UNIVERSITÉ DE SHERBROOKE
Faculté de génie
Département de génie électrique et de génie informatique

Masquage sémantique d'instances pour SLAM visuel dans des environnements dynamiques

Mémoire de maitrise
Spécialité : génie électrique

Jonathan Vincent

Sherbrooke (Québec) Canada

Juin 2019

MEMBRES DU JURY

François Michaud

Directeur

Pier-Marc Comtois-Rivet

Évaluateur

François Ferland

Évaluateur

RÉSUMÉ

Depuis quelques années, l'autonomie véhiculaire devient un sujet de plus en plus étudié. Présentement, les véhicules autonomes se localisent principalement par l'entremise de systèmes GPS (*Global Positioning System*). Toutefois, ces systèmes ne sont pas accessibles dans tous les types d'environnements (forêts, villes, décombres, neige, sous-terrain, etc.) et ne permettent pas de pouvoir réagir ou interagir avec l'environnement (arbres, chaises, voitures, etc.). Une alternative au GPS est l'utilisation de techniques visuelles pour faire de la localisation, de la même manière que l'humain le fait en percevant l'environnement avec ses yeux. C'est pour cette raison que la vision artificielle est de plus en plus utilisée dans ce type de systèmes mobiles. De plus, les derniers développements dans les GPU (*Graphical Processing Unit*) ont permis aux technologies d'apprentissage machine, plus particulièrement d'apprentissage profond, de faire des avancées technologiques importantes en ce qui concerne la vision artificielle.

Un des problèmes propres à la localisation visuelle est d'arriver à distinguer les objets dynamiques de l'arrière-scène afin d'être indépendante de leur présence lors de la cartographie. Ce projet de recherche propose d'utiliser un algorithme d'apprentissage profond pour segmenter les images de façon sémantique, permettant ainsi l'identification, le suivi et le masquage d'objets dynamiques afin d'améliorer la localisation et la cartographie pour du SLAM visuel. L'approche ISM-vSLAM (*Instance Semantic Masking for visual SLAM*) a donc été développée dans le cadre de cette maitrise. L'architecture d'ISM-vSLAM se sépare en trois sections : 1) la segmentation et la localisation des objets dynamiques à partir de l'image RGB d'entrée du système ; 2) la création et la mise jour de filtres de Kalman étendus pour chaque instance d'objets ; et 3) la gestion des instances sur les entrées de l'approche de vSLAM, et plus spécifiquement RTAB-Map utilisé pour la validation expérimentale de l'approche.

Les résultats obtenus sur les séquences du jeu de données du *Technical University of Munich* (TUM) [44] ainsi que nos tests en laboratoire suggèrent qu'ISM-vSLAM est robuste autant dans des environnements statiques que hautement dynamiques. De plus, les algorithmes développés sont basés sur l'intergiciel ROS (*Robot Operating System*), ce qui facilite l'utilisation et la maintenance de l'algorithme. Des applications utilisant ces informations sémantiques pourront donc être créées afin de permettre aux plateformes robotiques autonomes d'accroître leur éventail de fonctionnalités dans des environnements dynamiques, que ce soit pour améliorer la localisation, la planification de trajectoires ou encore pour interagir avec l'environnement.

Mots-clés : Intelligence artificielle, Apprentissage profond, Suivi spatial, Filtre de Kalman étendu, SLAM (*Simultaneous Localization and Mapping*)

À ma douce-moitié, Fanny, ainsi qu'à mon fils, Isaac.

REMERCIEMENTS

Mes recherches furent possibles grâce au soutien financier de l'IVI (Institut du véhicule innovant), de MITACS et de l'Université de Sherbrooke.

Je souhaite remercier mon directeur de maîtrise, François Michaud, pour son soutien et sa confiance qui m'ont permis d'atteindre mes objectifs tout au long de mon projet. Je tiens également à remercier l'IVI qui m'a intégré au sein de son équipe afin de développer mes compétences et qui m'ont fourni toute l'aide nécessaire, autant matériel que technique, pour la réalisation de mon projet. Un grand merci également à l'équipe du groupe de recherche IntRoLab pour leur soutien et leur aide quant à la réalisation de mes travaux.

Finalement, je tiens à exprimer mes plus sincères remerciements à ma douce moitié qui a su me soutenir et m'encourager tout au long de ma maitrise, et ce même avec la venue de notre futur enfant.

TABLE DES MATIÈRES

LISTE DES FIGURES

LISTE DES TABLEAUX

LISTE DES ACRONYMES

Acronyme	**Définition**
3D	Trois dimensions
2D	Deux dimensions
AP	Apprentissage profond
CCD	*Charge Coupled Device*
CCTT	Centre collégial de transfert technologique
CNN	*Convolutional Neural Network*
DARPA	*Defense Advanced Research Projects Agency*
GPU	*Graphical Processing Unit*
IA	Intelligence artificielle
ISM	*Instance Semantic Masking*
ISM-vSLAM	*Instance Semantic Masking for visual Simultaneous Localization and Mapping*
IVI	Institut du véhicule innovant
LIDAR	*Light Detection And Ranging*
RADAR	*RAdio Detection And Ranging*
RNN	*Recurrent Neural Network*
ROS	*Robot Operating system*
RTAB-Map	*Real-Time Appearance-Based Mapping*
RVIZ	*ROS VIZualisation*
SLAM	*Simultaneous Localization and Mapping*
TUM	*Technical University of Munich*
UdeS	Université de Sherbrooke
vSLAM	*visual Simultaneous Localization and Mapping*

CHAPITRE 1

Introduction

Depuis quelques années, l'autonomie véhiculaire est un sujet qui a pris de plus en plus d'importance et qui est au cœur de l'actualité. Ce sujet a connu un fort engouement depuis la première compétition de véhicules autonomes de DARPA, soit le *DARPA Grand Challenge* de 2004. Le but de cette compétition était d'accélérer le développement des véhicules autonomes pour permettre leur utilisation dans des applications militaires telles que des convois en zones dangereuses. Lors de cette compétition, aucun véhicule n'a réussi à faire le parcours de 225 kilomètres en entier. En effet, le meilleur véhicule a parcouru 13 kilomètres, ce qui représente seulement 5% de la distance totale. Malgré ces résultats, la compétition a tout de même été un succès puisque celle-ci a marqué le début d'une nouvelle ère pour les véhicules autonomes. De nouvelles technologies ont été développées pour répondre aux limitations observées, ce qui a permis au deuxième *DARPA Grand Challenge* de 2005 d'être un succès fulgurant. Lors de cette compétition, 23 équipes ont parcouru une distance plus grande que celle du gagnant de 2004 et cinq équipes ont réussi à finir la course dans le délai imposé de 10 heures.

Bien que ces événements aient défini les bases de l'autonomie véhiculaire, il reste encore beaucoup de travail à accomplir avant que les véhicules autonomes fassent partie de notre quotidien. Présentement, les véhicules autonomes se localisent principalement par des systèmes GPS (*Global Positioning System*). Toutefois, ces systèmes ne sont pas accessibles dans tous les environnements (forêts, villes, décombres, neige, sous-terrain, etc.) et ne permettent pas de pouvoir réagir ou interagir avec l'environnement (arbres, chaises, voitures, etc.). Une alternative au GPS est l'utilisation de techniques visuelles pour faire de la localisation, de la même manière que l'humain le fait en percevant l'environnement avec ses yeux. L'utilisation d'images visuelles pour la localisation et la cartographie simultanées (SLAM) peut fournir un large éventail d'informations sur l'environnement [14], par exemple en détectant les couleurs à de hautes résolutions (sémantique des objets, composition de la surface, orientation des objets, etc.)

Les techniques SLAM visuels (vSLAM) fonctionnent bien dans les environnements statiques en permettant d'extraire des caractéristiques visuelles stables à partir d'images. Cependant, dans les environnements avec des objets dynamiques (personnes, voitures, animaux, etc.), les performances diminuent de manière significative, car les caractéristiques

visuelles peuvent provenir de ces objets, ce qui rend la localisation moins fiable [14]. Par exemple, dans le contexte d'un robot mobile d'intérieur travaillant dans un bureau dans lequel des humains et des chaises sont rencontrés, il est peu probable que ces objets ne restent à un emplacement exact, ni entre les images ni entre les boucles. Par conséquent, les caractéristiques visuelles de ces éléments doivent être ignorées afin d'éviter toute dérive d'odométrie, ainsi que des artefacts indésirables dans la carte.

Les derniers développements dans les GPU (*Graphical Processing Unit*) ont permis aux technologies d'apprentissage machine, plus particulièrement d'apprentissage profond, de faire des avancées importantes surtout au niveau de la vision artificielle. Cela a permis le développement de nouvelles techniques telles que les CNN (*Convolutional Neural Network*), RNN (*Recurrent Neural Network*), etc. [17]. Ces nouvelles technologies, étant capables d'accomplir des tâches auparavant limitées à l'humain à cause de leur complexité, présentent beaucoup de potentiel pour les systèmes robotiques autonomes.

L'objectif du projet de recherche est donc de répondre à la question suivante : Est-ce que l'utilisation d'un réseau de neurones artificiels en amont d'un système de SLAM visuel permettrait d'obtenir une localisation ainsi qu'une cartographie plus robuste et fiable pour des scènes dynamiques ?

Ce mémoire présente une nouvelle approche qui utilise un algorithme d'apprentissage profond et un algorithme de suivi pour masquer des objets dynamiques dans un pipeline vSLAM en utilisant une base de connaissances sémantique antérieure. Notre hypothèse de recherche est que les algorithmes d'apprentissage profond peuvent être utilisés pour segmenter sémantiquement les images, en permettant l'identification, le suivi et la suppression d'objets dynamiques des scènes, afin d'améliorer à la fois la localisation et la cartographie d'un algorithme de vSLAM. L'architecture du réseau de neurones utilisée dans cet article est le MASK R-CNN [16], développé pour la segmentation d'instances, et l'algorithme de suivi est un filtre de Kalman étendu. Notre approche utilise la sortie de la segmentation sémantique pour créer un masque d'objets dynamiques basé sur des connaissances préalables et sur l'algorithme de suivi. Ce masque est ensuite utilisé pour filtrer les caractéristiques visuelles à l'entrée d'un algorithme de vSLAM. Ce faisant, l'approche vise à fournir trois avantages : 1) une meilleure performance de l'odométrie visuelle ; 2) une carte sans objets dynamiques ; 3) l'augmentation du nombre et de la qualité des détections de boucles pour la cartographie visuelle.

Le projet est fait en collaboration avec l'Institut du Véhicule Innovant (IVI), un Centre collégial de transfert technologique (CCTT) œuvrant dans le domaine véhicules électriques

et autonomes. L'IVI a récemment démarré des activités sur l'automatisation des transports dans le but de créer une valeur ajoutée aux PME du Québec et du Canada. Ce projet vise ainsi le transfert technologique vers les entreprises car tous les procédés et les instruments développés lors de ce projet seront directement accessibles pour les PME qui feront appel à l'expertise de l'IVI. Ce projet donnera également accès à de nouveaux marchés pour l'intelligence véhiculaire : les technologies actuelles ne fonctionnant pas dans des environnements dynamiques, celles-ci limitent les véhicules autonomes dans leur utilisation. Ce projet cherche donc à mitiger ces limitations tout en permettant d'accomplir des tâches qui leur étaient auparavant impossibles.

Le document est organisé de la façon suivante. Le chapitre 2 porte sur la revue de l'état de l'art sur les véhicules autonomes, les cartes sémantiques et l'apprentissage profond est effectuée. Le chapitre 3 présente l'approche proposée par l'entremise d'un article soumis dans le cadre de cette maîtrise. Finalement, le chapitre 4 fait un retour sur l'ensemble du projet et les suites possibles.

CHAPITRE 2

Revue des techniques de vision artificielle et de cartographie appliquée à la robotique autonome

Ce chapitre a pour but de situer le projet de recherche proposé par rapport aux travaux publiés dans le domaine de la cartographie et de la vision artificielle appliquée à la robotique mobile. Pour ce faire, le chapitre est divisé en quatre sections. Tout d'abord, les plateformes robotiques mobiles sont étudiées pour comprendre leur fonctionnement ainsi que leurs limitations. Par la suite, les différents capteurs utilisés sur ce type de plateformes sont analysés et comparés. Troisièmement, une revue de différentes techniques de pointe en vision artificielle est effectuée pour définir les meilleures techniques utilisables dans le cadre du projet de recherche. Finalement, une revue des systèmes de SLAM utilisés dans des systèmes robotiques autonomes est abordée pour définir les techniques les plus avancées ainsi que leurs limitations.

2.1 Plateformes robotiques autonomes

La notion de robot mobile autonome a beaucoup évolué depuis les dernières décennies, passant d'un chariot mobile jusqu'à une plateforme humanoïde. Malgré ces différentes instantiations une constante demeure, en ce sens qu'un robot mobile autonome est un système équipé d'un ordinateur qui prend des décisions et interagit avec l'environnement en fonction des différentes données provenant de capteurs [33, 50, 25].

2.1.1 Véhicules autonomes

Les véhicules robotiques ou véhicules autonomes ont connu des avancées significatives à la suite des compétitions DARPA. Tel que mentionné dans [13], les véhicules autonomes présentent beaucoup de potentiel. En effet, ceux-ci permettront d'augmenter la sécurité routière, de diminuer les coûts de déplacement, de diminuer la congestion routière, de réduire les besoins de stationnement et de permettre à des personnes inaptes de se déplacer plus facilement. D'après [3], 90% des accidents de la route aux États-Unis sont causés par les conducteurs. Depuis 2010, les avancées dans le domaine ont rapidement pris de l'am-

pleur, notamment à cause de l'intérêt grandissant des producteurs automobiles ainsi que des compagnies d'informatique qui ont axé leurs recherches sur ce domaine [51]. D'après un article de *Business Insider* [37], 19 compagnies prévoient avoir des véhicules autonomes disponibles pour le grand public d'ici 2020. Leur grande popularité permet un avancement significatif des technologies qui y sont associées de près ou de loin.

La compétition *Darpa Urban Challenge* a démontré que la navigation autonome dans des villes est possible. Les véhicules autonomes qui ont participé à celle-ci ont établi des bases pour le développement de futurs véhicules autonomes [36, 5, 6]. La figure 2.1 présente l'architecture logicielle standard d'un véhicule autonome, et le tableau 2.1 compare les composants des deux véhicules qui sont respectivement arrivés premier et deuxième lors de la compétition *DARPA Urban challenge*.

2.1.2 Véhicules hors route autonomes

Les véhicules hors route autonomes sont une branche connexe des véhicules autonomes, mais dans des environnements non contrôlés. Tels que décrit dans [22], les véhicules hors route autonomes présentent le plus grand défi de la navigation autonome. Ils peuvent utiliser un GPS pour se déplacer de façon autonome à l'extérieur. Par contre, ceux-ci sont limités dans leur champ d'application en raison de nombreux environnements qui ne permettent pas l'utilisation de tels systèmes de géolocalisation, tels que les forêts, les décombres, les sous-terrains, etc. Afin de répondre à cette problématique, l'utilisation de la vision s'avère la piste la plus prometteuse [4]. Dans une recherche dirigée par l'Institut d'aérospatial de l'Université de Toronto (UTIAS) [39], différentes technologies de navigation basées sur la vision dans des environnements hostiles sans GPS ont été testées. Plus particulièrement, leur système de VT&R (*Vision-based Teach and Repeat*) a été éprouvé dans trois contextes différents, soit dans l'environnement de test martien de l'Agence spatiale canadienne (CSA), dans un champ enneigé à l'UTIAS, ainsi que dans une prairie enneigée également à l'UTIAS. Suite à l'analyse des résultats obtenus, les auteurs ont constaté que pour mettre en fonction des systèmes fiables fonctionnant dans ces types d'environnement, certains problèmes liés à la vision doivent être résolus. Ces problèmes sont principalement liés à la lumière (ombres) ainsi qu'au manque de zones d'intérêt (*features*) dans l'environnement.

Il existe actuellement quelques véhicules hors route autonomes sur le marché. Deux exemples sont illustrés à la figure 2.2. Le Husky est une plateforme de développement créée sur mesure par Clearpath Robotics. Le S5 est, quant à lui, un robot de sécurité autonome développé par la compagnie SMP Robotics.

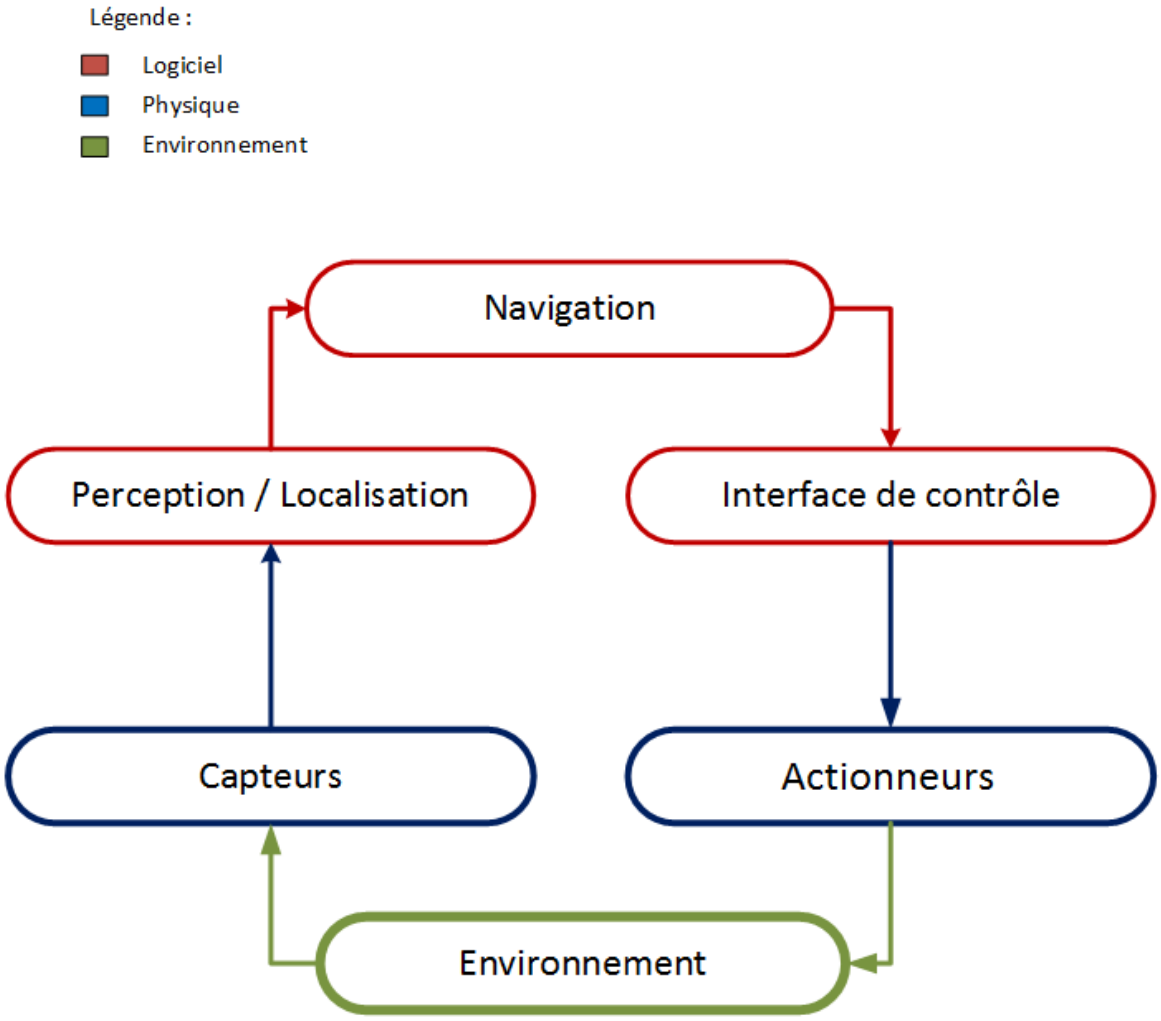
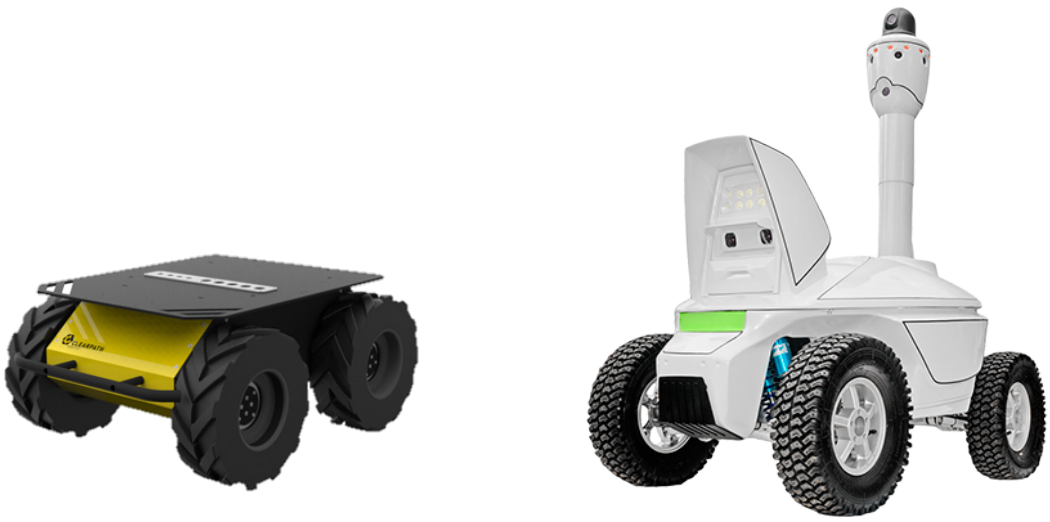


Figure 2.1 Schéma général de l'architecture d'un véhicule autonome

Tableau 2.1 Gagnants de la compétition Darpa Urban Challenge

Compétiteur	**Boss [6]**	**Junior [36]**
Système de contrôle	– Dix Processeurs 2.16-GHz Core2Duo	– Deux ordinateurs Intel Quad-core
Capteur passif	– Point Grey Firefly Camera 2D	– Aucun capteur passif
Capteur actifs	– Velodyne HDL-64 LIDAR – Continental ISF 172 LIDAR – IBEO Alasca XT LIDAR – Continental ARS 300 Radar – SICK LMS 291-S05/S14 LIDAR – Applanix POS-LV 220/420 GPS/IMU	– Velodyne HDL-64 LIDAR – IBEO Alasca XT LIDAR – SICK LMS 291-S05/S14 LIDAR – Applanix POS-LV 220/420 GPS/IMU – RIEGL LMS-Q120 LIDAR
Actionneurs	– Drive-by-wire	– Drive-by-wire



a: Grizzly (http ://www.clearpathrobotics.com)

b: S5 (http ://smprobotics.com)

Figure 2.2 Exemple de plateformes robotiques hors route autonomes

Les technologies utilisées dans les véhicules hors routes autonomes sont pratiquement les mêmes que celles disponibles pour les véhicules sur route autonome. Cela a pour avantage de donner accès à des technologies de pointe aux véhicules hors route. En effet, beaucoup moins de développements sont effectués sur les véhicules hors route autonomes. Ce type de système profite donc de l'engouement des véhicules routiers autonomes.

2.2 Capteurs

Les capteurs sont une partie essentielle de toutes plateformes autonomes. Ceux-ci servent à la perception du système. La perception permet de détecter les éléments importants dans l'environnement tels que les véhicules, les piétons, les arbres, les débris ainsi que tout autre détail important dans le contexte d'utilisation de la plateforme. Au niveau de celle-ci, deux problèmes peuvent survenir, soit des faux négatifs ou des faux positifs. Un faux négatif est le fait de ne pas détecter un élément alors qu'un faux positif est le fait d'en percevoir un alors qu'il n'est pas réellement présent. En fonction de l'application, l'un peut être préférable à l'autre. Par exemple, dans un véhicule sur l'autoroute, un faux positif peut être plus dommageable qu'un faux négatif. Un faux négatif effectué sur un petit objet est beaucoup moins dangereux qu'un faux positif qui pourrait forcer l'arrêt du véhicule et causer un carambolage. À l'inverse, un véhicule dans des rues résidentielles qui fait un faux négatif lorsqu'un enfant traverse la route est beaucoup plus grave qu'un faux négatif à une vitesse de 50 km/h où les véhicules derrière ont le temps de réagir. Dans des systèmes de perception complexes, il peut aussi y avoir des erreurs causées par une classification erronée. Par exemple, un piéton pourrait être reconnu comme un vélo. Si le système a été conçu pour réagir différemment en fonction du type de l'objet classifié, cela peut causer des accidents. Un autre type d'erreurs qui peut survenir est le défaut de fonctionnement du capteur. Que ce soit un défaut mécanique ou logiciel, ce type de défaut peut souvent être plus facile à détecter ou à contrer, par exemple en utilisant des protections logiciels ou bien de la redondance.

2.2.1 LIDAR

Un LIDAR (*LIght Detection And Ranging*) est un télémètre basé sur la lumière. Celui-ci envoie un pulse lumineux et mesure le temps avant que le faisceau revienne [7] pour définir la distance le séparant des obstacles. Ceux-ci ont été les premiers capteurs utilisés à grande échelle sur les véhicules autonomes. Tel que mentionné dans [5], ces capteurs dits actifs (ils émettent des ondes dans l'environnement) donnent de l'information beaucoup plus facile à traiter. En effet, les LIDARs donnent une information de distance directement utilisable à

l'instar des caméras qui donnent une information de couleur qui est beaucoup plus difficile à traiter. Les LIDARS ont été les capteurs de prédilection lors des différentes compétitions DARPA. Ceux-ci représentent l'environnement sous forme d'un nuage de points en 3D. La figure 2.3 illustre un nuage de points 3D.

La figure 2.4 illustre deux examples de LIDARs, soit le Velodyne H64 et le Quanergy S3. Tels que recensés dans [49, 4, 36, 5], les avantages et les inconvénients des LIDARs sont les suivants :

Avantages :

- La représentation de l'environnement étant un nuage de points, il est facile de faire de la séparation en profondeur sur les différents objets qui s'y retrouvent.
- Puisque ce type de capteur est actif, il peut fonctionner nuit et jour sans que sa fiabilité soit affectée.
- Il est très robuste aux interférences puisqu'il utilise seulement une longueur d'onde précise.
- Il possède un résolution beaucoup plus grande que celle des RADARs.

Inconvénients :

- Actuellement, ce type de capteurs coûte très cher. Par exemple, le Velodyne HDL-64 qui est couramment utilisé coûte approximativement 75 000 USD. Par contre, la forte demande pour des capteurs de ce type et à faible coût a favorisé le développement de cette technologie. Par exemple, le OS-1-16 de Ouster, qui est arrivé sur le marché en mars 2018, est un LIDAR à 16 faisceaux qui est disponible au coût de 3 500 USD.
- Bien que la résolution soit supérieure à celle des RADARs, elle reste faible comparée à d'autres capteurs (caméra).
- Ils ne voient pas les couleurs.
- La distance est limitée. Par exemple, le Velodyne H64 peut voir jusqu'à 60 mètres.
- Les LIDARs standard tels que le velodyne H64 possède des pièces mobiles. Ils sont donc plus susceptibles aux bris mécaniques.
- La fréquence est relativement basse. Un LIDAR standard tel que le Velodyne H64 possède une vitesse de rafraîchissement de 5 Hz. Cela occasionne de la distorsion lors des déplacements de la plateforme.
- Ceux-ci sont plus sensibles aux intempéries (pluie, neige, brouillard) que les caméras et peuvent voir des choses invisibles telles que les gaz d'échappement des véhicules.
- Les LIDARs fonctionnent mieux lorsqu'ils sont installés à l'extérieur (ils ne doivent pas être installés derrière un pare-brise).

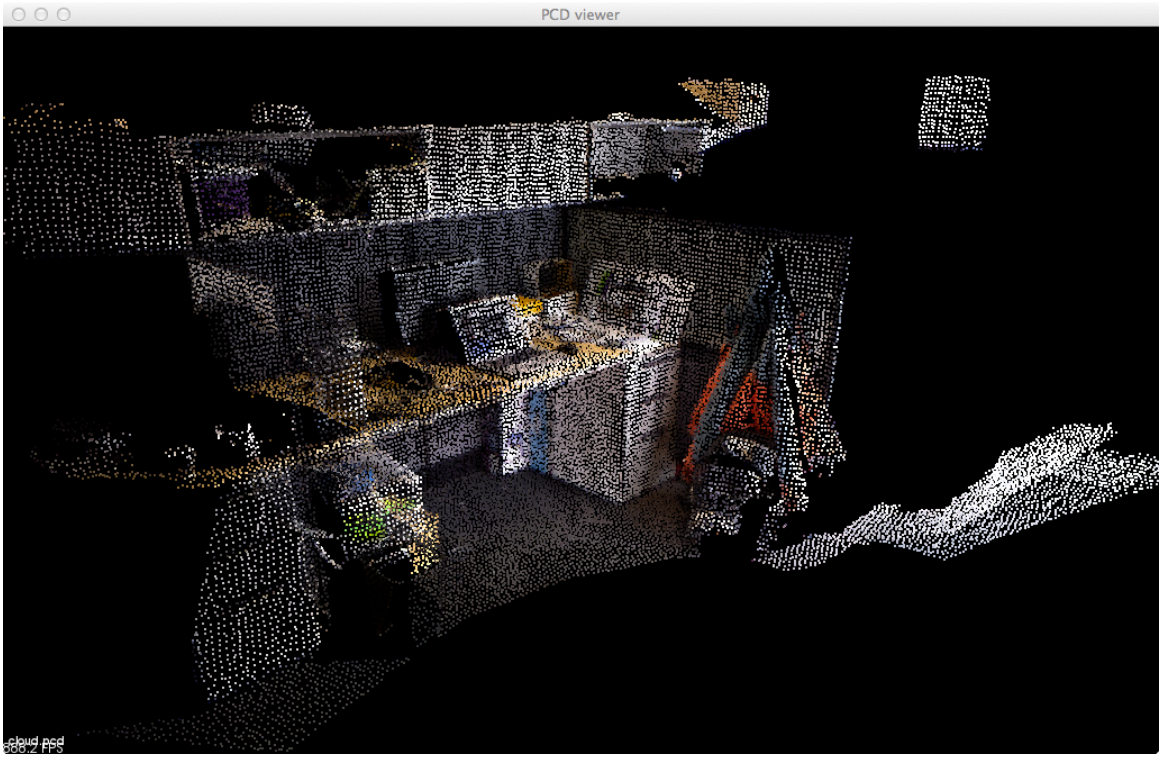


Figure 2.3 Nuage de points 3D (https ://github.com/introlab/rtabmap)

a: Velodyne H64 (http ://velodynelidar.com) b: OS-1-16 (https ://www.ouster.io)

Figure 2.4 Exemples de capteurs LiDARs

2.2.2 Caméra

Les caméras sont des capteurs qui ressemblent davantage à l'œil humain. La lumière passe au travers de lentilles pour venir exciter un capteur CCD (*Charge Coupled Device*). Chaque pixel du CCD atteint un niveau d'énergie différent en fonction de sa position, ce qui créer une image. Une image 3D peut également être créée avec plusieurs caméras 2D (triangulation). Il existe également des caméras spécialisées qui utilisent deux lentilles ou plus pour générer des images 3D, désignées comme étant des caméras stéréoscopiques.

La figure 2.5 illustre des caméras-types utilisées sur des véhicules autonomes. Les avantages et inconvénients de ce type de capteur [49, 4, 36, 5] sont les suivants :

Avantages :

- Ce type de capteur coûte peu cher, est facilement accessible et disponible en grande quantité. Par exemple, la caméra industrielle Firefly de la compagnie Point Grey 2.5 coûte environ 800 USD.
- Puisque ce sont des capteurs passifs (ils n'ont pas besoin d'émettre des ondes pour fonctionner), ceux-ci peuvent voir à de grandes distances en fonction de la résolution de la caméra et du type de lentille utilisé.
- Ils peuvent voir les couleurs, ce qui donne plus d'information sur l'environnement.
- Ils ne possèdent pas de pièces mobiles.
- Avec de très hautes résolutions ainsi que des couleurs, il est possible de retirer une quantité importante d'information sur l'environnement, ce qui facilite la compréhension de scènes.
- Il peuvent percevoir les détails créés par l'humain tels que les panneaux de signalisation et les lumières de circulation.
- Il est possible de faire du 3D avec une caméra stéréoscopique telle que la Zed de Stereolabs 2.5.

Inconvénients :

- Nécessite des algorithmes très avancés pour faire un bon traitement de l'énorme quantité d'information en temps réel.
- Ils sont affectés par les changements lumineux de l'environnement tels que les ombres et les sources lumineuses externes.
- Ils ont besoin d'illumination externe le soir pour fonctionner.

b: Stereolabs Zed (http ://www.stereolabs.com)

a: Pointgrey Firefly (http ://www.ptgrey.com)

Figure 2.5 Exemple de caméras utilisées sur des véhicules autonomes

2.3 Vision artificielle

La vision artificielle est une branche de l'intelligence artificielle qui a pour but de faire l'analyse d'images de façon autonome. Le traitement de l'information provenant d'une caméra peut être fait par différentes techniques. Les plus répandues sont les détecteurs de zones d'intérêts tels que le SIFT [32] et l'apprentissage profond [17].

2.3.1 Détecteur de zones d'intérêts

Le terme zone d'intérêts est utilisé pour faire référence au terme *feature* couramment utilisé en anglais. Ce terme fait référence à des caractéristiques dites intéressantes dans une image pour un problème précis. En fonction de ce qui est recherché, la complexité peut être de différents niveaux.

Il existe différentes technologies de détection de zones d'intérêts. Les premières à avoir fait leurs apparitions étaient basées sur la reconnaissance de contours. L'algorithme des opérateurs de Canny [8] est le premier à avoir vu le jour dans ce domaine. Cet algorithme est basé sur le gradient de l'image.

Par la suite, des techniques se basant sur des points d'intérêts ont fait leur apparition. La méthode la plus répandue est le détecteur de Harris [18]. Ce type de détecteurs se base sur des points spécifiques des contours, soit les coins. En effet, un coin correspond à un changement abrupte de la direction d'un contour. Ces points sont plus fiables que les contours et favorisent donc la répétabilité au niveau de la détection.

Finalement, les techniques basées sur les régions d'intérêts utilisent des descripteurs qui définissent le comportement du gradient à l'intérieur d'une certaine région. Cela permet

d'être fiable, peu importe la taille et la rotation de la région d'intérêt. Une région d'intérêt peut détecter des caractéristiques dans l'image qui seraient trop lisses pour être détectées avec les deux techniques présentées précédemment. La technique la plus répandue est le SIFT [32].

Toutefois, ces trois méthodes sont très sensibles aux changements lumineux ainsi qu'aux objets dynamiques (objets mobiles ou changeant de forme). C'est pourquoi différentes méthodes ont été développées pour remédier à ces problèmes. Les problèmes liés aux changements lumineux font référence aux différents niveaux d'éclairage d'une scène qui génèrent des ombres. En fonction de l'éclairage, celles-ci peuvent se déplacer rapidement et provoquer de faux positifs. Les ombres apparaissent plus particulièrement à l'aube et au crépuscule, mais apparaissent aussi lorsqu'il y a présence de sources lumineuses externes, peu importe le moment de la journée.

Un des procédés pour remédier à ce problème est le modèle de distorsion chromatique [29]. Celui-ci utilise les distorsions de couleur dans l'environnement pour y retirer de l'information. L'une de ces particularités est d'enlever les ombres pour obtenir de meilleurs résultats. Par contre, dans des environnements qui tendent vers un spectre de couleurs monochromatiques, les techniques utilisant ce type de procédé sont inefficaces [39]. Une autre technique intéressante est la PQRS *(Perceptual Qualitative Relation about Shadows)* [40]. Celle-ci consiste à utiliser l'information contenue dans les ombres au lieu d'essayer d'en faire abstraction. La plupart des méthodes actuelles essaient, en effet, d'éliminer les ombres en les considérant comme du bruit afin d'améliorer les résultats de navigation. Ce procédé est donc très intéressant puisque les ombres sont trop prédominantes dans les environnements où la lumière varie beaucoup, et ainsi ne peuvent être négligés [39].

2.3.2 Apprentissage profond

L'apprentissage profond est une branche de l'apprentissage-machine basée sur un ensemble d'algorithmes qui tentent de classifier des données avec un haut niveau d'abstraction [17]. Les techniques d'apprentissage profond ont permis de grandement améliorer les systèmes de pointe dans les domaines de la reconnaissance audio et visuelle.

Par exemple, les applications de ce type de technologie se retrouvent dans des systèmes tels que le système de surveillance de Camio (http ://www.camio.com) qui fait la détection d'objets et de personnes avec des réseaux de neurones. Celui-ci permet de donner des alertes aux usagers ou bien simplement faire des enregistrements axés sur des actions spécifiques

étiquetées (livraison de colis, quelqu'un qui sonne à la porte, un chien qui marche sur le terrain, etc.).

L'apprentissage profond correspond à l'utilisation de plusieurs neurones artificiels qui sont déployés en couches. Un neurone possède des poids, une fonction d'activation (sigmoïde, tanh, etc.) et un biais, comme l'illustre la figure 2.6. Chacune de ces couches ajoutées définit un niveau d'abstraction plus élevé que le précédent. Un grand réseau, tel que le GoogleNet [46], comporte 27 couches de neurones.

Ce type de système est défini comme étant de l'apprentissage supervisé, et nécessite des variables d'entrées X ainsi que des variables de sortie Y pour être en mesure d'apprendre une fonction qui fait le lien entre les deux soit, $Y = f(X)$. Le but de cette fonction est d'approximer Y pour un nouveau X. Une banque de données comporte des exemples d'entraînement qui viennent en paire : pour chaque entrée X la sortie désirée Y y est associée. Le système fait donc la prédiction de Y pour un X donné, et en fonction de l'exactitude de Y les poids sont modifiés. Cette modification est effectuée à l'aide de la descente de gradient. Cette méthode d'optimisation itérative cherche a minimiser une fonction de coût en essayant de mettre sa dérivée partielle en fonction de tous les poids à zéro. Donc plus la valeur prédite est proche du résultat escompté, moins les poids sont modifiés, et vice-versa. Lorsque le système réussi à prédire Y avec une précision désirée, l'entraînement se termine et le système est prêt à être déployé.

De manière plus spécifique, le développement d'un réseau de neurones se fait en quatre étapes :

- Créer l'architecture du réseau. L'architecture est primordiale dans le sens qu'elle définit la tâche du réseau. Par exemple, c'est dans l'architecture que la définition en pixel de la caméra utilisée doit être prise en compte. L'architecture définit la couche d'entrée du réseau (la couche qui reçoit directement les données), les couches cachées (les couches qui font le traitement sur les données) et finalement la couche de sortie (résultat de classification). Beaucoup d'architectures différentes existent et la plupart des nouveaux réseaux se basent en partie sur des réseaux déjà existants, comme le GoogleNet.
- Définir un ensemble de données d'entraînement. Les données d'entraînements servent à entraîner le réseau de neurones. Ceci est l'une des faiblesses de ce type de système. Pour entraîner un réseau et avoir de bons résultats, il faut une quantité importante de données d'entraînement. En fonction de la tâche voulue du réseau, cela peut être difficile ou simplement impossible d'acquérir assez de données pour être en mesure

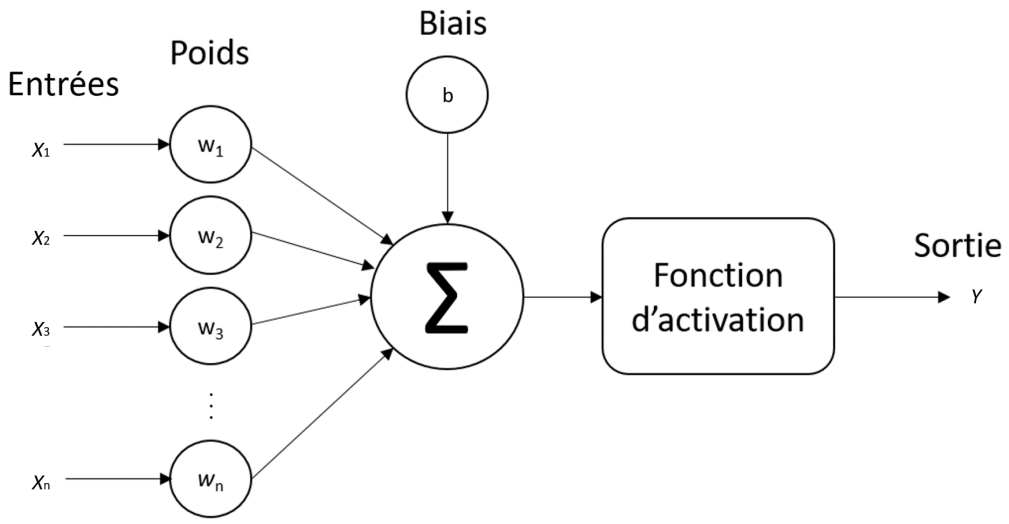


Figure 2.6 Schéma d'un neurone artificiel

d'entraîner un réseau. Heureusement, l'apprentissage profond connaît un engouement très important depuis quelques années et les bases de données d'entraînement ne cessent d'augmenter. Des bases de données créées à partir d'environnements virtuels font également leur apparition, telles que Synthia (http ://synthia-dataset.net/). Une autre grande avancée dans les réseaux de neurones artificiels sont les réseaux adverses génératifs plus connus sous le diminutif GANs (Generative Adversarial Networks). Ce type d'architecture utilise deux réseaux de neurones en compétition un contre l'autre dans un scénario de la théorie des jeux. C'est-à-dire que le premier réseau, nommé générateur, crée des données qui s'apparentent aux données provenant de la base de données initiale et le deuxième réseau, nommé le discriminateur, tente de prédire si l'information qu'il reçoit provient du générateur ou de la base de données. Il est donc possible d'entraîner un réseau de neurones avec beaucoup moins de données. Sachant que l'une des tâches les plus difficiles dans l'entraînement d'un réseau de neurones est la capacité à acquérir des données pour l'entraînement, ce type de réseau est très étudié dans le secteur académique. Toutefois, de par leur nature, les GANs ont beaucoup de problèmes de convergence rendant leurs utilisations difficiles.

– Entraîner le réseau. L'entraînement des réseaux de neurones est une tâche qui requiert une énorme puissance de calculs. Avec les dernières avancées dans le domaine des GPUs, il est possible d'avoir des ordinateurs relativement peu coûteux qui sont capables de faire des entraînements de qualité.
– Après quelques ajustements sur l'architecture du réseau (en itérant toutes les étapes), celui-ci est prêt à être déployé. Le déploiement est l'une des forces de l'apprentissage profond. En effet, lorsqu'un réseau est entraîné, il demande beaucoup moins de

puissance de calcul pour être utilisé en pratique. De plus, avec l'engouement autour des véhicules autonomes, plusieurs compagnies telles que la compagnie NVIDIA [1] fabriquent des systèmes embarqués capables d'implémenter des réseaux de neurones déployés, et ce, en consommant peu d'énergie. La gamme de systèmes embarqués Jetson de NVIDIA est un très bon exemple de ce type de systèmes.

Différents logiciels permettent la création de réseau de neurones. Chacun d'eux a des avantages et des inconvénients, comme le présente le tableau 2.2.

L'apprentissage profond présente beaucoup de potentiel de développement et des compagnies telles que Google, Nvidia, Microsoft, AMD travaillent activement au développement de cette technologie.

2.4 SLAM

Le SLAM (*Simultaneous Localization and Mapping*) est le nom donné aux techniques qui visent à gérer le problème de la construction et de la mise à jour d'une carte d'un environnement inconnu tout en localisant un agent à l'intérieur de celle-ci. Le SLAM fut popularisé lors de la compétition *DARPA Grand Challenge* de 2004 [36]. Plusieurs capteurs peuvent être utilisés pour faire du SLAM (LIDAR, RADAR, GPS, caméra 2D, caméra 3D, etc.). Cependant, tel que présenté précédemment, ces capteurs peuvent coûter cher et ne peuvent pas être utilisés dans tout les types d'environnements, d'où l'intérêt pour le vSLAM (visual SLAM). L'utilisation d'images visuelles pour la localisation et la cartographie simultanées peut fournir un large éventail d'informations sur l'environnement [14]. Les techniques de vSLAM fonctionnent bien dans les environnements statiques en permettant d'extraire des caractéristiques visuelles stables à partir d'images. Cependant, dans les environnements avec des objets dynamiques (personnes, voitures, animaux, etc.), les performances diminuent de manière significative, car les caractéristiques visuelles peuvent provenir de ces objets, ce qui rend la localisation moins fiable [14]. Par exemple, dans le contexte d'un robot mobile d'intérieur travaillant dans un bureau dans lequel des humains et des chaises sont rencontrés, il est peu probable que ces objets ne restent à un emplacement exact entre les images. Par conséquent, les caractéristiques visuelles de ces éléments doivent être

1. http ://www.nvidia.com
2. http ://caffe.berkeleyvision.org
3. http ://deeplearning.net/software/theano/
4. https ://www.tensorflow.org/
5. http ://torch.ch/
6. https ://pytorch.org/

Tableau 2.2 Logiciels de création de réseaux de neurones artificiels

Logiciels	**Avantages**	**Inconvénients**
Caffe [2]	– Permet d'entraîner des réseaux sans avoir à écrire de code – Optimisé pour des réseaux convolutifs – Bon pour ajuster des réseaux déjà existants – Possède une interface Python	– N'est pas optimal pour des réseaux récursifs – Devient rapidement complexe avec de grands réseaux – Permet difficilement de faire des réseaux faits maison
Theano [3]	– Possède une interface python – Permet facilement de faire des réseaux faits maison – Est très optimisé puisqu'il est très bas niveau – Il est possible d'utilisé des logiciels complémentaires hauts-niveau ce qui diminue la difficulté d'utilisation	– Est très complexe à utiliser puisqu'il est très bas niveau – Theano a cessé d'être développé en septembre 2017
Tensor Flow [4]	– Possède une interface python – Compile rapidement de gros réseaux	– Est beaucoup moins optimisé que les autres logiciels – Permet difficilement de faire des réseaux faits maison – Devient rapidement complexe avec de grands réseaux
Torch [5] & Pytorch [6]	– Très modulaire – Beaucoup de réseaux pré-entraînés	– Est écrit dans le langage Lua – Plus complexe à utiliser

ignorées afin d'éviter toute dérive d'odométrie, ainsi que des artefacts 3D indésirables de la carte.

Le vSLAM comprend deux concepts, la localisation et la cartographie. Ceux-ci sont explorés dans les sous-sections suivantes.

2.4.1 Localisation

En robotique, la localisation, ou l'odométrie, consiste à utiliser des données relatives à l'environnement pour déterminer l'évolution de la position du robot. Pour ce faire, des dispositifs tels que des encodeurs de roue, des centrales inertielles ou bien des GPS sont utilisés. Bien que ces méthodes soient utiles pour de nombreux types de plateformes robotique, elles ne conviennent pas à tous les types de robots et à tous les types d'environnements. L'odométrie visuelle est donc une technique permettant de déterminer la position et l'orientation du robot en analysant des images successives obtenues à l'aide d'une caméra. Ce type d'odométrie possède plusieurs avantages. :

- Système passif, c'est-à-dire qu'il ne nécessite pas de matériel externe pour fonctionner, contrairement aux GPS. Ils peuvent donc fonctionner dans des environnements restreints ;
- Beaucoup d'information sur le monde extérieur. En effet, une image possède des milliers de pixels contenant de l'information sur la couleur et sur la distance des objets présents dans le champ de vision de la caméra. Les systèmes de centrale inertielle ainsi que les encodeurs de roue ne donnent qu'un type de donnée (accélération et nombre de tours). Avec un système de vision, il y a beaucoup d'informations qui peuvent être extraites pour améliorer le système.

L'odométrie visuelle possède également des désavantages :

- Information est difficile à analyser. En effet, l'information provenant des pixels est très difficile à analyser puisqu'il n'est pas possible d'établir une relation directe entre ceux-ci et le mouvement de la caméra ;
- Très sensible aux environnements dynamiques. Contrairement aux autres types de capteurs, les systèmes basés sur la vision ont beaucoup de difficulté à gérer les objets dynamiques.

Plusieurs techniques tentent de pallier aux problèmes de l'odométrie visuelle. Li et al. [30] proposent une méthode de pondération statique par points. L'approche calcule un poids pour chaque point de bordure d'une image clé. Ce poids indique la probabilité qu'un point spécifique fasse partie de l'environnement statique. Ces poids sont déterminés par le mouvement d'un point entre deux images et sont ajoutés à une méthode IAA (*Intensity*

Assisted Iterative Closest Point) utilisée pour effectuer la tâche d'enregistrement dans un SLAM. Une autre approche intéressante est l'approche de Sun et al. [45], présentant une technique de suppression de mouvement pour augmenter la fiabilité de localisation dans des environnements dynamiques. Leur technique est composée de trois étapes : 1) détecter les objets en mouvement à l'aide de la différence entre des images compensées par leur propre odometrie ; 2) suivre le mouvement avec un filtre à particules ; et 3) appliquer un estimateur Maximum-A-Posterior (MAP) sur les images de profondeur pour déterminer l'arrière-plan. Cette approche est utilisée en amont du SLAM DenseVisual Odometry (DVO) [23]. Les approches de Li et al. et de Sun et al. ont toutes deux montré de bons résultats de localisation dans le jeu de données de l'Université technique de Munich (TUM) [44]. Cependant, dans ces techniques la cartographie n'a pas encore été abordée.

2.4.2 Cartographie

Une carte, pour un système autonome, est définie comme un ensemble de données montrant l'arrangement spatial ou la distribution des éléments dans l'environnement. Pour les véhicules autonomes, la carte est le point central d'où le système de contrôle et de navigation recueille toutes les informations qui lui sont nécessaires pour planifier ses actions. En effet, la carte est la représentation que le véhicule a du monde extérieur. Plus celle-ci contient d'informations, plus le véhicule autonome peut prendre des décisions appropriées et précises. Une carte permet au robot de naviguer et d'interagir avec son environnement. Avec la venue des véhicules autonomes, plusieurs firmes de cartographie se lancent dans la génération de cartes pour ceux-ci, telles que HERE [7], Tomtom [8] et Google.

Une carte contient de l'information spécifique à la fonction de la plateforme robotique. Dans le cas de véhicules sur route autonomes, celle-ci peut contenir de l'information sur le trafic et sur la signalisation. Une plateforme autonome agricole aura une carte contenant plutôt de l'information sur la densité du sol, sur la qualité des plantes, etc. Une carte peut être une représentation spatiale 2D ou 3D de l'environnement, et cet aspect est souvent défini en fonction de la tâche à accomplir et des capteurs utilisés.

Il y a deux types de cartes utiles pour les plateformes robotiques autonomes, soit les cartes topologiques et métriques [47]. La figure 2.7 montre les deux types de carte dans le contexte d'une maison. Les cartes métriques sont plus intuitives pour l'humain. C'est une représentation 2D ou 3D des objets par rapport à des coordonnées cartésiennes dans l'environnement. Ce type de représentation est difficile à créer de façon précise et est

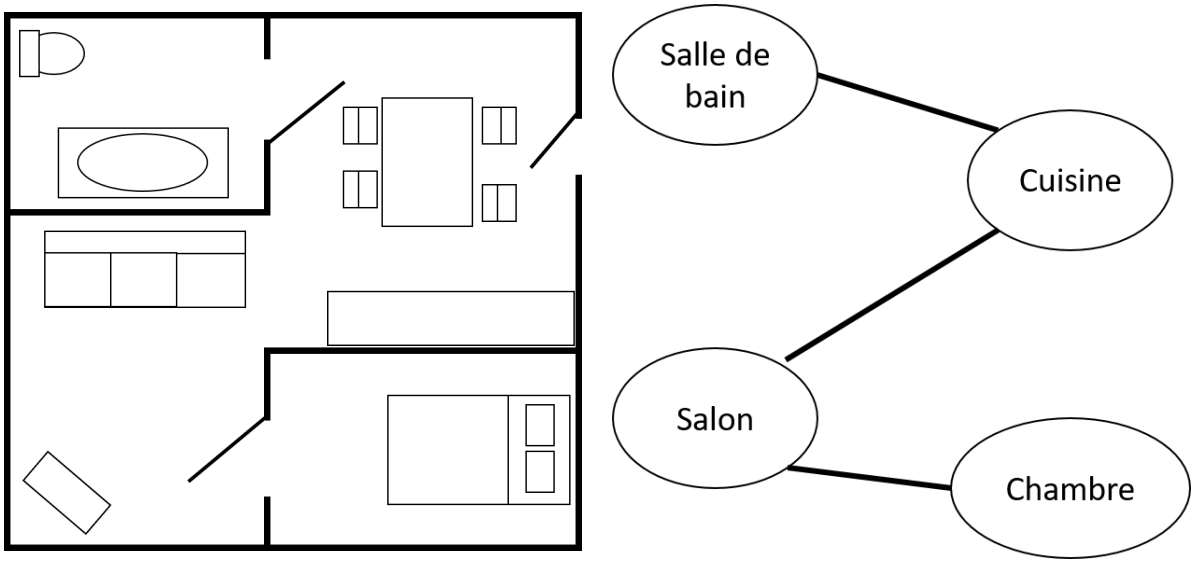
7. https ://here.com/en
8. https ://www.tomtom.com

très sensible au bruit. Les cartes topologiques quant à elles définissent les relations entre différents lieux ou objets. Ce type de carte est représenté comme un graphe dans lequel les nœuds sont les entités et les arcs correspondent aux relations entre ces différentes entités. Ce type de carte est beaucoup moins utilisé dans les plateformes mobiles autonomes puisque le but général est d'avoir une bonne localisation spatiale. Néanmoins, les cartes topologiques peuvent être très intéressantes pour accomplir des tâches plus complexes et pour faire la génération de trajectoire.

Un autre type de cartographie est la cartographie sémantique [15]. Ce type de cartographie est un mélange entre une carte topologique et une carte métrique et se base sur le concept de segmentation sémantique. La segmentation sémantique est utilisée pour faire la reconnaissance de scènes [43]. Pour chaque image, une étiquette parmi un ensemble de classes prédéfinies (voiture, piéton, ciel, route, etc.) est associée à chaque pixel. L'objectif est de prédire un masque de segmentation qui indique la catégorie de chaque pixel. Ceux-ci peuvent être classés à partir de différentes techniques de segmentation d'image. Les plus utilisés sont les techniques de segmentation par région, par contour et par classification. La segmentation sémantique est particulièrement effectuée sur des images 2D, mais elle peut aussi être utilisée sur des nuages de point ou de pixels volumétriques. Depuis les dernières années, l'intelligence artificielle a fait des avancées considérables dans le domaine de la vision artificielle, passant d'abord de la reconnaissance d'objets, à la localisation d'objets et maintenant la segmentation sémantique d'objets. La figure 2.8 illustre la classification, la détection et la segmentation appliquée à une image 2D. Dans le contexte d'une carte sémantique, une étiquette sémantique est attribuée à des objets qui sont affichés dans un espace cartésien. Avec ce type d'informations, il est donc possible pour la plateforme robotique de non seulement savoir avec quel objet interagir, mais aussi de savoir où il se trouve. Ce type de carte est plus complexe et nécessite une bonne gestion de l'information. Celle-ci peut être créée avec différents capteurs et avec différentes techniques de vision artificielle.

Les cartes sémantiques créées dans des environnements statiques ont grandement été étudiées [27, 28]. Par contre, les cartes créées dans des environnements dynamiques (éléments de l'environnement changeant dans le temps) sont encore un problème non résolu.

Plusieurs techniques basées sur les zones d'intérêt dynamiques ont été développés pour tenter de résoudre ce problème. Une des méthodes les plus simples est d'utiliser des empreintes RFID (*Radio-Frequency IDentification*) pour définir des objets semi-dynamiques [53]. Un robot avec un lecteur RFID lis les étiquettes RFID qui sont apposés sur des objets, ce qui lui permet d'acquérir le type de l'objet. Cette technique n'est malheureusement pas utilisable dans des environnements inconnus ou hostiles, mais s'avère cependant très utile



a: Carte métrique

b: Carte topologique

Figure 2.7 Exemple d'une carte métrique et d'une carte topologique d'une maison

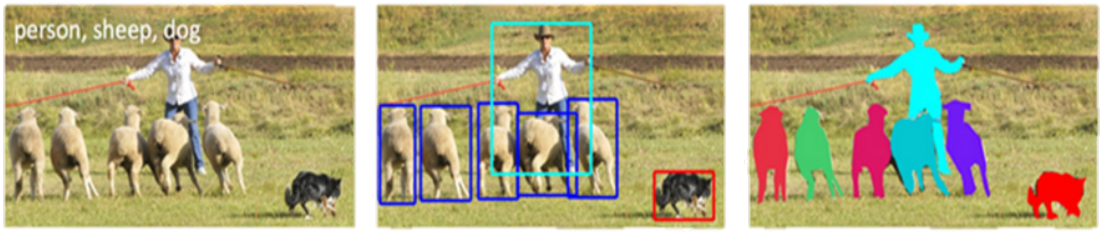


Figure 2.8 Classification, détection et segmentation appliqué à une image, https ://adeshpande3.github.io

dans des environnements connus et contrôlés tels des bâtiments ou des milieux industriels. Une autre technique prometteuse est l'approche collaborative SLAM en Environnements dynamiques (CoSLAM) [54], qui utilise plusieurs caméras mobiles indépendantes pour créer une carte globale robuste dans les environnements dynamiques, selon une méthode conventionnelle de structure par mouvement séquentiel (SFM). Le SLAM avec objets généralisés [21] utilise un système de classification des objets en fonction de leurs déplacements. Pour ce faire, des trajectoires pré-déterminées sont recherchées dans l'environnement afin de différencier les objets mobiles des objets statiques. Cette méthode apporte peu d'information puisqu'elle ne fait que classifier les objets dans trois catégories, soient des objets statiques, mobiles ou inconnus. Une variante, le SLAM++, est du SLAM orienté-objet [41] et est également à considérer. Celle-ci part de la prémisse que l'environnement possède une certaine symétrie (ou récurrence) dans les objets permettant de les reconnaître. Donc, au lieu de conserver les zones d'intérêts en nuage de points tels que dans le SLAM, le SLAM++ définit des objets à l'aide de zones d'intérêt pour, par la suite, les localiser et les cartographier. Cette technique a comme avantage d'être plus robuste que le SLAM traditionnel en ce qui concerne la localisation en environnement extérieur.

Une autre technique obtenant des résultats très prometteurs est développée par [20]. Ceux-ci génère un nuage de points 3D dense qui est segmenté en fonction de la sémantique de la scène. Tout d'abord, une segmentation sémantique 2D est effectué sur une image en utilisant un RDF (*Random Decision Forest*) ainsi qu'un CRF (*Conditional Random Field*) et ensuite, une reconstruction du nuage de points 3D est effectué avec de l'odométrie visuelle pour superposer la segmentation sémantique 2D au nuage de points 3D. Un RDF est une technique de classification de l'apprentissage-machine basée sur les arbres de décision. Dans cette application, le RDF reçoit des zones d'intérêts comme entrées et attribue aux pixels de cette zone d'intérêt une étiquette en fonction d'un arbre de décision. Par la suite, la sortie du RDF est envoyée dans un CRF pour obtenir une segmentation sémantique 2D beaucoup plus lisse. Le résultat est ensuite reporté sur le nuage de points 3D avec de l'odométrie visuelle. Cette technique donne de très bons résultats sur la base de donné *NYU Depth Dataset*[9].

En ce qui concerne la création d'une carte sémantique avec l'apprentissage profond, peu d'auteurs se sont penchés sur le sujet. Plus particulièrement, [34] ont généré des cartes 3D denses en sémantique, en temps réel (25 Hz) et avec détection de boucle en combinant des réseaux de neurones avec du SLAM. L'utilisation de l'apprentissage profond leur permet d'obtenir une précision supérieure à la technique développée par [20]. Leur approche est

9. http ://cs.nyu.edu/ silberman/datasets/

structurée en trois systèmes séparés, soit un système de SLAM temps-réel, un réseau de neurones convolutif et une inférence bayésienne. Le rôle du SLAM est de créer une carte 3D de l'environnement en temps réel. L'algorithme *ElasticFusion* est utilisé comme technique de SLAM [48]. Le réseau de neurones quant à lui fait de la segmentation sémantique sur une image 2D. Le réseau est implémenté avec le logiciel Caffe et est basé sur le réseau *Deconvolutional Semantic Segmentation Network Architecture* de [38]. Finalement, une inférence bayésienne suit la distribution des étiquettes de chaque élément de surface de la carte pour assurer une distribution homogène. Cette technique est la première à permettre la création d'une carte sémantique 3D en temps réel avec de l'apprentissage profond. Par contre, celle-ci est destinée à des environnements intérieurs et n'a pas été testée ou prévue pour des scènes extérieures. Suite à leurs travaux, ils proposent également qu'une interface tenant compte explicitement des objets au lieu de ne tenir compte que des éléments de surface serait avantageuse et permettrait d'ajouter de la fiabilité à ce type de technique. Zhou *et al.* [52] proposent un réseau de neurones de segmentation d'instance permettant de détecter des objets en mouvement. Ils ont utilisé l'architecture Mask R-CNN [19] avec des poids pré-entraînés basés sur la base de données COCO [31] et formés sur leur propre base de données pour segmenter les objets en mouvement. Leur ensemble de données utilisé a été créé à l'aide de techniques hors ligne standard pour détecter les mouvements dans les images. Ils ont établi un processus en cinq étapes : soustraction de trame, seuillage, vote, segmentation avec "super pixel" et filtrage morphologique. Cet ensemble de données a été créé à l'aide de données provenant du simulateur CARLA [12] et de données provenant du monde réel récolté avec leur propre équipement. Le Mask R-CNN (avec poids pré-entraîné) a été entraîné sur trois combinaisons des deux bases de données mentionnées précédemment : uniquement des données synthétiques, uniquement des données du monde réel et un mélange des deux. Leurs résultats sur le jeu de données Cityscape [10] montrent que le Mask R-CNN entrainé uniquement sur des images réelles donne de meilleurs résultats que les deux autres jeux de données (synthétique et mixtes). Ces résultats montrent également que leur méthode est capable de segmenter des classes non connues d'objets en mouvement, tels que les arbres, qui n'étaient pas présentes dans le Mask R-CNN original. Toutefois, cette technique semble donner de moins bons résultats sur la segmentation des classes pré-entraînées et il n'est pas clair si la segmentation des classes non connues est robuste.

CHAPITRE 3

ISM-vSLAM pour le masquage sémantique de scènes dynamiques de SLAM

3.1 Avant-propos

Auteurs et affiliation :

Jonathan Vincent : étudiant à la maitrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.

Mathieu Labbé : postdoctorant, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.

Jean-Samuel Lauzon : étudiant à la maîtrise, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.

Pier-Marc Comtois-Rivet : Ingénieur junior, Institut du Véhicule Innovant.

François Michaud : Professeur, Université de Sherbrooke, Faculté de génie, Département de génie électrique et de génie informatique.

Date de soumission : 10 juin 2019

Revue : *Robotic and Automation Letters*

Titre français : ISM-vSLAM : SLAM visuel masqué sémantiquement pour des scènes dynamiques

Contribution au document :

Suite à la revue, de la littérature présentée dans le chapitre 2, une méthodologie fut mise en place pour répondre à la question de recherche. L'article présenté dans ce chapitre en présente le résultat, soit la mise en place d'une approche de gestion des environnements dynamiques baptisé ISM-vSLAM (*Instance Semantic Mask for visual SLAM*) combinant des techniques d'apprentissage profond et d'apprentissage machine pour filtrer les objets dynamiques dans une architecture de vSLAM. Cette approche permet à une plateforme robotique de faire une gestion intelligente des objets dynamiques présents dans son environnement en filtrant les données visuelles présentées à un algorithme de SLAM, augmentant

ainsi la qualité de la localisation et de la cartographie. Cet article présente les travaux effectués ainsi qu'une analyse des résultats.

Résumé français :

Dans les environnements dynamiques, les performances des techniques de vSLAM peuvent être affectées par les caractéristiques visuelles associées aux objets en mouvement. Notre solution consiste à identifier ces objets afin que leurs caractéristiques visuelles puissent être filtrées en amont d'un algorithme de vSLAM. Cet article présente ISM-vSLAM, une approche d'apprentissage profond conçue pour masquer les objets dynamiques dans un pipeline vSLAM à l'aide de connaissances sémantiques antérieures, afin d'améliorer à la fois la localisation et la cartographie dans des environnements hautement dynamiques. Les résultats sur les séquences du jeu de données TUM suggèrent que ISM-vSLAM obtient de meilleures performances de localisation par rapport à d'autres méthodes de pointe, tout en fournissant également une carte 3D sans objet dynamique.

3.2 ISM-vSLAM : Instance Semantic Masked Visual SLAM for Dynamic Scenes

3.2.1 Abstract

In dynamic environments, performance of visual SLAM techniques can be impaired by visual features taken from moving objects. One solution is to identify those objects so that their visual features can be removed for localization and mapping. This paper presents the use of a deep neural network for semantic masking of dynamic objects in a visual SLAM pipeline, to improve both localization and mapping in dynamic environments. Results on the dynamic sequences from the TUM dataset using RTAB-Map as visual SLAM suggest that the approach achieves better localization performance compared to other state-of-the-art methods, while also providing a 3D map free of dynamic objects.

3.2.2 Introduction

To perform tasks effectively and safely, autonomous mobile robots need accurate and reliable localization from their representation of the environment. Compared to LIDARs (Light Detection And Ranging sensors) and GPS (Global Positioning System), using visual images for Simultaneous Localization and Mapping (SLAM) adds significant information about the environment [14], such as color, textures, surface composition that can be used for semantic interpretation of the environment. Standard visual SLAM (vSLAM) tech-

niques perform well in static environments by being able to extract stable visual features from images. However, in environments with dynamic objects (e.g., people, cars, animals), performance decreases significantly because visual features can come from those objects, making localization less reliable [14]. Deep learning architectures have recently demonstrated interesting capabilities to achieve semantic segmentation from images, outperforming traditional techniques in tasks such as image classification [9]. For instance, Segnet [2] is commonly used for semantic segmentation [16]. It uses an encoder and a decoder to achieve pixel wise semantic segmentation of a scene.

Therefore, to identify potential moving objects and to track and mask them during SLAM, this paper presents a novel approach that combines the Mask R-CNN [19] with Extended Kalman filtering (EKF) in a vSLAM pipeline. Our research hypothesis is that a deep learning algorithm can be used to semantically segment object instances in images using *a priori* semantic knowledge of dynamic objects, enabling the identification, tracking and removal of dynamic objects from the scenes to improve both localization and mapping in vSLAM. By doing so, the approach, referred to as Instance Semantic Masking for vSLAM (ISM-vSLAM) aims at providing increased visual odometry performance, increased number and quality of loop closure detection, and maps free of dynamic objects.

The paper is organized as follows. Section 3.2.3 presents related work of approaches taking into consideration dynamic objects during localization and during mapping. Section 3.2.4 describes our instance semantic masking approach applied as a pre-processing module to RTAB-Map [26], a vSLAM approach. Section 3.2.5 presents the experimental setup, and Section 3.2.6 provides comparative results on dynamic sequences taken from the TUM dataset.

3.2.3 Related Work

Some approaches take into consideration dynamic objects during localization. For instance, the Collaborative Visual SLAM in Dynamic Environments (CoSLAM) [54] approach uses multiple independent moving cameras to build a global map robust to dynamic objects, by following a conventional sequential structure-from-motion (SFM) method. The SFM uses multiple 2D images combined with odometry information to produce a 3D estimation of a scene. BaMVO [24] uses a RGB-D camera to estimate ego-motion. It uses a background model estimator combined with an energy-based dense visual odometry technique to estimate the motion of the camera. Li *et al.* [30] developed a static point weighting method which calculates a weight for each edge point in a keyframe. This weight indicates the likelihood of that specific edge point being part of the static environment. Weights are

determined by the movement of a depth edge point between two frames and are added to an Intensity Assisted Iterative Closest Point (IA-ICP) method used to perform the registration task in SLAM. Sun *et al.* [45] present a motion removal approach to increase the localization reliability in dynamic environments. It consists of three steps : 1) detecting moving objects' motion based on ego-motion compensated using image differencing ; 2) using a particle filter for tracking ; and 3) applying a Maximum-A-Posterior (MAP) estimator on depth images to determine the foreground. This approach is used as the frontend of Dense Visual Odometry (DVO) SLAM [23]. All of these approaches demonstrate good localization results using the Technical University of Munich (TUM) dataset [44]. However, mapping is yet to be addressed with those approaches.

SLAM++ [42] and Semantic Fusion [35] focus on the mapping aspect of SLAM in dynamic environments. SLAM++ [42] is an object-oriented SLAM which achieves efficient semantic scene description using 3D object recognition. SLAM++ defines objects using areas of interest to subsequently locate and map them. However, it needs predefined 3D object models to work. Semantic Fusion [35] creates a semantic segmented 3D map in real time using RGB-CNN [38], a convolutional deep learning neural network, and a dense SLAM algorithm. However, SLAM++ and Semantic Fusion do not adress SLAM localization accuracy in dynamic environments, neither do they remove dynamic objects in the 3D map.

3.2.4 Instance Semantic Masking for vSLAM

Figure 3.1 illustrates the ISM-vSLAM architecture. As a general overview of the approach, a set of objects of interest (OOI) are defined using *a priori* knowledge and understanding of dynamic objects classes that can be found in the environment. Instance segmentation is done using a neural network trained to identify the object classes from an RGB image.

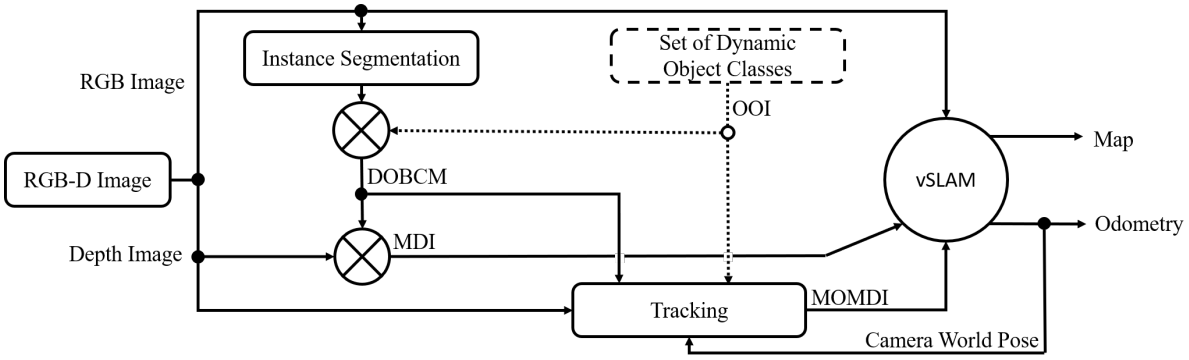


Figure 3.1 Architecture of ISM-vSLAM

A semantic pixel-wise binary mask is derived from the OOI bounding box and its dynamic object class. For each dynamic object instance, its bounding box, class type and binary mask are grouped for convenience and referred as DOBCM. The binary mask of the DOBCM is then applied to the original depth image, resulting in a masked depth image (MDI). The DOBCM is also sent to the Tracking module to determine if the object is idle or not. Computing a 3D centroid for each masked object, the Tracking module classifies the object as idle or not based on its class and its estimated velocity prediction. The Tracking module parameters are based on the OOI classes. The original depth image is then updated to only remove moving objects, resulting in the Moving Object Masked Depth Image (MOMDI). The original RGB image, the MDI and the MOMDI are used by the vSLAM algorithm for feature extraction and processing. The vSLAM algorithm uses the depth images as a mask for feature extraction thus ignoring features from the masked regions. The MOMDI is used by the visual odometry algorithm of the vSLAM approach while the MDI is used by its mapping algorithm, resulting in a map free of dynamic objects while still being able to use the features of the idle objects for visual odometry.

Instance Segmentation Using Mask R-CNN

One of the key elements of ISM-vSLAM is to find the appropriate approach for dynamic object segmentation. Deep learning algorithms such as Mask R-CNN recently proved to be useful to accomplish instance semantic segmentation [16]. In particular, Mask R-CNN achieves better results in the Microsoft Common Objects in Context (COCO) [31] segmentation challenge than the Fully Convolutional Instance-aware Semantic Segmentation (FCIS) and the Instance-aware Semantic Segmentation via Multi-task Network Cascades (MNC) [11] approaches, which respectively won the 2016 and 2015 COCO semantic segmentation challenge. Mask R-CNN is also more accurate for object segmentation than Segnet [2], mainly because it segments objects in region of interest rather than attempting to segment the whole scene.

Our instance segmentation module, mainly composed of the Mask-RCNN, takes the input RGB image and outputs the bounding box, class and binary mask for each instances. Two parameters are introduced :

– A dilation parameter d, which specifies the number of pixels used to enlarge the boundaries of the semantic mask. It is used to cope with the delay that can occur between the RGB image and its corresponding depth image. Such delay can generate errors on the semantic mask. For our trials, d was set empirically by evaluating the approximate offset in pixels between the semantic mask from the RGB frame and its corresponding depth image in a sequence with a fast moving object.

– A probability threshold p is used to easily change the acceptance level of a segmented object and minimize false positives or false negatives. For our trials, p was set empirically by observing the quality of the instance mask generated in the evaluated sequences.

Tracking Using EKF

Using the DOBCM from the Instance Segmentation module and odometry from vSLAM, the Tracking module predicts the pose and velocity of the objects in the world frame. This is useful when the camera is moving at speed similar to the objects to track (e.g., moving cars on the highway, robot following a pedestrian) or when idle objects have a high amount of features (e.g., person wearing a plaid shirt).

First, the Tracking module receives the DOBCM and the original depth image as a set, defined as $\mathbf{D}$:

$$\mathbf{D}^k = \{\mathbf{d}_1^k, ..., \mathbf{d}_I^k\} \tag{3.1}$$

$$\mathbf{d}_i^k = \left\{\mathbf{T}^k, \mathbf{B}_i^k, \zeta_i^k\right\} \tag{3.2}$$

where $\mathbf{d}_i^k$ is the object instance detected by the Instance Segmentation module, with $i \in I$, $I = \{1, ..., L\}$, L being the total number of object detection in the frame at time k. $\mathbf{T} \in \mathbb{R}^{m \times n}$ is the depth image , $\mathbf{B} \in \mathbb{Z}_2^{m \times n}$ is the binary mask and $\zeta \in J$ is the class ID, with $J = \{1, ..., W\}$, and W is the number of total trained classes in the Instance Segmentation module.

The DOBCM and the original depth image are used by EKF to determine if the detected object is currently moving or not. EKF provides steady tracking of each object instance corresponding to the object type detected by the Mask R-CNN neural network. An extended Kalman filter is instantiated for each new object, and *a priori* knowledge from the set of dynamic object classes defines some of the filter's parameters. This instantiation is made with the following parameters : the class of the object, its binary mask and its 3D centroid position. The 3D centroid is defined as the center of the corresponding bounding box. If the tracked object is observed in the DOBCM, its position is updated accordingly, otherwise its predicted position using EKF is used. If no observations of the object are made for e number of frames, the object is considered removed from the scene and therefore the filter is discarded. An object is defined as being idle or not based on the object class and the filter velocity prediction : if the velocity threshold v for a specific object class is exceeded, the object is considered to be moving. The original depth image is then

updated to only remove currently moving objects, resulting in the MOMDI. The MOMDI is sent to vSLAM odometry to update the camera pose.

To explain further how the Tracking module works, the following subsections presents in more details the Prediction and Update steps of EKF used in our ISM-vSLAM approach.

Prediction Let us define the hidden state $\mathbf{x} \in \mathbb{R}^{6\times 1}$ as the 3D pose and velocity of an object referenced in the global map in Cartesian coordinates.

$$\mathbf{x} = \begin{bmatrix} x_{xw} & x_{yw} & x_{zw} & x_{\dot{x}w} & x_{\dot{y}w} & x_{\dot{z}w} \end{bmatrix}^T \tag{3.3}$$

The *a priori* estimate of the state at time **k** is predicted based on the previous state at time $\mathbf{k-1}$ as in (3.4) :

$$\hat{\mathbf{x}}^{k|k-1} = \mathbf{F}\hat{\mathbf{x}}^{k-1|k-1} \tag{3.4}$$

$$\mathbf{F} = \begin{bmatrix} 1 & 0 & 0 & \Delta t & 0 & 0 \\ 0 & 1 & 0 & 0 & \Delta t & 0 \\ 0 & 0 & 1 & 0 & 0 & \Delta t \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \tag{3.5}$$

where $\mathbf{F} \in \mathbb{R}^{6\times 6}$ is the state transition matrix and Δt is the time between each prediction. This value is redefined before each processing cycle.

The *a priori* estimate of the state covariance at time k is predicted based on the previous state at time $k-1$ as given by (3.6) :

$$\mathbf{P}^{k|k-1} = \mathbf{F}\mathbf{P}^{k-1|k-1}\mathbf{F}^T + \mathbf{Q} \tag{3.6}$$

where $\mathbf{Q} \in \mathbb{R}^{6\times 6}$ is the process noise covariance matrix defined using the random acceleration model (3.7) :

$$\mathbf{Q} = \mathbf{\Gamma}\Sigma_a\mathbf{\Gamma}^T \tag{3.7}$$

$$\mathbf{\Gamma} = \begin{bmatrix} \frac{\Delta t^2}{2} & 0 & 0 & \Delta t^2 & 0 & 0 \\ 0 & \frac{\Delta t^2}{2} & 0 & 0 & \Delta t^2 & 0 \\ 0 & 0 & \frac{\Delta t^2}{2} & 0 & 0 & \Delta t^2 \end{bmatrix}^T \tag{3.8}$$

$$\Sigma_a = \begin{bmatrix} \sigma_{ax}^2 & 0 & 0 \\ 0 & \sigma_{ay}^2 & 0 \\ 0 & 0 & \sigma_{az}^2 \end{bmatrix} \tag{3.9}$$

where $\mathbf{\Gamma} \in \mathbb{R}^{6\times3}$ is the mapping between the random acceleration vector $\mathbf{a} \in \mathbb{R}^3$ and the state $\mathbf{x}$ defined by (3.8), and $\Sigma_a \in \mathbb{R}^{3\times3}$ is the covariance matrix of $\mathbf{a}$ given by (3.9). The acceleration components a_x, a_y and a_z are presupposed to be uncorrelated.

The dynamic of every detected objects may vary greatly depending on its class. For instance, a car does not have the same dynamic as a mug. To better track different types of objects, a set of σ_{ax}^2, σ_{ay}^2 and σ_{az}^2 is defined for each class to better represent their respective process noise. It should be noted that the process noise is referenced to the world in our situation.

Update In EKF, the Update step starts by evaluating the innovation $\tilde{\mathbf{y}}^k$ defined as (3.10) :

$$\tilde{\mathbf{y}}^k = \mathbf{z}^k - \hat{\mathbf{h}}^k(\hat{\mathbf{x}}^{k|k-1}) \tag{3.10}$$

where $\mathbf{z}^k \in \mathbb{R}^3$ is a 3D observation of a masked object in reference to the camera for each object instance :

$$\mathbf{z}^k = \begin{bmatrix} z_x & z_y & z_z \end{bmatrix}^T \tag{3.11}$$

Using a pinhole camera :

$$z_x = \frac{(centroid_x - C_x)z_z}{f_x} \tag{3.12}$$

$$z_y = \frac{(centroid_y - C_y)z_z}{f_y} \tag{3.13}$$

where C_x and C_y are the principal center point coordinate and f_x and f_y are the focal lengths expressed in pixels. z_z is approximated using the average depth from the masked region on the depth image. *centroid* is defined as the center of the bounding box.

To simplify the following equations, ($_1$, $_2$, $_3$) represents the Euler angles ϕ, θ, ψ (roll, pitch, yaw) and (S, C) represent respectively the sine and cosine operations. $h(\mathbf{x}^k) \in \mathbb{R}^4$ is the observation function which maps the true state space $\mathbf{x}^k$ to the observed state space $\mathbf{z}^k$. $\hat{\mathbf{h}}(\mathbf{x}^k)$ is the three first terms of $\mathbf{h}(\mathbf{x}^k)$. However, in our case, the transform between those spaces is not linear, justifying the use of EKF. The non-linear rotation matrix used to transform the estimate state $\hat{\mathbf{x}}^\mathbf{k}$ in the observed state $\mathbf{z}^k$ follows the (z, y, x) Tait-Bryan convention and is given by (3.14-3.17) :

$$h(\hat{\mathbf{x}}^k) = \begin{bmatrix} h_1 & h_2 & h_3 & 1 \end{bmatrix}^T \tag{3.14}$$

$$h_1 = (C_1C_2)\hat{x}_x + (C_1S_2S_3 - C_3S_1)\hat{x}_y + (S_1S_3 + C_1C_3S_2)\hat{x}_z + c_x \tag{3.15}$$

$$h_2 = (C_2S_1)\hat{x}_x + (C_1C_3 + S_1S_2S_3)\hat{x}_y + (C_3S_1S_2 - C_1S_3)\hat{x}_z + c_y \tag{3.16}$$

$$h_3 = -(S_2)\hat{x}_x + (C_2S_3)\hat{x}_y + (C_2C_3)\hat{x}_z + c_z \tag{3.17}$$

where c_x, c_y and c_z are the coordinate of the camera referenced to the world, which is derived using vSLAM odometry.

Given $h(\hat{\mathbf{x}}^k)$, the Jacobian $\mathbf{H}_k \in \mathbb{R}^{3\times6}$ is given by (3.18).

$$\mathbf{H}^k = \begin{bmatrix} C_1C_2 & C_1S_2S_3 - C_3S_1 & S_1S_3 + C_1C_3S_2 & 0 & 0 & 0 \\ C_2S_1 & C_1C_3 + S_1S_2S_3 & C_3S_1S_2 - C_1S_3 & 0 & 0 & 0 \\ -S_2 & C_2S_3 & C_2C_3 & 0 & 0 & 0 \end{bmatrix} \tag{3.18}$$

The innovation covariance $\mathbf{S}_k$ is defined as by (3.19) :

$$\mathbf{S}^k = \mathbf{H}^k\mathbf{P}^{k|k-1}\left(\mathbf{H}^k\right)^T + \mathbf{R}^k \tag{3.19}$$

$$\mathbf{R}^k = \begin{bmatrix} \mu_1 & 0 & 0 \\ 0 & \mu_2 & 0 \\ 0 & 0 & \mu_3 \end{bmatrix} \tag{3.20}$$

where $\mathbf{R}^k \in \mathbb{R}^{3\times3}$ is the covariance of the observation noise, and μ_1, μ_2 and μ_3 are the imprecision of the RGB-D camera.

The near optimal Kalman gain K^k is defined by (3.21).

$$\mathbf{K}^k = \mathbf{P}^{k|k-1} \left(\mathbf{H}^k\right)^T (\mathbf{S}^k)^{-1} \tag{3.21}$$

Finally, the updated state estimate $\hat{\mathbf{x}}^{k|k}$ and the covariance estimate are given respectively by (3.22) and (3.23).

$$\hat{\mathbf{x}}^{k|k} = \hat{\mathbf{x}}^{k|k-1} + \mathbf{K}^k \tilde{\mathbf{y}}^k \tag{3.22}$$

$$\mathbf{P}^{k|k} = (\mathbf{I} - \mathbf{K}^k \mathbf{H}^k) \mathbf{P}^{k|k-1} \tag{3.23}$$

3.2.5 Experimental Setup

To test our ISM-vSLAM approach, we chose to use the TUM dataset because it presents challenging indoor dynamic RGB-D sequences with ground truth to evaluate visual odometry techniques, and is commonly used to compare with other state-of-the-art techniques. We used sequences in static, low dynamic and highly dynamic environment.

For our experimental setup, ROS [1] is used as a middleware to make the interconnections between the input images, Mask R-CNN, EKF and RTAB-Map. Keras [2], with the deep learning library Tensorflow [3] as the backend, is used for the instance segmentation algorithm because of its simplicity and its large user base. The ResNet-101-FPN backbone is used because this configuration achieves the best results on the COCO dataset [19]. Our Instance segmentation module is based on the implementation of Mask R-CNN by Matterport [4] and its pre-trained weights. This implementation was slightly modified to interface it with the RTAB-Map library [26], which includes various state-of-the-art visual odometry algorithms, a loop closure detection approach and a 3D map render. The Mask R-CNN is trained on all 91 classes of the COCO dataset. It contains 91 object classes which gather the most common dynamic objects that an autonomous robot can find indoor or outdoor (e.g., person, bicycle, car, bus, cat, dog, bottle, cup, bowl, chair, laptop, remote, cellphone, etc.). It contains more than 80000 real images for training, 40000 for validation and 80000 for testing. The COCO dataset is often used to compare state-of-

1. http ://www.ros.org
2. https ://github.com/keras-team/keras
3. https ://www.tensorflow.org/
4. https ://github.com/matterport/Mask_RCNN

the-art instance segmentation approaches, which is why we chose to use it in our trials. In our tests, the OOI used were : person, chair, cup and bottle. Those objects were chosen because of their presence in the TUM dataset and in our in-house tests.

Table 3.1 presents the parameters used for ISM-vSLAM in our trials, based on empirical observations in the evaluated TUM sequences. It should be noted that those values where also driven by semantic understanding of the nature of the objects on our part. For instance, human movements are more difficult to predict for our tracking algorithm. The non-rigidity of humans and their erratic movement greatly helped set parameters empirically. A Kinect v1 is used in the TUM dataset, and the intrinsic parameters of this camera are as follow : $C_x = 336.113, C_y = 239.5, f_x = 525.0, f_y = 525.0$.

3.2.6 Results

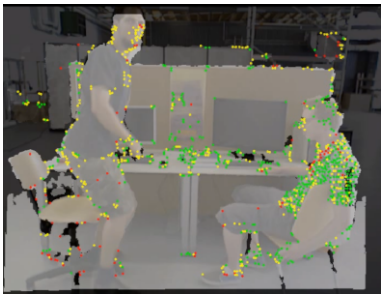
Trials were processed offline and comparisons are made with approaches by Kim and Kim [24], Sun *et al.* [45], Li *et al.* [30] and RTAB-Map, the latter being also used with ISM-vSLAM.

Figure 3.2a and Figure 3.2d show two original RGB frames in the TUM dataset, along with their superimposed RGB and depth images with features used by RTAB-Map (Fig. 3.2b and Fig. 3.2e) and with ISM-vSLAM (Fig. 3.2c and Fig. 3.2f). Using the depth image as a mask to filter outlying features, dynamic objects (i.e., humans in this case) are filtered out because the MDI includes the semantic mask. The MOMDI is used by RTAB-Map to compute visual odometry, keeping only the features from static objects as seen in Fig. 3.2c compared to Fig 3.2f with the colored dots representing visual feature used in for visual odometry. In Fig. 3.2c, the man on the left is classified by the Tracking module as moving, while the man on the right is classified as idle, resulting in keeping

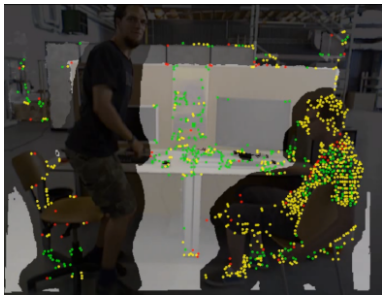
Tableau 3.1 Experimental Parameters

Parameter	Value	Description
e	10	Frame to terminate object tracking
d	30	Dilation threshold
p	0.3	Probability threshold
v_{person}	0.05	Velocity threshold for a person
v_{other}	0.5	Velocity threshold for the other objects
$\mathbf{a}_{x,y,z}^{person}$	0.62	Random acceleration for a person
$\mathbf{a}_{x,y,z}^{other}$	1	Random acceleration for other objects

a: Original RGB Image

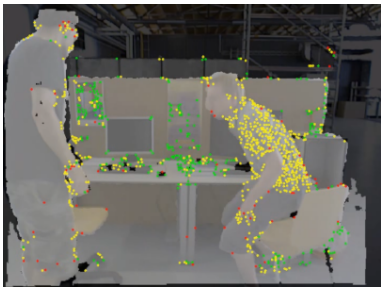


b: RGB and depth image superposed with RTAB-Map detectors

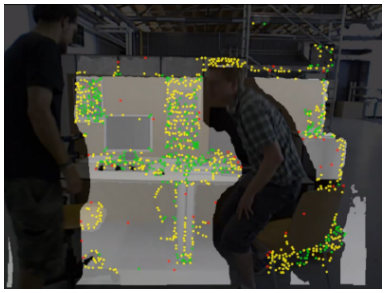


c: RGB and depth image superposed with ISM-vSLAM detectors

d: Original RGB Image



e: RGB and depth image superposed with RTAB-Map detectors



f: RGB and depth image superposed with ISM-vSLAM detectors

Figure 3.2 RTAB-Map detectors with and without ISM-vSLAM

his visual features. In Fig. 3.2f, the man on the right is also classified as moving because he is standing up, masking his visual features. Figure 3.3 illustrates the influence of MDI, which contains the depth mask of all the dynamic objects, either idle or not, to generate a map free of dynamic objects. This has two benefits : its create a more visually accurate 3D rendered map, and it improves loop closure detection. The differences in the 3D generated maps between RTAB-Map without and with ISM-vSLAM are very apparent : there are less artifacts of dynamic objects and less drifting. The *fr3/walking_ static* sequence shows greatly improved quality in the map, while the *fr3/walking_ rpy* sequence presents some undesirable artifacts. These artifacts are caused by the mask failing to identify dynamic objects that are tilted or upside down : this is because Mask R-CNN was trained mainly with images of objects in upward position, and could be improved by training it with images of objects rotated at different angles. The *fr3/sitting_ static* shows the result when masking idle object, resulting in completely removing the dynamic objects from the scene.

Table 3.2 characterizes the overall SLAM quality in terms of absolute trajectory error (ATE). In all cases, ISM-vSLAM improves the ATE and outperforms the other approaches in all dynamic sequences. This can be traced to its accurate odometry and its higher number and quality of loop closure detection. The decrease in performance in the *fr2/desk* sequence is caused by Mask R-CNN classifying for few frames the teddy bear present in the scene as a human.

Tableau 3.2 Absolute Transitional Error (ATE) RMSE (m) on the TUM dataset

TUM Dataset Sequences		BamVO [24]	Sun et al. [45]	Li et al. [30]	RTAB-Map	ISM-vSLAM	Improvements
Static Env.	fr2/desk	**0.0299**	-	-	0.0466	0.0473	-1.5%
Low Dyn. Env.	fr3/sitting_static	0.0248	-	-	0.0143	**0.0110**	23.1%
	fr3/sitting_xyz	0.0482	0.0317	0.0601	0.0310	**0.0175**	43.6%
	fr3/sitting_rpy	0.1872	-	-	0.0479	**0.0470**	1.9%
	fr3/sitting_half.	0.0589	0.0324	0.0432	0.0350	**0.0282**	19.4%
Highly Dyn. Env.	fr3/walking_static	0.1339	0.0656	0.0261	0.3121	**0.0122**	96.1%
	fr3/walking_xyz	0.2326	0.0932	0.0601	0.1971	**0.0400**	79.7%
	fr3/walking_rpy	0.3584	0.1333	0.1791	0.2271	**0.0711**	68.7%
	fr3/walking_half.	0.1738	0.1252	0.0489	0.3907	**0.0394**	89.9%

a: fr3/sitting_static without (top) and with (bottom) ISM-vSLAM

b: fr3/walking_static without (top) and with (bottom) ISM-vSLAM

c: fr3/walking_rpy without (top) and with (bottom) ISM-vSLAM

Figure 3.3 RTAB-Map 3D rendered map from the TUM sequences, without and with ISM-vSLAM

Table 3.3 presents the number of loop closure detection in the static and the dynamic sequences. In the static environment, ISM-vSLAM provides the same loop closure detection as RTAB-Map, suggesting that it does not hinder performance in static conditions. For dynamic sequences, ISM-vSLAM shows an improved number of detection in all sequences but *fr3/walking_static*, where the camera is static. RTAB-Map creates new nodes when the odometry exceeds a certain threshold in either position or rotation. With RTAB-Map alone, the features from the dynamic objects create an error which makes the odometry exceed the threshold and create a new node. This node is then optimized using loop closure. With ISM-vSLAM, which provides improved visual odometry, the error induced by the dynamic object is negligible, making the odometry never exceed the threshold and hence avoiding loop closure detection. In this particular sequence, less loop closure therefore reveals to be better.

We also tested ISM-vSLAM in a real environment as seen in Figure 3.4. In this sequence, the camera moved almost at the same speed as the chair held by a person while the other chairs remained static. The overall scene has very few features compared to the features of the chairs. Figure 3.5 shows the difference between RTAB-Map and ISM-vSLAM. The pink and blue lines represents the odometry of RTAB-Map without and with ISM-vSLAM. These results suggest qualitatively that ISM-vSLAM improves the odometry and the 3D map.

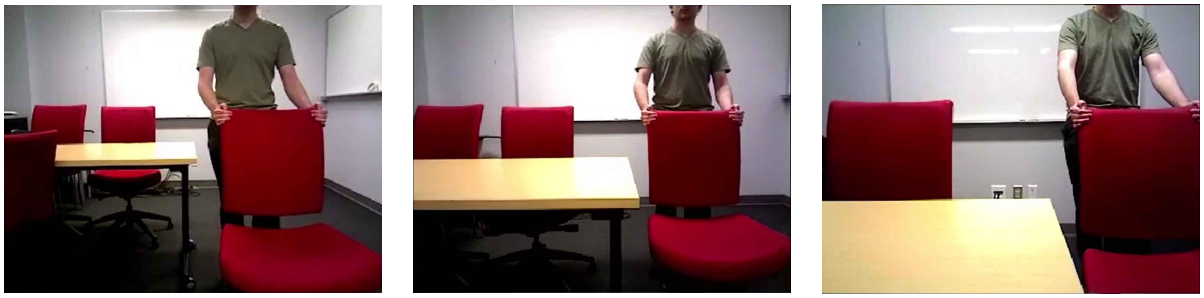
Those tests were run on a computer with an i7-6700K CPU, 32 GB of RAM and a GTX 1080 GPU. Our implementation runs with an average computation time of 330 msec per frame. It should be noted that our implementation is not optimized for real-time operation.

3.2.7 Conclusion

This paper presents ISM-vSLAM, an approach that uses a deep learning algorithm to semantically segment images, enabling the identification, tracking and masking of dynamic objects in scenes to improve both localization and mapping in vSLAM. Results on the TUM dataset suggest that using ISM-vSLAM with RTAB-Map provides better performance compared to other state-of-the-art localization approaches, while providing an improved 3D map and higher loop closure detection compared to RTAB-Map alone. Also, results suggest that ISM-vSLAM is robust to either static, low or highly dynamic environments. Because ISM-vSLAM pipeline is highly modular, it can also evolve with future improvements of deep learning architectures and new sets of dynamic object classes. In future work, we would like to add training for a wider range of dynamic objects, including

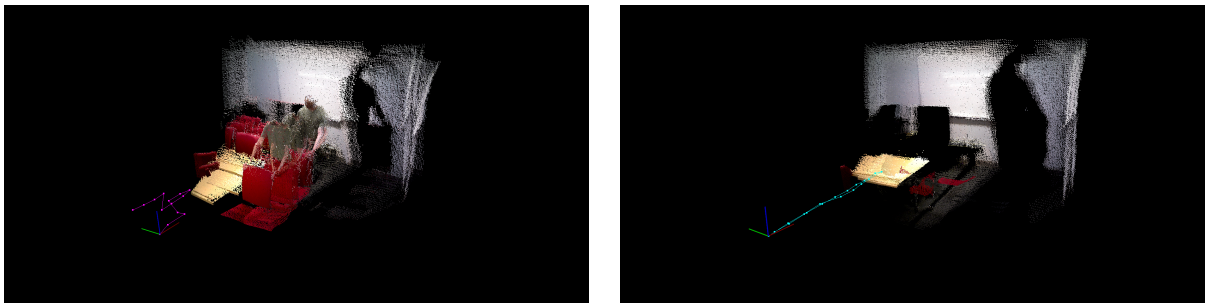
Tableau 3.3 Number of Loop Closure Detection

TUM Dataset Sequences		RTAB-Map	ISM-vSLAM
Static Env.	fr2/desk	**20**	**20**
Low Dyn. Env.	fr3/sitting_static	21	**28**
	fr3/sitting_xyz	39	**42**
	fr3/sitting_half.	30	**51**
	fr3/sitting_rpy	38	**47**
	fr2/desk_with_pers.	44	**49**
Highly Dyn. Env.	fr3/walking_static	**2**	0
	fr3/walking_xyz	19	**35**
	fr3/walking_halfs.	37	**38**
	fr3/walking_rpy	11	**49**



a: Frame at $t = 03$ sec. b: Frame at $t = 07$ sec. c: Frame at $t = 10$ sec.

Figure 3.4 Frames from the test scenario in a real environment



a: RTAB-Map b: ISM-vSLAM

Figure 3.5 RTAB-Map 3D map and odometry without (left) and with (rigth) ISM-vSLAM

rotated images, to increase robustness. We could use more complex neural networks [5] to add body keypoint tracking, which could significantly improve human feature extraction.

5. https ://github.com/daijucug/Mask-RCNN-TF_detection-human_segment-body_keypoint-regression

CHAPITRE 4

Conclusion

La robotique mobile est un domaine en pleine expansion. Avec les véhicules autonomes, beaucoup de nouvelles technologies sont développées pour répondre aux nombreux problèmes que ce type de systèmes soulèvent. Toutefois, bien que beaucoup d'innovations sont faites dans ce domaine, les environnements dynamiques sont encore un domaine peu exploré et difficile à gérer. Sachant que la plupart des applications de la robotique mobile sont dans des environnements dynamiques, il devient très pertinent de s'attaquer à ce problème. Plusieurs solutions basées sur des systèmes de SLAM ont été développées pour gérer ce type d'environnements. Toutefois aucune de ces techniques n'est capable de fournir une solution qui améliore la localisation ainsi que la cartographie en environnements dynamique. Dans le cadre de cette maîtrise, un système basé sur une approche d'apprentissage profond fonctionnant en amont d'un système vSLAM, pour filtrer de manière intelligente les objets dynamiques qui sont utilisés par le système SLAM, permet d'augmenter la qualité de l'odométrie ainsi que de la cartographie. L'approche ISM-vSLAM, utilisant un algorithme d'apprentissage profond pour segmenter les images de façon sémantique, permet ainsi l'identification, le suivi et le masquage d'objets dynamiques afin d'améliorer la localisation et la cartographie dans le système de SLAM RTAB-Map. Les résultats obtenus sur la base de données TUM suggèrent que l'utilisation d'ISM-vSLAM avec RTAB-Map offre de meilleures performances que d'autres approches de pointe, tout en fournissant une meilleure carte 3D et une meilleure détection de fermeture des boucles. L'un des points forts d'ISM-vSLAM est qu'il peut être utilisé pour masquer différents types d'objets, simplement en les intégrant dans la base de données d'entraînement. Les résultats suggèrent qu'ISM-vSLAM est robuste autant dans des environnements statiques que hautement dynamiques. L'architecture d'ISM-vSLAM étant hautement modulaire, il peut également évoluer avec des améliorations futures telles que des nouvelles architectures de réseaux de neurones et de nouvelles bases de données. ISM-vSLAM pourrait être très utile pour améliorer l'efficacité et la robustesse des véhicules autonomes dans des environnements dynamiques tels que des stationnements ou des autoroutes. En effet, ISM-vSLAM fait une gestion intelligente des objets dynamiques ce qui permet une localisation robuste dans des environnements ou la majorité des caractéristiques visuelles proviennent d'objets dynamiques. Toutefois, l'algorithme actuel fonctionne à une fréquence de rafraîchissement trop faible pour répondre aux besoins de réactivité des véhicules autonomes. De plus, l'al-

gorithme actuel utilise des connaissances a priori pour déterminer les objets qui sont de nature dynamique. Ceci limite l'utilisation de l'algorithme puisqu'il n'est pas en mesure de masquer des objets dynamiques inconnus.

Voici plusieurs voies possibles d'améliorations de l'algorithme :

1. Optimiser l'algorithme pour permettre un fonctionnement temps réel ;
2. Classifier les objets inconnus dynamiques à l'aide d'apprentissage machine ;
3. Entraîner le réseau sur un plus grand nombre d'objets dynamiques, y compris des images augmentées avec images pivotées, pour augmenter la robustesse d'ISM-vSLAM ;
4. Utiliser un réseau de neurones pour ajouter un suivi des membres du corps humain [1] pour améliorer considérablement l'extraction de caractéristiques des images.

Des essais sur véhicules routiers sont aussi envisagés afin d'accumuler des données qui permettront d'améliorer les connaissances a priori nécessaires pour l'algorithme. Finalement, l'algorithme développé dans le cadre de cette maîtrise pourrait, avec les modifications mentionnées ci-haut, améliorer considérablement la robustesse des véhicules autonomes dans des environnements dynamiques.

LISTE DES RÉFÉRENCES

[1] Alexandru, Iftimie Florentin *et al.*: *Mask-RCNN-TF*. `https://github.com/daijucug/Mask-RCNN-TF_detection-human_segment-body_keypoint-regression`, 2018.

[2] Badrinarayanan, Vijay, Alex Kendall et Roberto Cipolla: *Segnet : A deep convolutional encoder-decoder architecture for image segmentation*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 39(12) :2481–2495, 2017.

[3] Bellis, Elizabeth et Jim Page: *National motor vehicle crash causation survey (NMVCCS) SAS analytical users manual*. rapport technique, 2008.

[4] Ben Afia, Amani, Lina Deambrogio, Daniel Salós, Anne Christine Escher, Christophe Macabiau, Laurent Soulier et Vincent Gay-Bellile: *Review and classification of vision-based localisation techniques in unknown environments*. IET Radar, Sonar & Navigation, 8(9) :pp. 1059–1072, mai 2014. `https://hal-enac.archives-ouvertes.fr/hal-00996022`.

[5] Bibuli, Marco, Massimo Caccia et Lionel Lapierre: *Path-following algorithms and experiments for an autonomous surface vehicle*. IFAC Proceedings Volumes, 7(PART 1) :81–86, 2007, ISSN 14746670.

[6] Buehler, Martin, Karl Iagnemma et Sanjiv Singh: *The DARPA Urban Challenge : Autonomous Vehicles in City Traffic*. Springer Tracts in Advanced Robotics volume 56, 2009.

[7] Campbell, James B et Randolph H Wynne: *Introduction to Remote Sensing*. Guilford Press, 2011.

[8] Canny, John: *A computational approach to edge detection*. IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-8(6) :679–698, 1986.

[9] Ciregan, Dan, Ueli Meier et Jürgen Schmidhuber: *Multi-column deep neural networks for image classification*. Dans *Proceedings IEEE Conference on Computer Vision and Pattern Recognition*, pages 3642–3649, 2012.

[10] Cordts, Marius, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth et Bernt Schiele: *The cityscapes dataset for semantic urban scene understanding*. Dans *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3213–3223, 2016.

[11] Dai, Jifeng, Kaiming He et Jian Sun: *Instance-aware semantic segmentation via multi-task network cascades*. Dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3150–3158, 2016.

[12] Dosovitskiy, Alexey, German Ros, Felipe Codevilla, Antonio Lopez et Vladlen Koltun: *CARLA : An open urban driving simulator*. arXiv preprint arXiv :1711.03938, 2017.

[13] Fagnant, Daniel J. et Kara Kockelman: *Preparing a nation for autonomous vehicles : Opportunities, barriers and policy recommendations*. Transportation Research Part A : Policy and Practice, 77 :167–181, 2015, ISSN 09658564. `http://dx.doi.org/10.1016/j.tra.2015.04.003`.

[14] Fuentes-Pacheco, Jorge, José Ruiz-Ascencio et Juan Manuel Rendón-Mancha: *Visual simultaneous localization and mapping : A survey.* Artificial Intelligence Review, 43(1) :55–81, 2015.

[15] Galindo, Cipriano, Juan Antonio Fernández-Madrigal, Javier González et Alessandro Saffiotti: *Robot task planning using semantic maps.* Robotics and Autonomous Systems, 56(11) :955–966, 2008.

[16] Garcia-Garcia, Alberto, Sergio Orts-Escolano, Sergiu Oprea, Victor Villena-Martinez et Jose Garcia-Rodriguez: *A review on deep learning techniques applied to semantic segmentation.* arXiv preprint arXiv :1704.06857, 2017.

[17] Goodfellow, Ian, Yoshua Bengio et Aaron Courville: *Deep Learning.* MIT Press, 2016. `http://www.deeplearningbook.org`.

[18] Harris, Chris et Mike Stephens: *A combined corner and edge detector.* Procedings of the Alvey Vision Conference 1988, pages 147–151, 1988. `http://www.bmva.org/bmvc/1988/avc-88-023.html`.

[19] He, Kaiming, Georgia Gkioxari, Piotr Dollár et Ross Girshick: *Mask R-CNN.* Dans *Proceedings IEEE International Conference on Computer Vision*, pages 2980–2988, 2017.

[20] Hermans, Alexander, Georgios Floros et Bastian Leibe: *Dense 3D semantic mapping of indoor scenes from RGB-D images.* IEEE International Conference Robotics and Automation, pages 2631–2638, 2014.

[21] Hsiao, C. H. et C. C. Wang: *Achieving undelayed initialization in monocular SLAM with generalized objects using velocity estimate-based classification.* IEEE International Conference on Robotics and Automation, pages 4060–4066, 2011.

[22] Kelly, Alonzo, Omead Amidi, Mike Happold, Herman Herman, Tom Pilarski, Pete Rander, Anthony Stentz, Nick Vallidis et Randy Warner: *Toward Reliable Off-Road Autonomous Vehicle Operating in Challenging Environments.* International Symposium on Experimental Robotics, 25(5) :449–483, 2004, ISSN 0278-3649.

[23] Kerl, Christian, Jurgen Sturm et Daniel Cremers: *Dense visual SLAM for RGB-D cameras.* Dans *Proceedings IEEE/RSJ International Conference Intelligent Robots and Systems*, pages 2100–2106, 2013.

[24] Kim, Deok Hwa et Jong Hwan Kim: *Effective background model-based RGB-D dense visual odometry in a dynamic environment.* IEEE Transactions on Robotics, 32(6) :1565–1573, 2016.

[25] Kitano, Yukihiko, Shoichi Kobayashi, Mizuho Sakakibara, Hajime Kawano, Tatsuo Sakai, Hiroyuki Uematsu, Shintaro Kinoshita et Ryosuke Murai: *Autonomous mobile robot.* US Patent D663,334, juillet 2012.

[26] Labbé, Mathieu et François Michaud: *Online global loop closure detection for large-scale multi-session graph-based SLAM.* Dans *Proceedings IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2661–2666, 2014.

[27] Lakemeyer, Gerhard et Bernhard Nebel: *Exploring Artificial Intelligence in the New Millennium.* Morgan Kaufmann, 2003.

[28] Lang, Dagmar et Dietrich Paulus: *Semantic Maps for Robotics*. Proc. of the Workshop on AI Robotics, IEEE International Conference on Robotics and Automation, 2014.

[29] Le Cras, Jared, Jonathan Paxman et Brad Saracik: *Improving robustness of vision based localization under dynamic illumination*. Recent Advances in Robotics and Automation, pages 155–170, 2013.

[30] Li, Shile et Dongheui Lee: *RGB-D SLAM in dynamic environments using static point weighting*. IEEE Robotics and Automation Letters, 2(4) :2263–2270, 2017.

[31] Lin, Tsung Yi, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár et C Lawrence Zitnick: *Microsoft COCO : Common objects in context*. Dans *European Conference on Computer Vision*, pages 740–755. Springer, 2014.

[32] Lowe, David G: *Distinctive image features from scale-invariant keypoints*. International Journal of Computer Vision, 60(2) :91–110, 2004.

[33] Mattaboni, Paul J: *Autonomous mobile robot*. US Patent 4,638,445, janvier 1987.

[34] McCormac, John, Ankur Handa, Andrew Davison et Stefan Leutenegger: *SemanticFusion : Dense 3D semantic mapping with convolutional neural networks*. arXiv, 1609.05130, 2016.

[35] McCormac, John, Ankur Handa, Andrew Davison et Stefan Leutenegger: *SemanticFusion : Dense 3D semantic mapping with convolutional neural networks*. Dans *Proceedings IEEE International Conference on Robotics and Automation*, pages 4628–4635, 2017.

[36] Montemerlo, Michael, Jan Becker, Suhrid Bhat, Hendrik Dahlkamp, Dmitri Dolgov, Scott Ettinger, Dirk Haehnel, Tim Hilden, Gabe Hoffmann, Burkhard Huhnke, Doug Johnston, Stefan Klumpp, Dirk Langer, Anthony Levandowski, Jesse Levinson, Julien Marcil, David Orenstein, Johannes Paefgen, Isaac Penny, Anna Petrovskaya, Mike Pflueger, Ganymed Stanek, David Stavens, Antone Vogt et Sebastian Thrun: *Junior : The Stanford entry in the urban challenge*. Springer Tracts in Advanced Robotics, 56(October 2005) :91–123, 2009, ISSN 16107438.

[37] Muoio, Danielle: *19 Companies Racing To Put Self-Driving Cars on the Road By 2021*, 2016. `https://www.businessinsider.com/companies-making-driverless-cars-by-2020-2016-10`.

[38] Noh, Hyeonwoo, Seunghoon Hong et Bohyung Han: *Learning deconvolution network for semantic segmentation*. Dans *Proceedings of the IEEE International Conference on Computer Vision*, pages 1520–1528, 2015.

[39] Paton, Michael, François Pomerleau et Timothy D Barfoot: *In the dead of winter : Challenging vision-based path following in extreme conditions*. Field and Service Robotics, pages 563–576, 2016.

[40] Pereira, Valquiria Fenelon, Fabio Gagliardi Cozman, Paulo Eduardo Santos et Murilo Fernandes Martins: *A qualitative-probabilistic approach to autonomous mobile robot self localisation and self vision calibration*. Proceedings - Brazilian Conference on Intelligent Systems, pages 157–162, 2013.

[41] Salas-Moreno, Renato F, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly et Andrew J Davison: *SLAM++ : Simultaneous localisation and mapping at the level of objects*. Dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, 2013.

[42] Salas-Moreno, Renato F, Richard A Newcombe, Hauke Strasdat, Paul HJ Kelly et Andrew J Davison: *SLAM++ : Simultaneous localisation and mapping at the level of objects*. Dans *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1352–1359, 2013.

[43] Shapiro, Linda et George C Stockman: *Computer Vision*. Prentice Hall, 2001.

[44] Sturm, J., N. Engelhard, F. Endres, W. Burgard et D. Cremers: *A benchmark for the evaluation of RGB-D SLAM systems*. Dans *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, Oct. 2012.

[45] Sun, Yuxiang, Ming Liu et Max Q H Meng: *Improving RGB-D SLAM in dynamic environments : A motion removal approach*. Robotics and Autonomous Systems, 89 :110–122, 2017.

[46] Szegedy, Christian, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke et Andrew Rabinovich: *Going deeper with convolutions*. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pages 1–9, 2015.

[47] Thrun, Sebastian: *Learning metric-topological maps for indoor mobile robot navigation*. Artificial Intelligence, 99(1) :21–71, 1998.

[48] Whelan, Thomas, Stefan Leutenegger, Renato F Salas-Moreno, Ben Glocker et Andrew J Davison: *ElasticFusion : Dense SLAM Without A Pose Graph*. Robotics : Science and Systems, 11, 2015.

[49] Xu, Philippe, Franck Davoine, Jean Baptiste Bordes et Thierry Denoeux: *Fusion d'informations pour la compréhension de scènes*. Traitement du signal, 31(1-2) :57–80, 2014.

[50] Yamauchi, Brian: *Autonomous mobile robot*. US Patent 7,539,557, mai 2009.

[51] Zhang, Tao, Qing Li, Chang shui Zhang, Hua wei Liang, Ping Li, Tian miao Wang, Shuo Li, Yun long Zhu et Cheng Wu: *Current trends in the development of intelligent unmanned autonomous systems*. Frontiers of Information Technology & Electronic Engineering, 18(1) :68–85, 2017.

[52] Zhou, Guoxiang, Berta Bescos, Marcin Dymczyk, Mark Pfeiffer, José Neira et Roland Siegwart: *Dynamic objects segmentation for visual localization in urban environments*. arXiv preprint arXiv :1807.02996, 2018.

[53] Zhou, Hongjun et Shigeyuki Sakane: *Localizing objects during robot SLAM in semi-dynamic environments*. IEEE/ASME International Conference on Advanced Intelligent Mechatronics, pages 595–601, 2008.

[54] Zou, Danping et Ping Tan: *CoSLAM : Collaborative visual SLAM in dynamic environments*. IEEE Transactions on Pattern Analysis and Machine Intelligence, 35(2) :354–366, 2013.