HAL open science

Processus ponctuels et réseaux de neurones convolutifs pour la détection de véhicules dans des images de télédétection

Jules Mabon, Mathias Ortner, Josiane Zerubia

▶ **To cite this version:**

Jules Mabon, Mathias Ortner, Josiane Zerubia. Processus ponctuels et réseaux de neurones convolutifs pour la détection de véhicules dans des images de télédétection. ORASIS 2021 - 18èmes Journées francophones des jeunes chercheurs en vision par ordinateur, Centre National de la Recherche Scientifique [CNRS], Sep 2021, Saint Ferréol, France. hal-03339656

HAL Id: hal-03339656

https://hal.science/hal-03339656v1

Submitted on 9 Sep 2021

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Processus ponctuels et réseaux de neurones convolutifs pour la détection de véhicules dans des images de télédétection.

J. Mabon[1] M. Ortner[2] J. Zerubia[1]

[1] Inria, Université Côte d'Azur, Sophia-Antipolis, France
[2] Airbus Defense and Space, Toulouse, France

Contact : jules.mabon@inria.fr

Résumé

Cet article présente une méthode combinant processus ponctuels et réseaux de neurones convolutifs pour la détection de petits objets dans des images de télédétection. Dans un tel contexte, quand les objets sont petits et leur densité est élevée, nous utilisons des a priori au sein d'une simulation de processus ponctuels dont l'attache aux données est obtenue par apprentissage d'un réseau de neurones, permettant ainsi de se soustraire à la conception manuelle d'un terme de détection spécifique.

Mots Clef

reconnaissance de formes, détection de petits objets, CNN, processus ponctuels, télédétection,

Abstract

This paper presents a method to combine point processes and convolutional neural networks for detecting small objects in remotely sensed images. In such a context, when objects are small and their density is high, we use priors within a point process simulation. The data term of this point process has been learned with a neural network, thus avoiding the handcrafting of a specific detection term.

Keywords

pattern recognition, small objects detection, CNN, point process, remote sensing,

1 Introduction

Cet article propose un modèle pour la détection de véhicules dans des images aériennes ou satellitaires. Dans un tel contexte, les objets d'intérêt ont une taille de seulement quelques pixels de large, faisant perdre alors l'information de texture. De plus, ces objets sont parfois distribués densément (par exemple des voitures sur un parking), ce qui peut rendre la distinction des instances plus difficile.
Pour les méthodes de détection d'objets à base de CNN, on distingue deux familles de méthodes [18] : celles fondées sur la proposition de régions (comme Faster R-CNN [12]), où un premier réseau effectue une segmentation des régions d'intérêt, puis une seconde partie du modèle effectue un post-traitement pour en extraire les positions des objets. D'autres approches intègrent l'extraction des ancres des boites de détection dans un seul réseau (comme YOLO [11] ou RetinaNet [8]). Cependant, ces dernières sont peu adaptées à la détection de petits objets de par la taille des boites de détection. C'est pourquoi, nous nous intéressons aux méthodes de segmentation suivies d'un post-traitement, avec pour objectif d'extraire des informations de détection de bas niveau avec un CNN, puis d'ajouter des *a priori* dans la phase de post-traitement.
Dans cet article, nous nous plaçons dans le cadre des approches globales de type géométrie stochastique [7, 15] qui proposent de résoudre conjointement la détection et la sélection des objets. L'intérêt est double : d'une part il devient possible d'ajouter des modèles d'interaction entre les objets via des *a priori*, d'autre part de telles approches permettent de vectoriser l'information extraite de l'image. Ces approches nécessitent de construire une énergie de détection qui, pour chaque point du processus ponctuel, mesure la vraisemblance de la présence d'un objet à un emplacement donné dans l'image. Ces énergies sont faciles à construire pour des contextes où le contraste entre les objets d'intérêt et le fond est fort. Mais des situations d'illumination et des contextes visuels changeants rendent la conception d'une telle énergie plus complexe et coûteuse en temps de calcul. Dans cet article, nous proposons d'apprendre le terme d'attache aux données via un CNN, et d'utiliser la carte de détection obtenue pour échantillonner plus rapidement le processus ponctuel. Ceci constitue notre contribution principale. Les processus ponctuels permettront d'utiliser des *a priori* dans des contextes où la densité d'objets d'intérêt est élevée. Nous nous focalisons ici sur la détection de voitures dans des images de télédétection.

1.1 Travaux antérieurs

Les approches globales comme [2, 4, 7] montrent qu'il est possible de détecter des objets comme des bateaux dans un port, des routes, ou des cellules dans des images de microscopie via les processus ponctuels. Cependant, ce type d'approche utilise des mesures de contraste fondées sur

un contexte particulier : par exemple [2] se base sur le contraste entre les bateaux (clairs) sur la mer (sombre). Des modèles comme [11] ou [13] ont montré la capacité des CNN à détecter des objets que ce soit dans des scènes "du quotidien" ou des cas spécifiques d'images de biologie. Mais, des approches comme [11] se généralisent très mal pour des petits objets très proches (et c'est souvent le cas dans les images satellitaires de scènes urbaines).

2 Processus ponctuels

Nous définissons le support de l'image comme $S \subset \mathbb{R}^d$, une configuration de points $\mathbf{x}$ est un ensemble fini non ordonné d'éléments de S. Pour $n \in \mathbb{N}$, soit Ω_n l'ensemble des configurations de n points $\{x_1, \ldots, x_n\}$. Un processus ponctuel est une application mesurable X de l'espace des probabilités vers l'ensemble des configurations $\Omega = \cup_{n=0}^{\infty} \Omega_n$ tel que pour tout borélien $A \subset S$ le nombre de points $N_X(A)$ tombant dans A est une variable aléatoire finie. Le processus ponctuel canonique est le processus de Poisson uniforme sur $\mathbb{R}^d$ et est tel que :

1. Pour tout borélien $A \subset \mathbb{R}^d$, $N_X(A)$ suit une loi de Poisson d'intensité $\lambda|A|$, où $\lambda > 0$ est l'intensité du processus ponctuel.
2. Si $A_1, \ldots, A_k$ sont des boréliens disjoints, alors $N_X(A_1), \ldots, N_X(A_k)$ sont des variables aléatoires indépendantes.

La loi du processus ponctuel de Poisson sur $S \subset \mathbb{R}^d$ est alors définie pour tout $B \subset S$ par la mesure de probabilité suivante :

$$\mu(B) = \sum_{n=0}^{\infty} \frac{\lambda^n e^{-\lambda|S|}}{n!} \int_{S^n} \mathbf{1}_B(\{x_1, \ldots, x_n\}) dx_1 \ldots dx_n \tag{1}$$

On peut aussi définir des processus ponctuels non-uniformes via leur densité h par rapport au processus ponctuel de Poisson de mesure de référence μ. Si h est une fonction positive sur Ω, on peut définir la mesure ν du processus ponctuel de densité h par rapport à μ comme :

$$\nu(B) = \int_B h(X)\mu(dX) \tag{2}$$

Si $0 < \nu(B) < \infty$ alors ν peut être normalisé pour construire une mesure de probabilité π définie comme $\nu(B)/\nu(\Omega)$.
Les modèles de sélection et d'interaction entre les points sont généralement définis à partir d'une densité de Gibbs non normalisée :

$$h(X) = \exp(-U(X)) \tag{3}$$

Ainsi, afin de pouvoir inférer une configuration d'objets X à partir d'une image I, il faut préalablement construire une énergie $U_{tot}(X, I)$ dont le minimum est atteint pour la configuration optimale (elle est définie dans l'équation 16). Cette énergie est composée de deux parties, une énergie d'attache aux données $U_d(X, I)$ qui mesure la vraisemblance de la configuration par rapport à l'image observée I et une énergie *a priori* ($U_{ap}(X) = \gamma_{al} U_{al}(X) + \gamma_s U_s(X)$ voir paragraphes 3.1 et 3.2) qui mesure la cohérence interne de la configuration X.

3 Modèles *a priori* pour la détection de véhicules

La vérité terrain est caractérisée par certaines contraintes et heuristiques qui lui sont propres, par exemple deux véhicules au sol ne peuvent pas se superposer au même emplacement ; il y a une contrainte de distance forte entre deux voitures et une configuration contenant ce type de superposition serait peu vraisemblable. La modélisation par les processus ponctuels nous permet d'ajouter de tels *a priori* sur les configurations de points. Ces *a priori* sont exprimés comme des énergies qui pénalisent (énergies positives) ou favorisent (énergies négatives) des configurations de points. Pour un espace image $S \subset \mathbb{R}^2$, nous considérons des configurations de points $\mathbf{x} = \{x_1, \ldots, x_{n(\mathbf{x})}\}$ avec $x_i \in S$ et $n(\mathbf{x})$ le nombre de points dans la configuration $\mathbf{x}$.

3.1 Contrainte de superposition

Sachant qu'un point x_i correspond à une détection de véhicule, les configurations qui contiennent des points trop proches ou superposés sont moins vraisemblables. L'*a priori* présenté dans cette partie, sous forme d'énergie, a pour but de pénaliser de telles configurations. Inspiré de [3], pour deux points x_i, x_j nous définissons alors un ratio de superposition A dans $[0, 1]$, en considérant chaque point comme un disque de rayon r fixé (la taille des véhicules étant la même partout dans l'image, nous choisirons r comme la demi-largeur d'un véhicule) :

$$A(x_i, x_j) = \frac{\text{Aire}(x_i \cap x_j)}{\min(\text{Aire}(x_i), \text{Aire}(x_j))} \tag{4}$$

Si x_i et x_j ne se superposent pas, $A(x_i, x_j) = 0$, s'ils se superposent totalement alors $A(x_i, x_j) = 1$ L'énergie U_s de superposition sur la configuration $\mathbf{x}$ s'écrit alors :

$$U_s(\mathbf{x}) = \sum_{1 \leq i \leq n(\mathbf{x})} \max_{\substack{1 \leq j \leq n(\mathbf{x}) \\ j \neq i}} (q_s(x_i, x_j)) \tag{5}$$

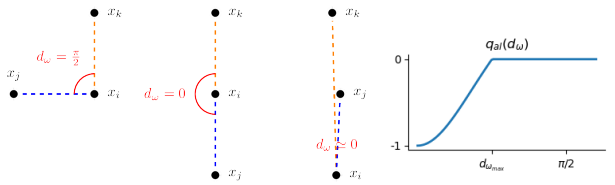
avec :

$$q_s(x_i, x_j) = \begin{cases} 0 & \text{si } A(x_i, x_j) < s \\ \frac{A(x_i, x_j) - s}{1 - s} & \text{sinon} \end{cases} \tag{6}$$

où s correspond au seuil de superposition autorisé. Le maximum calculé par point dans l'équation 5 permet d'éviter une augmentation quadratique de l'énergie U_s avec le nombre de points ; ceci peut poser problème et limiter artificiellement le nombre de points si les autres termes d'énergie dépendent linéairement du nombre d'objets.

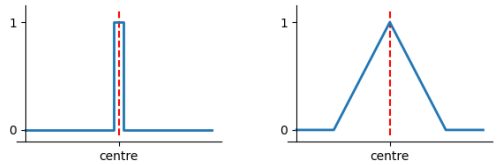
3.2 Contrainte d'alignement

Dans des contextes urbains ou sur des routes, nous constatons que les véhicules sont souvent alignés, qu'ils soient



(a) configurations x_i, x_j, x_k (b) $q_{al}(d_\omega)$

FIGURE 1 – (a) d_ω pour différentes configurations de x_i, x_j, x_k , (b) q en fonction de d_ω



(a) étiquettes sur les centres (b) étiquettes "relaxées"

FIGURE 2 – Relaxation du problème de détection des centres

dans une même file, garés dans un parking ou le long d'une rue. L'énergie d'alignement U_{al} a pour but de favoriser ces configurations. On définit une relation de voisinage comme suit :

$$x_i \sim_v x_j \Leftrightarrow d(x_i, x_j) < d_{max} \tag{7}$$

où d est la distance euclidienne sur $\mathbb{R}^2$. Le but étant de ne pas considérer l'alignement de points trop éloignés. Aussi, pour un triplet de points x_i, x_j, x_k définit-t-on une distance angulaire (illustrée dans la Figure 1a), en notant θ la mesure de l'angle $\widehat{x_j x_i x_k}$:

$$d_\omega(x_i, x_j, x_k) = \min\{|\theta|, |\pi - \theta|\} \tag{8}$$

Cette distance angulaire vaudra 0 que x_i, x_j, x_k soient alignés dans l'ordre j, i, k ou, par exemple, dans l'ordre i, j, k (voir Figure 1a). Ainsi U_{al} s'écrit :

$$U_{al}(\mathbf{x}) = \sum_{1 \leq i \leq n(\mathbf{x})} \min_{\substack{1 \leq j \neq k \leq n(\mathbf{x}) \\ j \neq i, x_j \sim_v x_i \\ k \neq i, x_k \sim_v x_i}} q_{al}(d_\omega(x_i, x_j, x_k)) \tag{9}$$

où

$$q_{al}(x) = -\varpi(\min\{x, d_{\omega_{\max}}\}, d_{\omega_{\max}}) \tag{10}$$

et

$$\varpi(x, x_{\max}) = \frac{1}{x_{\max}^2}\left[\frac{1 + x_{\max}^2}{1 + x^2} - 1\right], \text{pour } x \leq x_{\max}^2 \tag{11}$$

où ϖ est une fonction de qualité introduite par [10]. Comme $\varpi(0, d_{\omega_{\max}}) = 1$ et $\varpi(d_{\omega_{\max}}, d_{\omega_{\max}}) = 0$, si x_i, x_j, x_k sont alignés alors $q_{al}(d_\omega(x_i, x_j, x_k)) = -1$, moins ils le seront, plus la valeur de q_{al} sera proche de 0. La Figure 1b illustre les valeurs de q_{al} en fonction de d_ω. Des exemples de processus ponctuels où l'énergie est composée uniquement des *a priori* sont illustrés dans la Figure 4.

4 CNN pour l'attache aux données

Si les approches par processus ponctuels classiques [3, 7, 10] utilisent des énergies de détection construites manuellement, souvent fondées sur une mesure de contraste [3, 7], lorsque le contraste n'est pas constant (environnement varié, diversité des conditions d'illumination *etc.*) construire une telle énergie de détection devient plus ardu. C'est pourquoi nous nous proposons, dans cet article, d'exploiter la capacité des CNN à apprendre de multiples représentations d'un type d'objet dans des contextes variés pour construire le terme d'attache aux données.

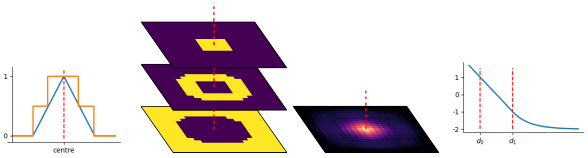
4.1 Détection de centres avec Unet

L'objectif étant d'identifier la probabilité d'un pixel d'appartenir au centre d'un objet, une approche simple consisterait à faire prédire par le réseau de neurones une carte binaire des centres des objets. Cela revient à considérer une tâche de classification de pixel entre deux classes possibles : centre ou non-centre d'objet (Figure 2a). Cependant la surface cumulée des pixels appartenant à un centre de véhicule est négligeable par rapport à la surface de l'image (ordre de grandeur de un pour 10^4). De plus, la position des centres dans la base de données d'apprentissage n'est pas assez précise pour pouvoir compter comme fausse une prédiction de centre avec un décalage de seulement quelques pixels. Il est alors nécessaire de relaxer l'objectif d'apprentissage afin de moins pénaliser les propositions autour des centres étiquetés. Pour chaque pixel i, la carte d'étiquettes y_i à prédire devient : $y_i = \max\left\{1 - \frac{d_{\text{center}}(i)}{d_{\max}}, 0\right\}$ où $d_{\text{center}}(i)$ est la distance de i au centre le plus proche, et $d_{\max}$ un terme de seuillage. Cette relaxation est illustrée dans la Figure 2b. La carte y_i correspond approximativement à une carte de partage des eaux (*Watershed map*), similaire à celle utilisée par [1] pour la segmentation d'instances. Afin de pouvoir équilibrer la fonction de coût (la carte y_i étant majoritairement constituée de valeurs nulles), les valeurs y_i sont discrétisées en n_c classes (Figure 3a), un véhicule produisant le motif illustré dans la Figure 3b. On revient ainsi à une tâche de classification par pixel.

Fonction de coût. Pour chaque pixel i, la fonction de coût correspond à une entropie croisée calculée comme suit :

$$-\sum_{c=1}^{n_c} y_{i,c} \log(p_{i,c}) \tag{12}$$

Avec $y_{i,c}$ l'étiquette binaire valant 1 si i est de classe c et 0 sinon, et $p_{i,c}$ la probabilité estimée que i appartienne à la classe c. Cependant, comme les étiquettes de véhicules sont minoritaires dans l'image par rapport au fond, la fréquence de la classe correspondant au fond (valeurs nulles dans la Figure 2) est bien plus élevée que celle de l'autre



(a) (b) (c) (d)

FIGURE 3 – (a) étiquettes relaxées en bleu et discrétisées en orange autour du centre d'un véhicule (axe pointillé rouge) (b) motif des cartes binaires d'appartenance aux $n_c = 3$ classes de valeurs, utilisé pour construire le filtre $F_{n_c,d_{max}}$ (c) réponse de la convolution par $F_{n_c,d_{max}}$ autour d'un véhicule (d) fonction qualité $q_s(x)$

classe. Ainsi, afin que l'estimateur ne converge pas vers une sortie constante (*ie.* prédire la classe correspondant au fond pour tous les pixels), l'erreur est pondérée par classe inversement à la fréquence de la classe. C'est pourquoi la fonction de coût à minimiser devient :

$$-\sum_{c=1}^{n_c} w_c y_{i,c} \log(p_{i,c}) \tag{13}$$

Où $w_c = 1 - \frac{1}{N}\sum_{i=0}^{N} y_{i,c}$ et N est le nombre de pixels.

Architecture. Le CNN prend en entrée une image x de taille $(h, l, 3)$, et retourne une matrice de taille (h, l, n_c) sortie d'une fonction *softmax* (*ie.* $\sum_{c=0}^{n_c} p_{i,c} = 1 \quad, \forall i$). Le modèle est un Unet comme décrit dans [13], auquel est ajouté des couches de *batch-normalisation* et de *dropout*.

4.2 Énergie d'attache aux données

A partir de la carte de probabilité des classes par pixel $p_{i,c}$ obtenue via le CNN pour une image d'entrée I, il faut construire le terme d'attache aux données du processus ponctuel. Nous cherchons une fonction f telle que $f(i) < 0$ si i est un pixel correspondant au centre d'un véhicule, et $f(i) > 0$ sinon. La carte $p_{i,c}$ est une matrice de taille (h, l, n_c) (h et l sont respectivement la hauteur et largeur de l'image). Connaissant - par construction - le motif généré par un objet sur cette carte (voir Figure 3b), un filtre de corrélation $F_{n_c,d_{max}}$ peut être construit pour mesurer la vraisemblance de la présence d'un objet à un point de l'image (voir Figure 3c). Le filtre $F_{n_c,d_{max}}$ utilisé correspond au motif généré par un objet au centre d'un carré de côté $3d_{max}$ (le motif faisant $2d_{max}$ de côté). La convolution de la sortie du CNN $p_{i,c}$ par le filtre $F_{n_c,d_{max}}$ donne une carte $l(i)$. Ainsi les valeurs de $l(i)$ sont proches de 1 si i correspond au centre du motif recherché sinon $l(i)$ est proche de 0 (voir Figure 3c). La Figure 6 présente un exemple de $p_{i,c}$ et $l(i)$ sur l'ensemble de test.
Ainsi l'énergie de détection s'écrit

$$U_d(\mathbf{x}, I) = \sum_{1 \le i \le n(\mathbf{x})} q_d(l(x_i)) \tag{14}$$

où

$$q_d(x) = \begin{cases} \frac{-2(x-d_0)}{d_1-d_0} + 1 & \text{si } x < d_1 \\ \exp\left(\frac{-2(x-d_0)}{d_1-d_0} + 2\right) - 2 & \text{sinon} \end{cases} \tag{15}$$

où $l(x_i)$ est la carte de détection inférée à partir de l'image I évaluée aux coordonnées du point x_i, et q_d assure une réponse linéaire de d_0 à d_1 (réponses négative et positive moyennes du filtre) de 1 à -1 puis au delà de d_1 la réponse est seuillée à -2 (voir Figure 3d).

Énergie totale. Ainsi l'énergie totale pour une configuration $\mathbf{x}$ s'écrit :

$$U_{tot}(\mathbf{x}, I) = \gamma_d U_d(\mathbf{x}, I) + \gamma_{al} U_{al}(\mathbf{x}) + \gamma_s U_s(\mathbf{x}) \tag{16}$$

où $\gamma_{al}, \gamma_s, \gamma_d$ sont des paramètres de pondération des énergies, permettant, entre autres, d'augmenter ou diminuer l'importance des *a priori* sur la détection.

5 Estimation des paramètres et optimisation

5.1 Entrainement du réseau de neurones

À la résolution spatiale utilisée, beaucoup d'objets peuvent avoir une apparence proche d'un véhicule. Cela donne lieu à un fort taux de faux positifs. Afin de minimiser ce problème, nous appliquons une approche de détection en "cascade" inspiré de diverses méthodes comme celle proposée par Viola et Jones [16]. Un premier modèle léger (Unet avec seulement une séquence de *pooling/up-convolution* contre 4 normalement) est entrainé sur des images 128x128 à prédire une carte binaire dilatée des centres. Un second modèle complet est entrainé sur de petites fenêtres (64x64 pixels), centrées sur les réponses positives du premier réseau (considérant comme positive toute probabilité de présence d'un objet au dessus de 5%).

5.2 Selection des paramètres d'énergie

En se basant sur les travaux de Yu et Medioni [17] et Craciun [2], la vérité terrain labellisée est utilisée pour inférer les paramètres de poids des énergies $\gamma_{al}, \gamma_s, \gamma_d$ introduits dans l'équation 16. Pour la configuration de la vérité terrain $\mathbf{x}^*$, nous pouvons calculer les valeurs des énergies U_{al}, U_s, U_d qui forment ainsi un vecteur d'énergies. Des configurations perturbées $\mathbf{x}_i$ - considérées non valides - sont générées en ajoutant, supprimant ou déplaçant des points de $\mathbf{x}^*$. Les vecteurs d'énergies des $\mathbf{x}_i$ sont aussi calculés. Dans cet espace vectoriel des énergies, les paramètres $\gamma_{al}, \gamma_s, \gamma_d$ sont obtenus en trouvant la meilleure séparation linéaire entre $\mathbf{x}^*$ (configuration valide) et les $\mathbf{x}_i$ (configurations non valides). Ces paramètres sont obtenus grâce à un séparateur à vaste marge (*Support-Vector Machine*).

5.3 Simulation d'un processus ponctuel

Pour une image donnée, la configuration optimale est obtenue par la simulation du processus ponctuel avec recuit

simulé. Le processus ponctuel est simulé via une chaîne de Markov Monte Carlo à sauts réversibles (*Reversible Jump Markov Chain Monte Carlo*) (RJMCMC) [5]. Cela consiste en la simulation d'une chaine de Markov $\mathbf{x}_t$ discrète qui effectue des sauts entre les Ω_i. La mesure π est par construction la mesure stationnaire de $\mathbf{x}_t$. A chaque étape, une transition de l'état actuel $\mathbf{x}$ vers un autre état $\mathbf{x}'$ est proposée par un noyau de proposition $Q(S \rightarrow \cdot)$, la transition est acceptée avec une probabilité $\alpha(\mathbf{x}, \mathbf{x}')$ donnée par le ratio de Green. Ce ratio d'acceptation est calculé pour vérifier la condition de réversibilité, condition nécessaire à la convergence vers π. Cette condition s'écrit :

$$\int_A \int_B \pi(d\mathbf{x}) P(\mathbf{x}, d\mathbf{x}') = \int_B \int_A \pi(d\mathbf{x}') P(\mathbf{x}', d\mathbf{x}) \quad (17)$$

où A et B sont deux ensembles de la tribu associée à Ω, et P la matrice de transition de $\mathbf{x}_t$. En supposant que $\pi(\cdot)Q(S \rightarrow \cdot)$ a une densité finie $\mathcal{D}$ par rapport à une mesure symétrique ψ sur $\Omega \times \Omega$, l'équation 17 est vérifiée si :

$$\alpha(\mathbf{x}, \mathbf{x}')\mathcal{D}(\mathbf{x}, \mathbf{x}') = \alpha(\mathbf{x}', \mathbf{x})\mathcal{D}(\mathbf{x}', \mathbf{x}) \quad (18)$$

Ainsi nous prenons $\alpha(\mathbf{x}, \mathbf{x}') = \min\{1, R(\mathbf{x}, \mathbf{x}')\}$ où R est le ratio de Green :

$$R(\mathbf{x}, \mathbf{x}') = \frac{\mathcal{D}(\mathbf{x}', \mathbf{x})}{\mathcal{D}(\mathbf{x}, \mathbf{x}')} \quad (19)$$

Green [5] décompose le noyau Q en plusieurs sous-noyaux q_i, chacun correspondant à un mouvement simple et réversible. Les noyaux de naissance et de mort sont suffisants pour permettre la convergence vers π, cependant l'ajout de noyaux de translation par exemple peut accélérer la convergence, car il est plus probable d'accepter un seul mouvement de translation qu'une mort, puis une naissance.
Contrairement aux approches classiques comme [3, 7, 15], dans notre cas d'application, nous n'utilisons pas de noyaux de changement d'échelle (*scaling*) ou de rotation, car notre modèle utilise des points non marqués.

5.4 Noyaux de proposition

Naissance et mort. Les noyaux de naissance et de mort permettent d'effectuer des transitions entre des espaces Ω_i de dimensions différentes. Cette perturbation choisit d'abord avec une probabilité p_b d'ajouter un élément ou p_d de retirer un élément. Pour une naissance, un nouvel élément x est choisi selon $\frac{\nu(\cdot)}{\nu(S)}$ où $\nu(\cdot)$ est la mesure associée au processus de Poisson sous-jacent. Le ratio de Green obtenu est donc :

$$R(\mathbf{x}, \mathbf{x} \cup x) = \frac{h(\mathbf{x} \cup x)}{h(\mathbf{x})} \frac{p_d}{p_b} \frac{\nu(S)}{(n(\mathbf{x}) + 1)} \quad (20)$$

Pour rappel, h est défini dans l'équation 3 comme une densité de Gibbs non-normalisée : $h(X) = \exp(-U(\mathbf{x}))$ et $n(\mathbf{x})$ est le nombre de points dans $\mathbf{x}$.
Il est à noter que, afin d'accélérer la convergence, le processus de Poisson sous-jacent est choisi avec une densité non uniforme, pour tout $B \subset S$, comme $\nu(B) = \lambda \frac{\sum_{i \in B} l(i)}{\sum_{i \in S} l(i)}$, où $l(i)$ est la carte de détection obtenue en 4.2. Ainsi il sera plus probable de proposer des points sur les pixels i où $l(i)$ est grand, et donc la détection plus probable. Dans le ratio de Green (équation 20) nous avons $\nu(S) = \lambda$.
Pour la mort, un point x à retirer est choisi uniformément parmi les $n(\mathbf{x})$. Le ratio de Green est défini similairement :

$$R(\mathbf{x}, \mathbf{x} \backslash x) = \frac{h(\mathbf{x} \backslash x)}{h(\mathbf{x})} \frac{p_b}{p_d} \frac{n(\mathbf{x})}{\nu(S)} \quad (21)$$

Translations. Le sous-noyau de translation choisit uniformément un point x dans $\mathbf{x}$ et propose une translation de $\delta_x \in [-\delta_{max}, +\delta_{max}]^2$ (où δ_{max} est un paramètre du modèle) selon une densité $p_x(\cdot)$. La densité p_x est construite pour favoriser le déplacement vers les emplacements où la probabilité de détection est la plus forte :

$$p_x(\delta_x) = \frac{l(x + \delta_x)}{\sum_{\delta \in [-\delta_{max}, \delta_{max}]^2} l(x + \delta)} \quad (22)$$

Ainsi, en notant $x_1 = x$ et $x_2 = x + \delta_x$ le ratio de Green est le suivant :

$$R(\mathbf{x}, (\mathbf{x} \backslash x_1) \cup x_2) = \frac{h((\mathbf{x} \backslash x_1) \cup x_2)}{h(\mathbf{x})} \frac{p_{x_2}(-\delta_x)}{p_{x_1}(\delta_x)} \quad (23)$$

5.5 Recuit simulé

Une fois la carte de détection obtenue par inférence du CNN, l'énergie de détection est pré-calculée en tout point, et le processus ponctuel peut alors être simulé. Afin de faire converger la simulation du processus ponctuel vers une configuration optimale $\hat{\mathbf{x}}$, nous simulons une succession de distribution π_T définies par la densité $h^{1/T}$, avec T diminuant progressivement vers 0. [14] montre la convergence pour une décroissance logarithmique de T. En pratique, cela est bien trop lent et nous utilisons une décroissance géométrique pour réduire le temps de calcul.

6 Résultats expérimentaux

6.1 Données : construction de COWC50

Pour entrainer et tester ce modèle, nous avons utilisé une version modifiée du jeu de données *Cars Overhead With Context* (COWC) [1] [9] que nous appelons COWC50. COWC contient, pour plusieurs villes, des images aériennes allant de 6000x6000 à 18000x18000 pixels, pour une résolution spatiale de 15 centimètres par pixel (cm/px). Pour chaque image, les voitures sont étiquetées en leur centre. Ce jeu de données est diversifié, il contient aussi bien des vues de champs que des environnements urbains denses.
Comme notre objectif est de développer des modèles de détection dans des images satellitaires, nous choisissons de réduire la résolution spatiale des images afin de simuler les images obtenues par un satellite haute résolution. C'est

1. https://gdo152.llnl.gov/cowc/

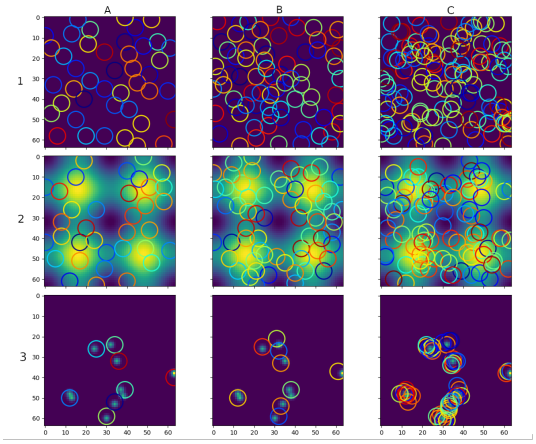


FIGURE 4 – Effets de la température (A : basse ($T = 10^{-4}$), B : moyenne $T = 1$, C : haute $T = 10^4$), pour plusieurs cartes d'intensité (1 : uniforme, 2 : cosinus 2D, 3 : petits "blobs"), sur les processus ponctuels simulés en utilisant uniquement les *a priori*

pourquoi nous nous plaçons à une résolution de 50 cm/px (Figure 5). Nous appliquons un sous-échantillonnage avec un filtre d'*antialiasing* afin d'éviter le crénelage et simuler l'acquisition avec une optique adaptée au 50 cm/px. Une telle réduction de résolution implique une augmentation de la difficulté de détection, car des objets proches peuvent devenir difficiles à dissocier si ces derniers ne font que quelques pixels de large.

On utilise uniquement les données couleurs de COWC. Pour chaque ville, un tiers est attribué à l'ensemble de test et les deux autres tiers sont utilisés pour l'apprentissage.

FIGURE 5 – Exemples d'images de COWC50, pour deux fenêtres de 512x512 à 50 cm/px échantillonnées aléatoirement.

6.2 Détection d'objets d'intérêt

Inférence du CNN. Les images (à 50 cm/px) sont passées en entrée du Unet après une normalisation locale (Figure 6a). Nous obtenons la carte de probabilité de classes $p_{i,c}$ (Figure 6b) dont est extraite la carte d'énergie de détection $l(i)$ (Figure 6d) par convolution avec le filtre $F_{n_c,d_{max}}$ décrit en 4.2.

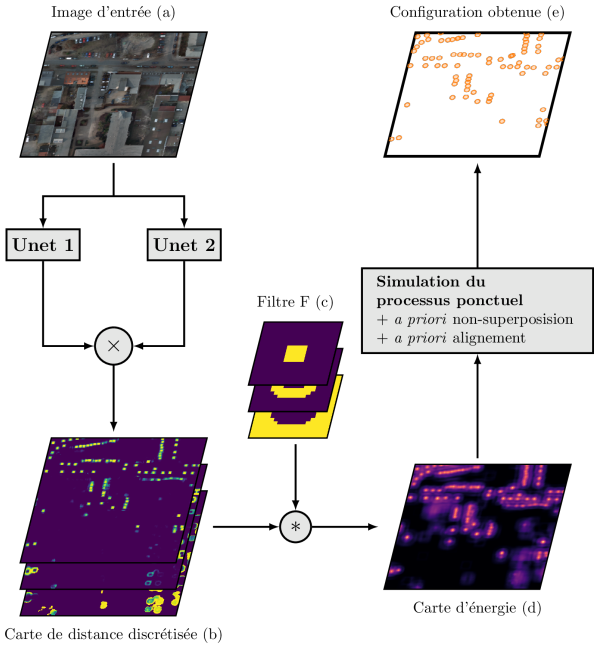


FIGURE 6 – Pipeline global ; la sortie du Unet (b) représente l'appartenance des pixels aux $n_c = 3$ classes de distance $p_{i,c}$.(d) correspond à la carte de détection $l(i)$

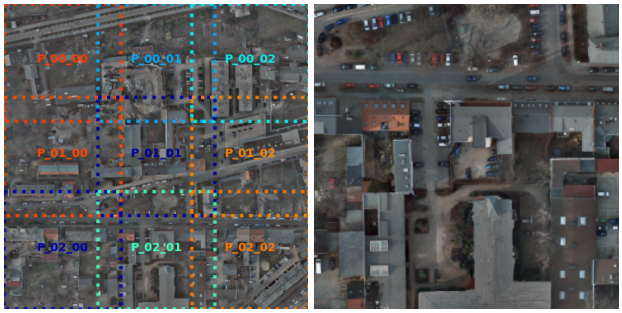
Simulation du processus ponctuel. La détection est effectuée sur des fenêtres de 256x256 pixels des images à 50 cm/px. Les fenêtres sont disposées de sorte à paver l'image comme illustré dans la Figure 7.

On simule le processus ponctuel via l'algorithme RJMCMC (voir partie 5.3) sur 20000 itérations avec un recuit simulé géométrique. Une fois la convergence atteinte, la configuration de points obtenue $\hat{\mathbf{x}}$ correspond à la prédiction des positions des voitures dans l'image. Le résultat sur une fenêtre est illustré dans la Figure 8.

6.3 Evaluation qualitative et quantitative des résultats

Afin de mesurer la pertinence d'une configuration de points estimée $\hat{\mathbf{x}}$, similairement à [9], nous faisons correspondre les voitures étiquetées du jeu de données à chaque point de $\hat{\mathbf{x}}$ seulement s'ils sont à moins d'un rayon r de distance (où r est le rayon choisi dans la simulation en 3.1 pour correspondre à une demi largeur de voiture environ).

Ainsi, une voiture est considérée comme une *détection* si elle correspond à au moins un point de $\hat{\mathbf{x}}$. Une voiture est un *faux négatif* si elle ne correspond à aucun point de $\hat{\mathbf{x}}$ (aucun point n'est à moins de r de distance). Un point de $\hat{\mathbf{x}}$ est un *faux positif* s'il n'y a aucune voiture lui correspondant. Une *double détection* advient si deux points ou plus correspondent à une unique voiture ; dans ce cas on compte un *faux positif* par détection supplémentaire. Une *fusion de détections* advient si un point correspond à deux voitures ou plus ; on compte un *faux négatif* par voiture non détectée (si fusion de trois voitures, on comptera deux *faux*



(a) image avec fenêtres (b) fenêtre P_02_01

FIGURE 7 – Exemple de fenêtres d'une image

(a) vérité terrain (b) détection

FIGURE 8 – Détection sur la fenêtre P_02_01 de COWC50 : (a) vérité terrain, (jaune : véhicules détectés, orange : faux négatifs) (b) détection $\hat{\mathbf{x}}$ (vert : vrais positifs, rouge : faux positifs)

(a) processus ponctuel (b) supp. des non maxima

FIGURE 9 – (a) résultat de détection avec les processus ponctuels, (b) résultats obtenus en appliquant une suppression des non maxima (*non-maximum supression*), (vert : vrais positifs, rouge : faux positifs, orange : faux négatifs)

nb. v.	VP	FP	FN	Précision	Rappel	F
74	48	17	12	73.43%	63.51%	68.12%

TABLE 1 – Résultats de détection sur la fenêtre P_02_01. *nb. v.* : nombre de véhicules dans la fenêtre, *VP*, *FP* et *FN* nombre de vrais positifs, faux positifs et faux négatifs. *F* : F-Score (moyenne harmonique de la précision et du rappel)

image	nb. v.	Précision	Rappel	F
(1) Potsdam	425	74.27%	66.24%	69.97%
(2) Potsdam	262	56.20%	48.03%	51.76%
(3) Potsdam	110	75.75%	68.20%	71.57%
(4) Selwyn	121	86.60%	49.70%	75.11%
(5) Toronto	3969	86.69%	63.17%	74.43%

TABLE 2 – Valeurs moyennes de précision et rappel sur les parcelles de chaque image de l'ensemble de test de COWC50. (1) top_potsdam_6_9_RGB, (2) top_potsdam_6_8_RGB, (3) top_potsdam_6_7_RGB, (4) Selwyn_BX22_Tile_RIGHT_15cm_0003, (5) 03747

négatifs par exemple).
Un exemple de précision, rappel et F-score calculés sur la fenêtre P_02_01, est présenté dans la Table 1.
Pour chaque image de l'ensemble de test, les métriques sont calculées par fenêtre, la Table 2 présente les moyennes des métriques pour toutes les fenêtres des images.
De plus, nous comparons la détection avec les processus ponctuels avec une méthodes similaire à celle utilisée dans [9] pour la détection d'objets, en utilisant la sortie du CNN et une suppression des non maxima (*non-maximum supression*). Cette comparaison est illustrée dans la Figure 9.

Temps de calcul. L'apprentissage du CNN sur les 6000 fenêtres (512x512) de l'ensemble d'apprentissage dure de l'ordre de 24 heures pour 64 *epochs* sur une carte *Nvidia Quadro RTX8000* en utilisant la librairie Python *flax* [6]. L'inférence du CNN sur un processeur *Intel Core i7 9th Gen.* pour une seule image de taille 5400x5400 (image de 18000 de coté sous échantillonnée de 15 cm/px à 50 cm/px) prend de l'ordre de 15 secondes. La simulation du processus ponctuel par RJMCMC de 20000 itérations sur une fenêtre de 256 par 256 pixels dure 30 secondes environ. Elle peut être effectuée en parallèle sur plusieurs fenêtres distinctes à la fois.

7 Conclusion

Si nos résultats sont pour l'heure sous-optimaux, du fait entre autres du grand nombre de faux positifs, cette approche utilisant processus ponctuels et CNN se montre encourageante. En effet, les processus ponctuels nous permettent d'incorporer des *a priori* dans des cas combinant petite taille des objets d'intérêt et forte densité spatiale. Nous avons montré que l'on peut utiliser les processus ponctuels combinés aux CNN pour la télédétection, afin de pouvoir, dans un futur proche, utiliser le plein potentiel

des *a priori* dans des situations où ils seront plus prévalents (par exemple pour du suivi d'objets dans des séquences vidéos). Aussi sera-t-il intéressant d'exploiter les processus ponctuels marqués pour pouvoir inférer des informations supplémentaires via notre modèle (par exemple l'orientation et la taille des objets).

Remerciements

Les auteurs sont reconnaissants envers l'infrastructure OPAL de l'Université Côte d'Azur (UCA) pour avoir fourni les ressources de calcul nécessaire à ce travail de recherche, ainsi qu'envers BPI France pour le soutien financier dans le cadre du contrat LiChiE.

Références

[1] M. BAI et R. URTASUN. « Deep watershed transform for instance segmentation ». *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, p. 5221-5229.

[2] P. CRĂCIUN. « Stochastic Geometry for Automatic Multiple Object Detection and Tracking in Remotely Sensed High Resolution Image Sequences ». Thèse de doct. Université Nice Sophia Antipolis, 25 nov. 2015. URL : https://tel.archives-ouvertes.fr/tel-01235255.

[3] S. DESCAMPS et al. « Automatic Flamingo Detection Using a Multiple Birth and Death Process ». *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*. Mar. 2008, p. 1113-1116. DOI : 10/fb794f.

[4] X. DESCOMBES. « Multiple Objects Detection in Biological Images Using a Marked Point Process Framework ». *Methods*. Image Processing for Biologists 115 (fév. 2017), p. 2-8. DOI : 10.1016/j.ymeth.2016.09.009.

[5] P. J. GREEN et D. I. HASTIE. « Reversible Jump MCMC ». *Genetics* 155.3 (2009), p. 1391-1403. URL : http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.408.3500&rep=rep1&type=pdf.

[6] J. HEEK et al. *Flax : A Neural Network Library and Ecosystem for JAX*. Version 0.3.3. 2020. URL : http://github.com/google/flax.

[7] C. LACOSTE, X. DESCOMBES et J. ZERUBIA. « Point Processes for Unsupervised Line Network Extraction in Remote Sensing ». *IEEE Transactions on Pattern Analysis and Machine Intelligence* 27.10 (oct. 2005), p. 1568-1579. DOI : 10.1109/TPAMI.2005.206.

[8] T.-Y. LIN et al. « Focal Loss for Dense Object Detection ». *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017.

[9] T. N. MUNDHENK et al. « A Large Contextual Dataset for Classification, Detection and Counting of Cars with Deep Learning ». *Computer Vision – ECCV 2016*. Sous la dir. de B. LEIBE et al. Lecture Notes in Computer Science. 2016, p. 785-800. DOI : 10.1007/978-3-319-46487-9_48.

[10] M. ORTNER, X. DESCOMBES et J. ZERUBIA. « A Marked Point Process of Rectangles and Segments for Automatic Analysis of Digital Elevation Models ». *IEEE Transactions on Pattern Analysis and Machine Intelligence* 30.1 (jan. 2008), p. 105-119. DOI : 10.1109/TPAMI.2007.1159.

[11] J. REDMON et al. « You Only Look Once : Unified, Real-Time Object Detection ». *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Las Vegas, NV, USA, juin 2016, p. 779-788. DOI : 10.1109/CVPR.2016.91.

[12] S. REN et al. « Faster R-CNN : Towards Real-Time Object Detection with Region Proposal Networks ». *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.6 (juin 2017), p. 1137-1149. DOI : 10.1109/TPAMI.2016.2577031.

[13] O. RONNEBERGER, P. FISCHER et T. BROX. « U-Net : Convolutional Networks for Biomedical Image Segmentation ». *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*. Sous la dir. de N. NAVAB et al. Lecture Notes in Computer Science. 2015, p. 234-241. DOI : 10/gcgk7j.

[14] M. VAN LIESHOUT. *Stochastic Annealing for Nearest Neighbour Point Processes with Application to Object Recognition*. 1993. URL : https://ir.cwi.nl/pub/5291/05291D.pdf.

[15] Y. VERDIÉ et F. LAFARGE. « Detecting Parametric Objects in Large Scenes by Monte Carlo Sampling ». *International Journal of Computer Vision* 106.1 (jan. 2014), p. 57-75. DOI : 10.1007/s11263-013-0641-0.

[16] Y.-Q. WANG. « An Analysis of the Viola-Jones Face Detection Algorithm ». *Image Processing On Line* 4 (juin 2014), p. 128-148. DOI : 10/ghrq4p.

[17] Q. YU et G. MEDIONI. « Multiple-Target Tracking by Spatiotemporal Monte Carlo Markov Chain Data Association ». *IEEE Transactions on Pattern Analysis and Machine Intelligence* 31.12 (déc. 2009), p. 2196-2210. DOI : 10.1109/TPAMI.2008.253.

[18] Z.-Q. ZHAO et al. « Object detection with deep learning : A review ». *IEEE transactions on neural networks and learning systems* 30.11 (2019), p. 3212-3232. DOI : 10.1109/TNNLS.2018.2876865.