

Analyse automatique de vidéos issues de caméras mobiles pour l'étude de comportements cyclistes

Gwenaël IELPO

Encadré par Stefan DUFFNER

Université Claude Bernard Lyon 1, France

Résumé Depuis l'explosion du Deep Learning en 2012, la vision par ordinateur a fait d'énormes progrès et de nombreuses approches et méthodes sortent chaque année pour repousser encore plus loin les limites de nos connaissances de ce domaine. Il reste malgré tout de nombreux verrous et défis scientifiques à résoudre dans le cadre de l'analyse de vidéo filmée en zone urbaine, notamment quand elles sont filmées à la première personne depuis un vélo. Il faut alors gérer les tremblements de la caméra ou encore les conditions climatiques et variations de luminosité . Pour résoudre ces problèmes, nous proposons des nouvelles méthodes à la croisée du Deep Learning et du traitement du signal afin dans un premier temps de stabiliser nos vidéos puis d'en extraire des caractéristiques ainsi que des comportements du cycliste.

Mots-clés: Intelligence Artificielle, Vision par ordinateur, Détection d'objet, Classification, CNN, Traitement du signal

Abstract. Since the rise of Deep Learning in 2012, computer vision has made gigantic progress and many new approaches and methods appear every year to push the boundaries of our knowledge in this area. Nevertheless, there are still many obstacles and scientific challenges to resolve in the analysis of video filmed in urban areas, especially when they are filmed at the first person from a bike. It is then necessary to manage the tremors of the camera or the climatic conditions and variations of luminosity. To solve these problems, we propose new methods at the crossroads of Deep Learning and signal processing in order to first stabilize our videos and then extract characteristics and behaviors of the cyclist.

Keywords: Artificial Intelligence, Computer Vision, Object Detection, Classification, CNN, Signal Processing

Table of Contents

Analyse automatique de vidéos issues de caméras mobiles pour l'étude de comportements cyclistes	1
<i>Gwenaël IELPO</i>	
<i>Encadré par Stefan DUFFNER</i>	
1 Introduction	4
2 Présentation de l'équipe de recherche	4
2.1 Présentation de l'équipe	4
2.2 Thématiques de l'équipe	5
2.2.1 Thème « Documents et transformation numérique »	5
2.2.2 Thème « Biométrie, visage et émotion »	6
2.2.3 Thème « Vision et Activités » (prise en compte explicite de la dimension temporelle, comme le suivi)	6
2.2.4 Thème “Interactions intelligentes et mobilité : de la capture aux traitements embarqués”	6
2.2.5 Thème Compréhension d’images et de scènes	7
3 Problématiques et état de l'art	7
3.1 Problématique des données :	7
3.2 État de l'art	8
3.2.1 Stabilisation vidéo	8
3.2.1.1 Scale Invariant Feature Transform (SIFT)	8
3.2.1.2 Speeded Up Robust Feature (SURF)	8
3.2.1.3 ORB	8
3.2.1.4 StabNet	9
3.2.2 Deep learning:	9
3.2.2.1 Réseaux de neurones convolutifs (CNN)	9
3.2.2.2 Réseaux de neurones récurrents (RNN)	10
3.2.2.3 Apprentissage par renforcement profond (DRL)	11
3.2.3 Détection d'objet	11
3.2.3.1 Region based CNN (R-CNN)	12

3.2.3.2	Fast Region-based Convolutional Network (Fast R-CNN)	13
3.2.3.3	Faster R-CNN	14
3.2.3.4	Region-based Fully Convolutional Network (R-FCN)	15
3.2.3.5	You Only Look Once (YOLO)	16
3.2.3.6	YOLOv2	17
3.2.3.7	Mask Region-based Convolutional Network (Mask R-CNN)	18
3.2.4	Détection des lignes de routes et du point de fuite	19
3.3	Limitations de l'apprentissage profond	19
4	Travaux et contributions	20
4.1	Stabilisation video	20
4.1.1	Apprentissage supervisé avec MLP	20
4.1.2	Stabilisation par apprentissage par renforcement profond	22
4.1.3	Corrélation de phase	22
4.2	Caractérisation de la scène par apprentissage automatique	23
4.2.1	Test des méthodes de segmentation sémantique et de détection d'objet les plus récentes	23
4.2.2	Détection des lignes de la route et du point de fuite	24
4.2.2.1	Dataset de Test	25
4.2.2.2	Détection de caractéristique dans la vidéo	27
4.2.2.3	Déterminer le point de fuite dans l'image	27
5	Conclusion	29

1 Introduction

Véléval est un projet s'intéressant à la cyclabilité des espaces urbains, c'est-à-dire aux conditions matérielles et techniques, mais aussi sociales qui entourent l'usage du vélo et conditionnent la qualité du déplacement cycliste. Leur objectif est d'intégrer les pratiques et vécus cyclistes dans l'évaluation des trajets et des lieux. Dans le cadre de ce projet, des cyclistes ont été équipé(e)s d'une caméra, montée sur le guidon, pour étudier leur comportement lors des trajets domicile-travail.

Or, l'analyse de vidéos filmées « à la première personne » depuis un vélo dans un contexte urbain comporte de nombreux défis scientifiques. Ces défis sont liés à une grande variabilité des données due aux changements d'éclairage et de conditions météorologiques, aux tremblements de la caméra ainsi qu'aux piétons, véhicules et d'autres objets en mouvement autour du cycliste. Beaucoup de recherches sont faites sur les voitures autonomes autour de ces problématiques actuellement grâce à des techniques de vision par ordinateur et d'Intelligence Artificielle, mais leur contexte est bien plus contraint, il s'agit de véhicules plus stables avec une caméra attachée à un endroit spécifique, et l'accès à beaucoup de capteurs et de types données.

Je me suis donc consacré durant mon stage au développement de nouveaux algorithmes d'analyse d'images et d'apprentissage machine afin d'analyser automatiquement les vidéos de mobilité urbaine produites dans le cadre du projet IMU Véléval.

Le travail a consisté en deux étapes principales :

- premièrement, l'étude des méthodes de traitement et d'analyse vidéo et la mise en œuvre d'un nouvel algorithme de stabilisation de vidéos mieux adapté au type de données du projet et qui permet un meilleur rendu et une analyse plus performante des vidéos enregistrées ;
- deuxièmement, le développement d'algorithmes de classification et de détection par apprentissage automatique permettant de caractériser différentes situations et extraire des attributs liés aux comportements des cyclistes.

2 Présentation de l'équipe de recherche

2.1 Présentation de l'équipe

J'ai réalisé mon stage au LIRIS au sein de l'équipe Imagine, celle-ci a été créée en janvier 2004. Il s'agit de l'équipe la plus grande au sein du laboratoire, elle regroupe à ce jour 21 permanents (8 PU et 13 MCF), 29 doctorants et 17 post-doctorants provenant de cinq des tutelles du LIRIS (Université Lyon 1, Université Lyon 2, CNRS, Ecole Centrale de Lyon et INSA Lyon) et répartie sur 3 sites différents(Ecully, Bron et la Doua).

L'équipe travaille sur de nombreux objectifs mais ayant toutes pour lien la compréhension d'images multisources et multicapteurs ainsi que la notion



Fig. 1: Imagine et cinq thèmes principaux de l'équipe

d'objet visuel, on trouvera ainsi des activités portant sur l'analyse d'image de personnes, d'objets et de scènes en 2D et 3D (scènes naturelles ou urbaines, images aériennes et satellites, visages... etc), des séquences d'images et des flux vidéo ou encore des documents numérisés (cartes, textes écrits et imprimés, partitions et symboles... etc).

2.2 Thématiques de l'équipe

2.2.1 Thème « Documents et transformation numérique » Le thème « Document » constitue un des axes de recherche les plus anciens, préexistant à la création de l'équipe Imagine en 2005 et qui maintient un positionnement et une reconnaissance internationale. L'activité de recherche portant sur cette thématique est centrée autour de la chaîne de transformation numérique du document à partir de son image numérisée pour permettre l'identification et l'indexation des contenus en lien étroit avec de nombreux usages et pour des applications très ouvertes. Concernant ce thème, l'équipe s'intéresse à une très grande diversité de contenus parmi lesquels on trouve aussi bien les documents anciens du patrimoine (textes enluminés de la période médiévale, manuscrits et brouillons rédactionnels d'auteurs, premiers imprimés de la Renaissance, partitions musicales manuscrites inédites...), que des documents contemporains

administratifs et d'entreprise, et des images (et des vidéos) contenant des textes incrustés en langues latine et arabe.

2.2.2 Thème « Biométrie, visage et émotion » L'équipe cherche ici à approfondir ses travaux sur l'analyse de visages 3D et 2D, en visant les challenges complexes du domaine en vue de proposer des solutions encore plus proches des applications réelles, notamment dans des scénarios moins contraints en terme de variations de poses, occultations, conditions d'éclairage, expressions faciales, vieillissement... D'un point de vue méthodologique, ils ont exploré des représentations faciales basées sur la géométrie différentielle, des outils statistiques ou encore sur l'apprentissage profond."

L'équipe a proposé diverses solutions pour traiter le problème de la reconnaissance faciale dans des conditions non contraintes. Elle a notamment travaillé sur les expressions faciales et les changements de pose ou la problématique des variations de la luminosité.

Les travaux récents ont principalement concerné les problèmes et verrous de l'apprentissage profond. De plus le calcul émotionnel (reconnaître les émotions que peuvent véhiculer les expressions faciales, les voix ou les images) et l'analyse de l'expression faciale a été une extension logique des travaux de l'équipe sur la reconnaissance de visages.

2.2.3 Thème « Vision et Activités » (prise en compte explicite de la dimension temporelle, comme le suivi) Les travaux sur ce thème consistent en des approches d'analyse automatique de flux vidéos ou de séquences d'images pour l'extraction d'informations sémantiques sur l'évolution de la scène ou de l'environnement observé ou sur des activités liées à des objets/personnes. L'objectif général est de développer des algorithmes génériques et robustes permettant d'extraire, d'analyser et de comprendre ces informations dans des conditions réelles qui sont, la plupart du temps, très difficiles en raison du bruit et des variations importantes dans l'image (l'éclairage, occultations, scènes complexes, mouvements abrupts, basse résolution, différents points de vue, déformations, etc.). Plus concrètement, trois différents types de scénarios ont été traités. Ils ont nécessité des approches différentes. Premièrement, le traitement de flux vidéos d'une caméra (de surveillance) ou d'un réseau de caméras pour l'estimation du mouvement dans la scène observée, le suivi d'objets ou de personnes et leur reconnaissance ou ré-identification. Deuxièmement, l'analyse de vidéos montrant une ou plusieurs personnes, par exemple, pour estimer et interpréter leurs postures, leurs activités et leurs interactions. Et enfin, l'analyse de l'évolution à plus long terme de l'environnement ou du terrain dans des séquences d'images satellites ou aériennes.

2.2.4 Thème “Interactions intelligentes et mobilité : de la capture aux traitements embarqués” Les activités menées par l'équipe sur ce thème consiste en du traitement du signal et des images et de l'apprentissage automatique, embarqués notamment sur smartphone. Dans ce cadre, les contraintes

imposées par le matériel sont prises en compte pour développer de nouvelles méthodes peu coûteuses en temps de calcul ou en impact mémoire. Par ailleurs, les instruments accessibles sur smartphone (boussole, accéléromètre, G.P.S., horloge, double caméra notamment) fournissent autant de données supplémentaires qui peuvent être fusionnées avec des données images ou considérées par les algorithmes pour les guider. De plus, les smartphones ouvrent la voie d'une interaction avec l'utilisateur, ce qui permet de prendre en compte leurs retours et critique sur l'application et de développer des algorithmes semi-automatiques ou avec des initialisations, ou des post-traitements, prenant en compte l'avis de l'utilisateur.

2.2.5 Thème Compréhension d'images et de scènes Les activités de recherche visent la compréhension automatique des images et vidéos. L'objectif est de développer de nouveaux algorithmes permettant d'identifier automatiquement des éléments d'intérêt dans les images et vidéos. De plus, ces dernières années ont été marquées par l'essor de l'apprentissage profond dont les performances dans ce domaine sont particulièrement remarquables, mais qui soulève de nouvelles problématiques, en particulier liées au fait que l'apprentissage de modèles de ce type requiert une quantité très importante de données, qui n'est pas toujours disponible :

1. détection d'objets faiblement supervisée ;
2. apprentissage par transfert pour la classification d'images ;
3. segmentation d'images ;
4. développement de bases volumineuses pour contribuer au développement de méthode d'apprentissage par des modèles profonds: Jacquard (HAL : hal-01753862) pour la détection dans des images de prises d'objets par des bras robotiques. LIRIS-ACCEDE (HAL : hal-01375518) pour la reconnaissance de l'émotion induite par les vidéos.

3 Problématiques et état de l'art

3.1 Problématique des données :

Les données sur lesquelles je vais travailler présentent de nombreux plusieurs défis, il s'agit donc ici de vidéos filmées à la première personne depuis un vélo (l'angle de vue serait depuis le guidon qui est généralement l'endroit où est fixé la gopro). Les différents défis proviennent de la variabilité des données, on pourra citer :

1. des changement d'éclairage ;
2. différentes types de conditions météorologiques ;
3. des tremblement de la caméra ;
4. des piétons, véhicules et objet en mouvement autour du cycliste.

Toutes ces contraintes sont assez similaires à celles des voitures autonomes, mais leurs résolutions posent moins de problèmes dans leurs cas, la voiture étant plus stable, avec une caméra fixée à un endroit précis, plusieurs autres types de capteurs pour acquérir d'autres données que la vidéo et le fait qu'une voiture roulera toujours en général sur la route.

3.2 État de l'art

3.2.1 Stabilisation vidéo

Plusieurs méthodes de stabilisation existent, elles peuvent être basées sur la description de features, le traitement du signal ou encore les réseaux de neurones.

Dans le domaine de la description de features, de nombreuses techniques existent, comme Harris corners, SIFT, SURF, FAST, BRIEF et ORB, je ne présenterais ici que SIFT, SURF et ORB.

3.2.1.1 Scale Invariant Feature Transform (SIFT)

SIFT [21] ou la "transformation de caractéristiques visuelles invariante à l'échelle" est un algorithme utilisé pour détecter et décrire les caractéristiques locales dans les images numériques. Il localise certains points clé et leur fournit ensuite des informations quantitatives (appelées descripteurs) qui peuvent par exemple être utilisées pour la reconnaissance d'objets. Les descripteurs sont supposés être invariants vis-à-vis de diverses transformations qui pourraient donner un aspect différent aux images bien qu'ils représentent le même objet. Mais SIFT n'est pas résilient qu'au changement de taille, mais également de rotation, luminosité ou encore de point de vue.

3.2.1.2 Speeded Up Robust Feature (SURF)

SURF [7] est un algorithme rapide et robuste, présenté par Herbert Bay et al. en 2006, pour la représentation et la comparaison d'images locales et invariantes par similarité. Il est inspiré de SIFT, est plus rapide que lui et serait plus résistant à différentes transformations d'images. Son intérêt principal par rapport à celui-ci réside dans le calcul rapide des opérateurs utilisant des filtres de boîte, permettant ainsi des applications en temps réel telles que le suivi et la reconnaissance d'objets. Il diffère dans le fait qu'il utilise des ondelettes de Haar pour l'extraction du descripteur de points d'intérêt.

3.2.1.3 ORB

Le détecteur ORB [4] est une combinaison de FAST et de BRIEF. Pour extraire les points clés, une variante du détecteur FAST (oFast) est utilisée pour déterminer les points clé, celle-ci en comparaison à FAST est invariante à l'échelle. Une fois les points clé détectés, une mesure du coin de Harris est utilisée pour les trier et seuls les N points supérieurs sont choisis en fonction d'un seuil. C'est ensuite une variante du descripteur BRIEF (rBRIEF) qui est utilisée afin d'être invariant aux rotations en calculant l'orientation des points d'intérêts.

Bien qu'ORB soit la méthode la plus rapide, elle n'est pas toujours celle qui performe le plus efficacement comme décrit ici.

3.2.1.4 StabNet

Stabnet [23] est un modèle de stabilisation en ligne (c'est-à-dire qu'il stabilise sans savoir qu'elle sera la prochaine frame de la vidéo) Le problème de stabilisation en ligne a ici été converti en un problème d'apprentissage supervisé de transformation conditionnelle sans calculer explicitement un chemin de caméra. Le modèle quant à lui est un réseau siamois à deux branches se partageant leurs paramètres prenant en entrée 5 frames stables et une instable. Chaque branche consiste en un encodeur qui extrait les features des images qui lui sont passées en entrée et d'un regresseur prédit les paramètres de transformation permettant de stabiliser les images.

Pour ce qui est de leur dataset, ils ont tournées eux même 60 vidéos de 20 à 30 s divisées en 44 vidéos pour la partie d'entraînement, 8 pour la validation croisée et 8 pour le test. Ces vidéos consistent en 2 fois la même scène filmée de façon stable et instable.

D'autres méthodes existent encore comme celle publiée par Google[22]

3.2.2 Deep learning: ^{1 2}

3.2.2.1 Réseaux de neurones convolutifs (CNN)

Les réseaux de neurones convolutifs ont été inspirés par la structure du système visuel.

Un CNN [29] comprend trois types principaux de couches neuronales, à savoir les couches de convolution, les couches d'agrégation et les couches entièrement connectées. Chaque type de couche joue un rôle différent. La figure 2 montre la différence entre une architecture MLP (Perceptron Multi Couche) et une architecture CNN. Chaque couche d'un CNN transforme le volume d'entrée en un volume de sortie d'activation de neurone, pour finalement aboutir aux couches finales entièrement connectées, ce qui permet de mapper les données d'entrée sur un vecteur de caractéristiques 1D. Les CNN ont rencontré un vif succès dans les applications de vision par ordinateur, telles que la reconnaissance des visages, la détection d'objets, l'optimisation de la vision en robotique et les voitures autonomes.

¹ Cours de Deep Learning du MIT <https://deeplearning.mit.edu/>

² "Convolutional Neural Networks for Visual Recognition" de Stanford <http://cs231n.stanford.edu/>

Convolutional Neural Networks

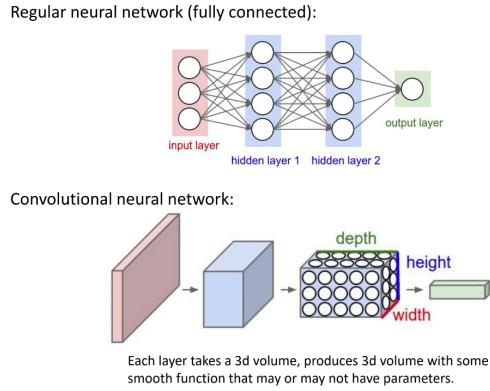


Fig. 2: CNN

3.2.2.2 Réseaux de neurones récurrents (RNN)

Les réseaux de neurones récurrents [1] sont un type particulier de réseau de neurones conçus pour les problèmes de séquence. Alors qu'un MLP prend en entrée un vecteur de taille fixe, ce qui limite son utilisation dans les situations impliquant une entrée de type "série" sans taille prédéterminée, le RNN est conçu pour prendre une série d'entrées sans taille prédéterminée.

La différence entre le réseau Perceptron multicouche utilisant le feed-forward standard et le RNN réside également dans l'ajout de boucles à l'architecture.

Ce sont ces boucles qui permettent au RNN de se "souvenir" du passé, ses décisions seront ainsi influencées par ce qu'il a appris auparavant. Les MLP "se souviennent" aussi des choses, mais ils se souviennent uniquement de ce qu'ils ont appris pendant leur entraînement.

Les neurones d'un RNN ne prendront donc en entrée pas uniquement les données d'entraînement, mais également ce qu'ils ont déjà appris.

Ces connexions récurrentes ajoutent un état ou de la mémoire au réseau et lui permettent d'apprendre des abstractions plus larges à partir des séquences d'entrée.

Long Short Term Memory

Le réseau LSTM [8] est un réseau de neurones récurrent formé à l'aide de la Backpropagation à travers le temps.

En tant que tel, il peut être utilisé pour créer de grands réseaux récurrents (empilés), qui à leur tour peuvent être utilisés pour résoudre des problèmes de séquence difficiles en apprentissage automatique et obtenir des résultats de pointe. De plus, au lieu de neurones, les réseaux LSTM ont des blocs de mémoire connectés en couches.

Un bloc LSTM a des composants qui le rendent plus "intelligent" qu'un neurone classique et une mémoire pour les séquences récentes. Un bloc contient des portes qui gèrent l'état et la sortie du bloc. Une unité fonctionne sur une séquence d'entrée et chaque porte dans une unité utilise la fonction d'activation sigmoïde pour contrôler si elles sont déclenchées ou non, rendant conditionnel le changement d'état et l'ajout d'informations circulant dans l'unité.

Il existe trois types de portes dans une unité de mémoire :

- forget Gate: décide conditionnellement quelles informations supprimer de l'unité ;
- input Gate: détermine de manière conditionnelle les valeurs de l'entrée pour mettre à jour l'état de la mémoire ;
- output Gate: détermine conditionnellement ce qu'il faut produire en fonction de l'entrée et de la mémoire de l'unité.

Les réseaux LSTM ont de nombreuses applications, notamment dans les domaines de la NLP ou de la vision par ordinateur.

3.2.2.3 Apprentissage par renforcement profond (DRL)

L'apprentissage par renforcement est un type important d'apprentissage automatique dans lequel un agent apprend à se comporter dans un environnement en effectuant des actions et en visualisant les résultats.

Le Deep Reinforcement Learning introduit les réseaux de neurones profonds pour résoudre les problèmes d'apprentissage par renforcement.

3.2.3 Détection d'objet

Les différentes méthodes vues précédemment ont alimenté de grands progrès dans une variété de problèmes de vision par ordinateur, tels que la détection d'objet, le suivi de mouvement, l'estimation de la pose humaine et la segmentation sémantique. Je vais ici passer en revue les principaux développements des architectures d'apprentissage en profondeur et des algorithmes pour les applications de vision par ordinateur. Je me concentrerai uniquement sur les architectures basées sur les réseaux de neurones convolutifs, car il prendrait trop de place de décrire ici toutes les étendues des différentes méthodes du domaine, que ce soient celles basées sur d'autres architectures de réseaux de neurones comme les LSTM et GAN ou celle de traitement du signal.

La détection d'objet est le processus de détection d'instances d'objets sémantiques d'une certaine classe (tels que des humains, des avions ou des oiseaux) sur des images numériques et des vidéos. Une approche commune pour les infrastructures de détection d'objets inclut la création d'un grand ensemble de fenêtres candidates qui sont classées dans la suite par des fonctions CNN. Un grand nombre d'œuvres est basé sur le concept de régions à caractéristiques CNN. Les approches qui suivent le paradigme de classification basée régions avec CNN ont généralement une bonne précision de détection. Cependant, de nombreuses méthodes tentent d'améliorer encore les performances de ces méthodes, dont certaines parviennent à trouver des positions approximatives de l'objet, mais ne

peuvent souvent pas déterminer avec précision sa position exacte. À cette fin, ces méthodes suivent souvent une approche conjointe de détection d'objet et de segmentation sémantique.

Il faut également ajouter qu'une grande majorité des travaux sur la détection d'objets utilisant l'apprentissage en profondeur appliquent une variante des CNN et non pas le CNN standard.

3.2.3.1 Region based CNN (R-CNN)

Les premiers modèles commencent intuitivement avec la recherche de région, puis effectuent la classification sur les régions sélectionnées. Dans R-CNN, la méthode de recherche sélective [16] est une alternative à la recherche exhaustive dans une image pour capturer la localisation d'un objet. Il initialise les petites régions d'une image et les fusionne avec un regroupement hiérarchique. Ainsi, le groupe final est une boîte contenant l'image entière. Les régions détectées sont fusionnées en fonction de divers espaces colorimétriques et métriques de similarité. Le résultat est un nombre limité de propositions de régions pouvant contenir un objet.

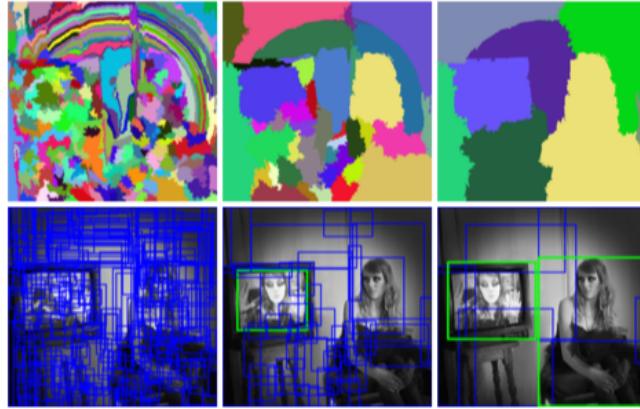


Fig. 3: RCNN

Le modèle R-CNN [24] combine la méthode de recherche sélective pour détecter les propositions de région et un apprentissage en profondeur pour découvrir l'objet dans ces régions. Chaque proposition de région est redimensionnée pour correspondre à l'entrée d'un CNN à partir duquel est extrait un vecteur d'entités de 4096 dimensions. Le vecteur de caractéristiques est introduit dans plusieurs classificateurs afin de produire des probabilités d'appartenir à chaque classe. Chacune de ces classes a un classifieur SVM formé pour déduire une probabilité de détecter cet objet pour un vecteur de caractéristiques donné. Ce vecteur alimente également un régresseur linéaire pour adapter les formes du cadre de

sélection pour une proposition de région et réduire ainsi les erreurs de localisation.

Le modèle CNN décrit par les auteurs est formé sur l'ensemble de données ImageNet 2012 du défi initial de la classification des images. Deux versions sont produites, l'une utilisant le jeu de données PASCAL VOC 2012 et l'autre le jeu de données ImageNet 2013 avec des cadres de sélection. Les classificateurs SVM sont également formés pour chaque classe de chaque jeu de données.

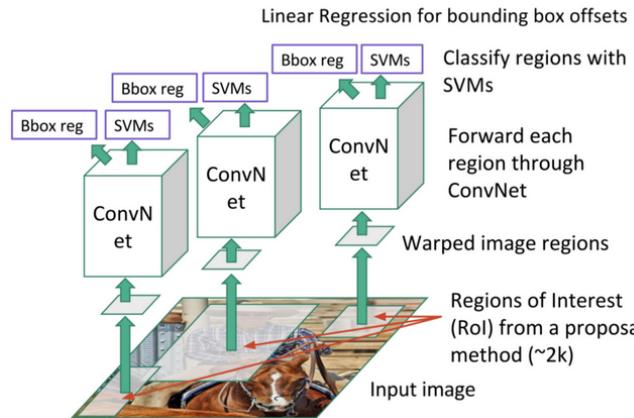


Fig. 4: RCNN

Source: J. Xu's Blog

3.2.3.2 Fast Region-based Convolutional Network (Fast R-CNN)

Fast R-CNN [6] a pour objectif de réduire la consommation de temps liée au nombre élevé de modèles nécessaires à l'analyse de toutes les propositions de région.

Un CNN principal avec plusieurs couches de convolution prend l'image entière en entrée au lieu d'utiliser un CNN pour chaque proposition de région (R-CNN). Les régions d'intérêt (RoI) sont détectées avec la méthode de recherche sélective appliquée sur les cartes de caractéristiques produites. Officiellement, la taille des cartes de caractéristiques est réduite à l'aide d'une couche de regroupement de RoI afin d'obtenir une région d'intérêt valide avec une hauteur et une largeur fixe en hyperparamètres. Chaque couche RoI alimente des couches entièrement connectées, créant ainsi un vecteur d'entités. Le vecteur est utilisé pour prédire l'objet observé avec un classifieur softmax et pour adapter les localisations du cadre de sélection avec un régresseur linéaire.

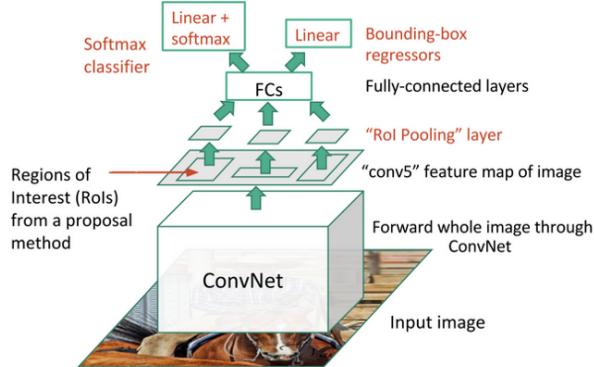


Fig. 5: FAST RCNN

Source: J. Xu's Blog

3.2.3.3 Faster R-CNN

Les propositions de région détectées avec la méthode de recherche sélective étaient encore nécessaires dans le modèle précédent, mais celle-ci sont coûteuses en calcul. S. Ren et al. ont introduit le Region Proposal Network [27] (RPN) pour générer directement des propositions de région, prédire les cadres de sélection et détecter des objets. Faster R-CNN est une combinaison du modèle RPN et du modèle Fast R-CNN.

Un CNN prend en entrée l'ensemble de l'image et applique un filtre de convolution pour en générer des cartes de caractéristiques (c'est-à-dire une nouvelle représentation de nos données dans un nouvel espace). Une fenêtre glissante applique ensuite ces différentes cartes et génère un vecteur de caractéristiques lié à deux couches entièrement connectées, une pour la régression et une pour la classification. Les propositions de régions multiples sont prédites par les couches entièrement connectées. Un maximum de k régions est fixé, ainsi la sortie de la couche de régression a une taille de $4k$ (coordonnées des boîtes, leur hauteur et leur largeur) et la sortie de la couche de classification de boîtes a une taille de $2k$ (scores «d'objectivité» décidant si un objet se trouve ou non dans une boîte). Les k propositions de région détectées par la fenêtre glissante sont appelées ancrées.

Lorsque les boîtes d'ancrage sont détectées, elles sont sélectionnées en appliquant un seuil sur le score d'objectivité pour ne conserver que les boîtes appropriées. Ces boîtes d'ancrage et les map de features calculées par le modèle CNN initial alimentent un modèle Fast R-CNN.

Faster R-CNN utilise RPN pour éviter la méthode de recherche sélective, il accélère les processus de formation et de test et améliore les performances. RPN utilise un modèle pré-formé sur le jeu de données ImageNet pour la classification et il est adapté au jeu de données PASCAL VOC. Ensuite, les propositions

de région générées avec des boîtes d’ancrage sont utilisées pour former le Fast R-CNN.

3.2.3.4 Region-based Fully Convolutional Network (R-FCN)

Les méthodes FAST/ER R-CNN consistent à détecter des propositions de région et à reconnaître un objet dans chaque région. Le R-FCN [11] est un modèle avec uniquement des couches convolutives permettant une rétro-propagation complète pour la formation et l’inférence. Les auteurs ont fusionné les deux étapes de base dans un même modèle pour prendre en compte simultanément la détection d’objet (invariant de localisation) et sa position (variante de localisation).

Un modèle ResNet-101 [18] prend l’image initiale en entrée. Les sorties de la dernière couche génèrent des cartes de caractéristiques, chacune étant spécialisée dans la détection d’une catégorie à un endroit donné. Ces cartes de caractéristiques sont appelées cartes de scores sensibles à la position car elles prennent en compte la localisation spatiale d’un objet particulier. Il se compose de $k * k * (C + 1)$ cartes de scores, où k est la taille de la carte de scores et C le nombre de classes.

En parallèle, il est nécessaire d’exécuter un RPN pour générer une région d’intérêt, nous couplons ensuite chaque région aux cartes de caractéristiques déterminer plus tôt et nous les comparons à la banque de résultats. Si suffisamment des parties de la carte sont activées, le patch vote ”oui”, je reconnais l’objet.

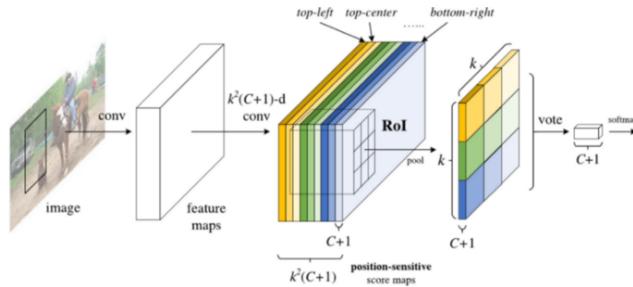


Fig. 6: RFCN

Source: J. Dai and al. (2016)

J. Dai et al. (2016) ont détaillé un exemple affiché ci-dessous. Pour la première image, ils montrent la réaction d’un modèle R-FCN spécialisé dans la détection d’une personne. RPN détecte une région d’intérêt au centre de l’image, les sous-régions des cartes de caractéristiques sont spécifiques aux modèles associés à une personne et votent donc ”oui, il y a une personne à cet endroit”. Dans la seconde image, le ROI est décalé vers la droite et n’est plus centré sur la personne. Les sous-régions dans les cartes de caractéristiques ne sont pas d’accord sur la détection d’une personne, elles votent donc ”non, il n’y a personne à cet endroit”.

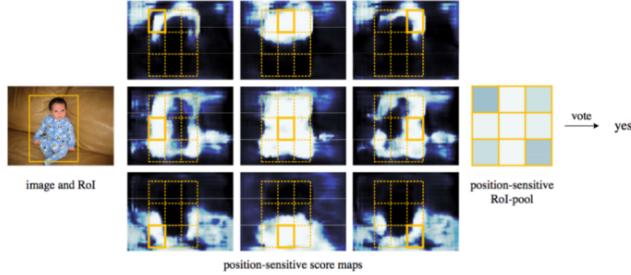
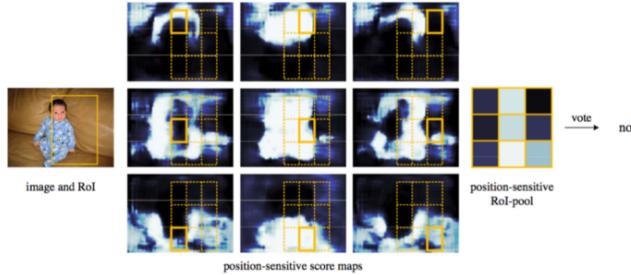
Figure 3: Visualization of R-FCN ($k \times k = 3 \times 3$) for the *person* category.

Figure 4: Visualization when an RoI does not correctly overlap the object.

Fig. 7: R-FCN

Source: J. Dai and al. (2016)

3.2.3.5 You Only Look Once (YOLO)

Le modèle YOLO [15] prédit directement les boîtes englobantes et les probabilités de classe avec un seul réseau dans une seule évaluation. C'est de cette caractéristique que vient son nom, il détecte de multiples objets en regardant l'image une seule fois. La simplicité du modèle YOLO permet des prédictions en temps réel, puisqu'il arrive à monter à 45 fps (Fast YOLOv1 accomplit lui le score de 155 fps).

Initialement, le modèle prend une image en entrée puis la divise en une grille SxS. Chaque cellule de cette grille prédit B boîtes englobantes ainsi qu'un score de confiance représentant la certitude qu'un objet se trouve dans cette boîte. De façon plus technique, cette confiance est la probabilité de détecter l'objet.

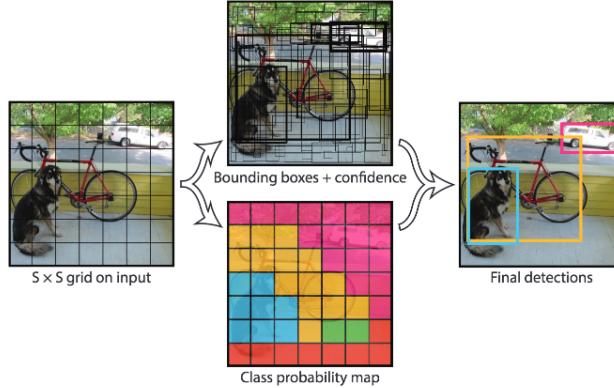


Fig. 8: YOLO

Le CNN utilisé est inspiré du modèle GoogLeNet qui a introduit le module Inception. Le réseau comporte 24 couches convolutives suivies de 2 couches entièrement connectées. Les couches de réduction avec 1×1 filtres suivies de 3×3 couches convolutives remplacent les modules initiaux. Le modèle Fast YOLO est une version plus légère avec seulement 9 couches convolutives et un nombre réduit de filtres. La plupart des couches de convolution sont pré-entraînées à l'aide du jeu de données ImageNet avec classification. Quatre couches de convolution, suivies de deux couches entièrement connectées, sont ajoutées au réseau précédent et sont entièrement reconvertis avec les jeux de données PASCAL VOC 2007 et 2012.

La couche finale produit un tenseur $S * S * (C + B * 5)$ correspondant aux prévisions pour chaque cellule de la grille. C'est le nombre de probabilités estimées pour chaque classe. B est le nombre fixe de boîtes d'ancrage par cellule, chacune de ces boîtes étant associée à 4 coordonnées (coordonnées du centre de la boîte, largeur et hauteur) et à une valeur de confiance.

Avec les modèles précédents, les boîtes englobantes prédites contenaient souvent un objet. Le modèle YOLO prédit toutefois un nombre élevé de boîtes englobantes. Il y a donc beaucoup de boîtes englobantes sans aucun objet. La méthode de suppression non maximale (NMS) est appliquée à la fin du réseau. Elle consiste à fusionner des boîtes englobantes qui se chevauchent d'un même objet en un seul. Bien que cette technique apporte de très grands progrès, on constate malgré tout encore quelques faux positifs.

3.2.3.6 YOLOv2

Le modèle YOLOv2 est axé sur l'amélioration de la précision tout en restant un détecteur rapide. La normalisation par lots est ajoutée pour éviter les surajustements sans utiliser les abandons. Les images de résolution supérieure sont acceptées en entrée. Le modèle YOLO utilise des images 448x448 tandis que

le YOLOv2 utilise des images 608x608, permettant ainsi la détection d'objets potentiellement plus petits.

La couche finale du modèle YOLO, entièrement connectée, qui prédit les coordonnées des boîtes de sélection, a été supprimée pour permettre l'utilisation des boîtes d'ancrage de la même manière que Faster R-CNN.

3.2.3.7 Mask Region-based Convolutional Network (Mask R-CNN)

Il s'agit également ici d'une extension du modèle Faster R-CNN publiée par K. He et al., une branche est ici ajouté parallèlement à la détection du cadre de sélection afin de prédire le masque des objets détectés. Le masque d'un objet est sa segmentation par pixel dans une image. Il ne s'agit donc plus ici de simple détection d'objet, on parle maintenant de segmentation sémantique. Ce modèle est plus performant que l'état de l'art dans les quatre défis de COCO : la segmentation d'instance, la détection du cadre de sélection, la détection d'objet et la détection de point clé.

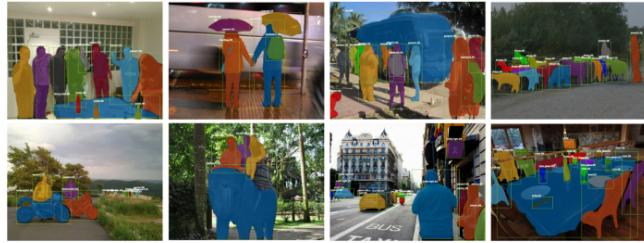


Fig. 9: MASK RC-NN

MASK R-CNN [17] utilise le pipeline Faster R-CNN avec trois branches de sortie pour chaque objet candidat : un label de classe, une boîte englobante et le masque de l'objet. Il utilise un RPN pour générer des propositions de boîtes englobantes et produit trois sorties en même temps pour chaque région d'intérêt.

Le modèle prend une image en entrée et alimente un réseau ResNeXt [26] avec 101 couches. Il détecte ensuite les RoI qui sont traités à l'aide d'une couche RoIAAlign. Une branche du réseau est liée à une couche entièrement connectée pour calculer les coordonnées des boîtes englobantes et les probabilités associées aux objets. L'autre branche est liée à deux couches convolutives, la dernière calcule le masque de l'objet détecté. Pour finir, les trois fonctions de coût de ces différentes tâches sont additionnées, puis cette somme est minimisée.

Mask R-CNN a donné d'excellents résultats, contrairement aux précédentes méthodes qui cherchaient juste à récupérer l'objet à détecter via une boîte englobante, nous avons ici une segmentation au pixel de l'objet à classifier, ce qui donne une bien meilleure localisation et aide la tâche de classification.

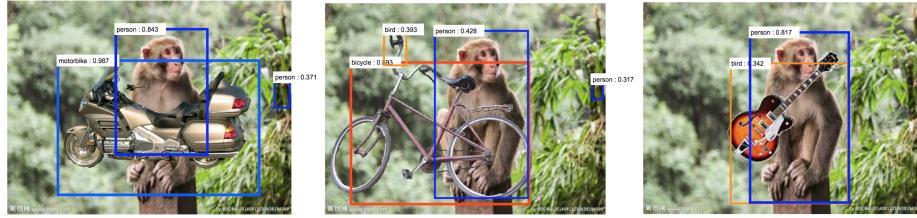


Fig. 10: Problème du changement de contexte

Il y a encore bien d'autres méthodes que j'ai analysé, mais dont je ne parlerai pas plus par manque de place tel que Single-Shot Detector [2], YOLO9000 [13], YOLOv3 [14]...etc

3.2.4 Détection des lignes de routes et du point de fuite

J'ai également fait un travail de bibliographie et d'analyse sur les différentes techniques pour détecter les lignes de la route ou le point de fuite mais que je ne détaillerai pas plus ici. Les techniques consistent en général sur l'application de hough lines [3], de réseaux de neurones [20] ou d'autres techniques d'analyse d'image [25] [28] [9].

3.3 Limitations de l'apprentissage profond

Malgré ses réussites, l'apprentissage profond connaît actuellement des problèmes et limites.

1. Besoin d'une grande source de donnée annotées. Il existe néanmoins des méthodes pour y pallier un peu comme le transfert learning, l'apprentissage non supervisé... mais elles n'ont pas donné d'aussi bons résultats pour le moment que l'apprentissage supervisé.
2. Les modèles ont beaucoup de succès sur les datasets de benchmark (MNIST, COCO, Pascal, Cityscapes...) mais peuvent avoir de mauvaises performances sur les datasets du monde réel. Les datasets peuvent avoir des biais, et un NN qui aura fait son apprentissage dessus pour faillir à détecter un objet à cause d'un changement de condition (arrière plan, point de vue) ou par des événements rares.
3. Les modèles sont très sensibles au changement dans l'image, par exemple les changements de contexte peuvent jouer, sur l'image ci-dessous on voit que le fait d'ajouter une moto ou une guitare au singe nous le classer comme un humain.
Ces problèmes peuvent aussi être considérés comme une conséquence directe de la taille trop peu importante d'un dataset
4. Problème de l'occlusion : les modèles actuels ont encore beaucoup de mal à détecter ou classifier un objet qui est partiellement caché. Les occlusions

se produisent sous deux catégories, d'une part, son auto-occlusion, ce qui signifie que, d'un certain point de vue, une partie d'un objet est occluse par une autre. Deuxièmement, son occlusion inter-objet qui signifie que deux objets suivis s'obstruent.

4 Travaux et contributions

4.1 Stabilisation vidéo

J'ai consacré la première partie de mon stage à travailler sur des algorithmes de stabilisation vidéo. Après avoir réalisé l'état de l'art sur les différentes méthodes utilisant ou non l'intelligence artificielle, j'ai commencé à implémenté l'idée de Stefan Duffner.

4.1.1 Apprentissage supervisé avec MLP

L'idée était de reprendre l'algorithme de stabilisation par matching des features [19] :

1. Lire les frames T et T-n
2. Détecer les features des deux frames
3. Sélectionner les correspondances entre les points
4. Estimer la transformation à appliquer depuis les bruits entre les correspondances
5. Appliquer les transformations et fluidification

Pour estimer les transformations, cet algorithme se base sur Random Sample Consensus (RANSAC), la majorité des correspondances entre les points sélectionnées plus tôt ne sont pas très précises. On observe que les correspondances dans le fond de l'image ne sont pas alignées sur le premier plan.



Fig. 11: Correspondances imprécises entre les points

La raison derrière ceci est que les caractéristiques d'arrière-plan sont suffisamment éloignées pour se comporter comme si elles se trouvaient sur un plan infiniment distant. Cela permet de supposer que le plan de l'arrière-plan est statique et ne changera pas de façon spectaculaire entre la première et la deuxième image ; au contraire, cette transformation capture le mouvement de la caméra. Ainsi, le processus de correction stabilisera la vidéo. L'algorithme RANSAC est donc répété plusieurs fois et à chaque exécution, le coût du résultat est calculé en projetant l'image B sur l'image A via la somme des différences absolues entre les deux images.



Fig. 12: Image stabilisée après transformation

Ce qui nous donne donc une image stabilisée.

Le but de notre méthode serait plutôt que d'utiliser Ransac, faire un apprentissage des vecteurs de mouvements des features de l'image avec un perceptron multi couche. Le réseau de neurones pourrait ainsi prédire les mouvements des features d'une vidéo, ce qui permettrait d'en corriger les bruits et secousses. J'ai donc commencé par écrire un script permettant de déterminer les features de ma vidéo frame par frame en testant la plupart des méthodes d'analyse d'image tel que SURF, SIFT ou ORB. Pour le premier essai, j'ai testé une architecture, standard de MLP 2-5-2 avec une fonction d'activation Sigmoid pour la couche cachée en lui passant en entrée les vecteurs de mouvements.

Cette méthode ne m'a malheureusement pas donné de bons résultats, le réseau de neurones n'arrivant pas à prédire de façon correcte les mouvements. j'ai essayé de changer le nombre de couche, de neurone ou encore la fonction d'activation, j'ai ajouté d'autres features comme la position de la features et non uniquement son vecteur de mouvement, mais sans succès. J'ai donc conclu que le problème venait des données en elle-même, qui n'était pas assez nombreuses et variées ou alors que le réseau de neurones n'arrivait pas à faire une modélisation du bruit de la vidéo.

4.1.2 Stabilisation par apprentissage par renforcement profond

Une autre idée que j'ai eue était l'utilisation de d'apprentissage par renforcement profond pour apprendre à un agent à stabiliser la vidéo en lui donnant la possibilité à l'agent d'appliquer des transformations comme des rotations, translations, homothéties ou similitudes comme le ferait un algorithme de stabilisation basique. La partie compliquée est de déterminer une fonction de récompense à optimiser, mon premier essai était de détecter la ligne d'horizon et de donner une récompense à mon agent quand il recentrait l'image sur elle, j'ai cependant abandonné cette piste après avoir reçu les vidéos du laboratoire d'urbanisme, car détecter la ligne d'horizon en paysage urbain peut s'avérer compliqué. A la place j'ai calculé le pourcentage de différence entre chaque frame F et F-1, après l'avoir tester sur une vidéo j'ai pu visualiser des pics successifs au moment où le cycliste subissait des chocs, cette fonction m'a donc semblait prometteuse.

J'ai également par la suite trouvé une autre fonction qui pourrait être utilisé pour ce cas. (dont je parlerai ici 4.2.2.2)

Je n'ai par contre pas persévéré sur cette piste après discussion avec mon tuteur.

4.1.3 Corrélation de phase

La dernière méthode que j'ai utilisée et celle du Cross Power Spectrum qui est basée sur des techniques de traitement du signal et plus spécifiquement des Transformées de Fourier 2D et s'éloigne des techniques d'IA. Elle est néanmoins celle qui m'a donné les meilleurs résultats, celle-ci n'ayant pas besoin de beaucoup de données pour pouvoir fonctionner.

Le but de cette technique est d'obtenir le mouvement apparent de la vidéo, pour cela, il faut corriger le mouvement global de la caméra entre 2 frames It et It+1. C'est ici qu'intervient le cross power spectrum, il prend en compte tous les mouvements des pixels à la fois dans le domaine fréquentiel.

Pour le déterminer, on calcule le rapport des transformées de Fourier de 2 images successives, ce rapport produit une onde que la transformée de Fourier restitue sous forme de diracs dont la position du dirac maximum est fonction du nombre de pixels au maximum corrélé.

En fonction de la vitesse du vélo, les différences entre 2 frames successives sont à la fois la translation globale de l'image due aux vibrations, une légère homothétie pour l'horizon et une plus importante homothétie pour les bandes proches. Il se peut également qu'une rotation soit nécessaire, car le vélo peut ne pas toujours être à l'horizontale quand le cycliste tourne. La rotation et la translation peuvent se déduire du spectre de Fourier. Mais l'homothétie d'une image se traduit par une homothétie dans le spectre de Fourier, ce qui s'avère complexe à détecter.

De plus, on ne peut pas détecter à la fois la rotation due au virage, la translation globale et l'homothétie en une seule fois.

Il est possible de détecter le mouvement global de la caméra en minimisant l'homothétie variable en appliquant le cross power spectrum à des log-images.

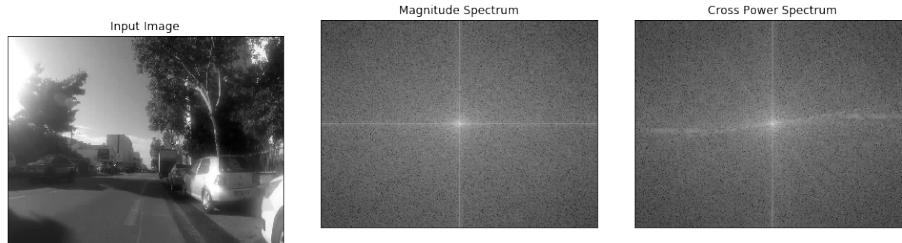


Fig. 13: Image original

Fig. 14: Transformée de Fourier de l'image

Fig. 15: Cross Power Spectrum

L'avantage d'une image en Log est qu'une homothétie de rapport Lambda en Log-Image se traduit en homothétie de rapport Log(Lambda) donc très minime. De plus dans mon cas, une homothétie dépend de la distance par rapport à la position de la caméra. Lambda est alors $\Lambda(y)$ avec $y=\text{ImageHeight}$ à $y=\text{ligne d'horizon}$ qu'il faut détecter.

Mais en Log-Image l'effet est réduit et il devrait être possible de mieux détecter le mouvement global de la caméra, puis les mouvements apparents et finalement de calculer l'homothétie qui donnera la vitesse du vélo.

4.2 Caractérisation de la scène par apprentissage automatique

4.2.1 Test des méthodes de segmentation sémantique et de détection d'objet les plus récentes

J'ai commencé par analyser et tester les différentes méthodes d'état de l'art des voitures autonomes, leur problème étant relativement proche du mien.

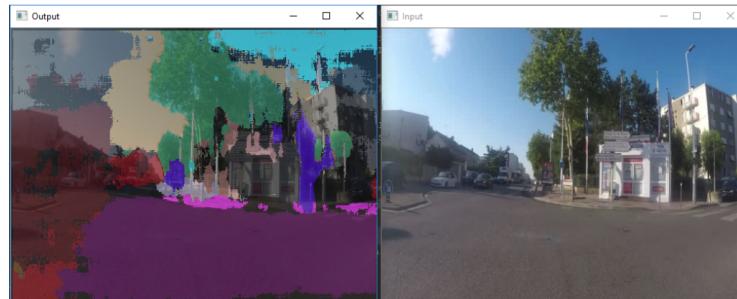


Fig. 16: MASK R-CNN

On constate que MASK R-CNN donne des résultats très mauvais, quelques zones sont parfois bien segmentées, mais il s'agit de cas rares et isolés.

J'ai également testé YOLO qui m'a donné de bien meilleurs résultats malgré quelques faux positifs (comme on peut le voir dans l'image ci dessous avec une voiture détectée où il n'y en a pas).



Fig. 17: YOLO

J'ai ensuite commencé à analyser les différentes pistes demandées par le laboratoire d'urbanisme, leurs problématiques et comment proposer des solutions :

1. classification du type de voie empruntée (route, trottoir, piste cyclable...) ;
2. position sur la chaussée ;
3. détecter panneaux, feux de signalisation... ;
4. le cycliste laisse t'il passer les piétons ?

Mes premières recherches se sont portées sur les méthodes de Deep learning, notamment les combinaisons de CNN et de LSTM [12] [10] [5].

4.2.2 Détection des lignes de la route et du point de fuite

Les detections des lignes de la route et du point de fuite sont très utiles pour déterminer des caractéristiques dans des vidéos comme les nôtres, elles aident à segmenter la route, déterminer la position sur la chaussée...

J'ai donc voulu mettre au point ma propre méthode afin de détecter les lignes de la route, une méthode qui serait capable de résoudre certaines problématiques de mes données. J'ai commencé par tester les différentes techniques existantes sur mes données afin de voir leurs résultats (comme celle basée sur les Hough lines). Ils leur arrivaient parfois de bien réussir, mais tout autant de faillir, cela venait souvent du fait qu'une ombre était sur la route, que la route était en mauvais état et avait été réparée grossièrement ou que le cycliste roulait sur des pavés ou des routes non goudronnées.

Le but de ma détection des lignes de la route était de pouvoir ensuite facilement extraire la route en elle-même pour ensuite la donner à un CNN et un

LSTM qui apprendrait par la suite à classifier le type de voie sur lequel le cycliste se trouve (la fonction du LSTM ici est de garder en mémoire la voie sur laquelle le cycliste est en ce moment pour n'avoir à refaire une classification uniquement lorsque celui-ci change de voie).

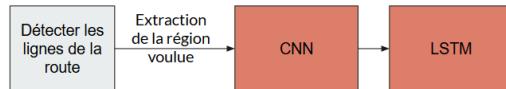


Fig. 18: Classification de voie

J'ai donc travaillé sur une idée différente d'apprentissage non pas basée sur l'analyse des pixels (avec un CNN) mais sur l'analyse des fréquences de l'image, mon idée été que les hough lines ou un réseau de neurones convolutionnel détectent toutes les lignes qui ressortent dans l'image sans réussir à les distinguer, mais la représentation fréquentielle d'une ligne du à une ombre ou à une bande blanche de la route elle diffère :

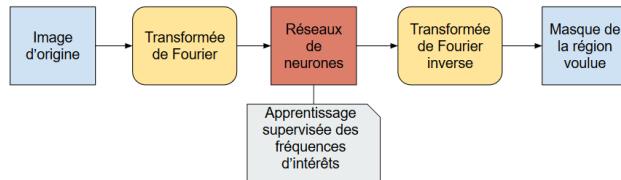


Fig. 19: Méthode d'extraction fréquentielle des lignes de la route

Ma méthode consistait donc à prendre une frame de ma vidéo, appliquée une transformée de Fourier 2D dessus pour en avoir la représentation fréquentielle, la donner en entrée à mon réseau de neurones qui s'occuperaient de prédire les fréquences correspondantes aux lignes de la route, puis d'appliquer une transformée de Fourier inverse sur cette image pour en récupérer le masque correspondant aux lignes.

4.2.2.1 Dataset de Test

Afin de tester mon idée, j'ai créé un dataset sur lequel travailler qui soit plus simple que les images du monde réel et moins bruitées pour déterminer si la méthode fonctionnerait sur un cas simple.

Comme vu ci-dessous, j'ai commencé par des formes géométriques simples sur des fonds de couleurs unies, puis j'ai petit à petit augmenté la difficulté par exemple en ajoutant des occlusions.

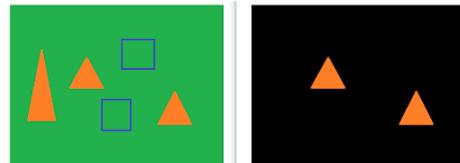


Fig. 20: Exemple



Fig. 21: Exemple d'occlusion

Je n'ai malheureusement pas continué à travailler sur cette idée, juste quelques jours après la construction de mon dataset j'ai participé à une réunion d'équipe où j'ai présenté mes travaux et ils se trouvent que quelques chercheurs de l'équipe avaient déjà travaillé sur une idée similaire sans résultats et m'ont expliqué pourquoi elle ne fonctionnait pas, je l'ai donc mise de côté et ai commencé à chercher une autre méthode.

J'ai de plus arrêté de chercher dans la voie des réseaux de neurones à cause de plusieurs points :

- premièrement ces méthodes sont très coûteuses en ressources et en puissance de calcul. N'ayant à ma disposition qu'une GTX 660, les entraînements auraient pris beaucoup trop de temps pour la durée de mon stage (je ne pouvais pas me permettre d'attendre une à deux semaines entre chaque entraînement pour voir s'il avait fonctionné)
- deuxièmement j'avais à la fois trop de données pour créer un dataset moi-même sans que ça me prenne plus d'un mois mais paradoxalement pas assez de données pour avoir de bons résultats, le deep learning étant très gourmand à ce niveau

- troisièmement mes données manquent cruellement de variété, le dataset consiste souvent en 5 ou 6 fois la même personne faisant le même trajet pour aller à son travail
- quatrièmement je ne pensais pas en travaillant tout seul et avec l'étendue de mes connaissances actuelles être capable de faire une vraie contribution dans ce domaine (le but de mon stage n'étant pas de juste reprendre un modèle et d'en changer un petit bloc)

Je me suis donc décidé à pousser l'accent de mes recherches sur la détection des zones d'intérêt de mon image en passant par des techniques d'analyse d'images "pures".

4.2.2.2 Détection de caractéristique dans la vidéo

Il m'est alors venu l'idée d'utiliser les connaissances en détection d'anomalies que j'ai acquéri lors de mon précédent stage et de mon TER durant mon M1, mon but était de voir si il aurait été possible de déterminer et visualiser des caractéristiques dans les vidéos en essayant de les modéliser comme des anomalies, que ce soit des à-coups du à la conduite du pilote, des mouvements du cahot de la route qui aurait besoin d'être stabilisé ou encore un "ADN" du cycliste (c'est-à-dire le caractériser en fonction de ses comportements).

J'en suis donc venu à cet algorithme :

1. Calcul du flot optique dense
2. Application de transformée en ondelette discrète 2D
3. Calcul de l'entropie puis calcul de l'entropie en fenêtre glissante (via la distance euclidienne)
4. Analyse du plot pour essayer de caractériser des comportements, les à-coups à stabiliser...etc avec du ML

L'entropie appliquée à ma transformée en ondelette discrète représente ici une estimation du mouvement dans l'image, le but était donc d'analyser le plot de l'entropie en fenêtre glissante dans ma vidéo, en faisant par exemple la distance euclidienne entre les entropies, si l'on aurait une constante cela signifierait par exemple une vidéo fluide et sans à-coups, mais si l'on avait des pics cela aurait pu correspondre à des cahots ou à des comportements caractéristiques du pilote, en repérant des patrons répétés plusieurs fois, on pourrait définir l'ADN du cycliste et apprendre à reconnaître sa conduite.

Mais cette idée m'a également permis de trouver un algorithme qui pourrait me permettre de déterminer le point de fuite dans mon image.

4.2.2.3 Déterminer le point de fuite dans l'image

Le point de fuite devrait logiquement être un des endroits où il y aura le moins de mouvement dans mon image, j'ai donc repensé à YOLO et à son idée de diviser l'image en plusieurs sous-images pour déterminer les régions d'intérêts et j'en suis venu à cet algorithme :

1. Calcul du flot optique dense

2. Division de l'image en sous-partie pour obtenir une grille, puis application d'une transformée en ondelette discrète sur chaque sous-régions
3. Calcul de l'entropie sur chaque cellule de la grille
4. Sélection de la sous-région correspondante au point de fuite (celle où le score d'entropie sera la plus bas)

Il est bien sûr possible d'avoir des faux positifs causés des objets se déplaçant à la même vitesse que le cycliste, il faudrait donc trouver un moyen par la suite de sélectionner la bonne sous région dans ces cas, mais pour le moment, je vais tester si cette méthode fonctionne sur des cas simples.

La réalisation d'un dataset de test ici s'avère plus compliquée que pour mon idée précédente, en effet ici il ne suffit pas d'analyser les frames de la vidéo une à une, le mouvement est essentiel au fonctionnement de l'algorithme puisque j'utilise le flot optique, il serait donc long et fastidieux de réaliser une vidéo frame par frame pour me permettre de tester ma méthode sur des données plus simple que le monde réel. J'en suis donc venu à regarder du côté des jeux vidéo de course des consoles tel que la super nes ou la mégadrive qui sont parfait pour mon problème, les graphismes de l'époque étant assez simple.



Fig. 22: Super Chase H.Q.



Fig. 23: Nakajima Satoru
Super F-1 Hero



Fig. 24: Kawasaki Super-
bike Challenge

Je suis actuellement en train de travailler et d'implémenter ces deux méthodes et je n'ai malheureusement pas encore pu les tester et donc de résultats à présenter.

5 Conclusion

Ce stage m'a permis de découvrir de manière plus approfondie le monde de la recherche. D'une part ses acteurs, j'ai pu voir de manière plus concrète en quoi consiste le travail d'un chercheur, que ce soit au niveau du doctorat par les différentes conversations que j'ai eu avec les doctorants du laboratoire ou via les travaux du chercheur (la bibliographie, analyser le travail de ses pairs...) D'une autre part ses enjeux, les responsabilités administratives qu'impliquent certains postes comme chef d'équipe ou directeur de thèse ou encore les différents événements et palier de la carrière d'un chercheur (j'ai par exemple assisté à la HDR de mon tuteur).

J'ai également eu la chance de pouvoir apprendre et travailler sur les domaines pointus que ce sont le Deep Learning et la vision par ordinateur. J'ai d'ailleurs trouvé fort intéressant et formateur de devoir les appliquer à des cas concrets et sur des problèmes et données du monde réelles. J'ai pu voir que dans le cas où les données posent problème, d'une part à cause de leur manque de variété ainsi que leur quantité, et d'autre part lorsque l'on manque de puissance de calcul, le Deep Learning n'est pas toujours la solution. Malgré les excellents résultats qu'il nous donne actuellement, les techniques d'analyse d'image pures et le traitement du signal peuvent être tout aussi utile.

Une leçon que je retiens donc est qu'avoir recours uniquement à l'intelligence artificielle n'est pas toujours une bonne idée, il est parfois plus sage de ne pas oublier tout le travail qui a été fait auparavant dans le domaine de la vision par ordinateur et d'essayer de coupler ses techniques avec les nouvelles.

D'un point de vue professionnel, le stage m'a permis de mieux formaliser mes projets futurs en me montrant les possibilités qu'offrait l'IA. Je suis intéressé à continuer de travailler dans l'Intelligence Artificielle qu'elle soit appliquée à la vision par ordinateur ou d'autres sujets qui m'intéressent tout autant (comme l'énergie, l'astrophysique ou la sécurité).

Remerciements : je remercie le LIRIS et l'équipe Imagine de m'avoir accueilli pour mon stage, ainsi que Stefan Duffner pour m'avoir encadré durant cette période. Je tiens aussi à remercier Véronique Eglin, Khalid Idrissi et Frank Lebourgeois pour les conseils et l'aide qu'ils ont pu me prodiguer. Je remercie également les doctorants pour leur bon accueil et Nicolas, le stagiaire avec qui j'ai partagé mon bureau ces 5 derniers mois.

References

1. David E. Rumelhart, G.E.H., Williams, R.J.: Learning representations by back-propagating errors (1986), <https://www.nature.com/articles/323533a0>
2. Detector, S.S.S.M.: Wei liu, dragomir anguelov, dumitru erhan, christian szegedy, scott reed, cheng-yang fu, alexander c. berg (2015), <https://arxiv.org/abs/1512.02325>
3. Ding, D., L.C., Lee, K.Y.: An adaptive road roi determination algorithm for lane detection (2013)
4. Ethan Rublee, Vincent Rabaud, K.K.G.B.: Orb: an efficient alternative to sift or surf (2011), http://www.willowgarage.com/sites/default/files/orb_final.pdf
5. Ghosh, R.: Deep learning for videos: A 2018 guide to action recognition (2018), <http://blog.quare.ai/notes/deep-learning-for-videos-action-recognition-review>
6. Girshick, R.: Fast r-cnn (2015), <https://arxiv.org/abs/1504.08083>
7. Herbert Bay, T.T., Gool, L.V.: Surf: Speeded up robust features (2006), <https://www.vision.ee.ethz.ch/~surf/eccv06.pdf>
8. Hochreiter, Schmidhuber: Long short term memory (1997), <https://dl.acm.org/citation.cfm?id=1246450>
9. Hui Kong, J.Y.A., Ponce, J.: General road detection from a single image (2009), <https://www.di.ens.fr/sierra/pdfs/tip10b.pdf>
10. Jeff Donahue, Lisa Anne Hendricks, M.R.S.V.S.G.K.S.T.D.: Long-term recurrent convolutional networks for visual recognition and description (2014), <https://arxiv.org/abs/1411.4389>
11. Jifeng Dai, Yi Li, K.H.J.S.: R-fcn: Object detection via region-based fully convolutional networks (2016), <https://arxiv.org/abs/1605.06409>
12. Joe Yue-Hei Ng, Matthew Hausknecht, S.V.O.V.R.M.G.T.: Beyond short snippets: Deep networks for video classification (2015), <https://arxiv.org/abs/1503.08909>
13. Joseph Redmon, A.F.: Yolo9000: Better, faster, stronger (2016), <https://arxiv.org/abs/1612.08242>
14. Joseph Redmon, A.F.: Yolov3: An incremental improvement (2018), <https://arxiv.org/abs/1804.02767>
15. Joseph Redmon, Santosh Divvala, R.G.A.F.: You only look once: Unified, real-time object detection (2015), <https://arxiv.org/pdf/1506.02640>
16. J.R.R. Uijlings, K.E.A. van de Sande, T.G., Smeulders, A.: Selective search for object recognition (2012), <http://www.huppelen.nl/publications/selectiveSearchDraft.pdf>
17. Kaiming He, Georgia Gkioxari, P.D.R.G.: Mask r-cnn (2017), <https://arxiv.org/abs/1703.06870>
18. Kaiming He, Xiangyu Zhang, S.R.J.S.: Deep residual learning for image recognition (2015), <https://arxiv.org/abs/1512.03385>
19. Kulkarni, S.: Video stabilization using feature point matching (2017), <https://iopscience.iop.org/article/10.1088/1742-6596/787/1/012017/pdf>
20. Lee, S., Kim, J., Shin Yoon, J., Shin, S., Bailo, O., Kim, N., Lee, T.H., Seok Hong, H., Han, S.H., So Kweon, I.: Vpgnet: Vanishing point guided network for lane and road marking detection and recognition. In: The IEEE International Conference on Computer Vision (ICCV) (Oct 2017), http://openaccess.thecvf.com/content_ICCV_2017/papers/Lee_VPGNet_Vanishing_Point_ICCV_2017_paper.pdf

21. Lowe, D.G.: Distinctive image features from scale-invariant keypoints (2004), <https://www.cs.ubc.ca/~lowe/papers/ijcv04.pdf>
22. Matthias Grundmann, Vivek Kwatra, I.E.: Auto-directed video stabilization with robust l1 optimal camera paths (2011), <https://static.googleusercontent.com/media/research.google.com/fr//pubs/archive/37041.pdf>
23. Miao Wang, Guo-Ye Yang, J.K.L.A.S.S.P.L.M.I.a.S.M.H.: Deep online video stabilization (2018), <https://arxiv.org/pdf/1802.08091.pdf>
24. Ross Girshick, Jeff Donahue, T.D.J.M.: Rich feature hierarchies for accurate object detection and semantic segmentation (2013), <https://arxiv.org/abs/1311.2524>
25. Rui Fan, N.D.: Real-time stereo vision-based lane detection system (2018), <https://arxiv.org/abs/1807.02752>
26. Saining Xie, Ross Girshick, P.D.Z.T.K.H.: Aggregated residual transformations for deep neural networks (2016), <https://arxiv.org/abs/1611.05431>
27. Shaoqing Ren, Kaiming He, R.G.J.S.: Faster r-cnn: Towards real-time object detection with region proposal networks (2015), <https://arxiv.org/abs/1506.01497>
28. Staravoitau, A.: Detecting road features (2017), <https://navoshta.com/detecting-road-features/>
29. Yann LeCun, Léon Bottou, Y.B., Haffner, P.: Gradient-based learning applied to document recognition (1998), <http://yann.lecun.com/exdb/publis/pdf/lecun-01a.pdf>