Détection des lignes aéroportuaires par méthode de filtrage particulaire: Évaluation de fonctions d'observations

Esteban Perrotin, Claire Meymandi-Nejad, Ariane Herbulot, Michel Devy, Fabrice Bousquet

Détection des lignes aéroportuaires par méthode de filtrage particulaire : Évaluation de fonctions d'observations

Esteban Perrotin[1,2] Claire Meymandi-Nejad[2] Ariane Herbulot[2]
Michel Devy [2] Fabrice Bousquet[1]

[1] AIRBUS Operation
[2] LAAS-CNRS

esteban.perrotin@airbus.com

Résumé

Ce papier présente une méthode exploitant le filtrage particulaire (méthode de Monte-Carlo séquentielle) pour la détection des marquages dans une zone aéroportuaire, depuis des images acquises par une caméra embarquée sur un aéronef. Un marquage est considéré comme étant une succession d'imagettes liées par une chaine de Markov. Nous faisons l'analogie entre la détection spatiale des marquages dans une image et le suivi temporel d'un véhicule automobile dont le modèle dynamique est similaire. Puis nous proposons une solution par filtrage particulaire. Les particules représentent les caractéristiques des imagettes, nous utilisons un classifieur d'image pour attribuer un poids à chaque particule. Trois classifieurs sont comparés ; le premier est basé sur le filtre de Gabor, le second sur la décomposition en ondelettes puis l'utilisation d'un SVM et enfin le troisième utilise un réseau de neurones à convolution. Les résultats sont validés sur des images réelles et simulées.

Mots Clef

vision par ordinateur, détection de ligne, filtrage particulaire

Abstract

This paper presents a method for ground marks detection in airport areas by using particle filtering (sequential Monte-Carlo method). The ground mark is considered as a succession of thumbnails linked by a Markov chain. We make an analogy between the spatial detection of lines in an image and the temporal tracking of a vehicle with a similar dynamic model. Then we propose a solution by particle filtering. The particles represent the characteristics of thumbnails, we use an image classifier to assign a weight to each particle from the thumbnails. Three classifiers are compared, the first is based on the Gabor filter, the second one exploits a polynomial SVM on a wavelets decomposition and the third one learn how to represent and classify a thumbnail by a CNN. The results are validated on real and simulated images.

Keywords

computer vision, line detection, particle filtering

1 Introduction

La conduite assistée est un enjeu majeur du $21^{ème}$ siècle. Dans ce contexte, la vision par ordinateur a pris une place capitale dans la perception de l'environnement des véhicules. Parmi ses nombreux objectifs, on peut citer la détection des obstacles, la détection des éléments de signalisations ou encore l'estimation du comportement du véhicule au regard de son environnement (odométrie visuelle, SLAM, ...). Dans notre cas, nous nous intéressons à la navigation autonome d'un aéronef sur les taxiways dans les zones aéroportuaires ; dans cet article, nous nous focalisons sur la détection de la signalisation acquises par des caméras visibles montées sur l'avion. Parmi les éléments de signalisation, les marquages au sol fournissent d'importantes informations pour la navigation. Ces informations pourront servir à construire diverses fonctions : garder une trajectoire fixe (maintien de l'axe), estimer la position relative et absolue de l'avion, suivre le tracé d'une ligne, ou encore alerter le pilote en cas de dérive afin de prévenir une potentielle sortie de piste. Dans nos travaux, nous cherchons à identifier quels pixels correspondent à des marquages dans l'image, puis à les regrouper selon le mar-

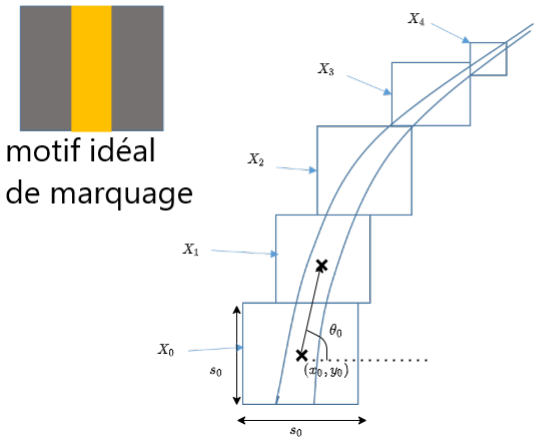

FIGURE 1 – Principe de détection de ligne selon un modèle Markovien. La succession des états X_k décrit la progression de la ligne à travers l'image

quage associé et enfin à extraire une représentation de ce marquage. Dans cet article, nous présentons une méthode basée sur le filtrage particulaire afin d'extraire une chaine de sous fenêtres (imagettes) de l'image décrivant une ligne. Les autres traitements (extraire l'équation de ligne, identifier le type du marquage...) ne sont pas évoqués dans la article.

1.1 Travaux antérieurs

Dans le domaine automobile, la détection de lignes est un sujet bien étudié depuis les années 2000. Plusieurs solutions ont été proposées. Les premières méthodes reposaient sur la transformation de Hough pour extraire de lignes et de courbes [1], dans le but de détecter les marquages des différentes voies de circulation. La détection de ces voies a aussi était réalisée par plusieurs méthodes de filtrage, notamment en utilisant le filtrage de Kalman [2]. Plus récemment, les méthodes par réseaux de neurones ont permis une avancée dans le domaine [3]. Ces différentes méthodes ont été essayées et comparées dans l'état de l'art pour le domaine automobile [4, 5].
Dans le contexte aéroportuaire les marquages au sol délimitent des voies de navigation, des trajectoires à suivre ou d'autres informations (numéro de la voie, information d'arrêt, etc.). Ces signalisations répondent à des normes précises quant à leur apparence (couleur) et leur forme (largeur, courbe...) [6]. Dans cet article notre objectif est de détecter les marquages au sol définissant les trajectoires à suivre et les délimitations des voies de circulation des avions. Ces marquages apparaissent principalement sous la forme de lignes. Pour détecter ces lignes, nous les considérons comme une succession de points liés par une chaine de Markov. Chacun de ces points peut se représenter sous la forme d'un vecteur d'état $X_k = (x_k, y_k, \theta_k, d\theta_k, s_k)$, avec (x_k, y_k) représentant les coordonnées du point dans le plan de l'image, θ_k l'orientation du marquage dans l'image en ce point, $d\theta_k$ sa vitesse angulaire et s_k le pas séparant deux points. En connaissant les paramètres d'un point, on peut prédire le point suivant selon un modèle dynamique.
En considérant un voisinage de taille $s_k \times s_k$ autour du point (x_k, y_k), on peut extraire la ligne comme une succession d'imagettes. La figure 1 schématise ce modèle. Le modèle dynamique de l'évolution spatiale de la ligne dans une image, s'apparente au modèle dynamique de l'évolution temporelle d'un véhicule. L'estimation de la trajectoire d'un véhicule à partir de mesures extérieures (GPS, capteur inertiel, etc.) est un problème récurrent. Pour y répondre, les travaux de Gustafsson et al. [7] proposent d'utiliser une méthode de filtrage particulaire. Nos travaux s'inspirent de cette approche en cherchant à estimer l'évolution des lignes dans le domaine de l'image selon le même type d'algorithme.
Pour détecter les marquages aéroportuaires, une précédente version avait été proposée [8]. Cette approche utilise aussi le principe du filtrage particulaire. Cependant, le modèle dynamique est plus simple et considère uniquement la propagation selon l'axe vertical de l'image. Il faut alors transposer l'image pour obtenir les marquages horizontaux puis fusionner les données. De plus, la sélection des différentes particules se fait selon deux informations, une basées sur la couleur (pour les marquages jaunes) et une sur le gradient entre tarmac et marquage. Dans notre cas, nous considérons un modèle plus complexe qui capture directement la géométrie des lignes. La sélection des particules se fait à partir d'un classifieur d'image qui prédit la probabilité que le voisinage d'une particule corresponde à un marquage. Notre modèle se rapproche aussi des travaux de Tran et al. [9] et des travaux de Prateek et al. [10]. Dans leurs travaux, Tran et al. traquent un objet au cours du temps. L'estimation du déplacement au cours du temps se fait par filtrage particulaire. La sélection des particules se fait par la transformation de Hough généralisée dont la référence est mise à jour à chaque itération.

1.2 Organisation de l'article

Cet article est organisé comme suit. Dans la section 2, nous discutons des différents éléments

du filtre particulaire et des modèles utilisés. Dans la section 3, nous nous concentrons sur les trois approches proposées représenter et classifier les imagettes, afin d'estimer la vraisemblance qu'elles correspondent à un marquage. En section 4, nous présentons nos résultats sur la classification des imagettes ainsi que des résultats qualitatifs du filtre. Puis nous concluons sur les résultats de l'utilisation des différentes méthodes dans la section 5.

2 Filtrage particulaire

Le filtrage particulaire est une méthode de simulation séquentielle de type Monte-Carlo. Les particules explorent l'espace d'état en évoluant selon un modèle dynamique. Un processus de sélection exploitant les observations vient concentrer les particules dans les régions d'intérêts de l'espace d'état.

2.1 Principe général

L'objectif d'un filtre particulaire est d'estimer la densité a postériori des variables d'état compte tenu des variables d'observations selon une densité de distribution a priori. Le filtrage particulaire est souvent utilisé dans le cas de série temporelle dont la distribution des paramètres du système évolue au cours du temps. On note X_k les variables d'états (cachées) et Z_k les variables d'observations. X_0, X_1, ..., X_k est un processus de Markov, ce qui signifie que X_k n'est dépendant que de l'itération précédente X_{k-1}. On souhaite décrire la densité de probabilité a postériori $p(X_k|Z_k)$. Pour estimer cette densité de probabilité, le filtre particulaire propose une approximation discrète à partir de N échantillons aléatoires de l'espace d'état $\{X_k^{(n)}\}_{n=1}^N$. À chacun de ces échantillons est attribué un poids $\{w_k^{(n)}\}_{n=1}^N$ correspondant à la probabilité que le vecteur d'état réel soit proche de l'échantillon X_k^n. L'ensemble $\{(X_k^{(n)}, w_k^{(n)})\}_{n=1}^N$ est alors vu comme une approximation de la densité $p(X_k|Z_k)$.

La figure 2 montre les principales parties du filtrage particulaire. Tout d'abord, la phase d'initialisation va attribuer des valeurs à l'état initial en se basant sur la détection d'un début par différentes analyses de l'image : une étude colorimétrique ou du gradient, dans des régions d'intérêt choisis en fonction de la position de la caméra (par exemple, recherche d'un début en bas de l'image là où la résolution est la meilleure). Ensuite, la phase d'échantillonnage va créer des

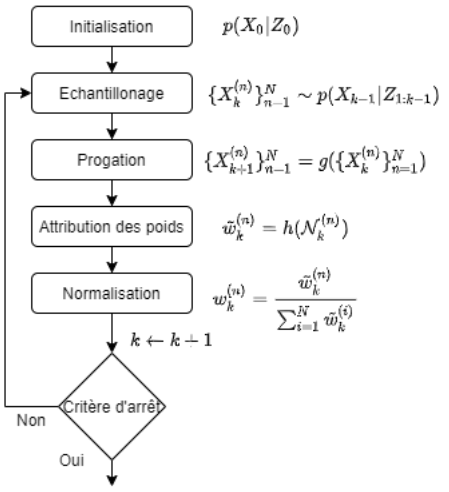

FIGURE 2 – Schéma général du filtre particulaire

particules autour de l'état initial pour chaque paramètre du vecteur d'état selon des distributions choisies a priori (dans notre cas, nous choisissons des Gaussiennes). La phase de propagation applique le modèle dynamique à chaque particule. Puis on attribue des poids à chacune des particules en utilisant un classifieur d'imagettes. Ces poids sont normalisés entre eux, de façon à représenter une densité de probabilité. Enfin, on décide, selon un critère d'arrêt fixé, soit d'arrêter le filtre, soit de le répéter en repartant de l'étape d'échantillonnage. On peut ensuite sélectionner une nouvelle région de début d'une autre ligne jusqu'à extraire toutes les lignes présentes dans l'image.

2.2 Vecteur d'état et vecteur d'observation

Dans cet article nous désirons détecter les lignes aéroportuaires en utilisant l'analogie avec le suivi de véhicule et le suivi de piéton. Fredrik Gustafsson et al. [11, 7] ont proposé plusieurs écritures et lois dynamiques pour faire du suivi temporelle de véhicule (automobile, nautique et aéronautique). De telles approches dites de Line Tracking ont aussi été proposées pour segmenter des courbes depuis des scans laser [12].

À la différence de nombreuses applications, nous ne disposons pas de mesures extérieures (altimètre, radar, etc.) pour obtenir un vecteur d'observation. Cependant, il est possible de créer un équivalent en utilisant directement l'image comme le propose Czyz J. et al. [13] pour la détection et le suivi temporelle de piéton. En créant une imagette $\mathcal{N}_k$ carrée à partir de l'image I et du vecteur X_k. L'imagette est centrée en (x_k, y_k)

avec une taille $s_k \times s_k$. On peut alors créer un vecteur d'observation $Z_k = (z_{1,k}, z_{2,k}, ..., z_{M,k})$ composé de M mesures, avec :

$$Z_k = h(\mathcal{N}_k) + V_K \tag{1}$$

Dans notre cas, une mesure correspond à un score que l'imagette $\mathcal{N}_k$ décrive correctement le motif d'une ligne. Ce score est donnée par un classifieur h entachée d'un bruit d'observation V_k (imprécision de l'attribution du score). Dans cet article, nous considérons une seule observation et notons $Z_K = z_k \in [0, 1]$ la valeur attribuée à une imagette $\mathcal{N}_k$ par la fonction h. Nous détaillons dans la section 3 la construction de cette fonction h.

2.3 Modèle dynamique

Le modèle dynamique décrit l'évolution des paramètres de l'espace d'état à chaque itération comme suit :

$$X_{k+1} = g(X_k, U_k) \tag{2}$$

où X_k désigne l'état courant et X_{k+1} l'état à l'itération suivante. La fonction g décrit la transition du système de l'itération k à l'itération $k+1$ selon un terme de bruit non connu noté U_k. On choisit le modèle d'évolution des paramètres comme suit :

$$x_{k+1} = x_k + s_k cos(\theta_k) \tag{3a}$$
$$y_{k+1} = y_k + s_k sin(\theta_k) \tag{3b}$$
$$\theta_{k+1} = \theta_k + d\theta_k \tag{3c}$$
$$d\theta_{k+1} = \theta_k - \theta_{k-1} \tag{3d}$$
$$s_{k+1} = min(max(s_k, 11), 51) \tag{3e}$$

Le pas s_k, qui correspond à la distance entre deux itérations (et fixe la taille de l'imagette $\mathcal{N}_k$), doit être fixé avec une valeur minimale et selon l'application une valeur maximale. En effet, au cours des itérations du filtre, si ce paramètre venait à se retrouver trop proche de zéro (par échantillonnage), le filtre n'avancerait plus. De plus, ce paramètre sert aussi de taille pour obtenir l'imagette $\mathcal{N}_k$ ce qui peut introduire des erreurs s'il vient à être trop petit. Nous avons empiriquement fixé cette valeur minimale à 11 pixels, ainsi qu'une valeur maximale à 50 pixels. Une autre solution consisterait à fixer ce paramètre en fonction de la distance réelle entre le pixel (x_k, y_k) et le morceau de la ligne observée.

2.4 Attribution des poids

Pour résoudre le problème précédent à partir des équations 1 et 2, le filtre particulaire crée N particules X_k^n représentant des propositions pour le vecteur d'état courant X_k. Puis, il leur attribue une probabilité que le vecteur d'état proposé corresponde à un morceau de ligne. De cette façon, l'ensemble $\{(x_k^{(n)}, y_k^{(n)}), w_k^{(n)}\}_{n=1}^N$ représente la densité de probabilité que les pixels de l'imagette $\mathcal{N}_k$ centrée sur (x_k, y_k) soient dans le voisinage d'une ligne.

Algorithm 1 (Un cycle) Calculer une approximation de la densité $p(X_k|Z_k)$ par l'ensemble $\{X_k^{(n)}, w_k^{(n)}\}_{n=1}^N$ en associant à chaque particule X_k^n un poids w_k^n à partir d'une distribution initiale $\{X_0^{(n)}, w_0^{(n)}\}_{n=1}^N$

(Prédiction) Construire X_{k+1}^n à partir de X_k^n en utilisant l'équation 3

(Obtention de mesures) Créer des mesures Z_k^n (équation 1)

(Attribution des poids) Calculer la vraisemblance de chaque particule avec l'équation 4. Puis, mettre à jour les poids :

$$\tilde{w}_k^n = w_{k-1}^n L_k(X_k^n)$$

(Normalisation des poids) Normaliser les poids des particules, de sorte que leur somme soit égale à un : $w_k^n = \frac{\tilde{w}_k^n}{\sum_{i=1}^N \tilde{w}_k^i}$

(Échantillonnage) Tirer un nouvelle échantillon de N particules selon la distribution $\{X_k^{(n)}, w_k^{(n)}\}_{n=1}^N$

Pour attribuer les poids w_k^n, différents algorithmes de filtrage particulaire existent []. L'algorithme 1 montre la solution choisie ici.

Cet algorithme part d'un échantillonnage $\{(X_0^{(n)}, w_0^{(n)})\}_{n=1}^N$ initial, dont les poids $\{w_0^{(n)}\}_{n=1}^N$ sont fixés à $\frac{1}{N}$. La sélection des états $\{X_0^{(n)}\}_{n=1}^N$ n'est pas détaillée dans cet article. Cet échantillonnage est ensuite propagé selon le modèle dynamique décrit par la fonction g. On obtient ainsi l'échantillonnage $\{X_1^{(n)}\}_{n=1}^N$. Puis à partir d'un classifieur h, nous attribuons une observation Z_k^n à chaque particule X_k^n de l'échantillonnage. L'observation Z_k^n correspond à la probabilité que l'imagette contienne un morceau de ligne selon le classifieur h. Dans notre cas, elle correspond à un simple scalaire noté $z_k^n \in [0, 1]$. Grâce à ces observations, on définit la fonction de vraisemblance entre une particule X_k^n et l'ensemble des couples états/observations

$\{(X_k^{(n)}, Z_k^{(n)})\}_{n=1}^N$ de l'itération k par :

$$L_k(X_k^n) = z_k^n \sum_{i=0}^{N} e^{-\frac{d(X_k^n; X_k^i)}{(z_k^i)^2}} \quad (4)$$

Avec d la fonction définissant la distance entre deux états. Dans notre cas, elle est correspond à la distance euclidienne :

$$d(X_k^i; X_k^j) = \|(x_k^i, y_k^i) - (x_k^j, y_k^j)\|_2 \quad (5)$$

Puis les poids sont mis à jour et normalisés. Finalement, en utilisant la méthode de rééchantillonnage utilisé dans [8], on tire N nouvelles particules auxquelles on affecte le poids de leur particule mère. On réitère l'algorithme jusqu'à ce que la vraisemblance maximale de l'itération courante soit en dessous de la vraisemblance maximale de la première itération.

À noter que la distance d n'utilise que deux paramètres des vecteurs d'états. Il serait possible de considérer des fonctions de distance plus complexes, afin de prendre en compte tous les paramètres des vecteurs d'états.

Ce modèle de filtrage est dépendant de la fonction h qui génère les observations. Dans la section suivante nous construisons et comparons trois classifieurs candidats comme fonction h.

3 Classification des observations

Pour attribuer le poids aux particules la fonction d'observation h est primordiale. Comme dit précédemment, la particule $X_k^n = \{x_k^n, y_k^n, \theta_k^n, d\theta_k^n, s_k^n\}$ décrit l'état d'une imagette $\mathcal{N}_k^n$ à l'itération k. L'idée principale est de construire un classifieur qui estime la possibilité qu'une imagette donnée contienne un morceau de ligne. Pour créer ce classifieur, différentes stratégies sont proposées. La première est de créer un classifieur en utilisant le filtre de Gabor. La seconde est d'extraire la décomposition en ondelettes de l'imagette puis de la classifier grâce à un SVM non linéaire. La troisième consiste à construire un CNN.
Pour extraire les éléments reconnaissables des imagettes, nous avons besoin d'un jeu de données d'apprentissage. Dans la partie suivante, nous détaillons la construction de ce jeu de données. Le jeu de données de validation est construit de la même façon (2000 imagettes) en utilisant des images différentes du jeu de données d'apprentissage.

3.1 Construction du jeu de données

Le jeu de données est composé d'imagettes de taille 33×33 pixels, sélectionnées à partir d'images annotées manuellement. On tire au hasard 2000 imagettes à partir de plusieurs images selon une probabilité uniforme. Si au moins 100 pixels de l'imagette ont été annotées comme appartenant à un marquage, alors on classe l'imagette dans la catégorie 'marquage'. Sinon, l'imagette est classée dans la catégorie 'non marquage'. La figure 3 montre des exemples d'imagettes.
Cette méthode de construction du jeu de données a l'avantage d'être rapide et facile à implémenter mais comporte deux inconvénients. L'annotation manuelle des pixels n'est pas parfaite, ce qui peut apporter des faux positifs dans la base de données. De plus, le seuil fixé à 100 pixels est arbitraire. Si une imagette contient 99 pixels manuellement annotées comme faisant partie de la classe 'marquage', l'imagette sera notée dans la classe 'non-marquage'. Ce qui peut être considéré comme un faux négatif. Les performances des algorithmes de classification sont donc à relativiser vis-à-vis de la précision de la base de données.

FIGURE 3 – La première ligne montre des imagettes considérées comme positives. La seconde ligne montre des imagettes considérées négatives.

L'un des principes de notre filtre est d'utiliser des imagettes de taille variable. Cependant, un classifieur doit généralement avoir des images de taille fixe en entrée. Le plus souvent les images sont redimensionnées pour correspondre au besoin du classifieur. Pour prendre en compte ce redimensionnement, nous construisons un deuxième jeu de données augmenté qui contient aussi 2000 imagettes mais dont la taille est tirée au hasard (entre 11 et 51 pixels) puis redimensionnées en 33×33.

3.2 Filtre de Gabor

Le filtre de Gabor est utilisé depuis longtemps pour classifier des textures. Il s'agit d'un filtre linéaire. Les textures sont caractérisées par leur réponse à ce filtre. Il peut être utilisé de façon supervisée ou non supervisée [14]. Le filtre de Gabor peut être vu comme le produit de convolution

d'une sinusoïde par une Gaussienne :

$$GB_{\lambda,\omega}(u,v) = e^{-\frac{(\hat{u}^2+\gamma^2\hat{v}^2)}{2\sigma^2}}\cos(\frac{2\pi}{\lambda}\hat{u})) \quad (6a)$$

$$\hat{u} = u\cos\theta + v\sin\theta \quad (6b)$$

$$\hat{v} = -u\sin\theta + v\cos\theta \quad (6c)$$

Les hyperparamètres σ et γ dépendent de l'implémentation utilisée. Ici, ils sont respectivement fixés à 1 et 0.5. Les paramètres ω et λ correspondent à l'orientation de la sinusoïde et à sa fréquence. Pour faire de l'apprentissage supervisé, on utilise la réponse du filtre de Gabor selon les différents paramètres λ et ω comme éléments descriptifs. Un classifieur vient ensuite apprendre à trier ces éléments. Dans cette première approche pour évaluer la pertinence d'une imagette, nous privilégions une méthode simple, donc quasiment non supervisée. Pour ce faire, nous calculons par convolution la réponse du filtre sur un motif de ligne dans une imagette idéal $\bar{\mathcal{N}}$, en cherchant les valeurs optimales des paramètres λ et ω :

$$\bar{C} = \max_{\omega,\lambda} |GB_{\lambda,\omega}(\bar{\mathcal{N}})| \quad (7)$$

À partir de cette réponse optimale, notée $\bar{C}$, nous attribuons un score de confiance, à l'imagette $\mathcal{N}$, de contenir un marquage :

$$\max_{\omega,\lambda} e^{-||GB_{\lambda,\omega}(\mathcal{N})|-\bar{C}|} \quad (8)$$

Plus la réponse du filtre pour l'imagette N est proche de la réponse de l'imagette idéale, plus le score se rapproche de 1. En revanche, plus la réponse s'en éloigne plus le score se rapproche de 0. Pour la comparer aux autres classifieurs, on regarde la distribution du résultat dans le jeu de donnée d'entrainement et/ou de validation. Il est ainsi possible de trouver un seuil pour fixer la décision si l'imagette est de la classe 'marquage' ou non. Cette méthode un peu ad oc a l'avantage d'obtenir une estimation de l'orientation de la ligne. En effet, la réponse optimale du filtre est obtenue lorsque l'orientation ω correspond au gradient entre la ligne et le tarmac.

3.3 Ondelettes et SVM

Les classifieurs de type SVM (support vector machine) sont très populaires pour résoudre des problèmes de classification. Dans le cas de la classification d'image, il est nécessaire d'extraire des éléments pertinents de l'image avant de les classifier avec le SVM. La stratégie choisie ici est d'utiliser la décomposition en ondelettes pour décrire l'image selon un dictionnaire d'ondelettes

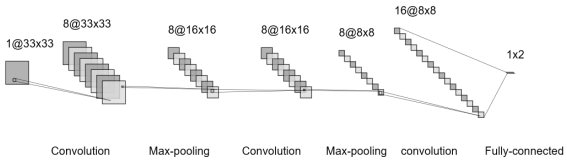

FIGURE 4 – Architecture du réseau de neurones utilisé. Il s'agit d'une succession de couche de convolution et de pooling relié à une couche complètement connectée.

après l'avoir convertie en niveau de gris. Plusieurs modèles d'ondelettes ont été essayés (Haar, Daubechies, etc.) et nous avons gardé l'ondelette de Daubechies 8. À première vue, l'ondelette de Haar semblait la plus adaptée au vu de la forme du motif à décrire (gradient entre le marquage et le tarmac). Cependant, lorsque le marquage est loin de la caméra, les imagettes sont très mal résolues et le niveau de bruit est très important. Le gradient entre le marquage et le tarmac est particulièrement diffus, ce qui peut expliquer que les ondelettes de Daubechies soient plus adaptées.

3.4 Convolutional Neural Network

Le troisième classifieur choisi dans cet article est un réseau de neurones à convolution (CNN). Ces dernières années les réseaux de neurones ont percé dans l'état de l'art dans de nombreux domaines. C'est en particulier le cas dans la classification d'images. Nous avons donc implémenté une architecture assez basique de CNN pour la comparer aux deux autres approches. L'architecture du réseau est montrée sur la figure 4.

L'imagette est donnée en entrée du réseau. Si nécessaire, elle est redimensionnée en 33×33. Une succession de plusieurs couches de convolutions et de pooling sont appliquées pour extraire des éléments descriptifs. La dernière couche est une couche totalement connectée qui produit en sortie un résultat entre 0 et 1 correspondant à la confiance sur la présence d'un marquage dans l'imagette. Nous avons aussi fait plusieurs essais sur l'importance de la normalisation des données. Il est apparu que les résultats sont meilleurs en normalisant les données à la sortie de chaque couches de convolution.

Filtre	Entrainement(1)	Validation(1)	Validation(2)
Gabor	57.2	55.7	55.1
SVM	64.0	63.6	64.3
CNN	88.3	87.2	89.1

TABLE 1 – Accuracy des différents filtres selon l'entrainement 'taille imagette fixe'(1) ou 'taille imagette variable'(2)

4 Résultats

4.1 Comparaison sur la classification des imagettes

Les résultats des trois classifieurs sont donnés dans le tableau 1. La métrique utilisé est l'accuracy. Il s'agit de la métrique la plus courante pour les problèmes de classifications binaires. On la donne sur trois jeux de données. Le premier jeu est celui utilisé pour l'apprentissage avec des taille d'imagette fixes (33×33 pixels). Le second est celui utilisé pour le jeu de données de validation. Dans ce jeu de données la taille des imagettes varies selon nos deux bornes. Enfin on donne le résultats sur le jeu de données de validation mais avec un entrainement à taille variable.

Comme attendu, le réseau de neurones affiche un score excellent pour ce type de problématique. Les autres types de classifieurs affichent des performances faibles. L'extraction des éléments d'intérêt est assez compliqué en utilisant des dictionnaires pré-établis (Gabor et ondelettes). Néanmoins, on peut espérer améliorer légèrement les performances en choisissant mieux certains hyperparamètres. Le redimensionnement des images dans le jeu de données d'apprentissage à un impact positif mais non significatif dans la qualité de la classification.

4.2 Résultats du filtre particulaire

On discute de notre modèle de filtrage particulaire ainsi que de l'impacte du choix des classifieurs comme fonction d'observation. La figure 5 montre la propagation du filtre le long d'une ligne. Le résultat est bien plus précis en utilisant le CNN que les autres classifieur. Cependant en montant le nombre de particules, il est possible d'obtenir des régions d'intérêts pertinentes même avec des classifieurs peu performants.

Le modèle su suivit de ligne est assez simple mais fonctionne correctement pour suivre une seule ligne. Dans le cas de croisement il est nécessaire

FIGURE 5 – Déplacement de la région d'intérêt définie selon la vraisemblance des particules pour la détection d'une ligne. La position de la particule avec le plus grand poids est affiché
.

de rajouter une heuristique : si le filtre s'est séparé en deux parties très vraisemblable dont les vecteurs d'état moyens comportent des angles très différents, alors il faut séparer le filtre en deux filtres distincts.

Le modèle peut continuer à donner des résultats assez pertinents même sur des parties d'images très peu résolues. Il est cependant nécessaire d'utiliser un classifieur très précis pour ces parties (augmenter le nombre de particule ne permet pas d'améliorer la détection si le classifieur est complètement incapable d'obtenir une prédiction comportant au moins quelques informations).

5 Conclusion

Dans cet article, nous présentons un modèle de filtrage particulaire spatial pour détecter les marquages des zones aéroportuaires. Ce modèle considère les marquages comme une succession d'état lié par une chaine de Makov et estime la densité de probabilité qu'un ensemble d'états décrivent correctement un marquage au sol. Pour ce faire, le filtrage particulaire se base sur l'appréciation d'un classifieur d'imagette qui évalue la possibilité que l'imagette en question décrive un marquage. Trois classifieurs sont évaluées. Le premier est basé sur le filtre de Gabor et une distance empirique. Un autre sur la décomposition en ondelette et un classifieur SVM. Le troisième est un réseau de neurones à convolution. Sans étonnement, le réseau de neurones obtient largement les meilleures performances en classification d'imagettes. Pour la problématique de classification, le CNN obtient très largement les meilleurs performances. En utilisant ce CNN comme fonction d'observation, cette méthode de détection de ligne par filtrage particulaire obtient visuellement de bonnes

performances sur des images réels et simulées. Par rapport à la première approche proposé dans [8], nous ne sommes plus du tout sensibles aux variations de la couleur, car la couleur sera seulement exploitée pour l'initialisation. Par ailleurs on peut extraire les lignes quelque soit leurs orientations.

Dans de futurs travaux, nous quantifierons numériquement les résultats de ce filtre. Puis, nous augmenterons ce filtre particulaire affin de détecter et de traquer les marquages spatialement et temporellement. Il est aussi possible de fusionner les résultat des différents classifieurs en espérant augmenter la confiance dans la détection. Finalement, les motifs détectés seront identifiés et catégorisés (stop bar, ligne central, runway, symboles alpha numérique).

Références

[1] A. Kumar and P. Simon, "Review of lane detection and tracking algorithms in advanced driver assistance system," *International Journal of Computer Science and Information Technology*, vol. 7, pp. 65–78, 08 2015.

[2] R. Aufrer, F. Marmoiton, R. Chapuis, F. Collange, and J. Derutin, "Détection de route et suivi de véhicules par vision pour l'acc," *GRETSI, Saint Martin d'Hères, France*, 2000.

[3] M. Haris and A. Glowacz, "Lane line detection based on object feature distillation," *Electronics*, vol. 10, no. 9, 2021.

[4] A. Kumar and P. Simon, "Review of lane detection and tracking algorithms in advanced driver assistance system," *International Journal of Computer Science and Information Technology*, vol. 7, pp. 65–78, 08 2015.

[5] J. Cao, S. Song, W. Xiao, and Z. Peng, "Lane detection algorithm for intelligent vehicles in complex road conditions and dynamic environments," *Sensors*, vol. 19, p. 3166, 07 2019.

[6] ICAO, *AERODROMES : aerodromes design and operations*. International Civil Aviation Organization (ICAO), 2018.

[7] F. Gustafsson, F. Gunnarsson, N. Bergman, U. Forssell, J. Jansson, R. Karlsson, and P.-J. Nordlund, "Particle filters for positioning, navigation, and tracking," *IEEE Trans. Signal Process.*, vol. 50, pp. 425–437, Feb. 2002.

[8] C. Meymandi-Nejad, S. E. Kaddaoui, M. Devy, and A. Herbulot, "Lane detection and scene interpretation by particle filter in airport areas," in *14th International Conference on Computer Vision Theory and Applications (VISAPP 2019)*, (Prague, Czech Republic), Feb. 2019.

[9] A. Tran and A. Manzanera, "A versatile object tracking algorithm combining particle filter and generalised hough transform," in *2015 International Conference on Image Processing Theory, Tools and Applications (IPTA)*, pp. 105–110, 2015.

[10] P. K. Gaddigoudar, T. R. Balihalli, S. S. Ijantkar, N. C. Iyer, and S. Maralappanavar, "Pedestrian detection and tracking using particle filtering," in *2017 International Conference on Computing, Communication and Automation (ICCCA)*, pp. 110–115, 2017.

[11] F. Gustafsson, "Particle filter theory and practice with positioning applications," *IEEE Aerospace and Electronic Systems Magazine*, vol. 25, no. 7, pp. 53–82, 2010.

[12] M. Peter, S. R. U. N. Jafri, and G. Vosselman, "Line segmentation of 2d laser scanner point clouds for indoor slam based on a range of residuals," *ISPRS Annals of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, pp. 363–369, 2017.

[13] J. Czyz, B. Ristic, and B. Macq, "A color-based particle filter for joint detection and tracking of multiple objects," in *Proceedings. (ICASSP '05). IEEE International Conference on Acoustics, Speech, and Signal Processing, 2005.*, vol. 2, pp. ii/217–ii/220 Vol. 2, 2005.

[14] J. Recio, L. Fernández, and A. Fernandez, "Use of gabor filters for texture classification of digital images," *Física de la tierra, ISSN 0214-4557, Nº 17, 2005, pags. 47-59*, 01 2005.