Comparaison de méthodes d'extraction fond/forme pour des scènes de circulation routière

N. Tronson[1] Y. Goyat[1] D. Gruyer[2]

[1] LCPC (Laboratoire Central des Ponts et Chaussées)
Route de Bouaye, BP 4129, 44341 Bouguenais – FRANCE
nicolas.tronson@lcpc.fr, yann.goyat@lcpc.fr

[2] LIVIC (Laboratoire sur les Interactions Véhicules-Infrastructure-Conducteurs)
14, route de la Minière - Bâtiment 824 - Satory 78000 Versailles – FRANCE
dominique.gruyer@inrets.fr

Résumé

Cet article compare trois méthodes pour isoler les objets en mouvement, souvent appelées méthodes d'extraction fond/forme. Nous nous penchons spécifiquement sur les problématiques liées aux scènes routières, dont les enjeux sont très importants en terme de surveillance et d'analyse du trafic. Pour tester indépendamment et avec qualité les influences d'éléments dégradants, nous proposons d'utiliser un logiciel permettant de générer des scènes virtuelles. L'avantage est d'obtenir des vérités terrain associées afin de mesurer précisément la qualité de l'extraction.

Mots clefs

Extraction fond/forme, vérités terrain, simulation, modèle virtuel, comparaison.

1 Introduction

Nous présenterons ici l'étude de trois méthodes utilisées pour l'extraction fond/forme sur une scène de circulation routière, selon différents réglages de paramètres propres à chaque méthode. Des scènes virtuelles permettant de créer la vérité terrain correspondante sont générées. Elles seront ensuite comparées avec les images extraites par chacun des algorithmes, qui pourront donc être notées à l'aide deux critères de qualité.

Nous allons tout d'abord décrire les différents problèmes typiques à une scène routière, puis, présenter une méthode très classique de la littérature et deux méthodes plus originales. Puis seront présentés, le principe utilisé pour générer les vidéos de test, les vidéos, et les critère de mesure utilisés. Enfin, les les tests avec la mesure de l'extraction permettront de comparer ces trois méthodes.

2 Problématiques de l'extraction dans une scène routière

Dans une scène routière, l'acquisition se fait à partir d'une caméra fixe et il existe de nombreuses causes pouvant altérer la qualité de l'extraction :

Le bruit sur une image se caractérise par des valeurs de pixel changeant légèrement. Étant donné que les méthodes d'extraction se basent sur la couleur du pixel, elles doivent donc avoir une certaine tolérance pour ne pas être trop sensible au bruit.

Le mouvement d'objets tel que les branches d'arbre est assez courant. Le but est de détecter ces objets comme appartenant au fond de l'image. Ce mouvement est de façon générale assez répétitif. La méthode d'extraction doit donc être capable d'apprendre les valeurs des pixels les plus récurrents sur un laps de temps à définir, et de considérer que ces valeurs correspondent au fond.

Le changement de luminosité est assez courant quand on observe des véhicules sur la route, ceci peut être lié à un passage de nuage. Les couleurs sur l'image varient donc et l'algorithme peut penser qu'il s'agit d'un objet. L'idéal serait d'être assez peu sensible au changement de luminosité, ou d'être capable de s'adapter rapidement pour limiter les mauvaises détections.

Les scènes sombres ont des niveaux de couleur assez faibles, les différences entre les couleurs sont moins marquées. Il peut donc être plus difficile de détecter le fond de la forme dans ce type de scène.

Les véhicules ayant une couleur proche du fond peuvent être plus difficilement détectés étant donné que les trois algorithmes se basent sur la couleur du pixel pour identifier le fond de la forme.

Les ombres sont des zones qui suivent les véhicules avec des changements de valeur de pixel par rapport au fond. Il est donc assez logique que les algorithmes puissent classer par erreur ces zones comme forme.

La densité de circulation peut poser problème. En effet, les méthodes de détection doivent apprendre le fond, et pour cela, elles considèrent que les valeurs des pixels les plus fréquentes sont des valeurs de fond. Ceci peut poser problème dans des situations urbaines à circulation très dense où le fond n'est pas souvent visible en raison du flux

dense et continu de véhicules.

3 Présentation des méthodes

Nous comparons ici l'extraction fond/forme à l'aide des trois méthodes :
– Mixture de gaussiennes
– Codebook 2 layers
– Vumètre
Le but est de comparer la qualité d'extraction avec chacune des méthodes sur une même scène, et avec une analyse de l'influence des causes altérant la qualité d'extraction décrites précédemment (voir section 2).

3.1 Mixture de gaussiennes (MOG)

Dans cette approche, chaque pixel est modélisé par une mixture de N gaussiennes, $2 \leq N \leq 5$ [1]. Pour $n = 1, \ldots, N$, un élément de la mixture de gaussiennes est représenté par une moyenne $\mu_{\mathbf{n}}$, un écart type σ_n, et un poids α_n ($\sum_n \alpha_n = 1$). On peut remarquer que σ_n est réduit à un scalaire, comme discuté dans [1].
Pour une nouvelle image traitée, la mixture de gaussiennes (pour tous les pixels) est mise à jour pour expliquer correctement les couleurs affichées par chaque pixel. Pour faire ceci, à un instant t, on considère que le modèle $\mathbf{M_t}$ généré pour chaque pixel à partir des mesures $\{\mathbf{Z_0}, \mathbf{Z_1}, ..., \mathbf{Z_{t-1}}\}$ est correct. La vraisemblance pour qu'un pixel appartienne au fond est :

$$P(\mathbf{Z_t}|\mathbf{M_t}) = \sum_{n=1}^{n=N} \alpha_n \mathcal{N}(\mu_{\mathbf{n}}, \mathbf{\Sigma_n}) \tag{1}$$

$$\mathcal{N}(\mu_{\mathbf{n}}, \mathbf{\Sigma_n}) = \frac{1}{(2\pi)^{d/2}|\mathbf{\Sigma_n}|^{1/2}} e^{-\frac{1}{2}(\mathbf{Z_t}-\mu_{\mathbf{n}})^T \mathbf{\Sigma}^{-1}(Z_t-\mu_{\mathbf{n}})} \tag{2}$$

avec d la dimension de l'espace de couleurs de la mesure $\mathbf{Z_t}$.
Pour mettre à jour le modèle, on associe d'abord la mesure $\mathbf{Z_t}$ à une gaussienne n' si

$$||\mathbf{Z_t} - \mu_{\mathbf{n}}|| < K\sigma_n \tag{3}$$

où K vaut 2 ou 3. L'opérateur $<$ est vrai si toutes les composantes du vecteur à gauche sont inférieures à $K\sigma_n$.
Cette mesure représente le fond si la gaussienne n' explique le fond de la scène. En fait, le poids $\alpha_{n'}$ est élevé. Cette gaussienne est alors mise à jour :

$$\alpha'_n \leftarrow (1-\delta)\alpha'_n + \delta \tag{4}$$
$$\mu'_{\mathbf{n}} \leftarrow (1-\delta)\mu'_{\mathbf{n}} + \delta\mathbf{Z_t} \tag{5}$$
$$\sigma^{2'}_n \leftarrow (1-\delta)\sigma^{2'}_n + \delta(\mathbf{Z_t} - \mu'_{\mathbf{n}})^T(\mathbf{Z_t} - \mu'_{\mathbf{n}}) \tag{6}$$

avec δ le coefficient d'apprentissage. Il représente la vitesse d'adaptation du modèle. Pour toutes les autres gaussiennes $n \neq n'$, la moyenne et la variance ne sont pas modifiées, mais :

$$\alpha_n \leftarrow (1-\delta)\alpha_n \tag{7}$$

Si le test 3 échoue, le pixel est associé au 1er plan. La gaussienne ayant le plus petit poids est réinitialisée avec la mesure actuelle :

$$\alpha_n = \delta \tag{8}$$
$$\mu_{\mathbf{n}} = \mathbf{Z_t} \tag{9}$$
$$\sigma_n^2 = \bar{\sigma}^2 \tag{10}$$

avec $\bar{\sigma}^2$ une variance élevée. Ces affectations sont aussi appliquées pour l'initialisation de la mixture.

3.2 Codebook 2 layers (CB2)

Cette méthode [4], est très largement inspirée du codebook [2]. Mais elle en diffère, en utilisant deux codebooks, bibliothèques de données pour chaque pixel contenant des informations pour modéliser le fond. Ceci a été réalisé de manière à pouvoir retenir des valeurs de pixels qui ont appartenu au fond, mais qui pourraient redevenir du fond, c'est typiquement le cas avec les mouvements de branches d'arbre.
Chaque codebook contient des éléments appelés codeword (CW) pour modéliser le fond de l'image, chacun des CW contient ces informations :
– v_i : valeur moyenne du pixel (R,V,B)
– I_{max} : limite maximale d'intensité du CW
– I_{min} : limite minimale d'intensité du CW
– f : fréquence du CW (nombre d'occurrences)
– λ : nombre maximal d'images où le CW ne correspond à aucun pixel
– p : première occurrence du CW
– q : dernière occurrence du CW
Le principe est le même qu'avec le codebook simple, mais avec deux codebooks par pixel : un principal appelé M, et un secondaire appelé H.
Le traitement se fait en 2 phases : une phase d'apprentissage qui sert à créer les codebooks principaux initiaux, et une phase de soustraction pour extraire le fond de la forme.
Pour chaque nouveau pixel $x_t = (R, V, B)$, son intensité I_t est calculée par $I_t = \sqrt{R^2 + V^2 + B^2}$
La distorsion de couleur δ entre ce pixel $x_t = (R, G, B)$ et un codeword c_i avec $v_i = (\bar{R}_i, \bar{V}_i, \bar{B}_i)$ peut être calculé par :

$$\langle x_t, v_i \rangle^2 = (\bar{R}_i R + \bar{V}_i V + \bar{B}_i B)^2 \tag{11}$$
$$\|v_i\|^2 = \bar{R}_i^2 + \bar{V}_i^2 + \bar{B}_i^2 \tag{12}$$
$$\|x_t\|^2 = R^2 + V^2 + B^2 \tag{13}$$
$$colordist(x_t, v_i) = \delta = \sqrt{\|x_t\|^2 - \frac{\langle x_t, v_i \rangle^2}{\|v_i\|^2}} \tag{14}$$

Un pixel x_t avec une intensité I_t correspond à un codeword c_i avec une valeur de pixel v_i et I_{min}, I_{max} si I_t est dans l'intervalle $[I_{min}, I_{max}]$ et la distorsion de couleur δ respecte $\delta < \epsilon$
En phase d'apprentissage, seule la couche M est construite, H reste vide. Pour un nouveau pixel x_t, on cherche un CW dans M correspondant à x_t. Si on en trouve un, il est mis à

jour avec x_t, sinon un nouveau CW est crée à partir de la valeur de x_t.
Un nouveau codeword est crée avec un pixel x_t de la façon suivante :

$$v_i \leftarrow (R, V, B) \tag{15}$$
$$I_{min} \leftarrow max\{0, I_t - \alpha\} \tag{16}$$
$$I_{max} \leftarrow min\{255, I_t + \alpha\} \tag{17}$$
$$f \leftarrow 1;\ \lambda \leftarrow t-1;\ p \leftarrow t;\ q \leftarrow t \tag{18}$$

avec α une valeur représentant une tolérance d'intensité.
Pendant la phase d'apprentissage, un codeword est mis à jour par un pixel x_t comme ceci :

$$\bar{R} \leftarrow \frac{\bar{R} \times f + R}{f+1} \text{ (de même pour V et B)} \tag{19}$$
$$I_{min} \leftarrow \frac{I - \alpha + f \times I_{min}}{f+1} \tag{20}$$
$$I_{max} \leftarrow \frac{I + \alpha + f \times I_{max}}{f+1} \tag{21}$$
$$f \leftarrow f+1;\ \lambda \leftarrow max\{\lambda, t-q\};\ p \leftarrow p;\ q \leftarrow t \tag{22}$$

En phase de soustraction, pour un nouveau pixel x_t, on cherche un CW dans M correspondant à x_t. Si on en trouve un, il est mis à jour avec x_t et le pixel est considéré comme appartenant au fond. Sinon, on cherche un CW correspondant à ce pixel dans H, si on un trouve un, on le met à jour avec x_t, sinon, on en crée un nouveau dans H avec la valeur de x_t.
Un CW est mis à jour comme précédemment dans la phase d'apprentissage à l'exception de I_{min} et I_{max} qui sont mis à jour de la façon suivante, avec β un coefficient pour changer la vitesse d'adaptation :

$$I_{min} \leftarrow (1-\beta)(I_t - \alpha) + \beta \cdot I_{min} \tag{23}$$
$$I_{max} \leftarrow (1-\beta)(I_t + \alpha) + \beta \cdot I_{max} \tag{24}$$

Ensuite, les modèles M et H sont affinés avec ces règles :
– Supprimer les CWs de H ayant $\lambda > T_H$
– Déplacer les CWs restant plus de T_{add} dans H vers M
– Supprimer les CWs de M n'apparaissant pas plus longtemps que T_{delete}

3.3 Vumètre (VUM)

Le vumètre par Goyat *et al.* [3] est un modèle non-paramétrique, basé sur une estimation discrète de la probabilité de distribution. Il s'agit d'une approche probabiliste pour définir le modèle du fond. Soit I_t une image à l'instant t, $y_t(u)$ donne les valeurs RVB du pixel u. Un pixel peut prendre deux états, (ω_1) s'il appartient au fond, (ω_2) s'il appartient au 1er plan. Cette méthode essaye d'estimer $p(\omega_1 \mid y_t(u))$. Avec 3 composantes de couleur i (Rouge, Vert, Bleu), la fonction de densité de probabilité peut être approximée par :

$$p(\omega_1 \mid y_t(u)) = \prod_{i=1}^{3} p(\omega_1 \mid y_t^i(u)) \tag{25}$$

avec

$$p(\omega_1 \mid y_t^i(u)) \approx K_i \sum_{j=1}^{N} \pi_t^{ij} \delta(b_t^i(u) - j) \tag{26}$$

où δ est le symbole de Kronecker, $b_t(u)$ donne le vecteur d'index de la classe associée à $y_t(u)$, j est un index de classe, et K_i est une constante de normalisation permettant de garder à chaque instant :

$$\sum_{j=1}^{N} \pi_t^{ij} = 1 \tag{27}$$

π_t^{ij} est une fonction de masse discrète représentée par une classe.
A la première image $(t = 0)$, les valeurs des classes sont mises à $\pi_0^{ij} = 1/N$ pour garder une somme à 1 comme dans l'équation 27. A chaque nouveau pixel, sa valeur correspond à une classe π_t^{ij}, son niveau est mis à jour de cette façon :

$$\pi_{t+1}^{ij} = \pi_t^{ij} + \alpha \cdot \delta(b_{t+1}^i(u) - j) \tag{28}$$

Après un certain nombre d'images, les classes modélisant le fond ont une valeur élevée. Pour pouvoir décider à quel moment un pixel appartient au fond ou non, un seuil T (voir figure 1) est défini.

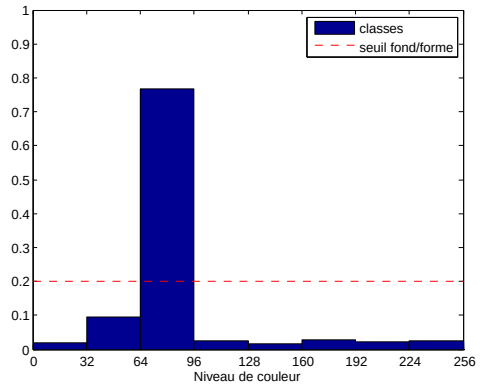

Figure 1 – *Vue des niveaux de classes et du seuil du vumètre*

Chaque nouveau pixel ayant une classe correspondante sous ce seuil sera détecté comme appartenant au fond.
En mode RVB, chaque pixel est modélisé par 3 vumètres (un par composante). Pour considérer un pixel comme fond, il doit être détecté comme fond par chaque vumètre.
Afin d'améliorer la détection et réduire des problèmes liés aux seuils entre deux classes, la valeur des classes au voisinage de la classe correspondante à un pixel est aussi mise à jour, mais de façon moindre.
Pour obtenir un bon apprentissage et une bonne adaptation de l'algorithme, il est nécessaire de bien choisir les paramètres (taux d'apprentissage α et seuil T). Ces valeurs peuvent être changées en fonction de la luminosité ou de la vitesse des véhicules suivis.

4 Génération des scènes et de leurs vérités terrains

Le principal problème pour mesurer la qualité de l'extraction provient de la difficulté à obtenir une vérité terrain que l'on pourra ensuite comparer à l'image extraite. La solution est d'identifier à l'aide d'un logiciel de dessin, de façon manuelle, les zones de formes et les zones de fond, et cela pour chacune des images de la séquence à analyser. Ce travail est assez fastidieux, et ne permet pas de tester les algorithmes avec beaucoup de séquences étant donné le temps nécessaire à la réalisation des vérités terrains.

La solution que nous proposons, consiste à générer des scènes virtuelles à l'aide d'un logiciel. Ceci a principalement deux gros avantages :

– La scène est paramétrée, donc on peut choisir d'isoler un seul facteur (exemple : le bruit), afin de voir le comportement de chaque algorithme face à celui-ci.
– Pour chaque séquence générée (figure 2), on obtient une séquence de vérités terrains associées (figure 3), permettant ainsi une mesure plus précise de la qualité d'extraction.

Figure 2 – *Scène générée*

Figure 3 – *Vérité terrain*

Les scènes sont réalisées grâce au logiciel SiVIC développé par le LIVIC [5], qui permet de générer à l'aide de scripts des scènes avec leurs vérités terrain associées. Les scripts permettent de contrôler chaque paramètre (luminosité, trajectoires des véhicules, bruit sur l'image, emplacement de la caméra, ...).

5 Méthodes de comparaison

Nous allons comparer les trois méthodes décrites précédemment sur différents critères. Le principal, celui dont nous allons parler dans cette partie, est bien entendu la qualité d'extraction. Nous tenons aussi compte de la vitesse d'exécution.

5.1 Classement des pixels

Afin d'analyser la qualité de l'extraction, l'image i obtenue après traitement par l'un des algorithmes est comparée avec l'image de vérité terrain correspondante. Les pixels sont donc classés suivant quatre catégories :

– VP_i (vrais positifs) : 1er plan détecté comme 1er plan
– FP_i (faux positifs) : fond détecté comme 1er plan
– VN_i (vrais négatifs) : fond détecté comme fond
– FN_i (faux négatifs) : 1er plan détecté comme fond

Pour chaque image i de la séquence, on compte le nombre de pixels dans chacune de ces quatre catégories.

5.2 Mesure Δ

Pour analyser la qualité de l'extraction à partir des pixels classés, on utilise une mesure appelée Δ. Le principe revient à calculer deux taux :

– La sensibilité (Se) : $Se_i = \frac{VP_i}{VP_i+FP_i}$
– La spécificité (Sp) : $Sp_i = \frac{VN_i}{VN_i+FP_i}$

La sensibilité reflète une bonne détection d'un objet, alors que la spécificité met plutôt en valeur la bonne détection du fond. L'idéal est d'avoir ces deux valeurs à 1.

On place ensuite les points Se_i en fonction de $1 - Sp_i$ pour toute une séquence (voir figure 4). La détection parfaite est caractérisée par le point de coordonnées (0,1). Plus on sera proche de ce point idéal, plus l'extraction pourra être considérée comme bonne. La droite de non discrimination passant par les points (0,0) et (1,1) montre que l'on ne parvient pas à différencier le fond de la forme.

Pour calculer la qualité de l'extraction, on mesure la distance Δ sur l'axe des ordonnées entre le point parfait (0,1) et la droite parallèle à la droite de non discrimination passant par le point à tester. On effectue cette mesure pour chaque point, puis on calcule la moyenne de ces distances, soit pour N points de coordonnées x_i et y_i, ce qui revient à calculer :

$$\Delta = \frac{1}{N} \sum_{n=1}^{n=N} 2 - Sp_i - Se_i \qquad (29)$$

L'idéal est d'avoir Δ qui tend vers 0.

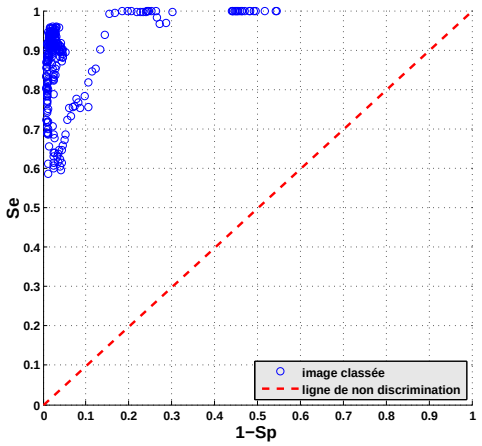

Figure 4 – *images dans un repère sensibilité/spécificité*

5.3 Mesure F

Un autre critère très intéressant est la mesure F. Pour cela on calcule la précision et le rappel d'après les pixels classés suivant les quatre catégories décrites précédemment. On a

donc

$$Prec_i(P) = VP_i/(VP_i + FP_i) \quad (30)$$
$$Prec_i(N) = VN_i/(VN_i + FN_i) \quad (31)$$
$$Rap_i(P) = VP_i/(VP_i + FN_i) \quad (32)$$
$$Rap_i(N) = VN_i/(VN_i + FP_i) \quad (33)$$
$$Prec_i = (Prec_i(P) + Prec_i(N))/2 \quad (34)$$
$$Rap_i = (Rap_i(P) + Rap_i(N))/2 \quad (35)$$
$$F_i = \frac{Prec_i \times Rap_i}{Prec_i + Rap_i} \quad (36)$$

L'idéal sera donc d'obtenir une mesure proche de 1, ce qui montrerait une extraction parfaite. La note F attribué sera la moyenne de tous les F_i sur la séquence.

6 Tests

6.1 Séquences

Pour analyser ces algorithmes, nous utilisons différentes scènes avec leur vérités terrain associées. La première est réelle, il s'agit de la séquence data3 venant de l'IPPR contest 2006 [1]. Les scènes suivantes, sont simulées à l'aide du logiciel SiVIC.

- **vidéo 1** (fig. 5) : Scène réelle, rue avec piétons et bus, scène assez sombre.
- **vidéo 2** (fig. 6) : carrefour, véhicules sur différentes voies, 3 voitures et piétons, bruit assez fort, changement de luminosité faible.
- **vidéo 3** (fig. 7) : rue, scène assez sombre, 1 bus, véhicules variés et des piétons, changement brusque de luminosité à mi-séquence.
- **vidéo 4** (fig. 8) : rond-point, un seul véhicule qui ne génère pas d'ombre, un arbre au milieu du rond-point avec ombre, l'arbre bouge, mouvement du soleil.
- **vidéo 5** (fig. 9) : circulation dense, voitures de tailles et couleurs différentes et 1 bus, ombres venant des véhicules et des bâtiments, masquages entre véhicules.
- **vidéo 6** (fig. 10) : identique à la vidéo 5 mais filmée sous un angle différent : de l'autre côté de la route.
- **vidéo 7** (fig. 11) : vue de dessus, bruit assez fort, automobiles de couleurs et tailles variées et 1 bus, ombres des véhicules.
- **vidéo 8** (fig. 11) : identique à la vidéo 7, bruit faible.

Figure 5 – *vidéo 1*

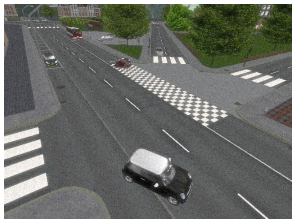

Figure 6 – *vidéo 2*

1. http ://media.ee.ntu.edu.tw/Archer_contest

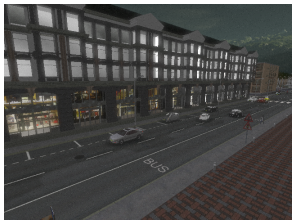

Figure 7 – *vidéo 3*

Figure 8 – *vidéo 4*

Figure 9 – *vidéo 5*

Figure 10 – *vidéo 6*

Figure 11 – *vidéos 7 & 8*

Vidéo	Durée	Images	Taille	img/sec
1	00 : 59, 80	299	320×240	5
2	00 : 28, 96	724	640×480	25
3	00 : 29, 00	725	640×480	25
4	00 : 39, 24	981	640×480	25
5	00 : 23, 04	576	640×480	25
6	00 : 23, 04	576	640×480	25
7	00 : 26, 84	671	640×480	25
8	00 : 26, 48	662	640×480	25

Tableau 1 – *Caractéristiques des séquences de test*

6.2 Paramètres des tests

Étant donné que le paramétrage de l'algorithme lors du test aura toute son importance pour la qualité de l'extraction, chaque séquence est testée avec les mêmes paramètres. On définit des configurations possibles pour chacune des méthodes (voir tableau 2). Et on choisit les paramètres qui obtiennent les meilleurs résultats.

7 Analyse des résultats

7.1 Qualité d'extraction

On calcule la qualité de l'extraction comme décrit précédemment. Cette mesure est faite pour chacune des

	MOG		CB2		VUM	
config.	δ	K	α	β	α	T
1	0,9	0,01	0,4	1,1	0,01	0,2
2	0,8	0,01	0,4	1,3	0,01	0,4
3	0,5	0,01	0,4	1,5	0,01	0,1
4	0,9	0,005	0,55	1,1	0,005	0,2
5	0,8	0,005	0,55	1,3	0,005	0,4
6	0,5	0,005	0,55	1,5	0,005	0,1
7	0,9	0,02	0,7	1,1	0,02	0,2
8	0,8	0,02	0,7	1,3	0,02	0,4
9	0,5	0,02	0,7	1,5	0,02	0,1

Tableau 2 – *Paramètres des tests avec les trois méthodes*

séquences avec chaque méthode et chaque configuration (cf. tableau 2). On garde la configuration qui obtiendra les meilleurs résultats globalement pour toutes les méthodes. Les résultats obtenus avec ce réglage de paramètres optimums pour les différentes vidéos sont rapportés dans le tableau suivant :

	MOG *(config 3)*		CB2 *(config 9)*		VUM *(config 6)*	
vid	Δ	F	Δ	F	Δ	F
1	0,317	0,813	0,152	0,884	0,188	0,914
2	0,517	0,615	0,528	0,643	0,517	0,613
3	0,570	0,724	0,393	0,796	0,501	0,828
4	0,366	0,705	0,315	0,766	0,317	0,786
5	0,364	0,850	0,396	0,854	0,298	0,883
6	0,433	0,816	0,475	0,822	0,357	0,859
7	0,293	0,798	0,195	0,870	0,266	0,728
8	0,244	0,904	0,144	0,928	0,103	0,919
moy	0,388	0,778	0,325	0,820	0,318	0,816

Tableau 3 – *Mesure de Δ et F*

Étant donné que l'idéal est d'avoir Δ=0 et F=1, on remarque que la qualité est globalement plus mauvaise avec la mixture de gaussiennes, mais en revanche, la qualité d'extraction est à peu près identique de façon générale entre le vumètre et le codebook 2 layers. Le vumètre est meilleur sauf dans les deux scènes où le bruit y est très exagéré. Ceci est dû à la largeur d'une classe qui ne peut pas contenir les variations liées au bruit.

7.2 Vitesse d'exécution

Les tests ont été réalisés sur un processeur Intel Xeon® E5520 cadencé à 2,26 GHz, la vitesse d'exécution dépend donc de ces caractéristiques. Pour que ces vitesses soient comparables, tous les algorithmes testés ont été codés en langage C.

Le temps moyen de calcul pour chaque séquence avec les différents paramètres possibles a été mesuré. On peut en déduire la vitesse moyenne de traitement d'une image avec chacune des méthodes (cf. tableau 4).

La vitesse d'exécution est plus rapide avec le vumètre, puis le codebook 2 layers et enfin la mixture de gaussiennes. Mais dans nos conditions de tests, aucune de ces trois méthodes ne permet de faire un traitement en temps réel à 25 img/s sur des images de taille 640×480.

	fps		
Séquence	MOG	CB2	VUM
vidéo 2	5,08	10,31	11,90
vidéo 3	5,21	10,31	13,70
vidéo 4	5,18	11,24	13,51
vidéo 5	5,24	10,64	13,51
vidéo 6	5,21	8,40	12,66
vidéo 7	5,13	9,80	13,33
vidéo 8	5,18	10,64	12,20
moyenne	5,13	9,90	12,66

Tableau 4 – *Comparaison des temps d'exécution*

8 Conclusion

Nous avons montré une comparaison de trois méthodes d'extraction fond/forme. Celles-ci ont été testées avec des scènes simulant les problématiques de circulation routière. On remarque qu'en règle générale, la méthode du vumètre de Goyat *et al.* [3] est meilleure aussi bien sur la qualité d'extraction que sur la vitesse d'exécution. Elle possède néanmoins ses limites avec des bruits extrêmement forts. Le codebook 2 layers [4] donne également de très bons résultats. La mixture de gaussiennes [1] est la moins bonne des trois méthodes testées. Il pourrait être intéressant, par la suite, de pouvoir mettre à disposition ces vidéos avec leurs vérités terrains, afin que chacun puisse comparer ses propres méthodes avec les mêmes critères. Une réflexion est en cours pour réaliser un site de partage de données.

Références

[1] C. Stauffer et W. E. L. Grimson, Adaptative background mixture models for a real-time tracking. Conference on Computer Vision and Patern Recognition, 1999, vol. II, pp. 246–252.

[2] K. Kim, T. H. Chalidabhongse, D. Harwood et L. Davis. Real-time foreground-background segmentation using codebook model. *Real-time Imaging*, 11(3) :167–256, 2005.

[3] Y. Goyat, T. Chateau, L. Malaterre et L. Trassoudaine. Vehicle trajectories evaluation by static video sensors. *9th IEEE International Conference on Intelligent Transportation Systems*, 2006.

[4] M. H. Sigari et M. Fathy. Real-time Background Modeling/Subtraction using Two-Layer Codebook Model. *International MultiConference of Engineers and Computer Scientists*, 2008.

[5] D. Gruyer, C. Royere, N. du Lac, G. Michel et J.-M. Blosseville. SiVIC and RTMaps, interconnected platforms for the conception and the evaluation of driving assistance systems. *World Congress and Exhibition on Intelligent Transport Systems and Services*, 2006.