k}(x_1, x_2)\ \omega(x_1, x_2) \quad dx_1 dx_2$

Puis on normalise ce polynôme de la façon suivante :

$$\phi_{i,j}(x_1, x_2) = \frac{\psi_{i,j}(x_1, x_2)}{\sqrt{\int\int \psi_{i,j}^2(x_1, x_2)\ \omega(x_1, x_2) \quad dx_1 dx_2}} \tag{3.5}$$

Au final, les polynômes contenus dans $\{\phi(x_1, x_2)\}$ sont orthogonaux deux à deux et normalisés. On va donc pouvoir les utiliser pour créer la base $\mathcal{B}$.

3.1.3 Opérations applicables sur la base

On vient de créer une base polynomiale orthonormée. Pour pouvoir exprimer le mouvement dans cette base, il est nécessaire de lui associer de nouvelles opérations.

Projection d'un champ de vecteurs sur un polynôme

La projection d'un champ de vecteurs sur un polynôme est obtenue en calculant le produit scalaire entre ce champ et ce polynôme. Comme ici le champ de vecteurs est composé de deux fonctions $\mathcal{U}(x_1, x_2)$ et $\mathcal{V}(x_1, x_2)$, sa projection sur un polynôme P va correspondre à deux scalaires :

$$\begin{cases} \langle\ \mathcal{U} \mid P\ \rangle = \displaystyle\sum_{x_1=a}^{b} \sum_{x_2=a}^{b} \mathcal{U}(x_1, x_2)\ P(x_1, x_2)\ \omega(x_1, x_2) \\ \langle\ \mathcal{V} \mid P\ \rangle = \displaystyle\sum_{x_1=a}^{b} \sum_{x_2=a}^{b} \mathcal{V}(x_1, x_2)\ P(x_1, x_2)\ \omega(x_1, x_2) \end{cases} \tag{3.6}$$

avec $[a; b]$ le domaine de définition des polynômes.

Projection d'un champ de vecteurs sur la base

Par extension, la projection d'un champ de vecteurs sur une base est obtenue en calculant le produit scalaire entre les deux fonctions associées au champ et chaque polynôme de la base. Les scalaires obtenus correspondent aux coefficients des deux polynômes caractéristiques.

Soient $\mathcal{U}(x_1, x_2)$ et $\mathcal{V}(x_1, x_2)$ les deux fonctions associées au champ de vecteurs.

Soit $\{P_{i,j}(x_1, x_2)\}_{i\in[0,I]}^{j\in[0,J]}$ l'ensemble des polynômes de la base $\mathcal{B}$.

Soient $P_{\mathcal{U}}(x_1, x_2)$ et $P_{\mathcal{V}}(x_1, x_2)$ les deux polynômes caractéristiques du mouvement.

On a alors :

$$\begin{cases} P_{\mathcal{U}} = \displaystyle\sum_{i=0}^{I} \sum_{j=0}^{I} \alpha_{i,j}\ (x_1)^i\ (x_2)^j \qquad \text{avec } \alpha_{i,j} = \langle\ \mathcal{U}\ \mid\ P_{i,j}\ \rangle \\ \\ P_{\mathcal{V}} = \displaystyle\sum_{i=0}^{I} \sum_{j=0}^{I} \beta_{i,j}\ (x_1)^i\ (x_2)^j \qquad \text{avec } \beta_{i,j} = \langle\ \mathcal{V}\ \mid\ P_{i,j}\ \rangle \end{cases} \tag{3.7}$$

3.1.4 Exemple : la base de Legendre

Durant tout le stage, la méthode a été testée en utilisant les polynômes de Legendre.

Génération de la base :

Ces polynômes sont définis par la formule de récurrence suivante :

$$\begin{cases} P_{0,0} = 1 \\ \\ P_{1,0} = x_1 \\ \\ P_{0,1} = x_2 \\ \\ P_{i+1,j} = \dfrac{2i+1}{i+1}\ x_1\ P_{i,j} \quad - \quad \dfrac{i}{i+1}\ P_{i-1,j} \\ \\ P_{i,j+1} = \dfrac{2j+1}{j+1}\ x_2\ P_{i,j} \quad - \quad \dfrac{j}{j+1}\ P_{i,j-1} \end{cases} \tag{3.8}$$

À partir de cette formule, on calcule donc les différents polynômes de la base puis on applique l'algorithme de Gram-Schmidt vu précédemment afin de rendre cette base orthonormée.

La fonction de poids associée à ces polynômes est $\omega(x_1, x_2) = 1$.

Domaine de définition :

Tous ces polynômes sont définis sur l'intervalle $[-1; 1]$. Les valeurs de x_1 et de x_2 doivent donc être comprises dans cet intervalle. Par exemple, les coordonnées (i, j) des pixels d'une image de taille MxN devront être ramenées dans cet intervalle, avant chaque opération, de la façon suivante :

$$\begin{aligned} X_{1\ [-1;1]} &= -1 + \frac{2}{M-1} * i_{[0;M-1]} \\ X_{2\ [-1;1]} &= -1 + \frac{2}{N-1} * j_{[0;N-1]} \end{aligned} \tag{3.9}$$

Degré de la base :

Ici, on appelle *degré de la base* le degré le plus élevé des polynômes qui la constituent. Par exemple, la base suivante est de degré 2 :

	$(x_2)^0$	$(x_2)^1$	$(x_2)^2$
$(x_1)^0$	$P_{0,0}$	$P_{0,1}$	$P_{0,2}$
$(x_1)^1$	$P_{1,0}$	$P_{1,1}$	$-$
$(x_1)^2$	$P_{2,0}$	$-$	$-$

Le degré de la base est directement lié à la complexité du mouvement à caractériser. En effet, une base de degré faible (par exemple, 0 ou 1) ne permet pas de modéliser un mouvement complexe. Malheureusement, le temps de calcul et la précision ne permettent pas d'utiliser des bases de degrés trop élevés. Par exemple, la projection d'un champ de vecteur de taille 320x240 sur une base de degré 3 est très rapide (quelques millisecondes) alors que sur une base de degré 10, elle peut prendre plusieurs minutes.

Représentation des polynômes de la base :

La génération d'une base de degré 2 fournit les polynômes suivants (la taille des champs de vecteurs pour cet exemple est de 32x32) :

	$(x_2)^0$	$(x_2)^1$	$(x_2)^2$
$(x_1)^0$	$P_{0,0} = 0.03$	$P_{0,1} = 0.05\ x_2$	$P_{0,2} = -0.03 + 0.1\ (x_2)^2$
$(x_1)^1$	$P_{1,0} = 0.05\ x_1$	$P_{1,1} = 0.09\ x_1\ x_2$	$-$
$(x_1)^2$	$P_{2,0} = -0.03 + 0.1\ (x_1)^2$	$-$	$-$

Il est alors possible de représenter chacun de ces polynômes en calculant leur champ de vecteurs associé (voir chapitre *Synthèse du mouvement*).

Dans le cas présent, les figures 3.6 et 3.7 (situées à la fin de ce chapitre) ne représentent que les champs de vecteurs où $P_{\mathcal{V}}$ a été affecté avec un des polynômes de la base et $P_{\mathcal{U}} = 0$.

Exemple de combinaison linéaire :

Les différents champs de vecteurs ci-dessus représentent un polynôme de la base. Mais, pour générer des mouvements plus complexes, il est possible d'associer aux polynômes caractéristiques non plus un unique polynôme mais une combinaison linéaire de toute ou partie de ces éléments.

Par exemple, à partir de cette même base, si

$$\begin{cases} P_{\mathcal{U}} = 3\ P_{0,2} + 2\ P_{1,1} \\ P_{\mathcal{V}} = 2\ P_{0,0} - 2\ P_{1,0} \end{cases}$$

alors on obtient le mouvement représenté figure 3.1.

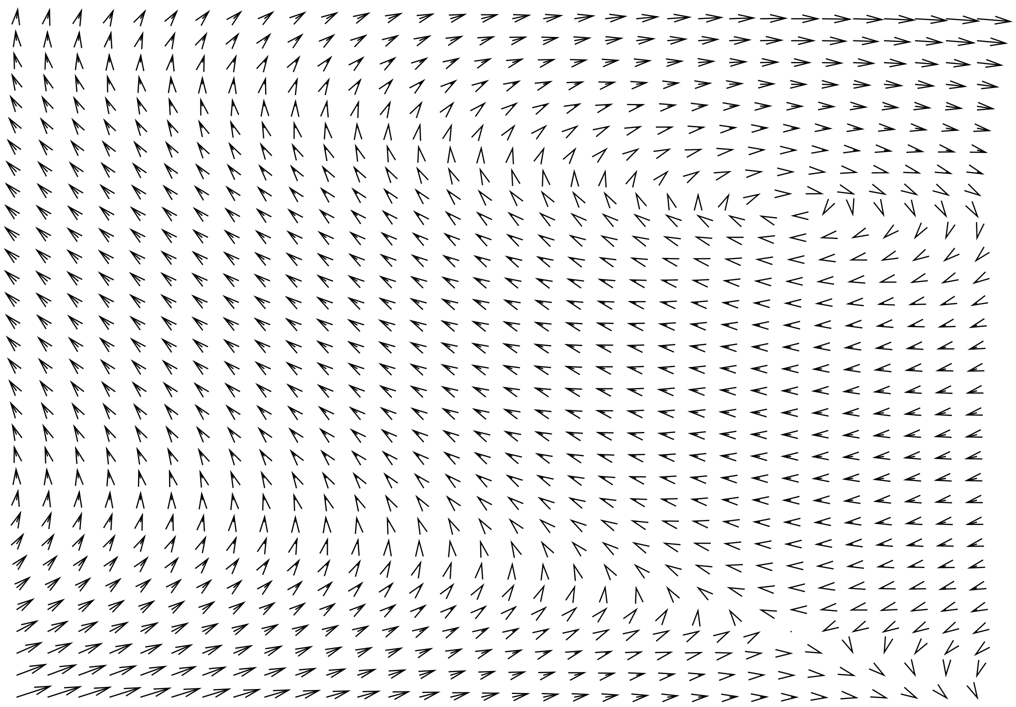


FIG. 3.1 – Exemple d'une combinaison linéaire de polynômes.

3.1.5 Autres types de bases

Durant ce stage, seuls les polynômes de Legendre ont été utilisés. Ils ont été choisis pour leur rapidité de calcul ainsi que pour la simplicité de la fonction de poids associée. De plus, ils offrent l'avantage d'être définis sur un intervalle fermé. Mais beaucoup d'autres types de polynômes existent. On peut, par exemple, citer les polynômes de Zernike, Tchebychev, Hermite, Laguerre... Il serait intéressant de comparer les bases générées à partir de ces différents polynômes pour savoir si certaines d'entre elles sont plus propices à caractériser des mouvements spécifiques.

3.2 Résistance au bruit

Avant de concevoir une méthode de reconnaissance de mouvement, il est nécessaire de tester la résistance au bruit du système mis en place.

Processus de test :

La méthode de test est la suivante (voir figure 3.2) :

1. On génère une base $\mathcal{B}$;
2. On crée les deux polynômes caractéristiques $P_{\mathcal{U}o}$ et $P_{\mathcal{V}o}$ par combinaison linéaire des différents polynômes de la base $\mathcal{B}$, les coefficients de cette combinaison étant tirés aléatoirement ;
3. On calcule le champ de vecteurs $\mathcal{C}_o$ associé à ces deux polynômes $P_{\mathcal{U}o}$ et $P_{\mathcal{V}o}$ (cette étape est expliquée en détail dans le chapitre "Reconstruction du mouvement") ;
4. On ajoute à ce champ $\mathcal{C}_o$ un bruit Gaussien $\mathcal{G}$ de moyenne nulle et d'écart-type variable pour obtenir un champ de vecteurs bruité $\mathcal{C}_b$;
5. On projette ce champ $\mathcal{C}_b$ sur la base $\mathcal{B}$ pour obtenir les deux polynômes caractéristiques correspondant au champ bruité $P_{\mathcal{U}b}$ et $P_{\mathcal{V}b}$;
6. On crée un nouveau champ de vecteurs $\mathcal{C}_r$ à partir des polynômes "bruités" $P_{\mathcal{U}b}$ et $P_{\mathcal{V}b}$;
7. La résistance au bruit est alors mesurée en comparant le champ de vecteurs initial $\mathcal{C}_o$ et le champ $\mathcal{C}_r$ calculé dernièrement. Plus ces champs sont similaires, moins le système est sensible au bruit.

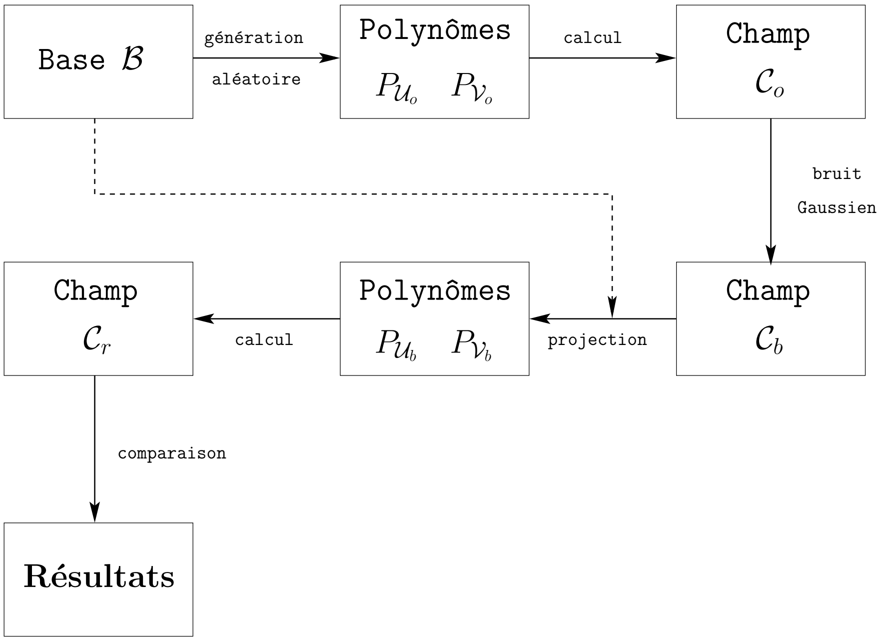


FIG. 3.2 – Processus utilisé pour le test de résistance au bruit.

Paramètres utilisées :

Les deux paramètres utilisés pour les tests sont :
- la taille de la base ;
- le SNR.

Ce dernier exprime le rapport signal sur bruit. Plus le SNR est grand moins il y a de bruit et inversement, plus le SNR tend vers 0 plus il y en a. Si le SNR est inférieur à 1, on considère, en général, que l'image est fortement bruitée.

Pour les tests, on va faire varier ce SNR dans un intervalle prédéfini (ici, entre 0.1 et 2.0 par palier de 0.1) et en déduire, à chaque fois, l'écart-type du bruit Gaussien $\mathcal{G}$ à générer par la relation :

$$\begin{aligned} \mathrm{SNR}(\mathcal{C}_o, \mathcal{C}_b) &= \frac{\mathrm{variance}(\mathcal{C}_o)}{\mathrm{variance}(\mathcal{C}_b)} \\ \Leftrightarrow \mathrm{variance}(\mathcal{C}_b) &= \frac{\mathrm{variance}(\mathcal{C}_o)}{\mathrm{SNR}} \\ \Leftrightarrow \text{écart-type}(\mathcal{G}) &= \sqrt{\mathrm{variance}(\mathcal{C}_b)} \end{aligned} \tag{3.10}$$

Mesures :

Soit (M, N) la taille du champ $\mathcal{C}_o$.

Soit $\{\vec{\nu_o}(m,n)\}_{m\in[0,M]}^{n\in[0,N]}$ l'ensemble des vecteurs du champ original $\mathcal{C}_o$.

Soit $\{\vec{\nu_r}(m,n)\}_{m\in[0,M]}^{n\in[0,N]}$ l'ensemble des vecteurs du champ résultat $\mathcal{C}_r$.

Trois mesures sont utilisées pour comparer ces deux champs :

- **L'erreur de norme** correspond à la valeur absolue de la différence entre la norme d'un vecteur $\vec{\nu_o}(m,n)$ du champ original $\mathcal{C}_o$ et la norme du vecteur $\vec{\nu_r}(m,n)$ correspondant dans le champ résultat $\mathcal{C}_r$. Elle est calculée de la façon suivante :

$$\mathcal{E}_{\text{norme}} = \frac{1}{M \text{ x } N} \sum_{m=0}^{M} \sum_{n=0}^{N} \mid \|\vec{\nu_o}(m,n)\| - \|\vec{\nu_r}(m,n)\| \mid \qquad (3.11)$$

- **L'erreur angulaire** représente la différence d'orientation (exprimée ici en degré) entre les deux vecteurs $\vec{\nu_o}(m,n)$ et $\vec{\nu_r}(m,n)$. Elle est calculée en prenant le cosinus inverse du produit scalaire des deux vecteurs normalisés :

$$\mathcal{E}_{\text{angle}} = \frac{1}{M \text{ x } N} \sum_{m=0}^{M} \sum_{n=0}^{N} cos^{-1}(\ \langle \vec{\nu_o}(m,n) \ \mid \ \vec{\nu_r}(m,n) \rangle\ ) \qquad (3.12)$$

- **Le MSE** exprime la distance entre le vecteur original $\vec{\nu_o}\begin{pmatrix} x_1^o \\ x_2^o \end{pmatrix}(m,n)$ et le vecteur résultat $\vec{\nu_r}\begin{pmatrix} x_1^r \\ x_2^r \end{pmatrix}(m,n)$. Il est calculé ainsi :

$$\text{MSE} = \frac{1}{M \text{ x } N} \sum_{m=0}^{M} \sum_{n=0}^{N} \sqrt{(x_1^o(m,n) - x_1^r(m,n))^2 + (x_2^o(m,n) - x_2^r(m,n))^2} \qquad (3.13)$$

Le MSE peut être vu comme une synthèse des deux erreurs précédentes. En effet, la notion de distance entre deux vecteurs prend en compte leur différence de norme et l'angle qu'ils forment entre eux. Il fournit donc un bilan sur la qualité des résultats mais la présence des deux autres mesures permet de savoir si l'une d'entre elles est plus sensible au bruit que l'autre.

Résultats :

Les tests sont effectués sur des champs de vecteurs de taille 320x240 (format vidéo standard). Ces champs sont générés de façon aléatoire : les coefficients des polynômes caractéristiques sont tirés selon une distribution uniforme. Le bruit est un bruit Gaussien de moyenne nulle et d'écart-type variable. Les résultats obtenus sont présentés sur la figure 3.3. Les graphiques représentent l'évolution des erreurs moyennes présentées précédemment pour le champ bruité $\mathcal{C}_b$ et le champ résultat $\mathcal{C}_r$, en fonction du degré de la base et du SNR.

A titre d'exemple, les figures 3.4 et 3.5 montrent la qualité de reconstruction d'un champ de vecteur bruité. Dans les deux cas, la base est de degré 3 et les champs de vecteurs sont de dimension 64x64.

On constate que le MSE est relativement stable, même si l'on a un SNR relativement faible, et cela, quelle que soit la taille de la base. La principale conclusion que l'on peut tirer de ces résultats est que **le système est peu sensible au bruit**. Les deux autres courbes montrent que la reconstruction permet de réduire, de façon aussi efficace, les erreurs de norme et d'angle.

De plus, ces résultats apportent une nouvelle idée de recherche : comme le système n'est apparemment pas sensible au bruit Gaussien, on peut se demander si les étapes de lissage, qui peuvent être très longues, sont encore nécessaires. A l'heure actuelle, cette piste n'a pas encore été approfondie mais elle pourrait donner lieu à quelques travaux très prochainement.

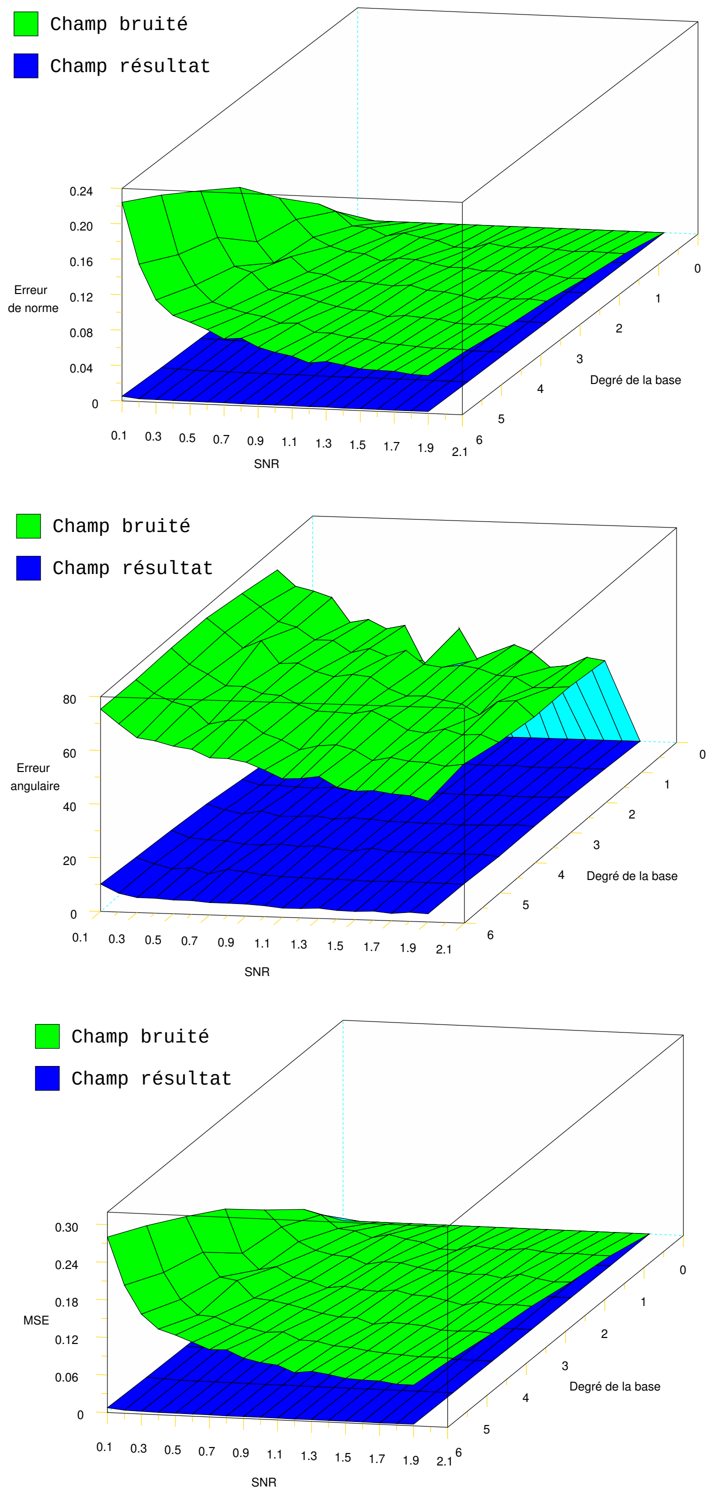


FIG. 3.3 – Influence du bruit sur le système mis en place.

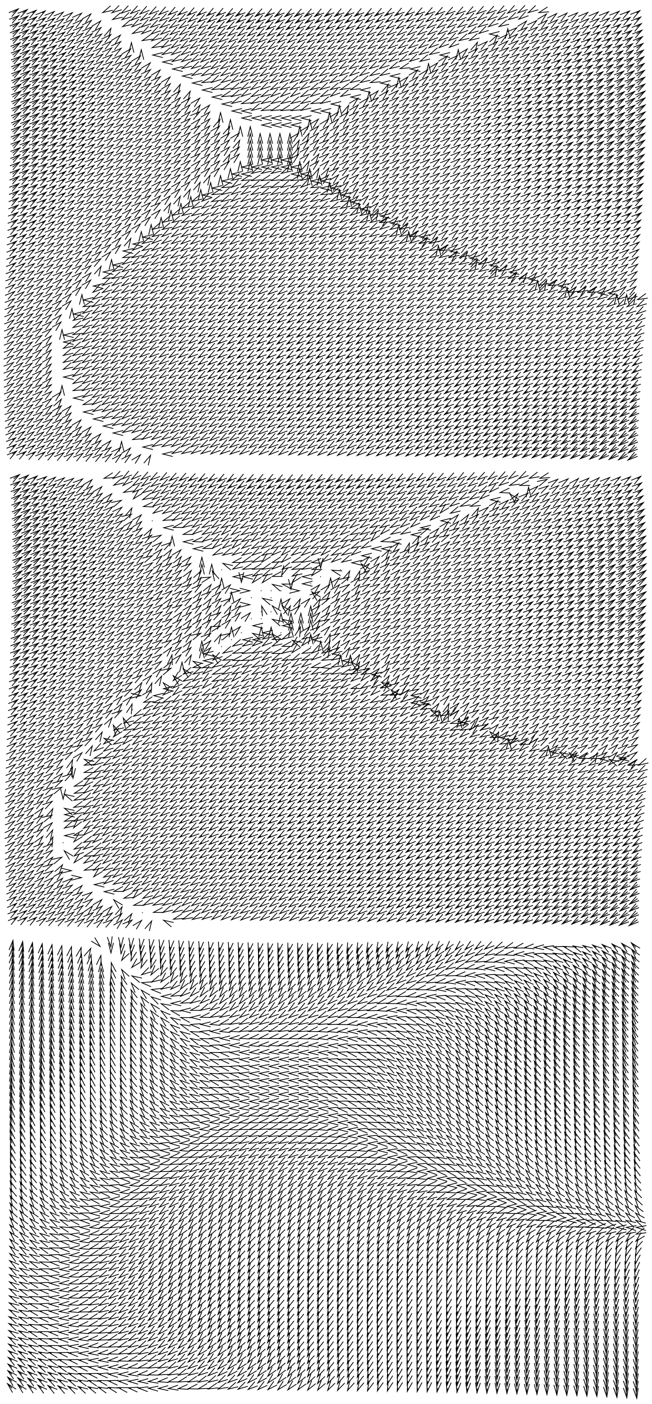


FIG. 3.4 – Reconstruction d'un champ de vecteurs faiblement bruité ($\sigma = 0.01$) dont les valeurs sont comprises entre -1 et 1. De haut en bas : champ original, champ bruité et champ résultat.

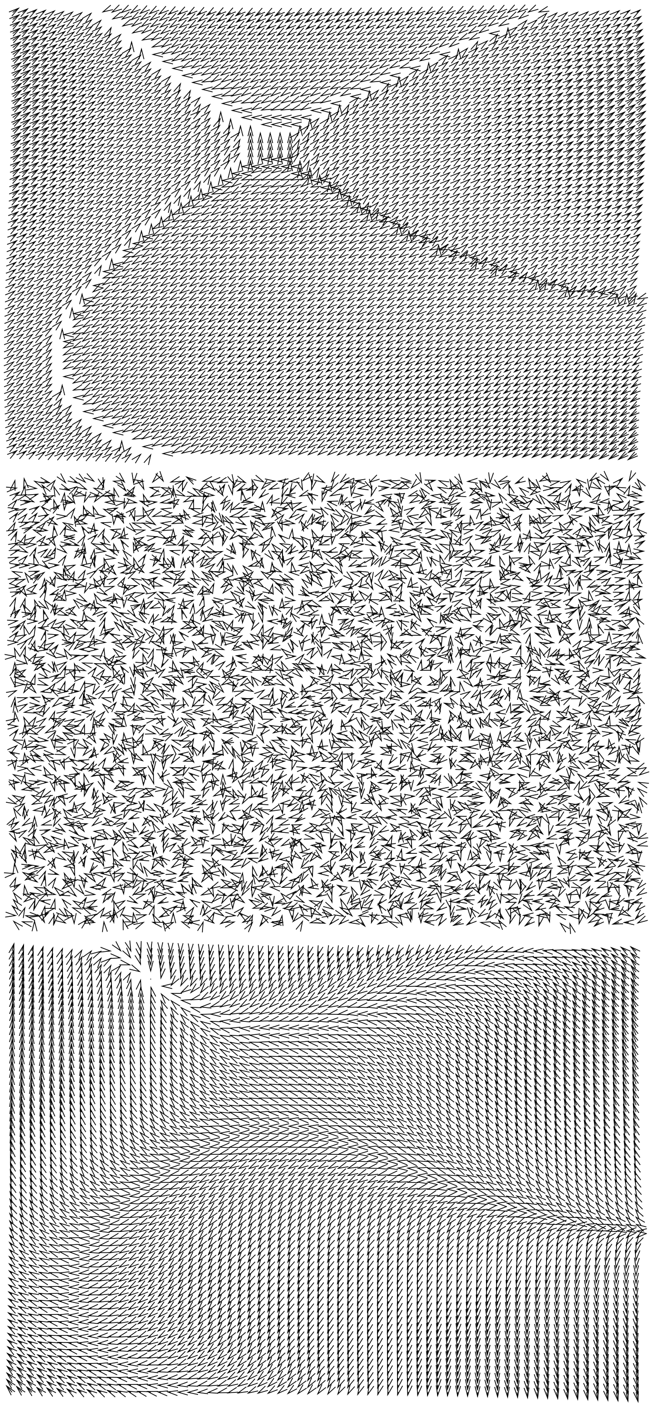
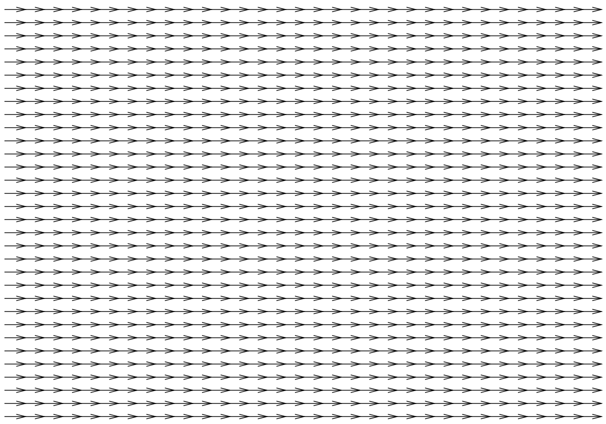
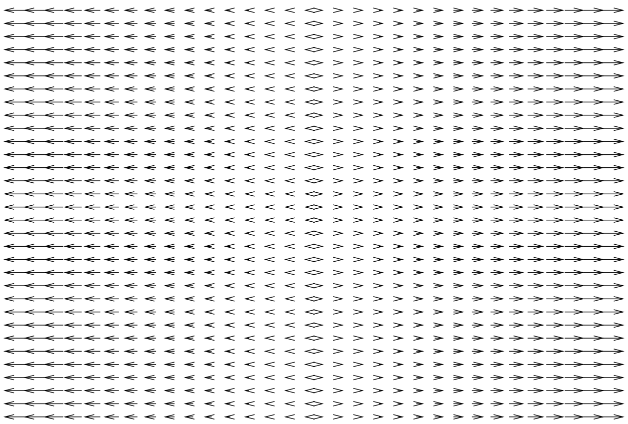


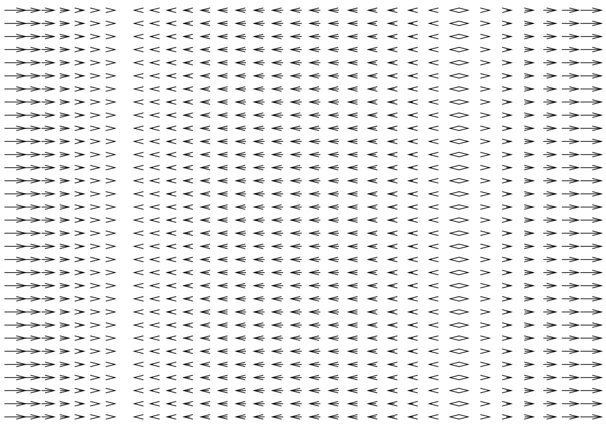
FIG. 3.5 – Reconstruction d'un champ de vecteurs fortement bruité ($\sigma = 1.0$) dont les valeurs sont comprises entre -1 et 1. De haut en bas : champ original, champ bruité et champ résultat.



$P_{\mathcal{U}} = 0 \quad P_{\mathcal{V}} = P_{0,0}$

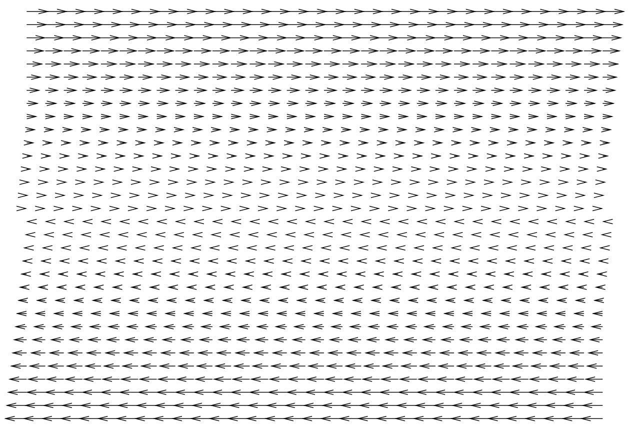


$P_{\mathcal{U}} = 0 \quad P_{\mathcal{V}} = P_{1,0}$

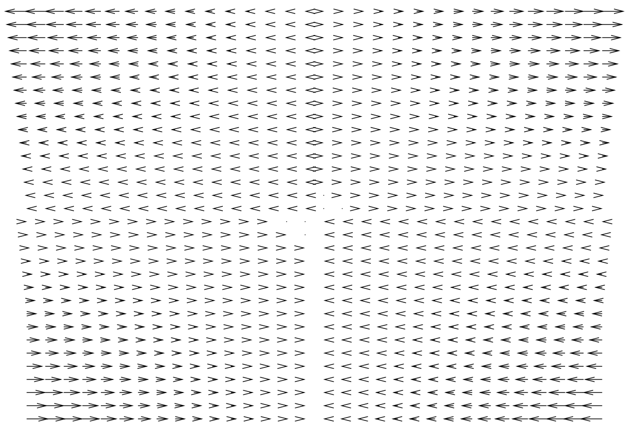


$P_{\mathcal{U}} = 0 \quad P_{\mathcal{V}} = P_{2,0}$

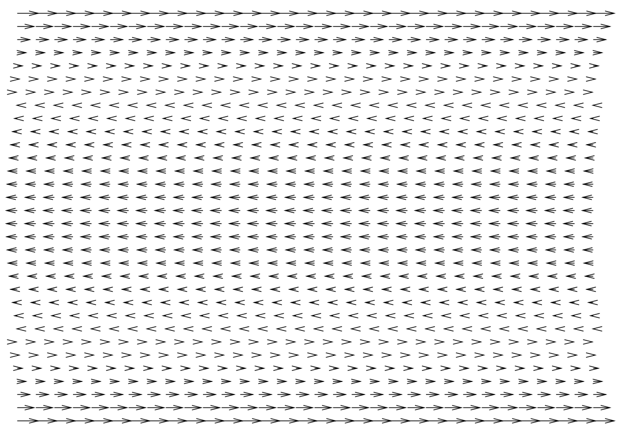
FIG. 3.6 – Ensemble des champs de vecteurs d'une base de degré 2.



$P_{\mathcal{U}} = 0 \quad P_{\mathcal{V}} = P_{0,1}$



$P_{\mathcal{U}} = 0 \quad P_{\mathcal{V}} = P_{1,1}$



$P_{\mathcal{U}} = 0 \quad P_{\mathcal{V}} = P_{0,2}$

FIG. 3.7 – Ensemble des champs de vecteurs d'une base de degré 2 (suite et fin).

Chapitre 4

Reconnaissance du mouvement

4.1 Principe

Le chapitre précédent explique comment exprimer le mouvement contenu dans une image par l'intermédiaire des polynômes caractéristiques. Mais ceux-ci ne contiennent que l'information à un instant donné. Or, les mouvements à étudier peuvent se dérouler sur un intervalle de temps relativement long. On ne peut donc pas se restreindre à l'étude d'une seule image.

La reconnaissance d'un mouvement se fait donc en étudiant l'évolution dans le temps des coefficients des polynômes caractéristiques.

4.2 Expérimentations

Un exemple concret est maintenant présenté : nous allons déterminer les mouvements contenus dans une séquence en utilisant tous les outils vus jusqu'à présent.

4.2.1 Données

La vidéo, acquise par une webcam, montre le visage d'une personne. Celle-ci tourne la tête à gauche, à droite et vers le bas de façon aléatoire, sans comporter de séquence prédéfinie. La vidéo est au format 320x240. Elle est composée de 1050 images. La figure 4.1 montre le contenu de cette vidéo à deux instants précis.

FIG. 4.1 – À gauche : position initiale. À droite : mouvement vers la gauche.

4.2.2 Processus utilisé :

Le processus de reconnaissance est le suivant :

1. à partir de la séquence originale, on extrait tous les champs de vecteurs ;

2. pour chacun de ces champs, on calcule les polynômes caractéristiques par projections sur la base ;

3. on étudie les variations dans le temps des coefficients de ces polynômes.

Afin de déterminer si certains coefficients portent plus d'informations que d'autres, une *analyse en composantes principales* est effectuée.

4.2.3 Résultats :

Le repère utilisé est le suivant :
La base utilisée est de degré 2. Elle est de la forme :

	$(x_2)^0$	$(x_2)^1$	$(x_2)^2$
$(x_1)^0$	$P_{0,0}$	$P_{0,1}$	$P_{0,2}$
$(x_1)^1$	$P_{1,0}$	$P_{1,1}$	–
$(x_1)^2$	$P_{2,0}$	–	–

On obtient donc, pour chaque image, six coefficients pour le polynôme $P_{\mathfrak{U}}$ et

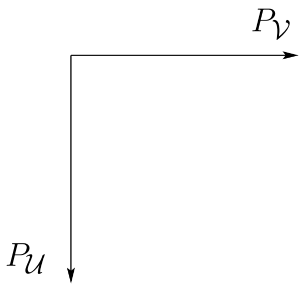


FIG. 4.2 – Repère utilisé.

six coefficients pour le polynôme $P_{\mathcal{V}}$.

Pour déterminer si ces coefficients sont corrélés, nous faisons une *analyse en composantes principales*. Pour simplifier les explications, seule l'étude des coefficients du polynôme $P_{\mathcal{V}}$ est détaillée ici.

Dans un premier temps, nous construisons le tableau de données ayant la structure suivante :

$D(P_{\mathcal{V}}) =$

	image 1	image 2	...	image 1050
$P_{\mathcal{V}(0,0)}$				
$P_{\mathcal{V}(1,0)}$				
$P_{\mathcal{V}(2,0)}$				
$P_{\mathcal{V}(0,1)}$				
$P_{\mathcal{V}(1,1)}$				
$P_{\mathcal{V}(0,2)}$				

Ensuite, nous centrons ces données pour obtenir la matrice $D_c(P_{\mathcal{V}})$.

Partant du principe que toutes les images ont la même importance, les coefficients de la matrice de poids sont identiques. Sa taille est de 1050x1050 et elle est de la forme :

$$M = \begin{pmatrix} \frac{1}{1050} & 0 & \dots & 0 \\ 0 & \frac{1}{1050} & \dots & 0 \\ \vdots & \vdots & \ddots & \dots \\ 0 & 0 & \dots & \frac{1}{1050} \end{pmatrix}$$

De là, nous pouvons calculer la matrice de covariance de la façon suivante :

$$Cov(P_{\mathcal{V}}) = D_c(P_{\mathcal{V}}) * M * (D_c(P_{\mathcal{V}}))^T$$

A partir de cette matrice, nous calculons ses valeurs propres $\lambda_1(P_{\mathcal{V}}) \dots \lambda_6(P_{\mathcal{V}})$ et ses vecteurs propres associés $\vec{v_1}(P_{\mathcal{V}}) \dots \vec{v_6}(P_{\mathcal{V}})$. L'étude de ces valeurs propres, triées par ordre décroissant, permet de savoir si certains coefficients portent plus d'informations que d'autres. Pour cela, nous calculons les contributions de chaque coefficient de la façon suivante :

$$\text{contribution } = \sum_{i=1}^{6} \quad \frac{\lambda_i(P_{\mathcal{V}})}{\lambda_1(P_{\mathcal{V}}) + \ldots + \lambda_6(P_{\mathcal{V}})}$$

Nous obtenons les résultats suivants :

valeur propre	contribution
$\lambda_1(P_{\mathcal{V}})$	94.37 %
$\lambda_2(P_{\mathcal{V}})$	4.17 %
$\lambda_4(P_{\mathcal{V}})$	0.81 %
$\lambda_6(P_{\mathcal{V}})$	0.36 %
$\lambda_5(P_{\mathcal{V}})$	0.22 %
$\lambda_3(P_{\mathcal{V}})$	0.07 %

On constate qu'une très grande partie de l'information est portée par un seul axe. On effectue alors la projection de tous les coefficients sur l'axe factoriel $\vec{E}(P_{\mathcal{V}})$ fournit par le vecteur propre $\vec{v_1}(P_{\mathcal{V}})$.

De la même façon, nous calculons les contributions de chaque coefficient du polynôme $P_{\mathcal{U}}$ et nous obtenons les résultats suivants :

valeur propre	contribution
$\lambda_1(P_{\mathcal{U}})$	82.95 %
$\lambda_2(P_{\mathcal{U}})$	17.05 %
$\lambda_3(P_{\mathcal{U}})$	0.00 %
$\lambda_4(P_{\mathcal{U}})$	0.00 %
$\lambda_5(P_{\mathcal{U}})$	0.00 %
$\lambda_6(P_{\mathcal{U}})$	0.00 %

Les résultats sont relativement similaires : une grande partie de l'information est portée par un seul axe factoriel $\vec{E}(P_{\mathcal{U}})$ fournit par le vecteur propre $\vec{v_1}(P_{\mathcal{U}})$.

Dans notre cas, étudier le mouvement de la séquence revient donc à étudier l'évolution des coefficients après projection sur l'axe $\vec{E}(P_{\mathcal{U}})$ et l'axe $\vec{E}(P_{\mathcal{V}})$ (voir figure 4.3).

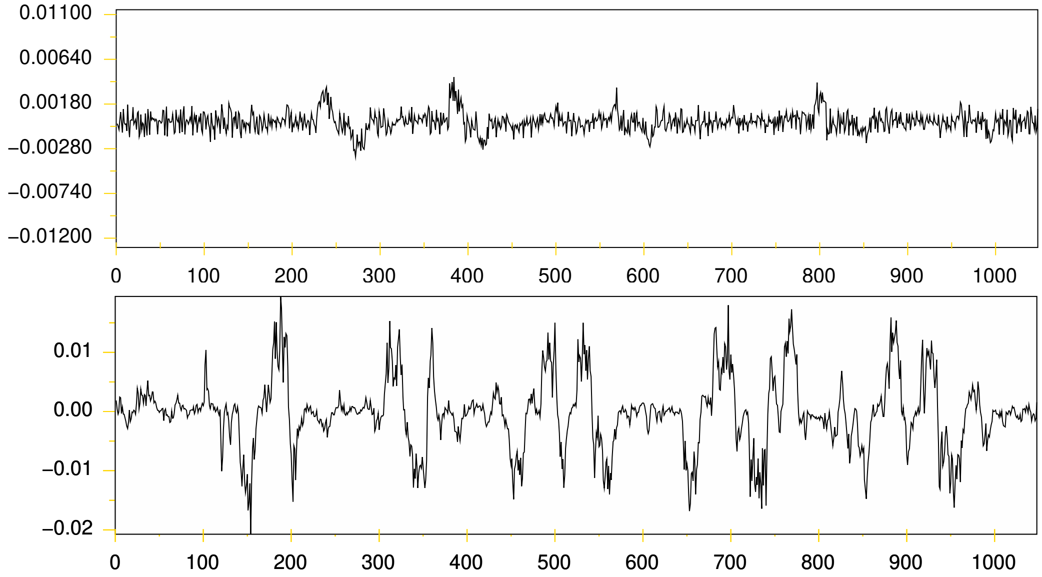


FIG. 4.3 – En haut : projection des coefficients du polynôme $P_{\mathcal{U}}$ sur l'axe factoriel $\vec{E}(P_{\mathcal{U}})$. En bas : projection des coefficients du polynôme $P_{\mathcal{V}}$ sur l'axe factoriel $\vec{E}(P_{\mathcal{V}})$.

4.2.4 Interprétation :

Pour interpréter la signification de ces courbes, il faut commencer par étudier les valeurs des deux vecteurs propres $\vec{v_1}(P_{\mathcal{U}})$ et $\vec{v_1}(P_{\mathcal{V}})$ utilisés pour la projection. Ces valeurs sont les suivantes :

$$\vec{v_1}(P_{\mathcal{U}}) = \begin{pmatrix} 0.8869693 \\ 0 \\ 0 \\ 0 \\ -0.4618283 \\ 0 \end{pmatrix} \qquad \vec{v_1}(P_{\mathcal{V}}) = \begin{pmatrix} 0.5968097 \\ -0.0014940 \\ -0.2901533 \\ -0.0125158 \\ 0.0168392 \\ -0.7477879 \end{pmatrix}$$

Si on calcule les deux champs de vecteurs $\mathcal{C}_{P_{\mathcal{U}}}$ et $\mathcal{C}_{P_{\mathcal{V}}}$ en faisant une combinaison linéaire des polynômes P de la base pondérés par les valeurs des deux vecteurs propres $\vec{v_1}(P_{\mathcal{U}})$ et $\vec{v_1}(P_{\mathcal{V}})$, on obtient les champs représentés figure 4.4 (pour pouvoir être insérés dans ce rapport, les champs de vecteurs ont été réduits en 40x30).

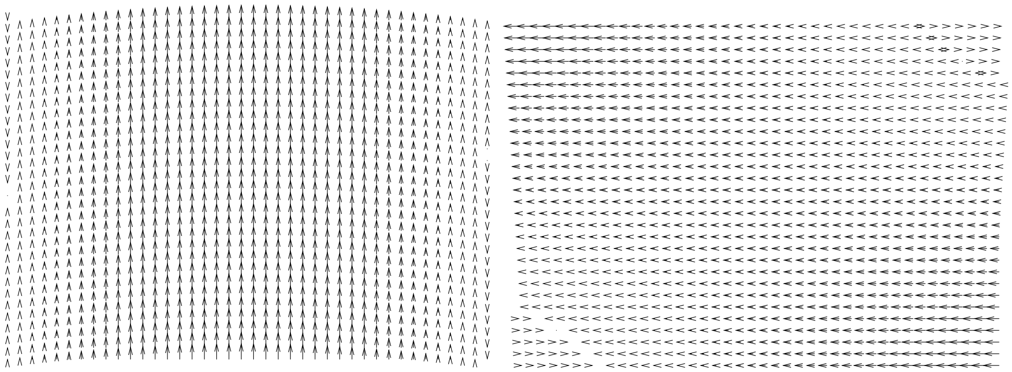


FIG. 4.4 – À gauche : champ obtenu par une combinaison linéaire des polynômes de la base avec le vecteur propre $\vec{E}(P_{\mathcal{U}})$. À droite : champ obtenu par une combinaison linéaire des polynômes de la base avec le vecteur propre $\vec{E}(P_{\mathcal{V}})$.

On constate que la plupart des vecteurs du champ $\mathcal{C}_{P_{\mathcal{U}}}$ est orientée verticalement vers le haut. Un pic sur la courbe se traduit alors par un mouvement vers le haut et une chute des coefficients se traduit par un mouvement de tête vers le bas. Le même raisonnement peut s'appliquer au champ $\mathcal{C}_{P_{\mathcal{V}}}$: un pic correspond à un mouvement vers la gauche et un creux à un mouvement vers la droite.

Ces courbes représentent les mouvements effectués dans le temps et non la position de la tête par rapport à la position de référence. Par exemple, sur la figure 4.3, un creux suivi d'un pic ne veut pas dire que la personne tourne la tête à droite puis à gauche mais qu'elle effectue un mouvement vers la droite puis vers la gauche. Si sa position d'origine est face à la caméra, cette suite de mouvements peut vouloir dire qu'elle a tourné la tête vers la droite puis qu'elle a effectué un mouvement vers la gauche pour revenir à sa position initiale, sans tourner la tête vers la gauche.

Comme en cinématique le vecteur vitesse est la dérivée du vecteur position, pour étudier la position de la tête au cours du temps, il faut calculer l'intégrale des deux courbes précédentes. On obtient alors les courbes représentées par la figure 4.5.

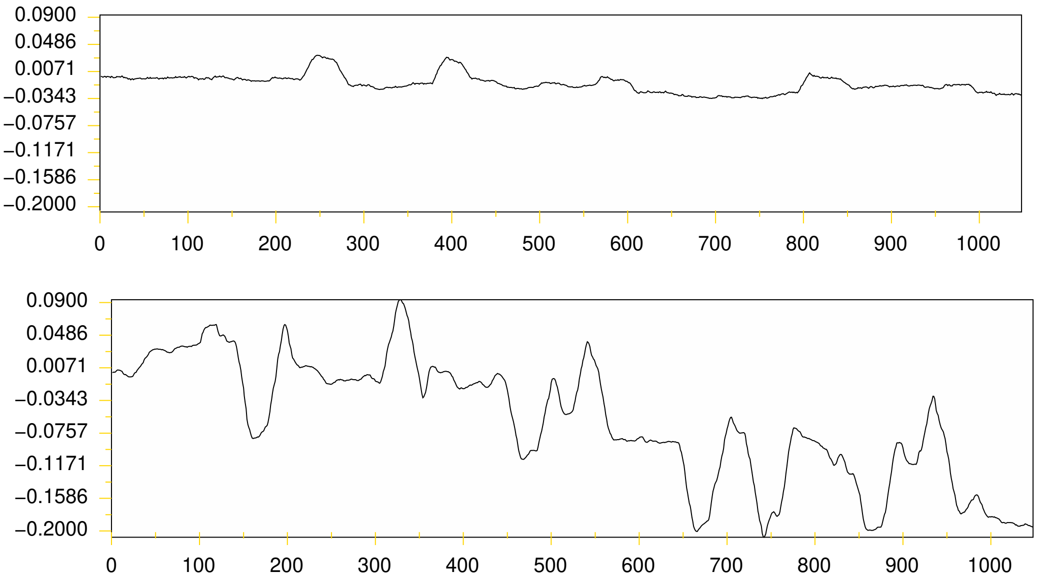


FIG. 4.5 – Représentation de la position de la tête, au cours du temps. En haut : position verticale. En bas : position horizontale.

La figure 4.6 représente la séquence à un instant précis. Les deux courbes sont orientées de façon à suivre le mouvement plus facilement. Par exemple, lorsque la courbe de droite subit une variation vers la gauche, cela signifie que la tête de la personne est orientée vers la gauche.

4.2.5 Conclusion :

Le système mis en place permet de reconnaître le mouvement de la séquence. Il serait maintenant intéressant de le tester sur de nouveaux mouvements tels que des rotations, des zooms, des panoramiques, …

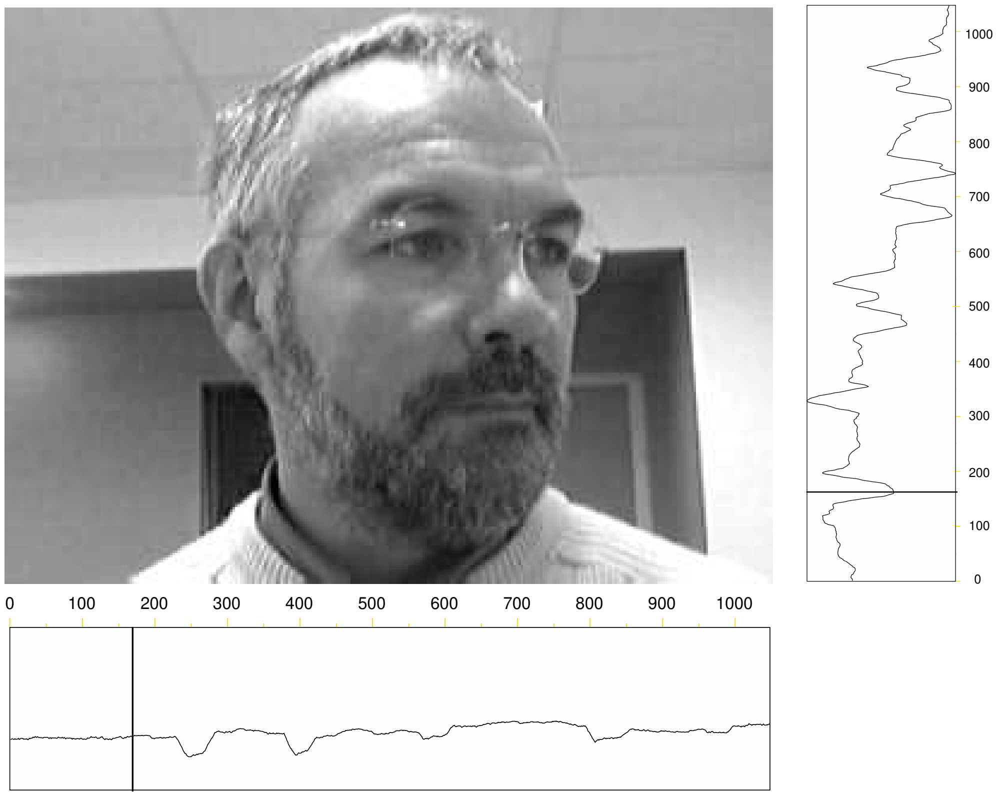


FIG. 4.6 – Exemple de variations des poids sur un mouvement vers la droite.

Les courbes affichées soulèvent également une question : est-il possible d'associer une grandeur physique réelle, telle que le mètre, au mouvement effectué à partir de la valeur des poids ?

Chapitre 5

Synthèse du mouvement

La synthèse peut être vue comme l'opération inverse de la reconnaissance. Ici, on connaît l'expression du mouvement. On cherche alors à reconstruire l'animation correspondante. Pour cela, on calcule, pour chaque image de la séquence à générer, les polynômes caractéristiques $P_{\mathcal{U}}$ et $P_{\mathcal{V}}$ correspondant. Puis, pour chaque vecteur du champ, on calcule ses deux composantes en fixant les deux variables des polynômes caractéristiques en fonction de sa position dans le champ :

$$\forall i, \forall j \in [-1, 1] \left\{ \begin{array}{l} \mathcal{U}(i,j) = P_{\mathcal{U}}(i,j) \\ \mathcal{V}(i,j) = P_{\mathcal{V}}(i,j) \end{array} \right.$$

Chaque image de l'animation est alors calculée en déplaçant les pixels de l'image précédente selon les vecteurs du champ. Cette technique peut être utilisée pour différentes applications. Une des plus évidente concerne la simulation du déplacement d'un objet donc le mouvement est connu. Mais elle pourrait également être utilisée pour faire de la compression. En effet, il suffirait alors de ne garder que l'image de départ et l'ensemble des polynômes caractéristiques.

N'ayant pas eu l'occasion d'approfondir ce sujet, aucun résultat ne peut être donné ici.

Quatrième partie

Conclusion et perspectives

Conclusion

L'objectif du stage était d'étudier la possibilité de caractériser tout type de mouvement comme une combinaison linéaire de polynômes issus d'une base orthonormée. Nous avons vu que la méthode mise en place permettait de reconnaître des mouvements simples. De plus, cette méthode est robuste au bruit.

Dès le début du stage, des problèmes liés aux traitements de vidéos sont apparus. Ces problèmes peuvent être classés en deux catégories :

- **Problèmes techniques :** ils sont principalement liés à la structure même des ordinateurs ou de l'électronique. On peut citer, par exemple, les problèmes liés à la puissance de calcul, à la capacité de stockage, au flot important de données à traiter, aux bruits d'acquisition ou de compression...

- **Problèmes d'interprétation :** ces problèmes sont liés aux contenus des séquences tels que la détection de contours, l'étude d'objets déformables comme le corps humain, les mouvements de caméra, l'occultation d'objets qui peut créer une discontinuité du mouvement de l'objet caché...

Tous ces problèmes m'ont vite fait comprendre qu'en traitement du signal aucun "processus-type" ne peut être établi mais qu'il faut savoir choisir les bons outils en fonction du problème posé.

Ce travail pose les bases d'une éventuelle méthode de reconnaissance de mouvement. Mais beaucoup de travaux doivent encore être effectués avant d'obtenir un système complet et fiable.

En effet, une phase importante qui pourrait être effectuée prochainement serait de tester la méthode sur des mouvements plus complexes.

Il pourrait également être intéressant d'utiliser, pour la génération de la base, d'autres polynômes que ceux de Legendre.

De même, l'utilisation d'un seul polynôme caractéristique complexe à la place des deux polynômes $P_{\mathfrak{U}}$ et $P_{\mathcal{V}}$ réels permettrait de compacter les différentes équations et donc de simplifier le processus de reconnaissance. De plus, l'étude d'un seul polynôme pourrait faire apparaître plus facilement les résultats théoriques.

Il pourrait aussi être intéressant de créer une base de mouvements-types pour identifier automatiquement les déplacements.

Des travaux plus approfondis sur la synthèse permettrait d'étudier la possibilité d'utiliser le système pour faire de la compression.

Bibliographie

[1] J.L. Barron, D.J. Fleet, and S.S. Beauchemin. Performance of Optical Flow Techniques. *International Journal of Computer Vision*, 12(1) :43–77, 1994.

[2] C. Cedras and M. Shah. Motion-based recognition : A survey. *IVC*, 13(2) :129–155, March 1995.

[3] B. Galvin, B. McCane, K. Novins, D. Mason, and S. Mills. Recovering motion fields : An evaluation of eight optical flow algorithms. In *British Machine Vision Conference*, 1998.

[4] B.K.P. Horn and B.G. Schunck. Determining optical flow. *Artificial IntelligenceI*, 17(1-3) :185–203, August 1981.

[5] F.B. Lauze, P. Kornprobst, C. Lenglet, R. Deriche, and M. Nielsen. Sur quelques méthodes de calcul de flot optique à partir du tenseur de structure : Synthèse et contribution. 2004.

[6] B.D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *DARPA81*, pages 121–130, 1981.

[7] Carman Neustaedter. An evaluation of optical flow using lucas and kanade's algorithm. 2002.

[8] H. Scharr and J. Weickert. An anisotropic diffusion algorithm with optimized rotation invariance, 2000.

[9] Hagen Spies and Hanno Scharr. Accurate optical flow in noisy image sequences. pages 587–592.