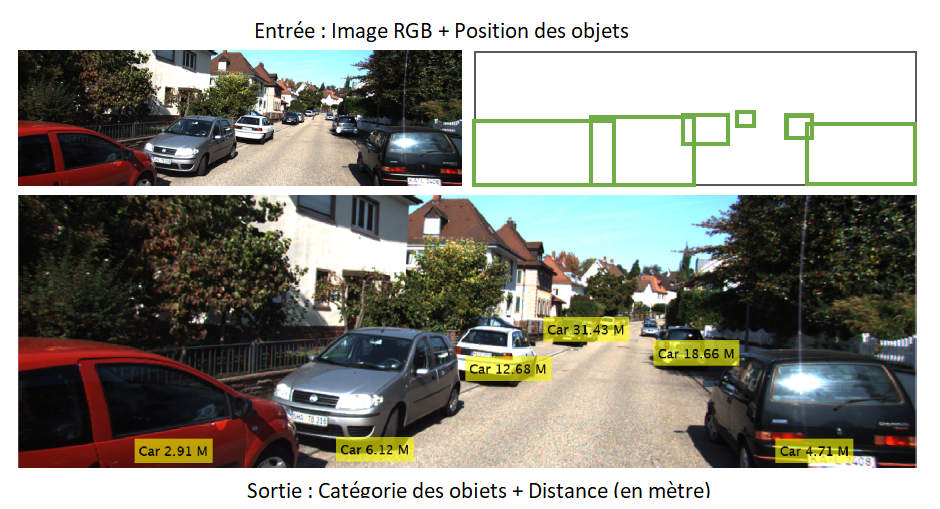
Bibliographie :

Learning Object-Specific Distance from a Monocular Image - 2019

Jing Zhu, Yi Fang, Husam Abu-Haimed, Kuo-Chin Lien,Dongdong Fu,Junli Gu



**Introduction :**

Depuis plusieurs années la vision par ordinateur n’a eu cesse de se développer, encore plus avec l’accroissement des techniques de Machine Learning. Dans le domaine des voitures autonomes les tâches les plus populaires sont la détection et la classification des objets. Cependant l’estimation de la distance, une tâche pourtant nécessaire afin d’éviter des obstacles, a été plus négligé que les autres.

L’une des méthodes principales pour estimer la distance avec des obstacles se base sur l’inverse perspective mapping, une méthode efficace mais peu précise dans les cas où l’obstacle se situe sur les côtés ou à une distance supérieure à 40m.

L’objectif de cette étude est donc de mettre en place une méthode d’estimation de distance d’obstacle basé sur le Machine Learning, fiable pour n’importe quel type d’image, plus précise et plus efficace que les techniques déjà existantes.

**Objectifs :**

* Mettre en œuvre une méthode d’estimation de distance automatique à partir d’image unique sans annotations
* Développer un estimateur de coordonnées (XYZ) pour chaque objet afin d’obtenir des résultats plus précis
* Dataset adapté à l’entrainement d’algorithme d’estimation de distance

**Travaux associés :**

Pour ce qui est de l’estimation de distance en machine learning il y a déjà eu des travaux basés sur la machine learning aussi, cependant ces derniers se concentraient principalement sur la classification et la géométrie des boites d’ancrage propre à chaque objet contrairement à l’objectif des chercheurs pour cette étude où l’objectif est de pouvoir déterminer la distance par rapport à la caméra seulement grâce à des features propres à chaque objet et pas seulement la boite d’ancrage.

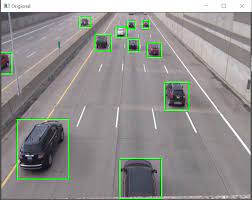


Figure : Boite d’ancrage en vert

**Leur approche :**

Leur modèle de machine learning est composé de la manière suivante :

Feature extractor :

Dans un premier temps un feature extractor qui va venir extraire les caractéristiques de l’image.

Les feature extractor utilisés dans l’étude sont VGG16 et RES50 deux réseaux de neurones convolutionnels.

On vient ensuite injecter la sortie de ces feature extractor dans une de ROI Pooling afin d’avoir en sortie un vecteur de taille fixe afin de pouvoir le mettre en input du Classifier et du Distance estimator.

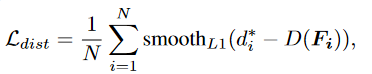
Distance estimator :

L’estimateur de distance prédit la distance en mètres (coordonnée Z), il est composé de 3 couches de neurones FC (avec une fonction d'activation softplus sur la dernière couche) et une sortie (la distance Z).

Il a une fonction d'activation softplus sur la sortie de la dernière couche pour s'assurer qu'il s'agit d'une distance positive.

FC : Fully connected : entièrement connecté

**Fonction de perte :**



N = Nombre d’objet

di = Distance réelle

D(Fi) = Distance prévue

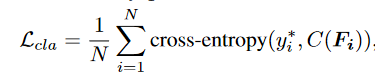
x = di- D(Fi)

La fonction de perte SmoothL1 est utile car elle est stable est elle est continue. Elle permet d’éviter que les gradients de la fonction ne deviennent trop haut.

Classifier :

Le classificateur est composé d’une couche entièrement connectée (avec un nombre de neurones égal au nombre de catégories dans l'ensemble de données) et une fonction SoftMax (probabilités pour chaque catégorie d'être vraie).

**Fonction de perte :**



Yi = Catégorie

C(Fi) = Sortie du classificateur

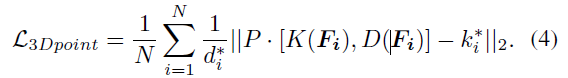
## Cross-entropy = *L* = Σ−yi\*log(C(Fi))

Keypoint Regressor :

Le keypoint regressor prédit la position X et Y des objets présents dans la scène. Pour cela ils ont mis en place un réseau de neurone composé de 3 couches FC.

Pour entrainer leur réseau de neurone à déterminer les coordonnées 2D (X et Y) ils ont récupéré une partie des database utilisés lors de cette étude et ont recalculé eux-même les coordonnées X et Y des objets dans les différentes scènes (grâce à une matrice de passage 2D vers 3D).

**Fonction de perte :**



La fonction de perte utilisée pour le keypoint regressor effectue la différence entre position réelle des objets avec la position prédite.

Optimizer :

Les auteurs ont utilisé ADAM Optimizer afin d’obtenir les paramètres optimaux pour l’entrainement du réseau de neurones (taux d’apprentissage).

**Dataset :**

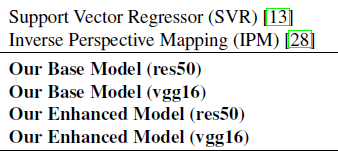
Les deux datasets utilisé dans cet article sont destiné principalement à l’entrainement d’algorithmes de Machine Learning dédié à la conduite autonome

**NuScenes :** Dataset composé de plus de 1000 vidéo annotée des rues de Singapour et Boston. (Distance, catégorie des objets, boite d’ancrage)

**KITTI :** Dataset composé de vidéo prise dans la ville de Karlsruhe (Allemagne) avec plusieurs annotations (profondeur, odométrie, catégorie des objets, boite d’ancrage).

Evaluation des réseaux de neurones :

Pour évaluer l’efficacité de leur méthode ils ont comparé leur modèle (avec et sans keypoint regressor) avec l’inverse perspective mapping et le support vector régression qui sont tous les deux des méthodes d’apprentissage supervisé.

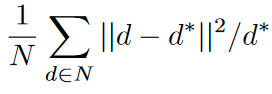


On observe un **base model** et un **enhanced model**.

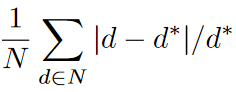
Le base model est le réseau de neurones sans le keypoint regressor et le enhanced model (modèle amélioré) et le réseau de neurones (classificateur + estimateur de distance) avec en plus le keypoint regressor pour ajouter de la précision.

Afin de les comparer avec leur méthode ils ont utilisé divers types d’erreurs :

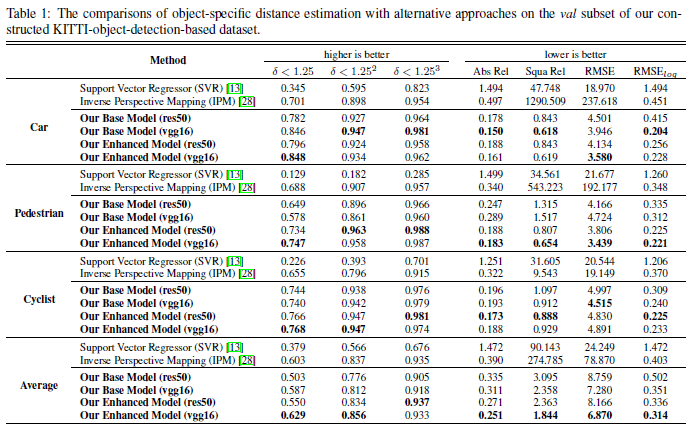
**RMSE** (Root Mean Square Deviation) : Indique à quel point vous êtes proche de l'ajustement des données (plus il est faible, mieux c'est)



**absolute relative difference:**



RESULTAT :



**Pour le dataset KITTI :**

Nous pouvons voir qu'en moyenne le modèle utilisant utilisant vgg16 comme feature extractor est le meilleur alors que celui qui a le plus mauvais résultat est la méthode SVR.

Nous pouvons également noter que la méthode IPM a un résultat similaire à leur modèle.

**Pour la voiture :**

La méthode ayant les meilleurs résultats en moyenne est le modèle de base (vgg16) mais la méthode améliorée (vgg16) a un résultat très similaire.

Tous les modèles de l'étude ont de très bons résultats alors que de SVR et IPM ont des résultats très inconsistants.

Par exemple, le RMSE est d'environ 3.5 pour les 4 modèles de l'étude mais l'IPM a un RMSE de 237 et le SVR de 19.

**Pour le piéton :**

La méthode ayant les meilleurs résultats en moyenne est le modèle amélioré (vgg16) et avec les piétons, nous pouvons voir une nette différence entre le modèle amélioré et le modèle de base.

La méthode SVR a un très mauvais résultat mais l'IPM a un résultat similaire au modèle de base.

**En moyenne :**

Le modèle amélioré à de meilleurs résultats en moyenne que le modèle de base mais cela dépend de l'obstacle. Le modèle amélioré est meilleur avec les piétons et les cyclistes mais a des résultats similaires pour les voitures.

Il est également important de noter que leur modèle a été entraîné avec une partie des jeux de données KITTI et nuScenes.

**CONCLUSION :**

Un des problèmes principaux pour mettre en place une méthode d’estimation de distance en machine learning est le manque de dataset. Les auteurs de l’étude ont réussi à outrepasser ce problème en améliorant les datasets KITTI et nuScenes afin d’y ajouter les coordonées X et Y et ainsi pouvoir entrainer leur keypoint regressor. On peut d’ailleurs observer que le modèle amélioré a de meilleurs résultats en moyenne.

L’intérêt de cette étude est le Keypoint regressor qui nous donnerait les coordonées X, Y et Z. Ainsi la calibration de caméra pour être effectuée.