CR TP Flot optique – Séquences Vidéo

Analyse du mouvement et du flot optique dans des séquences vidéo

Wassim Chikhi

Master 2 Vision et Machine Intelligente – 2025/2026

Le notebook Python utilisé pour ce TP est disponible sur Kaggle : https://www.kaggle.com/code/wassmed/tp-2-seqvid-v1

1. Objectifs du TP

L'objectif de ce TP est d'explorer expérimentalement les notions de **mouvement apparent** et de **flot optique** présentées au cours *Séquences Vidéo* : à partir de séquences d'images, il s'agit d'estimer et de visualiser le mouvement des structures visibles dans la scène.

Plus précisément, on cherche à :

- illustrer le lien entre **variation spatio-temporelle de luminosité** et mouvement à travers l'équation de conservation de l'illumination ;
- mettre en œuvre une détection de **points d'intérêt** adaptée au suivi (GoodFeaturesToTrack / Shi–Tomasi) à partir du gradient spatial ;
- implémenter et comparer une méthode de **flot optique épars** (Lucas–Kanade pyramidal) et une méthode de **flot optique dense** (Farnebäck) sous OpenCV ;
- étudier l'impact des **paramètres algorithmiques** (taille de fenêtre, niveaux de pyramide, qualité des points, etc.) sur la qualité des estimations ;
- analyser de manière critique les résultats obtenus sur des **vidéos personnelles** et des **vidéos fournies** (*Séquences routières, scènes urbaines, objets simples*) en lien avec les limites théoriques : problème de l'ouverture, surfaces peu texturées, changement d'illumination.

L'idée générale est de passer d'une séquence d'images considérée comme une simple succession de textures à une représentation explicite sous forme de **champs de vecteurs** décrivant le mouvement apparent dans le plan image (flot optique).

2. Données et outils

2.1. Séquences vidéo utilisées

Deux types de séquences ont été utilisées :

- **Vidéos personnelles**, filmées au smartphone (format `.mp4`) :
  - une scène d'intérieur avec un objet se déplaçant sur un fond presque fixe ;
  - une marche dans un couloir (mouvement de caméra dominant, scène rigide).
- **Vidéos fournies par l'enseignant** via la plateforme Helios :
  - séquences issues de scènes routières et urbaines (*Paris street view*) ;
  - séquences plus rapides (véhicules, piétons) présentant plusieurs objets en mouvement simultanément.

Ces vidéos représentent des cas typiques discutés en cours : translation de la caméra avec environnement rigide, mouvement d'objets indépendants, fonds relativement uniformes mais aussi zones fortement texturées.

2.2. Environnement logiciel

Toutes les manipulations ont été réalisées dans un notebook **Jupyter** (Python 3) avec :

- `numpy` pour les calculs matriciels et la gestion de tableaux ;
- `opencv-python` (`cv2`) pour :
  - la lecture de vidéos (`VideoCapture`),
  - la conversion couleur → niveaux de gris,
  - la détection de coins (`goodFeaturesToTrack`),

  - le calcul du flot optique épars (`calcOpticalFlowPyrLK`),
  - le calcul du flot optique dense (`calcOpticalFlowFarneback`).
- `matplotlib` pour la visualisation des résultats dans le notebook (remplacement de `cv2.imshow`, incompatible avec certains environnements).

Les images illustrant les résultats sont automatiquement sauvegardées via `cv2.imwrite` dans un dossier dédié `resultats_tp_seqvid` afin d'être intégrées facilement dans ce rapport.

3. Méthodologie

3.1. Rappel théorique minimal

Champ de mouvement et flot optique

On distingue le **champ de mouvement** réel (projection du mouvement 3D des objets sur le plan image) et le **flot optique**, défini comme le mouvement apparent des motifs lumineux en image. Sous les hypothèses de surfaces lambertiennes et d'illumination constante, la **conservation de l'illumination** s'écrit :

$$\frac{dE(x(t), y(t), t)}{dt} = \frac{\partial E}{\partial x}\frac{dx}{dt} + \frac{\partial E}{\partial y}\frac{dy}{dt} + \frac{\partial E}{\partial t} = 0,$$

ce qui conduit à l'équation classique du flot optique

$$E_x u + E_y v + E_t = 0$$

où (u, v) est le vecteur déplacement local (flot optique), E_x et E_y les dérivées spatiales, E_t la dérivée temporelle.

Cette équation ne fournit qu'une **contrainte scalaire** pour deux inconnues (u, v) : c'est le *problème de l'ouverture*. On ne peut estimer que la composante du mouvement *dans la direction du gradient spatial*. Il faut donc exploiter soit un voisinage spatial plus large (approche différentielle locale), soit des contraintes de régularité globale.

Points d'intérêt pour le suivi

Comme vu au cours, les points d'une image se répartissent grossièrement en :

— zones **uniformes** (gradient nul ou très faible) ;
— **bords** simples (gradient fort dans une seule direction) ;
— **coins** ou *points d'intérêt* (gradient fort dans deux directions).

Les coins sont les plus adaptés au suivi par flot optique local, car ils permettent de lever partiellement le problème de l'ouverture. L'algorithme **GoodFeaturesToTrack** (Shi–Tomasi) sélectionne justement les points où la matrice des gradients possède deux valeurs propres élevées.

3.2. Pipeline général du TP

Le pipeline mis en œuvre dans le notebook se décompose comme suit :

1. **Chargement d'une vidéo** et extraction de la première frame ;
2. **Conversion en niveaux de gris** pour toutes les étapes d'analyse ;
3. **Détection de coins** sur la première image avec GoodFeaturesToTrack ;
4. **Calcul du flot optique épars** par Lucas–Kanade pyramidal pour suivre ces coins au fil des images ;
5. **Calcul du flot optique dense** par Farnebäck sur la même séquence ;
6. **Visualisation et sauvegarde** des champs de vecteurs :
   — flèches superposées à l'image (Lucas–Kanade) ;
   — champ dense encodé en couleur (Farnebäck, représentation HSV).

Le choix d'utiliser la même vidéo pour les deux méthodes permet une comparaison visuelle directe (même contenu, même type de mouvements).

3.3. Détection de coins avec GoodFeaturesToTrack

La fonction `cv2.goodFeaturesToTrack` est utilisée sur la première image en niveaux de gris. Les principaux paramètres manipulés sont :

— `maxCorners` : nombre maximal de coins retournés ; on a testé des valeurs entre 100 et 500 pour couvrir suffisamment la scène sans saturer l'affichage ;
— `qualityLevel` : seuil relatif (entre 0 et 1) appliqué sur la mesure de "qualité" du coin ; un seuil bas (0,01) conserve plus de points mais inclut des coins plus fragiles ;
— `minDistance` : distance minimale entre deux coins ; en pratique, fixée autour de 7–10 pixels pour éviter une forte redondance locale ;
— `blockSize` : taille du voisinage utilisé pour calculer les gradients ; des valeurs autour de 7 pixels offrent un bon compromis entre bruit et précision ;

— `useHarrisDetector` : permet de basculer entre critère de Shi–Tomasi (par défaut) et détecteur de Harris classique.

Les coins détectés sont superposés à l'image et sauvegardés dans un fichier `gftt_frame0.png` (Figure 1).

3.4. Flot optique épars : Lucas–Kanade pyramidal

Le flot optique épars est calculé avec `cv2.calcOpticalFlowPyrLK`, en prenant comme points d'entrée les coins détectés. L'algorithme Lucas–Kanade repose sur l'approximation d'un mouvement *constant* dans une petite fenêtre, et sur une résolution aux moindres carrés de l'équation de conservation de l'illumination.

Les paramètres principaux sont :

— `winSize` : taille de la fenêtre (ex. 21×21) dans laquelle on suppose le mouvement local constant ;
— `maxLevel` : nombre de niveaux de la pyramide d'images (typiquement 3) ; permet de traiter des mouvements plus grands que quelques pixels ;
— `criteria` : critère d'arrêt de l'itération (nombre max d'itérations et seuil de convergence).

Pour chaque paire d'images consécutives, l'algorithme renvoie les nouvelles positions des points, un vecteur *status* permettant d'identifier les points correctement suivis, et une estimation d'erreur. On ne conserve que les points pour lesquels `status == 1`. Le déplacement est visualisé sous forme de flèches vertes (originant à la position précédente, pointant vers la position suivante). Des exemples d'images annotées sont sauvegardés sous la forme `lk_frame_XXXX.png` (Figure 2).

3.5. Flot optique dense : méthode de Farnebäck

Le flot optique dense est calculé avec `cv2.calcOpticalFlowFarneback`. Cette méthode modélise localement la variation du niveau de gris par des polynômes quadratiques, ce qui permet d'approximer le champ de mouvement dans chaque voisinage.

Les paramètres testés incluent :

— `pyr_scale` : facteur de réduction entre niveaux de pyramide (ex. 0,5) ;
— `levels` : nombre de niveaux dans la pyramide (ex. 3) ;
— `winsize` : taille de la fenêtre de moyenne locale (ex. 15) ;
— `iterations` : nombre d'itérations par niveau (ex. 3) ;
— `poly_n` : taille du voisinage pour le modèle polynomial (ex. 5) ;
— `poly_sigma` : écart-type de la gaussienne de pré-lissage (ex. 1,2).

Le flot résultant est un tableau de vecteurs (u, v) par pixel. Pour la visualisation, on le convertit dans l'espace **HSV** :

— la teinte (*Hue*) encode la **direction** du mouvement ;
— la valeur (*Value*) encode la **norme** du déplacement ;
— la saturation est fixée à 255 pour avoir des couleurs bien marquées.

Les images obtenues sont sauvegardées en `farneback_frame_XXXX.png` (Figure 3).

4. Résultats expérimentaux

4.1. Distribution des points d'intérêt

La Figure 1 montre les points détectés sur la première image d'une séquence de marche dans un couloir. On constate que :

— les coins se concentrent naturellement sur les zones texturées : arêtes de portes, objets sur les murs, motifs du sol ;
— très peu de points sont retenus dans les zones uniformes (murs lisses) ;
— la variation de `qualityLevel` permet de contrôler l'équilibre entre densité de points et robustesse du suivi : un seuil trop élevé ne garde que quelques points, un seuil trop faible introduit des coins peu discriminants.

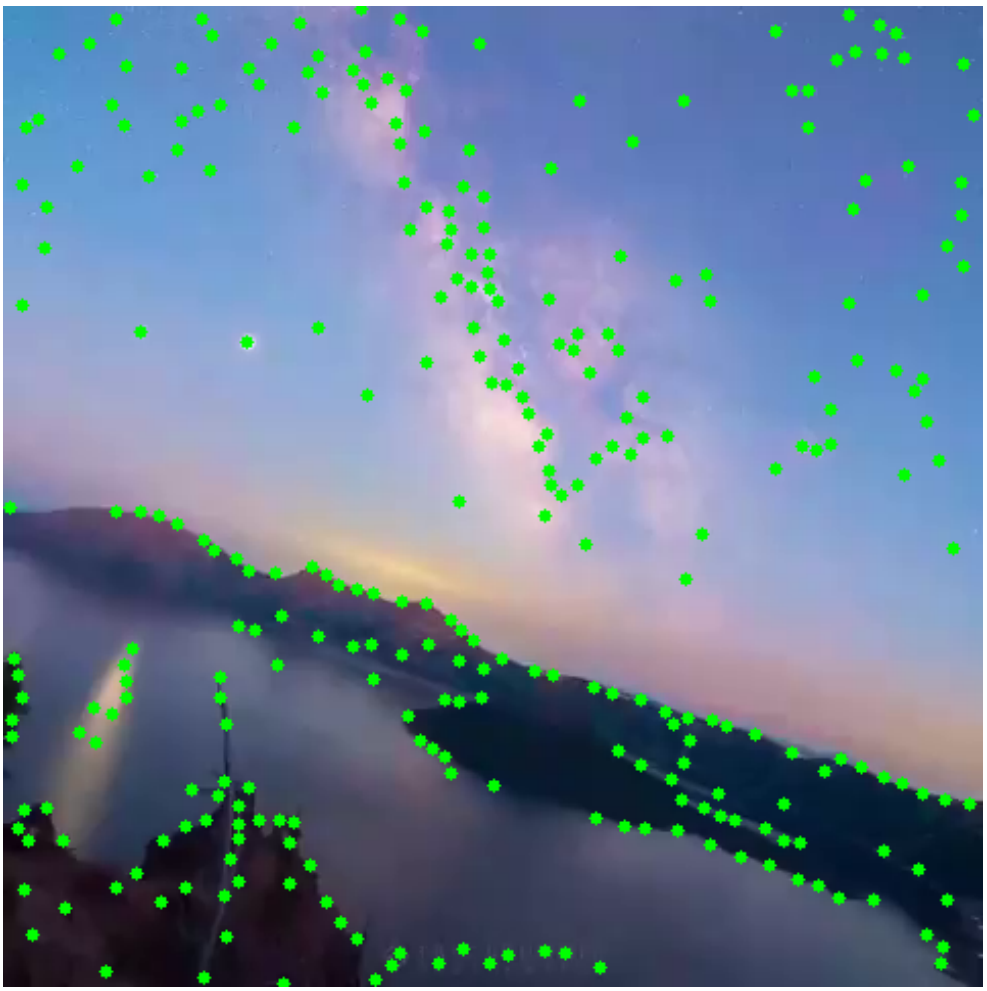

FIGURE 1. Exemple de détection de coins (`goodFeaturesToTrack`) sur la première image d'une séquence. Les points verts correspondent aux coins retenus pour le suivi.

4.2. Flot optique épars (Lucas–Kanade)

Sur la même séquence de mouvement de caméra, les résultats de Lucas–Kanade (Figure 2) révèlent :

— un **pattern global cohérent** avec la géométrie du champ de mouvement pour une caméra en translation : les vecteurs tendent à diverger d'un point de fuite situé vers le centre de l'image, ce qui rappelle les exemples de champ radial vus en cours ;
— dans les scènes d'objet en mouvement sur fond fixe, les vecteurs sont significatifs surtout sur les points appartenant à l'objet, tandis que les points sur le fond restent quasi-statiques ;
— les points proches des bords ou dans les zones peu texturées sont rapidement perdus (statut `status = 0`) ce qui illustre la sensibilité à la qualité des coins initiaux et au problème de l'ouverture.

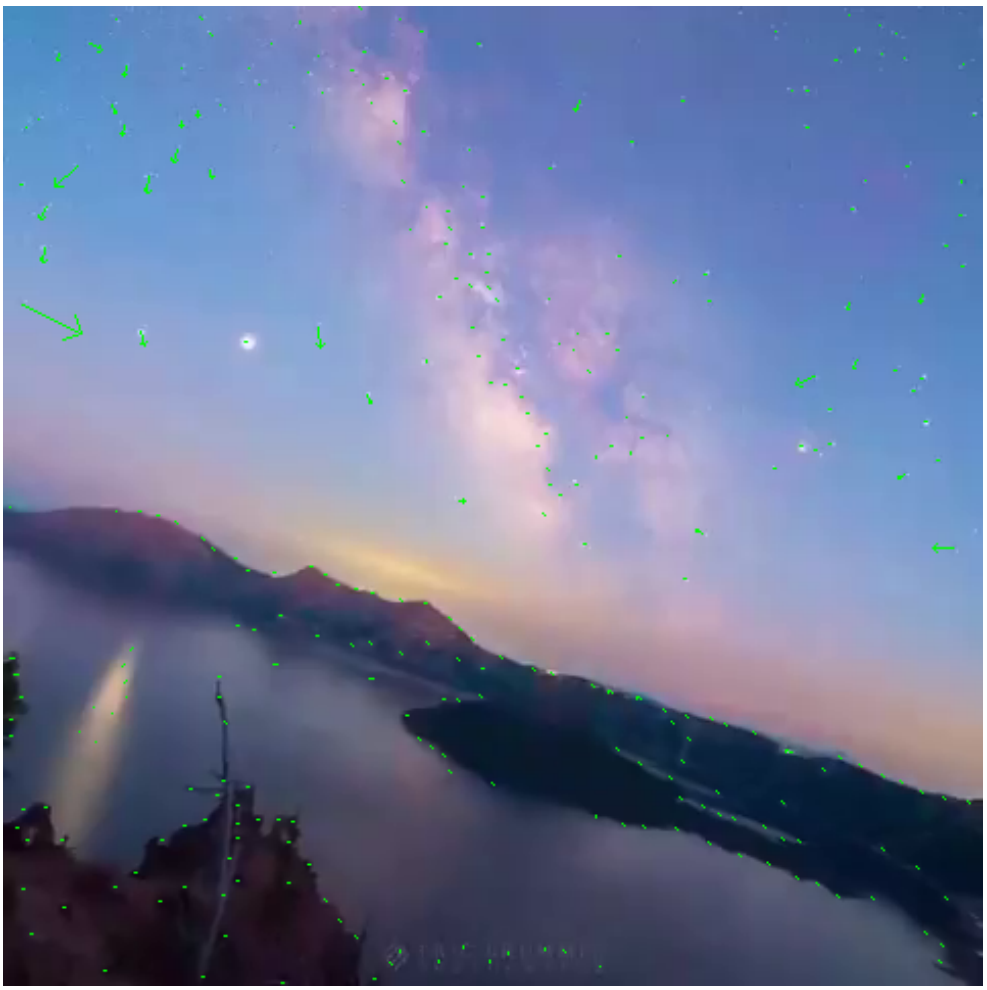

FIGURE 2. Suivi de points par Lucas–Kanade : les flèches vertes représentent le déplacement estimé entre deux images consécutives.

4.3. Flot optique dense (Farnebäck)

La Figure 3 montre le flot optique dense calculé par Farnebäck sur une séquence où la caméra se déplace dans un couloir. On observe que :

— les couleurs varient de manière continue dans l'image, reflétant la direction locale du mouvement : les zones proches du point de fuite partagent des teintes similaires ;
— la magnitude (codée par la luminosité) est plus forte en périphérie de l'image qu'au centre, ce qui est cohérent avec la projection perspective d'une translation ;
— dans les zones uniformes (grands pans de mur), la direction apparaît plus bruitée, confirmant que la méthode est limitée par la qualité du gradient spatial local ;
— dans les séquences avec objet mobile, les régions de l'objet se distinguent clairement du fond par une couleur différente, permettant une segmentation grossière par le mouvement.

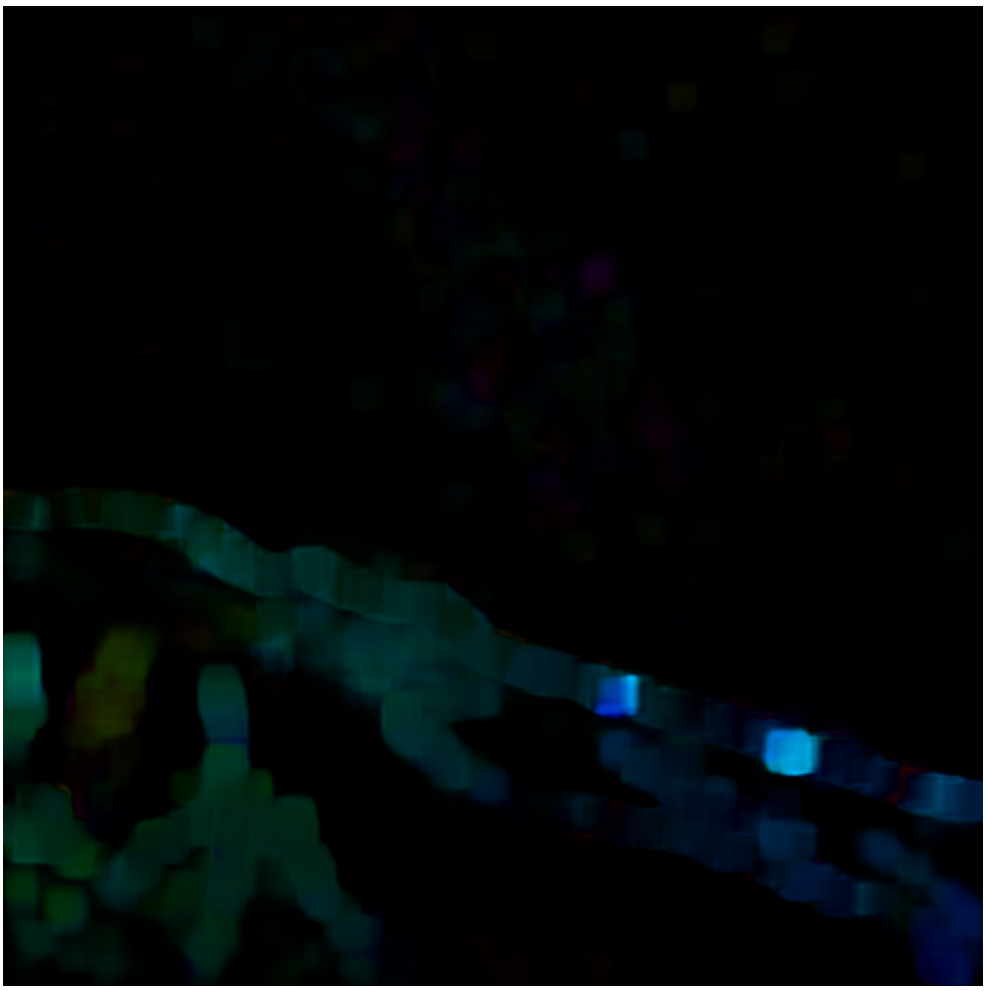

FIGURE 3. Flot optique dense (Farnebäck) encodé en couleur : teinte = direction, luminosité = norme du déplacement.

4.4. Influence des paramètres

Quelques observations empiriques :

— **GoodFeaturesToTrack** : augmenter `maxCorners` densifie la carte de points, mais au-delà d'un certain seuil les nouveaux points sont souvent peu discriminants. Une `minDistance` trop faible aboutit à des clusters de points très proches qui n'apportent pas d'information supplémentaire et peuvent se gêner mutuellement lors du suivi.

— **Lucas–Kanade** : une `winSize` trop petite ne capture pas suffisamment de contexte et rend la solution très sensible au bruit ; une fenêtre trop grande lisse les détails fins et peut fusionner des mouvements distincts (par exemple objet vs fond). Le nombre de niveaux pyramidaux `maxLevel` doit être suffisant si les déplacements image sont importants.

— **Farnebäck** : augmenter `winsize`, `poly_n` ou `poly_sigma` renforce le lissage spatial et temporel du champ de mouvement, produisant des champs plus réguliers mais moins sensibles aux petits détails. À l'inverse, des valeurs trop faibles rendent le champ plus bruité, surtout dans les zones uniformes.

5. Analyse critique et liens avec le cours

5.1. Limites théoriques illustrées par le TP

Les expérimentations confirment plusieurs points discutés au cours :

— **Problème de l'ouverture** : sur les bords isolés (par exemple le bord d'une porte sur un mur blanc), le flot optique reste mal contraint ; les vecteurs semblent "glisser" le long du bord. Seuls les coins bien définis fournissent des mouvements stables.
— **Rôle du gradient spatial** : dans les grands aplats (murs, ciel), le flot optique dense est essentiellement du bruit coloré, car les dérivées E_x et E_y sont faibles : la solution de l'algorithme devient très dépendante de la régularisation et de lissage.
— **Différence flot optique / champ de mouvement** : dans les scènes contenant des objets déformables ou des changements d'illumination (reflets, scintillements), le mouvement apparent de la luminosité ne correspond pas exactement au mouvement géométrique des surfaces, comme anticipé par la modélisation de surfaces lambertiennes.

5.2. Comparaison Lucas–Kanade vs Farnebäck

D'un point de vue pratique :

— Lucas–Kanade fournit un **flot épars** mais de bonne qualité sur les points d'intérêt sélectionnés ; il est adapté au *tracking* de primitives (par exemple, pour la reconstruction 3D ou la mesure de trajectoires).
— Farnebäck produit un **champ dense** qui permet une visualisation globale du mouvement, utile pour la segmentation par le mouvement ou pour des applications de stabilisation vidéo, mais il est plus coûteux et plus sensible au bruit dans les zones peu texturées.
— Les deux méthodes reposent sur l'hypothèse de **petits mouvements** entre frames successives et sur une forme de **lissage spatial** du champ de mouvement, cohérente avec l'approximation du champ de mouvement d'un plan par un modèle affine ou quadratique vue en cours.

5.3. Qualité visuelle vs métriques quantitatives

Dans ce TP, l'analyse est restée essentiellement **qualitative** : on n'a pas accès à un "flot optique vérité terrain" pour comparer les vecteurs estimés à des données annotées, contrairement à certains benchmarks de la littérature. Néanmoins :

— la cohérence globale du champ (radial pour la translation de caméra, cohérent sur les objets rigides) est un bon indicateur de validité ;
— les incohérences évidentes (vecteurs divergents, bruit en zones uniformes) illustrent bien les limites des modèles et des hypothèses (illumination constante, mouvements petits, rigidité de la scène).

Dans un contexte plus avancé, on pourrait compléter ce travail par une évaluation quantitative sur des bases annotées (Middlebury, KITTI, etc.), comme évoqué en bibliographie.

6. Conclusion

Ce TP a permis de mettre en pratique les concepts du cours *Séquences Vidéo* sur des données réalistes :

1. La **détection de coins** par GoodFeaturesToTrack montre l'importance des gradients spatiaux pour le suivi : seuls les points présentant une variation significative dans deux directions fournissent un flot optique stable et exploitable.
2. Le **flot optique épars** par Lucas–Kanade permet d'estimer des trajectoires robustes pour un nombre limité de points bien choisis. Il est particulièrement adapté au *tracking* d'objets et aux applications de reconstruction 3D à partir du mouvement.
3. Le **flot optique dense** par Farnebäck offre une vue globale du mouvement dans l'image, utile pour la segmentation ou l'analyse de scènes, au prix d'une plus grande sensibilité au bruit et à la texture.
4. Les résultats obtenus confirment les **limites théoriques** présentées au cours : problème de l'ouverture, dépendance vis-à-vis du gradient spatial, différence entre flot optique et champ de mouvement réel.

En résumé, ce TP montre que le flot optique n'est pas seulement une "sortie magique" d'OpenCV, mais le résultat d'un compromis entre hypothèses physiques (illumination, rigidité), choix algorithmiques (Lucas–Kanade vs Farnebäck) et qualité des données (texture, bruit, échantillonnage temporel). Une bonne compréhension de ces éléments est indispensable pour interpréter correctement les champs de vecteurs obtenus et les utiliser dans des systèmes de vision plus complexes (suivi, navigation, reconstruction).

Annexe A – Liste des figures et fichiers

Figure	Fichier	Description
1	`gftt_frame0.png`	Détection de coins (GoodFeaturesToTrack) sur la première image.
2	`lk_frame_0000.png`	Suivi de points par Lucas–Kanade (flot optique épars).
3	`farneback_frame_0000.png`	Flot optique dense (Farnebäck) encodé en couleur HSV.

TABLE 1. Récapitulatif des figures et des fichiers d'images associés au TP flot optique.

Auteur : Wassim Chikhi **Année :** 2025/2026