

République Tunisienne

Ministère de la Défense Nationale

Armée de l’Air

Ecole de l’Aviation de Borj el Amri

**Mémoire de projet de fin d'étude**

Spécialité

**Télémécanique**

Par

**SLT Boukary DERRA**

**Détection de petites cibles mobiles dans des arrière-plans complexes en utilisant des capteurs optiques par traitement bio-inspiré**

Président de jury : ….

Rapporteur : …..

Encadrant : Lt / CL Tijeni Delleji

Année Universitaire : 2022 – 2023

Dédicace :

*Je dédie ce travail,*

*À ma famille,*

*À mes amis et à mes proches pour leurs appuis et leurs encouragements,*

*À tous ces anonymes qui ont contribué d’une quelconque manière à ma croissance,*

*Au LCL Tijeni Delleji, sans qui ce projet n’aurais pas eu lieu.*

*Boukary DERRA*

Remerciements :

*Je souhaiterais tout d’abord remercier* ***Monsieur le Colonel Major*** *Commandant de l’Ecole de l’Aviation de Borj El Amri ainsi que Monsieur le* ***Colonel*** *Directeur des études pour tout le support et l’aide qu’ils m’ont consacrés.*

*Je remercie également* ***Monsieur le directeur des études universitaires*** *pour ses encouragements et ses conseils avisés tout au long de la réalisation de mon travail.*

*Sans oublier les honorables* ***Membres de jury*** *qui ont aimablement accepté de discuter ce travail et apporté leurs précieuses critiques et suggestions.*

*Au terme de ce travail, il est de mon devoir de remercier vivement et chaleureusement tous ceux qui ont contribué à la réalisation de ce modeste travail.*

*Les mots sont faibles pour exprimer toute la gratitude que je porte au* ***LCL Tijeni Delleji****, Professeur à l’ENSIT, qui m’a soutenu et m’a encadrée dans ce travail et chez qui j’ai trouvé l’appui scientifique et moral désirable. Je tiens tout personnellement à le remercier de m’avoir accordée sa confiance et de m’avoir orientée avec ses précieux conseils.*

*Je tiens également à remercier tous le personnel de l’ ́Ecole de l’Aviation de Borj el Amri avec qui j’ai passé ces 5 merveilleuses années.*

Table des matières

[Acronymes: 8](#_Toc135731012)

[INTRODUCTION GÉNÉRALE 9](#_Toc135731013)

[1. Les détecteurs de mouvement traditionnels 11](#_Toc135731014)

[1.1. Introduction 11](#_Toc135731015)

[1.2. Détection de mouvement par flux optique 12](#_Toc135731016)

[1.3. Détection de mouvement par soustraction de fond 15](#_Toc135731017)

[1.4. Détection de mouvement par différenciation de cadre 16](#_Toc135731018)

[1.5. Détection de mouvement par différenciation temporelle 18](#_Toc135731019)

[1.6. Détection de mouvement par infrarouge 19](#_Toc135731020)

[1.7. Conclusion 20](#_Toc135731021)

[2. Formulation des modèles bio-inspiré 22](#_Toc135731022)

[2.1. Introduction et présentation 22](#_Toc135731023)

[2.2. Approche biologique et applications 23](#_Toc135731024)

[2.3. ESTMD avec un nouveau mécanisme d'inhibition latérale (LIM) 29](#_Toc135731025)

[2.3.1. Présentation 29](#_Toc135731026)

[2.3.2. Modélisation 30](#_Toc135731027)

[2.3.3. La couche Retina 32](#_Toc135731028)

[2.3.4. La couche Lamina 33](#_Toc135731029)

[2.3.5. La couche medulla 35](#_Toc135731030)

[2.3.6. La couche lobula 40](#_Toc135731031)

[2.4. Conclusion 40](#_Toc135731032)

[3. Implémentation et discussion 41](#_Toc135731033)

[3.1. Introduction 41](#_Toc135731034)

[3.2. Initialisation de la class STMD 43](#_Toc135731035)

[3.3. Le noyau gaussien 44](#_Toc135731036)

[3.4. Photorécepteur 45](#_Toc135731037)

[3.6. Le filtre passe bas 47](#_Toc135731038)

[3.7. LMC 48](#_Toc135731039)

[3.8. Les canaux ON / OFF 48](#_Toc135731040)

[3.9. FDSR 49](#_Toc135731041)

[3.10. SIGMA 50](#_Toc135731042)

[3.11. HWR 51](#_Toc135731043)

[3.12. L’inhibition latérale 53](#_Toc135731044)

[3.12.1. Block based motion estimation 53](#_Toc135731045)

[3.12.2. Find optical motion vector 53](#_Toc135731046)

[3.12.3. Motion components 55](#_Toc135731047)

[3.12.4. Le noyau de convolution H 56](#_Toc135731048)

[3.12.5. La matrice W 57](#_Toc135731049)

[3.13. Le retard et la sortie finale 58](#_Toc135731050)

[3.14. Conclusion 59](#_Toc135731051)

[CONCLUSION GENERALE 60](#_Toc135731052)

[BIBLIOGRAPGIE 61](#_Toc135731053)

[Annexe 65](#_Toc135731054)

[A1. Faneback pour la détection de mouvement par flux optique 65](#_Toc135731055)

[A2. MOG2 pour la détection de mouvement par soustraction de fond 66](#_Toc135731056)

[A3. Détection de mouvement par différenciation de cadre 68](#_Toc135731057)

[A4. Détection de mouvement par différenciation temporel 70](#_Toc135731058)

[A5. Fonction Gaussienne 71](#_Toc135731059)

[A6. Résultats des différents couches du frame V1-47 77](#_Toc135731060)

[A7. Paquages et Installations 83](#_Toc135731061)

[A7.1. Python 83](#_Toc135731062)

[A7.2. OpenCV 83](#_Toc135731063)

[A7.3. Scipy 83](#_Toc135731064)

[A7.4. Matplotlib 84](#_Toc135731065)

[A7.5. Numpy 84](#_Toc135731066)

[A7.6. ATOM 85](#_Toc135731067)

[A7.7. Windows PowerShell 85](#_Toc135731068)

[A7.8. Environnement virtuel (venv ) 86](#_Toc135731069)

[A7.9. Django 87](#_Toc135731070)

[A7.10. Git et GitHub 89](#_Toc135731071)

Table des figures

[Figure 1: 47th frame de V1 (V1-47) 11](#_Toc135729121)

[Figure 2: Détection de mouvement par flux optique sur V1-47 13](#_Toc135729122)

[Figure 3: Détection de mouvement par soustraction de fond sur V1-47 15](#_Toc135729123)

[Figure 4: Détection de mouvement par différenciation de cadre sur V1-47 16](#_Toc135729124)

[Figure 5: Détection de mouvement par différenciation temporelle sur V1-47 18](#_Toc135729125)

[Figure 6:Schéma en bloc des différents types de détection bio-inspiré 21](#_Toc135729126)

[*Figure 7: (a) un sphinx. (b) un syrphe. (c) Une libellule.* 22](#_Toc135729127)

[Figure 8: Réponse des neurones STMD. (a) Réponses neuronales de STMD au mouvement de trois cibles de tailles différentes (0,8°, 3° et 15° de hauteur sur 0,8° de largeur) dérivant sur des arrière-plans lumineux : les barres horizontales indiquent la durée du mouvement et les flèches indiquent la direction du mouvement de la cible (Figure 28-a du document [7]). (b) La réponse d'un STMD à des cibles de hauteur variable (Figure 28-b du document [7]). 23](#_Toc135729128)

[Figure 9 : Réponse directionnelle des neurones STMD. (a) Réponses brutes du neurone STMD sélectif (en direction PD ou ND) qui préfère le mouvement de la cible vers la gauche, testé par le mouvement de trois cibles de tailles différentes (0,8◦, 3◦ et 15◦ de hauteur sur 0,8◦ de largeur) dérivées contre des arrière-plans lumineux : les barres horizontales indiquent la durée des stimuli et les flèches indiquent la direction du mouvement de la cible (Figure 29-b du document [7]). (b) Réponses du neurone STMD qui préfère le mouvement de la cible vers le bas, aux cibles dérivées contre des arrière-plans encombrés (Figure 29-b du document [7]). 25](#_Toc135729129)

[Figure 10: Modèles et Application du système visuel des insectes [7] 27](#_Toc135729130)

[Figure 11: Schéma d’ESTMD avec un nouveau mécanisme d’inhibition latéral. [9] 30](#_Toc135729131)

[Figure 12: Fonction Gaussienne 71](#_Toc135729132)

[Figure 13: Sortie photorécepteur de V1-47 76](#_Toc135729133)

[*Figure 14: sortie lipetz transformation de V-63* 77](#_Toc135729134)

[*Figure 15: sortie low pass filter de V-63* 77](#_Toc135729135)

[*Figure 16: sortie LMC de V-63* 78](#_Toc135729136)

[*Figure 17: LMC en OFF channel de V1-63* 78](#_Toc135729137)

[*Figure 18: LMC en ON channel de V1-63* 79](#_Toc135729138)

[*Figure 19: sortie OFF FDSR de V1-63* 79](#_Toc135729139)

[*Figure 20: Sortie ON FDSR de v1-63* 80](#_Toc135729140)

[*Figure 21: Sortie OFF SIGMA de V1-63* 80](#_Toc135729141)

[*Figure 22: Sortie ON SIGMA V1-63* 81](#_Toc135729142)

[*Figure 23: Sotie OFF HWR de V1-63* 81](#_Toc135729143)

[*Figure 24: Sortie ON HWR de V1-63* 82](#_Toc135729144)

# Acronymes:

**BPF**: Band Pass Filtering

**DCMD**: Descending Contra-lateral Movement Detector

**DS**: Direction Selective

**DSNs**: Direction Selective Neuron(s)

**EMDs**: Elementary Motion Detector(s)

**FDSR**: fast depolarization, slow repolarization

**GB**: Gaussian Blur

**HPF**: High Pass Filtering

**HW-R**: half wave rectifier

**LGMDs**: Lobula Giant Movement Detector(s)

**LI**: Lateral Inhibition

**LIM**: Lateral Inhibition Mechanism

**LPF**: Low Pass Filtering

**LPNM**: Looming Perception Neuron Models

**LPTCs**: Lobula Plate Tangential Cell(s)

**LSM**: Looming Sensitive Models

**MOG2**: Mixture of Gaussians 2

**ND**(s): null or non-preferred direction(s)

**ODE**: ordinary differential equation

**OF**: Optic Flow

**PD**(s): preferred direction(s)

**STMDs**: Small Target Motion Detector(s)

**STMSM**: Small Target Motion Sensitive Models

**TSM**: Translation Sensitive Models

**UAV**: Unmanned Aerial Vehicle

# INTRODUCTION GÉNÉRALE

Pour tout être vivant, être capable de détecter un objet en mouvement est très important et même souvent vital. Cette opération est encore plus cruciale pour certains animaux comme les insectes qui doivent être capables d'analyser les actions autour d'eux, de pouvoir suivre leurs semblables, d'éviter les prédateurs, ou encore d'attaquer des proies.

Dans l'ère des progrès techniques et technologiques, le même problème de détection d'un objet en mouvement existe ; surtout dans les domaines tels que la robotique, les véhicules autonomes, la défense aérienne, etc. Cette détection s'avère encore plus compliquée lorsque la taille de la cible est réduite, ou que le fond dans lequel elle se trouve est relativement complexe ou brouillant. Le but de ce projet est de relever ce défi tout en s'inspirant des modèles biologiques, notamment les insectes.

En effet, avec des millions d'années d'évolution, le système visuel des insectes fait partie des plus efficaces pour tout ce qui est de la détection des mouvements. Par exemple, les criquets peuvent voler à des centaines de kilomètres dans un environnement très dense sans faire de collision ; les abeilles arrivent à suivre les déplacements de leurs partenaires malgré la rapidité ; les mantes religieuses peuvent surveiller les petites proies en mouvement dans un environnement complexe.

Mais avant d’aller plus loin, il est légitime de nous poser la question suivante : Pourquoi s’inspirer d’un modèle biologique alors qu’il existe des techniques de détection qui sont moins complexes et simples à implémenter ?

Traditionnellement, il existe plusieurs méthodes pour détecter des cibles en mouvement ; parmi ces méthodes, on a : les ondes infrarouges (Infrared Small Target Motion Detector), les flux optiques (Optical flow), la soustraction de fond (background subtraction), la différenciation temporelle (temporal differencing), etc.

Cependant, ces méthodes traditionnelles perdent de leur efficacité lorsque la cible visée se réduit à une taille minime, de l'ordre de quelques pixels, ou lorsque le contexte dans lequel elle évolue est particulièrement complexe, instable, et présente des caractéristiques visuelles qui peuvent se confondre avec celles de la cible elle-même. Cette complexité croissante souligne l'importance d'explorer de nouvelles approches pour la détection, de chercher des sources d'inspiration innovantes telles que la biologie et plus particulièrement l'entomologie, l'étude des insectes.

Plusieurs modèles basés sur les insectes ont été développés durant ces dernières années. Parmi ces modèles bio-inspirés, on a les modèles sensibles aux profils (LSM), principalement utilisés pour la détection des collisions ; les modèles sensibles aux translations (TSM), principalement utilisés pour déterminer les directions prises par des objets ; et les modèles sensibles au mouvement des cibles de petites tailles (STMSM). Ces modèles sont caractérisés par les détecteurs STMDs qui feront l’objet de notre projet.

Rappelons que l’objectif principal de notre travail est la détection du mouvement des petites cibles (pouvant être de l'ordre de quelques pixels) dans un fond relativement complexe pouvant être mobile.

Tout d’abord, nous allons faire le tour des modèles de détection de mouvement conventionnels et leurs limites dans le cadre de notre projet dans la Section I. Dans la section II, nous allons entrer dans le cœur du travail qui consiste à l’étude des différents modèles et applications du système visuel des insectes puis nous nous attarderons particulièrement sur l’étude du modèle élémentaire ESTMD avec une nouvelle inhibition latérale. Pour la Section III, nous implémenterons et analyserons les sorties de chaque couche du modèle élémentaire jusqu’à l’obtention du résultat finale. Et enfin, nous allons conclure le travail par les limites du modèle choisi, les possibilités et les perspectives.

# Les détecteurs de mouvement traditionnels

## Introduction

Dans ce premier chapitre, nous allons nous pencher sur les méthodes traditionnelles de détection de mouvement. Ces techniques ont servi de base à de nombreux systèmes de surveillance, de robotique et de véhicules autonomes. Bien qu'elles aient fait leurs preuves, elles présentent certaines limites, en particulier lorsqu'il s'agit de détecter de petits objets en mouvement dans un environnement complexe ou changeant.

Nous allons explorer les principaux types de détection de mouvement traditionnelle :

* La détection de mouvement par flux optique, qui est basée sur le changement de l'intensité lumineuse entre deux images consécutives.
* La détection de mouvement par soustraction de fond, qui consiste à isoler l'objet en mouvement en soustrayant l'image de fond actuelle de celle précédente.
* La détection de mouvement par différenciation de cadre, qui repose sur la comparaison entre deux images successives pour mettre en évidence les zones de changement.
* La détection de mouvement par différenciation temporelle, qui met en évidence les mouvements en analysant les variations de l'image sur une période de temps.
* Et enfin, la détection de mouvement par infrarouge, qui exploite les variations de température causées par le mouvement d'un objet pour le détecter.

Chaque technique sera examinée en détail, en discutant de ses principes de base, de ses forces, de ses faiblesses et de ses applications typiques. Cette compréhension approfondie des méthodes traditionnelles nous permettra de mieux apprécier les avantages que peut offrir l'inspiration biologique pour la détection de mouvement, sujet que nous aborderons dans le chapitre suivant.

Pour les différents tests, nous avons utilisé une vidéo de 5 seconds qu’on va appeler ici V1. C’est la vidéo d’un drone en vol dans un fond en mouvement et relativement complexe (nuages + montagnes). Nous avons subdivisé cette vidéo en 151 frames. Dans ce rapport nous allons afficher les résultats du frame 47 (V1-47) autour du quelle les mouvements de fond sont très remarquables.



Figure 1: 47th frame de V1 (V1-47)

## Détection de mouvement par flux optique

Le flux optique est une méthode utilisée en vision par ordinateur pour suivre le mouvement d'objets dans un flux vidéo. L'idée de base consiste à utiliser le mouvement apparent des pixels entre des cadres consécutifs pour calculer la vitesse de chaque pixel, ce qui peut ensuite être utilisé pour détecter des objets en mouvement. La méthode de flux optique implique de calculer le vecteur de flux pour chaque pixel dans le cadre actuel en fonction des modifications d'intensité entre le cadre actuel et le cadre précédent. Les vecteurs de flux résultants peuvent être utilisés pour estimer le mouvement des objets dans la scène, ce qui peut être utilisé pour diverses tâches de vision par ordinateur, y compris la détection de mouvement.

Le processus de détection de mouvement par flux optique s'articule autour de quatre étapes principales. Tout d'abord, l'extraction des caractéristiques visuelles, telles que les coins, est réalisée à partir de la première image. Ensuite, les déplacements de ces caractéristiques entre les deux images sont calculés en utilisant une formule mathématique qui modélise la relation entre les pixels de l'image courante et l'image suivante, ce qui permet de calculer les vecteurs de mouvement. Ces vecteurs sont ensuite utilisés pour construire une carte de mouvement, représentant les déplacements des objets dans la scène. Enfin, cette carte de mouvement peut être représentée visuellement, par exemple sous forme de vecteurs fléchés ou de couleurs, pour indiquer la direction et l'ampleur du mouvement dans la scène.

La détection de mouvement par flux optique offre plusieurs avantages significatifs. Elle permet une analyse détaillée et précise des mouvements dans une scène, grâce à l'extraction des caractéristiques visuelles et au calcul des vecteurs de mouvement. Elle est capable de détecter des mouvements subtils qui pourraient échapper à d'autres techniques, ce qui la rend particulièrement utile pour la surveillance, la robotique et d'autres applications nécessitant une détection de mouvement de haute précision. De plus, la représentation visuelle de la carte de mouvement fournit une compréhension intuitive et claire du mouvement dans la scène. Enfin, en tant que technique de traitement d'image purement numérique, elle ne nécessite pas de capteurs spécifiques ou coûteux et peut donc être mise en œuvre de manière relativement économique.

Malgré ses avantages, la détection de mouvement par flux optique comporte également certaines limitations. Elle peut être sensible au bruit, aux variations d'éclairage et aux ombres, ce qui peut entraîner des erreurs de détection. De plus, elle suppose que la luminance d'un pixel reste constante entre deux images successives, ce qui n'est pas toujours le cas dans les situations réelles, en particulier lors de mouvements rapides ou de changements d'éclairage. Par ailleurs, le calcul des vecteurs de mouvement peut être complexe et nécessite des ressources de calcul importantes, ce qui peut limiter son utilisation dans des systèmes en temps réel ou à ressources limitées. Enfin, elle peut avoir du mal à gérer les mouvements de caméra, les rotations ou les changements d'échelle, nécessitant ainsi des étapes de prétraitement ou post-traitement supplémentaires pour obtenir des résultats précis.

Il existe plusieurs algorithmes de flux optique différents qui peuvent être utilisés pour la détection de mouvement. Certains algorithmes populaires incluent Lucas-Kanade, Horn-Schunck et Farneback. Ces algorithmes varient en termes de complexité et de précision, et le choix de l'algorithme dépendra de l'application spécifique et des exigences. Voire le document [14] pour plus d’information sur la détection par flux optique.



Figure 2: Détection de mouvement par flux optique sur V1-47

Pour obtenir le résultat ci-dessus nous avons calculé le flux optique en utilisant la méthode Farneback mise en œuvre dans OpenCV.

L'algorithme de Farneback calcule un flux optique dense, qui peut être utilisé pour suivre le mouvement des pixels d'une image à une autre. Dans le script [A1] que nous avons posté dans l’annexe, l'algorithme calcule les vecteurs de flux pour chaque pixel, qui sont ensuite transformés en une image pour la visualisation.

On remarque bien que la méthode par flux optique détecte la cible (le drone). Mais aussi les sommets des montagnes. Ces erreurs sont dues au fait que le fond dans lequel nous voulions détecter ce drone soit en mouvement.

La détection de mouvement par flux optique est largement utilisée dans divers domaines en raison de sa capacité à capturer des mouvements détaillés et précis. Dans le domaine de la robotique, elle est utilisée pour la navigation et l'évitement d'obstacles. En matière de surveillance de la sécurité, elle permet de détecter les mouvements suspects et d'activer des alertes. Dans les domaines de la réalité augmentée et des jeux vidéo, elle facilite le suivi des mouvements du joueur pour une interaction plus immersive. En outre, elle est également utilisée dans l'analyse du trafic routier pour détecter et suivre les véhicules, ainsi que dans les applications médicales pour le suivi des mouvements des patients dans les thérapies de réhabilitation.

## Détection de mouvement par soustraction de fond

La détection de mouvement par soustraction de fond est une technique de traitement d'image largement répandue qui se concentre sur l'identification et l'isolation des objets en mouvement dans une scène. L'approche de base de cette technique repose sur l'utilisation d'une image de référence, souvent désignée comme image de fond ou modèle de fond. L'idée est que tout objet en mouvement introduit dans une scène causera une perturbation du fond stable. En soustrayant cette image de fond de l'image actuelle, nous sommes en mesure de détecter ces perturbations, qui représentent les objets en mouvement.

Le processus de détection de mouvement par soustraction de fond se déroule en plusieurs étapes. Tout d'abord, une image de fond est créée, qui sert de référence pour la scène sans mouvement. Celle-ci est souvent générée en prenant la moyenne de plusieurs images capturées à différents moments. Ensuite, chaque image nouvelle ou actuelle est soustraite de cette image de fond pour produire une image de différence. Les pixels dans l'image de différence qui dépassent un certain seuil sont identifiés comme des pixels d'objet en mouvement. Enfin, une étape de post-traitement est souvent nécessaire pour éliminer le bruit et améliorer les résultats, par exemple en éliminant les petites régions détectées ou en comblant les trous dans les régions détectées.

L'un des principaux avantages de la détection de mouvement par soustraction de fond est son efficacité dans les scènes où le fond est principalement statique. Elle offre une segmentation nette des objets en mouvement, rendant l'analyse et le suivi de ces objets beaucoup plus facile. De plus, cette méthode est assez simple à comprendre et à mettre en œuvre, la rendant accessible pour une grande variété d'applications.

Cependant, la détection de mouvement par soustraction de fond a aussi ses inconvénients. Premièrement, elle suppose que le fond est majoritairement statique, ce qui n'est pas toujours le cas, surtout dans les environnements extérieurs où divers facteurs peuvent affecter l'image de fond. De plus, cette technique est sensible aux changements d'éclairage et aux ombres, qui peuvent être interprétés comme des mouvements. En outre, l'obtention d'un modèle de fond approprié peut être un défi, surtout dans les scènes avec des variations dynamiques. Voire le document [15] pour plus de détail sur la détection de mouvement par soustraction de fond.



Figure 3: Détection de mouvement par soustraction de fond sur V1-47

Il existe plusieurs méthodes pour la soustraction de fond. Nous avons utilisé la méthode MOG2 pour générer le résultat ci-dessus. Voire l’annexe [A2] pour plus de détail.

On remarque bien que la méthode par soustraction de fond détecte la cible. Mais aussi les nuages et les sommets des montagnes. C’est erreur sont dû au fait que le fond dans lequel nous voulions détecter ce drone soit assez complexe et en mouvement.

En dépit de ces défis, la détection de mouvement par soustraction de fond est largement utilisée dans diverses applications. Dans le domaine de la surveillance de la sécurité, elle est utilisée pour détecter tout mouvement suspect. Dans le domaine du trafic et des transports, elle est utilisée pour le suivi et le comptage des véhicules. D'autres applications incluent la navigation et l'évitement d'obstacles en robotique, le suivi des mouvements dans les vidéos sportives, l'analyse du comportement des animaux en biologie, et bien plus encore.

## Détection de mouvement par différenciation de cadre

La détection de mouvement par différenciation de cadre est une méthode populaire utilisée pour identifier les objets en mouvement dans une séquence d'images ou de vidéos. Comme son nom l'indique, cette technique fonctionne en différenciant ou en soustrayant l'image actuelle de l'image précédente, ce qui permet de détecter les changements ou les mouvements dans la scène.

Le processus de détection de mouvement par différenciation de cadre commence par la prise de deux images consécutives ou cadres. Le cadre actuel est ensuite soustrait du cadre précédent, créant une image de différence. Dans cette image de différence, les objets stationnaires seront principalement noirs, car il n'y a pas eu de changement entre les deux cadres. Les objets en mouvement, en revanche, seront visibles en raison des différences entre les cadres. Enfin, des techniques de seuillage et de post-traitement sont souvent appliquées pour améliorer les résultats et réduire le bruit.

L'un des avantages clés de la détection de mouvement par différenciation de cadre est qu'elle est relativement simple à mettre en œuvre, ne nécessitant pas de modèle de fond complexe. Elle est également capable de détecter efficacement les mouvements dans des scènes dynamiques où le fond peut changer fréquemment. De plus, cette méthode peut facilement être adaptée pour travailler en temps réel, ce qui est particulièrement utile pour les applications de surveillance et de suivi.

Cependant, la détection de mouvement par différenciation de cadre présente également quelques inconvénients. Par exemple, elle peut avoir du mal à détecter les mouvements lents ou minimes, car la différence entre les cadres peut ne pas être suffisante pour dépasser le seuil. De plus, elle est sensible au bruit et aux variations d'éclairage, qui peuvent être interprétés comme des mouvements.



Figure 4: Détection de mouvement par différenciation de cadre sur V1-47

Voire l’annexe [A3] pour découvrir de façons détaillé la manière d’obtenir ce résultat.

On remarque que la méthode par soustraction de cadre détecte la cible. Mais aussi quelques nuages et les sommets des montagnes. C’est erreur sont dû au fait que le fond dans lequel nous voulions détecter ce drone soit assez complexe et en mouvement.

Malgré ces limitations, la détection de mouvement par différenciation de cadre est largement utilisée dans une variété de domaines. Elle est particulièrement utile pour le suivi d'objets dans les vidéos de surveillance, l'analyse du mouvement dans les sports et la médecine, la navigation et l'évitement d'obstacles pour les drones ou les véhicules autonomes, et de nombreuses autres applications nécessitant la détection en temps réel des mouvements.

Voire le document [16] de Fatima et al. pour plus de détail sur la détection de mouvement par différentiation de cadre.

## Détection de mouvement par différenciation temporelle

La détection de mouvement par différenciation temporelle est une technique efficace de traitement d'images qui vise à identifier les variations ou les changements dans une séquence d'images sur une période de temps. Cette méthode est particulièrement adaptée aux environnements dynamiques où les objets en mouvement peuvent être distingués des objets immobiles en se basant sur le changement de leur position au fil du temps.

Le processus de détection par différenciation temporelle commence par la collecte de trois images consécutives, ou cadres, dans une séquence. Ces cadres sont ensuite différenciés deux à deux, et le résultat final est obtenu en effectuant une opération logique "ET" entre les deux images de différence. Cela permet d'obtenir une représentation précise des objets en mouvement en éliminant les variations d'éclairage et les bruits temporaires.

Les avantages de la détection de mouvement par différenciation temporelle sont nombreux. Premièrement, cette méthode offre une grande précision en déterminant les objets en mouvement, même dans des scènes complexes. Deuxièmement, elle est assez robuste aux changements de luminosité et aux variations d'éclairage, ce qui la rend utile dans diverses conditions d'éclairage. Troisièmement, elle n'exige pas une grande capacité de traitement, ce qui la rend appropriée pour des applications en temps réel.

Cependant, la détection de mouvement par différenciation temporelle présente également certains défis. Par exemple, elle peut ne pas être efficace pour détecter des mouvements lents en raison de la nécessité de capturer des changements significatifs entre les cadres. De plus, elle peut avoir du mal à gérer les scènes avec un mouvement de caméra ou un mouvement de fond complexe.



Figure 5: Détection de mouvement par différenciation temporelle sur V1-47

Voire l’annexe [A4] pour découvrir de façons détaillé la manière d’obtenir ce résultat.

Comme les méthodes précédentes, la différenciation temporelle à elle aussi du mal à supprimer les nuages et les sommets des montagnes.

Malgré ces défis, la détection de mouvement par différenciation temporelle est largement utilisée dans divers domaines. Elle est employée dans les systèmes de vidéosurveillance pour la détection de mouvement et le suivi d'objets, dans les systèmes de navigation autonome pour l'évitement d'obstacles, et dans l'analyse du mouvement humain dans les domaines du sport et de la médecine, entre autres. [14].

## Détection de mouvement par infrarouge

La détection de mouvement par infrarouge est une technique qui repose sur l'utilisation de capteurs infrarouges passifs (PIR) pour détecter la présence et le mouvement d'objets, en particulier des êtres vivants, en mesurant les variations de chaleur émises sous forme de rayonnement infrarouge.

Le processus de détection commence par la surveillance constante de la température ambiante par le capteur. Lorsqu'un objet en mouvement entre dans le champ de vision du capteur, il détecte un changement abrupt dans les ondes infrarouges dues à la différence de chaleur entre l'objet et l'arrière-plan. Cette variation déclenche un signal qui peut ensuite être utilisé pour déclencher une alarme, activer une caméra, allumer une lumière ou exécuter toute autre action programmée.

Les principaux avantages de la détection de mouvement par infrarouge résident dans sa sensibilité et sa fiabilité. Les capteurs infrarouges sont très sensibles et peuvent détecter des mouvements minimes avec une grande précision. Ils sont également très fiables et moins susceptibles de déclencher de fausses alarmes, car ils ne réagissent qu'aux objets qui émettent une certaine quantité de chaleur, comme les êtres humains et les animaux.

Cependant, il y a aussi quelques inconvénients. Les capteurs infrarouges peuvent être affectés par des sources de chaleur externes, comme le soleil, les appareils de chauffage ou les voitures en mouvement. De plus, ils peuvent ne pas être en mesure de détecter des objets qui n'émettent pas de chaleur ou dont la température est similaire à celle de l'environnement. Ce qui fait que cette méthode ne peut pas résoudre notre problème.

La détection de mouvement par infrarouge a de nombreuses applications, notamment dans les systèmes de sécurité et de surveillance où elle est utilisée pour détecter les intrus. Elle est également utilisée dans les systèmes d'éclairage automatique, où les lumières s'allument lorsqu'un mouvement est détecté, et dans d'autres applications comme les systèmes d'automatisation domestique, les portes automatiques et les appareils ménagers. Cependant, cette technique de détection nécessite généralement du matériel spécifique qui est capable de capturer des images ou des vidéos dans le spectre infrarouge. En pratique, cela est souvent accompli avec des caméras thermiques ou des capteurs infrarouges passifs (PIR).

## Conclusion

Dans ce premier chapitre, nous avons exploré diverses techniques traditionnelles de détection de mouvement : le flux optique, la soustraction de fond, la différenciation de cadre, la différenciation temporelle et la détection par infrarouge. Chacune de ces méthodes a ses forces et ses faiblesses et est particulièrement adaptée à certaines applications.

La détection de mouvement par flux optique excelle dans la visualisation de mouvements globaux dans des scènes complexes, mais peut rencontrer des difficultés avec des mouvements rapides ou minimes. La soustraction de fond, quant à elle, est particulièrement efficace dans des environnements stables où le fond reste constant, mais peut être perturbée par des changements dynamiques de l'environnement.

La différenciation de cadre et la différenciation temporelle sont des techniques simples et directes pour détecter des mouvements, mais elles peuvent être sujettes à des erreurs dans des conditions d'éclairage variables. Enfin, la détection de mouvement par infrarouge est très sensible et fiable pour détecter des objets émettant de la chaleur, mais peut être affectée par des sources de chaleur externes.

Alors que ces techniques traditionnelles ont beaucoup contribué à la détection de mouvement dans divers domaines, elles rencontrent toutes des limites, en particulier lorsqu'il s'agit de détecter de petits objets en mouvement dans des environnements complexes ou changeants. C'est là que l'inspiration biologique, en particulier à partir des insectes, pourrait offrir de nouvelles perspectives et solutions, comme nous le verrons dans le chapitre suivant.

# Formulation des modèles bio-inspiré

## Introduction et présentation

Dans le chapitre I, nous avons mis en évidence les limites des modèles conventionnels de détection du mouvement des objets de petites tailles. Ces limites qui sont principalement la complexité et/ou la mobilité des fonds, et aussi la taille très réduite des cibles.

C’est pour palier à ces problèmes que nous allons nous intéresser aux modèles bio-inspirés ; plus précisément les modèles de détection de mouvement basé sur le système visuel des insectes.

Figure :Schéma en bloc des différents types de détection bio-inspiré

D’une manière générale, les modèles de vision artificielle basée sur le système visuel des insectes peuvent être classés en 03 grandes catégories [7] qui sont :

* Looming Sensitive Models (LSM). Ces modèles sensibles aux profiles ou à l’approche sont principalement utilisés pour la détection des collisions. Par exemple les robots mobiles, les UAVs et les véhicules terrestres. Il est inspiré par les systèmes visuels de la sauterelle. Le système inclut deux modèles neuronaux : le LGMD1 et le LGMD2.
* Translation Sensitive Models (TSM). Ces modèles sensibles aux translations, principalement utilisés pour déterminer les directions prises par des objets. Il est inspiré par les neurones de détection de mouvement directionnellement sélectifs (DSN) chez les insectes.
* Small Target Motion Sensitive Models (STMSM**).** Ces modèles sensibles au mouvement des cibles de petites tailles sont caractérisés par les détecteurs STMDs. Ils sont efficace pour la détection des petites cibles même si l’environnement est complexe ou en mouvement.

Sans perdre de temps ni d’énergie nous allons nous focaliser sur le troisième modèle qui fait l’objet de notre projet.

## Approche biologique et applications

Dans les systèmes visuels des insectes, une classe de neurones sensibles aux mouvements à champ visuel spécifique, appelé les détecteurs STMD est caractérisé par sa réactivité exquise pour les mouvements des cibles de petites tailles [7]. Ces neurones ont été observés chez plusieurs groupes d’insectes parmi lesquelles on a les sphinx (Figure 6-a), les syrphes (Figure 6-b) et les libellules (Figre 6-c). C’est au cours de ces deux dernières décennies que l'anatomie et la physiologie des neurones STMD ont été étudiées plus en détail grâce aux chercheurs tels que Nordstro K, Barnett PD, O’Carroll DC etc. Voir les articles [10, 11, 12, 13] pour plus de détails sur ces recherches.

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |

*Figure 7: (a) un sphinx. (b) un syrphe. (c) Une libellule.*

La différence principale entre le STMD et d'autres neurones, tels que le LGMD, le DSN, etc., est que le STMD est très sensible pour les mouvements à petit champ. Plus précisément, le STMD montre des réponses maximales à des cibles compris entre 1 à 3◦ du champ de vision, mais n'a aucune réponse aux barres plus grandes (généralement > 10◦). Tandis que les autres neurones sont plutôt sensibles au mouvement à champ large.

Pour démontrer clairement la sélectivité de taille des neurones STMD, la réponse de ces neurones pour des cibles de taille variables est présentée dans les figures 9-a et 9-b. À partir de la figure 9-a, on peut voir que les deux cibles plus petites dont les tailles sont égales à 0,8◦ et 3◦, respectivement, peuvent susciter une réponse neuronale plus forte des neurones STMD. Cependant, la réponse à la cible plus grande dont la taille est égale à 15◦ est beaucoup plus faible.

Dans la figure 9-b, on peut voir que le STMD a une sensibilité optimale correspondant à la réponse neuronale la plus forte (aux alentours de 2°). Lorsque la hauteur de la cible est supérieure ou inférieure à cette taille optimale, la réponse neuronale diminuera significativement.

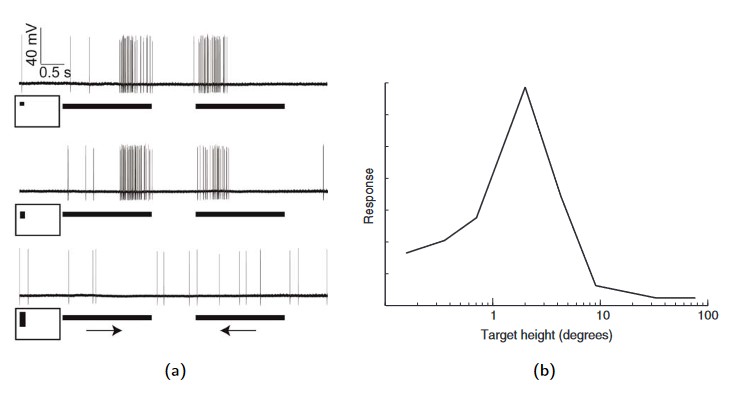


Figure 8: Réponse des neurones STMD. (a) Réponses neuronales de STMD au mouvement de trois cibles de tailles différentes (0,8°, 3° et 15° de hauteur sur 0,8° de largeur) dérivant sur des arrière-plans lumineux : les barres horizontales indiquent la durée du mouvement et les flèches indiquent la direction du mouvement de la cible (Figure 28-a du document [7]). (b) La réponse d'un STMD à des cibles de hauteur variable (Figure 28-b du document [7]).

Les recherches [10] et [11] ont également permit de mettre en évidence la sélectivité directionnelle des neurones STMD. Ces neurones réagissent fortement aux mouvements de cibles orientées selon la "Preferred Direction" (PD), mais montrent une réponse plus faible ou nulle aux mouvements dans la "Non-Preferred Direction" (ND).

**PD et ND :**

Dans le contexte de la recherche sur le système visuel et les neurones, les termes "Preferred Direction" (PD) et "Non-Preferred Direction" (ND) sont utilisés pour décrire la direction du mouvement à laquelle une cellule ou un neurone répond le plus ou le moins fortement.

La "Preferred Direction" (PD) est la direction du mouvement qui provoque la réponse la plus forte d'un neurone ou d'une cellule. Par exemple, si un neurone répond le plus fortement lorsqu’un stimulus se déplace de gauche à droite, on dirait que sa "Preferred Direction" est de gauche à droite.

Inversement, la "Non-Preferred Direction" (ND) est la direction du mouvement qui provoque la réponse la plus faible ou aucune réponse d'un neurone ou d'une cellule. En utilisant l'exemple précédent, si le neurone répond faiblement ou pas du tout lorsque le stimulus se déplace de droite à gauche, on dirait que sa "Non-Preferred Direction" est de droite à gauche.

Ces termes sont importants dans l'étude du système visuel et des neurones parce qu'ils aident à comprendre comment les cellules et les neurones traitent et interprètent les informations visuelles. Par exemple, des neurones avec des "Preferred Directions" différentes peuvent travailler ensemble pour aider un organisme à comprendre la direction générale du mouvement dans son environnement.

Selon les documents que nous avons exploités [10, 11, 12, 13], la "Preferred Direction" spécifique des neurones STMD peut varier selon les espèces d'insectes et les conditions spécifiques. En général, les neurones STMD sont connus pour leur capacité à répondre fortement au mouvement de petits objets contre des arrière-plans en mouvement ou encombrés, indépendamment de la direction et de la vitesse du mouvement de l'arrière-plan. La "Preferred Direction" pour un neurone STMD spécifique pourrait être déterminée par des facteurs tels que la direction du mouvement de la cible qui génère la réponse la plus forte chez ce neurone.

La figure 10-a illustre les réponses d'un neurone STMD sélectif (en direction) face à trois cibles de tailles différentes, montrant que la plus grande cible, ayant une hauteur de 15°, ne peut pas activer le neurone STMD, même lorsqu'elle se déplace dans la direction PD. Cependant, le neurone STMD réagit fortement au mouvement PD de cibles plus petites, ayant des hauteurs de 0.8° et 3°, tout en restant inactif lorsque ces cibles plus petites se déplacent dans la direction ND. Des recherches supplémentaires [11, 12] ont montré que la sélectivité de taille et de direction des neurones STMD est indépendante du mouvement de fond. En d'autres termes, les neurones STMD répondent robustement au mouvement de petites cibles contre des arrière-plans visuellement encombrés, indépendamment de la direction et de la vitesse du mouvement de fond.

La figure 10-b montre que le neurone STMD répond fortement à de petites cibles se déplaçant dans la direction PD (vers le bas), mais faiblement à celles se déplaçant dans la direction ND (vers le haut). Cette réponse est cohérente, quel que soit la direction ou la vitesse du mouvement de fond.

En résumé, les neurones STMD sont sélectifs (en direction) et peuvent reconnaître le mouvement de petites cibles même en l'absence de mouvement relatif entre les objets en mouvement et l'arrière-plan.

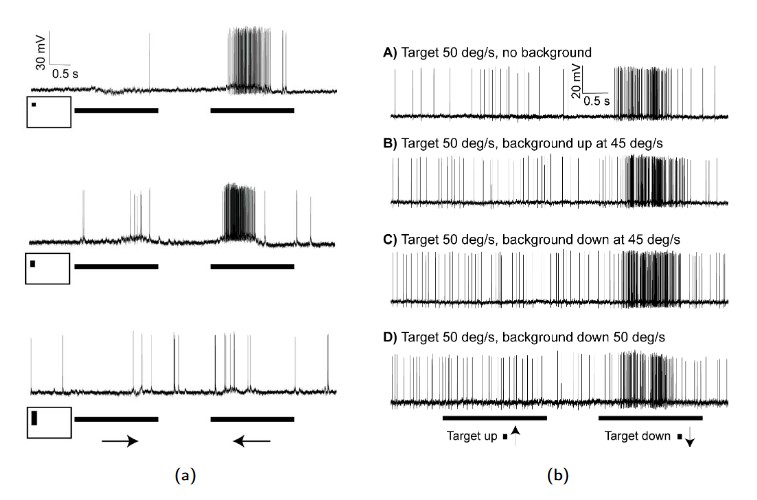


Figure  : Réponse directionnelle des neurones STMD. (a) Réponses brutes du neurone STMD sélectif (en direction PD ou ND) qui préfère le mouvement de la cible vers la gauche, testé par le mouvement de trois cibles de tailles différentes (0,8◦, 3◦ et 15◦ de hauteur sur 0,8◦ de largeur) dérivées contre des arrière-plans lumineux : les barres horizontales indiquent la durée des stimuli et les flèches indiquent la direction du mouvement de la cible (Figure 29-b du document [7]). (b) Réponses du neurone STMD qui préfère le mouvement de la cible vers le bas, aux cibles dérivées contre des arrière-plans encombrés (Figure 29-b du document [7]).

**Applications :**

Plusieurs modèles mathématiques ont été proposés au cours de la dernière décennie pour simuler le STMD sur la base de découvertes biologiques. L'un des premiers modèles est le détecteur de mouvement de petite cible élémentaire (ESTMD) développé par Steven D. Wiederman, Patrick A. Shoemaker, et David C. O'Carrol [1]. Ce modèle rend compte de la sélectivité de taille. Cependant, le modèle ESTMD est incapable de réaliser la DS (Direction Selective) du STMD révélée par les biologistes. Pour tenir compte de la DS, deux modèles hybrides ont été proposés : l'ESTMD-EMD et l'EMD-ESTMD, pour réaliser la DS du STMD [2]. Plus précisément, l'ESTMD-EMD indique que l'ESTMD est en cascade avec l'EMD, tandis que l'EMD-ESTMD indique que l'EMD est en cascade avec l'ESTMD. Ces deux modèles hybrides ont été utilisés avec succès pour le suivi de cibles sur des arrière-plans encombrés dans un robot mobile autonome terrestre [22, 23, et 24].

Afin de progresser plus rapidement, nous nous concentrerons davantage sur les modèles développés par Wang et ses équipes [3, 4, 5, 6, 7, et 9]. Ces modèles sont basés sur les travaux de Wiederman et al. et ont l'avantage d'être plus récents.

Figure 10: Modèles et Application du système visuel des insectes [7]

## ESTMD avec un nouveau mécanisme d'inhibition latérale (LIM)

### Présentation

Sur la base du modèle ESTMD (présenté par Wiederman et al.), Wang et al. ont proposé un nouveau modèle avec un nouveau mécanisme d'inhibition latérale. Inspiré par le processus visuel biologique chez la mouche, leur modèle proposé est composé de quatre couches neuronales : la rétine, la lamina, la médulla et la lobula.

Dans les sections précédentes, nous avons fait remarquer que la détection de petites cibles en mouvement dans des arrière-plans encombrés est une tâche cruciale pour les animaux. Et que récemment, les biologistes ont découvert un type spécifique de neurone dans le complexe de la lobula, connu sous le nom de STMDs, qui fait preuve d'une sélectivité extraordinaire pour la détection de petites cibles dans un encombrement visuel. De plus, certains chercheurs soutiennent que l'inhibition latérale joue un rôle significatif dans la différenciation du mouvement de la cible par rapport au mouvement de l'arrière-plan et pourrait même expliquer de nombreuses caractéristiques d'ajustement des neurones visuels de niveau supérieur. Inspiré par la découverte que l'inhibition latérale complète ne se produit que lorsque les régions centrale et périphérique se déplacent de manière identique, Wang et al. ont proposé un nouveau mécanisme d'inhibition latérale qui utilise la vitesse et la direction du mouvement pour améliorer les performances du modèle ESTMD (détecteur de mouvement de petite cible élémentaire). Cette section présente le modèle ESTMD avec le mécanisme innovant d'inhibition latérale.

Comme indiqué dans [1] et [11], le mécanisme d'inhibition latérale pourrait jouer un rôle crucial dans le système de traitement visuel des insectes et aider à façonner l'ajustement de la réponse aux petites cibles. Cependant, sa mise en œuvre n'a pas été claire. Wang et al. suggèrent que l'inhibition latérale est un mécanisme biologique omniprésent dans le système de traitement visuel des insectes, et des études récentes sur l'inhibition latérale [18] ont révélé que certains neurones ne répondent au mouvement local sur la rétine que lorsque la trajectoire du mouvement est différente de celle d'une grande région environnante. Plus précisément, les auteurs ont découvert que ces neurones répondent lorsqu'un objet dans le centre de leur champ récepteur se déplace par rapport à l'arrière-plan, mais sont presque entièrement supprimés lorsque l'objet se déplace avec l'arrière-plan. De plus, les résultats de [13] démontrent que l'inhibition latérale est sélective en fonction de la vitesse et de la direction. En fait, l'inhibition latérale est la plus prononcée lorsque l'objet et la cible distracteur se déplacent à la même vitesse et dans la même direction à l'intérieur d'une certaine plage, mais reste faible ou silencieuse lorsque l'objet et la cible distracteur se déplacent à la même vitesse mais dans des directions de mouvement différentes.

Bien que l'ESTMD proposé par [1] contienne deux mécanismes d'inhibition latérale, situés respectivement dans la couche lamina et la couche médulla, ces deux mécanismes d'inhibition latérale ne sont pas totalement en accord avec les découvertes biologiques mentionnées ci-dessus. En fait, selon ces deux mécanismes d'inhibition latérale, l'objet reçoit la même quantité d'inhibition latérale, qu'il y ait un mouvement relatif entre l'objet et l'arrière-plan ou non. Le mouvement de l'objet peut être considérablement affaibli par le mouvement de l'arrière-plan, même lorsque le mouvement de l'objet diffère de celui de l'arrière-plan. Par conséquent, les performances de détection de l'ESTMD sont instables, en particulier lorsque l'objet se déplace à travers un arrière-plan en mouvement encombré.

Inspirés par le phénomène d'inhibition latérale dépendant de la vitesse et de la direction du mouvement mentionné ci-dessus, Wang et al. ont amélioré le modèle ESTMD proposé en [1] avec un nouveau mécanisme d'inhibition latérale qui prend en compte la vitesse et la direction du mouvement. Ils ont démontré que leur nouvelle inhibition latérale surpasse l'inhibition latérale proposée dans [1] et peut améliorer les performances de détection des cibles.

### Modélisation

Photoreceptor

***Image***

***Preprocessing***

Luminance or Gray Scale Image

Input Image

**I**

**L**

Low Pass Filter

Lipetz

Transformation

**L**

**Retina**

**Layer**

**Lc**

Low Pass Filter

**P**

High Pass Filter

(LMCs)

**Lamina**

**Layer**

**x**

**YON**

**YOFF**

FDSR

FDSR

**Medulla**

**Layer**

**-SON**

**-SOFF**

**YON**

**YOFF**

**SON**





**FON**

**FOFF**

HW-R

HW-R

**HWON->FON**

**HWOFF->FOFF**

**Lobula**

**Layer**



LI

LI



FDSR

**LobOFF**



**O**

Figure 11: Schéma d’ESTMD avec un nouveau mécanisme d’inhibition latéral. [9]

### La couche Retina

La couche de l'œil appelée rétine (ou retina) est constituée d'une matrice de M rangées et N colonnes de photorécepteurs, chacun correspondant à un point de pixel. Ces photorécepteurs reçoivent des niveaux de luminance ou de gris d'images consécutives. Pour simuler le flou spatial causé par l'optique de la mouche, Wang et al. convolvent l'intensité de chaque pixel  dans une trame d'image à l'instant t, notée par, avec un masque de convolution Gaussien. C'est-à-dire :

 (1)

Où le noyau gaussien vaut:

 (2)

Dans notre cas ici, l'opération de flou vise à atteindre deux objectifs principaux. Premièrement, elle contribue à la réduction du bruit en lissant et en atténuant l'impact des fluctuations aléatoires ou des petites variations dans l'intensité des pixels. Deuxièmement, elle facilite l'extraction de caractéristiques en moyennant les détails fins d'une image, ce qui permet une détection plus aisée des éléments plus importants et significatifs au sein de celle-ci.

Après le flou spatial, les photorécepteurs transforment la luminance d'entrée en potentiel de membrane. Ce processus est mis en œuvre en utilisant la fonction Lipetz avec l'exposant u fixé à 0.7.

 (3)

 dans l'équation (3) est la version filtrée en basse fréquence de  et satisfait la relation suivante.



 (4)

Où  est une constante de temps.

(4) est une équation différentielle ordinaire (ODE) du premier ordre sous la forme de :



Avec et b  ; notons que est une constante pour chaque t.

La solution de l’ODE est :



Où  est une constante.

Dans les autres couches, nous allons rencontrer des ODE de la même forme que (4) et leurs solutions resteront les même dépendant des constantes a, b et.

La transformation en potentiel de membrane est une étape essentielle du processus de détection de la lumière dans le système visuel, que ce soit chez les humains, les autres animaux ou dans les modèles bio-inspirés basés sur le système visuel des insectes.

Dans le contexte de la vision, le potentiel de membrane est le signal électrique qui est généré par un photorécepteur (une cellule sensible à la lumière dans l'œil) en réponse à la lumière. Lorsque de la lumière frappe un photorécepteur, cela provoque une modification de la tension électrique à travers la membrane cellulaire du photorécepteur, créant un potentiel de membrane. Ce potentiel de membrane est ensuite transmis aux autres cellules du système visuel, où il est traité pour produire une perception de la lumière et du mouvement.

Dans le contexte de notre travail, la transformation en potentiel de membrane est une façon de simuler comment un photorécepteur réel transforme la lumière en signal électrique. Cela permet au modèle d'imiter plus fidèlement le fonctionnement du système visuel réel et peut aider à améliorer la précision de la détection du mouvement des petits objets.

### La couche Lamina

La sortit de la couche Rétine correspond à l’entrée de la couche Lamina avec un léger retard modélisé par l’équation différentielle suivante :

  (5)

Où  est une constante de temps et (5) une ODE de premier ordre.

Ce léger retard dans la transmission de l'information entre la couche rétinienne et la couche laminaire est une représentation du temps de traitement biologique nécessaire à l'information pour passer d'une cellule à une autre dans le système visuel. Dans le système visuel réel, il y a un petit délai entre le moment où une cellule (comme un photorécepteur dans la rétine) détecte une information (comme la lumière) et le moment où cette information est traitée et transmise à la cellule suivante (comme une cellule dans la lamina).

Dans le contexte de notre modèle, ce retard pourrait aider à rendre le modèle plus précis et plus réaliste en tenant compte du temps de traitement nécessaire dans un système visuel réel. Il pourrait également jouer un rôle dans la manière dont le modèle traite l'information visuelle et détecte le mouvement des petits objets.

Le signal retardé est utilisé comme entrée pour les grandes cellules monopolaire (LMCs) situées dans la couche laminaire. Les articles [19] et [20] sur les LMCs suggèrent qu'elles peuvent éliminer l'information redondante et améliorer la transmission de l'information. La fonctionnalité des LMCs peut être exprimée en utilisant les équations suivantes :



 (6)

 (7)

Où   est la sortit des cellules LMC ;  est la version passe-bas du filtre de premier ordre de  ;  est une constante de temps.

### La couche medulla

La sortie des cellules LMCs () est transmise à la couche de la médulla, où elle est divisée en canaux ON et OFF dans la partie initiale de la couche de la médulla. Cette opération peut être représentée mathématiquement comme suit :

 (8)

 (9)

Où  et  sont respectivement les signaux des canaux ON et OFF.

Pour simuler des processus biophysiques crédibles, les canaux ON et OFF pour chaque pixel (i, j) sont transformés en un "état d'adaptation" en les soumettant à un filtre passe-bas non linéaire présentant un attribut de dépolarisation rapide et de repolarisation lente [21]. Connue sous le nom de mécanisme FDSR (Fast Depolarization, Slow Repolarization), il supprime efficacement les informations de texture qui changent rapidement et améliore le contraste des changements novateurs.

Le mécanisme FDSR ou dépolarisation rapide et repolarisation lente, est une caractéristique clé de certains neurones et modèles neurologiques. Il est utilisé pour modéliser comment les neurones, ou dans ce cas les modèles bio-inspirés basés sur le système visuel des insectes, traitent l'information.

Dans le contexte de notre modèle, le FDSR sert à filtrer l'information visuelle de manière à améliorer la détection de petits objets en mouvement. Plus précisément, il supprime les informations de texture qui changent rapidement et améliore le contraste des nouveautés ou des changements dans l'image. Cela peut rendre le modèle plus efficace pour détecter les petits objets qui se déplacent contre un arrière-plan complexe ou en mouvement.

Par ailleurs, la "repolarisation lente" de ce mécanisme joue également un rôle important. Elle permet aux neurones ou aux éléments du modèle de revenir lentement à leur état de repos après une dépolarisation, ce qui peut aider à accentuer ou à mettre en évidence les changements plus lents ou plus durables dans l'image.

Ensemble, ces deux processus permettent au mécanisme FDSR de filtrer efficacement l'information visuelle, en supprimant le bruit et en mettant en évidence les éléments importants, tels que les petits objets en mouvement.

En somme, le FDSR permet d'améliorer la précision et l'efficacité du modèle en aidant à différencier les petits objets en mouvement du bruit de fond et en mettant en évidence les changements nouveaux ou importants dans l'image.

Nous désignons respectivement par, comme étant le signal des canaux ON et OFF après FDSR, alors :





Posons  et  ; on a :



 (10)

Où et sont des constantes de temps et vérifie.

De même, on a :

 (11)

Ensuite, les signaux filtrés, est soustrait des signaux originaux,.

 (12)

 (13)

Où,  sont les sorties de la couche medulla.

Une fois que le mécanisme FDSR est appliqué, les signaux résultants et sont transmis à un redresseur en demi-onde (HW-R). Les canaux ON et OFF sont alors notés et après le processus de redressement en demi-onde. Alors

 (14)

 (15)

Pour simplifier l’écriture par la suite, Wang et al. utilisent et pour désigner le signal après redressement en demi-onde. Des recherches précédentes ont proposé un mécanisme d'inhibition latérale où les canaux centraux ON et OFF sont inhibés par les canaux environnants de la même polarité [1]. Cependant, ce mécanisme semble contredire des découvertes récentes, qui montrent que l'inhibition latérale est seulement forte lorsque les régions centrales et périphériques se déplacent à la même vitesse et dans la même direction [13], [18]. Par conséquent, un mécanisme plus raisonnable devrait prendre en compte la vitesse et la direction du mouvement. Dans l'article [9], Wang, et al. proposent un nouveau mécanisme d'inhibition latérale basé sur la vitesse et la direction du mouvement, et calculent le vecteur de vitesse de chaque pixel. Alors que le modèle EMD peut détecter le mouvement et la direction, il dépend du contraste et de la vitesse, ce qui rend leurs réponses ambiguës. Par conséquent, ils utilisent un algorithme de correspondance générale pour calculer le vecteur de mouvement pour chaque pixel. Le critère de correspondance est défini comme suit :

 (16)

Où est la taille de la fenêtre de recherche et  est l'image d'entrée à l'instant t.

Le vecteur de mouvement, qui est le vecteur de translation, est obtenu en identifiant la valeur minimale du paramètre D.

 (17)

Où et est la plage de recherche.

En utilisant l'algorithme de correspondance générale mentionné précédemment, nous pouvons calculer le vecteur de mouvement  pour chaque pixel, où et représentent respectivement les composantes horizontales et verticales du vecteur de mouvement. Par conséquent, nous pouvons définir la matrice du vecteur de mouvement  et pour une image d'entrée comme suit :

 (18)

 (19)

Pour calculer la différence de vitesse entre les régions centrales et périphériques, nous convoluons U et V avec H, ce qui peut s'exprimer comme suit :

 (20)

 (21)

Où \* est l'opérateur de convolution et

 (22)

Pour clarifier la fonction de et , Wang et al. commencent par définir un voisinage (PR) pour le pixel (i, j),

 (23)

Où , , , sont déterminés par la taille de la petite cible.

Si le vecteur de mouvement du pixel est le même que le vecteur de mouvement d'un pixel s dans la région périphérique (PR), c'est-à-dire

 (24)

Alors, après avoir convolué et  avec , et  deviennent nuls. A l'inverse, s'il y a une différence de vitesse entre le pixel  et sa région périphérique,  et  deviennent non nuls. L'amplitude de  et  ( et ) est un indicateur de la différence de vitesse entre les régions centrales et périphériques (). Plus l'amplitude est élevée, plus la différence de vitesse est grande. La différence de vitesse totale entre le pixel et son voisinage est donnée par l'équation :

 (25)

Le mécanisme d'inhibition latérale proposé par Wang et al. est mis en œuvre en multipliant le signal des canaux ON et OFF par .

 (26)

 (27)

La raison de ce mécanisme est d'inhiber le mouvement de fond et d'améliorer le mouvement de petites cibles, car c'est plus plausible biologiquement. Le nouveau mécanisme est basé sur l'observation que si un pixel appartient à l'arrière-plan, son vecteur de mouvement sera le même que le vecteur de mouvement de sa région périphérique. Dans ce cas, le signal du pixel devrait être fortement inhibé. Cependant, si un pixel appartient à une cible en mouvement, alors son vecteur de mouvement ne sera pas le même que le vecteur de mouvement des autres pixels dans sa région périphérique, à moins que la petite cible et l'arrière-plan aient le même mouvement. Dans ce cas, le signal du pixel devrait être renforcé. L'ampleur de l'inhibition latérale dépend de la valeur de, et peut être ajustée par le paramètre. Ce mécanisme renforce la saillance des petites cibles en mouvement plus petites que et inhibe les faux positifs causés par le mouvement de l'arrière-plan en mouvement, améliorant ainsi la performance du modèle.

### La couche lobula

La couche lobula présente une corrélation entre le canal OFF, qui a un retard, et le canal ON non retardé. Le retard dans le canal OFF est obtenu grâce à un filtre passe-bas de premier ordre, et la longueur du retard dépend de la taille et de la vitesse de la petite cible.

 (28)

Et la sortie finale de la couche lobula est :

 (29)

## Conclusion

Dans ce chapitre, nous avons fait le tour des différentes couches du modèles ESTMD. Nous avons traité et commenté les équations qui définissent ses couches. Le chapitre suivant sera consacré pour l’implémenter et l’analyse des résultats.

# Implémentation et discussion

## Introduction

Dans cette section, nous allons créer une classe Python que nous appellerons STMD. Cette classe sera basée sur les différentes équations et couches du modèle STMD étudiés dans le chapitre 2. Notre tâche ici consistera à implémenter ces équations, à varier les différents paramètres pour obtenir des résultats intéressants. La classe STMD devrait permettre la détection de petits mouvements de cibles dans un arrière-plan relativement complexe et potentiellement mouvant. Les différentes méthodes de la classe Python STMD représenteront les couches des détecteurs STMD.

Pour la suite, nous utiliserons l'éditeur ATOM [25] pour écrire nos codes; et Windows PowerShell [26] pour les installations de packages, la gestion des environnements virtuels, et l'exécution des scripts."

Pour la suite, nous allons utiliser la même vidéo V1 et afficher le frame V1-47 (fig. 1) comme décrit dans l’introduction générale. Puis nous afficherons les résultats couches par couches jusqu’à la sortit finale.

Les résultats après chaque couche se trouvent dans l’annexe A1 [Figure 13 à 24].

Toutes les ressources et les codes source sont disponibles sur notre compte GitHub.

<https://github.com/boukary-derra/pfe/tree/main/code>

Lorsqu’on regarde de très près la figure 11, on remarque qu’on peut subdiviser notre modèle en deux parties que nous allons appeler ici détection de cible « target detection » et détection de mouvement « motion detection ».

* Target detection : formé par les deux première couche consiste essensiellement à la mise en évidence de la cible par rappord au fond. Cette partie à juste besoin du frame courant pour fonctionner.
* Motion detection : formé par les deux dernières couches (Medulla et Lobula). Cette section prend en compte le déplacement des éléments présent sur le frame il a donc besoin de tenir compte du frame passé dans ses opérations.

## Détection de cibles (Retina et Lamina)

Cette section du modèle à pour rôle de mettre en évidence les objets par rapport au fond de l’image. Il est constitué par les couches de la rétine et le lamina.

**Couche Retina :**

Dans la couche retina, le module photorécepteur applique un effet de flou à l'image d'entrée en utilisant un filtre gaussien. Et l’effet de flou ici a pour but de réduire le bruit et les détails dans chaque frame. Théoriquement, nous devons parcourir chaque pixel de l'image et appliquez le noyau gaussien aux pixels voisins. [Equation 1].

|  |  |
| --- | --- |
|  |  |
|  |  |

Ensuite, le module Lipetz transformation applique une transformation de Lipetz modifiée à l'image. C'est une technique couramment utilisée en bio-inspiration, et plus précisément dans la modélisation de la réponse des photorécepteurs, pour transformer la luminance d'entrée en potentiel de membrane. La transformation de Lipetz consiste à utiliser un filtre passe-bas sur l'image, puis à effectuer une transformation non linéaire sur le rapport entre l'image originale et l'image filtrée. [Equation 3 et 4].

|  |  |
| --- | --- |
|  |  |
|  |  |

**Couche Lamina :**

Dans cette couche, on applique un passe-bas à l'image transformée par Lipetz, ce qui donne le léger délai décrit plus haut dans le chapitre 2.

|  |  |
| --- | --- |
|  |  |
|  |  |

Puis nous effectuons une opération de type "Centre-surround", couramment utilisée dans les modèles de détection pour réduire les informations redondantes et maximiser la transmission de l'information. Dans cette opération, nous appliquons encore un autre filtre passe-bas à l'image, puis nous soustrayons le résultat de l'image d'origine. C'est une manière courante de mettre en évidence les contours dans une image, car cela permet de soustraire la moyenne locale de chaque pixel, mettant ainsi en évidence les régions où l'intensité change rapidement (c'est-à-dire les contours).

|  |  |
| --- | --- |
|  |  |
|  |  |

A partir de cette section déjà, nous commençons à avoir des résultats intéressants. Nous somme capable de détecter des petites cibles dans des arrière-plans relativement complexes. L’implémentation des autres couche serait donc d’essayer de tenir compte de la complexité des font dans lesquels se déplace notre cible.

Voici globalement comment se présentera notre class python STMD :



## Initialisation de la class STMD

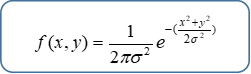
Dans la méthode \_\_init\_\_(), nous commençons par une initialisation claire des variables et des contrôles de validité pour les entrées en utilisant np.ndarray de la bibliothèque numpy. Cette bibliothèque est un bon choix pour le traitement d'image en Python en raison de sa haute performance pour les opérations matricielles.

Ensuite nous initialisons une variété de paramètres et de constantes dans notre méthode, y compris un filtre gaussien, des paramètres pour une transformation de Lipetz, des paramètres pour un filtre passe-bas, et des constantes pour le modèle LMC.



## Le noyau gaussien

Le noyau gaussien est généré en utilisant la formule de la densité de probabilité de la distribution normale (gaussienne), qui est défini comme suite :



Où

*  est l'écart-type de la distribution,
* x est la distance au centre du noyau, et
* e est la constante d'Euler (la base du logarithme naturel).

Dans l’annexe [A5], nous avons montré de façons détaillé comment obtenir cette formule de la densité.

La méthode get\_gaussian\_kernel()génère une matrice de taille (size, size) représentant un noyau de convolution gaussien, qui est souvent utilisé pour lisser une image.



Il est important de noter que la taille du noyau (size) doit être un nombre impair, de sorte qu'il y ait un pixel central unique.

Si une erreur se produit lors de la génération du noyau, notre méthode retourne self.w comme noyau par défaut et imprime l'erreur.

## Photorécepteur

La méthode get\_photoreceptor() applique un effet de flou à l'image d'entrée en utilisant un filtre gaussien. Et l’effet de flou ici a pour but de réduire le bruit et les détails dans chaque frame. Théoriquement, nous devons parcourir chaque pixel de l'image et appliquez le noyau gaussien aux pixels voisins [Equation 1]; cela reviendra à utiliser une boucle for imbriquée pour appliquer le filtre à l’image.



En Python, les boucles for peuvent être assez lentes, surtout pour des images de grande taille. Une approche plus efficace serait d'utiliser donc la fonction scipy.ndimage.filters.convolve de la bibliothèque scipy, qui peut effectuer une convolution 2D sur l’image avec le noyau gaussien. Cela éliminerait le besoin de boucles for imbriquées, ce qui pourrait considérablement accélérer l’exécution de notre code.

* 1. **Lipetz transformation**

La méthode get\_lipetz\_transformation() applique une transformation de Lipetz modifiée à l'image d'entrée. C'est une technique couramment utilisée en bio-inspiration, et plus précisément dans la modélisation de la réponse des photorécepteurs, pour transformer la luminance d'entrée en potentiel de membrane.

La transformation de Lipetz consiste à utiliser un filtre passe-bas sur l'image, puis à effectuer une transformation non linéaire sur le rapport entre l'image originale et l'image filtrée.



## Le filtre passe bas

La méthode get\_low\_pass\_filter() applique un filtre passe-bas à l'image transformée par Lipetz, ce qui donne un léger délai, comme nous l’avons noté en commentaire dans le code.



## LMC

La méthode get\_lmc() implémente une opération de type "Centre-surround", couramment utilisée dans les modèles de détection de mouvement pour réduire les informations redondantes et maximiser la transmission de l'information.

Dans cette méthode, nous appliquons encore un autre filtre passe-bas à l'image, puis nous soustrayons le résultat de l'image d'origine. C'est une manière courante de mettre en évidence les contours dans une image, car cela permet de soustraire la moyenne locale de chaque pixel, mettant ainsi en évidence les régions où l'intensité change rapidement (c'est-à-dire les contours).

A partir de cette couche déjà, nous commençons à avoir des résultats intéressants. Nous somme capable de détecter des petites cibles mobiles dans des arrière-plans moins complexes. L’implémentation des autres couche serait donc d’essayer de tenir compte de la complexité des font dans lesquels se déplace notre cible.



## Les canaux ON / OFF

La méthode get\_on\_off\_channels() est utilisée pour séparer les canaux ON et OFF à partir du résultat de la couche LMC. C'est une approche courante dans les modèles de détection de mouvement bio-inspirés, où les canaux ON et OFF sont utilisés pour détecter respectivement les augmentations et les diminutions de l'intensité lumineuse.

Ici, nous effectuons cela en calculant la moitié de la somme (pour le canal ON) et la moitié de la différence (pour le canal OFF) de l'image LMC et de sa valeur absolue [Equations 8 et 9]. Cela a pour effet de séparer les valeurs positives et négatives de l'image LMC en deux images distinctes.



## FDSR

La méthode get\_fdsr()implémente un mécanisme de suppression des changements rapides de texture et d'amélioration des changements de contraste. Ce type de mécanisme est souvent utilisé dans les modèles de détection de mouvement pour filtrer les changements de texture qui ne sont pas pertinents pour la détection de mouvement.

Dans cette méthode, nous comparons les images ON et OFF actuelles avec celles du dernier cadre, et nous appliquons un flou gaussien différent en fonction de si l'intensité a augmenté ou diminué. Cela a pour effet de réduire la réponse aux changements de texture qui se produisent rapidement dans le temps, tout en améliorant la réponse aux changements de contraste.



## SIGMA

La méthode get\_sigma() calcule la différence entre le signal original (y\_on et y\_off) et le signal filtré (s\_on et s\_off). C'est une approche courante dans les modèles de détection de mouvement pour isoler les changements importants qui se sont produits entre les images.

Nous faisons cela en utilisant la fonction cv2.subtract pour soustraire les images filtrées des images originales. Cette fonction effectue une soustraction élément par élément entre les deux images, ce qui est exactement ce que nous voulons dans ce cas.

Cependant, nous devons nous assurer que les images sont du même type avant de les soustraire. C’est pour cela que nous les convertissons np.float32 avant la soustraction.



## HWR

La méthode get\_hwr() implémente une fonction de rectification demi-onde (half-wave rectification), qui est souvent utilisée dans les modèles de détection de mouvement pour supprimer les valeurs négatives.

Nous avons fait cela en créant une fonction max\_with\_zero() sur les images f\_on et f\_off. Cette fonction renvoie le maximum entre l'image et zéro, effectué pixel par pixel, ce qui aurait pour effet de remplacer toutes les valeurs négatives par zéro.



Où :



Alternativement, nous pourrions obtenir le même effet en utilisant la fonction np.maximum de numpy, qui effectue également une opération max élément par élément max\_image = np.maximum(image, 0). Cependant, l'utilisation de cv2.max est légèrement plus rapide pour de grandes images, car OpenCV est optimisé pour le traitement d'images.



## L’inhibition latérale

### Block based motion estimation

La fonction block\_based\_motion\_estimation() divise l'image actuelle en blocs de taille donnée et trouve le vecteur de mouvement optimal pour chaque bloc en recherchant dans l'image précédente.



### Find optical motion vector

La fonction find\_optimal\_motion\_vector() calcule le vecteur de mouvement qui minimise la somme des différences absolues (SAD) entre le bloc actuel et chaque bloc déplacé dans l'image précédente. Cette fonction utilise une recherche exhaustive dans une plage de recherche donnée autour de la position actuelle du bloc.



En outre, l'utilisation d'une recherche exhaustive pour l'estimation du mouvement peut être très coûteuse en temps de calcul, surtout si la plage de recherche est grande. Il existe des méthodes plus efficaces pour l'estimation du mouvement basée sur des blocs, comme la recherche en spirale ou la recherche hiérarchique.



### Motion components

L'extraction des composantes de mouvement à partir des vecteurs de mouvement est assez simple. Les vecteurs de mouvement sont généralement stockés dans un tableau 3D, où le premier axe correspond aux lignes de l'image, le deuxième axe correspond aux colonnes de l'image, et le troisième axe correspond aux composantes du vecteur de mouvement (généralement l'horizontale et la verticale).

La fonction get\_motion\_components() que nous avons définie fonctionne en extrayant simplement les tranches correspondantes du tableau 3D. En Python, l'opérateur « : » est utilisé pour sélectionner tous les éléments le long d'un axe. Par conséquent, motion\_vectors[:,:,0] sélectionne tous les éléments le long du premier et du deuxième axe (c'est-à-dire toutes les lignes et toutes les colonnes), et la composante 0 le long du troisième axe (c'est-à-dire la composante horizontale du vecteur de mouvement). De même, motion\_vectors[:,:,1] sélectionne la composante verticale du vecteur de mouvement.

L'implémentation de la fonction serait comme suite :



### Le noyau de convolution H

Ces deux fonctions créent un noyau de convolution, puis convoluent deux images (ou matrices) avec ce noyau.

La fonction create\_convolution\_kernel(p, q, a) crée un noyau de convolution de taille p par q avec la valeur a au centre et la valeur b aux bords. La valeur b est calculée comme -a / (4 \* (q - 1)) pour maintenir la somme de tous les éléments du noyau égale à zéro.

La fonction convolve\_uv\_with\_h(U, V, H) prend deux images (ou matrices), U et V, et un noyau de convolution H, puis convolue chaque image avec le noyau en utilisant la fonction convolve2d de la bibliothèque scipy. Le paramètre mode='same' indique que la sortie doit avoir la même taille que l'entrée.

Voici les implémentations de ces fonctions :





### La matrice W

La fonction calculate\_w prend en entrée deux matrices U\_c et V\_c (qui peuvent être des images ou des composantes de vecteurs de mouvement), et calcule la norme euclidienne (ou distance euclidienne) pour chaque paire correspondante de points dans U\_c et V\_c. Cela se fait en élevant chaque matrice au carré, en les additionnant, puis en prenant la racine carrée de chaque élément du résultat. Le résultat w est une matrice de la même taille qu’U\_c et V\_c contenant les normes euclidiennes.

Voici l’implémentation de cette fonction :



Notons que cette fonction utilise le broadcasting de numpy pour effectuer les opérations d'élévation au carré, d'addition et de racine carrée sur chaque élément des matrices U\_c et V\_c.

Et enfin nous calculons le LI en utilisant les deux matrices F\_ON et F\_OFF (qui représentent les canaux ON et OFF d'une image), une matrice w (qui représente la magnitude du mouvement), et deux constantes k1 et k2. Chaque élément de F\_ON et F\_OFF est multiplié par w, puis additionné à lui-même. Nous obtenons deux nouvelles matrices F\_ON\_new et F\_OFF\_new qui sont ensuite renvoyées.

Une fois de plus, nous avons profité du broadcasting de numpy pour effectuer les opérations de multiplication et d'addition sur chaque élément des matrices F\_ON, F\_OFF, et w.



## Le retard et la sortie finale

La méthode get\_delay() applique un filtre passe-bas à l'image f\_off\_li en utilisant tau\_delay comme paramètre du filtre. Si f\_off\_li n'est pas fourni, il génère une erreur.

La méthode get\_final\_output() multiplie les images f\_on\_li et lob\_off ensemble. Si l'une de ces images n'est pas fournie, il génère une erreur.



## Conclusion

# CONCLUSION GENERALE

# BIBLIOGRAPGIE

[1] S. D. Wiederman, P. A. Shoemaker, and D. C. O’Carroll, “A model  
for the detection of moving targets in visual clutter inspired by insect  
physiology,” PLoS One, vol. 3, no. 7, pp. 1–11, Jul. 2008.

[2] S. D. Wiederman and D. C. O’Carroll, “Biologically inspired feature  
detection using cascaded correlations of off and on channels,” Dec. 2013.

[3] H. Wang, J. Peng, and S. Yue, “A directionally selective small target  
motion detecting visual neural network in cluttered backgrounds,” IEEE  
Trans. Cybern., vol. 50, no. 4, pp. 1541–1555, Apr. 2020.

[4] H. Wang, J. Peng, X. Zheng, and S. Yue, “A robust visual system for  
small target motion detection against cluttered moving backgrounds,”  
IEEE Trans. Neural Netw. Learn. Syst., vol. 31, no. 3, pp. 839–853,  
Mar. 2020.

[5] Ho. Wang, J. Zhao, Hu. Wang, J. Peng and S. Yue, “Attention and Prediction Guided Motion Detection for Low-contrast Small Moving Targets,”  
IEEE Trans, Dec. 2021.

[6] H. Wang, J. Peng, and S. Yue, “A Feedback Neural Network for Small Target Motion Detection in Cluttered Backgrounds,”  
July. 2018.

[7] Q. Fu, H. Wang, C. Hu, and S. Yue, “Towards Computational Models and Applications of Insect Visual Systems for Motion Perception: A Review,”  
IEEE Trans, Apr. 2019.

[8] Warrant, E.J.: Matched filtering and the ecology of vision in insects. In: The

Ecology of Animal Senses, pp. 143–167. Springer (2016)

[9] H. Wang, J Peng and S. Yue, “Bio-inspired Small Target Motion Detector with a new Lateral Inhibition Mechanism,”  
Conference Paper - July 2016.

[10]. Nordstro ̈m K, O’Carroll DC (2006) Small object detection neurons in female  
hoverflies. P Roy Soc B-Biol Sci 273: 1211–1216.

[11]. Nordstro ̈m K, Barnett PD, O’Carroll DC (2006) Insect detection of small targets  
moving in visual clutter. PLOS Biol 4: 378–386.

[12]. Barnett PD, Nordstro ̈m K, O’Carroll DC (2007) Retinotopic organization of  
small-field-target-detecting neurons in the insect visual system. Curr Biol 17:  
569–578.

[13] Bolzon, D. M., Nordstr ̈om, K., & O’Carroll, D. C. (2009). Local and large-  
range inhibition in feature detection. Journal of Neuroscience, 29 (45),  
14143–14150.

[14] "Computer Vision: Algorithms and Applications" by Richard Szeliski.

[15] "Background subtraction techniques: A review" by Radke et al.

[16] "Moving Object Detection Using Background Subtraction Techniques and Frame Difference" by Fatima et al.

[17] Nordstr ̈om, K. (2012). Neural specializations for small target detection in  
insects. Current Opinion in Neurobiology, 22 (2), 272–278.

[18] B. P. ̈Olveczky, S. A. Baccus, and M. Meister, “Segregation of object  
and background motion in the retina,” Nature, vol. 423, no. 6938, pp.  
401–408, 2003.

[19] M. V. Srinivasan, S. B. Laughlin, and A. Dubs, “Predictive coding: a  
fresh view of inhibition in the retina,” Proceedings of the Royal Society  
of London B: Biological Sciences, vol. 216, no. 1205, pp. 427–459,  
1982.

[20] J. Van Hateren, “Theoretical predictions of spatiotemporal receptive  
fields of fly lmcs, and experimental validation,” Journal of Comparative  
Physiology A, vol. 171, no. 2, pp. 157–170, 1992.

[21] K. J. Halupka, S. D. Wiederman, B. S. Cazzolato, and D. C. O’Carroll,  
“Discrete implementation of biologically inspired image processing for  
target detection,” in Intelligent Sensors, Sensor Networks and Informa-  
tion Processing (ISSNIP), 2011 Seventh International Conference on.  
IEEE, 2011, pp. 143–148.

[22]. Bagheri, Z. M., Cazzolato, B. S., Grainger, S., OʼCarroll, D. C., & Wiederman, S. D. (2017). An autonomous robot inspired by insect neurophysiology pursues moving features in natural environments. Journal of Neural Engineering, 14(4), 046030.

[23]. Bagheri, Z. M., Wiederman, S. D., Cazzolato, B. S., Grainger, S., & OʼCarroll, D. C. (2015). Properties of neuronal facilitation that improve target tracking in natural pursuit simulations. Journal of the Royal Society Interface, 12(108), 20150083.

[24]. Bagheri, Z. M., Wiederman, S. D., Cazzolato, B. S., Grainger, S., & OʼCarroll, D. C. (2017). Performance of an insect-inspired target tracker in natural conditions. Bioinspiration & Biomimetics, 12(2), 025006.

[25] [To download atom editor.](https://atom.en.uptodown.com/windows)

[26] PowerShell is available by default on Windows; it is also possible to use Terminal, which is available on Mac and Unix systems.

[27] Gaussian function 2D source code: <https://github.com/boukary-derra/pfe/blob/main/code/gauss/plot_gauss_1d.py>

[28] Gaussian function 3D source code : <https://github.com/boukary-derra/pfe/blob/main/code/gauss/plot_gauss_2d.py>

[29] <https://github.com/boukary-derra/pfe/tree/main/code>

[30] <https://github.com/boukary-derra/pfe/tree/main/code>

[31] Python: <https://www.python.org/>

[32] OpenCv: <https://opencv.org/>

[33] Scipy: <https://scipy.org/>

[34] Matplotlib: <https://matplotlib.org/>

[35] Numpy: <https://numpy.org/>

# Annexe

## A1. Faneback pour la détection de mouvement par flux optique



Dans ce code, nous lisons d'abord la vidéo frame par frame. Ensuite, nous convertissons chaque image en niveaux de gris pour effectuer la détection de mouvement.

Le flux optique est calculé entre l'image courante et l'image précédente en utilisant la fonction calcOpticalFlowFarneback() de OpenCV. Cette fonction renvoie un champ de vecteurs qui représente le mouvement de chaque pixel d'une image à l'autre.

Ensuite, nous convertissons les vecteurs de mouvement en magnitude et en angle en utilisant la fonction cartToPolar(). Cela nous permet de colorer l'image en fonction de la direction du mouvement : l'angle du mouvement est utilisé pour la teinte et la magnitude du mouvement est utilisée pour la valeur (luminosité).

Enfin, nous affichons l'image colorée et attendons que l'utilisateur appuie sur la touche 'esc' pour quitter.

Il est important de noter que la détection de mouvement par flux optique peut être sensible au bruit, aux variations d'éclairage et aux changements de scène. De plus, elle peut ne pas fonctionner correctement lorsque le mouvement est trop rapide ou lorsque la texture de l'image est trop uniforme.

## A2. MOG2 pour la détection de mouvement par soustraction de fond

La méthode MOG2 (Mixture Of Gaussians) est une technique de soustraction de fond qui est utilisée pour segmenter les régions d'intérêt en décomposant chaque pixel de l'image en une combinaison de distributions Gaussiennes.

Voici un exemple basique de code en Python utilisant OpenCV pour détecter le mouvement en utilisant la soustraction de fond. Notez qu'OpenCV offre plusieurs méthodes pour la soustraction de fond, mais ici nous allons utiliser la méthode MOG2.

MOG2 est un algorithme amélioré par rapport à l'approche originale MOG. MOG2 a la capacité de mieux gérer les changements d'éclairage, et il a également la capacité de détecter et de neutraliser les ombres, ce qui peut améliorer les performances de la segmentation du premier plan.



Dans cette méthode, chaque pixel dans l'image est représenté comme un mélange de distributions Gaussiennes. L'algorithme apprend et crée un nouveau modèle de fond d'une image en identifiant quelles distributions de Gauss sont associées à l'arrière-plan. Ensuite, lorsque de nouvelles images arrivent, le modèle est comparé à ces nouvelles images et les pixels qui ne s'adaptent pas bien aux distributions de Gauss de l'arrière-plan sont considérés comme faisant partie du premier plan.

L'implémentation MOG2 dans OpenCV a plusieurs paramètres que vous pouvez ajuster pour l'adapter à vos besoins spécifiques. Par exemple, vous pouvez contrôler le nombre de mélanges de Gauss utilisés, le seuil de classification de l'arrière-plan et du premier plan, et si l'algorithme doit tenter de détecter les ombres ou non.

La méthode MOG2 est particulièrement utile pour les tâches de suivi d'objets et de surveillance vidéo, où vous avez besoin de segmenter les objets en mouvement du fond statique.

## A3. Détection de mouvement par différenciation de cadre

La détection de mouvement par différenciation de cadre consiste à comparer deux images consécutives et à déterminer les différences entre elles. Cela peut être fait en utilisant des méthodes de seuillage ou en calculant la différence absolue entre les deux images. Voici un exemple de code Python qui utilise OpenCV pour détecter le mouvement en utilisant la différenciation de cadre :

Dans le code ci-dessous, nous commençons par lire deux images consécutives à partir d'une vidéo. Nous calculons ensuite la différence absolue entre ces deux images, que nous convertissons en niveaux de gris et à laquelle nous appliquons un flou gaussien pour améliorer la détection de contour.

Ensuite, nous appliquons un seuil à l'image pour obtenir une image binaire, que nous dilatons pour combler les trous dans les objets en mouvement. Nous trouvons ensuite les contours dans cette image dilatée.

Pour chaque contour trouvé, nous calculons un rectangle englobant et nous le dessinons sur l'image originale. Enfin, nous affichons l'image avec les rectangles dessinés.

Le processus se répète avec les images suivantes de la vidéo jusqu'à ce que l'utilisateur appuie sur la touche 'q' pour arrêter.



## A4. Détection de mouvement par différenciation temporel



La détection de mouvement par différenciation temporelle implique de comparer le cadre actuel avec le cadre précédent. Si un objet a bougé, les différences de pixels entre ces deux images seront significatives. Le code ci-dessus est un exemple qui peut être utilisé pour accomplir cela avec OpenCV.

Ici, nous capturons deux images (frame1 et frame2) à partir de la vidéo Dans la boucle while, nous calculons la différence absolue entre ces deux images pour obtenir l'image diff. Ensuite, nous convertissons cette image en niveau de gris et nous l'adoucissons avec un flou gaussien. Un seuil est appliqué à cette image pour binariser les pixels qui ont une intensité supérieure à 20. L'image est ensuite dilatée pour éliminer le bruit restant.

Ensuite, nous utilisons la fonction findContours pour obtenir les contours de toutes les zones en mouvement. Nous dessinons un rectangle autour de ces contours sur l'image d'origine (frame1). Enfin, nous mettons à jour frame1 et frame2 pour la prochaine itération de la boucle. Le processus est répété pour chaque paire d'images consécutives dans la vidéo.

## A5. Fonction Gaussienne

**Présentation :**

La fonction gaussienne est une fonction mathématique largement utilisée dans divers domaines de la science, notamment la physique, l'ingénierie et les mathématiques. Elle est nommée d'après le mathématicien Carl Friedrich Gauss, qui l'a introduite pour la première fois au début du XIXe siècle.

La fonction est définie comme suit :



Où :

* *x* est la variable indépendante
* *A* est l'amplitude de la courbe
* *μ* est la moyenne ou le centre de la courbe
* σ est l'écart-type de la courbe (ou la largeur)

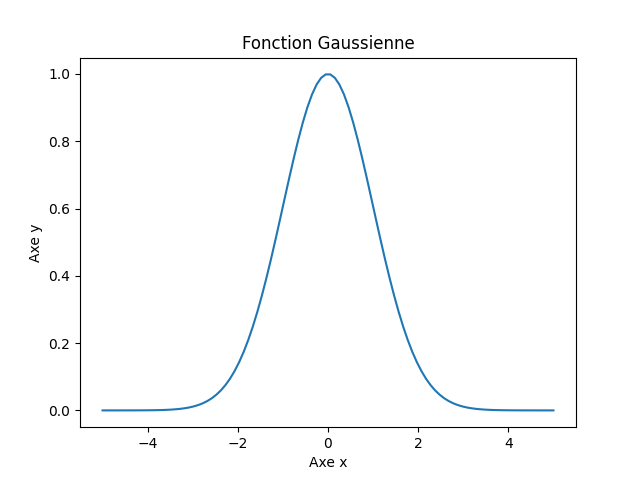


Figure 12: Fonction Gaussienne

La courbe de la fonction gaussienne est en forme de cloche qui est symétrique par rapport à sa moyenne. Elle est caractérisée par son amplitude, sa moyenne et son écart-type.

L'amplitude est la valeur maximale de la fonction, atteinte à la moyenne. L'écart-type détermine la largeur de la courbe.

La fonction gaussienne possède plusieurs propriétés importantes qui la rendent utile dans diverses applications. L'une de ses propriétés les plus importantes est qu'elle est une solution de l'équation de diffusion. Cela signifie qu'elle décrit la diffusion d'une quantité, telle que la température, dans le temps et l'espace. Cette propriété la rend utile pour modéliser divers phénomènes physiques, tels que la diffusion de chaleur et le rayonnement électromagnétique.

La fonction gaussienne est également largement utilisée en théorie des probabilités et en statistique. Dans ces domaines, elle est souvent utilisée pour modéliser la distribution de variables aléatoires. Par exemple, la distribution normale, qui est une distribution de probabilité largement utilisée en statistique, est un cas particulier de la fonction gaussienne.

En plus de ses applications en physique, en ingénierie et en mathématiques, la fonction gaussienne est également utilisée dans divers autres domaines, tels que le traitement d'images, le traitement de signaux et l'apprentissage automatique. Dans ces domaines, elle est souvent utilisée pour lisser ou filtrer les données, modéliser le bruit ou l'erreur ou estimer les paramètres d'un modèle.

**La fonction gaussienne bidimensionnelle**

La fonction gaussienne bidimensionnelle (c’est-à-dire avec deux variables x et y) ou encore fonction gaussienne à deux dimensions. Elle est généralement utilisée en traitement d'images, en traitement de signal, en vision par ordinateur et en apprentissage automatique.

La forme générale de la fonction gaussienne bidimensionnelle est la suivante :



Où :

* A est l'amplitude de la courbe
* x0 et y0 sont les moyennes ou centres de la courbe selon x et y respectivement
* σx et σy sont les écart-types de la courbe selon x et y respectivement

Pour la suite nous allons supposer que la courbe de la fonction est centrée à 0 selon x tout comme selon y ; donc  et  .

Et que les écart-types de la courbe selon x et y sont les mêmes ; donc.

La fonction Gaussienne vaut maintenant :



Notre objectif est d’utiliser cette fonction pour appliquer des effets de flou (**blur**) sur des images.

Cette opération de flou (**blurring**) a pour objectif :

* **Réduction du bruit** : Le floutage peut aider à lisser et à réduire l'impact du bruit aléatoire ou des petites variations dans les valeurs d'intensité des pixels.
* Extraction de caractéristiques : En floutant, les détails fins dans une image peuvent être moyennés, ce qui facilite la détection de caractéristiques plus importantes et significatives dans l'image.
* Redimensionnement d'image : Le floutage Gaussien peut être utilisé comme une méthode de redimensionnement d'image, où l'image est redimensionnée à une taille plus petite puis agrandie, produisant un résultat plus lisse et plus attrayant visuellement.

Pour l’amplitude A de la courbe, on prendra : 

En effet, l'amplitude de la courbe dans la fonction gaussienne bidimensionnelle est souvent définie de manière à normaliser la courbe et à faciliter sa comparaison avec d'autres courbes. L'amplitude est choisie de manière à ce que l'intégrale sur toute la surface de la courbe soit égale à 1 ( ). La justification mathématique de cette amplitude peut être trouvée en utilisant **le théorème de Fubini**.

Soit 

En substituant  par expressions dans cette équation, on obtient :



Ensuite, nous intégrons cette fonction g(x) par rapport à x de moins l'infini a plus l'infini :



En égalant cette expression à 1 et en résolvant pour A, on obtient :



Cela démontre que si nous choisissons l'amplitude de la fonction gaussienne à être, l'intégrale sur toute la surface de la courbe est égale à 1.

Finalement, on obtient :



**Forme discrète : Le noyau gaussien**

Un noyau gaussien est une matrice de pixels de taille déterminée par l'écart type () de la fonction gaussienne, qui est utilisé pour pondérer les pixels voisins de l'image lors de la convolution. Le noyau gaussien est une représentation discrète de la fonction gaussienne multidimensionnelle, qui est utilisée pour modéliser la distribution de probabilité de certaines caractéristiques visuelles dans le traitement d'images. Le noyau gaussien est souvent utilisé dans le domaine de la vision par ordinateur pour réduire le bruit d'une image, améliorer sa qualité ou encore pour effectuer une opération de flou.

Le noyau gaussien est une matrice carrée de taille (2k+1) x (2k+1) où k est un nombre entier positif. Les coefficients de cette matrice sont calculés à partir de la fonction gaussienne. Plus précisément, le coefficient situé à la position  du noyau gaussien est donné par :



Où () est l'écart type de la distribution gaussienne.

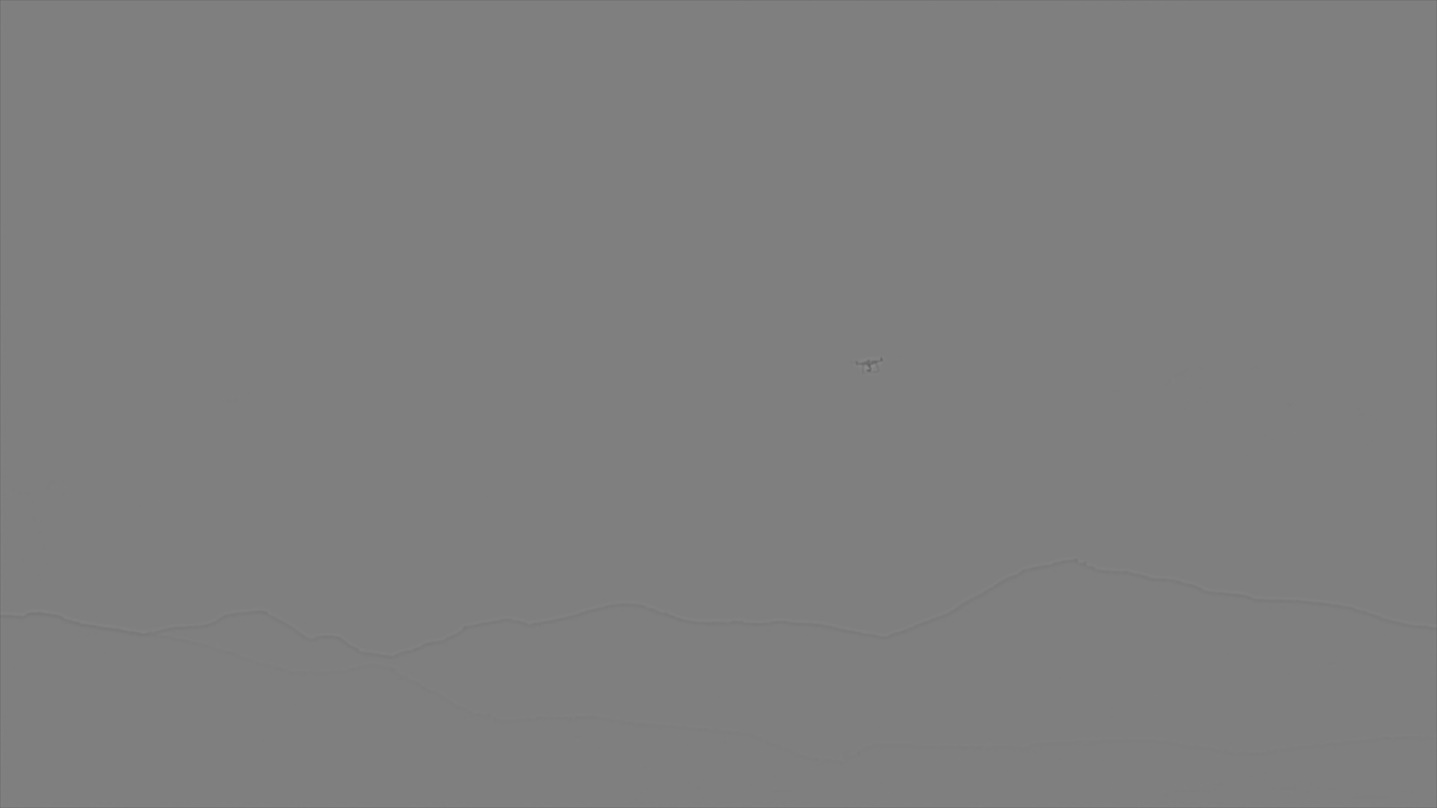
La taille du noyau gaussien est généralement choisie en fonction de l'écart type sigma. Plus l'écart type est grand, plus la courbe gaussienne est plate et plus il faut une grande taille de noyau pour capturer toute la courbe. À l'inverse, si l'écart type est petit, la courbe est plus pointue et une petite taille de noyau peut suffire.

L'utilisation du noyau gaussien dans le filtrage d'images est basée sur le fait que la convolution d'une image avec un noyau gaussien permet de flouter l'image tout en préservant les contours et les détails importants. Cela est possible car la fonction gaussienne est une fonction de densité de probabilité qui répartit uniformément les valeurs autour de la moyenne. Ainsi, en utilisant un noyau gaussien, on donne plus de poids aux pixels voisins de l'image, tout en réduisant l'importance des pixels plus éloignés.

## A6. Résultats des différents couches du frame V1-47



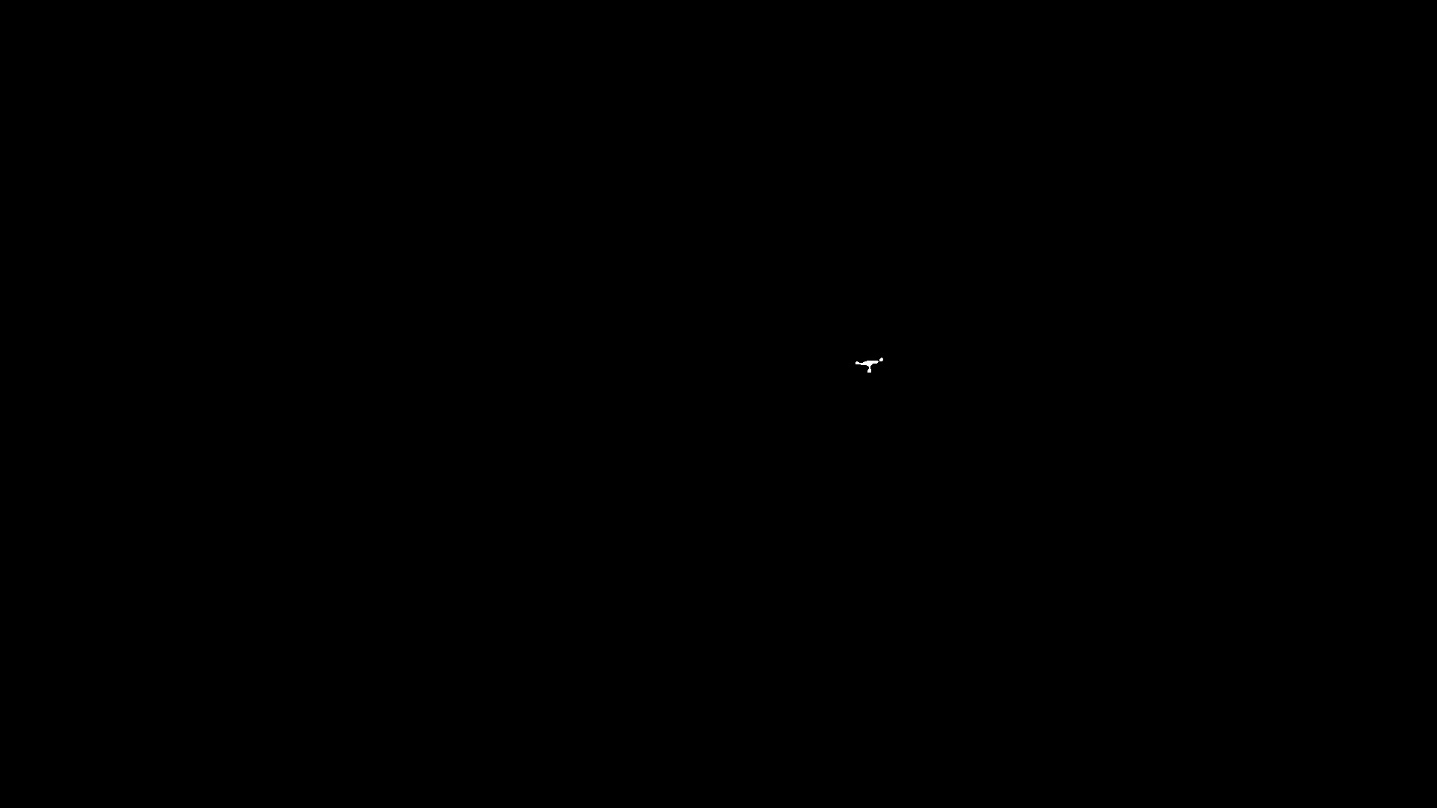
Figure : Sortie photorécepteur de V1-47



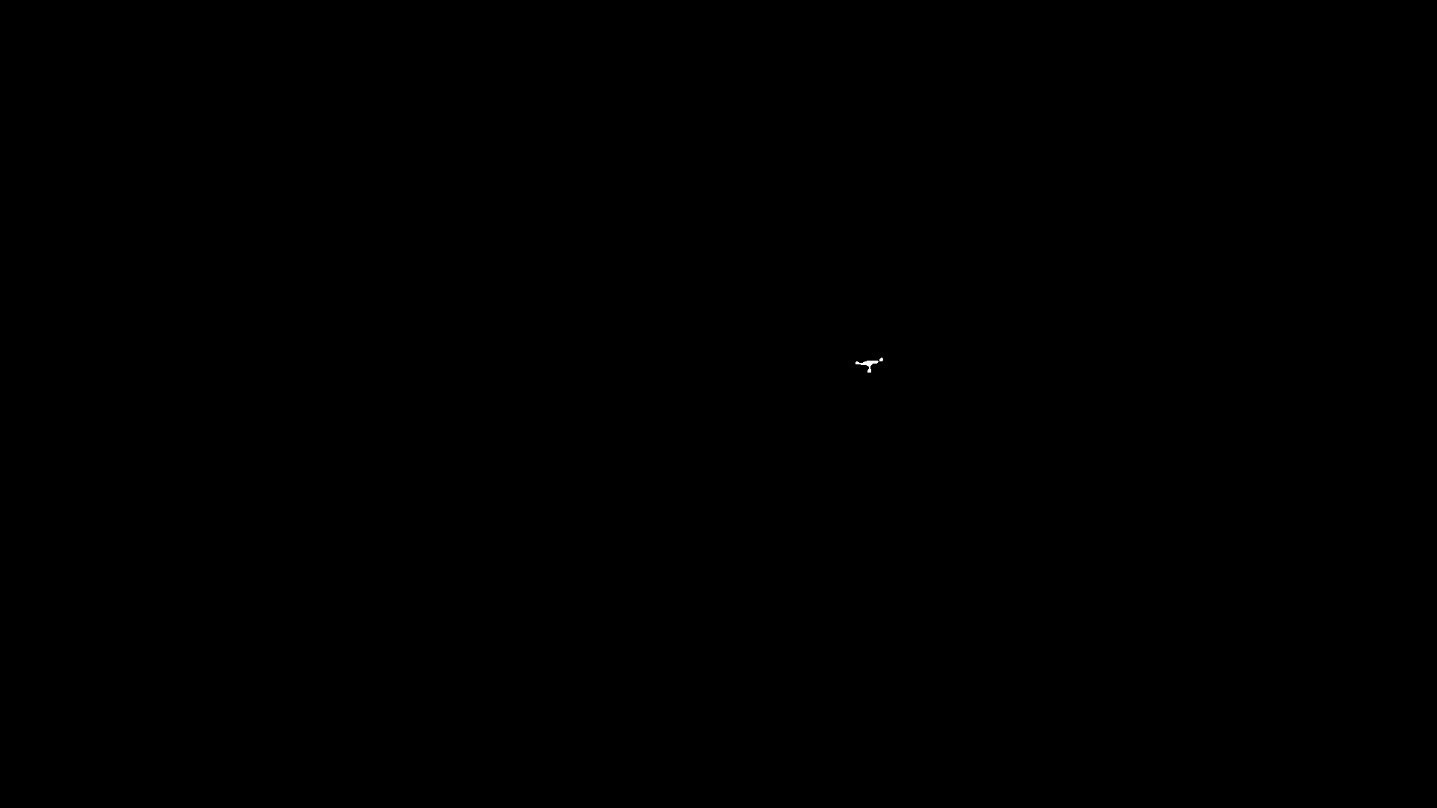
*Figure 14: Sortie lipetz transformation de V1-47*



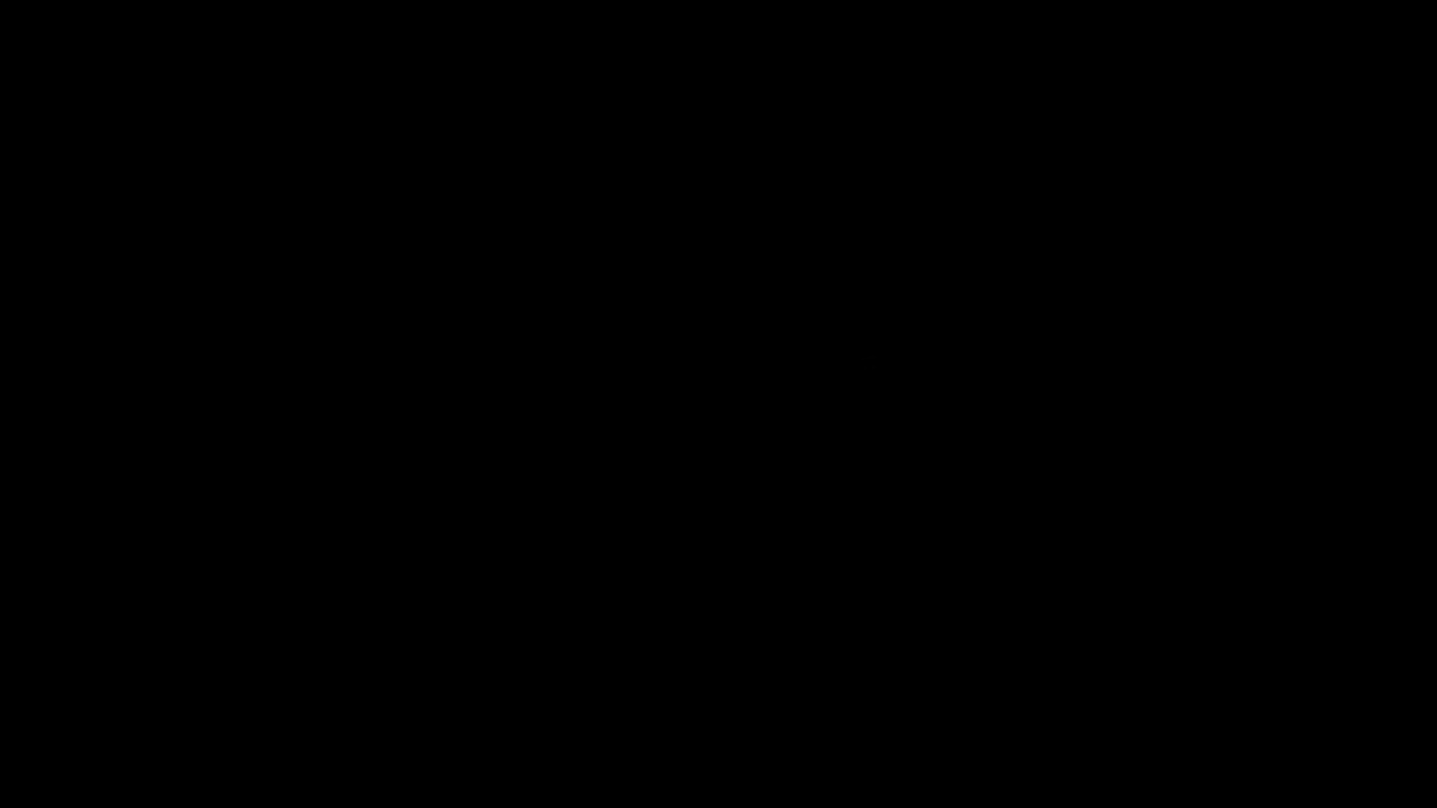
*Figure 15: Sortie low pass filter de V1-47*



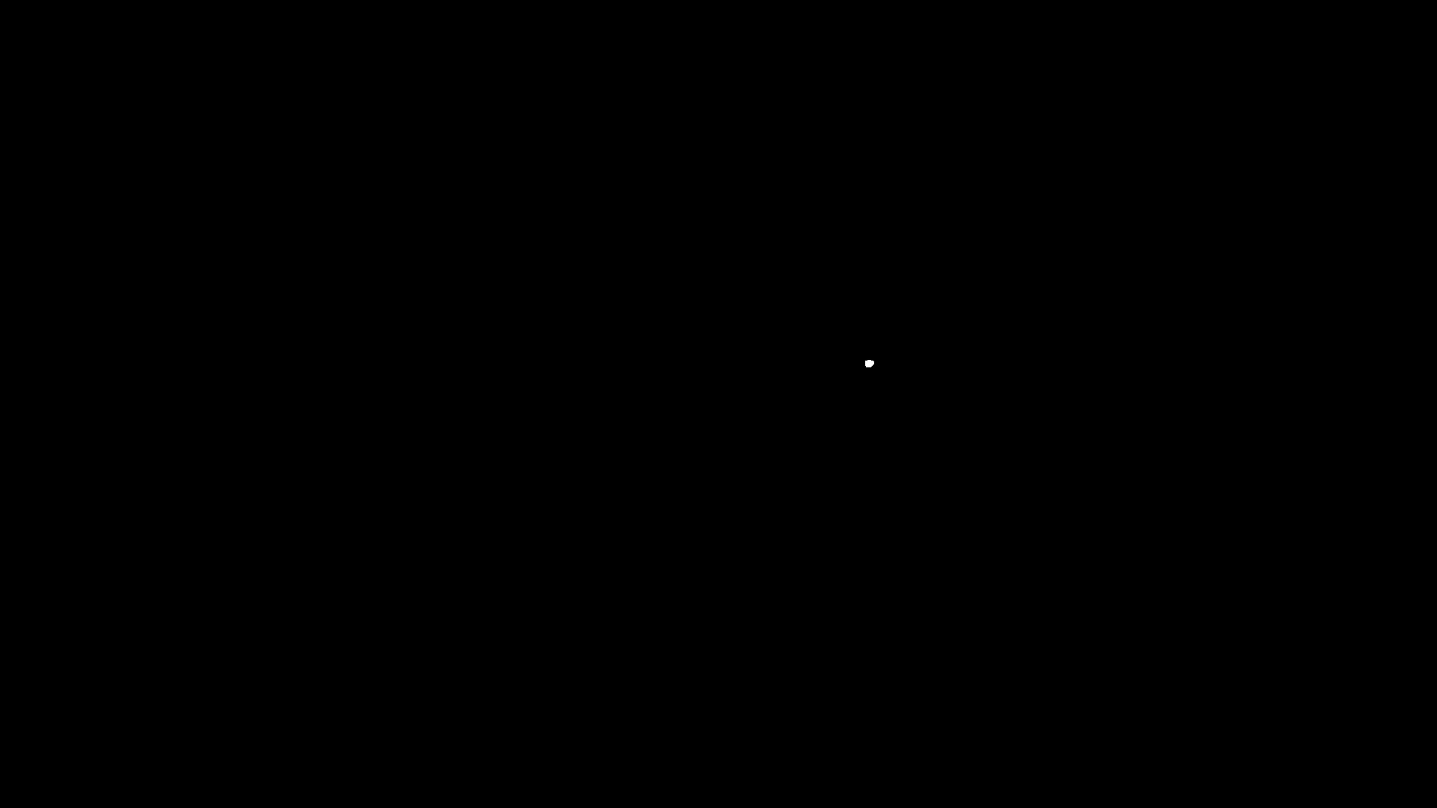
*Figure 16: Sortie LMC de V1-47*



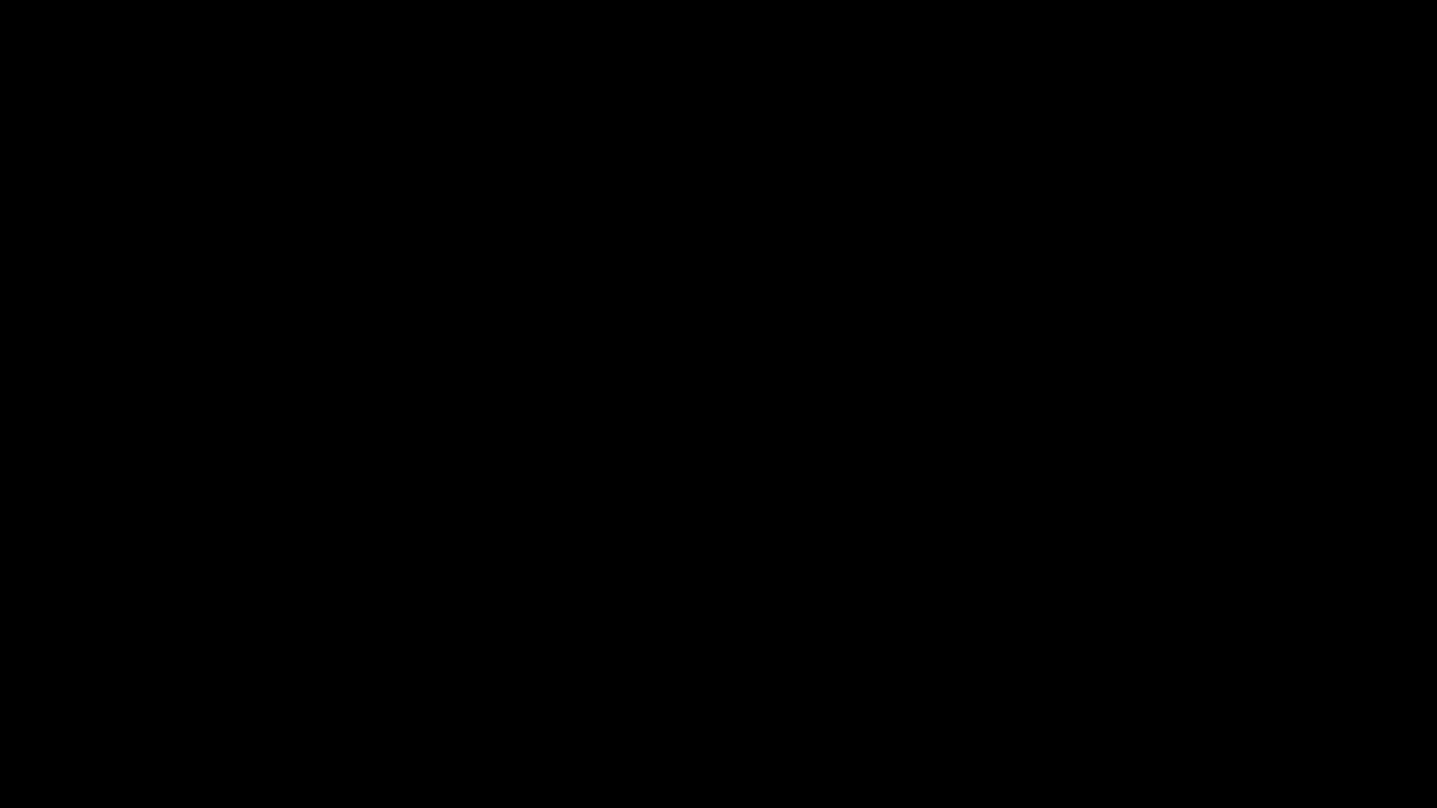
*Figure 17: LMC en OFF channel de V1-47*



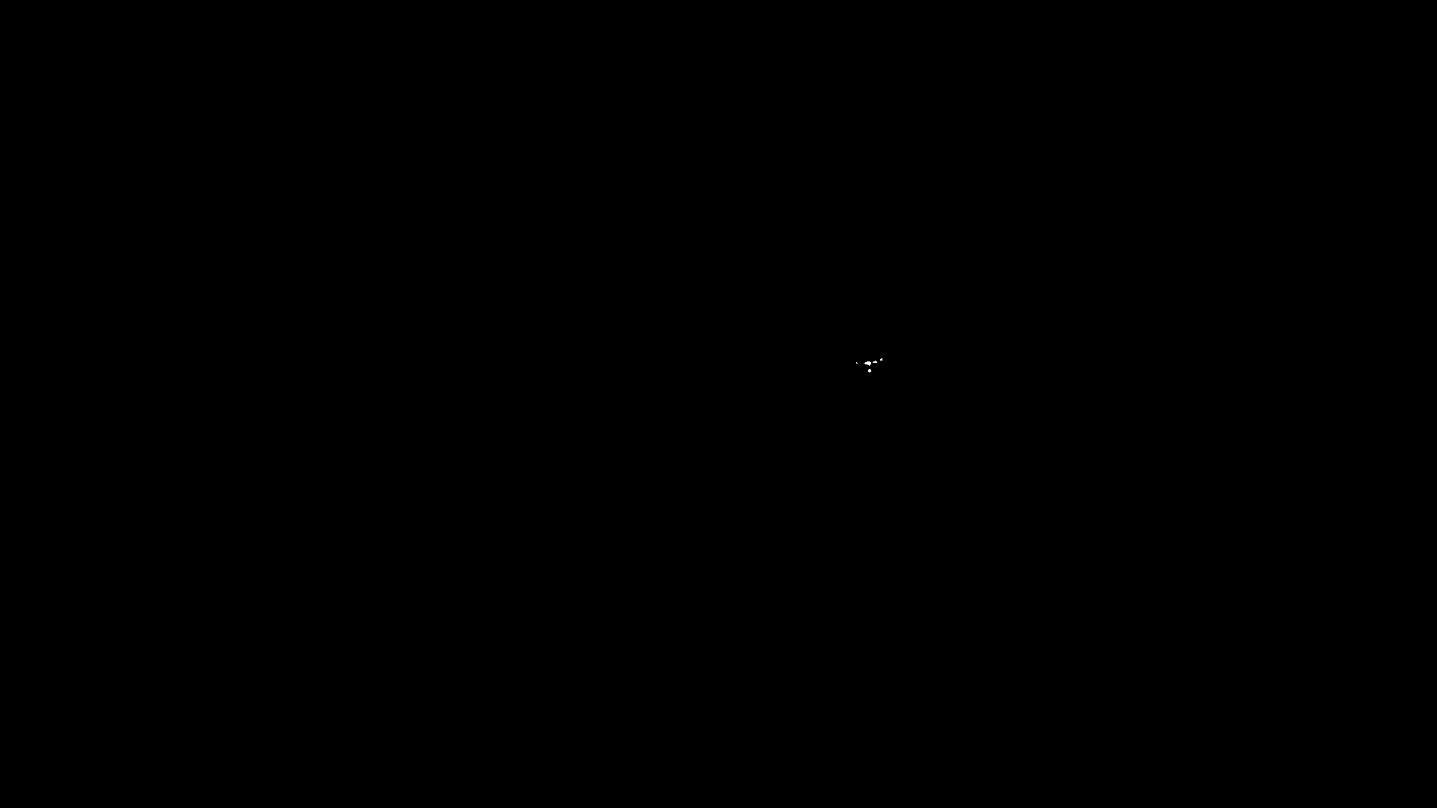
*Figure 18: LMC en ON channel de V1-47*

**

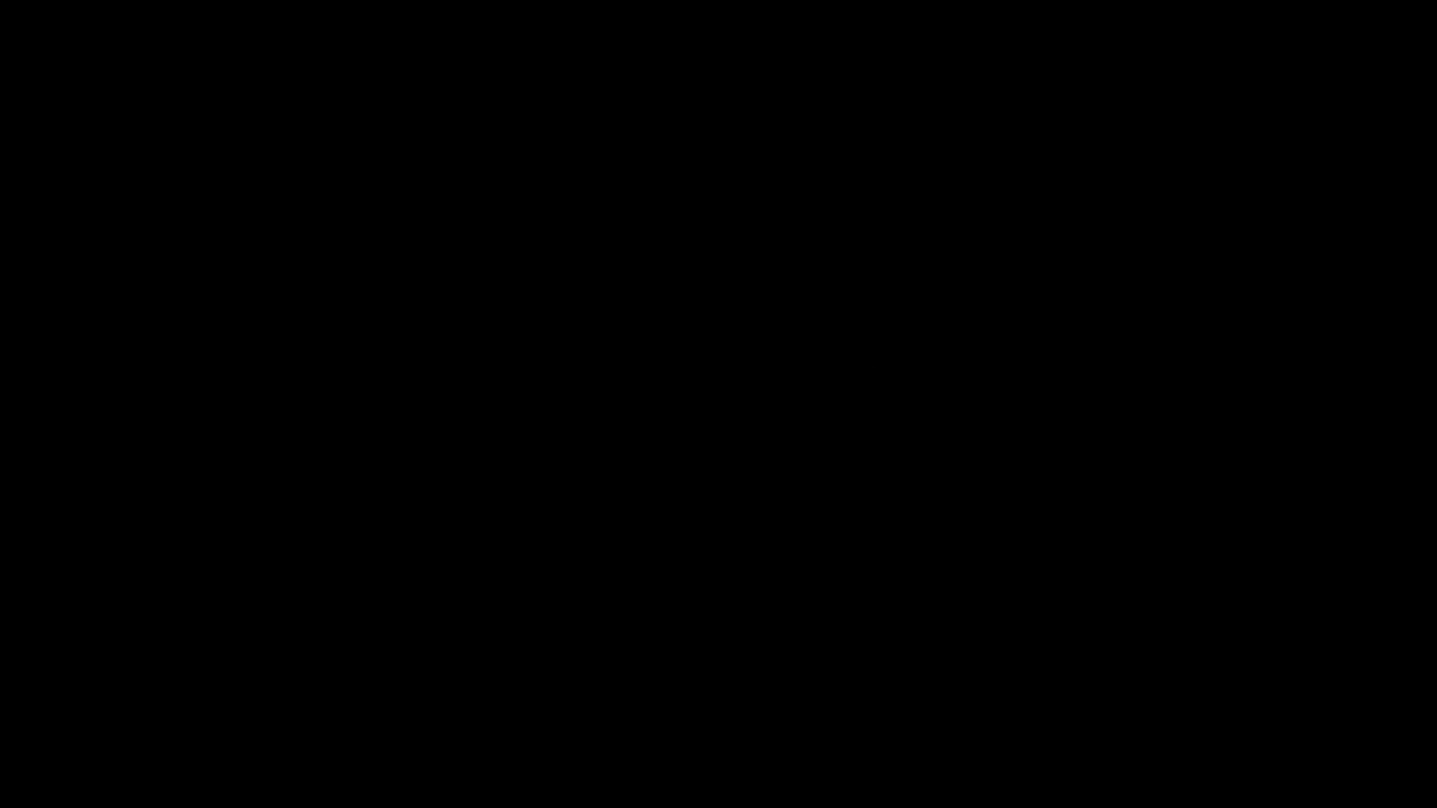
*Figure 19: Sortie OFF FDSR de V1-17*



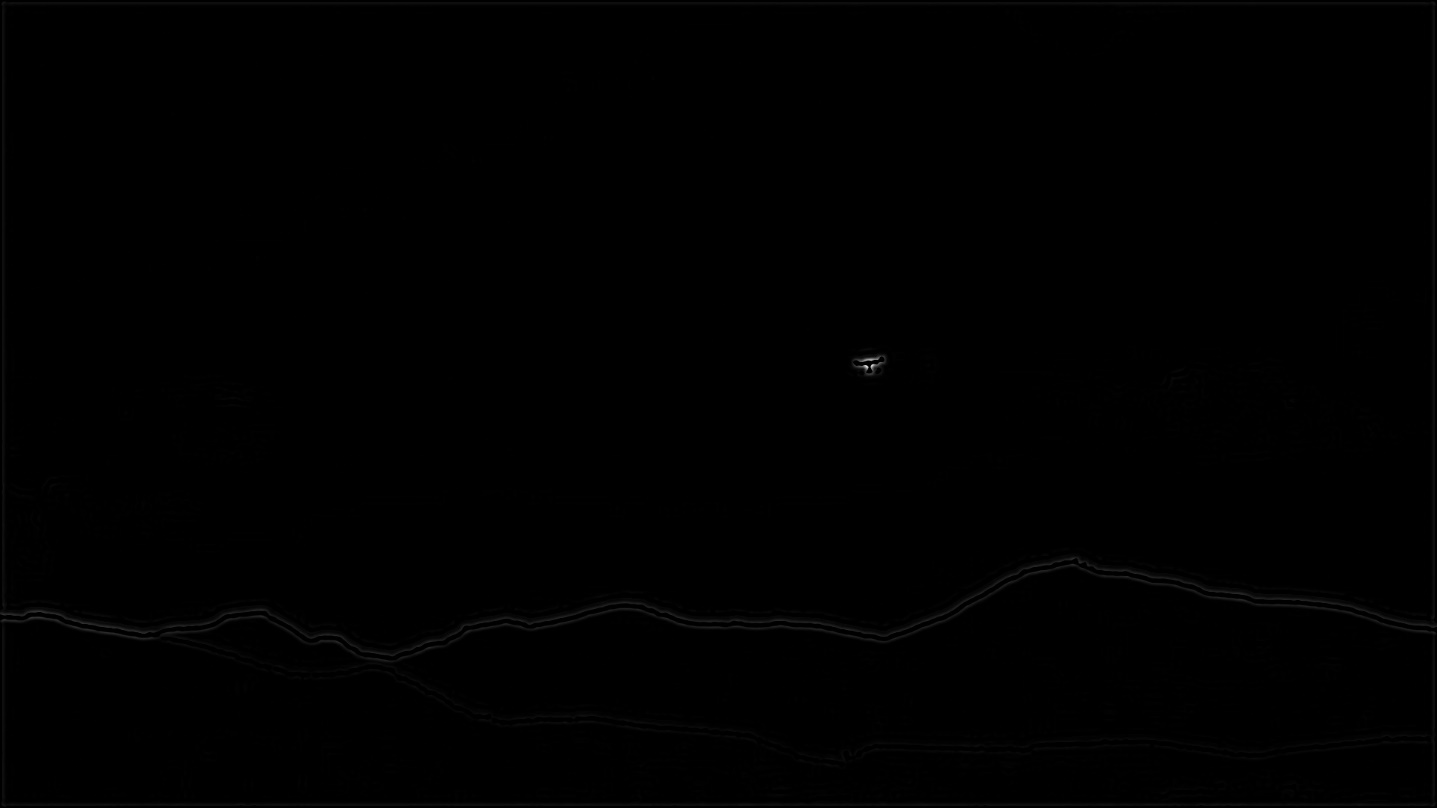
*Figure 20: Sortie ON FDSR de V1-47*



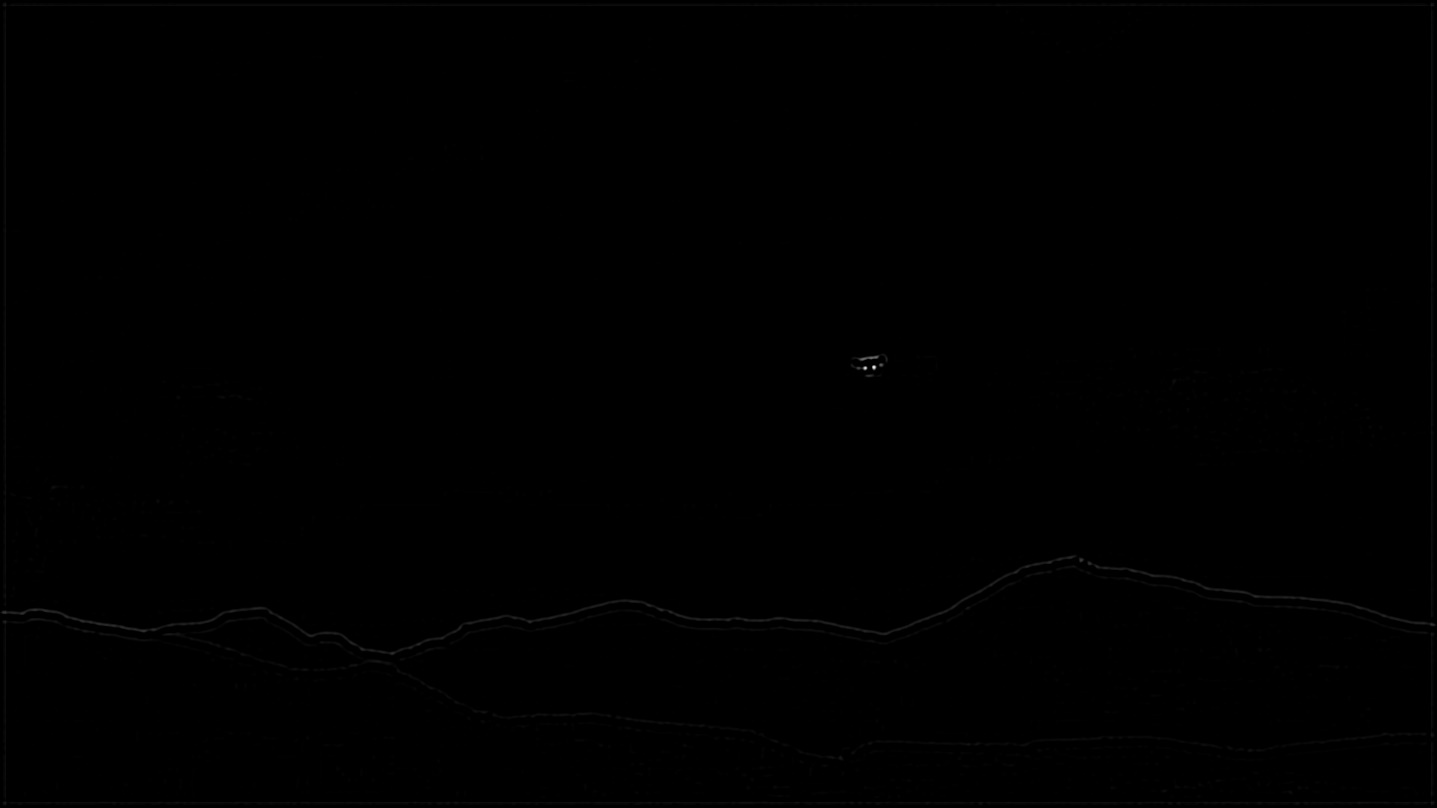
*Figure 21: Sortie OFF SIGMA de V1-47*



*Figure 22: Sortie ON SIGMA V1-47*



*Figure 23: Sotie OFF HWR de V1-47*



*Figure 24: Sortie ON HWR de V1-47*

## A7. Paquages et Installations

### A7.1. Python

Python est un langage de programmation polyvalent et convivial, largement utilisé dans le domaine du traitement d'images et de la vision par ordinateur. Sa syntaxe simple et expressive en fait un choix idéal pour développer des algorithmes de détection du mouvement de petites cibles. Dans mon projet, j'ai utilisé Python pour écrire le code principal, gérer les flux d'images, manipuler les données et interagir avec les autres bibliothèques. Python dispose également d'une vaste communauté de développeurs qui fournissent des ressources et des packages supplémentaires, ce qui facilite la mise en œuvre de solutions de traitement d'images avancées. Pour installer Python, vous pouvez télécharger la dernière version depuis le site officiel de Python (<https://www.python.org/>) et suivre les instructions d'installation.



### A7.2. OpenCV

OpenCV, abréviation de « Open Source Computer Vision », est une bibliothèque populaire et puissante utilisée pour le traitement d'images et la vision par ordinateur. Elle offre un large éventail de fonctionnalités pour manipuler et analyser des images, y compris la détection du mouvement. Dans mon projet, j'ai utilisé la bibliothèque OpenCV (cv2) pour capturer les images en temps réel, extraire les contours et effectuer l'analyse du mouvement des petites cibles. OpenCV m'a offert des outils robustes pour traiter les images, notamment pour l'extraction de caractéristiques et la segmentation. Pour installer OpenCV, vous pouvez utiliser la commande pip install opencv-python dans votre environnement Python.



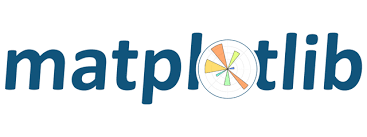
### A7.3. Scipy

SciPy est une bibliothèque puissante et polyvalente utilisée pour le calcul scientifique et l'analyse de données en Python. Elle fournit des fonctionnalités avancées pour l'optimisation, l'algèbre linéaire, l'intégration, la statistique et bien plus encore. Dans mon projet, j'ai utilisé SciPy pour effectuer des opérations mathématiques complexes, telles que la transformation de Fourier discrète (DFT), qui est utile pour analyser les signaux de mouvement dans les images. SciPy m'a également permis d'exploiter des fonctionnalités statistiques pour analyser les données de mouvement. Pour installer SciPy, vous pouvez utiliser la commande pip install scipy dans votre environnement Python.



### A7.4. Matplotlib

Matplotlib est une bibliothèque de visualisation de données en Python. Elle permet de créer des graphiques, des diagrammes, des histogrammes et des visualisations personnalisées pour analyser et présenter les résultats. Dans mon projet, j'ai utilisé Matplotlib pour afficher les images capturées en temps réel avec les contours détectés et les zones de mouvement identifiées. Cette visualisation m'a aidé à comprendre et à valider les résultats de détection du mouvement. Pour installer Matplotlib, vous pouvez utiliser la commande pip install matplotlib dans votre environnement Python.



### A7.5. Numpy

NumPy est une bibliothèque fondamentale pour le calcul scientifique en Python. Elle offre des structures de données efficaces pour la manipulation de tableaux multidimensionnels, ainsi que des fonctions mathématiques de haut niveau pour effectuer des opérations sur ces tableaux. Dans mon projet, j'ai utilisé NumPy pour manipuler les images capturées, extraire des régions d'intérêt, effectuer des opérations matricielles et analyser les données de mouvement. Les performances optimisées de NumPy permettent d'accélérer les calculs et de gérer efficacement les grands ensembles de données. Pour installer NumPy, vous pouvez utiliser la commande pip install numpy dans votre environnement Python. Assurez-vous d'avoir Python préalablement installé avant d'installer NumPy.



### A7.6. ATOM

ATOM est un éditeur de code open-source, léger et personnalisable, qui offre une expérience de développement agréable pour les programmeurs Python. J'ai utilisé ATOM pour écrire et organiser mon code Python de détection du mouvement. L'éditeur offre des fonctionnalités pratiques telles que la coloration syntaxique, la numérotation des lignes, le pliage du code et l'intégration avec Git. Pour installer ATOM, vous pouvez visiter le site officiel d'ATOM (<https://atom.io/>) et télécharger le package d'installation correspondant à votre système d'exploitation.



### A7.7. Windows PowerShell

Windows PowerShell est un shell de ligne de commande et un langage de script développé par Microsoft, intégré aux systèmes d'exploitation Windows modernes. J'ai utilisé Windows PowerShell pour exécuter mes scripts Python et effectuer des tâches de gestion des fichiers, de navigation dans le système de fichiers et d'automatisation des tâches. PowerShell offre une grande flexibilité et une grande puissance pour interagir avec le système d'exploitation Windows. Il est préinstallé sur les systèmes Windows récents, donc aucune installation supplémentaire n'est nécessaire.



### A7.8. Environnement virtuel (venv )

Un environnement virtuel est un outil essentiel pour isoler et gérer les dépendances spécifiques d'un projet Python. Il permet de créer un environnement de développement autonome où vous pouvez installer des packages et des bibliothèques sans interférer avec d'autres projets Python sur votre système. Dans mon projet de traitement d'images, j'ai utilisé un environnement virtuel (venv) pour maintenir une configuration propre et spécifique à mon projet. Cela m'a permis de gérer les dépendances requises, d'installer des bibliothèques tierces comme OpenCV, NumPy et autres, sans risquer de conflits avec d'autres projets ou systèmes. Pour créer un environnement virtuel avec venv, vous pouvez exécuter les commandes suivantes dans votre invite de commandes ou votre terminal : python -m venv mon\_env.

Cela créera un nouvel environnement virtuel nommé mon\_env dans le répertoire actuel. Ensuite, pour activer l'environnement virtuel, vous devez exécuter la commande correspondante en fonction de votre système d'exploitation :

* Sur Windows : mon\_env\Scripts\activate
* Sur Linux/MacOS : source mon\_env/bin/activate

Une fois l'environnement virtuel activé, vous pouvez installer les packages spécifiques à votre projet en utilisant la commande pip, sans affecter votre système global.

**Avantages de l'environnement virtuel (venv) :**

L'utilisation d'un environnement virtuel présente plusieurs avantages. Tout d'abord, il vous permet de maintenir un environnement de développement propre et bien organisé, en isolant les dépendances de chaque projet. Cela facilite la gestion des versions des packages et garantit que votre projet utilise les versions spécifiques dont il a besoin. De plus, l'environnement virtuel offre une portabilité, vous permettant de partager facilement votre projet avec d'autres développeurs sans vous soucier des conflits de dépendances. Enfin, l'utilisation de venv facilite également le processus de déploiement en fournissant un moyen reproductible de créer et de gérer l'environnement de production de votre projet.

**Installation de venv :**

L'environnement virtuel (venv) est une fonctionnalité intégrée à Python à partir de la version 3.3. Par conséquent, il est généralement inclus dans l'installation standard de Python. Pour créer un environnement virtuel, vous devez exécuter la commande python -m venv nom\_de\_l\_environnement dans votre invite de commandes ou votre terminal, comme indiqué précédemment. Assurez-vous d'avoir Python installé correctement sur votre système avant de créer un environnement virtuel.

### A7.9. Django

**Présentation**

Django est un framework web puissant et populaire écrit en Python. Il offre une approche basée sur le modèle MVC (Modèle-Vue-Contrôleur) pour le développement d'applications web robustes et évolutives. Dans mon projet, j'ai utilisé Django pour créer une plateforme graphique conviviale qui permet aux utilisateurs d'effectuer différents tests de traitement d'images sans avoir à manipuler des commandes en ligne de commande complexes. Django m'a permis de gérer facilement les requêtes HTTP, de structurer mon application en modules réutilisables, de gérer les autorisations utilisateur et de générer des interfaces utilisateur interactives. L'utilisation de Django a considérablement simplifié le processus de développement et a offert une expérience utilisateur améliorée.

Pour installer Django, vous pouvez exécuter la commande suivante dans votre environnement virtuel (venv) : pip install django .

Assurez-vous d'avoir Python et venv préalablement installés sur votre système. Après l'installation, vous pouvez créer un nouveau projet Django en utilisant la commande django-admin startproject nom\_du\_projet. Cela générera une structure de projet initiale avec des fichiers de configuration et des dossiers nécessaires. Ensuite, vous pouvez créer des applications Django à l'intérieur du projet à l'aide de la commande python manage.py startapp nom\_de\_l\_application. Ces applications peuvent contenir des modèles de données, des vues et des templates qui seront utilisés pour construire votre plateforme graphique.

**Simplification des manipulations en ligne de commande :**

L'utilisation de Django pour créer une plateforme graphique a permis de simplifier les manipulations en ligne de commande souvent nécessaires lors du traitement d'images. Plutôt que de demander aux utilisateurs d'exécuter des commandes complexes et potentiellement déroutantes, l'interface graphique fournie par Django permet de présenter les fonctionnalités et les options de manière claire et intuitive. Les utilisateurs peuvent simplement interagir avec l'interface utilisateur pour charger les images, sélectionner les tests à effectuer, ajuster les paramètres et afficher les résultats. L'utilisation de Django a donc considérablement réduit la courbe d'apprentissage pour les utilisateurs et a facilité l'exploration et l'utilisation des fonctionnalités de traitement d'images de mon projet.

Pour utiliser votre plateforme graphique basée sur Django, vous devez exécuter le serveur de développement Django à l'aide de la commande python **manage.py runserver**. Cela lancera le serveur local qui permettra aux utilisateurs d'accéder à votre plateforme via un navigateur web. Vous pouvez ensuite personnaliser les vues, les templates et les fonctionnalités de votre application Django pour répondre aux besoins spécifiques de votre projet.

N'oubliez pas que pour déployer votre plateforme graphique Django dans un environnement de production, vous devrez prendre en compte des considérations supplémentaires, telles que la configuration du serveur web, la gestion des bases de données, la sécurité, etc.



### A7.10. Git et GitHub

**Git :**

Git est un système de contrôle de version décentralisé, largement utilisé dans le développement de logiciels pour gérer les modifications du code source. Il permet de suivre l'historique des modifications, de collaborer avec d'autres développeurs, de gérer les branches de développement et de fusionner les modifications de manière efficace. Dans mon projet de traitement d'images, j'ai utilisé Git pour suivre et enregistrer les différentes versions de mon code, ce qui m'a permis de revenir à des versions antérieures en cas de besoin et de travailler en collaboration avec mon encadrant. Pour installer Git, vous pouvez visiter le site officiel de Git (<https://git-scm.com/>) et télécharger le package d'installation correspondant à votre système d'exploitation. Suivez les instructions d'installation fournies pour configurer Git sur votre machine. Après l'installation, vous pouvez vérifier si Git est correctement installé en exécutant la commande git --version dans votre invite de commandes ou votre terminal. Assurez-vous d'avoir une connexion Internet active pour télécharger les fichiers d'installation et configurer Git correctement.

**GitHub :**

GitHub est une plateforme d'hébergement de code source basée sur Git. Elle offre des fonctionnalités supplémentaires telles que le suivi des problèmes, la gestion des projets et la collaboration en équipe. Dans mon projet, j'ai utilisé GitHub pour héberger mon code et le rendre accessible à d'autres développeurs. Cela m'a permis de partager facilement mon travail, de recevoir des commentaires et des contributions, et de synchroniser les mises à jour entre différents environnements de développement. Pour utiliser GitHub, vous devez créer un compte sur le site officiel de GitHub (<https://github.com/>) et configurer un dépôt pour votre projet. Vous pouvez ensuite utiliser des commandes Git telles que **git clone**, **git commit** et **git push** pour interagir avec votre dépôt distant et gérer les versions de votre code.

**Avantages de Git et GitHub :**

L'utilisation de Git et GitHub présente de nombreux avantages pour la gestion de projet et la collaboration. Tout d'abord, Git permet de garder une trace précise des modifications apportées au code, ce qui facilite la gestion des versions et le suivi des erreurs. De plus, Git facilite la collaboration en permettant à plusieurs développeurs de travailler simultanément sur différentes branches de développement et de fusionner les modifications de manière contrôlée. GitHub ajoute une couche supplémentaire de fonctionnalités en fournissant un espace centralisé pour héberger et partager le code, ainsi que des outils de gestion de projet pour suivre les problèmes, les demandes de fusion et les tâches. L'intégration de Git et GitHub dans le flux de travail de développement offre une meilleure traçabilité, une collaboration efficace et une facilité de déploiement.



Toutes les ressources et les codes source du projet sont disponibles sur notre compte GitHub.

<https://github.com/boukary-derra/pfe/tree/main/code>