Apprentissage profond et IA pour l'amélioration de la robustesse des techniques de localisation par vision artificielle

Achref Elouni

Université Clermont Auvergne
École Doctorale des Sciences Pour l'Ingénieur

Thèse présentée par :

Achref Elouni

Pour obtenir le grade de :

Docteur d'Université

Spécialité : Informatique

Apprentissage profond et IA pour l'amélioration de la robustesse des techniques de localisation par vision artificielle

Soutenue publiquement le 13 avril 2021 devant le jury :

Rapporteur	Catherine Achard	Professeur à Sorbonne Université
Rapporteur	Frédéric Lerasle	Professeur à Université Paul Sabatier
Présidente du jury	Najoua ben amara essoukri	Professeur à Université de Sousse
Directeur de thèse	Michel Dhome	Directeur de recherche CNRS
Examinateur	Eric Royer	MCF à Université Clermont Ferrand
Examinateur	Marc chevaldonné	MCF à Université Clermont Ferrand

Remerciements

Je remercie également les membres du jury de thèse – avec qui j'ai pris grand plaisir à partager – pour l'intérêt qu'ils ont porté à ces travaux, que j'ai pu apprécier de part la qualité de leurs rapports ainsi que lors de leurs remarques et questions pertinentes durant la soutenance.
Je remercie également mes encadrants en commençant par Michel Dhome, mon directeur de thèse et mes co-encadrants Eric Royer et Marc Chevaldonné. Il m'a été bien agréable de travailler avec eux pendant ces trois années.

Je remercie les dirigeants et tous les membres permanents, temporaires, doctorants et stagiaires de l'équipe ComSee. Enfin, je ne peux clore cette page sans remercier tous ceux qui m'ont aidé, quelle que soit l'ampleur de la tâche, dans l'accomplissement de ce travail ; qu'ils soient des membres de la famille, des amis ou d'autres...

Achref ELouni

Résumé

Le travail réalisé dans le cadre de ce doctorat se place dans le contexte d'un projet collaboratif ayant pour objectif la mise au point d'un casque de réalité augmenté. Afin de faire fonctionner un tel dispositif il s'avère nécessaire de calculer la position d'une caméra embarquée dans l'environnent d'intervention de l'utilisateur. Récemment, deux technologies dénommées SLAM (pour « Simultaneous Localization And Mapping ») et SfM (pour « Structure From Motion ») ont fait preuve de performances indéniables pour la reconstruction 3D d'un environnement à partir d'une collection d'images. Nous nous sommes intéressés à elles afin de résoudre le problème délicat de l'initialisation de notre dispositif ou de sa ré-initialisation en cas d'échec du suivi temps réel de la position. En effet, malgré les travaux de recherche réalisés ces dernières années, plusieurs limitations empêchent les système de localisation d'estimer une pose parfaite dans toutes les conditions. Ces conditions incluent les changements légers du contexte comme les variations de la luminosité, du point d'observation ou des modifications géométriques telles que l'ajout d'objets.

Pour faire face à ces limitations et afin de proposer une solution facile à déployer, nous avons étudié la possibilité d'intégrer dans le processus de localisation des informations invariantes qui pourraient augmenter la probabilité d'avoir une pose précise. Deux types d'information invariante (sémantique et géométrique) ont été exploitées dans cette thèse pour aider le système de localisation à trouver sa position.

Les solutions proposées ont été validées sur plusieurs jeux de données internes et externes (Dubrovnik, Rome, Oxford, Musée) grâce auxquels nous avons pu comparer nos résultats avec les travaux décrits dans l'état de l'art. Deux types d'images requêtes ont été étudiées dans cette thèse : celle composée d'une seule image et celle issue d'un dispositif stéréo. L'avantage d'utiliser une paire stéréo est de pouvoir trianguler des points homologues afin d'extraire leur hauteur et d'exploiter cette dernière dans le processus de localisation. L'autre approche envisagée consiste à utiliser comme invariant le label des pixels obtenu par un algorithme de segmentation sémantique basé sur un réseau de neurones convolutionnel. Dans les deux cas, les résultats obtenus montrent une amélioration sensible sur la précision des poses estimées

Mots clés : Reconstruction 3D, Estimation de la pose, Réalité Augmenté, Apprentissage Profond.

Table des matières

Table des figures **viii**

1 Introduction générale **1**
1.1 Contexte . . . 2
1.1.1 objectif . . . 2
1.1.2 Projet . . . 2
1.1.3 Application . . . 3
1.1.4 Contributions . . . 5
1.1.5 Organisation du mémoire . . . 6

2 État de l'art **7**
2.1 Le problème de la reconstruction 3D . . . 7
2.1.1 Descripteurs d'image . . . 7
2.1.2 La reconstruction 3D . . . 13
2.1.3 Géométrie épipolaire et triangulation . . . 14
2.1.4 Calcul de la pose . . . 16
2.1.5 Méthodes de reconstruction par vision . . . 17
2.2 Localisation visuelle basée sur l'image . . . 21
2.2.1 Définition . . . 21
2.2.2 Méthodes indirectes . . . 21
2.2.3 Méthodes directes . . . 25
2.2.4 Le problème des changements visuels dans l'environnement . . . 27
2.2.5 Correspondance entre les caractéristiques . . . 28
2.2.6 La segmentation sémantique . . . 31
2.2.7 Bases de données pour la localisation . . . 38
2.3 Conclusion . . . 40

3 Descripteur géométrique-visuel pour une localisation améliorée basée sur l'image utilisant un a priori sur la verticale **42**
3.1 Introduction . . . 42
3.2 Méthode proposée . . . 44

3.2.1 Descripteurs visuels géométriques . . . 45
3.2.2 Sac de mot géométrique (G-BoVW) . . . 46
3.2.3 Limitation . . . 47
3.3 Étude expérimentale . . . 48
3.3.1 Implémentation . . . 48
3.3.2 Jeux de données . . . 49
3.3.3 Résultats . . . 57
3.4 Conclusion . . . 63

4 La segmentation sémantique pour améliorer la localisation visuelle 65
4.1 Introduction . . . 65
4.2 Méthode proposée . . . 68
4.2.1 Correspondance sémantique . . . 70
4.2.2 Labellisation multiple basée sur la segmentation sémantique . 72
4.2.3 Labellisation multiple basée sur une combinaison de la segmentation sémantique et détection de contour sémantique. . . 74
4.2.4 Labelisation multiple basée sur la combinaison de couches SoftMax de la segmentation sémantique et de détecteur du contour sémantique . . . 77
4.2.5 Descripteur probabiliste . . . 78
4.2.6 Labellisation des points 3D . . . 78
4.2.7 Correspondance des caractéristiques et des points 3D . . . 79
4.3 Étude expérimentale . . . 80
4.3.1 Implémentation . . . 80
4.3.2 Résultats . . . 81
4.4 Conclusion . . . 88

5 La segmentation sémantique pour améliorer la recherche d'image par le contenu 89
5.1 Introduction . . . 89
5.2 Méthode proposée . . . 90
5.2.1 Création d'une signature d'image en se basant sur les informations sémantiques . . . 91
5.2.2 Création d'une signature d'image en se basant sur la combinaison des informations sémantique et visuelle . . . 95
5.3 Étude expérimentale . . . 96
5.3.1 Base des données pour la segmentation sémantique . . . 96
5.3.2 Résultats . . . 98
5.4 Conclusion . . . 105

6 Conclusion et Perspectives 107

Bibliographie 110

Table des figures

1.1 Un scénario « Atelier des Charrons ». 4
1.2 Découvrir un lieu virtuellement. 5

2.1 Des points d'intérêt avec 3 détecteurs différents. 8
2.2 Détection des points d'intérêt avec deux détecteurs différents SuperPoint[33] et Harris[32]. 10
2.3 Résultats d'appariements avec des correspondances correctes en vert pour LF-Net [78] contre SIFT, SURF et A-KAZE. 11
2.4 Correspondance entre deux images en utilisant le descripteur visuel SIFT. 12
2.5 Correspondance entre des images en utilisant un descripteur par apprentissage profond (SuperPoint [33]). 13
2.6 Nuage de points 3D (les points 3D sont représentés en noir et les caméras sont en rouge [10]. 14
2.7 Deux caméras observent la même scène de deux positions différentes [6]. 15
2.8 Triangulation dans le cas idéal de la géométrie épipolaire (pas de bruit sur la localisation des points homologues x1 et x2). 16
2.9 Architecture générale de l'algorithme SLAM 18
2.10 Architecture générale de l'algorithme SfM. 19
2.11 (A) méthode indirecte (B) méthode directe. 21
2.12 Modèle de Sac de Mots visuels [29]. 23
2.13 Exemple d'architecture CNN pour prédire les chiffres [1]. 24
2.14 Illustration de la méthode directe. Représentations des correspondances 2D-3D entre l'image requête et la carte 3D [91]. 26
2.15 Architecture de PointNet[84] 28
2.16 Des images du même endroit avec différentes conditions d'éclairage (jour, coucher de soleil, nuit) [111]. 29
2.17 Des images du même endroit au cours des quatre saisons (hiver, automne, été, printemps) [8]. 30

2.18 Classification des pixels dans une image [7]. 31
2.19 Correspondances 2D-2D entre des images prises dans des conditions différentes [9]. 32
2.20 Les différentes couches de l'architecture FCN et leurs tailles [55]. . . . 33
2.21 Les différentes couches de l'architecture U-Net [88]. 34
2.22 Les différentes couches de l'architecture SegNet [18]. 35
2.23 Les différentes couches de l'architecture PSPNet [125]. 35
2.24 Exemple d'une prédiction en utilisant l'apprentissage profond pour détecter les différents éléments d'un visage [51]. 36
2.25 Illustration des systèmes de perception d'un véhicule autonome à l'aide de l'apprentissage profond [2]. 37
2.26 Exemple d'utilisation de l'apprentissage profond pour l'agriculture de précision [11]. 38
2.27 Le modèle 3D pour la base de données Dubrovnik [60]. 38
2.28 Exemples d'images de la base de données Dubrovnik [60]. 39
2.29 Le modèle 3D pour la base de données Rome [60]. 39
2.30 Exemples d'images de la base de données Rome [60]. 40

3.1 Exemples des systèmes exploitant une tête stéréo. 43
3.2 Les différentes étapes de la méthode proposée. 44
3.3 Système de coordonnées pour un robot ou un véhicule. 45
3.4 Processus de construction du dictionnaire visuel [5]. 46
3.5 Des images de la base de données du musée montrent que Z est bornée entre le sol et le plafond. 47
3.6 Des images de la base de donnée du oxford montrent que la hauteur Z est non bornée. 48
3.7 Exemple de kd-Tree construit en utilisant des plans médians de l'espace [4]. 49
3.8 Plan et la carte 3D du musée. 50
3.9 Carte 3D d'Oxford Robotcar [65]. 50
3.10 Caméra stéréo 3D (Tara [3]). 51
3.11 Schéma fonctionnel du module de caméra TARA [3]. 52
3.12 Les deux cartes 3D de la base musée utilisées pour évaluer notre approche : (a) la carte à partir de laquelle les images requêtes sont extraites (b) la carte référence. 53
3.13 Exemples d'images de la base musée. 54
3.14 Capteurs utilisés pour collecter l'ensemble des données Oxford RobotCar [65]. 55
3.15 Les deux cartes 3D de la base Oxford Robotcar utilisées pour évaluer notre approche : (a) la carte référence (b) la carte à partir de laquelle les images requêtes sont extraites. 56
3.16 Exemples d'images de la base de données Oxford Robotcar [65]. . . . 56
3.17 Différentes étapes de l'algorithme Fast Search[90] 57

3.18 Différentes étapes de l'algorithme Active Search[90] 57
3.19 Pour chaque image de requête de la base musée, le nombre d'inliers lors du calcul de la pose avec P3P-RANSAC. 59
3.20 Quelques résultats de notre méthode sur la base de données "musée". 60
3.21 Quelques résultats de notre méthode sur la base de données "Oxford". 61
3.22 Quelques résultats de notre méthode sur les deux bases de données (musée, Oxford) 62
3.23 La carte 3D (nuage de points en bleu) et les poses requêtes bien localisées (les points en rouge) pour la base de données "Oxford". . . . 63
3.24 La carte 3D (nuage de points en bleu) et les poses requêtes bien localisées (les points en rouge) pour la base de données "musée". . . . 63

4.1 Exemple de la segmentation sémantique d'une image dans une zone urbaine [39]. 66
4.2 Correspondance 2D-2D en exploitant les informations sémantiques. . 67
4.3 Framework proposé. 68
4.4 Graphe de visibilité [128] : Les points oranges P3, P4 sont visibles par les caméras (C1, C2, C3). Le point vert P5 est visible par les caméras (C2,C3) et les points bleus ne sont pas visibles. 69
4.5 Graphe de poses de la carte SFM. Si la sortie de la correspondance sémantique et du filtrage est (4,3,1) alors nous extrayons tous les points 3D associés aux images clés (1,2,3,4,5). 70
4.6 Correspondance entre deux images sans prendre en considération les informations sémantiques [60]. 71
4.7 Correspondance sémantique entre les points clés 72
4.8 (a) Chevauchement entre deux classes (b) La courbe bleue présente les probabilités de la classe ciel et la courbe rouge présente les probabilités de la classe immeuble (c) L'image contient deux classes (Ciel, Immeuble) et montre les points échantillonnés selon la courbe représentée en (b). (d) L'image segmentée montre le chevauchement entre deux classes à la frontière. En outre, la figure montre les points-clés détectés : (1) points-clés avec une fausse étiquette de classe (symbole rouge), (2) points-clés avec une vraie étiquette de classe (symbole vert). 73
4.9 (a) Images clés utilisées pour construire le modèle de SFM (Dubrovnick) (b) Images clés segmentées à l'aide de l'algorithme Encnet [123] entraîné sur ADE20K [127], (c) détections des pixels avec de faibles écarts de probabilités entre les classes sémantiques (en vert clair). . . 74
4.10 Exemples de carte de bord sémantique globale détectée à l'aide de l'algorithme DFF [44] entraîné sur Ade20k et leurs pixels correspondants. 75

4.11 Détection du bord sémantique pour chaque classe sémantique prédite à l'aide de l'algorithme DFF [44] entraîné sur Ade20k. Dans la figure, nous prenons deux pixels (A, B) comme exemple. En fonction des probabilités extraites, le premier pixel (A) a trois étiquettes (Ciel, Bâtiment, Végétation) et le second (B) a deux étiquettes (Bâtiment, Personne). 76
4.12 labelisation des pixels par la segmentation sémantique et le détecteur sémantique de contour. 77
4.13 Étiquetage des pixels en combinant les probabilités données par la couche SoftMax à partir de la segmentation sémantique et du détecteur de bord sémantique. 77
4.14 Exemple des images annotées sur Ade20K [125]. 80
4.15 L'architecture EncNet [124]. 81
4.16 Nuage des points 3D récupérées à partir des correspondances 2D-3D avec les images requêtes et la carte 3D globale de la base Dubrovnik. 83
4.17 Nuage des points de la base de données Dubrovnik (points en verts) et la projection des points 3D d'images requêtes (points en violets). . 84
4.18 (a) image requête. (b) nuage des points 3D donnée par notre approche multi-labels. (c) nuage de points 3D de la carte globale (Dubrovnik). (d) projection de points 3D de l'image requête sur la carte globale. . 85
4.19 Comparaison des quartiles avec les méthodes de l'état de l'art, notre méthode (méthode multi-labels) atteint la meilleure précision de localisation sur l'ensemble de données de Dubrovnik. 87

5.1 Les étapes clés de la méthode indirecte. 90
5.2 L'approche globale. 90
5.3 Architecture de segmentation sémantique avec une couche supplémentaire. 92
5.4 Les différentes étapes pour construire une signature sémantique . . . 93
5.5 Un exemple de conversion d'un bloc sémantique vers un vecteur binaire sémantique. 93
5.6 Exemple d'encodage de la proportion. Avec une image divisée en 4 blocs, nous sélectionnons itérativement chaque bloc pour calculer la proportion de la classe sémantique à l'intérieur. 94
5.7 Les différentes étapes de la création de la signature d'image 95
5.8 Exemples d'images de la base de donnée Coco-Stuff [20] et les images segmentées. 97
5.9 Exemples d'images de la base de donnée Mseg [56] et les images segmentées. 98
5.10 Exemples de la liste d'images obtenue après l'utilisation des deux signatures. 99
5.11 Nuage des points 3D extrait à partir des 100 images les plus proches de l'image requête en utilisant la signature sémantique. 101

5.12 (a) image requête. (b) nuage des points 3D après le filtrage par la signature sémantique visuelle. (c) nuage de points 3D de la carte globale (Dubrovnik). (d) projection de points 3D de l'image requête sur la carte globale. 103

Liste des tableaux

3.1 Caractéristiques de la caméra Tara. 51
3.2 Comparaison du nombre d'images requêtes qui sont bien localisées par notre approche et par les méthodes ré-implémentées dans le cas de la base intérieure (musée). 58
3.3 Comparaison du nombre d'images requêtes localisées par notre approche et par méthodes ré-implémentées dans le cas de la base extérieure (Oxford). 58

4.1 Résultats détaillés de la méthode multi-labels pour l'ensemble de données Dubrovnik avec différentes tailles de fenêtre. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation. 82
4.2 Détails sur les labels des points 3D et leurs descripteurs 82
4.3 Résultats de comparaison entre mono label (méthode 1) et les méthodes multi-labels proposées. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation. 82
4.4 Comparaison de notre approche avec des méthodes de l'état de l'art pour Dubrovnik. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation. 86
4.5 Comparaison entre les résultats de nos méthodes pour Rome. 86
4.6 Comparaison de notre approche avec des méthodes de l'état de l'art pour Rome. 87

5.1 Comparaison de notre approche avec des méthodes de l'état de l'art pour Dubrovnik. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation. 104
5.2 Comparaison de notre approche avec des méthodes de l'état de l'art pour Rome (Mseg Dataset). 104

Chapitre 1

Introduction générale

La localisation est l'une des compétences les plus fondamentales requises par un robot autonome, car la connaissance de sa localisation est un prérequis essentiel pour prendre des décisions sur les actions futures. Celle-ci est utilisée pour accomplir des tâches très variées telles que le transport, la surveillance, etc. La localisation est le processus de détermination de la pose (rotation, translation) d'un robot dans un environnement connu. Ces environnements peuvent être reconstruits en 3D à partir de vidéos ou d'un ensemble d'images. Il existe deux technologies principales capables de reproduire la géométrie de points 3D remarquables d'une scène et de connaître les coordonnées 3D à partir d'un ensemble d'images prises sous différents points de vue d'une scène : SLAM (Simultaneous Localization And Mapping ou localisation et cartographie simultanées) et SfM (Structure From Motion ou structure à partir du mouvement). Ces deux techniques SfM et SLAM visuel ont suscité un intérêt considérable de la part des communautés de la vision artificielle par ordinateur et de la robotique.

Au cours de la construction d'une carte 3D, le système fait appel à la tâche de localisation dans deux situations différentes. Le premier cas correspond à la localisation d'une nouvelle image alors que la pose de l'image précédente est connue. Ce n'est pas cette situation qui est traitée dans cette thèse. La deuxième situation survient à l'initialisation ou lorsque le système de localisation n'a pas réussi à estimer sa position dans l'environnement, d'où la nécessité de le réinitialiser et trouver la position de l'image dans la carte 3D.

C'est ce cas auquel nous nous intéressons, dans ce travail de thèse, c'est-à-dire le problème de la détermination de la pose (rotation + translation) d'une image requête lorsqu'on ne dispose d'aucun a priori. Un algorithme de localisation idéal devrait être capable de gérer les changements visuels légers induits par difficiles causes : cycle jour/nuit et changement de saison (hiver/printemps/automne/été), différence de points de vue ou modifications de la géométrie ou contenu de la scène. Pour une localisation robuste, deux types d'informations invariantes ont été exploités dans cette thèse pour aider le système de localisation à trouver sa position :

(1) l'information sémantique extraite en utilisant l'apprentissage profond basé sur les réseaux de neurones convolutionnels, (2) l'information géométrique issue de la hauteur des éléments observés dans la scène. Nous avons mené nos expérimentations sur deux types de bases de données. Les bases de données publiques (Dubrovnik, Rome, Oxford) grâce auxquelles nous avons pu comparer nos résultats avec les travaux décrits dans l'état de l'art. Une autre base de données privée a été créée par notre équipe sur le site du musée de Solutré avec une tête stéréo. Dans le cadre d'un projet collaboratif, l'avantage d'utiliser une paire stéréo au lieu d'une seule caméra est de pouvoir trianguler des points 3D d'une paire d'images requête afin d'extraire leur hauteur pour utiliser cette dernière dans le processus de localisation.

1.1 Contexte

1.1.1 objectif

Le verrou scientifique adressé par ce travail est de développer des méthodes pour estimer la pose d'une caméra (sans a priori) pour aider un casque de réalité augmentée à se localiser dans un environnement connu. Pour ce faire, nous avons utilisé des approches classiques pour calculer la pose au moyen de mises en correspondances entre l'image requête et l'ensemble de la scène 3D reliant des points 3D et leurs projections 2D. Le défi ici est que le processus d'appariement devient critique dans les situations difficiles. Malgré la puissance des méthodes existantes, l'estimation de la pose reste toujours un challenge. En prenant en compte les différents défis, nous avons oeuvré dans cette thèse au développement et à l'amélioration de nouvelles techniques d'estimation de la pose.

1.1.2 Projet

Ce travail de recherche a été mené pendant la période 2017 - 2020 à l'Institut Pascal de Clermont Ferrand au sein de l'axe Image, Systèmes de Perception et Robotique. Il s'inscrit dans le cadre d'un projet REVE5D. Le projet REVE5D a été sélectionné en avril 2016 par le Fonds Unique Interministériel (Call 21).

Il avait préalablement été labellisé par les Pôles de compétitivité :

— Minalogic

— Cimes

Il est cofinancé par :

— L'Union européenne

— BPI France

— La région Auvergne Rhône-Alpes

— St-Etienne Métropole

Ce projet REVE5D a pour objectifs de :

— Construire un système de visualisation d'images de la scène réelle et des images de synthèse.

— Développer un logiciel permettant d'incruster des objets virtuels dans une séquence d'images.

— Valider l'ergonomie d'utilisation d'un casque de réalité virtuelle ou augmentée en situation opérationnelle pour les différents secteurs applicatifs ciblés (Culture, Infrastructures, Formation).

Ce projet est une collaboration entre :

— 2 grands groupes : Thales Angénieux et Nexter Training

— 3 PME : Vapérail, SFI et Studio R Bouquet

— 2 laboratoires de recherche : ENISE Génie Sensoriel à St-Etienne et Institut Pascal à Clermont

1.1.3 Application

Application culturelle

Dans le cadre de l'application culturelle, il s'agit d'équiper le site du Musée de Solutré en Bourgogne pour une visite immersive en Réalité Augmentée (RA) intitulée "Une journée au Solutréen" sur un scénario original du cabinet de muséographie : l'Atelier des Charrons. L'objectif est de plonger le visiteur dans des scènes de la vie quotidienne des solutréens, 18 000 - 22 000 ans avant notre ère. Dans le cadre intérieur du musée mais aussi en pleine nature au pied de la fameuse Roche de Solutré, équipé de lunettes de RA, l'usager intégrera un groupe de Solutréens "virtuels", pourra visualiser leurs habitats mais aussi vivre et parfois participer à quelques-unes de leurs activités : chasse d'animaux en images de synthèse, fabrication et utilisation d'outillage, allumage d'un feu, confection des repas ...

FIGURE 1.1 – Un scénario « Atelier des Charrons ».

Application Industrie

Les applications de réalité augmentée dans l'industrie sont multiples.

Dans le cadre du projet REVE 5D trois usages sont à l'étude :

— L'aide à la maintenance : le mécanicien est guidé dans son processus de maintenance, à travers le système qui lui indique directement dans son champ de vision les étapes à suivre. Chaque étape peut inclure des informations visuelles et sonores qui seront placées dans son environnement réel. Ceci permet un guidage précis et intuitif sans avoir à quitter son poste de travail. Ce module inclura également de la télé-assistance où un expert à distance pourra guider le mécanicien grâce à une communication vocale et en ajoutant des informations visuelles dans son champ de vision.

— Vérification de configuration : le système permettra de vérifier la configuration d'un produit. Il indiquera en réalité augmentée la configuration théorique et le superposera avec le produit réel. L'opérateur pourra donc identifier les éventuelles différences et valider que le produit réel est conforme à la théorie.

— L'exploration de système : l'utilisateur sera capable d'explorer un système, en éclatant celui-ci en sous-systèmes, jusqu'à arriver aux détails des pièces. A chaque niveau il sera possible d'avoir des informations (documents, précaution, fonctionnements...) sur toutes les pièces. L'utilisateur sera seul ou accompagné par ses collaborateurs lors de cette exploration.

FIGURE 1.2 – Découvrir un lieu virtuellement.

1.1.4 Contributions

Les travaux effectués durant cette thèse ont donné lieu à plusieurs publications scientifiques :

1. Dans El OUNI et al.[80], une nouvelle méthode d'estimation de la pose a été présentée. Cette méthode est basée sur deux hypothèses : la direction verticale est connue et la hauteur de la caméra est fixe.

2. Dans El OUNI al.[66], nous présentons une nouvelle méthode capable de gérer plusieurs labels à chaque point-clé (par l'intégration des informations sémantiques dans le descripteur visuel) afin de rendre la phase de mise en correspondance entre des caractéristiques 2D et les points 3D plus pertinente.

3. Dans El OUNI al.[79], une nouvelle méthode de recherche d'image par similarité est fondée sur la combinaison des informations sémantiques avec les descripteurs visuels.

4. Dans El OUNI al.[67], ensemble de méthodes pour la recherche d'image par similarité est obtenue par le codage de la sortie 2D de l'architecture de réseaux de neurones profonds.

1.1.5 Organisation du mémoire

Ce manuscrit est divisé en quatre principaux chapitres :

1. Le premier chapitre présente une bibliographie des différents algorithmes de SLAM et SfM. Le problème de localisation et de cartographie est rappelé. Différentes applications du SLAM dans le domaine de la robotique mobile sont présentées. Ensuite, les algorithmes de localisation visuelle à partir d'une image sont étudiés en fonction de l'approche utilisée (directe, indirecte), les types de correspondance (2D-3D, 3D-2D) ainsi que leurs performances en termes de résultat de localisation. Puis, différents algorithmes de la segmentation sémantique et leurs applications seront étudiées. Finalement, les bases de données utilisées dans le contexte de la localisation sont présentées.

2. Dans le deuxième chapitre, nous présentons une méthode indirecte pour calculer la pose à partir d'une paire d'images prise d'une caméra montée sur un véhicule ou un robot à roues. L'idée principale est de créer un nouveau descripteur combinant des informations géométriques et visuelles pour les utiliser dans le processus d'appariement et l'intégrer dans le processus de construction de la notion de "mots visuels". Ensuite, nous présentons les résultats de la méthode proposée avec le descripteur KAZE sur deux bases de données distinctes (Musée, Oxford).

3. Dans le troisième chapitre, nous présentons une méthode basée sur les réseaux de neurones convolutionnels qui traite le problème de la labellisation des points d'intérêt dans l'image. Deux contributions principales sont présentées dans ce chapitre. La première permet d'attribuer plus d'un label à chaque point-clé. La seconde permet d'intégrer les informations sémantiques dans le descripteur visuel. Nous montrons expérimentalement sur plusieurs bases de données distinctes (Rome, Dubrovnick) que l'ajout d'informations sémantiques à la fois pour la requête et la carte 3D améliore clairement la précision de la pose récupérée.

4. Les méthodes indirectes transforment la tâche de localisation en un problème de récupération d'images et fournit des informations approximatives sur l'emplacement d'acquisition de l'image requête en fonction des images récupérées. Dans le quatrième chapitre, nous exploitons de deux façons différentes la segmentation sémantique pour localiser une requête dans une carte 3D. La première approche est basée sur la combinaison entre les informations sémantiques et les caractéristiques visuelles. Une deuxième contribution permet de créer une signature visuelle basée uniquement sur la sortie 2D de l'architecture de réseaux de neurones.

Chapitre 2

État de l'art

2.1 Le problème de la reconstruction 3D

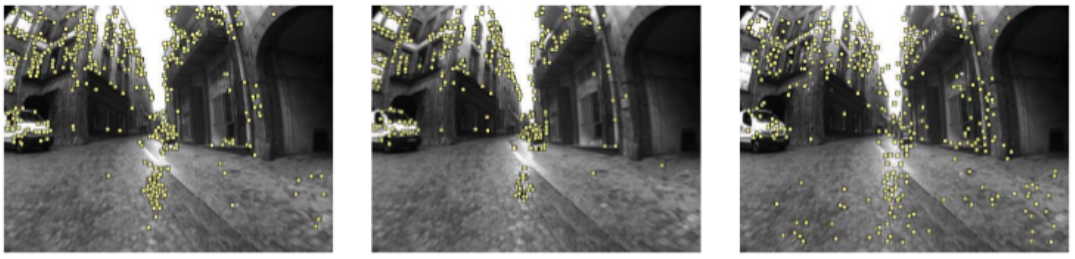
2.1.1 Descripteurs d'image

Dans la vision par ordinateur, les descripteurs d'images permettent de décrire les caractéristiques visuelles qui caractérisent le contenu des images. Plus précisément, ces algorithmes produisent des descriptions sous forme d'un vecteur numérique par point clé. Ils décrivent des caractéristiques telles que la forme, la couleur, la texture, etc... Les descripteurs d'images sont généralement séparés en deux grandes familles : les descripteurs visuels et les descripteurs basés sur l'apprentissage profond. Dans la suite de cette section, nous montrons en détail les caractéristiques et les performances de chaque catégorie.

2.1.1.1 Détecteur visuel

La détection des points clés est l'une des étapes essentielles pour de nombreuses applications en vision par ordinateur, telles que la recherche d'image par le contenu, la détection d'objets, la correspondance entre deux images, etc... Les points d'intérêt d'une image sont localisées dans des zones jugées « intéressantes ». Ces zones peuvent apparaître sous la forme de points, de régions, etc... De nombreux algorithmes reposent sur la détection de points d'intérêt, ou points clés et sur le calcul d'une description à partir de la région entourant le point d'intérêt. Il existe une quantité importante de détecteurs de points d'intérêt. Nous n'en décrivons que quelques-uns. Le détecteur de coins de Harris [32] est un détecteur de coin couramment utilisé dans les algorithmes de vision artificielle pour extraire les coins et déduire les caractéristiques d'une image. Il a été introduit pour la première fois par Chris Harris et Mike Stephens en 1988 lors de l'amélioration du détecteur de coins de Moravec. Il a été amélioré et adopté dans de nombreux algorithmes afin de prétraiter les images pour certaines applications. Un coin est un point dont le voisinage contient deux directions de bord ou de contour dominantes et différentes. En d'autres termes, un coin

peut être interprété comme la jonction de deux bords, un bord étant défini comme un changement soudain de la luminosité. Les coins sont des éléments importants de l'image. Ils sont généralement désignés comme des points d'intérêt invariants par rapport à la translation, la rotation et à l'éclairage. Le détecteur FAST (Features from Accelerated Segment Test)[89] est une méthode de détection d'angles, qui peut être utilisée pour extraire des points, puis pour suivre et cartographier des objets dans de nombreuses tâches de vision par ordinateur. Le détecteur de coins FAST a été développé à l'origine par Edward Rosten et Tom Drummond et a été publié en 2006. L'avantage le plus notable du détecteur de coin FAST est son efficacité. De plus, lorsque des techniques d'apprentissage automatique sont appliquées, des performances supérieures en ce qui concerne le temps de calcul et les ressources peuvent être obtenues. Le détecteur de coin FAST convient très bien aux applications de traitement vidéo en temps réel en raison de ses performances en terme de vitesse. Le détecteur Hessian [71] est un détecteur de caractéristiques utilisé dans les domaines de la vision par ordinateur et de l'analyse d'images. Comme les autres détecteurs de caractéristiques, le détecteur Hessian est utilisé comme étape de prétraitement pour les algorithmes qui s'appuient sur les points clés. Le détecteur de régions Harris appartient à la catégorie de détection de caractéristiques.



(a) Harris (b) FAST (c) Fast Hessian

FIGURE 2.1 – Des points d'intérêt avec 3 détecteurs différents.

La Figure 2.1 montre un exemple d'image sur laquelle on a appliqué trois détecteurs différents : Harris, FAST, et Hessian. En résumant, un détecteur de points d'intérêt consiste à calculer un score d'intérêt pour chaque pixel de l'image et à retourner les points les plus saillants. Cependant, il existe de nombreux algorithmes qui transforment les scores des points d'intérêt sous forme d'un vecteur. Les descripteurs codent les informations intéressantes à partir des points clés puis les utilisent pour différencier une caractéristique d'une autre. Idéalement, cette information serait invariante lors de petites transformations d'image afin de pouvoir retrouver les mêmes descripteurs si l'image subit des changements mineurs (translation, rotation, etc).

2.1.1.2 Les descripteurs locaux

Le descripteur SIFT (Scale Invariant Feature Transform) [64] est un algorithme de détection des caractéristiques visuelles permettant de détecter et de décrire les caractéristiques locales dans les images. Il a été breveté au Canada par l'Université de la Colombie-Britannique et publié par David Lowe en 1999. Les applications incluent la reconnaissance d'objets, la cartographie et la navigation robotique, l'assemblage d'images, la modélisation 3D, la reconnaissance des visages, le suivi vidéo. Le descripteur SIFT peut identifier de manière robuste les objets car le descripteur SIFT est invariant au changement d'échelle, d'orientation, et à des modifications d'éclairage. Les caractéristiques extraites par SIFT sont également robustes au bruit et aux changements mineurs de points de vue. Le descripteur SIFT met facilement en correspondance des points entre deux images, mais la grande dimension du vecteur produit peut être un problème, généralement des algorithmes de faible complexité, tels que les arbres, sont utiles pour optimiser le temps d'exécution. Le descripteur SURF [19](Speeded Up Robust Features) est un algorithme de détection de caractéristiques et aussi un descripteur. Il est utilisé dans le domaine de la vision par ordinateur, pour des tâches de détection d'objets ou de reconstruction 3D. SURF est partiellement inspiré par le descripteur SIFT, qu'il surpasse en rapidité et, selon ses auteurs, en robustesse pour différentes transformations d'images. L'obtention du descripteur SURF est une suite de deux étapes. La première étape consiste à fixer une orientation reproductible sur la base des informations d'une région circulaire autour du point clé. Puis, la création d'une région carrée selon l'orientation sélectionnée pour l'extraction du descripteur SURF.

Le descripteur KAZE [14] est une nouvelle méthode inspirée du descripteur SIFT. La méthode KAZE est un algorithme multi-échelles de détection et de description des caractéristiques 2D. Une version plus rapide de KAZE nommée AKAZE [14] utilise un descripteur binaire pour augmenter encore la vitesse. Pour créer la description du point clé, AKAZE utilise une version modifiée du descripteur Local Difference Binary (LDB). Dans KAZE il existe trois types de base de détecteurs de point d'intérêt dans une image :

— un détecteur basé région

— un détecteur basé contour

— un détecteur basé forme

La figure **??** montre une image avec des points clés détectés avec ces trois différents détecteurs.

2.1.1.3 Descripteurs par apprentissage profond

Afin d'adapter les descripteurs aux spécificités de la collection considérée, les réseaux de neurones à architecture profonde ont récemment connu du succès. L'approche L2-Net [109] est l'un des descripteurs locaux profonds les plus récents et les

plus performants. L2-Net applique la fonction de perte (loss) aux cartes de fonctionnalités intermédiaires et la fonction de perte intègre plusieurs attributs. L'approche HardNet [72] étend L2-Net et constitue actuellement le descripteur le plus récent. Leur caractéristique commune, partagée par tous les ancêtres, est qu'ils utilisent des couches CNN communes dans leur architecture. En conséquence, les informations spatiales des cartes ne sont pas explicitement codées, mais uniquement traitées avec une couche FC (fully connected) standard. L'approche SuperPoint [33] est une méthode semi-supervisée basée sur l'apprentissage profond adapté pour les détecteurs de points d'intérêt et les descripteurs. Cette méthode est capable de résoudre un grand nombre de problèmes géométriques à vues multiples dans la vision par ordinateur. Patch-CKN est basé sur les réseaux à noyau convolutif (CKN). Les CKN ont été initialement introduits pour la classification des images. L'approche [81] propose un descripteur de patch basé sur une architecture CKN, en utilisant une procédure stochastique simple et rapide pour calculer les caractéristiques dans l'image.

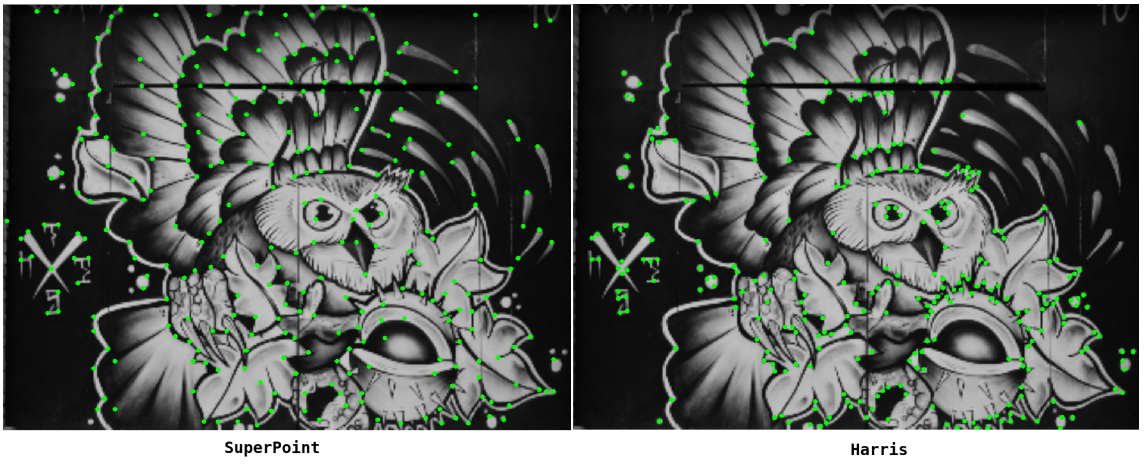


FIGURE 2.2 – Détection des points d'intérêt avec deux détecteurs différents SuperPoint[33] et Harris[32].

L'approche LF-Net[78] a deux composants principaux. Le premier est un réseau dense, multi-échelles et entièrement convolutionnel qui renvoie les emplacements, échelles et orientations des points clés. Il est conçu pour obtenir un temps d'inférence rapide et pour être indépendant de la taille de l'image. Le second est un réseau qui produit des descripteurs locaux à partir de correctifs re-cadrés autour des points clés produits par le premier réseau.

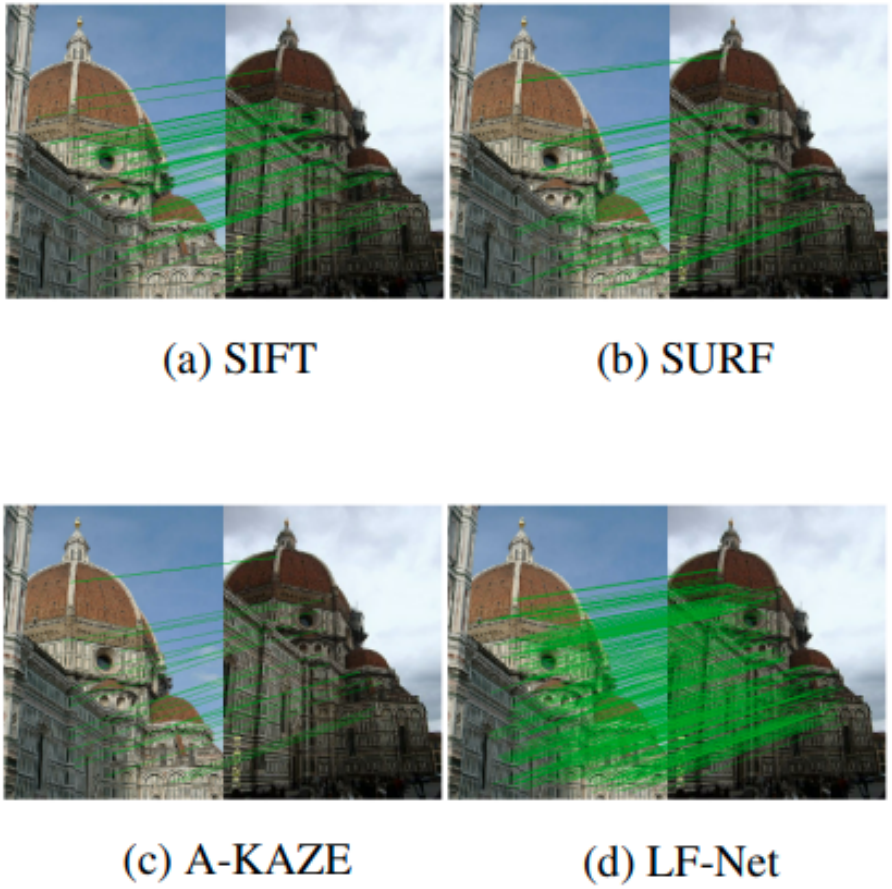


FIGURE 2.3 – Résultats d'appariements avec des correspondances correctes en vert pour LF-Net [78] contre SIFT, SURF et A-KAZE.

La figure 2.3 montre l'efficacité de l'approche LF-Net [78] en comparant les appariements trouvés par différentes méthodes basées sur la description visuelle des caractéristiques.

2.1.1.4 Correspondance entre les caractéristiques

La correspondance des images fait partie de nombreuses applications de vision par ordinateur telles que le calibrage de caméra, la reconnaissance d'objets et l'établissement des correspondances entre deux images de la même scène ou objet. L'approche consiste à détecter un ensemble de points d'intérêt associés chacun à des descripteurs tel que SIFT ou SuperPoint à partir d'une image. Une fois les caractéristiques extraites, l'étape suivante consiste à établir des correspondances entre les images et apparier les points similaires en calculant la norme 2 de la distance entre les descripteurs. La correspondance entre les caractéristiques K_q et K_{p1} est acceptée

si le test ratio suivant est vérifié :

$$\frac{||d_{kq} - d_{kp1}||_2}{||d_{kq} - d_{kp2}||_2} < \epsilon \tag{2.1}$$

où d_{kq} est le descripteur dans la première image, d_{kp1} d_{kp2} sont les deux descripteurs issus de la deuxième image candidate les plus proche de d_{kq}.

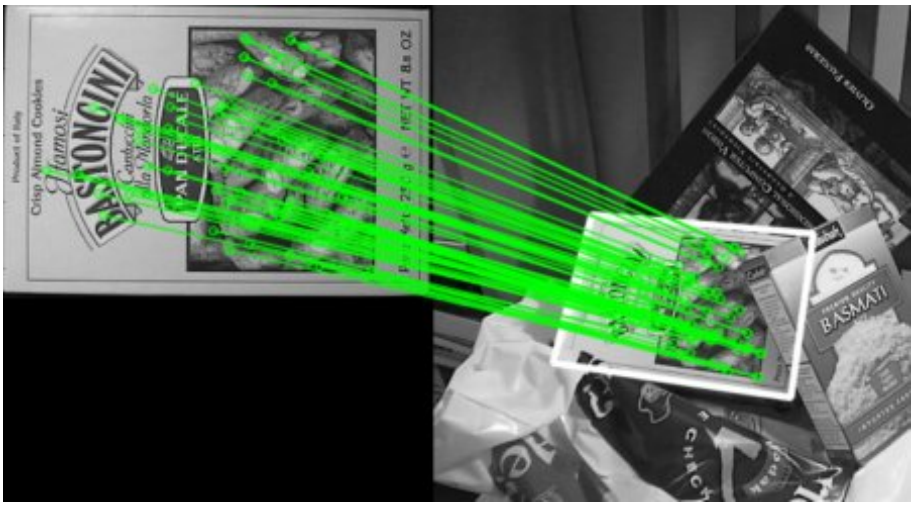


FIGURE 2.4 – Correspondance entre deux images en utilisant le descripteur visuel SIFT.

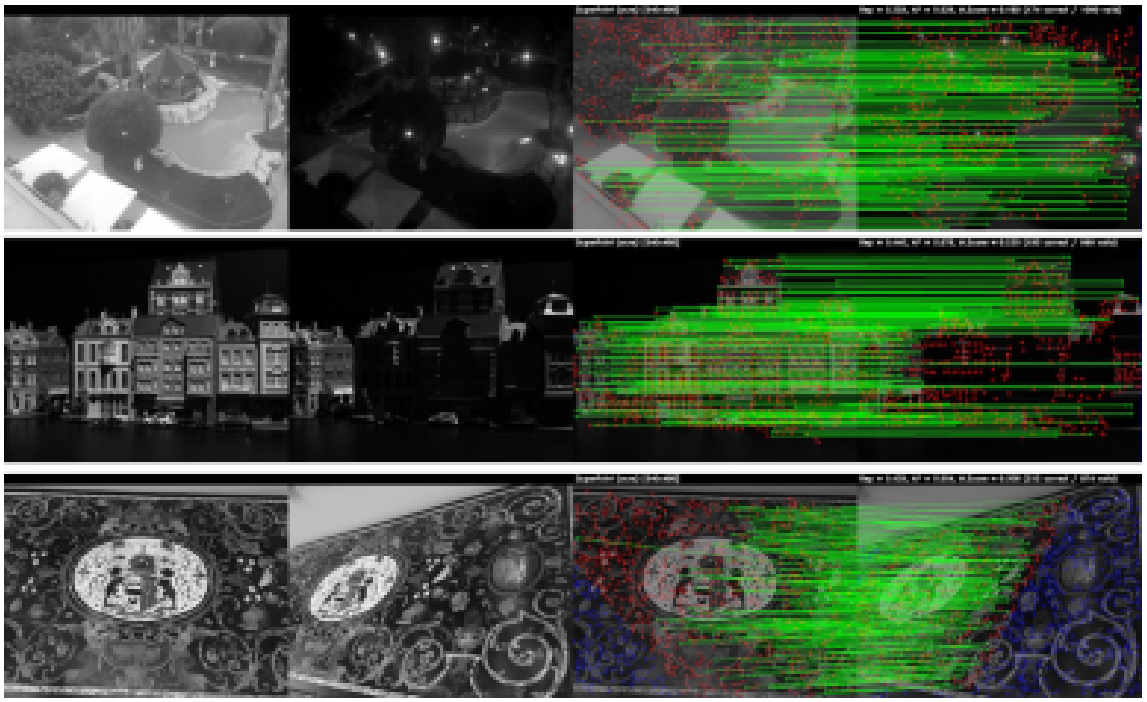


FIGURE 2.5 – Correspondance entre des images en utilisant un descripteur par apprentissage profond (SuperPoint [33]).

Les figures 2.4 et 2.5 montrent comment les points clés peuvent être liés par l'équation 2.1. La première figure montre les points clés extraits par les descripteurs visuels (dans notre exemple le descripteur est SIFT). Dans la deuxième, un algorithme de détection des points d'intérêt basé sur l'apprentissage profond [33] a été utilisé pour extraire les descripteurs.

2.1.2 La reconstruction 3D

Le problème de la reconstruction 3D à partir d'un ensemble d'images ou vidéo prise par une caméra est un sujet important dans le domaine de la vision par ordinateur. C'est un verrou scientifique et une technologie de base pour un large éventail de domaines, tels que la conception géométrique assistée par ordinateur, l'infographie, l'animation par ordinateur, la vision par ordinateur, l'imagerie médicale, la réalité virtuelle, les médias numériques, etc. Il s'agit de reconstruire la géométrie 3D d'une scène (figure 2.6) à partir d'un ensemble d'images prises de différents points de vue. Ce processus peut être accompli par des méthodes actives ou passives. Le principe des méthodes actives est de remplacer une des caméras par un appareil qui projette une lumière structurée sur l'objet ; Les méthodes passives n'utilisent que des caméras pour les acquisitions. Elles peuvent utiliser une ou plusieurs caméras, mais doivent prendre au moins deux images de l'objet sous différents angles de vue pour réaliser une reconstruction 3D. Deux technologies principales sont capables de construire

une carte 3D d'une scène à partir des images : les approches SLAM censées fonctionner en temps réel sur une séquence ordonnée d'images acquises à partir d'une configuration de caméra ; les approches SfM appliquées sur un ensemble d'images non ordonné.

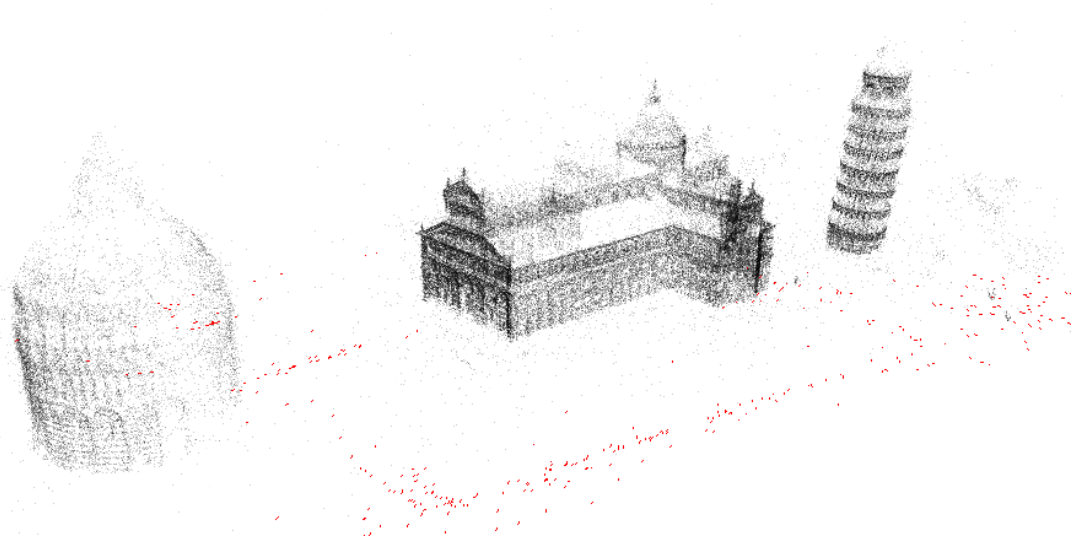


FIGURE 2.6 – Nuage de points 3D (les points 3D sont représentés en noir et les caméras sont en rouge [10].

2.1.3 Géométrie épipolaire et triangulation

La géométrie épipolaire est capable de décrire la relation entre les points 3D et les positions des points correspondants dans une image stéréo. Une relation géométrique entre les points 3D et leurs projections sur les images 2D existe lorsque deux caméras observent une même scène 3D à partir de deux vues différentes (voir la figure 2.7). Ces relations sont basées sur l'hypothèse que les caméras peuvent être approximées par le modèle de caméra sténopé. La géométrie épipolaire est utilisée pour simplifier la recherche de correspondances, calculer le déplacement entre les caméras, et reconstruire la scène à partir d'une paire d'images stéréoscopiques.

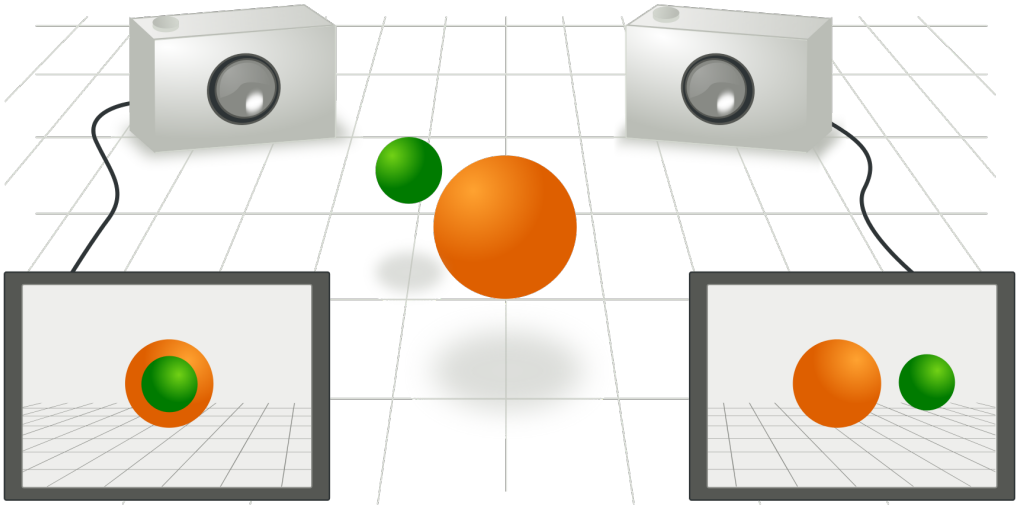


FIGURE 2.7 – Deux caméras observent la même scène de deux positions différentes [6].

Dans la reconstruction 3D, après avoir utilisé les contraintes géométriques épipolaires pour estimer le mouvement de la caméra, nous devons également estimer la profondeur des points mis en correspondance par la triangulation. La triangulation est parfois appelée reconstruction ou intersection. C'est une technique permettant de calculer la position 3D d'un point à partir de ses projections dans les deux images stéréoscopiques. Dans le cas d'une caméra en mouvement, les caractéristiques 2D doivent être suivies sur plusieurs images jusqu'à ce qu'elles soient observées avec une ligne de base suffisamment large pour fournir une triangulation stable. La figure 2.8 montre le calcul des coordonnées du point 3D X à partir des appariements x1 et x2 (points homologues correspondants). Le point 3D est issu de l'intersection des rayons noirs.

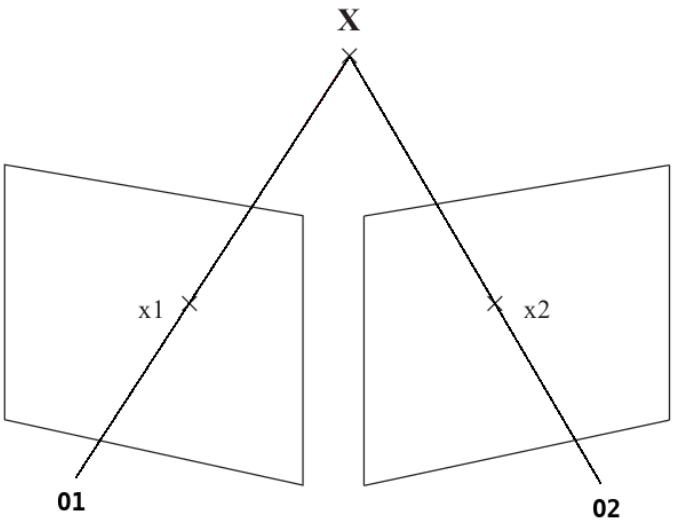


FIGURE 2.8 – Triangulation dans le cas idéal de la géométrie épipolaire (pas de bruit sur la localisation des points homologues x1 et x2).

2.1.4 Calcul de la pose

L'estimation de pose d'une caméra consiste à déterminer la rotation et la translation d'une caméra dans un environnement connu. Dans cette thèse, nous avons travaillé sur l'estimation de la pose d'une image acquise avec une caméra calibrée. La première étape est d'extraire les descripteurs 2D de l'image requête et les descripteurs 2D de la carte 3D. La seconde étape consiste en la mise en correspondance (points homologues) de ces descripteurs afin d'obtenir des appariements 2D-3D. Deux points homologues sont deux pixels représentant le même objet dans deux images différentes. À partir de ces correspondances 2D-3D ou 3D-2D l'estimation de la pose peut être effectuée. Lorsque les caméras sont calibrées l'estimation peut s'établir par au moins trois correspondances 2D-3D et 6 correspondances dans le cas des caméras non calibrées. Les correspondances 2D-3D entre deux images permettent de définir une relation géométrique entre les repères des deux caméras. Le calcul de pose possède un deuxième intérêt autre que l'estimation de la relation géométrique entre deux points de vue. Deux paramètres sont fournis afin de calculer les correspondances entre deux images : la liste des correspondances et la matrice essentielle.

En effet, la matrice essentielle permet de filtrer les valeurs aberrantes des appariements en ayant recours à une méthode de filtration de type RANSAC (RANdom SAmple Consensus [38]). L'algorithme RANSAC est très utilisé en vision par ordinateur, par exemple pour le calcul de pose ou le calcul de la géométrie épipolaire. Dans le contexte de notre thèse, RANSAC est utilisé pour détecter puis éliminer les

valeurs aberrantes parmi les correspondances 2D-3D.

Un avantage de RANSAC est sa capacité à effectuer une estimation robuste des paramètres du modèle, c'est-à-dire qu'il peut estimer les paramètres avec un degré élevé de précision, même si un nombre significatif de valeurs aberrantes sont présentes dans l'ensemble de données. L'inconvénient de RANSAC est que lorsque le nombre d'itérations calculées est limité, la solution obtenue peut ne pas être optimale et il se peut que le modèle contienne du bruit. Il existe de nombreuses alternatives à l'algorithme classique RANSAC. P-RANSAC [26] donne la priorité à des caractéristiques spécifiques au cours de l'étape de sélection aléatoire. On peut aussi citer LO-RANSAC utilisé dans [83] et AC-RANSAC dans [86] [85]. La nouvelle méthode F-SORT présentée par Chan et al. [24] montre des résultats remarquables en terme de qualité et d'efficacité de calcul.

2.1.5 Méthodes de reconstruction par vision

Les environnements inconnus peuvent être reconstruits à partir de vidéos ou d'un ensemble d'images. Ici, il s'agit de deux technologies principales capables de reproduire la géométrie 3D d'une scène et de connaître les coordonnées 3D des points d'intérêt à partir d'un ensemble d'images prises sous différents points de vue d'une scène : SLAM et SfM. Le processus de localisation et cartographie simultanées (SLAM) consiste à construire la carte de l'environnement exploré, tout en localisant le robot sur celle-ci. Initialement, le robot n'a aucune information a priori sur l'environnement et sa position. Simultanément, les poses successives de la caméra sont calculées en temps réel. Si les environnements inconnus sont reconstruits à partir d'images multi-vues non ordonnées, on parle de la structure acquise à partir du mouvement (SfM).

2.1.5.1 Cartographie et localisation simultanées : SLAM.

L'approche SLAM est une méthode couramment utilisée pour aider les robots à cartographier les zones et à trouver leur chemin. Donc à l'aide du SLAM, les robots créent leurs propres cartes au fur et à mesure qu'ils découvrent leurs environnements d'évolution. Cet algorithme leur permet de connaître leur position et orientation en traitant les données collectées à partir des capteurs pour créer une carte de navigation. La dernière décennie a vu le développement rapide du SLAM visuel basé sur l'image clé (figure 2.9). Les capteurs les plus utilisés sont les caméras. Dans ce contexte, le terme générique "caméra" peut représenter trois types de dispositif : caméra monoculaire, tête stéréo et caméra RGB-D. Si nous avons une seule caméra avec laquelle nous capturons la vidéo alors ce système est appelé un système monoculaire. L'estimation de la profondeur à partir de ce système est une tâche très difficile. Une tête stéréo est une combinaison de deux caméras monoculaires dont on connaît la position relative des deux caméras. Une caméra RGB-D également appelée caméra de profondeur, est une caméra active qui estime directement la pro-

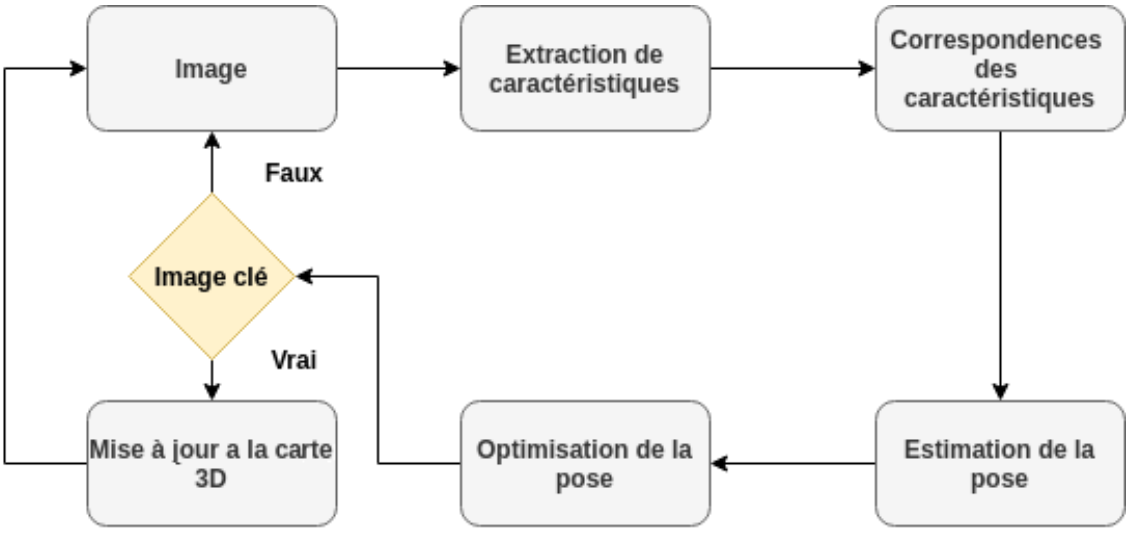


FIGURE 2.9 – Architecture générale de l'algorithme SLAM

fondeur de chaque pixel. Plusieurs systèmes SLAM ont été implémentés en se basant sur ces différents types de caméras. MonoSLAM [30] est le premier système monoculaire en temps réel basé sur le filtre EKF (extended kalman filter). ORB-SLAM [75] est une solution pour plusieurs types de caméras : caméras monoculaires, stéréo et RGB-D basé sur le descripteur binaire ORB. Il est capable de calculer en temps réel la trajectoire de la caméra et de construire la scène 3D dans un environnement de grande variété. Il comporte une initialisation automatique et robuste des scènes planaires et non planaires et il est capable aussi de détecter les boucles fermées et de relocaliser la caméra en temps réel. Il sélectionne les images clés les plus adaptées pour permettre de maintenir une carte compacte. OpenVSLAM [103] est un SLAM visuel monoculaire, stéréo et RGBD basé sur l'algorithme SLAM tel ORB-SLAM [75], ProSLAM[96]. Il est compatible avec différents types de modèles de caméras et peut être facilement personnalisé pour d'autres modèles de caméras. Les cartes créées peuvent être stockées et chargées, puis OpenVSLAM peut localiser de nouvelles images sur la base des cartes prédéfinies. LSD-SLAM [35] est une technique directe de SLAM monoculaire : au lieu d'utiliser des points clés, il fonctionne directement sur les contenus d'images à la fois pour le suivi et la cartographie. La caméra est suivie à l'aide de l'alignement direct du contenu de l'image, tandis que la géométrie est estimée sous la forme de cartes de profondeur semi-denses. [36] le rend compatible avec les caméras stéréo et [22] le rend compatible avec les caméras omnidirectionnelles. D'autres travaux similaires avec des caméras omnidirectionnelles peuvent être vus dans [58]. PTAM [53] (Parallel Tracking and Mapping) est un système monoculaire de suivi par caméra pour la réalité augmentée. Il est le premier système SLAM qui met en parallèle le suivi et la cartographie.

2.1.5.2 Structure acquise à partir d'un mouvement : SfM

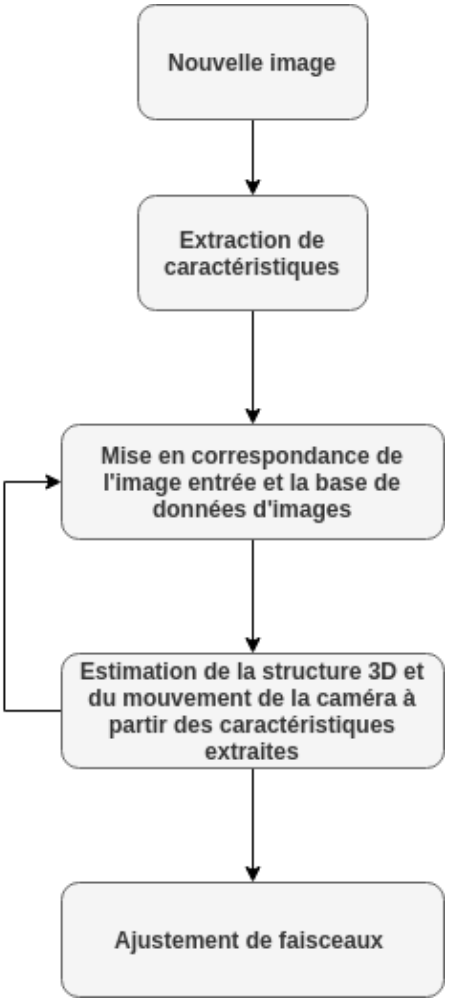


FIGURE 2.10 – Architecture générale de l'algorithme SfM.

Pour la reconstruction de scènes complexes et pour la navigation autonome, il existe différentes méthodes de reconstruction 3D. En utilisant une seule caméra, la structure 3D de la scène peut être obtenue par les techniques dites SfM. Cette méthode exploite le mouvement de la caméra dans la scène pour obtenir différents points de vue et ensuite reconstruit la structure 3D de la scène. À partir d'au moins deux images, la scène peut être reconstruite par les méthodes de stéréovision. La stéréovision consiste à retrouver la structure géométrique de la scène par le calcul de cartes de disparités ou par le calcul de cartes de profondeurs afin d'avoir une reconstruction 3D dense. L'algorithme SfM traite la reconstruction en plusieurs étapes itératives (figure 2.10) qui aboutissent à un nuage des points 3D. Dellaert et al. [31] ont présenté un outil permettant de résoudre le problème SfM sans avoir besoin

d'informations de correspondances a priori. Cet outil peut traiter des images données dans n'importe quel ordre et prises à partir de points de vue très différents. L'algorithme final est simple, facile à mettre en œuvre et rapide. Szeliski et al [108] ont développé des techniques permettant de récupérer la structure et le mouvement de points vus avec 2 caméras ou plus. Ils ont estimé la position de chaque point sur deux images ou plus, en supposant que certains points sont coplanaires. Ces techniques nous permettent d'exploiter directement les homographies existantes entre différentes régions de l'image. Jin et al [48] ont présenté un algorithme permettant d'estimer la structure et le mouvement à l'aide d'une séquence d'images. L'algorithme intègre l'information visuelle en utilisant une classe paramétrable de modèles géométriques pour la scène. L'image courante et l'estimation du mouvement sont combinées en une boucle fermée. Ils ont considéré le problème de la SfM dans le cadre du filtrage non linéaire. La structure inconnue et le mouvement sont estimés en reconstruisant l'état d'un système dynamique non linéaire via un filtre de Kalman. Scaramuzza et al [95] ont discuté d'une technique flexible pour un étalonnage précis à partir du mouvement. La méthode nécessite seulement une caméra pour observer un motif planaire montré sous quelques orientations différentes. Ce travail sur l'étalonnage de la caméra omnidirectionnelle est motivé par l'utilisation de capteurs de vision panoramique pour la reconstruction SfM. Les résultats de la reconstruction sont très bons et la procédure proposée est facile à utiliser et flexible. Agarwal [13] construit un système de reconstruction 3D à partir de grandes collections non organisées de photographies. Le système utilise des algorithmes d'appariement entre les images et la reconstruction 3D. Le système est divisé en trois étapes principales. On commence par l'étape de pré-traitement où les images sont disponibles dans une base de données centrale à partir de laquelle elles sont distribuées aux nœuds de taille fixe. Ensuite, une vérification est faite par l'algorithme RANSAC sur l'ensemble des appariements pour supprimer les "outliers". Enfin, vient le regroupement des caractéristiques afin que l'algorithme d'estimation de la géométrie puisse estimer un seul point 3D à partir de toutes les caractéristiques. Sturm et al [102] ont proposé une méthode utilisant uniquement des matrices fondamentales et des épipoles estimées à partir des données d'images pour récupérer la forme projective et le mouvement à partir d'images multiples d'une scène par la factorisation d'une matrice contenant les cordonnées de tous les points dans toutes les vues. La factorisation n'est possible que lorsque les points de l'image sont correctement mis à l'échelle. L'algorithme fonctionne rapidement et fournit des reconstructions précises. Crandall [28] a présenté une méthode pour les collections d'images non structurées qui prend en compte toutes les photos en même temps plutôt que de créer progressivement une solution. En utilisant toutes les photos disponibles, l'approche calcule une estimation initiale de la position de la caméra, puis utilise un ajustement de faisceaux pour affiner cette estimation. La méthode donne de meilleures reconstructions et est plus rapide que les approches par ajustement de faisceaux incrémental.

2.2 Localisation visuelle basée sur l'image

2.2.1 Définition

La localisation visuelle consiste à estimer la pose (position + orientation) de la caméra ayant pris l'image. Par exemple, récupérer la pose d'une caméra qui a pris une image relativement à un ensemble d'images géo-localisées ou un modèle 3D. La localisation visuelle est un sujet de recherche de plus en plus étudié au cours de la dernière décennie. Cet intérêt est dû à la mise à disposition de grandes bases de données d'images géo-localisées, à la multiplication des systèmes d'acquisition d'images intégrés (caméra sur smartphone) et à la limitation du système de localisation GPS en milieu urbain. Les méthodes de localisation sont divisées en deux catégories (figure 2.11) : les méthodes indirectes qui transforment la tâche de localisation en un problème de récupération d'images et fournissent des informations approximatives sur l'emplacement de l'image requête ; les méthodes directes qui tentent de trouver des correspondances en comparant directement les caractéristiques de l'image requête aux points 3D de la scène.

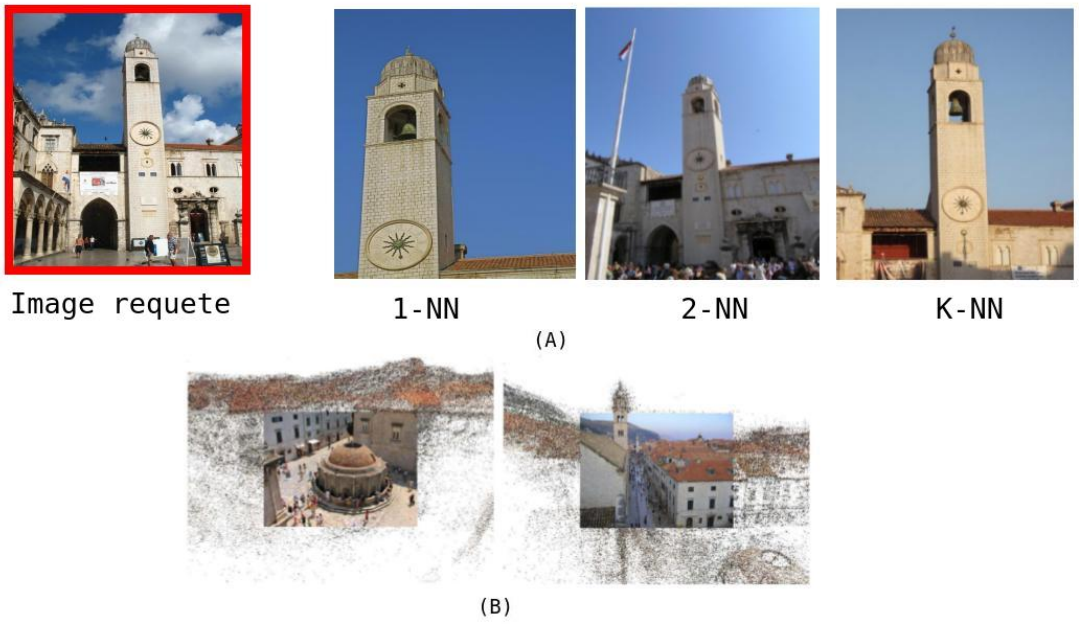


FIGURE 2.11 – (A) méthode indirecte (B) méthode directe.

2.2.2 Méthodes indirectes

Le but des méthodes indirectes est de récupérer l'ensemble des images présentes dans la base de données similaires à l'image requête en entrée. Ceci est un problème

lié à la thématique de la recherche d'images par le contenu. Le problème de récupération d'images comporte trois étapes : description des données visuelles, mesure de similarité entre les descripteurs précédemment extraits et classement des candidats.

2.2.2.1 Description des données visuelles

La récupération d'images basée sur le contenu (CBIR pour Content Based Image Retrieval) est une technique largement utilisée pour récupérer des images à partir de vastes bases de données d'images non étiquetées. Les méthodes de quantification ont été largement adoptées dans ce domaine depuis la contribution de Zisserman [99]. Ils considèrent le problème de la récupération des objets dans une image décrite par des descripteurs locaux de la même manière que la recherche dans des documents textes. Les mots sont équivalents dans le domaine de l'image aux descripteurs locaux et un dictionnaire est construit sur un large ensemble de descripteurs extraits depuis la base de données visuelles. Ces descripteurs sont regroupés pour réduire la taille du dictionnaire ; les centroïdes des groupes sont alors appelés mots visuels. Le sac de mots (BoVW 2.12) est un vecteur de la dimension du dictionnaire contenant la fréquence visuelle des mots d'un document visuel spécifique. Avec cette représentation, la similitude des données peut être calculée efficacement par un simple produit scalaire entre les vecteurs visuels représentatifs de la fréquence des mots. Il y a donc trois étapes principales pour construire les sacs de mots :

1. Détection et extraction des descripteurs de chaque image et construction d'un dictionnaire visuel. La détection des caractéristiques et l'extraction des descripteurs dans une image peuvent être effectuées à l'aide d'algorithmes d'extraction de caractéristiques (par exemple, SIFT, KAZE, etc.).
2. Création de clusters à partir des descripteurs (avec K-Means, DBSCAN ou un autre algorithme de clustering). Le centre de chaque groupe sera utilisé comme vocabulaire du dictionnaire visuel.
3. Création des histogrammes de fréquence de vocabulaire dans l'image. Ces histogrammes sont appelés "sac de mots visuels (BoVW)".

Cependant, les techniques traditionnelles ne sont pas satisfaisantes pour une récupération robuste des informations. Les méthodes d'apprentissage profond sont proposées pour répondre à ces questions. Ces algorithmes utilisent des données pour former des réseaux neuronaux permettant d'effectuer diverses tâches d'apprentissage automatique, telle que la classification de différentes classes d'objets. Les réseaux de neurones convolutionnels sont des algorithmes d'apprentissage profond particulièrement adaptés à l'analyse d'images. La figure 2.13 montre une architecture composée de différentes couches de réseaux de neurones convolutionnels. Tout d'abord, une image est poussée dans le réseau ; c'est ce qu'on appelle l'image d'entrée. Ensuite, l'image d'entrée passe à travers différentes couches ; c'est la partie convolutionnelle du réseau. Enfin, le réseau de neurones peut interpréter le contenu de l'image d'entrée

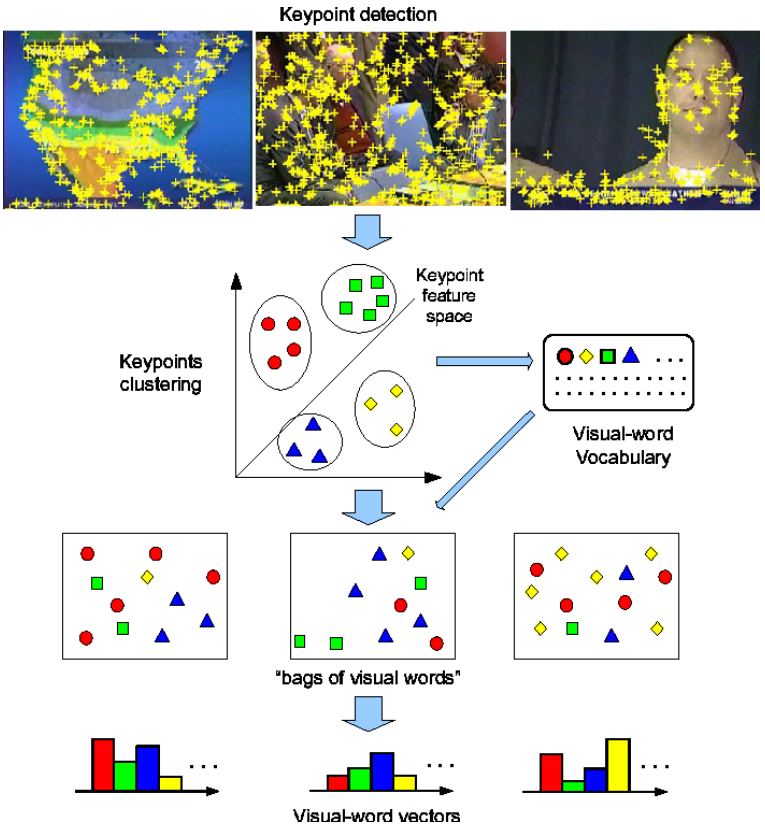


FIGURE 2.12 – Modèle de Sac de Mots visuels [29].

(7 ici). AlexNet[45] était parmi les premières implémentions pour la reconnaissance d'image par l'apprentissage profond. Le réseau de neurones fournit un vecteur de caractéristiques globales qui est ensuite classé par rapport à chaque vecteur de la base de données en fonction de sa distance euclidienne. Plusieurs améliorations [107] [126] [115] au cours de ces dernières années ont été appliquées sur les descripteurs CNN. NetVlad [15] est une architecture adaptée aux tâches de localisation dans les situations différentes telles que le changement d'échelle ou de saison.

2.2.2.2 Similarité entre les descripteurs

La recherche de similarité peut être coûteuse en calcul lorsque les données sont décrites par un descripteur volumineux, c'est-à-dire un vecteur de grande dimension. En particulier, les caractéristiques locales permettent de produire un grand nombre de descripteurs pour chaque image. La comparaison entre descripteurs est une opération simple. Elle consiste en un simple calcul de distance (norme L2) entre les vecteurs. Cependant, lorsque le nombre de descripteurs est très grand, une approche exhaustive ne peut pas être envisagée. Des algorithmes de recherche de similarité sont alors utilisés. La réduction de la dimension du descripteur est souvent effec-

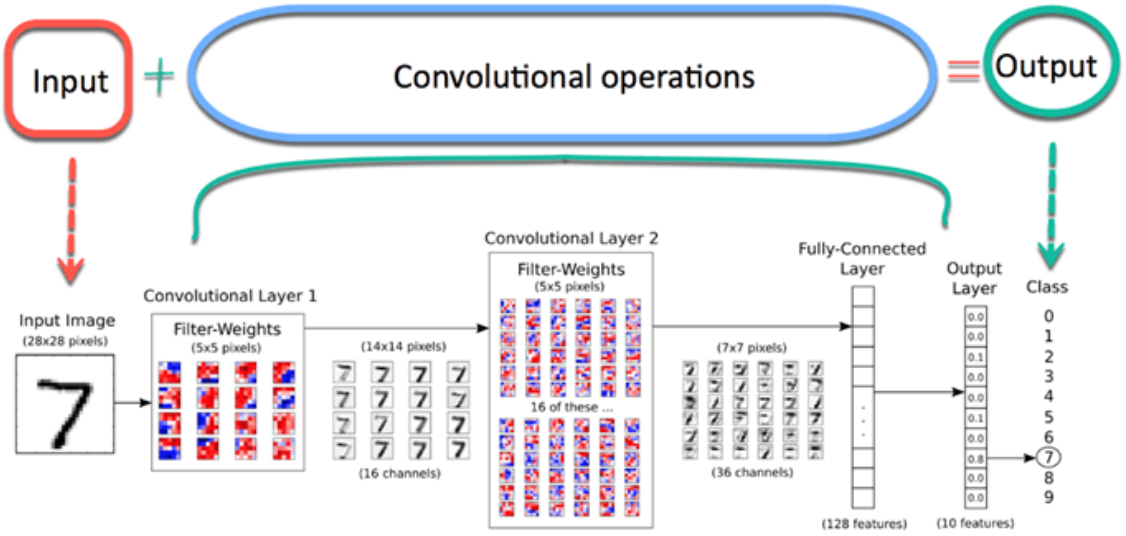


FIGURE 2.13 – Exemple d'architecture CNN pour prédire les chiffres [1].

tuée pour réduire le temps d'exécution. La technique la plus utilisée reste l'analyse en composantes principales (ACP). L'approche ACP est appliquée sur un vecteur de haute dimension, on peut citer par exemple les poids extraits des couches de CNN [15], [41]. L'ACP a également été utilisée pour réduire la taille des descripteurs locaux [52], [112] ou des descripteurs globaux [77]. Dans certains travaux, lorsque la quantité de données à comparer reste acceptable, une procédure de récupération par force brute peut être utilisée pour récupérer les voisins les plus proches. C'est le cas lorsqu'un seul vecteur est utilisé pour décrire une image, c'est-à-dire lorsque des descripteurs globaux sont utilisés. La recherche exacte du plus proche voisin devient impossible lorsque la quantité et/ou la dimension des caractéristiques sont trop grandes. D'où la nécessité d'avoir des algorithmes performants en vitesse et de faible complexité. Plusieurs algorithmes de recherche de voisins les plus proches sont implémentés dans la bibliothèque FLANN [74].
L'apprentissage supervisé est une alternative aux méthodes de recherche du voisin le plus proche. Le classifieur SVM (support vecteur machine) est utilisé dans de nombreux travaux [98], [21] [69], [17] pour transformer la recherche de similarité en tâche de classification. Dans [69], [17] les auteurs créent des classificateurs linéaires sur les descripteurs HOG pour récupérer de manière robuste des images similaires qui présentent des changements d'apparence. Aubry et al. [69] profite des avantages de la représentation des données par analyse discriminante linéaire (LDA) pour éviter l'entraînement coûteux de SVM. De même, Kim et al. [49] entraîne le classificateur SVM pour prédire les descripteurs. Cela améliore le processus d'appariement et réduit le nombre de descripteurs à comparer avec la base de données.

Les données peuvent être traitées après la recherche par similarité pour améliorer le résultat final. Ces méthodes sont largement utilisées pour reclasser la liste des

candidats, améliorant ainsi la pertinence des données extraites. Un système parfait devrait récupérer dans un premier temps la position du document visuel le plus proche présent dans la base de données. Cependant, les candidats les mieux classés dans la liste des données extraites pourraient bénéficier d'une étape ultérieure d'estimation de la pose. Le système présenté par Cao et Snavely [21] vise à augmenter la diversité des images récupérées en introduisant un reclassement probabiliste. Enfin, le contrôle de la cohérence géométrique est souvent utilisé pour rejeter les erreurs d'appariement. La pose entre la requête et les candidats de la base de données est calculée en considérant l'homographie et les candidats qui produisent la pose la plus cohérente. Contrairement aux méthodes conventionnelles (CNN) qui consistent à la récupération des objets, la méthode indirecte peut bénéficier des informations de géo-localisation associées aux documents présents dans la base de données (généralement des images). Ces informations peuvent être utilisées pour construire un graphique structuré pour la recherche de similarité [113] [21] ou exploitées pour la reclassement des candidats [120], [121], [90]. Zamir et Shah [120] introduisent ce reclassement géographique après un algorithme de récupération d'images pour éliminer rapidement les candidats non pertinents. Les auteurs [121] intègrent le processus de mise en correspondance dans un schéma de graphe afin de récupérer des candidats cohérents en fonction de l'étiquette GPS associée aux données visuelles. [90] améliore la pertinence de la liste des candidats classés en utilisant la position et les méta-données associées aux images de la base de données.

2.2.3 Méthodes directes

À ce stade, nous introduisons la notion de méthodes directes (figure 2.14) qui récupèrent instantanément la pose exacte. En effet, les approches directes tentent de trouver des correspondances en comparant directement les caractéristiques de l'image et les points de la scène 3D construits selon l'approche SfM ou SLAM. La procédure consiste donc à associer à chaque *feature* un point 3D du nuage de points.

2.2.3.1 Méthode par point d'intérêt

Cette famille consiste à faire correspondre les caractéristiques dans l'image requête et dans le modèle 3D. Sattler et al [93] proposent une méthode directe basée sur des mots visuels pour établir les correspondances entre les caractéristiques requête et les caractéristiques de la scène 3D. Pour les images qui contiennent moins de primitives, Sattler et al [92] améliorent leur travail en ajoutant une direction 3D-2D pour augmenter le nombre de correspondances trouvées à l'aide de la stratégie 2D-3D. D'autre part, cette idée conduit à des correspondances de meilleure qualité. L'utilisation de la covisibilité ou de la cooccurrence dans la localisation visuelle a été largement utilisée dans plusieurs travaux. [101] propose une méthode basée sur des graphes et des mots visuels. Le graphe composé de nœuds qui représente les informations visuelles extraites d'une image et les liens entre deux nœuds s'ils sont

visibles dans une même image. Dans une approche similaire, [62] connecte tous les points 3D dans un $map - graph$ G (V, E) où V indique l'ensemble des nœuds et E l'ensemble des arêtes. La connexion entre les points 3D dépend de la visibilité simultanée sur une même image. [122] supprime les correspondances incorrectes en

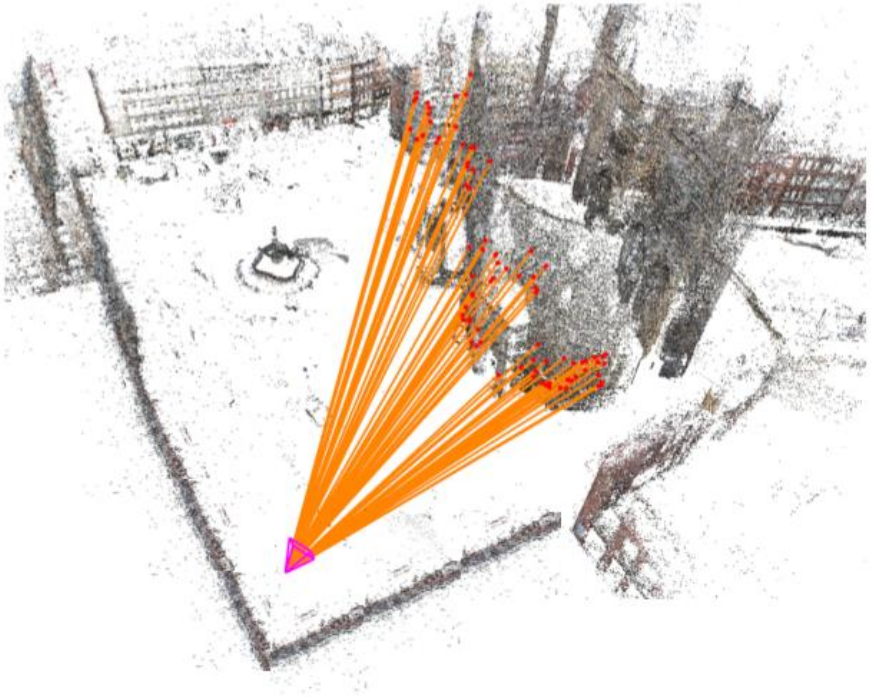


FIGURE 2.14 – Illustration de la méthode directe. Représentations des correspondances 2D-3D entre l'image requête et la carte 3D [91].

se basant sur le graphique de visibilité.

2.2.3.2 Méthode directe par les arbres de régression

Les algorithmes d'apprentissage (arbre de régression) sont aussi utilisés pour estimer la pose. Dans les premiers travaux, [97] code grâce aux données RGB-D, la position globale de chaque pixel associé à un environnement connu dans un arbre de régression. Au moment de l'exécution, un ensemble de pixels est traité dans l'arbre de régression. L'hypothèse de pose multiple obtenue pour chaque pixel est ensuite optimisée pour faire régresser la position et l'orientation de la caméra. Cette méthode est rapide et précise. Cependant, les informations de profondeur associées à chaque pixel sont nécessaires et elles doivent former un arbre spécifique pour chaque scène 3D. Cette méthode a été améliorée dans [42], où les auteurs prennent en considération plusieurs candidats pour la régression de pose finale obtenue par des

prédicteurs entraînés. Valentin et al [114] introduisent des mixtures de gaussienne pour améliorer de façon significative l'estimation des 6 dégrés de liberté en intégrant cette information dans l'étape de régression. L'arbre de régression a été remplacé par le réseau de neurones (NN) dans [68], ce qui permet d'obtenir des résultats légèrement supérieurs en termes de calcul. Meng et al. [70] ne prennent en compte que les images RGB comme requête sans prendre en considération l'information de profondeur. D. Glocker et al. [40] présentent un système basé sur la régression pour associer à une image RGB-D un descripteur binaire puis associer directement la signature de l'image à la pose 3D dans la scène. Cavallari et al. [23] proposent une nouvelle méthode basée sur un arbre de régression préentraîné. Cette méthode permet de retrouver la pose d'une caméra RGB-D sans connaissance préalable de la scène 3D et est plus précise que [40].

2.2.3.3 Méthode directe par CNN

Introduit en 2015 par Kendall et al. [50], PoseNet consiste à entraîner un CNN pour la tâche de localisation. Le réseau est formé sur un ensemble d'images / pose appariées et régresse automatiquement la pose aux 6 dégrés de liberté d'une caméra ayant acquis une image couleur. La pose obtenue par cette méthode n'est pas aussi précise que celle obtenue avec les méthodes directes «classiques» [37], [94] mais offre une grande tolérance aux changements d'échelle et d'apparence. Par rapport à l'arbre de régression, les CNN semblent plus appropriés pour gérer un grand environnement et ne reposent pas sur des informations de profondeur. Liu et al. [63] intègrent l'architecture CNN avec des informations de profondeur pour trouver la pose d'une caméra (RGB-D) dans une totale obscurité. Les travaux de Walch et al.[43] présentent une combinaison des deux approches PoseNet et LSTM branchées à la sortie du réseau afin de coder des informations spatiales plus précises à partir de l'image. Cette combinaison améliore la précision du système. De manière différente, des travaux récents de Weyand et al. [116] considèrent le problème de localisation comme une tâche de classification. Selon une image donnée, un CNN, nommé PlaNET, estime une carte d'emplacement probable pour la requête. Le réseau de classification nommé PointNet [84] prend en entrée n points, applique les transformations sur les données, puis agrège les points par la couche Max Pool. Le réseau de segmentation est une extension du réseau de classification.

2.2.4 Le problème des changements visuels dans l'environnement

Un algorithme de localisation idéal devrait être capable de gérer les changements visuels légers induits par différentes sources : cycles jour/nuit et saisonnier (hiver/printemps/automne/été), différences de point de vue ou modifications légères de la géométrie de la scène. En général ce problème est traité sous le nom *reconnaissance de lieu*. Une caractéristique majeure qui sépare la reconnaissance de lieu

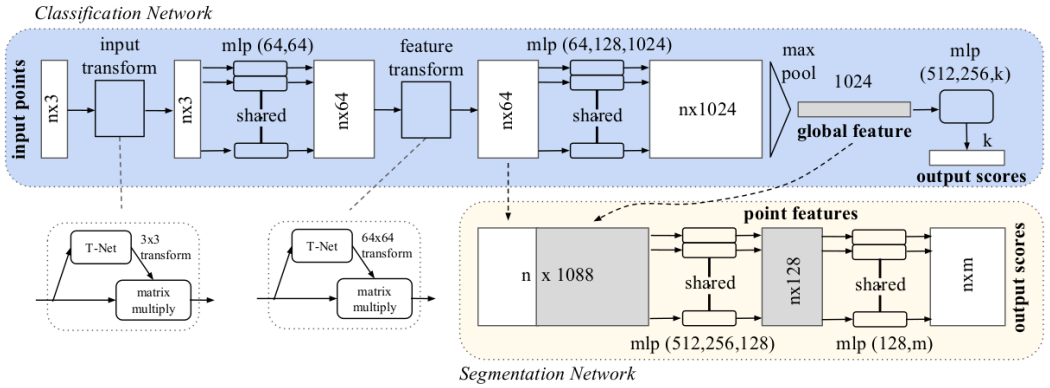


FIGURE 2.15 – Architecture de PointNet[84]

des autres tâches de reconnaissance visuelle est que la reconnaissance de lieu doit résoudre le problème de la reconnaissance à un degré de robustesse que beaucoup d'autres domaines ne possèdent pas. La question est comment peut-on identifier de manière robuste le même lieu du monde réel qui subit des changements d'aspect, par exemple une variation de l'éclairage (Figure 2.16), un changement de saison (Figure 2.17) ou des conditions météorologiques. La tâche de reconnaissance de lieu est traditionnellement considérée comme une tâche de récupération d'image. À son début, la reconnaissance des lieux était dominée par des extracteurs locaux-invariants tels que SIFT et SURF et des descripteurs d'image globaux tels que GIST, LBP et les sacs des mots (2003). Ces techniques traditionnelles d'extraction des caractéristiques ont permis de faire un grand pas dans la qualité des résultats obtenus. En revanche, les dernières années ont vu une avancée encore plus importante dans la reconnaissance du contenu visuel grâce a la puissance de réseaux de neurones convolutifs (CNN). Sünderhauf et al. [105] évaluent et comparent de façon exhaustive l'utilité et les propriétés invariantes de CNNs. Ils ont montré que les caractéristiques extraites des couches intermédiaires de CNN ont une bonne robustesse contre les changements conditionnels, y compris les changements d'éclairage et les changements saisonniers et météorologiques. Arandjelović et al. [15] (2016) conçoivent une nouvelle architecture CNN basée sur VGG16 et la représentation VLAD (Vector of Locally Aggregated Descriptors) ; ils suppriment toutes les couches entièrement connectées et y insèrent les couches VLAD en les rendant différentielles.

2.2.5 Correspondance entre les caractéristiques

La recherche des correspondances entre deux images est un problème clé dans la localisation visuelle. Pour y répondre, deux stratégies de recherche 'Features to Point' (F2P) et 'Point to Features' (P2F) ont été élaborées. Les descripteurs dans l'image requête sont nommés *features* et *points* dans le modèle 3D.

FIGURE 2.16 – Des images du même endroit avec différentes conditions d'éclairage (jour, coucher de soleil, nuit) [111].

2.2.5.1 Stratégie de correspondance : F2P

Dans la stratégie F2P, les *features* requête sont recherchés dans le modèle pour établir des correspondances directes entre les caractéristiques 2D de l'image requête et les points 3D. Donc l'algorithme commence par la détection et l'extraction des points d'intérêt depuis l'image requête et fait appel aux points dans le modèle 3D. En appliquant l'algorithme du plus proche voisin, on trouve les deux points 3D p, q ($p \neq q$) dont les descripteurs d_p, d_q sont les plus proches voisins de d_f. Une correspondance entre les descripteurs de f et de p est établie si le test suivant est vérifié $||d_{f1} - d_p||_2 / ||d_{f1} - d_q||_2 < \epsilon$. L'auteur dans [16] présente un système qui récupère la pose d'une caméra d'un téléphone portable en confrontant une image à un sous-ensemble de points 3D qui devraient être visibles dans la requête selon une information de pose préalable. Irschara et al [46] introduisent la première méthode F2P basée sur la représentation de l'environnement SfM. Les auteurs effectuent des améliorations en enregistrant le nuage de points dans des documents visuels synthétiques couvrant l'ensemble du modèle. F2P est une approche standard, mais

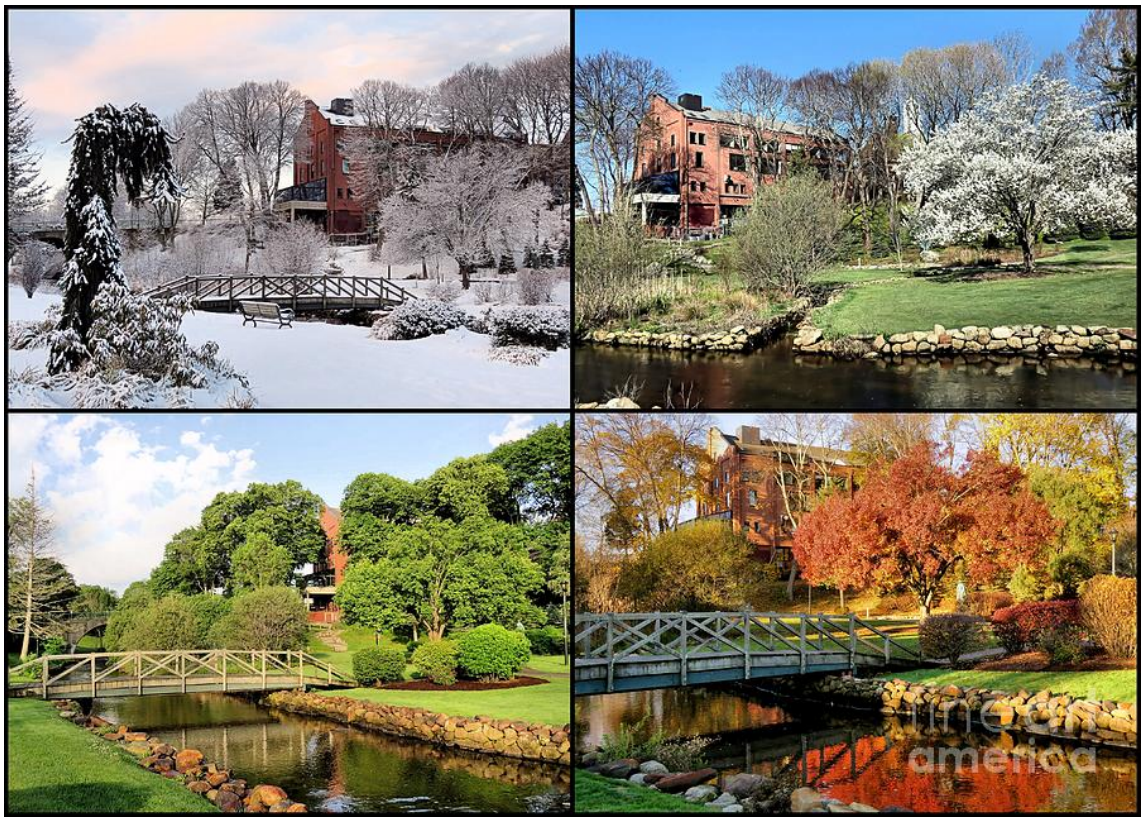


FIGURE 2.17 – Des images du même endroit au cours des quatre saisons (hiver, automne, été, printemps) [8].

inefficace lorsque le nombre de features de la requête est faible.

2.2.5.2 Stratégie de correspondance : P2F

Li et al. [60] inversent le processus en recherchant les correspondances à partir du nuage de points vers l'image (P2F), au lieu de faire correspondre les caractéristiques de l'image aux points. Cette formulation est coûteuse en temps du calcul. Li et al. [59] montrent que les méthodes introduites dans [59] [46] peuvent traiter des environnements de grande taille en augmentant la correspondance P2F avec l'hypothèse de la co-occurrence des points 3D présents dans un voisinage proche. De même que l'algorithme F2P, en appliquant l'algorithme du plus proche voisin on trouve les deux *features* f_1, f_2 ($f_1 \neq f_2$) dont les descripteurs d_{f1}, d_{f2} sont les plus proches voisins de d_p. Une correspondance entre les descripteurs de f_1 et du point p est établie s'il valide le test $||d_p - d_{f1}||_2 / ||d_p - d_{f2}||_2 < \epsilon$. La sortie dans ce cas est une liste plus longue m car le nombre de points 3D est généralement supérieur au nombre de *features* dans l'image requête. Par conséquent, le calcul de la pose prend plus de temps. Cependant, cela peut améliorer le nombre de poses calculées correctement.

En comparant les deux stratégies, 'Point to Features' est plus puissante car elle est capable de comparer les caractéristiques du modèle global avec les caractéristiques de l'image requête.

2.2.6 La segmentation sémantique

2.2.6.1 Définition

La segmentation sémantique est un problème classique de vision par ordinateur. C'est un algorithme de Deep Learning qui associe une étiquette (label) ou une catégorie à chaque pixel d'une image. Elle permet de reconnaître un ensemble de pixels qui forment des catégories distinctes. Par exemple, un véhicule autonome doit identifier les véhicules, les piétons, les panneaux de signalisation, les trottoirs et autres éléments de l'environnement routier. La sortie est généralement une image de haute résolution qui conserve généralement la même taille que l'image en entrée.

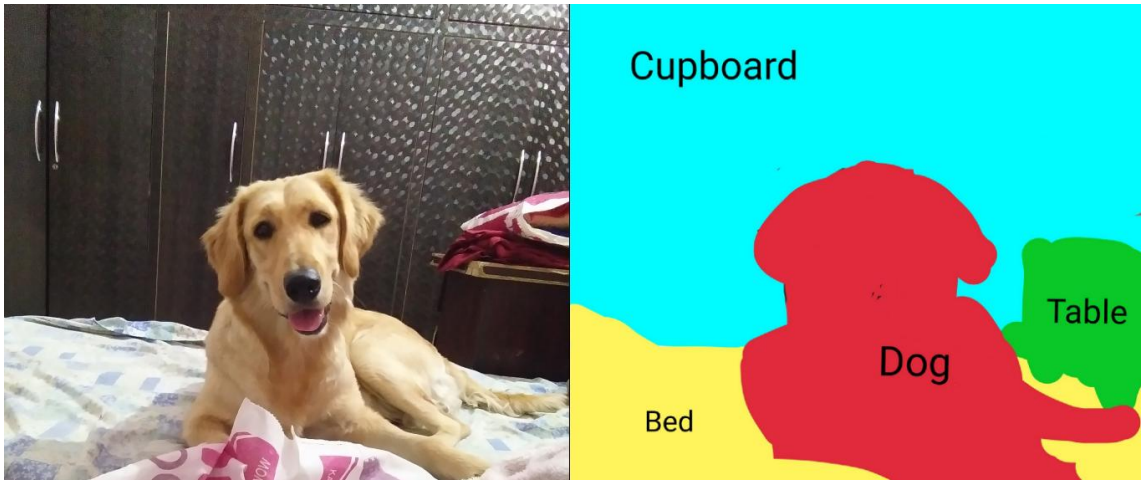


FIGURE 2.18 – Classification des pixels dans une image [7].

Dans le contexte de localisation visuelle, la segmentation sémantique est un moyen efficace pour éliminer les mauvais appariements. L'idée est de comparer les points d'intérêt correspondant à une classe commune bien définie avec le CNN. Par exemple dans la figure 2.19 les points d'intérêt correspondant à la classe *rue* dans l'image I_1 sont appariés avec les points clés de la même classe *rue* dans l'autre image I_2. [57] présente une méthode qui entraîne un réseau neuronal convolutionnel basée sur la segmentation sémantique par des correspondances 2D-2D entre des images prises dans différentes conditions. L'application rend l'algorithme de segmentation final robuste aux changements saisonniers.

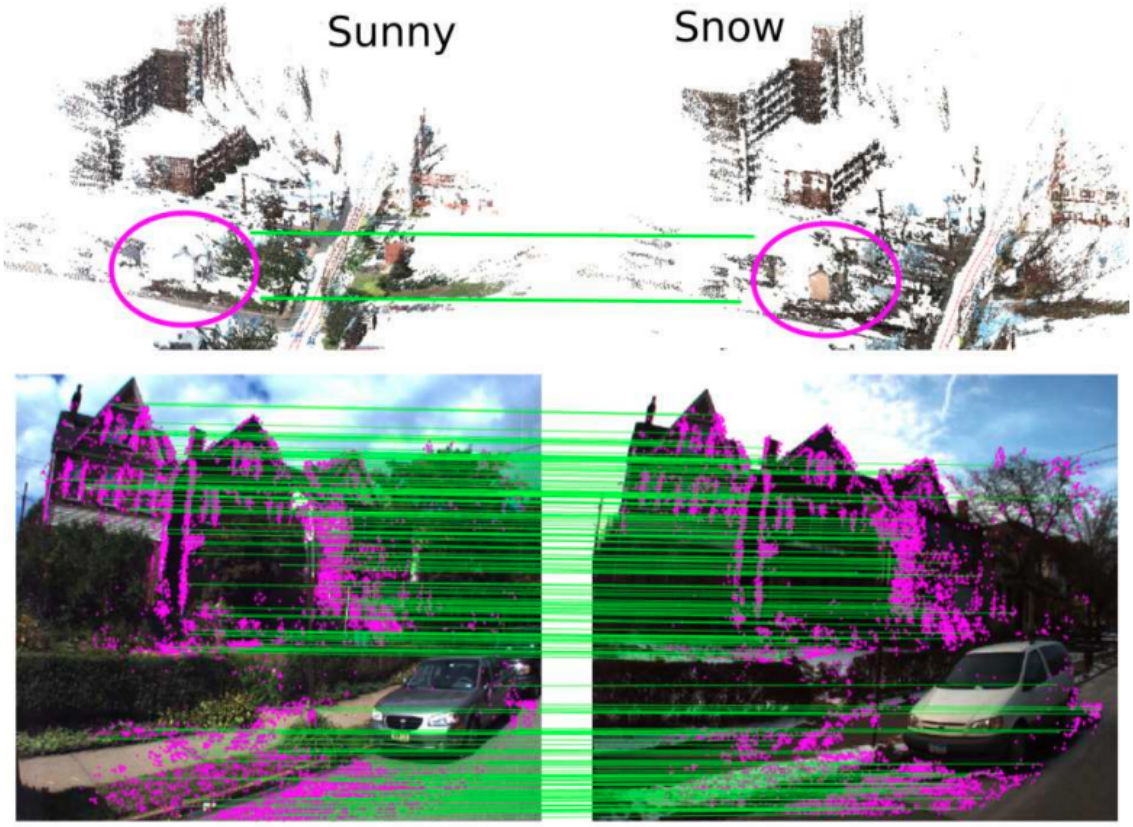


FIGURE 2.19 – Correspondances 2D-2D entre des images prises dans des conditions différentes [9].

2.2.6.2 Méthodes fondées sur un apprentissage profond

La plupart des méthodes de segmentation modernes utilisent des techniques d'apprentissage profond, bien que certaines d'entre elles utilisent encore des aspects des méthodes de segmentation classiques. Les réseaux de neurones convolutionnels (CNN) sont basés sur une architecture composée par plusieurs couches convolutionnelles pour effectuer la segmentation sémantique. Il s'agit de coder l'image d'entrée en faibles dimensions, puis de la récupérer à la sortie du réseau.

L'architecture d'un CNN

Un réseau CNN est un type populaire de réseau de neurones artificiel qui est principalement utilisé pour analyser des images. Ce qui différencie un réseau CNN des autres réseaux de neurones artificiels, ce sont les couches cachées appelée «couches convolutives». Avec les couches convolutives, un réseau CNN est capable de détecter des motifs (par exemple les bords, les formes ou même les textures d'un objet) dans les images. Ces modèles sont déterminés en utilisant des filtres dans l'opération de convolution. Les réseaux de neurones convolutionnels sont composés des couches suivantes : convolutionnelle, *pooling* et *Fully connected*.

Exemples des architecture pour la segmentation sémantique

Le réseau FCN [55] est l'un des premiers modèles proposés pour la segmentation sémantique. Ce réseau est inspiré des modèles standards tels que VGG et AlexNet. Des modifications majeures ont été effectuées sur ce réseau pour qu'il devienne compatible avec la tâche de segmentation.

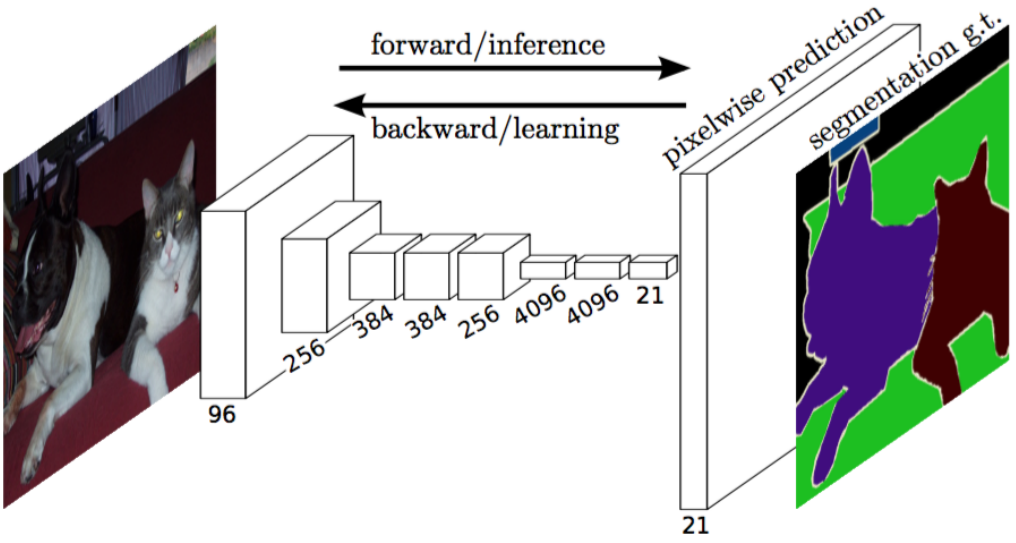


FIGURE 2.20 – Les différentes couches de l'architecture FCN et leurs tailles [55].

Le réseau UNet [88] a été développé par Olaf Ronneberger et al. pour la segmentation des images bio-médicales. Il s'agit d'un réseau entièrement convolutif (FCN), c'est-à-dire qu'il ne contient que des couches convolutives et ne contient aucune couche dense grâce à laquelle il peut accepter des images de toutes tailles. Pour les images dans le domaine médical, le réseau UNet est le meilleur choix. Le réseau UNet peut également être utile pour les scènes intérieures / extérieures avec des images de petite taille.

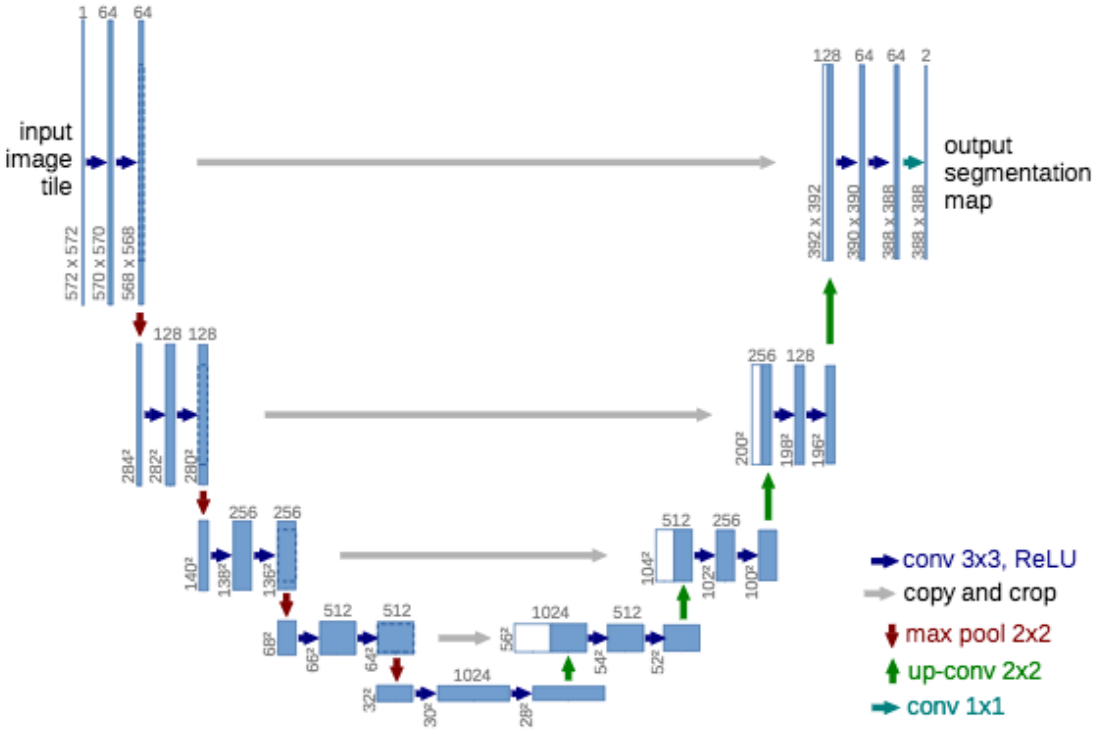


FIGURE 2.21 – Les différentes couches de l'architecture U-Net [88].

Le réseau SegNet [18] est une architecture (codeur-décodeur) profond pour la segmentation sémantique. Il est développé par des membres du groupe "vision par ordinateur et robotique" de l'Université de Cambridge, au Royaume-Uni. L'architecture se compose d'une séquence de couches de traitement non linéaire (encodeurs) et d'un ensemble correspondant de décodeurs suivi d'un classificateur pixel par pixel. En règle générale, chaque codeur se compose d'une ou plusieurs couches convolutives (voir la figure 2.22). Un ingrédient clé de SegNet est l'utilisation d'indices max-pooling dans les décodeurs pour effectuer un sur-échantillonnage (upsampling) des cartes de caractéristiques basse résolution. Ceci présente les avantages importants de conserver les détails dans les hautes fréquences dans les images segmentées et également de réduire le nombre total de paramètres pouvant être entraînés dans les décodeurs.

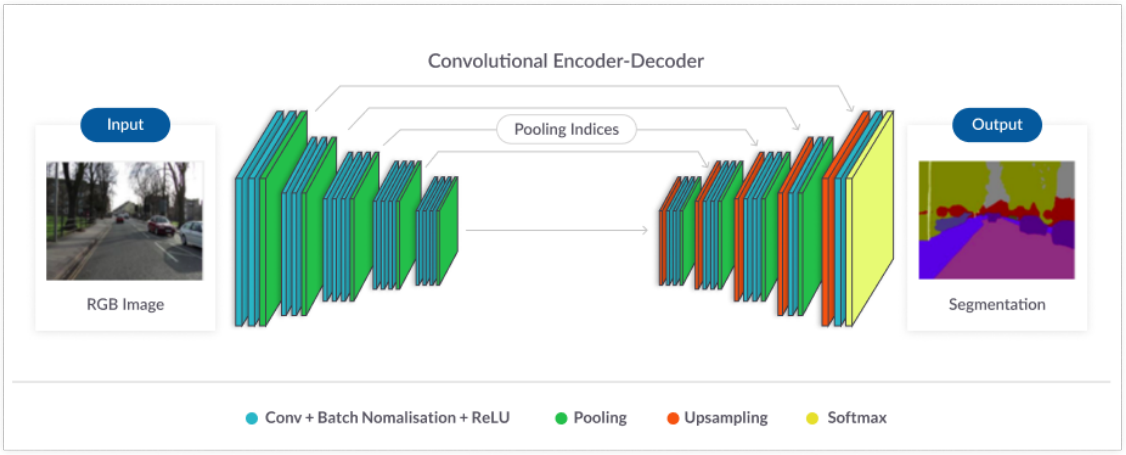


FIGURE 2.22 – Les différentes couches de l'architecture SegNet [18].

Le réseau PSPNet [125] : le réseau d'analyse de scène Pyramid est optimisé pour apprendre une meilleure représentation du contexte global d'une scène. Tout d'abord, l'image est transmise au réseau de base pour obtenir une carte des caractéristiques (Feature Map). La carte des caractéristiques est sous-échantillonnée (downsampled) à différentes échelles. La convolution est appliquée aux cartes de caractéristiques regroupées. Après cela, toutes les cartes de caractéristiques sont suréchantillonnées (upsampled) à une échelle commune et concaténées ensemble. Enfin, une autre couche de convolution est utilisée pour produire les sorties de segmentation finales. Ici, les petits objets sont bien capturés par les entités regroupées à une haute résolution, tandis que les gros objets sont capturés par les entités regroupées à une taille plus petite.

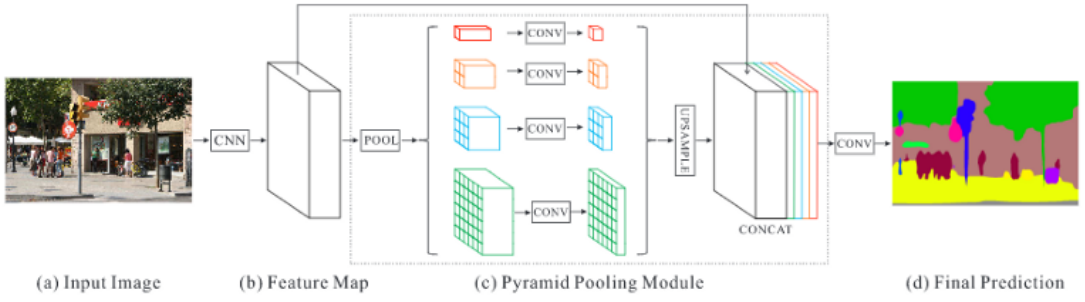


FIGURE 2.23 – Les différentes couches de l'architecture PSPNet [125].

Pour les images contenant des scènes intérieures et extérieures, PSPNet est préférable, car les objets sont souvent présents avec des tailles différentes. Ici, la taille d'entrée doit être assez grande, environ 500x500.

2.2.6.3 Applications de la segmentation sémantique

Les applications incluent entre autres la reconnaissance faciale, l'identification des plaques d'immatriculation et l'analyse d'images issues de satellites. Des applications commerciales comme la vente au détail et la mode utilisent la segmentation d'images, par exemple, dans les recherches basées sur l'image. Les véhicules autonomes l'utilisent pour comprendre leur environnement.

Segmentation faciale

La segmentation sémantique peut aider les systèmes de vision par ordinateur à réaliser plusieurs tâches. Elle permet de déterminer les relations entre les objets, ainsi que le contexte des objets dans une image. La segmentation sémantique permet de trouver dans un visage les différents éléments : la bouche, le menton, le nez, les yeux et les cheveux.

FIGURE 2.24 – Exemple d'une prédiction en utilisant l'apprentissage profond pour détecter les différents éléments d'un visage [51].

Conduite autonome

La conduite autonome est une tâche extrêmement complexe qui nécessite une perception et une analyse en temps réel de l'environnement du véhicule. La segmentation sémantique est utilisée pour identifier des objets comme d'autres voitures, les panneaux de signalisation, les régions et les trottoirs. La segmentation des instances est utilisée dans la conduite autonome, car les véhicules, les piétons, les panneaux, etc., doivent être détectés et leurs évolutions estimées. Par conséquence, les voitures autonomes devraient être capables de percevoir et de comprendre leur environnement afin de conduire en toute sécurité. La segmentation sémantique permet aux voitures autonomes de déterminer quelles zones de l'environnent sont sans danger.

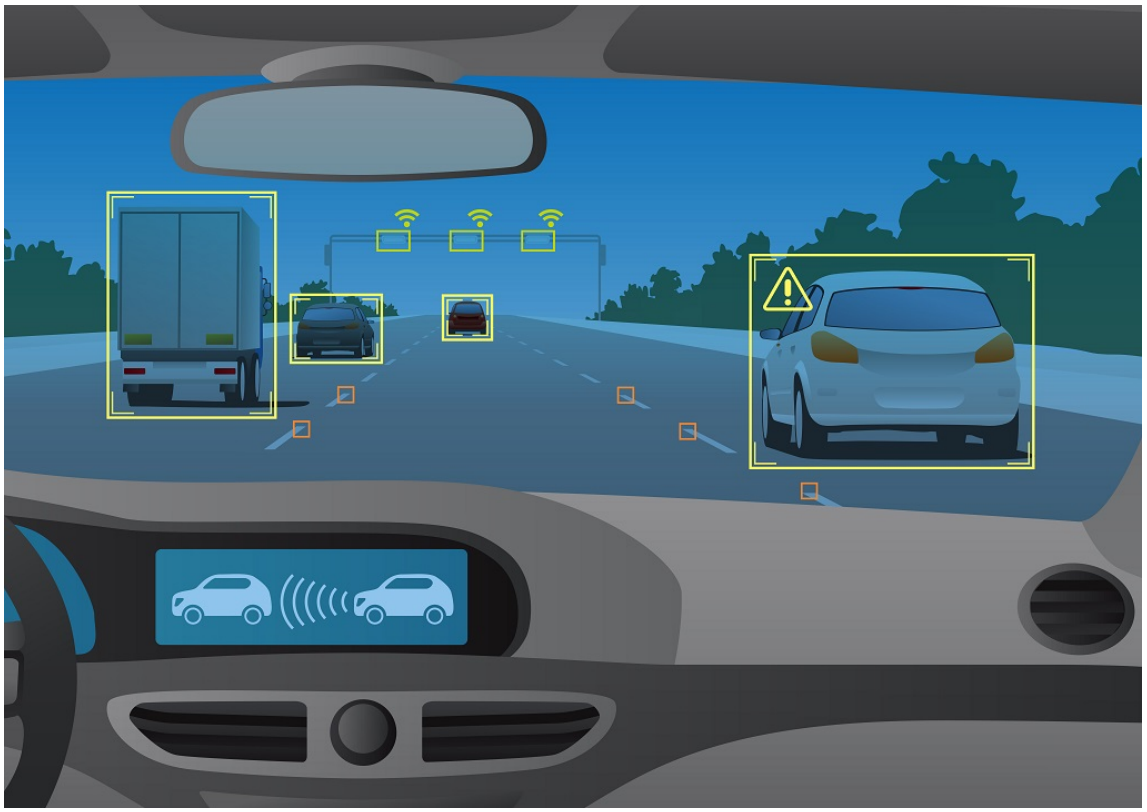


FIGURE 2.25 – Illustration des systèmes de perception d'un véhicule autonome à l'aide de l'apprentissage profond [2].

Agriculture de précision

La segmentation sémantique est souvent utilisée pour analyser les images des champs agricoles et déterminer les zones à traiter. En reconnaissant les zones de champs où les plantes sont endommagées par des parasites ou des maladies, les interventions peuvent se limiter aux zones correspondantes. Une autre utilisation de la segmentation sémantique en agriculture est la détection des mauvaises herbes dans les champs, cependant, cela se fait avec des images prises beaucoup plus près.

FIGURE 2.26 – Exemple d'utilisation de l'apprentissage profond pour l'agriculture de précision [11].

2.2.7 Bases de données pour la localisation

Nous présentons dans cette section les bases des données utilisées dans la localisation basée sur l'image pour évaluer les performances des méthodes directes et indirectes.

2.2.7.1 Dubrovnik

FIGURE 2.27 – Le modèle 3D pour la base de données Dubrovnik [60].

Dubrovnik 6K [60] est une base de données composée de 6,044 images clé et de 800 images de test disponibles sur Internet. Elles proviennent de la vieille ville

de Dubrovnik en Croatie, classée au patrimoine mondial de l'UNESCO. Les images sont principalement capturées par les touristes avec une grande variété de types d'appareils photo. Les poses de vérité terrain pour cet ensemble de données ont été calculées en utilisant l'approche SfM. L'algorithme génère 1,8 million de points 3D et 9,6 millions de descripteurs.

FIGURE 2.28 – Exemples d'images de la base de données Dubrovnik [60].

2.2.7.2 Rome

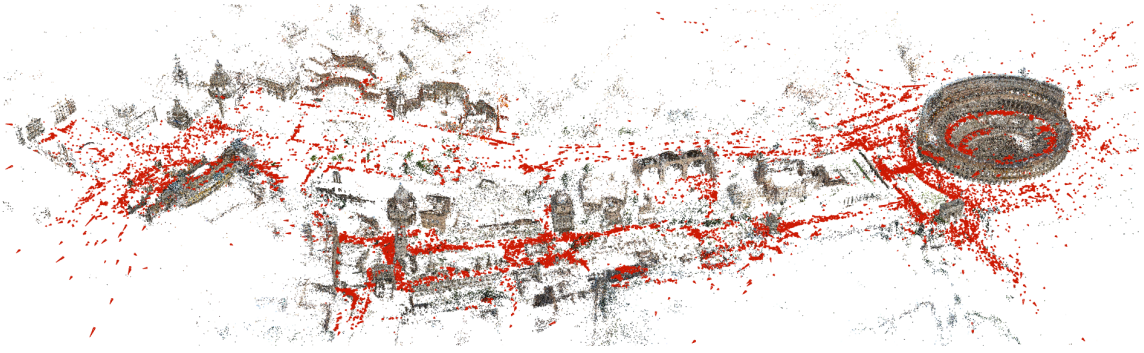


FIGURE 2.29 – Le modèle 3D pour la base de données Rome [60].

Rome [60] est une large base de données de 15,179 images clé et 1,000 images de test construite par l'algorithme [100]. L'un des détecteurs de caractéristiques les plus largement utilisés est le scale-invariant feature transform (SIFT). Il est utilisé lors de la construction du nuage de points de cette base. Chaque point 3D du modèle est donc associé à une liste de descripteurs SIFT. Cette base contient 4,3 millions de points 3D et 21,5 millions de descripteurs.

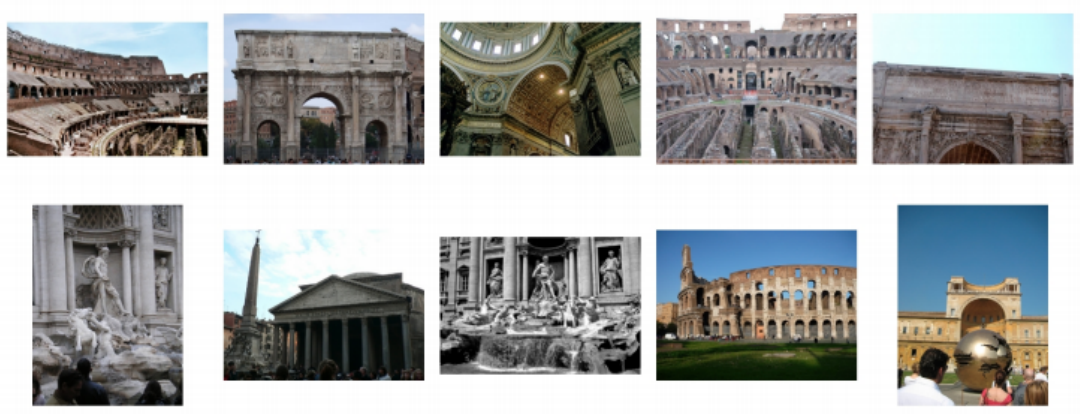


FIGURE 2.30 – Exemples d'images de la base de données Rome [60].

2.3 Conclusion

Nous avons commencé dans ce chapitre par introduire la technique de la construction 3D et les algorithmes de la localisation et de cartographie (SLAM, SfM). Nous avons également montré que parmi les méthodes, le SLAM basé image clé semble le plus adapté à notre objectif à savoir une localisation temps réel. La deuxième partie est consacrée à définir la technologie SfM et son architecture.

La deuxième partie de ce chapitre est une étude bibliographique qui consiste à revoir les différentes méthodes de localisation : méthode indirecte qui transforme la tâche de localisation en un problème de récupération de l'image requête et fournit des informations sur l'emplacement de la requête puis la méthode directe qui tente de trouver des correspondances en comparant directement les caractéristiques d'image aux points 3D de la scène. Aussi, nous avons expliqué brièvement les deux stratégies de correspondance (F2P, P2F) ainsi que les avantages et les inconvénients de chacune d'entre elles.

Dans la troisième partie du chapitre, nous avons abordé l'approche principale pour effectuer une segmentation sémantique. Les méthodes d'apprentissage profond utilisent les algorithmes basés sur les réseaux de neurones convolutionnels. Dans notre cas, les CNN convolutionnels sont les plus adaptés à notre objectif car ils ont montré leur performance dans le contexte de la localisation. Ce chapitre pré-

sente aussi une étude bibliographique sur l'architecture du CNN ainsi que quelques exemples basés sur le principe codeur-décodeur.

Chapitre 3

Descripteur géométrique-visuel pour une localisation améliorée basée sur l'image utilisant un a priori sur la verticale

3.1 Introduction

Dans ce chapitre, nous avons choisi le SLAM visuel pour résoudre le problème de l'estimation de la pose d'un casque de réalité augmentée intégrant une IMU et équipé par une paire de caméras. Le système développé pendant cette thèse est dédié à une application de réalité augmentée pour un musée dont l'objectif est de plonger le visiteur dans des scènes de la vie quotidienne des solutréens. Nous cherchons donc un approche de construction 3D en temps réel. Le SLAM visuel est l'algorithme le plus adapté aux applications temps réel afin d'estimer les poses des caméras et de réaliser une carte 3D représentant l'environnement. Deux problèmes fondamentaux dans le contexte de la localisation sont traités dans ce chapitre : (1) la correspondance entre les descripteurs ; (2) la recherche d'image par similarité. La recherche des correspondances est une étape clé dans la localisation. Dans les approches SLAM, les images peuvent être comparées en mettant en correspondance les descripteurs qui les représentent. Cette comparaison consiste à apparier des descripteurs extraits d'une image requête et des descripteurs candidats extraits à partir des images clé. Pour chaque descripteur requête, on calcule la similarité avec tous les descripteurs dans les images clés. Un critère est alors utilisé pour décider quels candidats sont appariés avec un descripteur requête. La méthode la plus utilisée en pratique, due à D. Lowe [64], consiste à comparer le rapport des distances entre les deux plus proches voisins avec un seuil égal à 0.7. Ce critère donne souvent de bons résultats, mais il souffre de plusieurs limitations dues aux changements visuels dans l'image ce qui entraîne l'impossibilité d'estimer la pose.

Pour résoudre ce problème, nous avons essayé d'améliorer l'étape de correspondance en ajoutant aux descripteurs visuels extraits par le descripteur KAZE, des in-

formations géométriques. Pour ce faire, nous imposons deux hypothèses sur l'image requête et la carte 3D obtenue en utilisant le SLAM visuel. Nous supposons que la direction verticale est connue et que la hauteur des caméras est constante. Notre hypothèse est valable lorsque les caméras sont montées sur un véhicule ou un robot mobile (figure 3.1). La carte 3D du musée et les images requêtes sont prises par une caméra stéréo *tara*. L'avantage de l'utilisation de la stéréo ici est de trianguler les points de sorte que chaque point de l'image requête soit directement associé aux coordonnées 3D. Dès lors, chaque point clé est caractérisé par deux éléments : ses descripteurs visuels locaux (KAZE [14]) et sa hauteur Z. Enfin, nous les concaténons pour former un nouveau descripteur combinant des informations géométriques et visuelles. Deux usages sont possibles pour le nouveau descripteur dans notre contexte : (1) utilisation dans le processus d'appariement 2D-3D ; (2) où intégration descripteurs géométriques dans le processus de construction des mots visuels.

FIGURE 3.1 – Exemples des systèmes exploitant une tête stéréo.

3.2 Méthode proposée

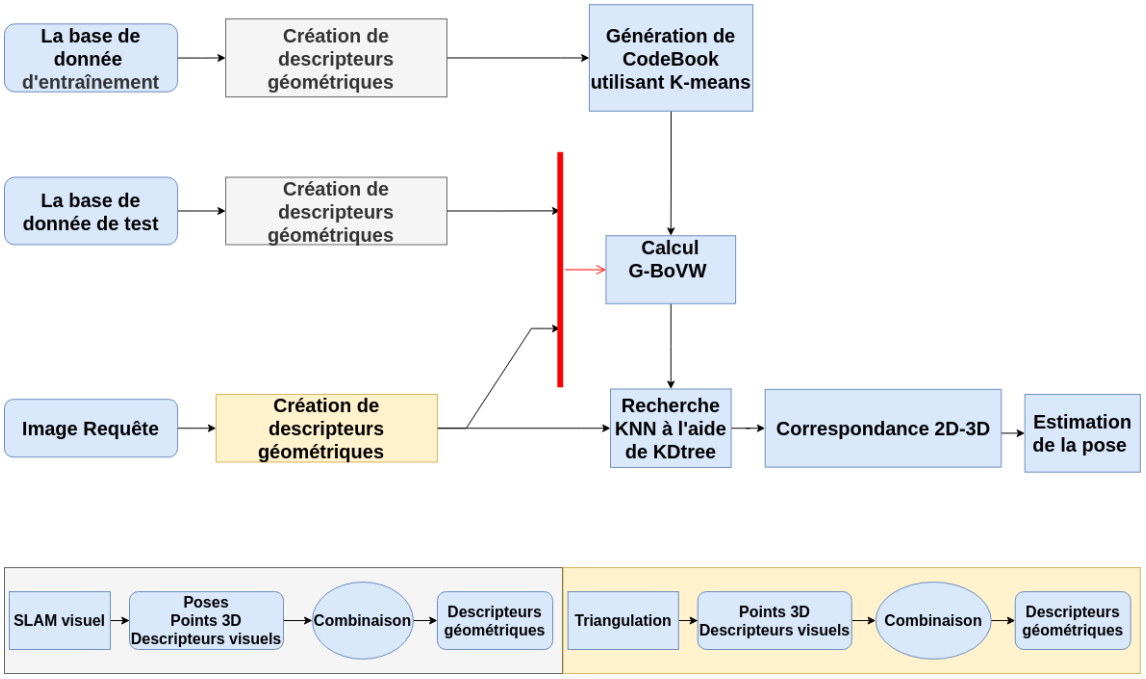


FIGURE 3.2 – Les différentes étapes de la méthode proposée.

Dans cette section, nous présentons une méthode indirecte (3.2) pour calculer la pose à partir d'une paire d'images. L'estimation de la pose est une étape clé et délicate dans le SLAM. Cependant, il est possible d'exploiter des informations géométriques pour gérer les mauvaises mises en correspondance afin de récupérer une pose précise. Nous robustifions l'étape de mise en correspondance en combinant la hauteur et les descripteurs visuels. La figure 3.3 présente le système de coordonnées (X_v, Y_v, Z_v) utilisé dans un véhicule autonome. Dans ce cas, les positions sont exprimées en mètres et l'axe Z_v est orienté selon la verticale.

Les étapes de notre approche commencent par la construction de la carte 3D à l'aide du SLAM visuel. Une fois les caractéristiques visuelles des images (images clés et requête) extraites, nous construisons un descripteur visuel géométrique. Ensuite, nous sélectionnons les images clés les plus proches de l'image requête à l'aide de sacs de mots visuels intégrant des informations géométriques(GBoVW). Enfin, en utilisant les correspondances entre les descripteurs géométriques, nous calculons la pose et nous considérons que la pose est correcte si et seulement si le nombre des appariements (inliers) est supérieur à 12.

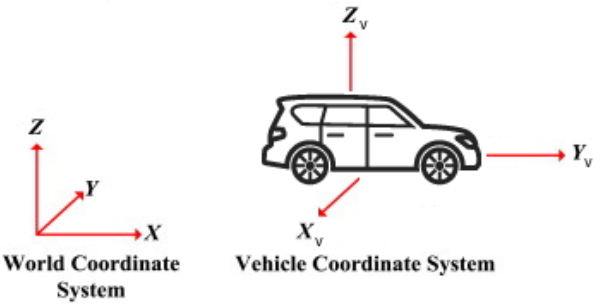


FIGURE 3.3 – Système de coordonnées pour un robot ou un véhicule.

3.2.1 Descripteurs visuels géométriques

La localisation basée sur l'image (IBL) est établie sur les descripteurs locaux extraits à partir des images. Dans ce travail nous avons utilisé le descripteur KAZE [14] pour la construction 3D ainsi que pour l'extraction des caractéristiques visuelles à partir d'une image prise par une paire de caméras. L'avantage d'utiliser une paire stéréo comme requête au lieu d'une image monoculaire est de pouvoir trianguler des points 3D afin d'obtenir des informations comme la hauteur Z. Chaque caractéristique de la paire stéréo est alors caractérisée par deux éléments : un ensemble de descripteurs locaux et un point 3D dans le système de coordonnées de la caméra. Parmi les 3 axes en cohérence avec les hypothèses retenues, seule la hauteur (axe Z) ne dépend pas de la pose de la requête. Ainsi, pour chaque point 3D, nous extrayons la coordonnée Z à partir d'une carte créé par le SLAM visuel. Pour l'image requête, nous avons utilisé la technique de triangulation stéréo pour obtenir la hauteur. Ensuite, nous concaténons le descripteur visuel KAZE (K_p) avec cette information géométrique invariante afin d'obtenir un descripteur combiné [K_p, Z_n]. Z_n est la transformée de la hauteur Z brute au moyen d'une fonction de normalisation. Il est nécessaire de normaliser Z pour deux raisons : premièrement pour minimiser l'impact des valeurs aberrantes et deuxièmement pour équilibrer le poids relatif des descripteurs visuels et de la hauteur.

La triangulation stéréo peut produire des valeurs aberrantes avec une hauteur très élevée ou très basse. Pour résoudre ce problème, nous définissons une grille dans le plan horizontal de la carte du SLAM et nous supprimons les 10 valeurs les plus élevées et les 10 valeurs les plus basses dans chaque cellule de la grille. Cette étape induit une réduction du nombre de points 3D dans la carte globale. Après l'étape d'élimination du bruit, il est nécessaire de rendre le poids de la coordonnée Z égal au poids du descripteur visuel avec l'équation (3.1) qui dépend de deux paramètres principaux :

— ΔH_c : Différence de hauteur de la caméra entre l'étape de cartographie et l'étape de localisation.

— L'amplitude Z dans la carte 3D, c'est la différence entre Z_{max} et Z_{min}, après l'étape d'élimination du bruit.

$$Z_n = \frac{Z + \Delta H_c}{Z_{max} - Z_{min}} DimDesc \tag{3.1}$$

où $DimDesc$ est la taille des descripteurs (64 pour KAZE).

3.2.2 Sac de mot géométrique (G-BoVW)

L'algorithme dit de "sac de mots visuels" (BoVW) est l'une des puissantes techniques de récupération des images similaires à une image requête en utilisant leurs caractéristiques visuelles. Nous montrons dans cette section que l'ajout des informations géométriques dans le processus de construction du dictionnaire améliore la performance de l'algorithme. Dans la première étape, nous appliquons l'algorithme SLAM visuel dans le but d'attribuer aux caractéristiques visuelles leurs hauteurs correspondantes puis nous les extrayons. Après avoir normalisé la hauteur, nous recueillons toutes les caractéristiques géométriques et nous générons un codebook (vocabulaire visuel) en appliquant l'algorithme de clustering K-means.

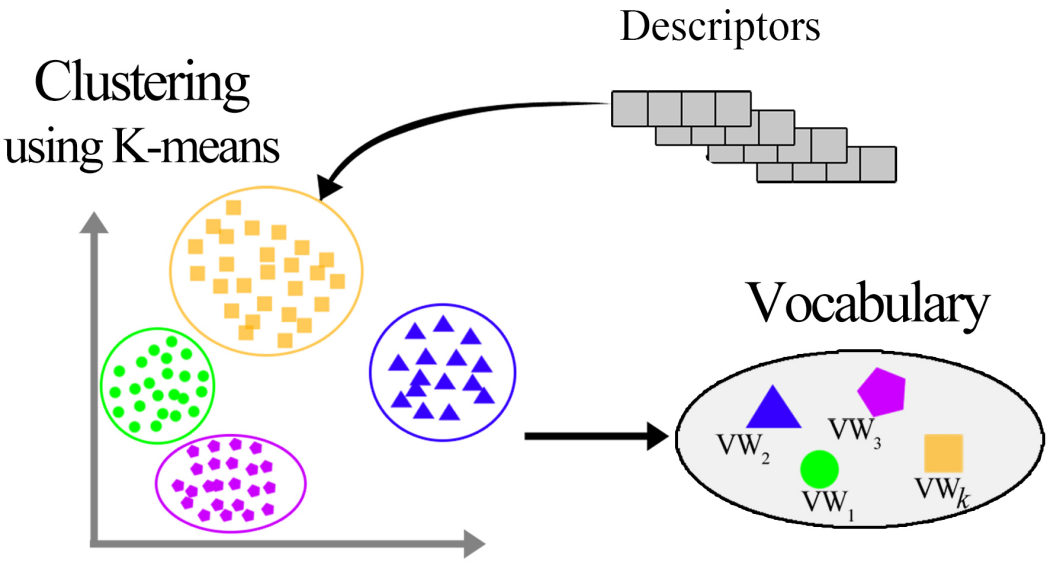


FIGURE 3.4 – Processus de construction du dictionnaire visuel [5].

Les mots visuels sont alors définis comme les centres des groupes (figure 3.4). Les descripteurs utilisés pour générer ces mots sont extraits à partir d'une séquence vidéo enregistrée dans le même environnement que la carte. De manière identique que l'algorithme BoVW, on associe aux descripteurs de l'image requête et des images clé les mots visuels les plus proches en utilisant la métrique L2 puis on crée l'histogramme contenant la fréquence des mots pour chaque image. Enfin, la similitude entre l'image requête et les candidats est mesurée par la distance entre les deux histogrammes.

3.2.3 Limitation

L'ajout des informations géométriques aux descripteurs visuels est une idée simple et facile à implémenter. Aussi, cette méthode a une faible complexité ce qui permet de réduire la charge dans le CPU et renvoie les résultats rapidement. Malgré les bénéfices obtenus par cette combinaison (géométriques + visuels), cette méthode présente une limitation majeure dans le cas où la carte 3D est construite dans un environnement extérieur. Dans ce cas la hauteur Z est non bornée car il n'est pas limité spatialement. Au contraire, l'effet de Z est plus prononcé à l'intérieur car la hauteur à l'intérieur est bornée entre le sol et le plafond comme le montre la figure 3.5.

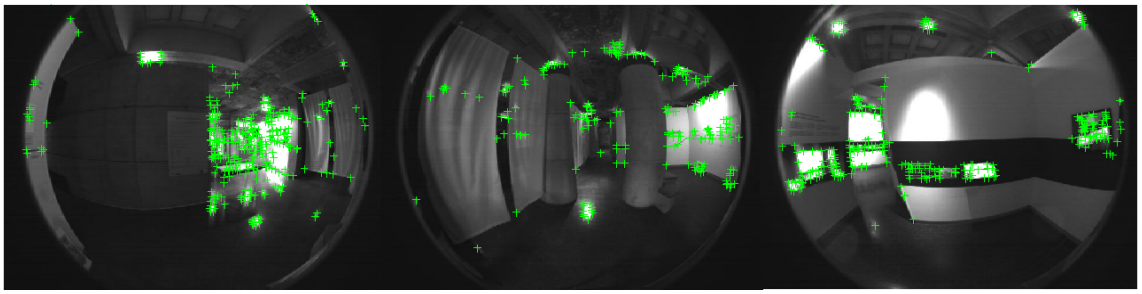


FIGURE 3.5 – Des images de la base de données du musée montrent que Z est bornée entre le sol et le plafond.

Dans la figure 3.6, quelques exemples montrent qu'en extérieur il n'y a pas de plafond pour la hauteur Z. Par conséquent, les mots visuels deviennent moins robustes. Les images clés récupérées sont moins précises et l'estimation de la pose devient délicate.

FIGURE 3.6 – Des images de la base de donnée du oxford montrent que la hauteur Z est non bornée.

3.3 Étude expérimentale

3.3.1 Implémentation

Dans toutes les expériences, nous avons utilisé le descripteur KAZE [14] pour extraire les points clés dans les images requêtes et les images clés. Pour trouver les images clés les plus similaires d'une image requête, il est nécessaire de calculer la distance entre les histogrammes en utilisant une métrique. Il existe plusieurs fonctions de calcul de distance, notamment, la distance euclidienne, la distance de Manhattan, la distance de Minkowski, celle de Jaccard, la distance de Hamming, etc. Dans notre cas nous avons utilisé la distance euclidienne.

La recherche de similarité peut être coûteuse en calculs lorsque les données sont décrites par un descripteur volumineux. La méthode naïve implique le calcul de toutes les distances entre toutes les paires de la base de données : pour N descripteurs de dimension D, la complexité est $O[DN^2]$. Des recherches de voisins par cette méthode naïve peuvent être très compétitives pour une base de données de petite taille. Cependant, lorsque le nombre des descripteurs N augmente, l'approche par force brute devient rapidement irréalisable. Toutefois, il existe de nombreuses autres approches pour calculer la similarité entre deux images d'une manière robuste et

rapide. Il s'agit d'une amélioration significative par rapport à la "force brute" pour les données volumineuses. Ici, nous avons utilisé l'algorithme KDtree (figure 3.7) pour accélérer la recherche et nous arrêtons après avoir trouvé 10 candidats pour chaque requête. Puis, la performance de notre approche est testée à l'aide de deux listes contenant les indices des images similaires (KNN) obtenues par les algorithmes BoVW et GBoVW.

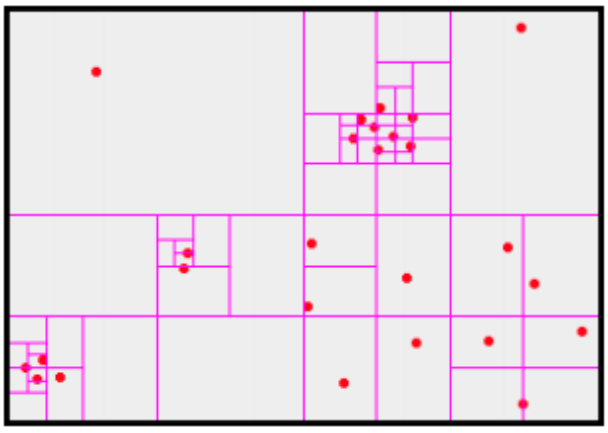


FIGURE 3.7 – Exemple de kd-Tree construit en utilisant des plans médians de l'espace [4].

Afin de récupérer les images les plus proches d'une image requête donnée nous passons à l'étape de la mise en correspondance entre les descripteurs de deux images pour affecter à chaque point clé un point 3D. Nous avons retenu expérimentalement un seuil ($epsilon = 0,7$) dans l'équation (4.1). Nous comparons les descripteurs entre l'image requête et les deux plus proches dans l'image clé et nous validons une correspondance si le test ratio suivant est vérifié :

$$\frac{||[K_p, Z_n] - [K_{p1}, Z_{n1}]||_2}{||[K_p, Z_n] - [K_{p2}, Z_{n2}]||_2} < \epsilon \tag{3.2}$$

òu $[K_p, Z_n]$ est le descripteur d'un point clé de l'image requête, $[K_{p1}, Z_{n1}]$ $[K_{p2}, Z_{n2}]$ sont les descripteurs des deux points 3D les plus proches dans l'image clé analysée.

3.3.2 Jeux de données

La méthode proposée est principalement destinée à la navigation en intérieur mais nous l'évaluons également à l'extérieur. Nous testons donc notre approche sur deux bases de données différentes : navigation en intérieur (musée, Figure 3.8) et en extérieur (Oxford Robotcar dataset [65], Figure 3.15) [1].

1. https ://robotcar-dataset.robots.ox.ac.uk/datasets/2014-08-11-10-22-21/ ?fbclid=IwAR2rgyv58BnS6QLiuIayMKsC64mCc3V4TDk1S8RaZV2$_o$6$pcRvqOlvTHmLc$

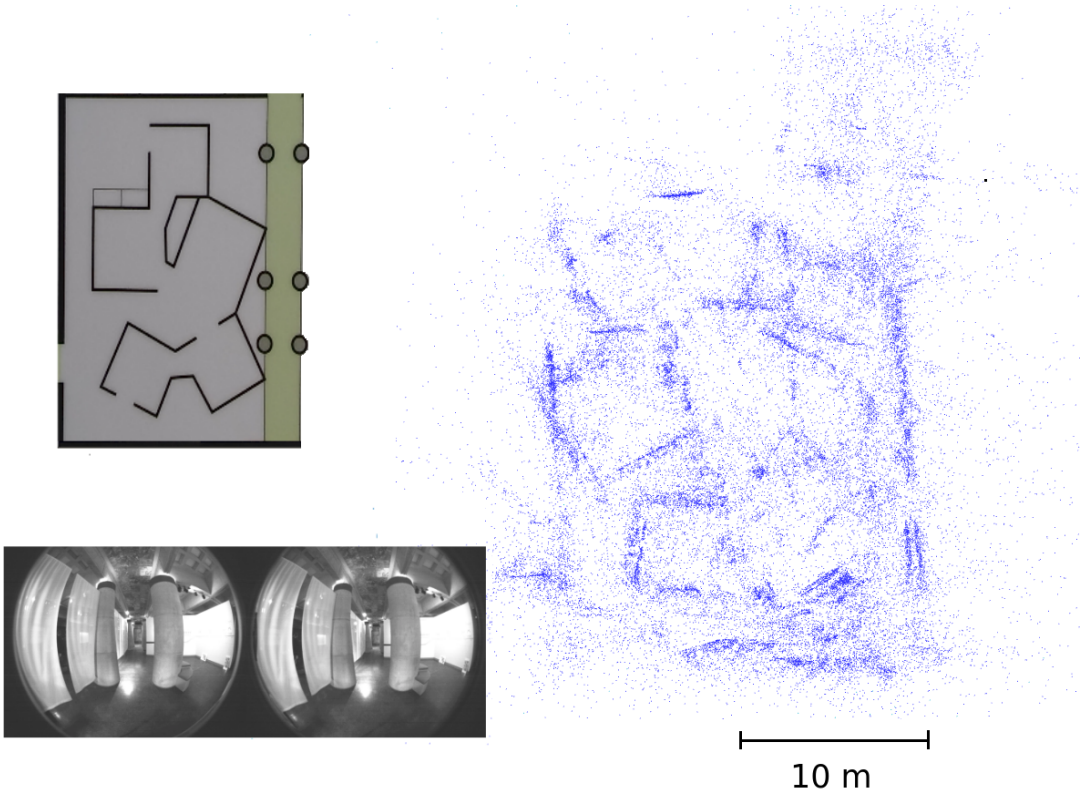


FIGURE 3.8 – Plan et la carte 3D du musée.

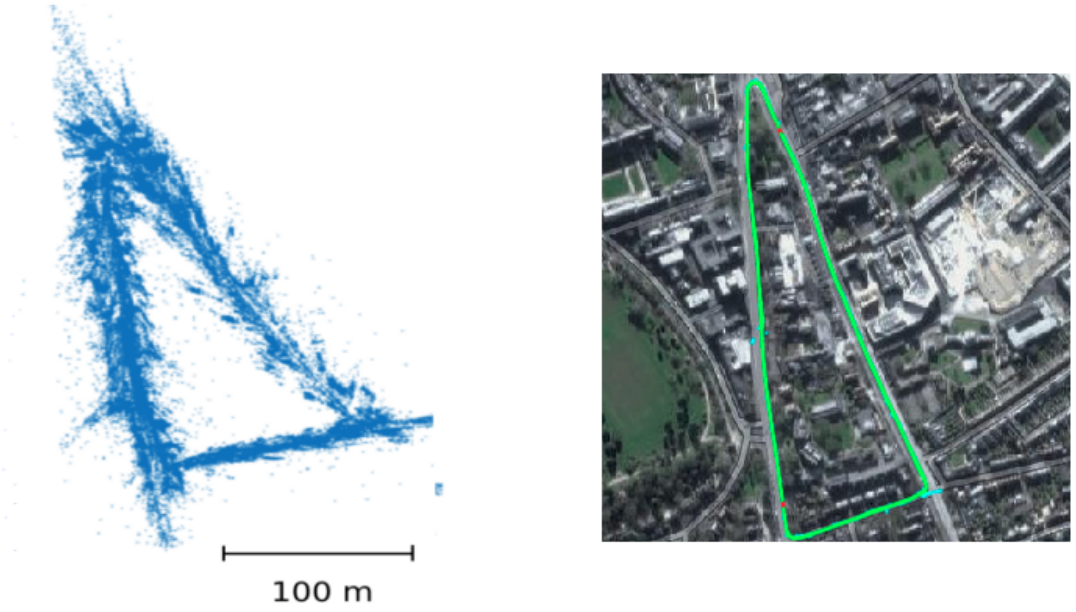


FIGURE 3.9 – Carte 3D d'Oxford Robotcar [65].

3.3.2.1 Musée

La base de données musée est composée de 443 images requêtes et 859 images clés. À l'aide d'un algorithme SLAM, un nuage de points est construit. Cela génère 61 192 points 3D et 1,6 millions de descripteurs (voir la figure 3.8). Les images en intérieur utilisées pour construire la carte 3D ont une dimension de 752*480 pixels issus de la caméra tara (figure 3.10). C'est une caméra 3D stéréo basée sur le capteur stéréo MT9V024 qui prend en charge le format WVGA non compressé à 60 fps sur USB 3.0. Le tableau 3.1 montre les différents formats et résolutions supportés par la caméra Tara.

FIGURE 3.10 – Caméra stéréo 3D (Tara [3]).

Format	Résolution	USB 3.0	USB 2.0
WVGA	(2 * 752) x 480	60 fps	30 fps
VGA	(2*640) x 480	60 fps	30 fps
QVGA	(2*320) x 240	60 fps	30 fps

TABLE 3.1 – Caractéristiques de la caméra Tara.

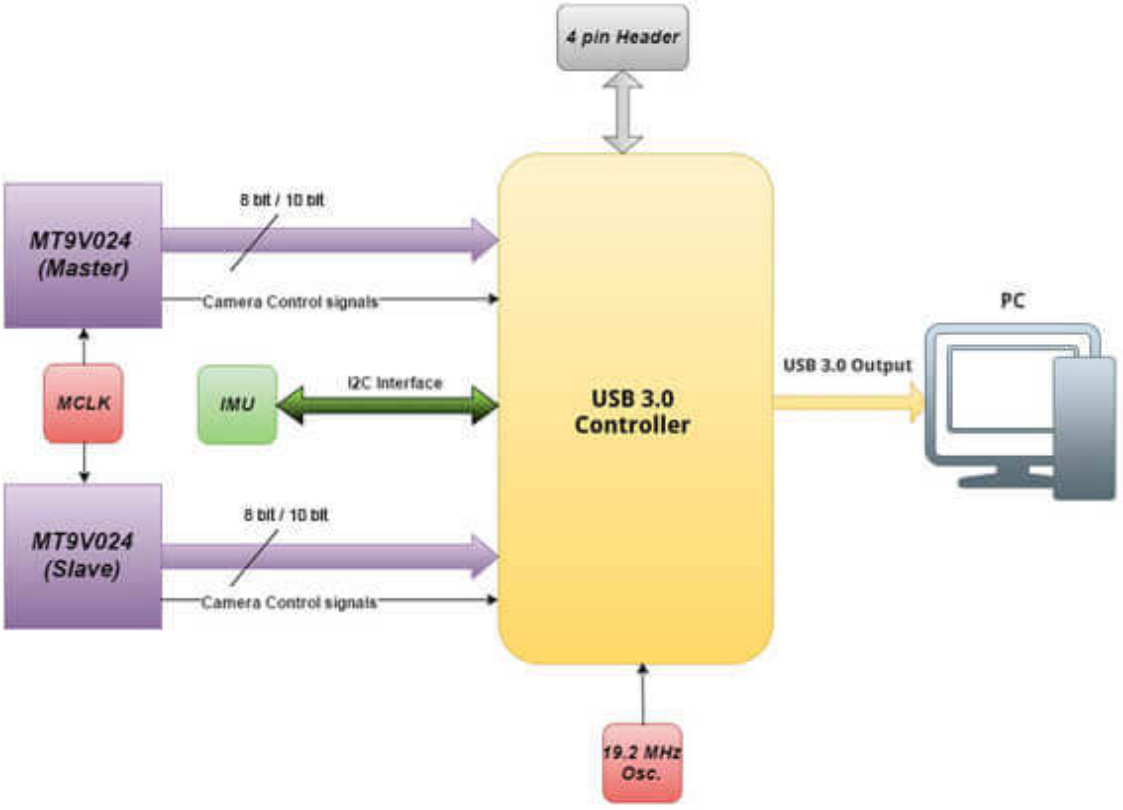


FIGURE 3.11 – Schéma fonctionnel du module de caméra TARA [3].

La caméra stéréo Tara est idéale pour des applications telles que la détection de profondeur, la carte de disparité, la construction de nuages de points 3D, la vision industrielle, l'enregistrement vidéo 3D, la robotique, etc.

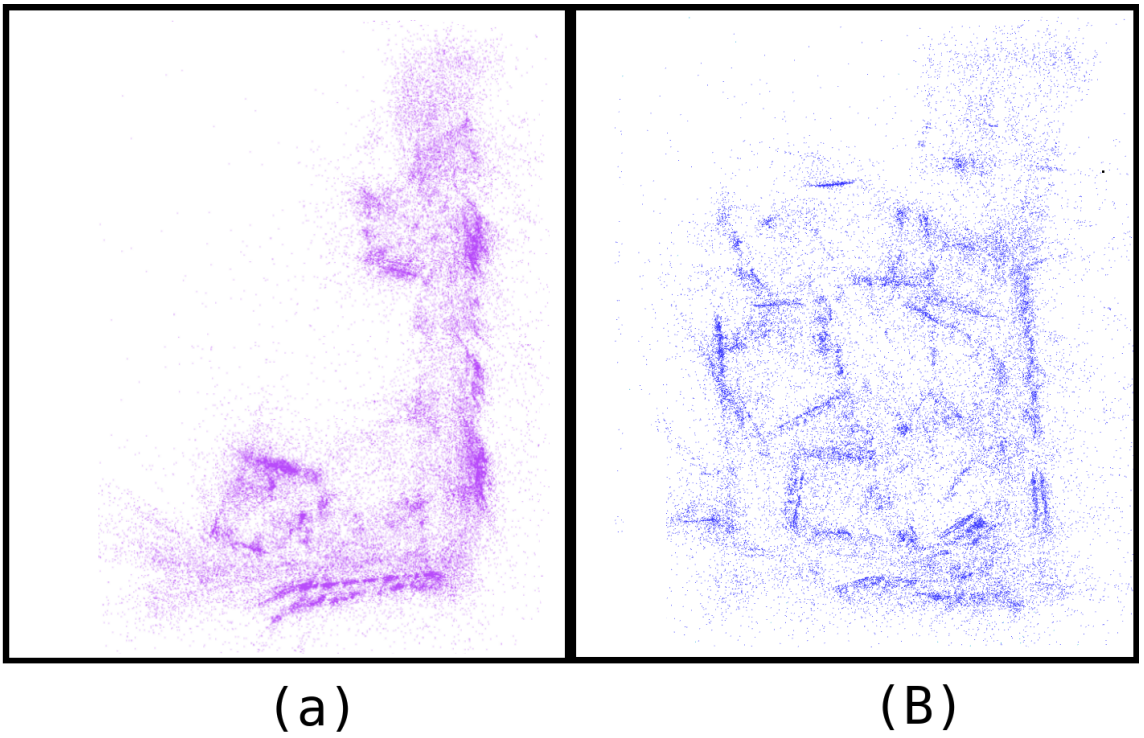


FIGURE 3.12 – Les deux cartes 3D de la base musée utilisées pour évaluer notre approche : (a) la carte à partir de laquelle les images requêtes sont extraites (b) la carte référence.

Dans notre expérimentation nous avons utilisé deux cartes de la base musée. Nous montrons dans la figure 3.13 quelques exemples d'images clés utilisées pour construire les cartes 3D. La première carte contient 443 images. Ces images ont été utilisées comme des images requêtes. La deuxième est utilisée pour construire le nuage de points avec 500 images clés qui génère 635 447 points 3D et 3,2 millions de descripteurs.

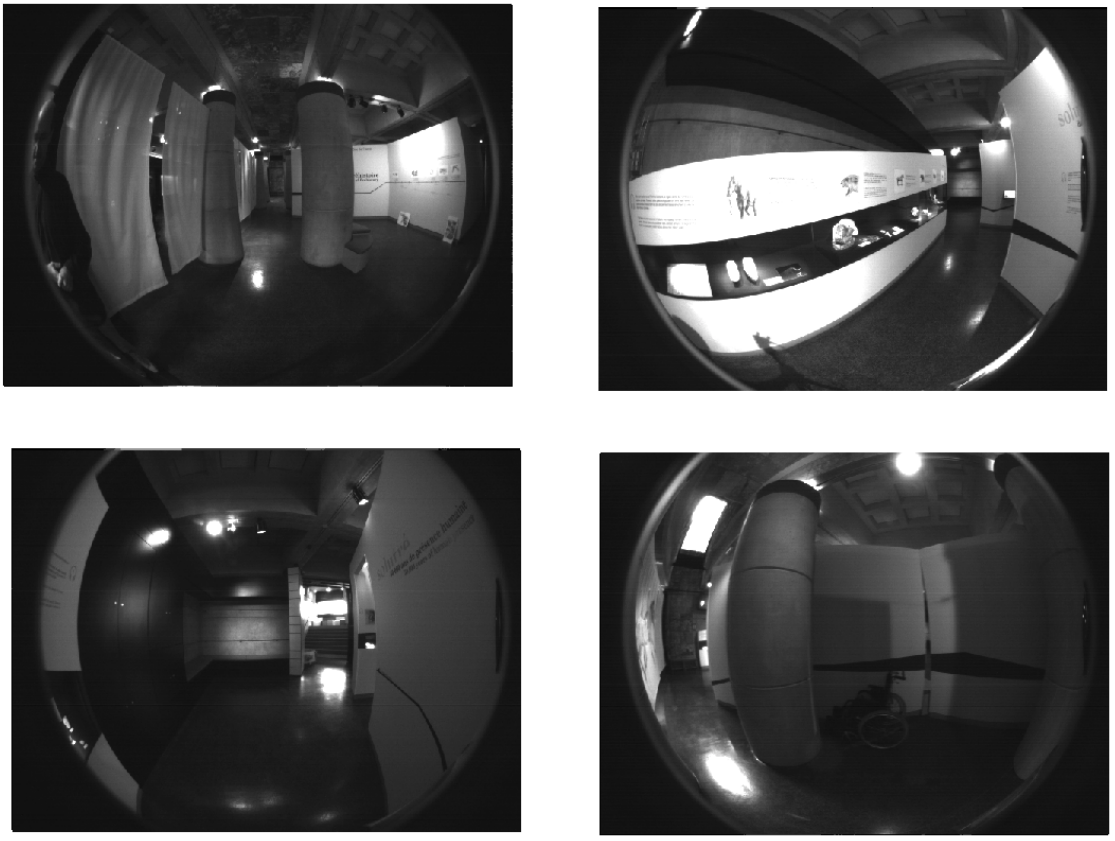


FIGURE 3.13 – Exemples d'images de la base musée.

3.3.2.2 Oxford RobotCar

Entre la période de mai 2014 à décembre 2015, Will Maddern, Geoffrey Pascoe, Chris Linegar et Paul Newman [65] ont parcouru deux fois par semaine en moyenne un itinéraire à travers le centre d'Oxford en utilisant la plateforme Robotcar d'Oxford. Cela s'est traduit par plus de 1000 km de conduite enregistrés avec près de 20 millions d'images recueillies à partir de 6 caméras montées sur le véhicule.

FIGURE 3.14 – Capteurs utilisés pour collecter l'ensemble des données Oxford RobotCar [65].

Le RobotCar Oxford est équipé par des capteurs suivants :

- 1 x caméra stéréo XB3 (BBX3-13S2C-38), 1280 x 960 x 3, 16 Hz, CCD Sony ICX445 1/3 ", obturateur global, objectif 3,8 mm, HFoV 66 °, 12/24 cm de base.
- 3 x Caméra monoculaire Point Grey Grasshopper2 (GS2-FW-14S5C-C), 1024 x 1024, 11,1 Hz, CCD Sony ICX285 2/3 ", obturateur global, objectif fisheye 2,67 mm (Sunex DSL315B-650-F2.3), 180 ° HFoV.
- 2 x LIDAR 2D SICK LMS-151, FoV 270°, 50Hz, portée 50m, résolution 0,5 °.
- 1 x LIDAR 3D SICK LD-MRS, 85 ° HFoV, 3,2 ° VFoV, 4 plans, 12,5 Hz, portée 50 m, résolution 0,125 °.
- GPS / INS : 1 x système de navigation inertielle et GPS NovAtel SPAN-CPT ALIGN, 6 axes, 50 Hz, GPS / GLONASS, double antenne.

Dans notre expérimentation nous avons utilisé deux cartes de la base Oxford Robotcar (la figure 3.15) [65]. Nous montrons dans la figure 3.16 quelques exemples d'images clés utilisées pour construire les cartes 3D. La première carte contient 1000 images. Ces images ont été utilisées comme des images requêtes. La deuxième était utilisée pour construire le nuage de points avec 1100 images clés qui génère 159 959

points 3D et 2,5 millions de descripteurs. Les images sont de dimensions 1280*768 pixels.

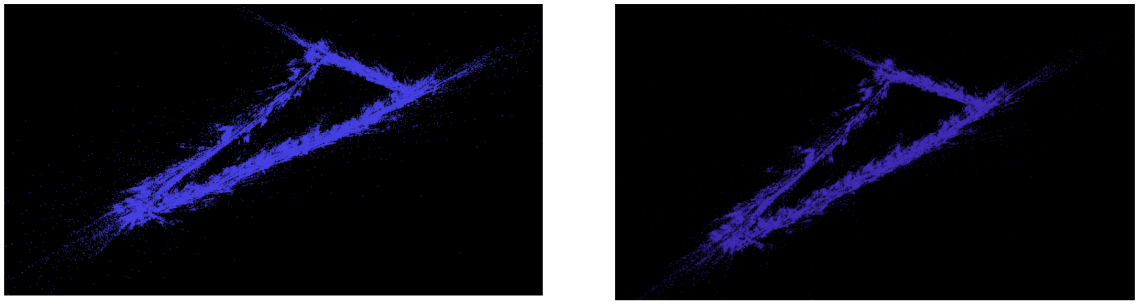


FIGURE 3.15 – Les deux cartes 3D de la base Oxford Robotcar utilisées pour évaluer notre approche : (a) la carte référence (b) la carte à partir de laquelle les images requêtes sont extraites.

FIGURE 3.16 – Exemples d'images de la base de données Oxford Robotcar [65].

3.3.3 Résultats

Dans la localisation basée sur l'image (IBL), toutes les méthodes proposées sont destinées à l'exploration d'images monoculaires et évaluées sur des bases de données telles que Dubrovnik 6k et Rome. Dans notre cas, la requête est donnée par une caméra stéréo, il est donc impossible de tester la méthode proposé avec le contenu de ces bases. Par conséquent, nous avons implémenté les deux méthodes [93] [90] qui utilisent le vocabulaire visuel pour estimer la pose et nous les testons sur les bases de données présentées sur la figure 3.8.

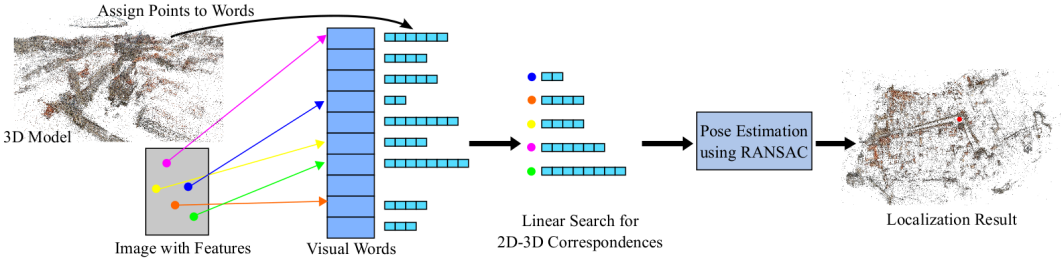


FIGURE 3.17 – Différentes étapes de l'algorithme Fast Search[90] .

La méthode décrite en [93] commence à associer chaque descripteur 2D et point 3D à un mot visuel. Les points 2D et 3D sont associés au même mot visuel sont considérés comme une relation 2D-3D (figure 3.17). Afin de regrouper toutes les correspondances obtenues entre les descripteurs 2D et les points 3D, on applique l'algorithme RANSAC pour supprimer les mauvais appariements puis on calcule la pose finale avec l'algorithme P3P [10].

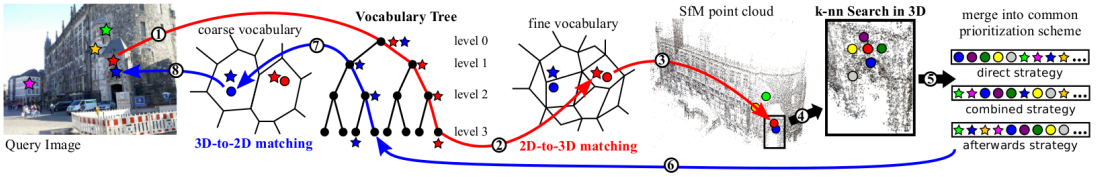


FIGURE 3.18 – Différentes étapes de l'algorithme Active Search[90] .

La figure 3.18 illustre les étapes de l'approche Active search [90]. Commençant par une mise en correspondance 2D-3D (ligne rouge), la recherche active utilise la position 3D du point correspondant et trouve les N points 3D les plus proches. Chacun de ces points est un candidat pour la recherche réciproque 3D-2D. Ensuite, on applique l'approche P2F sur l'ensemble des données 2D dans l'image requête pour trouver les correspondances 3D-2D (ligne bleue).

Descripteur	descripteur visuel	descripteur visuel et géométrique
Fast search [93]	271/443	312/443
Active search [90]	281/443	326/443
Indirecte(F2P)	191/443	256/443
Directe(F2P)	205/443	257/443

TABLE 3.2 – Comparaison du nombre d'images requêtes qui sont bien localisées par notre approche et par les méthodes ré-implémentées dans le cas de la base intérieure (musée).

Descripteur	descripteur visuel	descripteur visuel et géométrique
Fast search [93]	626/1000	661/1000
Active search [90]	631/1000	671/1000
Indirecte(F2P)	550/1000	639/1000
Directe(F2P)	607/1000	660/1000

TABLE 3.3 – Comparaison du nombre d'images requêtes localisées par notre approche et par méthodes ré-implémentées dans le cas de la base extérieure (Oxford).

Nous montrons dans les tableaux 3.2 et 3.3 le nombre de poses localisées avec succès dans les différents environnements (Oxford, Musée). Dans la première partie de tableau, nous montrons le nombre d'images requêtes localisées en utilisant le descripteur visuel KAZE avec les quatre méthodes ré-implémentées. Dans la deuxième partie, nous testons notre descripteur mixte combinant informations géométrique et visuelle sur les deux bases de données.

Le descripteur visuel et géométrique proposé montre clairement son efficacité sur toutes les méthodes ré-implémentées ([93], [90]) et les méthodes indirecte et directe. Aussi, nous avons montré que l'ajout d'informations géométriques (la hauteur Z) augmente le nombre d'images correctement localisées. Même si une pose est calculée avec succès sans information géométrique, nous avons prouvé que lorsque nous utilisons l'information de hauteur Z la pose devenait plus précise car calculée à partir de mises en correspondance plus nombreuses.

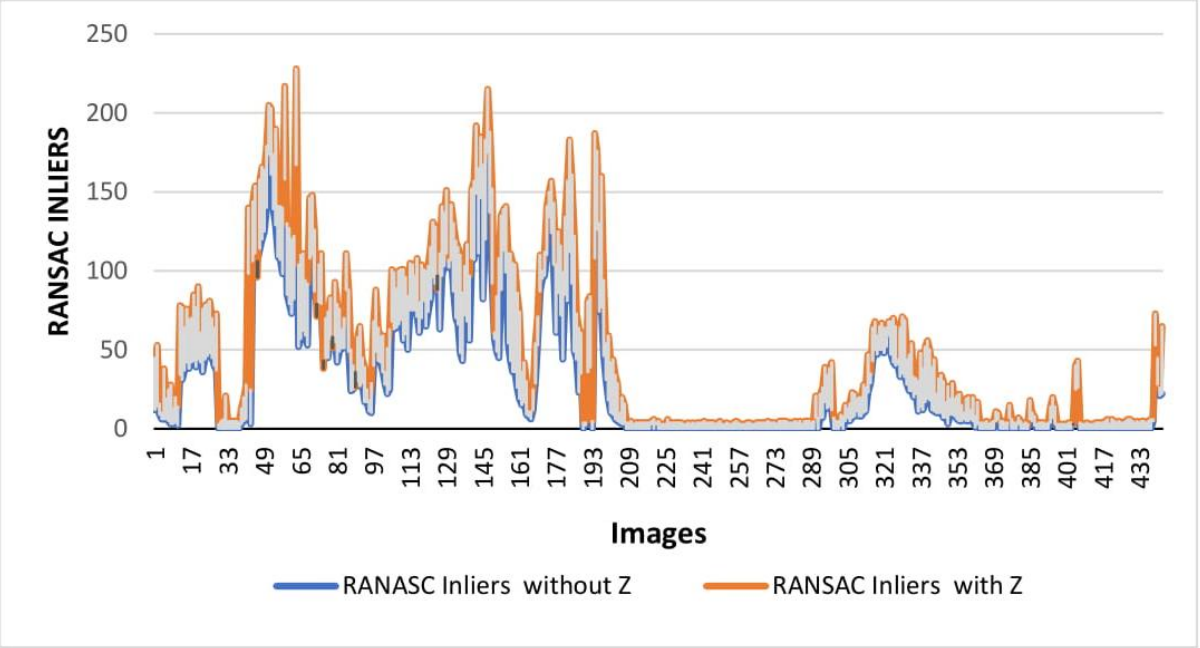


FIGURE 3.19 – Pour chaque image de requête de la base musée, le nombre d'inliers lors du calcul de la pose avec P3P-RANSAC.

La figure 3.19 montre les bons appariements "inliers" calculés par image avec deux méthodes. La première méthode (courbe bleue) consiste à calculer les poses sans information de profondeur. Donc, les descripteurs 2D de l'image et les points 3D de la carte sont directement appariés par la méthode classique [64]. La deuxième méthode (courbe orange) présente notre approche basée sur le descripteur mixte. Les résultats obtenus montrent clairement que l'ajout d'informations géométriques (Z) augmente le nombre d'inliers. Par conséquent, obtenir plus d'inliers influe positivement sur la précision des poses.

Nous présentons dans les figures 3.20 et 3.21 quelques images bien récupères par notre méthode pour les deux bases de données (musée, Oxford). Deux résultats différents sont illustrés ici : les images qui sont considérées similaires à l'image requête en vert et les images non similaires en rouge.

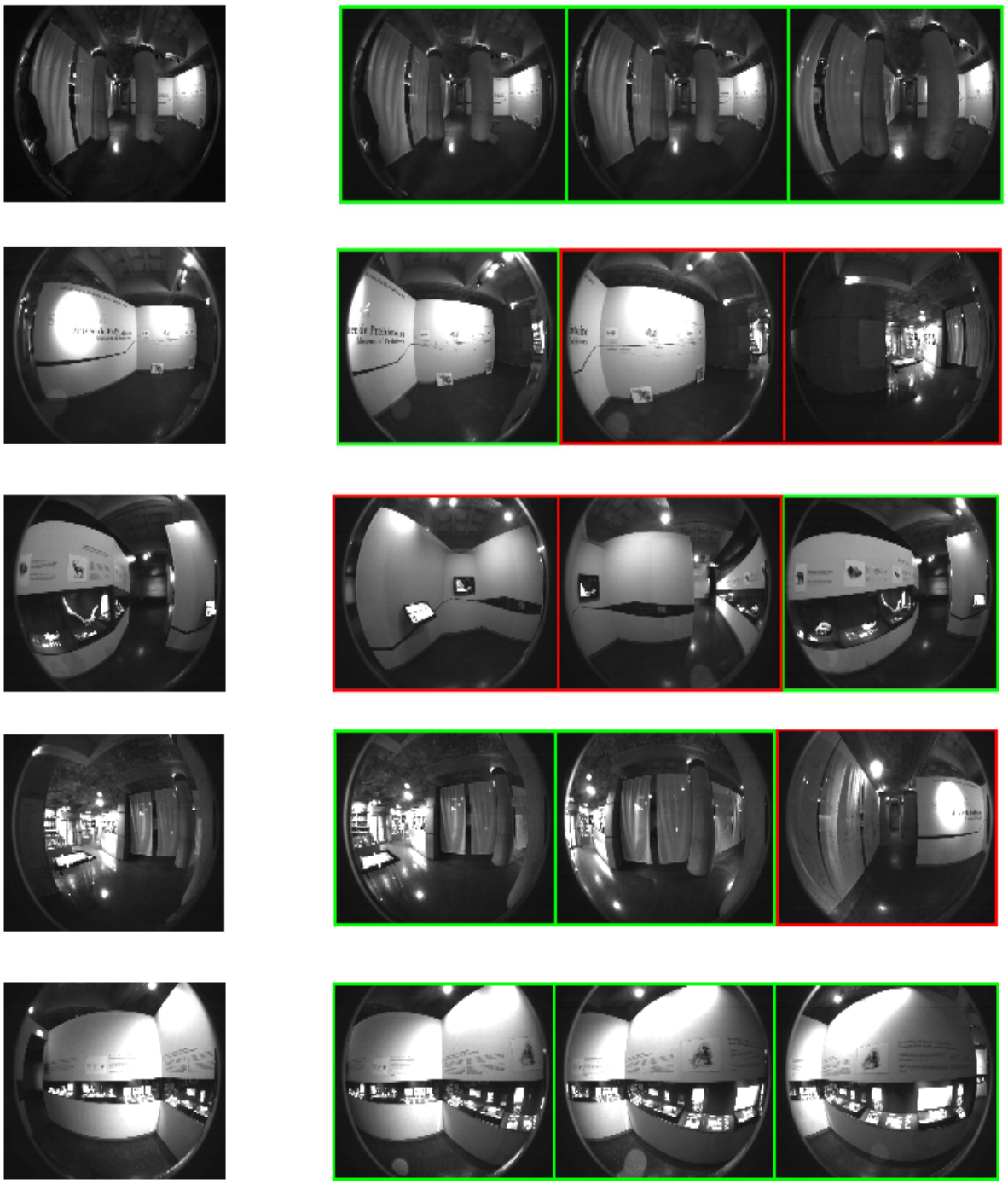


FIGURE 3.20 – Quelques résultats de notre méthode sur la base de données "musée".

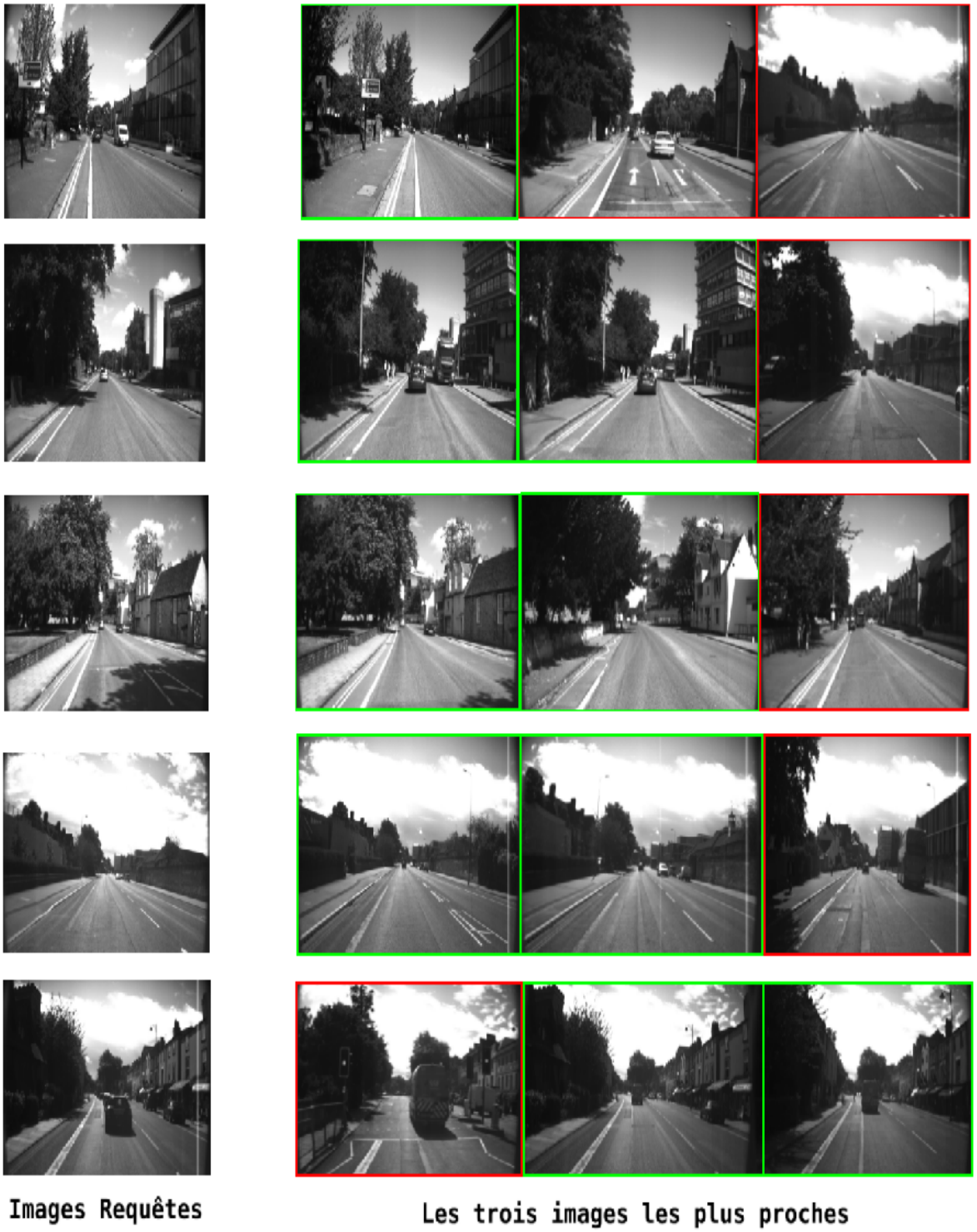


FIGURE 3.21 – Quelques résultats de notre méthode sur la base de données "Oxford".

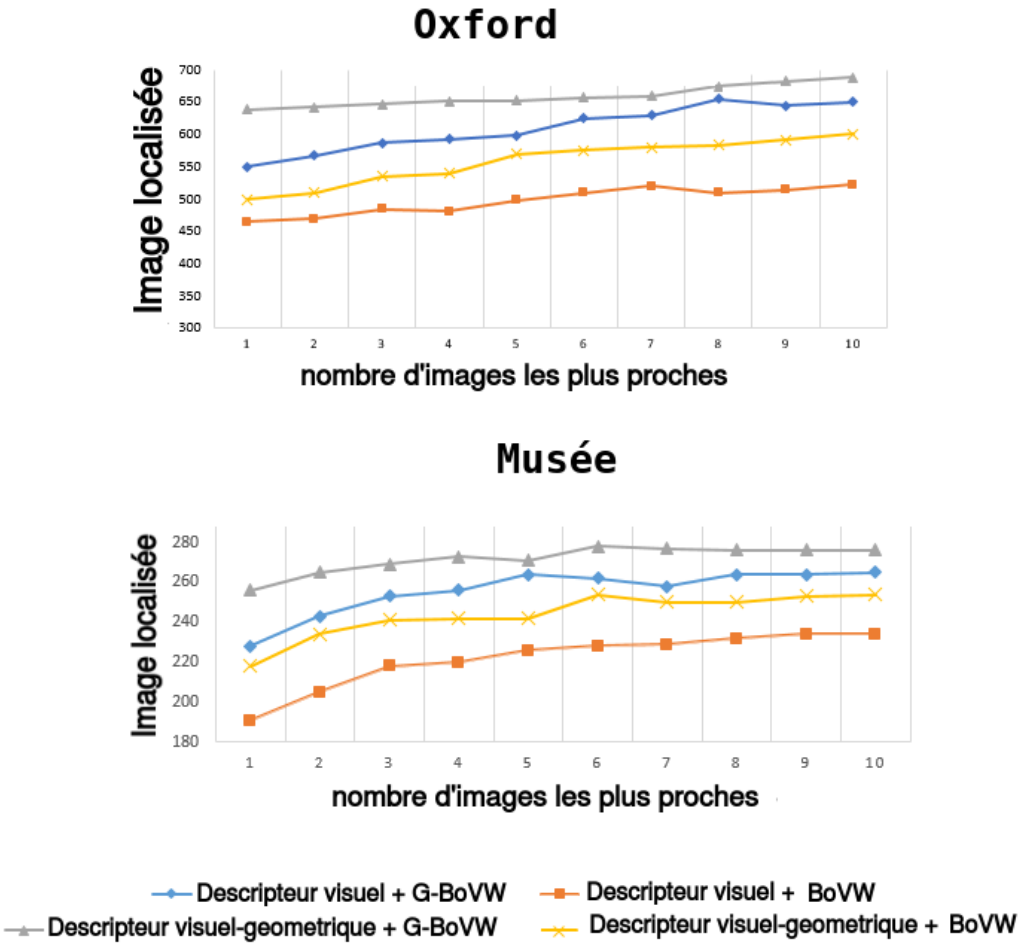


FIGURE 3.22 – Quelques résultats de notre méthode sur les deux bases de données (musée, Oxford)

La figure 3.22 présente le nombre de poses correctement récupérées lors de l'utilisation de deux listes de Knn : la première liste de KNN est obtenue à partir du sac de mots visuels BoVW (courbes jaunes et oranges) et la deuxième liste de KNN est obtenue à partir de sac de mots GBoVW (courbes grises et bleues). Dans chaque cas, nous répétons le test en changeant la taille de la liste des voisins les plus proches de 1 à 10. En outre, en utilisant les descripteurs visuels mixtes dans tous les cas (intérieur / extérieur) nous avons un nombre plus grand de poses. Bien que le Musée ait moins de requêtes qu'Oxford, le contexte est plus difficile en raison des apparences fortement identiques et des structures répétitives.

Dans les figures 3.23 et 3.24, nous montrons les cartes 3D construites par l'algorithme SLAM visuel pour les deux bases de données (Oxford, musée) ainsi que les poses d'images requêtes qui sont bien localisées.

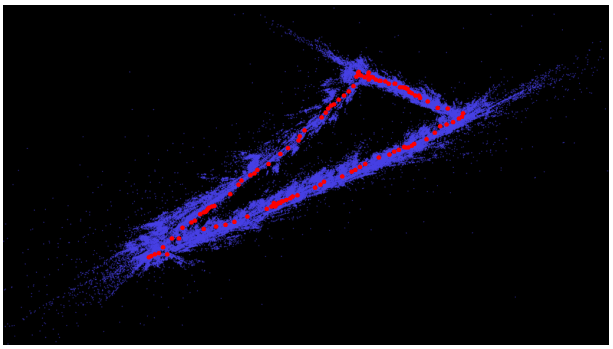


FIGURE 3.23 – La carte 3D (nuage de points en bleu) et les poses requêtes bien localisées (les points en rouge) pour la base de données "Oxford".

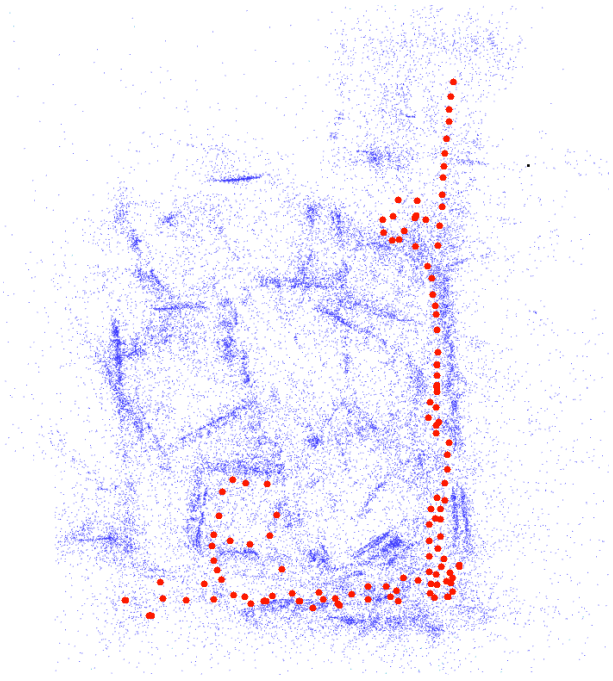


FIGURE 3.24 – La carte 3D (nuage de points en bleu) et les poses requêtes bien localisées (les points en rouge) pour la base de données "musée".

3.4 Conclusion

Dans ce chapitre, nous avons présenté une méthode d'estimation de la pose basée sur un nouveau descripteur mixte (visuel+géométrique). Nous supposons que la direction verticale est connue et que la hauteur des caméras reste constante. Pour l'image requête, nous avons utilisé la technique de triangulation pour obtenir la hauteur. Ensuite, nous concaténons le descripteur visuel KAZE avec cette information géométrique invariante afin d'obtenir un descripteur augmenté. Pour chaque point

3D nous extrayons la coordonnée Z à partir d'une carte créée par l'algorithme SLAM visuel ainsi que le descripteur visuel. Puis nous combinons ces informations pour obtenir le descripteur augmenté des images clés de la carte 3D. Nous avons pu illustrer deux types de gains dus à l'utilisation du nouveau descripteur : rendre la correspondance entre les descripteurs plus pertinente et améliorer la recherche d'images par similarité afin de générer des mots visuels robustes. La méthode est testée dans le cas d'un environnement intérieur (musée) et aussi dans un environnement extérieur (Oxford). Les expérimentations sur les méthodes re-implémentées et les méthodes classiques (indirecte, directe) montrent le gain apporté par notre approche sur les deux bases de données.

Dans ce chapitre, les résultats trouvés ont permis la mise en évidence de plusieurs limitations dans la méthode proposée, en particulier la correspondance entre les points clés 2D et les points 3D dans un environnement caractérisé par les structures répétitives. Dans ce cas, les correspondances obtenues influent négativement sur la performance du système de localisation. En utilisant les réseaux de neurones conventionnels basés sur la segmentation sémantique, nous pouvons rendre la description des points-clés 2D dans l'image et les points 3D dans la carte plus robuste dans des cas difficiles comme les structures répétitives. Ceci sera abordé dans la suite de ces travaux.

Chapitre 4

La segmentation sémantique pour améliorer la localisation visuelle

4.1 Introduction

Dans ce chapitre nous traitons le problème de la détermination de la pose (rotation + translation) d'une image requête dans une carte 3D créée par l'algorithme SfM. La méthode classique consiste à apparier les descripteurs 2D de l'image requête avec des descripteurs extraits d'une carte 3D. Puis, on calcule la pose à partir de N correspondances 2D-3D. Malgré les bonnes performances en général, les résultats sont dégradés dans de nombreuses situations à cause de la grande quantité de données aberrantes ou des changements de point de vue, ce qui rend le processus d'estimation de la pose délicat. Pour une localisation robuste, il est nécessaire d'exploiter des informations invariantes dans le processus de mise en correspondance. D'où vient l'idée d'utiliser l'information sémantique extraite en utilisant l'apprentissage profond basé sur les réseaux de neurones convolutionnels.

Dans la suite de ce chapitre nous allons montrer que la qualité des correspondances entre les descripteurs 2D et les points 3D peut être améliorée par l'ajout d'informations sémantiques à la description visuelle d'une image. Cette méthode est basée sur la segmentation sémantique. C'est un algorithme de Deep Learning qui associe une étiquette (label) ou une catégorie à chaque pixel d'une image. Elle permet de reconnaître des ensembles de pixels qui forment des catégories distinctes. De plus, la segmentation sémantique a été largement utilisée et s'est avérée utile dans les problèmes de localisation basée sur l'image de systèmes, de contrôle du trafic, de vidéo surveillance, de détection d'objets et d'imagerie médicale. La sortie est généralement une image de haute résolution qui conserve la même taille que l'image en entrée (figure 4.1).

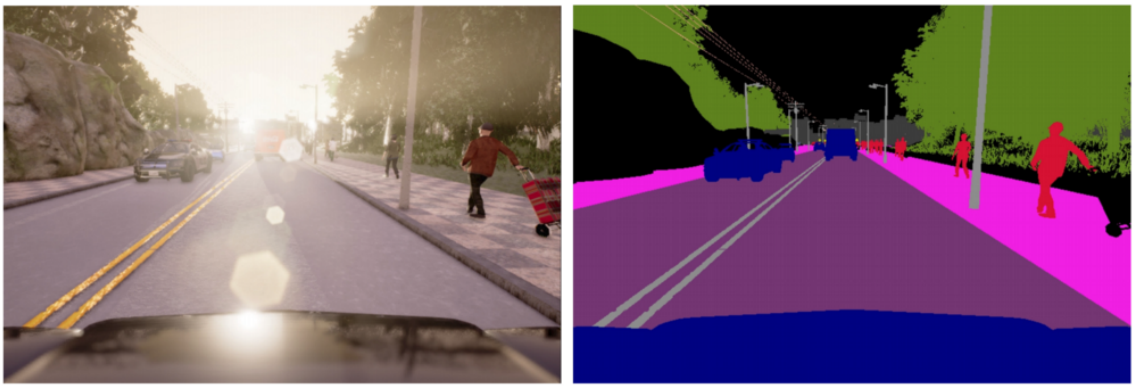


FIGURE 4.1 – Exemple de la segmentation sémantique d'une image dans une zone urbaine [39].

Nous montrons par un exemple sur la figure 4.2 que l'information sémantique peut jouer le rôle d'un filtre et qu'elle est capable d'augmenter le nombre des bons appariements. Motivés par ces résultats, nous présentons dans ce chapitre une approche afin d'améliorer la qualité des appariements par l'intégration de l'information sémantique dans la phase de la mise en correspondance 2D-3D.

FIGURE 4.2 – Correspondance 2D-2D en exploitant les informations sémantiques.

La segmentation sémantique ne produit pas toujours une segmentation parfaite. Il y a des débordements aux limites de certaines classes et un effet de flou aux frontières. Même avec l'augmentation des performances des algorithmes de segmentation sémantique, la sortie de l'image segmentée rend les frontières inter-classes imprécises ce qui nécessite un lissage. Les détecteurs de points clés produisent beaucoup de caractéristiques sur ou près des frontières des objets parce que ces régions dans l'image satisfont les contraintes des détecteurs. Donc notre but est de réduire l'erreur de segmentation sémantique à la frontière par une méthode d'étiquetage robuste.

Pour évaluer les performances de nos approches, nous allons commencer les tests par la méthode mono-label et voir dans quelle mesure cela améliore la localisation visuelle en terme de précision. En prenant en compte les résultats précédents, nous allons faire les mêmes tests pour montrer l'efficacité de nos approches (multi-labels et descripteur probabiliste) par rapport à la méthode standard (mono-label).

En fonction des informations fournies par le réseau, nous suggérons d'améliorer la labellisation basée sur la sortie de la couche SoftMax du réseau de neurones de quatre manières différentes : (1) Descripteur Multi-label : un point clé est associé à plus d'un label si la probabilité de la classe principale n'est pas décisive. Sinon, le

pixel est associé à un seul label. (2) Nous combinons la sortie de deux réseaux de neurones convolutifs pour générer un ensemble d'étiquettes pour chaque pixel. Le premier réseau est un réseau de segmentation sémantique classique. Le second est un détecteur de bord sémantique qui donne un ensemble d'étiquettes pour le pixel à la frontière entre deux classes ou plus. Ensuite, en fonction de la sortie de ces réseaux, on garde soit une seule étiquette du premier réseau, soit un ensemble d'étiquettes du second. (3) Nous combinons les probabilités données par les couches SoftMax de la segmentation sémantique et du détecteur de bord sémantique pour étiqueter les pixels à la frontière. (4) Descripteur probabiliste : nous augmentons l'efficacité du descripteur en concaténant le descripteur visuel avec le vecteur de probabilité donné par la couche SoftMax. Ces deux manières d'intégrer l'information sémantique seront évaluées et comparées dans la partie expérimentale.

4.2 Méthode proposée

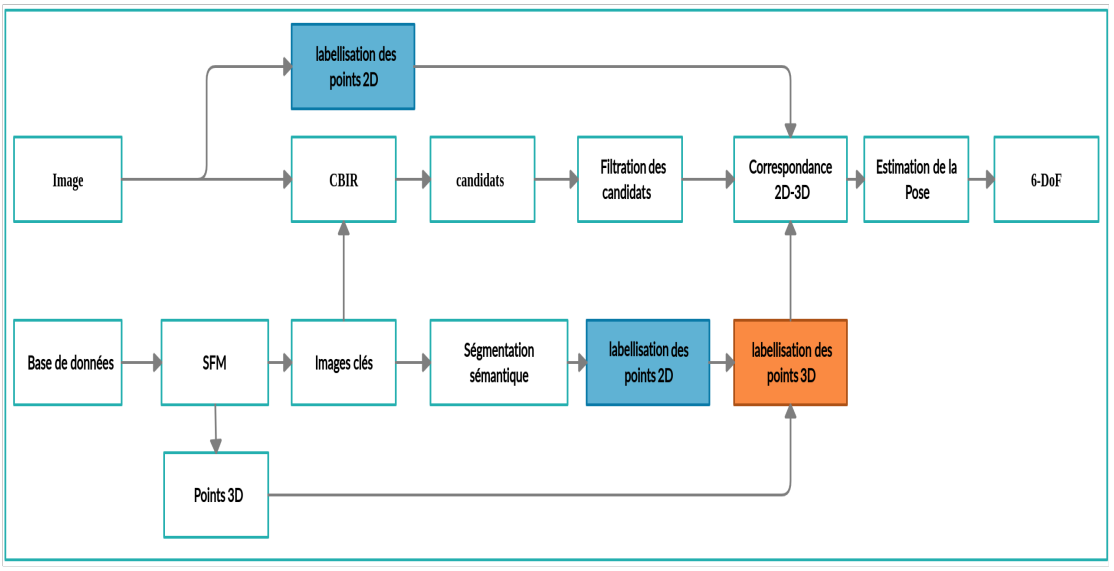


FIGURE 4.3 – Framework proposé.

Notre approche (figure 4.3) est composée d'un ensemble d'étapes clés. Nous commençons par trouver les images clés les plus proches d'une image requête par l'algorithme de recherche par similarité basée sur l'apprentissage profond décrit dans [54]. Notre objectif est d'utiliser les images les plus proches (KNN) pour extraire une région du modèle 3D basée sur la liste de visibilité dans laquelle l'image requête est probablement vue.

Dans la reconstruction 3D (SfM, SLAM), le modèle 3D se compose de m poses de caméras, de n points 3D et de leur liste de visibilité. Cette liste contient pour chaque point 3D l'ensemble des caméras où il peut être vu. La figure 4.4 présente un exemple de visibilité sur un ensemble de points 3D P_i et ses caméras C_j correspondants. Les informations de visibilité sont très utiles pour améliorer l'efficacité de la recherche de correspondances. Cela permet de supprimer les points qui sont moins susceptibles de générer des correspondances potentielles.

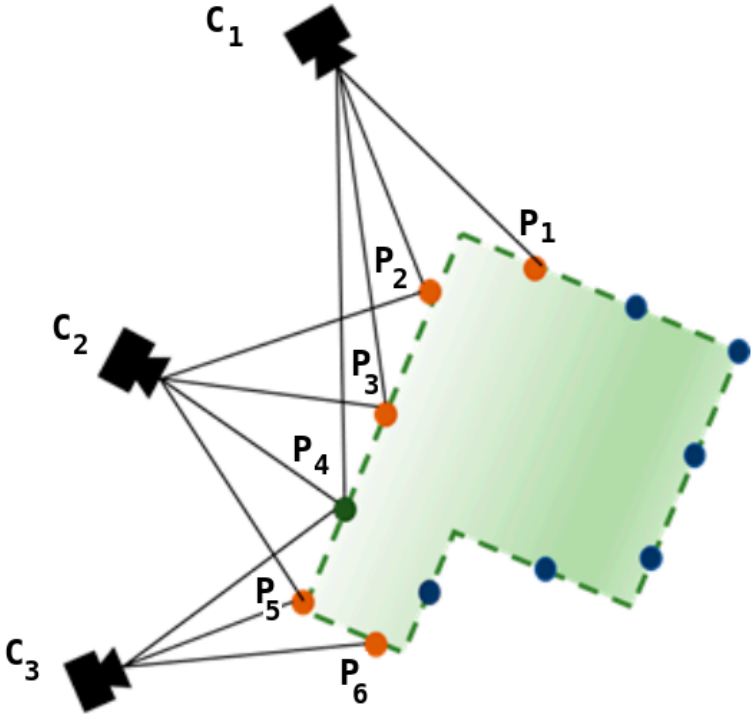


FIGURE 4.4 – Graphe de visibilité [128] : Les points oranges P3, P4 sont visibles par les caméras (C1, C2, C3). Le point vert P5 est visible par les caméras (C2,C3) et les points bleus ne sont pas visibles.

La liste d'images les plus proches (KNN) obtenue contient un ensemble d'images qui ne sont pas nécessairement similaires à l'image requête à cause de la limite de l'algorithme CBIR. Pour sélectionner les plus similaires de manière robuste, nous avons ajouté les informations sémantiques à la fois à l'image requête et aux images clés et nous les associons en attribuant un score à chaque paire en fonction du nombre d'inliers en supprimant les images qui ont un nombre d'inliers inférieur à 20. Par conséquent nous obtenons une liste d'images clés que nous augmentons avec les voisines dans le graphe de poses de la carte SFM, comme illustré sur la figure 4.5. Pour rappel, deux poses sont connectées dans le graphe de pose si elles partagent des points 3D. Ensuite, nous considérons tous les points 3D visibles dans cette nouvelle

liste d'images clés. L'étape suivante consiste à faire correspondre les caractéristiques en utilisant la stratégie 2D-3D et en tenant compte des informations sémantiques. La dernière étape consiste à calculer la pose relative.

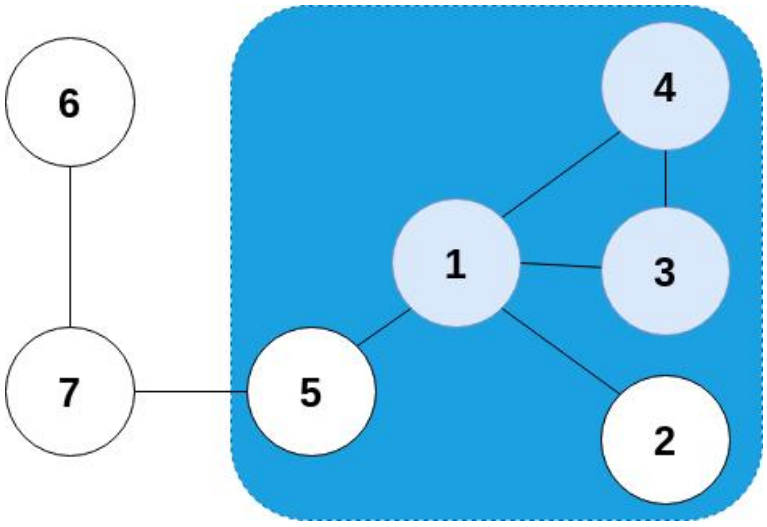


FIGURE 4.5 – Graphe de poses de la carte SFM. Si la sortie de la correspondance sémantique et du filtrage est (4,3,1) alors nous extrayons tous les points 3D associés aux images clés (1,2,3,4,5).

4.2.1 Correspondance sémantique

En général, deux images sont mises en correspondance par l'extraction des points clés kp à l'aide d'un détecteur (Harris dans notre cas), puis en faisant correspondre leurs descripteurs locaux. D'après [64], une correspondance sera acceptée si le test ratio est :

$$\frac{||d_{kp} - d_{kp1}||_2}{||d_{kp} - d_{kp2}||_2} < \epsilon \tag{4.1}$$

où d_{kp} est le descripteur associé à un point dans la requête, d_{kp1} et d_{kp2} sont les deux descripteurs les plus proches de d_{kp} issus de l'image clé. ϵ est généralement fixé à 0.7. Une faiblesse de la méthode standard est qu'elle ne prend pas en considération deux points sont appartenus à la même classe d'objets (figure 4.6). Cela provoque une augmentation des données aberrantes même après le filtrage RANSAC[77].

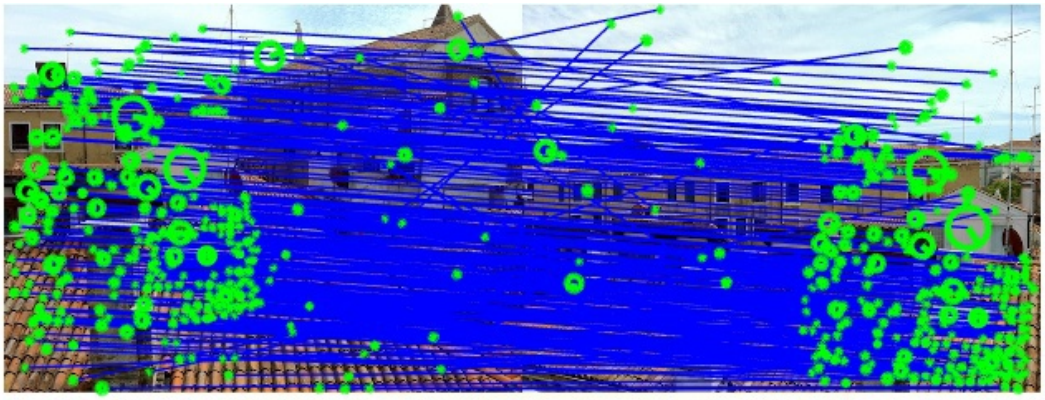


FIGURE 4.6 – Correspondance entre deux images sans prendre en considération les informations sémantiques [60].

Pour résoudre ce problème, nous détectons d'abord les points clés k_p et extrayons leurs descripteurs fi de l'image d'entrée I. Ensuite, nous attribuons à chaque point clé k_p un ensemble de labels $L = \{l_1, .., l_n\}$ avec la méthode décrite en dessous. Pour chaque point clé k_p, nous supprimons les labels correspondant aux classes non stationnaires (bus, animal, personne, ...). Ensuite, si la liste de labels est vide, nous supprimons ce point-clé. Nous faisons correspondre un point-clé $kp1$ dans la première image à tous les points-clés de la deuxième image si $L_1 \cap L_2 \neq \varnothing$ où L1, L2 sont les listes des labels des points-clés dans la première et la deuxième image (figure 4.7). Nous utilisons RANSAC [38] pour calculer la matrice fondamentale. Puis, nous vérifions si le nombre d'inliers trouvés est supérieur à un seuil (20 inliers dans nos expériences). Si c'est le cas alors nous conservons l'image, sinon nous la supprimons. Nous affectons un score à chaque paire en fonction du nombre d'inliers trouvés.

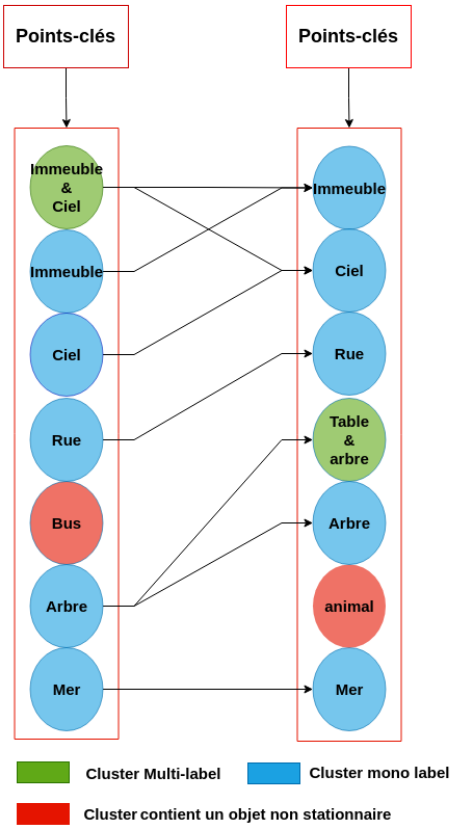


FIGURE 4.7 – Correspondance sémantique entre les points clés

4.2.2 Labellisation multiple basée sur la segmentation sémantique

Affecter un label à chaque pixel est toujours un défi difficile pour de nombreuses raisons, telles que la faiblesse dans les algorithmes de segmentation sémantique ou le débordement au voisinage d'une frontière dans l'image. Les points clés sont largement utilisés dans les techniques de localisation basée image (IBL) et la détection se concentre sur des points spécifiques d'une image sélectionnée en fonction de critères comme les coins et les points avec de fortes variations de texture. Habituellement, les positions des points détectés sont autour et à l'intérieur de l'objet. Lorsque l'image segmentée contient plus d'une classe (ciel, bâtiment, ...), on ne peut pas prendre en compte le label de classe obtenu par le point détecté à la frontière pour deux raisons principales : (1) la probabilité de la classe majeure du point détecté n'est pas décisive (2) le chevauchement entre deux classes oblige à produire un label de classe imprécis comme le montrent les figures 4.8 et 4.9. Pour résoudre ce problème, nous avons proposé d'utiliser une méthode multi-label détaillée dans la section suivante.

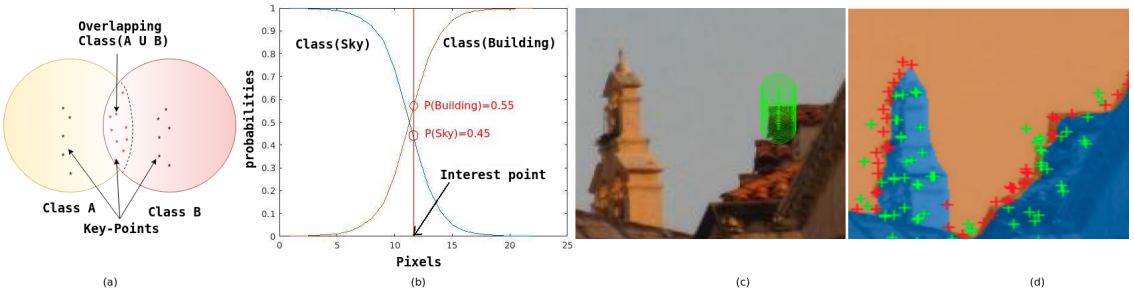


FIGURE 4.8 – (a) Chevauchement entre deux classes (b) La courbe bleue présente les probabilités de la classe ciel et la courbe rouge présente les probabilités de la classe immeuble (c) L'image contient deux classes (Ciel, Immeuble) et montre les points échantillonnés selon la courbe représentée en (b). (d) L'image segmentée montre le chevauchement entre deux classes à la frontière. En outre, la figure montre les points-clés détectés : (1) points-clés avec une fausse étiquette de classe (symbole rouge), (2) points-clés avec une vraie étiquette de classe (symbole vert).

Dans cette section, nous proposons différentes manières de labellisation des pixels par les réseaux de neurones. Dans la première méthode appelée mono-labels, nous affectons à chaque point-clé kp le label du pixel sous-jacent. C'est la façon classique pour labelliser les points clés. Dans l'architecture de réseaux de neurones avant la sortie finale (la couche ArgMax), le réseau contient la couche SoftMax. Cette couche a la même taille que l'image entrée et contient pour chaque pixel un vecteur de probabilités $y = [y_1, .., y_N]$. Ce vecteur de taille N correspond au nombre de classes sur lesquelles le réseau a été entraîné. Dans la méthode mono-label, le label de chaque pixel est défini par :

$$l = \underset{i}{\operatorname{argmax}}\, y_i \tag{4.2}$$

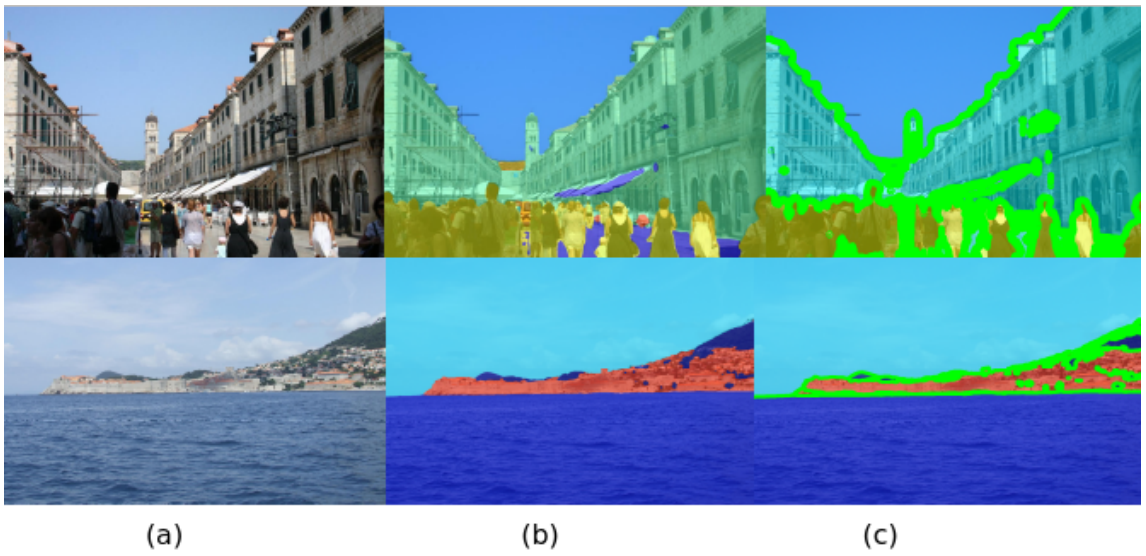


FIGURE 4.9 – (a) Images clés utilisées pour construire le modèle de SFM (Dubrovnick) (b) Images clés segmentées à l'aide de l'algorithme Encnet [123] entraîné sur ADE20K [127], (c) détections des pixels avec de faibles écarts de probabilités entre les classes sémantiques (en vert clair).

Dans une deuxième méthode nommée multi-labels, pour améliorer la labellisation aux frontières, nous utilisons la couche SoftMax du réseau de neurones pour extraire plusieurs labels pour les points clés. Nous prenons également en compte les pixels voisins pour réduire le bruit donné par la segmentation sémantique. Nous appliquons d'abord un filtre moyenneur à la couche SoftMax avec une taille de fenêtre allant de 3×3 à 7×7 dans les expériences. Pour un pixel nous conservons le label avec la probabilité la plus élevée p^* puis nous ajoutons tous les labels avec la probabilité p lorsque

$$p/p^* > \epsilon \tag{4.3}$$

4.2.3 Labellisation multiple basée sur une combinaison de la segmentation sémantique et détection de contour sémantique.

La détection des contours sémantiques [118, 119, 12, 44] est un nouveau domaine de recherche dans le domaine de la vision par ordinateur. Il est basé sur un réseau de neurones convolutifs (CNN) pour détecter des pixels situés dans des contours entre deux ou plusieurs classes sémantiques (figure 4.10). Le réseau prend une image en entrée et sort N 2D-maps où N est le nombre de classes sémantiques. Chaque map 2D est une matrice de probabilités de pixels appartenant à une classe sémantique spécifique. Dans la figure 4.11, nous présentons un exemple de sortie sémantique de

détection de bord [44] pour les classes principales existant dans l'image. Pour chaque classe sémantique, nous présentons sa carte 2D (en bas) et les pixels correspondants dans l'image (en haut).

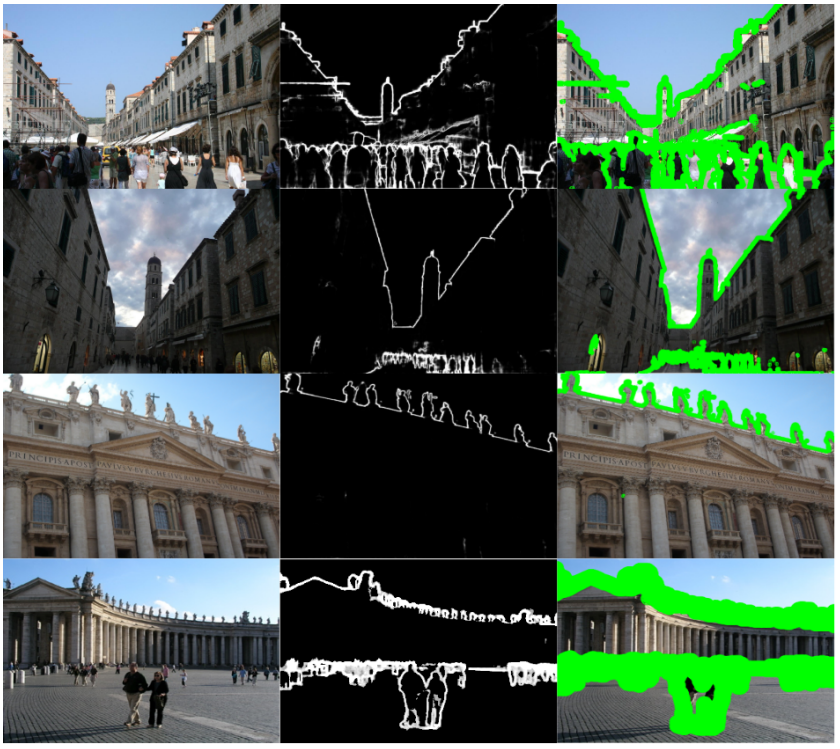


FIGURE 4.10 – Exemples de carte de bord sémantique globale détectée à l'aide de l'algorithme DFF [44] entraîné sur Ade20k et leurs pixels correspondants.

Chaque pixel est caractérisé par un vecteur de 150 probabilités. L'indice dont les probabilités sont au-dessus de zéro correspondent aux étiquettes du pixel. Dans la figure 4.11, nous présentons deux exemples : (A) le pixel est affecté à trois étiquettes (Bâtiment, Ciel, Végétation) (B) le pixel est affecté à deux étiquettes (Bâtiment, Personne).

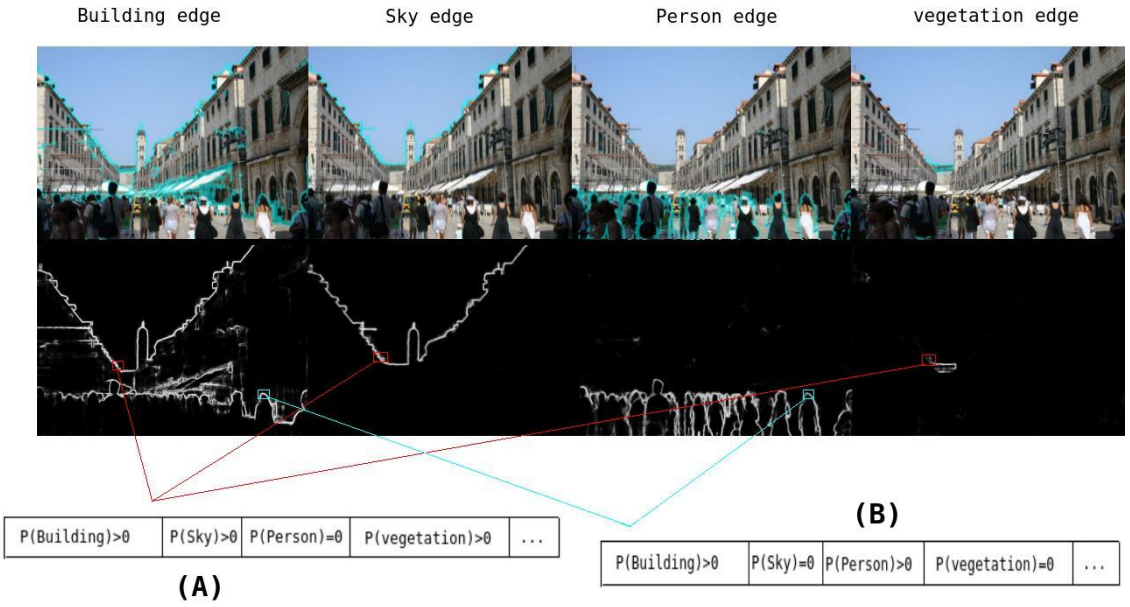


FIGURE 4.11 – Détection du bord sémantique pour chaque classe sémantique prédite à l'aide de l'algorithme DFF [44] entraîné sur Ade20k. Dans la figure, nous prenons deux pixels (A, B) comme exemple. En fonction des probabilités extraites, le premier pixel (A) a trois étiquettes (Ciel, Bâtiment, Végétation) et le second (B) a deux étiquettes (Bâtiment, Personne).

La différence entre la méthode précédente et celle-ci est que les pixels de bordure sont étiquetés à l'aide d'un détecteur sémantique. Donc, étant donné une image, nous commençons par détecter l'étiquetage à l'aide de deux réseaux de neurones (voir figure 4.12). Le premier réseau attribue à chaque pixel une étiquette (segmentation sémantique). Le deuxième réseau attribue un ensemble d'étiquettes aux pixels situés sur le bord (détecteur de bord sémantique). Nous combinons les résultats des deux réseaux profonds avec l'intention d'améliorer la labellisation. Nous attribuons plusieurs étiquettes au pixel s'il a été détecté par le détecteur de contour sémantique. Sinon, le pixel est affecté à une seule étiquette obtenue à l'aide de l'algorithme de segmentation sémantique. L'avantage d'utiliser un détecteur de bord est d'augmenter la précision d'affectation des étiquettes aux pixels entre les classes sémantiques.

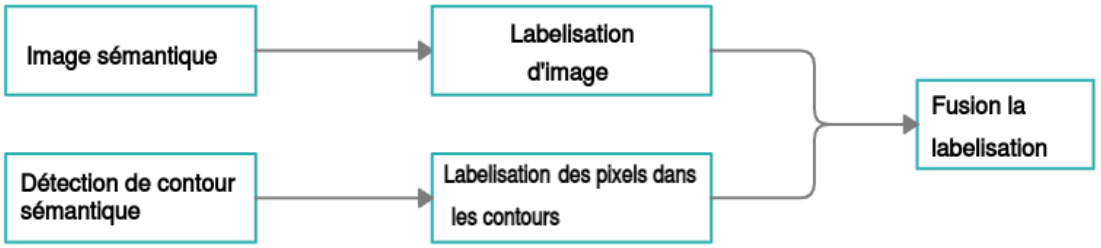


FIGURE 4.12 – labelisation des pixels par la segmentation sémantique et le détecteur sémantique de contour.

4.2.4 Labelisation multiple basée sur la combinaison de couches SoftMax de la segmentation sémantique et de détecteur du contour sémantique

La labelisation dans cette partie dépend des deux réseau de neurones convolutionnels (sémantique et de contours). La principale distinction ici est que les pixels à la frontière partageront les probabilités extraites de la couche SoftMax des deux réseaux profonds. Comme décrit dans la section précédente, cette couche contient un vecteur par pixel dont chaque élément est la probabilité de l'étiquette correspondante au pixel.

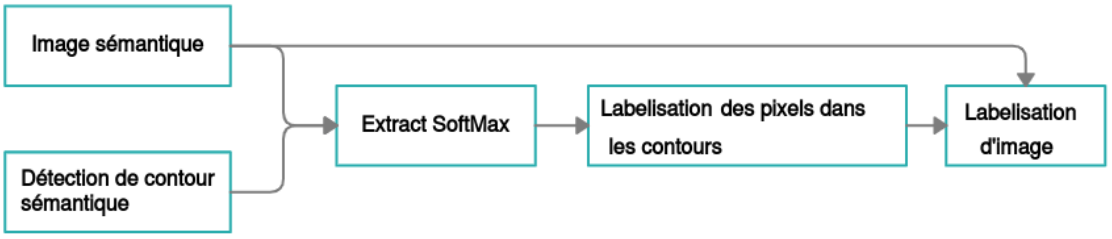


FIGURE 4.13 – Étiquetage des pixels en combinant les probabilités données par la couche SoftMax à partir de la segmentation sémantique et du détecteur de bord sémantique.

Nous décrivons dans la figure 4.13 toutes les étapes de notre proposition. Nous commençons par reconnaître les pixels identifiés par le détecteur sémantique. Pour ces pixels, nous séparons leur vecteur de probabilités des couches SoftMax sur les deux réseaux profonds. À ce stade, les pixels à la frontière sont décrits par deux composantes : (1) Vecteur de probabilités $y_{seg} = [y_{seg_1}, .., y_{seg_N}]$ de l'algorithme de segmentation sémantique (2) Vecteur de probabilités $y_{edge} = [y_{edge_1}, .., y_{edge_N}]$ de l'algorithme de segmentation de détection du sémantique . Enfin, nous combinons les vecteurs de probabilité pour obtenir y_{se} selon :

$$y_{se} = \frac{y_{seg} + y_{edge}}{2} \tag{4.4}$$

Le but de la combinaison des vecteurs de probabilité est d'améliorer le degré de pertinence de l'étiquetage des points-clés. Après avoir obtenu y_{se} pour chaque point-clé, nous utilisons l'équation 4.3 pour attribuer à chaque point-clé leurs étiquettes correspondantes. Pour les pixels non détectés par le détecteur de contour sémantique, cela signifie qu'ils ne sont pas dans la frontière. Nous leur attribuons donc les étiquettes données par la segmentation sémantique seule. Dans la dernière étape, l'équation 5.1 est utilisée pour déterminer pour chaque point 3D leurs étiquettes correspondantes.

4.2.5 Descripteur probabiliste

En exploitant l'information sémantique d'une façon différente, nous allons présenter dans cette section une alternative à la méthode multi-label. Les deux méthodes sémantiques seront ensuite évaluées et comparées dans la partie expérimentale sur deux bases de données de large échelle.

L'ajout d'informations aux descripteurs locaux s'est avéré utile dans [23] qui concatène les descripteurs de caractéristiques (SIFT) et leur profondeur correspondante pour former un descripteur mixte. Inspirés par ce travail, nous abordons le problème de labellisation dans cette section de manière différente en intégrant l'information sémantique dans le descripteur visuel. Donc, au lieu d'assigner un label à chaque point clé, nous travaillons directement avec le vecteur de probabilités extrait de la couche Softmax. Comme décrit dans la section 4.2.2, ce vecteur contient les probabilités que le pixel appartienne à chaque classe. Par conséquent, chaque point-clé est caractérisé par deux éléments : descripteur local Kp et son vecteur de probabilités P. Par conséquent, nous les concaténons pour former un descripteur probabiliste combinant les probabilités et les informations visuelles à utiliser dans le processus d'appariement. Dans nos expériences, le descripteur local est SIFT (vecteur de dimension 128) et le nombre de classes est de 150. En conséquence, la taille du descripteur final est de 278. La distance entre deux descripteurs est calculée en utilisant la norme L2.

4.2.6 Labellisation des points 3D

Dans le nuage de points SFM, chaque point 3D Q est associé à N points-clés kp_i. Dans le cas mono-label, [110] attribue le label le plus fréquent à un point 3D. Dans notre cas, chaque point clé a un ensemble de labels $L_i = \{l_{i,j}\}$ et les probabilités associées $P_i = \{p_{i,j}\}$. Pour chaque label j nous calculons la probabilité PL_j que le point 3D Q soit associé au label j :

$$PL_j = \frac{\sum_i p_{i,j}}{N} \tag{4.5}$$

Comme certains labels ont été rejetés précédemment à cause de l'équation (4.3), la somme de PL_j n'est pas nécessairement égale à un. Il y a une probabilité $1-\sum_j PL_j$ qu'un point n'ait pas de label. Pour un point Q donné, on garde le label avec la probabilité la plus élevée PL^* puis on ajoute tous les labels avec la probabilité PL de sorte que $PL/PL^* > \epsilon$ ($\epsilon = 0.5$ dans notre expérimentation). Si la labelisation n'est pas correcte pour une minorité d'images, les labels dérivés de ces images seront supprimés en utilisant ce processus.

4.2.7 Correspondance des caractéristiques et des points 3D

Dans la localisation basée sur l'image, nous avons deux manières différentes pour faire correspondre les caractéristiques : 2D-3D c'est à dire chercher les correspondances à partir des descripteurs requêtes (F2P) ou 3D-2D où le processus d'appariement est inversé (P2F). L'algorithme ci-dessous montre comment nous calculons la liste M des correspondances avec la stratégie 2D-3D. Afin de trouver N appariements, nous utilisons l'algorithme P3P [117] avec RANSAC [38] pour calculer la pose de la caméra de l'image requête.

Algorithm 1 Correspondance 2D-3D avec des contraintes sémantiques

Require: point clé requête $KP_{i=1..n}$ avec label l_1 ,
point clé 3D $Q_{j=1..m}$ avec label l_1
$M=\emptyset$
for `i=1 to n` **do**
  for `j=1 to m` **do**
    d(j)= $\|\text{KP}_i$ - $Q_j\|_2$
  end for
  d=Sort(d)
  if $\frac{d(1)}{d(2} \leq \epsilon$ **then**
    $M=M \cup\{\text{KP}_i,Q_j\}$
  end if
end for
return M

Pour la stratégie P2F, nous avons juste besoin d'inverser l'ordre des boucles. La sortie dans ce cas est une liste plus longue de correspondances M parce que le nombre de points 3D est généralement plus élevé que le nombre des caractéristiques dans l'image de requête. Par conséquent, le calcul de pose prend plus de temps.

4.3 Étude expérimentale

4.3.1 Implémentation

Nous utilisons le réseau EncNet [124] (figure 4.15) pré-entraîné sur l'ensemble de données ADE20K [127] qui contient 150 catégories sémantiques annotées sur 20210 images. Les images ADE20K sont annotées de manière dense avec des objets. Dans la figure 4.14, la première ligne montre les exemples d'images, la deuxième ligne montre l'annotation des objets, et la troisième ligne montre l'annotation des images par instance. Au total ADE20K contient 150 catégories sémantiques qui incluent des classes comme ciel, route, herbe et des objets comme personne, voiture, lit.

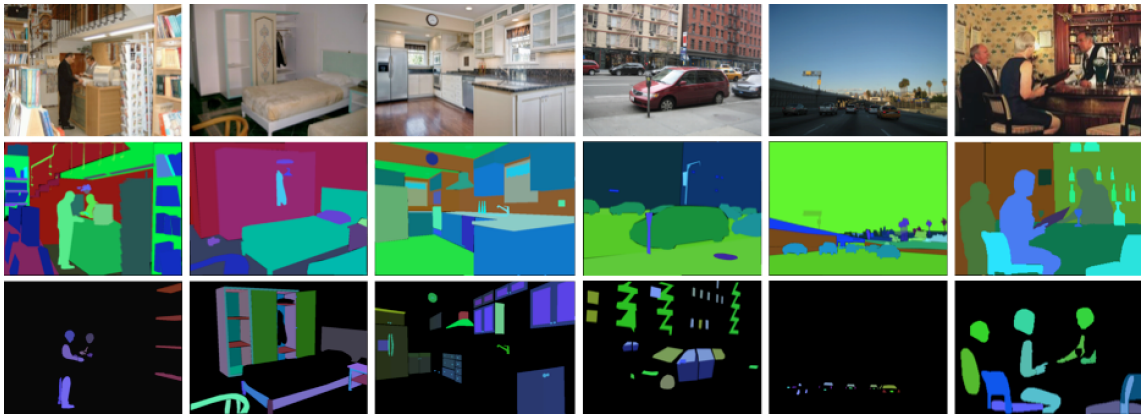


FIGURE 4.14 – Exemple des images annotées sur Ade20K [125].

Nous commençons par calculer les 100 images les plus proches de l'image requête en utilisant une amélioration récente sur Resnet [87], l'architecture CNN utilisée pour générer les signatures pour les images requêtes et les images clés. Nous calculons la similarité entre les images en utilisant la bibliothèque FLANN [73] pour construire un KDtree et nous arrêtons le processus après avoir trouvé 100 candidats. Pour estimer la pose relative, nous avons utilisé le solveur P3P [117] à l'intérieur de RANSAC [38] avec 4096 itérations sur l'ensemble des correspondances obtenues en faisant correspondre les caractéristiques requête avec les points 3D avec une erreur de reprojection maximale $\epsilon \leq 6$ à pixel. D'après [60], [91], nous considérons qu'une requête est bien enregistrée avec succès uniquement si le nombre d'inliers est supérieur à 12 après l'étape de filtrage avec RANSAC. Nous appliquons un filtre dans le but de supprimer les mauvaises images : d'abord, nous calculons le score pour chaque paire en fonction de la correspondance sémantique et nous supprimons les voisins les plus proches avec des scores nuls. Dans la plupart des cas, nous avons remarqué que les candidats restants sont connectés dans le graphe de poses, ce qui signifie qu'ils ont de nombreux points 3D en commun. Nous utilisons la stratégie points clés 2D vers points 3D (F2P) pour tester les performances de notre approche.

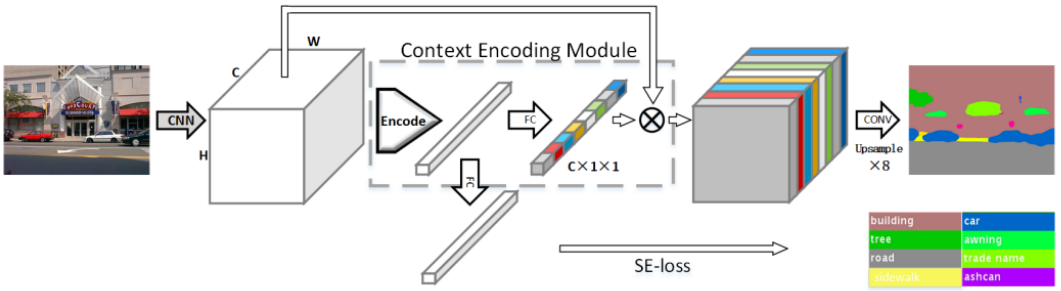


FIGURE 4.15 – L'architecture EncNet [124].

4.3.2 Résultats

Dans cette section nous montrons l'efficacité de notre approche sur les deux bases de données de large échelle Rome et Dubrovnik. Les résultats obtenus sur ces deux base publiques nous permettent de comparer nos méthodes avec les méthodes de l'état de l'art. La base Dubrovnik contient 800 images requêtes et 6044 images clés. La précision de l'image localisée s'effectue par la détermination de la distance euclidienne entre la pose estimée et la vérité-terrain. Deux seuils ont été choisis par l'auteur de [60] pour évaluer la précision de la pose récupérée. Les images qui ont une erreur de position inférieure à 18.3 m et les images qui ont une erreur de position supérieure à 400 m. Pour la base Rome, une image est considérée comme localisée si le nombre d'inliers est supérieur à 12. Ces seuils ont été repris par la suite par de nombreux auteurs qui ont travaillé sur ces bases.

Pour une description robuste, nous avons combiné les informations sémantiques partagées entre le point-clé et ses pixels voisins. Puis, nous avons testé la performance de notre méthode en augmentant chaque fois la taille de la fenêtre qui définit le voisinage d'un point en prenant 1×1, 3×3, 5×5, puis 7×7 pixels. Cette augmentation dans la taille des fenêtres a pour but de partager le maximum d'informations utiles.

Le tableau 4.1 montre l'effet de la taille de la fenêtre autour d'un point clé : lorsque la taille de la fenêtre augmente, le pixel partage plus d'informations avec ses voisins.

Taille de la fenêtre (pixel)	register images	Q1 (m)	Q2 (m)	Q3 (m)	images avec erreur<18.3 m	images avec erreur>400 m
1*1	797	0.35	0.92	2.86	727	7
3*3	**798**	**0.25**	**0.78**	**2.35**	**734**	**6**
5*5	796	0.34	0.96	2.75	729	9
7*7	797	0.37	1.01	2.96	725	13

TABLE 4.1 – Résultats détaillés de la méthode multi-labels pour l'ensemble de données Dubrovnik avec différentes tailles de fenêtre. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation.

La précision peut être évaluée en regardant Q1, Q2, Q3 qui représentent les quartiles des erreurs de localisation. Nous montrons expérimentalement que le partage d'informations entre les voisins et le pixel cible augmente le nombre d'images requêtes correctement localisées. Pour la suite, nous choisissons une fenêtre de taille 3×3, et nous montrons que l'étiquetage des pixels dans les bordures avec plus d'un label réduit le taux d'erreur et augmente le nombre d'images requêtes localisées, le nombre d'inliers et la précision des poses calculées (tableau 4.3). Comme le montre le tableau 4.2, nous avons obtenu 40% des points clés avec plus d'un label. Ce tableau indique également le nombre des caractéristiques et de points 3D étiquetés comme objets dynamiques.

Base de données	Dubronvik	Rome
nombre de descripteurs avec 2 labels ou plus	4,527,894	7,745 439
nombre de descripteurs avec un seul label	5,078,423	13,769,671
nombre de points 3D avec 2 labels ou plus	754753	1408261
nombre de points 3D avec un seul label	1,132,131	2,658,858
nombre de descripteurs dynamiques	315.418	796.467
nombre de points 3D dynamiques	90541	216554

TABLE 4.2 – Détails sur les labels des points 3D et leurs descripteurs

Méthodes	# register imgs	Q1 (m)	Q2 (m)	Q3 (m)	imgs < 18.3 m	imgs >400 m	temps (s)
Méthode 1	791	0.45	1.22	3.78	715	9	**0.66**
Méthode 2	798	0.24	0.78	2.35	734	**6**	0.72
Méthode 3	**799**	0.22	0.81	2.29	735	9	0.97
Méthode 4	**799**	**0.20**	**0.74**	**2.25**	**738**	8	1.07
Méthode 5	797	0.25	0.80	2.29	730	8	0.95

TABLE 4.3 – Résultats de comparaison entre mono label (méthode 1) et les méthodes multi-labels proposées. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation.

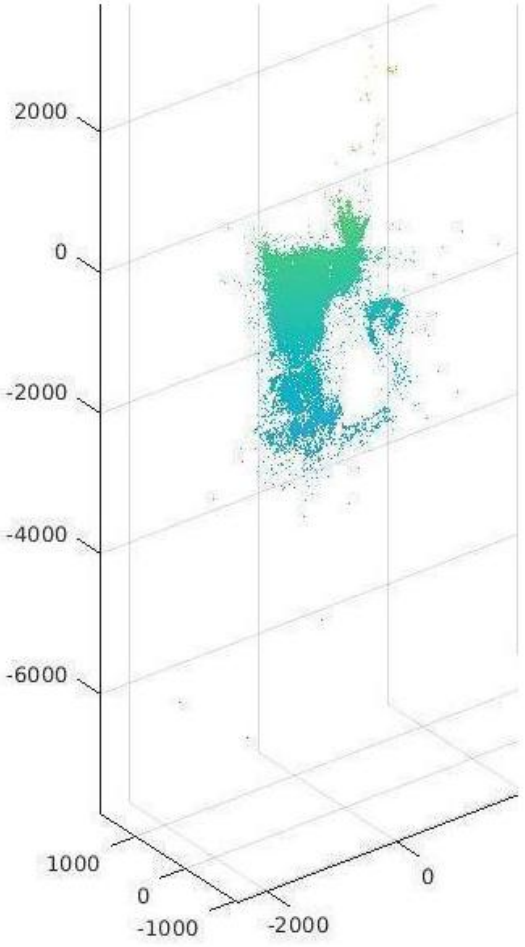


FIGURE 4.16 – Nuage des points 3D récupérées à partir des correspondances 2D-3D avec les images requêtes et la carte 3D globale de la base Dubrovnik.

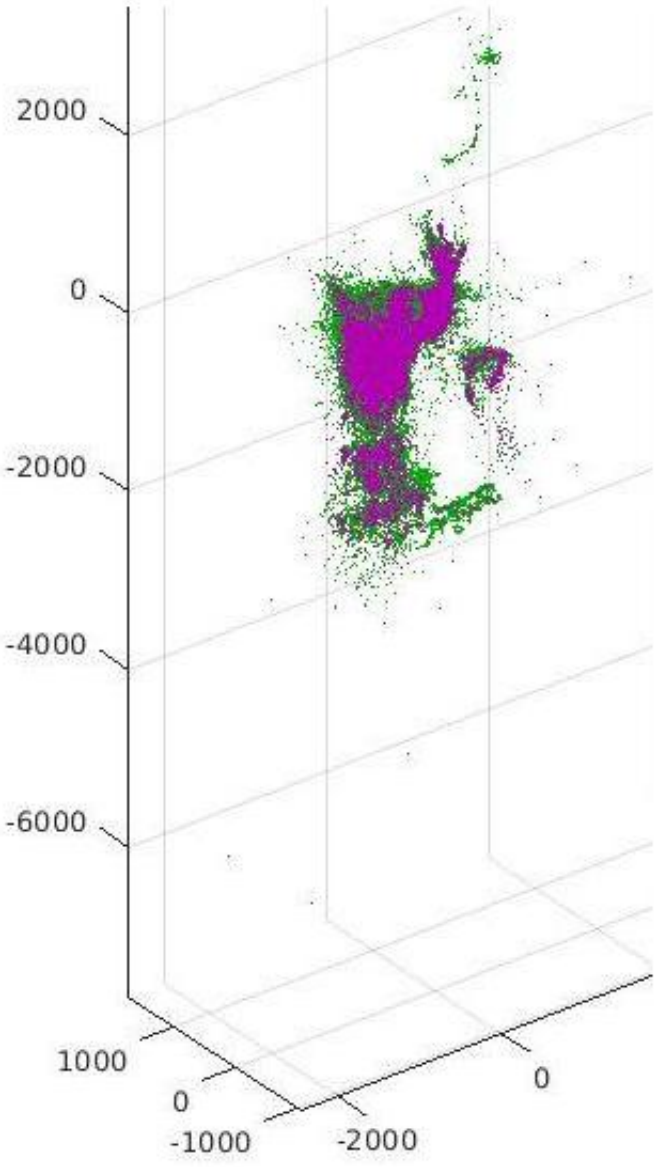


FIGURE 4.17 – Nuage des points de la base de données Dubrovnik (points en verts) et la projection des points 3D d'images requêtes (points en violets).

La figure 4.16 présente les points 3D calculés à partir des correspondances entre les descripteurs d'images requêtes et les descripteurs de la carte 3D globale. Puis dans la figure 4.17, nous avons projeté les points 3D trouvés à partir d'images requêtes sur les nuages de points 3D de la base de données Dubrovnik.

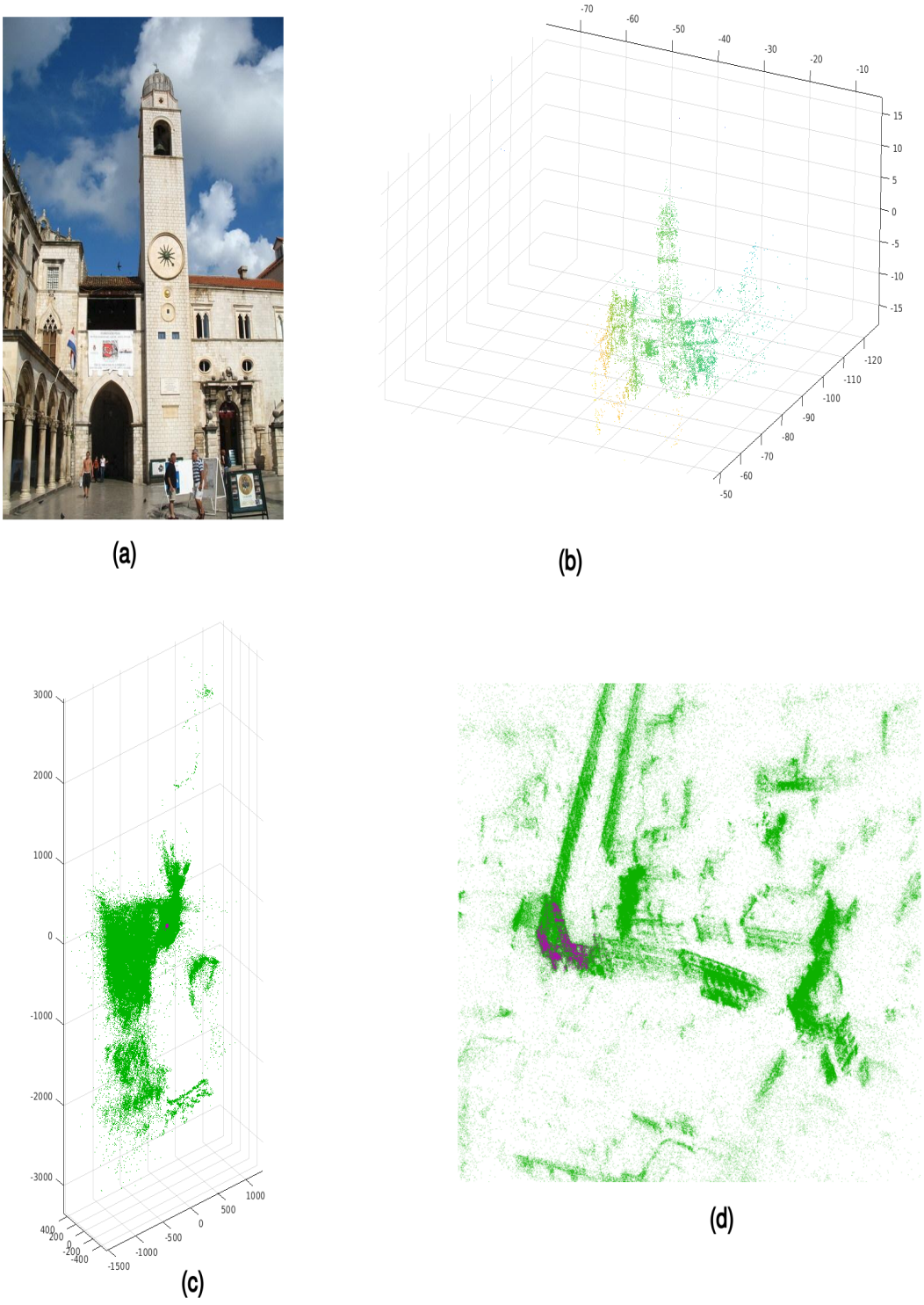


FIGURE 4.18 – (a) image requête. (b) nuage des points 3D donnée par notre approche multi-labels. (c) nuage de points 3D de la carte globale (Dubrovnik). (d) projection de points 3D de l'image requête sur la carte globale.

La figure 4.18 montre l'exemple d'une image requête bien localisée dans la carte globale et ses points 3D calculés en fonctions de la liste d'images les plus proches. À partir de cette liste nous avons collecté les points 3D correspondant aux images clés et nous les avons projeté sur la carte globale de la base Dubrovnik.

Méthodes	# register images	Q1 (m)	Q2 (m)	Q3 (m)	images avec erreur < 18.3 m	images avec erreur >400 m	temps (s)
Sans information a priori							
Méthode 4	799	**0.20**	**0.74**	**2.25**	**738**	**8**	1.07
Sattler [91]	783	0.40	1.40	5.90	685	16	0.31
Sattler [92]	795	0.40	1.40	5.30	704	9	**0.25**
Raul [34]	**800**	1.09	7.92	27.76	550	10	0.62
Youji [37]	781	0.42	1.28	4.67	-	12	0.46
Li [60]	753	7.50	9.30	13.40	655	-	-
Choudhary [25]	788	0.88	3.10	11.83	-	-	-
Avec information a priori (direction verticale et hauteur connues)							
Zeisl [122]	796	**0.19**	**0.56**	**2.09**	744	7	**3.78**
Svarm [106]	**798**	-	0.56	-	**771**	3	5.06

TABLE 4.4 – Comparaison de notre approche avec des méthodes de l'état de l'art pour Dubrovnik. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation.

Dans le tableau 4.4, nous comparons nos résultats avec deux catégories de méthodes. Les méthodes qui utilisent des informations a priori [122] [106] (direction verticale et hauteur connues) et les méthodes sans information a priori [93] [92] [34] [25] [60]. Les méthodes qui utilisent des informations a priori présentent de meilleurs résultats relativement aux erreurs liées au seuils $\leq$ 18,3 m et >400 m. Ces deux seuils ont été proposée par Noah Snavely dans [60]. Les méthodes que nous proposons n'utilisent pas d'information a priori. Elles permettent d'obtenir des résultats légèrement inférieurs aux méthodes avec a priori. Mais comparées aux autres méthodes sans a priori, elles donnent le plus grand nombre de poses avec une erreur inférieure à 18,3 m.

Méthodes	# register image	temps	Inliers
Méthode 1	988	**0.77**	356
Méthode 2	997	0.89	387
Méthode 3	998	0.97	396
Méthode 4	**999**	1.12	**401**
Méthode 5	995	0.96	346

TABLE 4.5 – Comparaison entre les résultats de nos méthodes pour Rome.

Nous testons les performances de nos propositions sur la base de données de Rome dans le tableau 4.5. Pour chaque méthode proposée, nous estimons le nombre de poses localisées avec succès, le temps moyen et le nombre d'inliers trouvés après l'étape RANSAC.

Méthodes	# register images	inliers	temps
Sattler [91]	977	100	0.29
Sattler [92]	991	200	**0.28**
Youji [37]	985	-	0.33
Li [60]	924	-	0.87
Méthodes 4	**999**	1.12	**401**

TABLE 4.6 – Comparaison de notre approche avec des méthodes de l'état de l'art pour Rome.

Dans le tableau 4.6, nous comparons les résultats avec les méthodes de l'état de l'art. Nous obtenons le plus grand nombre d'images enregistrées pour la base de données Rome. Nous rappelons ici que la pose estimée est considérée comme valide si le nombre d'inliers est supérieur à 12 après filtrage des correspondances 2D-3D à l'aide de RANSAC. Nous avons réussi à localiser la majorité d'images requêtes avec un nombre d'inliers important.

La figure 4.19 compare les quartiles des erreurs de localisation pour toutes les méthodes. Nous pouvons voir que la précision des poses récupérées sont meilleures avec notre méthode. Nos contributions augmentent clairement le nombre d'inliers avec un grand nombre de poses trouvées. L'obtention de plus d'inliers influence positivement la précision des poses.

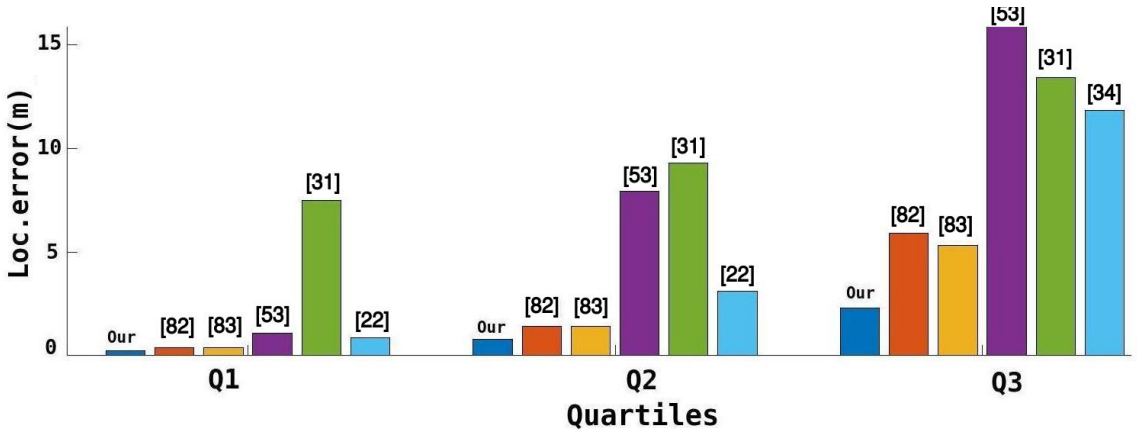


FIGURE 4.19 – Comparaison des quartiles avec les méthodes de l'état de l'art, notre méthode (méthode multi-labels) atteint la meilleure précision de localisation sur l'ensemble de données de Dubrovnik.

4.4 Conclusion

Dans ce chapitre, nous avons présenté une approche basée sur la segmentation sémantique. Nous avons montré que la qualité des correspondances entre les descripteurs 2D et les points 3D peut être améliorée par l'ajout d'informations sémantiques à la description visuelle d'une image. Deux contributions ont été proposées dans ce chapitre pour améliorer la labellisation en se basant sur la sortie de la couche SoftMax : (1) Descripteur multi-labels : un point clé est associé à plus d'un label si la probabilité de la classe principale n'est pas décisive, sinon, le pixel est associé à un seul label. (2) Descripteur probabiliste : nous augmentons l'efficacité du descripteur en concaténant le descripteur visuel avec le vecteur de probabilité donné par la couche SoftMax.

Les résultats obtenus par ces deux méthodes montrent l'efficacité de notre approche en termes de précision et temps par rapport aux méthodes de l'état de l'art. En comparant les deux méthodes proposées, la méthode multi-labels fournit de meilleurs résultats que la méthode probabiliste mais elle demande moins de temps de calcul. Par contre, la méthode probabiliste est plus rapide mais moins précise. Une implémentation sur CPU de l'approche proposée est difficile car la segmentation sémantique a besoin de ressources matérielles très puissantes. L'algorithme de segmentation par réseaux de neurones réclame un temps d'exécution égale à 7 s par image. D'où la nécessite d'utiliser des GPU pour accélérer la segmentation et réduire le temps d'exécution. Ce temps est réduit à 0.3 s par image ce qui rend le système de localisation plus rapide.

L'analyse des résultats obtenus dans ce chapitre a permis la mise en évidence de plusieurs limitations relatives à la méthode proposée, en particulier la détection des contours sémantiques entre les classes. Cela peut engendrer des mauvais appariements entre les points clés 2D et les points 3D dans la carte. Ces limitations peuvent être améliorées en utilisant des informations supplémentaires pour rendre la pose plus précise. Les réseaux de neurones convolutionnels basés sur la détection de contours ont récemment connu plusieurs succès. Ces algorithmes sont capables d'identifier les pixels sur les contours sémantiques entre les classes. En combinant les informations sémantique données par les deux réseaux (labellisation de contour, labellisation de l'image), la description sémantique des points clés sera plus pertinente et fiable. En perspective d'amélioration, nous pensons que la combinaison de ces deux types de réseaux peut nous permettre d'obtenir des poses plus précises.

Chapitre 5

La segmentation sémantique pour améliorer la recherche d'image par le contenu

5.1 Introduction

Dans la localisation basée sur l'image, les méthodes sont divisées en deux catégories principales : (1) méthodes directes et (2) méthodes indirectes. Les méthodes directes recherchent les correspondances en comparant directement les caractéristiques des images requêtes aux points 3D de la scène. Ces méthodes ont montré de bonnes performances sur des bases de données de petite taille en utilisant une information supplémentaire (comme des données GPS, l'orientation de la prise de vue, etc.) pour rendre l'estimation de la pose plus précise. Les méthodes indirectes sont, elles, destinées aux bases de données de large échelle. L'approche consiste à sélectionner, parmi les images de la base de données, celles qui sont le plus similaires à l'image requête afin de guider et simplifier le processus de localisation de celle-ci. La méthode est composée de trois étapes présentées dans la figure 5.1 : (1) chercher, parmi les images clés, celles qui sont les plus similaires à une image requête en entrée, (2) extraire des caractéristiques 2D de l'image requête en utilisant un descripteur, puis trouver des correspondances en les comparant aux points 3D de la scène, (3) calculer la pose à partir des correspondances trouvées, en utilisant l'algorithme $P3P$.

Le coeur de cette méthode indirecte est donc la recherche d'images similaires à une image requête. Ce chapitre présente le travail réalisé concernant l'amélioration d'un système de recherche d'images par similarité à une image requête.

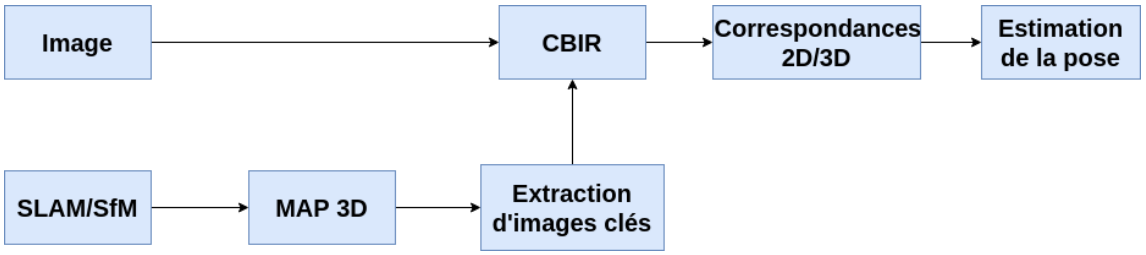


FIGURE 5.1 – Les étapes clés de la méthode indirecte.

5.2 Méthode proposée

La qualité et l'efficacité d'un système de recherche d'images par le contenu dépendent du choix de la signature d'image. Les images que nous recherchons font partie d'un ensemble d'images qui ont été utilisées par l'algorithme SFM (Structure From Motion) pour construire une carte 3D. La signature d'image doit donc permettre, par similarité, de sélectionner le sous-ensemble d'images permettant de simplifier et d'améliorer la localisation de la pose correspondant à l'image requête.

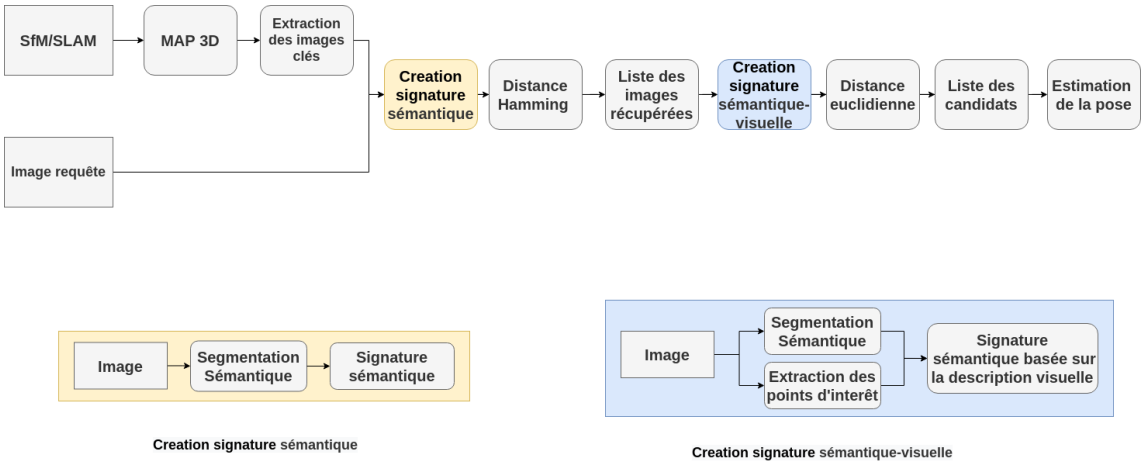


FIGURE 5.2 – L'approche globale.

Notre approche (figure 5.2) exploite la puissance discriminative des réseaux de neurones profonds basés sur la segmentation sémantique pour créer des signatures d'image de deux façons différentes, que nous expliciterons dans la description suivante de l'approche globale.

La première étape consiste à construire la carte 3D à l'aide du SLAM visuel. À partir de celle-ci, nous extrayons les caractéristiques visuelles des images (aussi bien les images clés utilisées par le SLAM, que l'image requête). Puis, nous appliquons la segmentation sémantique sur ces images afin de calculer une signature pour chacune d'entre elles. Les réseaux de segmentation sémantique produisent une carte

2D qui associe une étiquette sémantique *ie* une classe comme (mer, arbre, trottoirs, voiture, ciel) à chaque pixel (voir la figure 4.2). A partir de cette carte 2D, nous pouvons connaître les pixels de l'image qui corresponds aux classes sémantiques et leur proportion dans l'image. En d'autres termes, la signature obtenue à l'aide des réseaux de neurones est purement sémantique. Ensuite, nous utilisons la distance de Hamming pour comparer les signatures des images clés et la signature de l'image requête. Les images les plus proches sont considérées sémantiquement similaires à l'image requête, même si elles ne partagent pas nécessairement des positions spatiales proches dans la carte 3D. Pour éliminer les images sémantiquement proches, mais spatialement éloignées, et ainsi minimiser le taux d'erreur, nous proposons ensuite l'utilisation d'une deuxième signature combinant les informations sémantiques et les descripteurs visuels. Cette signature nécessite donc deux éléments : des descripteurs visuels (SIFT par exemple) pour caractérisation du voisinage spatial et la segmentation sémantique pour la proximité en termes de contenu. En combinant ces informations, nous obtenons une description plus robuste qui prend en compte à la fois l'analyse sémantique et l'agencement spatial des descripteurs visuels d'une image. Ensuite, on ne garde que les images les plus proches de l'image requête donnée en entrée selon ces deux critères. À partir de cette liste d'images candidates, on calcule la pose à partir de N correspondances entre les points 3D issus des images clés et les descripteurs 2D de l'image requête.

Dans la suite de cette partie, nous présenterons tout d'abord comment sont créées les deux signatures : la première à partir des informations sémantiques, et la deuxième combinant les informations sémantiques et visuelles.

5.2.1 Création d'une signature d'image en se basant sur les informations sémantiques

La récupération d'images basée sur le contenu (CBIR) consiste à trouver des images similaires à une image requête donnée, à partir du contenu des images. Dans notre travail, nous nous intéressons au contenu sémantique d'une image. Deux images similaires représenteront donc les mêmes objets (e.g. un chien, ou un bâtiment + un ciel + un trottoir, ou une route + des voitures + des feux de signalisation). Nous proposons pour cela d'utiliser des réseaux de segmentation sémantique basés sur des réseaux de neurones profonds, couplés à un codage spatial binaire.

La figure 5.3 montre une architecture de segmentation sémantique avec une couche supplémentaire proposée (Binary Encoding Layer). Cette couche transforme la sortie de la couche ArgMax (carte 2D) en une signature sémantique binaire.

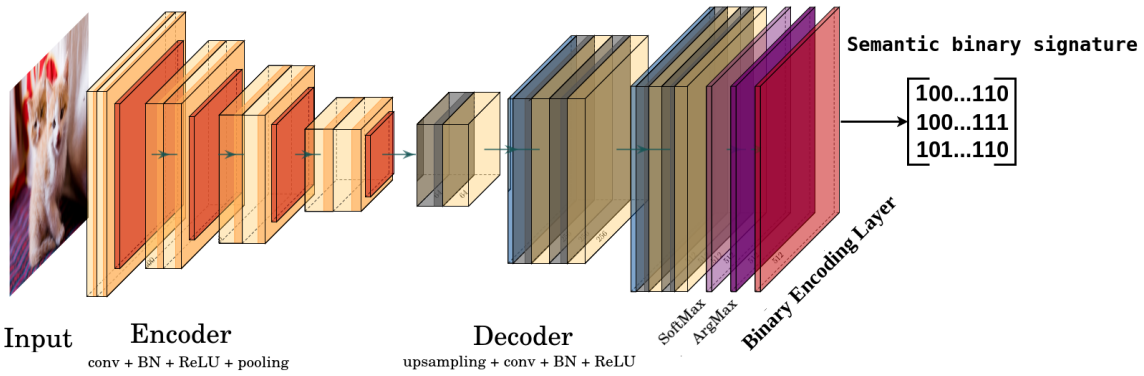


FIGURE 5.3 – Architecture de segmentation sémantique avec une couche supplémentaire.

Le codage d'une image est le processus de conversion des données de cette image dans un autre format (nombre, vecteur, matrice, labels...) pour simplifier son exploitation dans différents contextes. Dans le cas des algorithmes de récupération d'images basés sur le contenu (CBIR), l'encodage du contenu d'images offre de nombreux avantages en termes de recherche, de récupération et d'augmentation de la précision.

De nombreuses approches basées sur l'encodage telles que BoVW [29], Fisher vector encoding [82], VLAD [47], CNN [54] permettent d'obtenir d'excellentes performances. Par conséquent, le codage du contenu de l'image est un élément clé permettant d'augmenter les performances du système CBIR. Inspiré par les récents succès de l'apprentissage profond, nous proposons une nouvelle méthode d'encodage de l'image, exploitant la sortie de l'étape de segmentation sémantique. À partir d'une carte 2D sémantique, notre méthode transforme les informations sémantiques en une matrice spatiale binaire. La construction de la signature se base sur deux informations issues de la segmentation sémantique : (i) le codage des informations spatiales sémantiques dans l'image, (ii) le codage de la proportion de chaque objet sémantique dans l'image.

Comme le montre la figure 5.4, étant donné une image de requête I_q, nous avons obtenu l'image segmentée I_{seg} en utilisant l'algorithme sémantique de Sun et al [104]. Celle-ci associe à chaque pixel de l'image d'origine, une classe sémantique décrivant la nature de l'objet représenté par le pixel (ciel, mer, bâtiment, arbre, etc.). Ensuite, nous divisons l'image segmentée I_{seg} en plusieurs blocs I_{sub}. Pour chaque bloc, nous encodons les informations spatiales et de proportion des classes sémantiques dans une matrice binaire. La concaténation de ces deux informations permet d'obtenir une signature sémantique exploitable. Les deux sous-sections suivantes présentent les étapes d'encodage des informations spatiales et d'encodage des proportions des classes sémantiques dans une image.

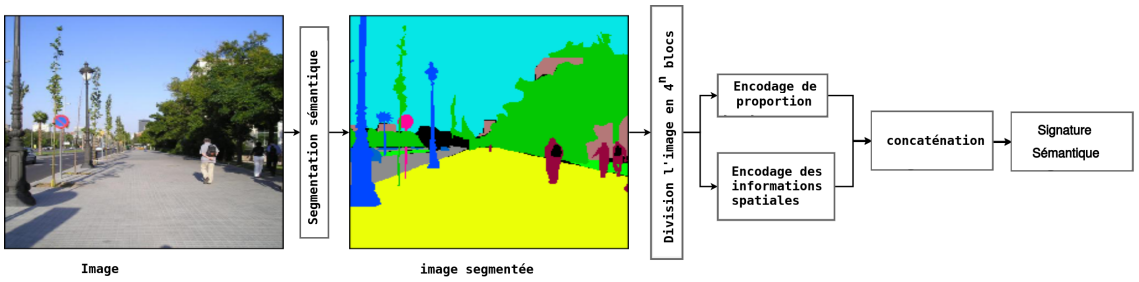


FIGURE 5.4 – Les différentes étapes pour construire une signature sémantique

5.2.1.1 Encodage des informations spatiales

Nous proposons d'encoder les informations spatiales à l'aide d'un encodage binaire multi-échelles. Pour ce faire, dans un premier temps, l'image est divisée en zones de même taille, de manière récursive. Pour le niveau 1, l'image est divisée en 2 x 2 zones spatiales qui sont désignées comme des blocs. La même opération est alors réalisée pour chaque bloc de niveau 1, puis pour chaque bloc de niveau 2, et ainsi de suite. Il en résulte que pour les niveaux L, le processus de fractionnement récursif génère un ensemble de $n_b = 4^n$ blocs. Dans un deuxième temps, un vecteur binaire est associé à chaque bloc. Ce vecteur binaire contient des informations sur les classes sémantiques existantes dans le bloc et leur proportion liée à la taille du bloc : si une classe sémantique est présente dans le bloc, on lui attribue un 1, sinon un 0. On obtient ainsi un vecteur binaire pour chaque bloc qui indique la présence ou non des classes sémantiques.

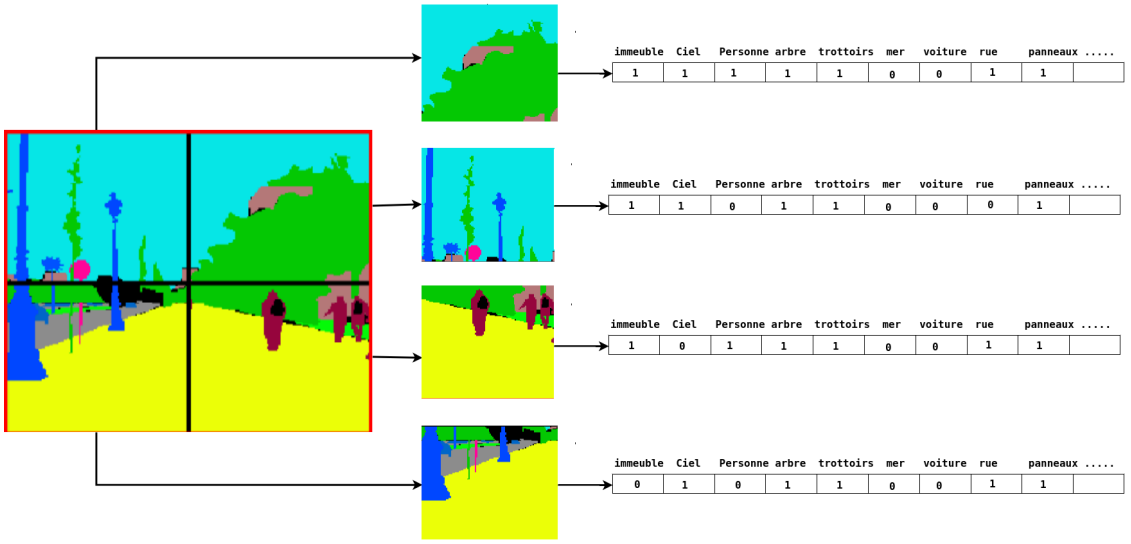


FIGURE 5.5 – Un exemple de conversion d'un bloc sémantique vers un vecteur binaire sémantique.

La figure 5.5 présente une division spatiale en quatre blocs de l'image sémantique. Un vecteur binaire est affecté à chaque bloc pour indiquer la présence des classes sémantiques. Notre exemple ici montre par la valeur 1 la présence des classes sémantiques telles que {ciel, immeuble, personne, etc.} et par 0 les classes manquantes. Le processus de création des vecteurs binaires s'arrête lorsqu'on obtient quatre vecteurs correspondant aux quatre blocs. Enfin, nous concaténons dans un ordre donnée les vecteurs binaires de tous les blocs pour obtenir la signature globale S_s d'une image d'entrée.

5.2.1.2 Encodage de proportion

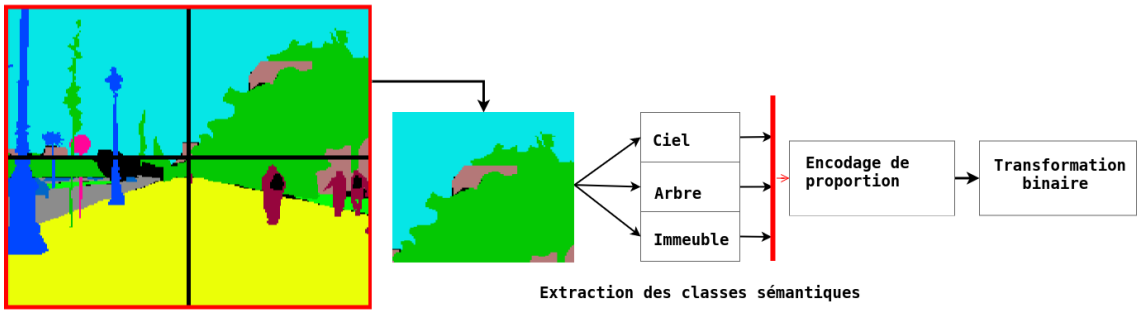


FIGURE 5.6 – Exemple d'encodage de la proportion. Avec une image divisée en 4 blocs, nous sélectionnons itérativement chaque bloc pour calculer la proportion de la classe sémantique à l'intérieur.

Dans la deuxième étape, nous complétons la représentation spatiale binaire avec des informations sur la proportion de chaque classe sémantique. Pour ce faire, nous proposons d'encoder la proportion des classes sémantiques à partir de l'image segmentée en utilisant la même division spatiale utilisée lors de l'encodage des informations spatiales. Étant donné une image segmentée I_{seg}, nous détectons les classes sémantiques présentes dans chaque bloc à l'aide du réseau de neurones. Ensuite, pour chaque classe sémantique C on calcule sa proportion en pourcentage P_c dans le bloc. Après avoir attribué les pourcentages de toutes les classes, un processus de conversion binaire sera appliqué à chaque P_c indiqué dans l'équation (5.1) afin de créer une signature binaire par bloc nommée B_{Pc}.

$$\begin{cases} \text{if} \quad 0 < P_c <= 0.25 \quad B_{Pc} = 0001 \\ \text{if} \quad 0.25 < P_c <= 0.5 \quad B_{Pc} = 0011 \\ \text{if} \quad 0.5 < P_c <= 0.75 \quad B_{Pc} = 0111 \\ \text{if} \quad P_c > 0.75 \quad B_{Pc} = 1111 \end{cases} \tag{5.1}$$

Pour les cas où la classe sémantique C_i n'est pas présente dans le bloc, une affectation lui est automatiquement attribuée dans le bloc $B_{Pc} = [0000]$. Afin de conserver tous

les scores, nous les collectons ensemble dans la matrice B_{Sub-Pc}. Cette matrice est une description binaire des propriétés des différents classes présentent dans un bloc.

$$B_{Sub-Pci} = \begin{bmatrix} B_{Pc1} \\ B_{Pc2} \\ ..... \\ B_{PcM-1} \\ B_{PcM} \end{bmatrix}$$

où M est le nombre de classes que le réseau a appris à détecter. Enfin, nous concaténons toutes les conversions binaires $B_{Sub-Pci}$ pour obtenir une signature de proportion globale S_P correspondant à l'image en entrée segmentée I_{seg} où S_P=$\{B_{Sub-Pc_1}.B_{Sub-Pc_2}.....B_{Sub-Pc_M}\}$.
Nous commençons les tests par de grands blocs, puis nous les réitérons avec des blocs de plus en plus petits. Lorsque $n_b = 1$ cela signifie qu'aucune division spatiale n'était pas appliquée sur l'image. Par conséquent, nous encodons uniquement les informations de proportion sémantique B_{Sub-Pc}.

5.2.2 Création d'une signature d'image en se basant sur la combinaison des informations sémantique et visuelle

Nous présentons dans cette section une nouvelle idée pour construire une signature d'image. Cette signature est appliquée sur l'ensemble d'images fourni par la signature précédente. Cette liste d'images qui est sémantiquement similaire à l'image requêtée, est ensuite filtrée pour garder seulement les plus proches spatialement.

La signature combine les informations issues de l'analyse sémantique et celles issues de l'utilisation des descripteurs visuels. La figure 5.7 présente les différentes étapes de la construction. Notre approche est composée de trois étapes : (i) détection et extraction des caractéristiques visuelles, (ii) extraction d'informations sémantiques, (iii) regroupement des points clés par classe (étiquettes) puis on calcule leurs centres.

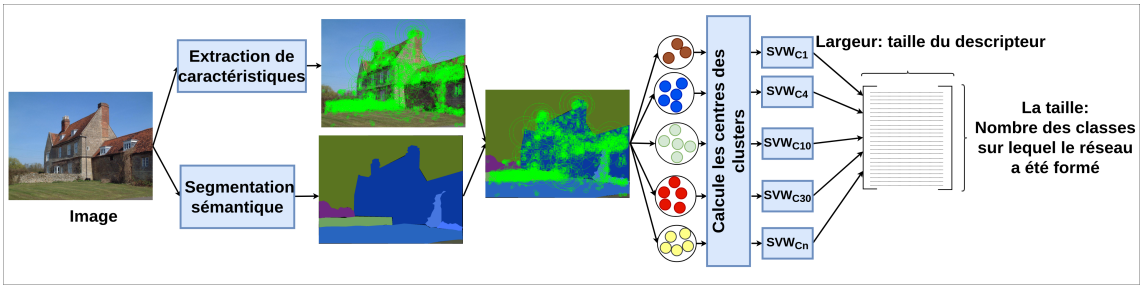


FIGURE 5.7 – Les différentes étapes de la création de la signature d'image

Nous commençons par détecter les points clés dans une image à l'aide du détecteur de Harris [32]. Pour cela, nous extrayons leurs descripteurs de type SURF [19].

En parallèle, nous appliquons la segmentation sémantique sur l'image pour obtenir une image segmentée sémantiquement, c'est-à-dire une carte 2D qui associe chaque pixel à une étiquette sémantique. Ensuite, on projette les coordonnées (x,y) des points clés détectés dans l'image d'origine sur l'image segmentée sémantiquement. Nous obtenons ainsi pour chaque point clé : un descripteur visuel et un label sémantique. Nous pouvons dès lors regrouper les points clés par classe sémantique puis calculer les centres des classes. Pour cela, nous sélectionnons l'ensemble des points-clés qui appartiennent à une même classe sémantique et nous appliquons l'algorithme de clustering (K-MEANS) avec $K = 1$ (la moyenne d'un centre). Chaque classe est alors représentée par un vecteur numérique de H réels correspondant à la taille du descripteur. Le choix de $K = 1$ ici est pour obliger l'algorithme K-MEANS à renvoyer la moyenne des points clés par groupe.

Afin d'appliquer la segmentation en utilisant K-MEANS sur tous les groupes des points clés, la signature de l'image sera composée de M centres sémantique visuels $SVW_{1..M}$ qui représentent les classes sémantiques existantes dans l'image. Il est peu probable que l'image contienne toutes les classes sémantiques que le réseau de neurones a appris à reconnaître. Nous attribuons donc un vecteur nul aux classes non détectées dans l'image. Au final, nous obtenons une signature sous la forme d'une matrice de H * M réels où la largeur H correspond à la taille du descripteur (SURF 64) et la hauteur M correspond au nombre de classes sur lesquelles le réseau a été entraîné.

5.3 Étude expérimentale

5.3.1 Base des données pour la segmentation sémantique

De nombreuses bases de données pour la segmentation sémantique ont été proposées ces dernières années telles que [27], Mapillary [76], COCO [61], ADE20K [127], Coco-Stuff [20], Mseg [56]. Les bases de données sémantiques sont divisées en deux catégories principales. Les objets (Things) qui ont des caractéristiques physiologiques comme véhicule, chien, ordinateur, etc. et les objets (Stuff) qui décrivent des objets amorphes comme la mer, le ciel, l'arbre, etc. Par conséquent, les ensembles de données de segmentation sémantique sont divisés en trois catégories principales : (i) Stuff (ii) Things (iii) Stuff Things. Pour obtenir une prédiction robuste, nous utilisons l'architecture récente HRNetW48 [104] entraînée sur les base de données Coco-stuff [20] et Mseg [56]. Le principal avantage de l'utilisation des ensembles de données Coco-Stuff et Mseg est qu'ils sont capables de reconnaître les objets : Stuff Things avec un nombre élevé de classes connues pour une image donnée.

5.3.1.1 CocoStuff

La figure 5.8 présente plusieurs images annotées sur la base de données COCOStuff. Cette base de données contient 164K images dont 120K pour entraîner le

réseaux, 5K pour la validation, 20k pour les tests et 20K test-challenge. Sémantiquement CoCo-Stuff couvre 172 classes : 80 classes d'objets Stuff, 91 classes d'objets Things et 1 classe sans étiquette.

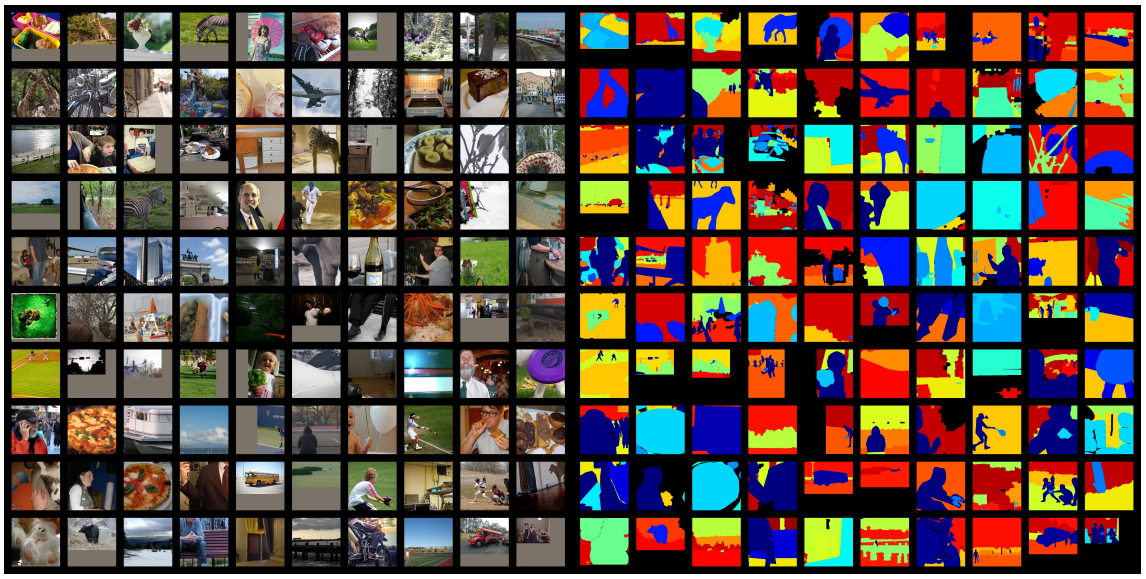


FIGURE 5.8 – Exemples d'images de la base de donnée Coco-Stuff [20] et les images segmentées.

5.3.1.2 Mseg

MSeg est une base de données pour la segmentation sémantique composée par plusieurs bases Cityscapes [27], Mapillary [76], COCO [61], ADE20K [127], Coco-Stuff [20]. L'ensemble de données produit environ 200K images avec 316 classes sémantiques (après la fusion des classes avec des noms synonymes).

FIGURE 5.9 – Exemples d'images de la base de donnée Mseg [56] et les images segmentées.

5.3.2 Résultats

Dans cette section nous allons montrer l'efficacité de notre approche sur deux bases de données (Rome, Dubrovnik) de localisation à large échelle. Notre approche commence par la segmentation sémantique de toutes les images clés et images requêtes en utilisant les réseaux de neurones HRNet-W48 [104] entraînés sur les bases des données Coco-stuff et Mseg.

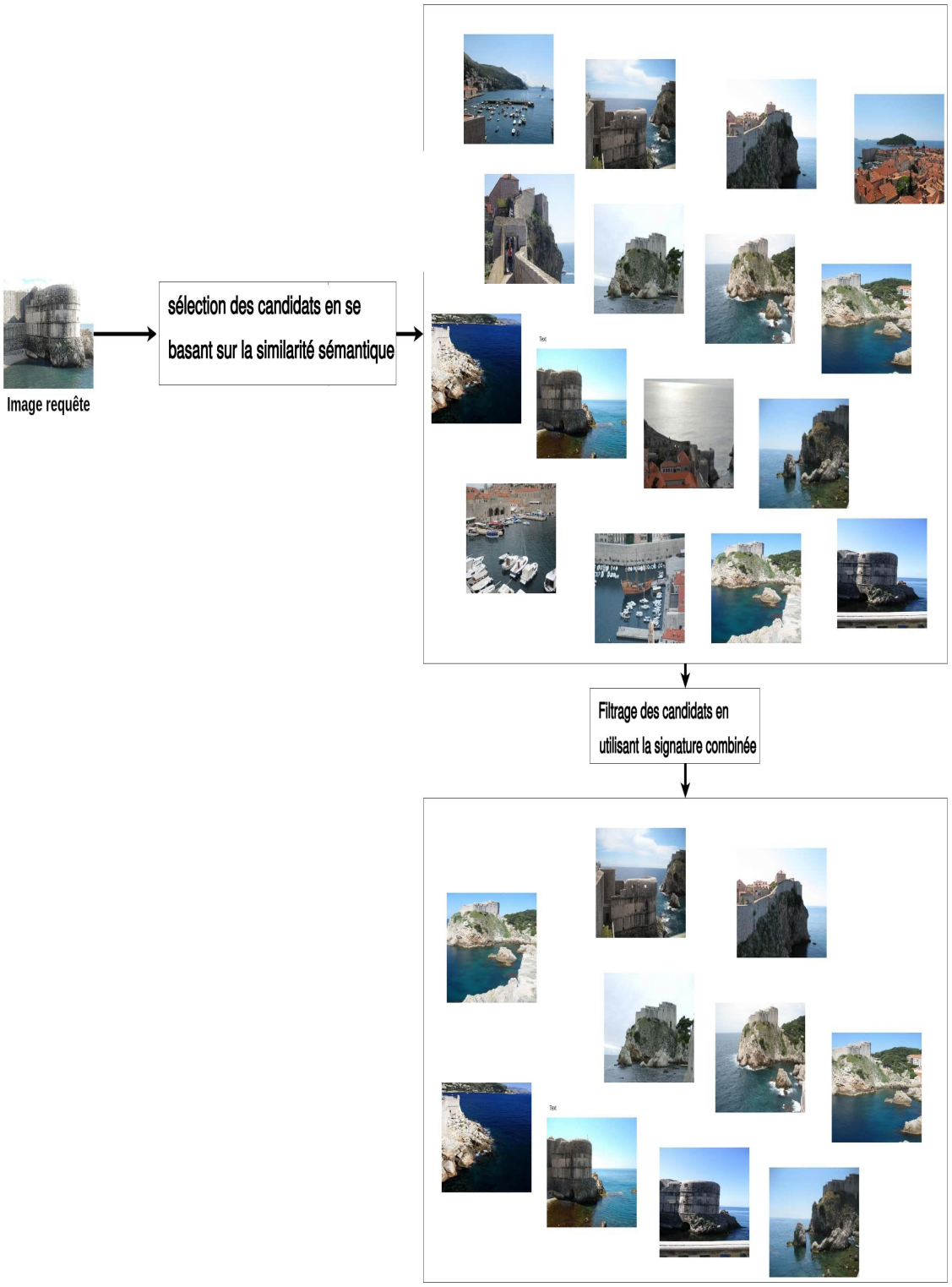


FIGURE 5.10 – Exemples de la liste d'images obtenue après l'utilisation des deux signatures.

Nous montrons par un exemple comment se réalisent la sélection et le filtrage des images par notre approche. Dans la figure 5.10, nous présentons une image requête qui contient trois classes sémantiques : mer, immeuble et ciel. Dans un premier temps, nous sélectionnons depuis la map 3D toutes les images clés qui partagent les mêmes informations sémantiques que l'image requête en utilisant la signature sémantique.

Pour choisir les images les plus proches parmi toutes celles considérées similaires d'un point de vue sémantique, nous appliquons un deuxième filtre sur les images retenues en utilisant la signature basée sur la fusion des données sémantiques et spatiales (à partir des descripteurs visuels). En résultat, nous obtenons une liste d'images, proches sémantiquement et spatialement à l'aide de laquelle l'estimation de la pose est calculée. Cette pose est calculée en extrayant les points 3D correspondant aux descripteurs de l'image requête, puis en appliquant l'algorithme P3P pour estimer la pose.

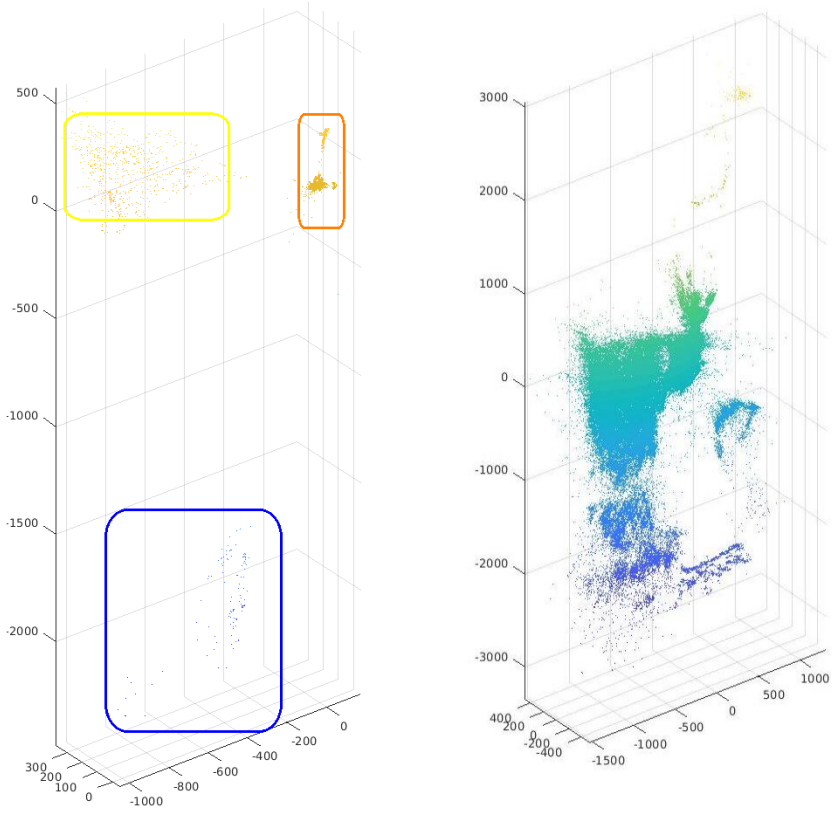


FIGURE 5.11 – Nuage des points 3D extrait à partir des 100 images les plus proches de l'image requête en utilisant la signature sémantique.

Dans la figure 5.11, nous présentons l'image requête et le nuage des points 3D issu des 100 images clés les plus proches du point de vue sémantique et la carte 3D globale de la base de données Dubrovnik. Rappelons que la signature sert à sélectionner les images qui sont sémantiquement proche de l'image requête donnée. Les résultats sur la figure montrent trois endroits spatialement distincts. Notre but par la suite est de ne garder que le nuage de points 3D correspondant à l'image requête et d'éliminer les points 3D qui n'ont pas de correspondances visuelles avec l'image requête.

Pour ce faire, nous avons filtré l'ensemble des points 3D en supprimant les images non pertinentes à l'aide de la signature combinée exploitant la sémantique et les critères visuels (voir figure 5.12). Au final, nous parvenons à localiser l'image requête dans la carte 3D en appliquant notre approche.

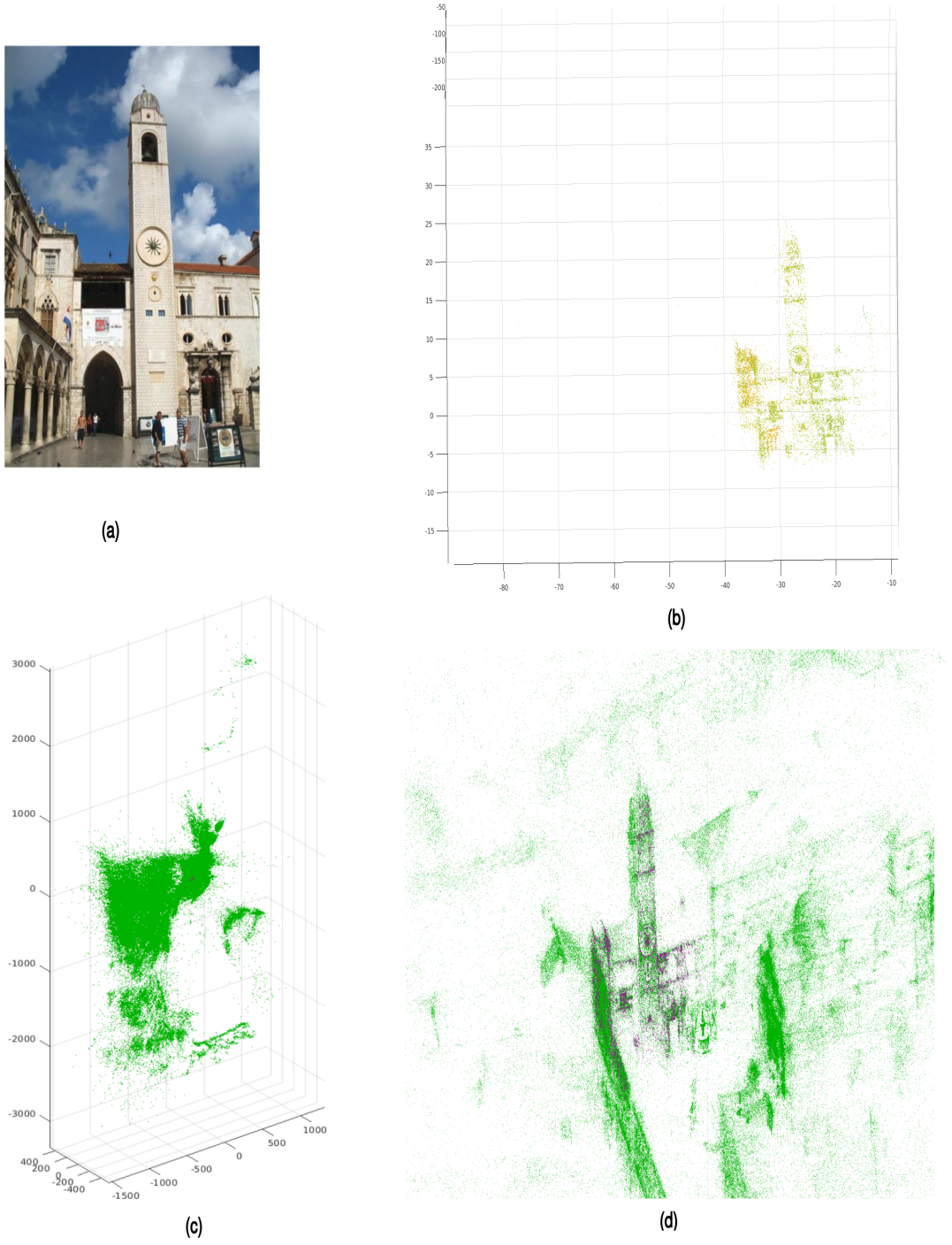


FIGURE 5.12 – (a) image requête. (b) nuage des points 3D après le filtrage par la signature sémantique visuelle. (c) nuage de points 3D de la carte globale (Dubrovnik). (d) projection de points 3D de l'image requête sur la carte globale.

Méthodes	# reg images	Q1 (m)	Q2 (m)	Q3 (m)	images avec erreur < 18.3 m	images avec erreur >400 m	temps (s)
Sans information a priori							
Méthode proposée (Mseg)	796	**0.33**	**0.88**	2.47	728	10	0.22
Méthode proposée (Coco-Stuff)	793	0.45	0.97	2.77	720	12	**0.17**
Sattler [91]	783	0.40	1.40	5.90	685	16	0.31
Sattler [92]	795	0.40	1.40	5.30	704	**9**	0.25
Raul [34]	**800**	1.09	7.92	27.76	550	10	0.62
Youji [37]	781	0.42	1.28	4.67	-	12	0.46
Li [60]	753	7.50	9.30	13.40	655	-	-
Choudhary [25]	788	0.88	3.10	11.83	-	-	-
Avec information a priori							
Zeisl [122]	796	**0.19**	**0.56**	**2.09**	744	7	**3.78**
Svarm [106]	**798**	-	0.56	-	**771**	3	5.06

TABLE 5.1 – Comparaison de notre approche avec des méthodes de l'état de l'art pour Dubrovnik. Q1, Q2 et Q3 sont les quartiles de l'erreur de localisation.

Méthodes	# reg images	inliers	temps
Sattler [91]	977	100	0.29
Sattler [92]	991	200	0.28
Youji [37]	985	-	0.33
Li [60]	924	-	0.87
Méthode proposée (Mseg)	**996**	**366**	0.25
Méthode proposée (Coco-Stuff)	992	321	**0.21**

TABLE 5.2 – Comparaison de notre approche avec des méthodes de l'état de l'art pour Rome (Mseg Dataset).

La segmentation sémantique joue un rôle prépondérant dans notre système de localisation. Pour obtenir une segmentation parfaite, nous avons utilisé les bases de données (Mseg, Coco-Stuff). Ces deux bases nous offrent une segmentation avec un grand nombre de classes sémantiques. Par conséquent, nous obtenons une description sémantique pertinente pour les images clés et les images requêtes. Les résultats obtenus avec la base sémantique Mseg sont meilleurs en termes de performance que ceux obtenus avec la base CoCo-Stuff. Dans les tableaux 5.1, 5.2 nous comparons nos résultats à l'état de l'art sur deux bases de données de grande échelle (Dubrovnik, Rome). La comparaison est ici effectuée avec deux méthodes différentes : les méthodes utilisant des informations a priori [122],[106] (direction verticale et hauteur

connue) et les méthodes sans information a priori [60] [37] [92] [91] [34] [25]. Les méthodes qui utilisent des informations a priori présentent de meilleurs résultats pour des seuils correspondant à l'erreur inférieure à 18,3 m et supérieure à 400 m. Dans ce travail, nous nous sommes intéressés à comparer ces résultats sans information a priori à l'état de l'art. Notre méthode qui n'utilise pas d'informations a priori réussit à localiser un grand nombre de poses avec une erreur inférieure à 18,3 m. Nos résultats sont meilleurs que toutes les autres méthodes sans informations supplémentaires en terme de précision.

Même si les résultats obtenus dans le chapitre IV sont meilleurs en terme de précision, l'approche proposée ici est capable de récupérer la pose dans un environnement de large échelle plus rapidement. L'avantage principal est que la comparaison entre l'image requête et les images clés sont basées sur la similarité sémantique par la signature décrite en 5.1.2. ce qui permet un gain de temps pour notre système de localisation.

5.4 Conclusion

Dans ce chapitre, nous avons présenté une méthode indirecte pour améliorer l'estimation de la pose. Notre approche est basée sur l'utilisation conjointe de la segmentation sémantique de l'image basée sur l'apprentissage profond, et les caractéristiques spatiales et visuelles de l'image (descripteurs SURF).

La méthode indirecte est composée de trois étapes principales : (1) chercher les images clés les plus similaires à l'image requête donnée en entrée ; (2) chercher les correspondances 2D-3D entre l'image requête et les images clés ; (3) calculer la pose en utilisant l'algorithme PnP. Dans ce travail, nous nous sommes concentrés sur l'amélioration de la première étape dans la méthode indirecte. Cette étape consiste à chercher dans l'ensemble des images clés utilisées pour construire la carte 3D les images qui sont comme considérées similaires à l'image requête. Deux propositions dans le contexte de recherche d'image par similarité étaient proposées pour aider le système de localisation à trouver sa position avec un maximum de précision. La première signature a été créée en utilisant les réseaux de neurones convolutionnels dans le but de sélectionner la zone où les images clés partagent les mêmes informations sémantiques que l'image requête. La deuxième signature a été obtenue par la combinaison des informations sémantiques avec les descripteurs visuels. Le but de la deuxième signature est de filtrer la liste d'images retenues. Puis, nous avons cherché les correspondances 2D-3D entre l'image requête et les image candidates. Enfin, la pose est estimée par l'algorithme P3P après la suppression des correspondances aberrantes par l'algorithme RANSAC.

L'information sémantique dans ce chapitre était obtenue à partir de deux différentes bases de données sémantiques (Mseg, Coco-Stuff). En exploitant ces informations, nous avons montré que l'utilisation de la sémantique dans le contexte de la localisation augmente la précision des poses récupérées et rend le système de locali-

sation plus robuste. Comparée à l'état de l'art, notre approche permet d'obtenir de meilleurs résultats en termes de précision.

L'analyse des résultats obtenus dans ce chapitre a permis la mise en évidence de plusieurs limitations. Nous citons en particulier la limite de technique de la recherche d'images par le contenu. Le résultat de cet algorithme est une liste d'images considérées comme similaires à l'image requête donnée en entrée. Mais, deux images proches sémantiquement ne partagent pas forcément des positions spatiales proches. Par exemple deux images qui ont les mêmes contenus sémantiques (immeuble, rue) peuvent être sémantiquement similaires mais représenter deux endroits différents. Ces limitations peuvent être contournées en utilisant des informations supplémentaires. En particulier, la liste de visibilité obtenue par l'algorithme SfM est un moyen efficace pour vérifier la relation spatiale entre les images dans la carte 3D. Donc, à partir de cette liste nous pouvons regrouper spatialement l'ensemble des images candidates. Puis, nous comparons les descripteurs 2D de l'image requête avec les points 3D dans chaque groupe. Ensuite, nous pouvons obtenir comme résultat une pose pour chaque groupe d'images. Enfin, une étape de filtrage sur l'ensemble des poses trouvées permet de sélectionner la pose qui a le nombre d'appariements le plus important.

Chapitre 6

Conclusion et Perspectives

La localisation visuelle basée sur l'image est un domaine très actif depuis plusieurs décennies dans lesquelles plusieurs solutions existaient pour résoudre le problème du positionnement d'un robot dans un environnement connu. Notre objectif dans cette thèse était d'étudier ces approches puis de proposer des nouveaux algorithmes pour rendre le système de localisation plus robuste.

Dans le premier chapitre, nous avons commencé par une étude bibliographique qui nous a permis de poser le problème de la reconstruction 3D et des technologies associées. Ici, nous parlons principalement des deux approches SfM et SLAM. Puis, nous avons étudié les approches proposées dans le contexte de la localisation visuelle basée sur l'image. En effet, de nombreux algorithmes proposés sont principalement basés sur deux approches principales : méthodes directes et méthodes indirectes. En comparant ces deux méthodes, les méthodes directes sont efficaces dans un environnement de taille limité, par contre les méthodes indirectes sont plus robustes dans un environnement de large échelle. Ces algorithmes utilisent principalement les descripteurs visuels 2D extraits à partir d'une image requêtes pour réaliser des mises en correspondances avec les descripteurs des points 3D de la carte globale. Malgré les succès des approches basées sur les descripteurs visuels (SIFT par exemple), d'autres alternatives récemment publiées sont basées sur l'apprentissage profond et les arbres de régression. Dans la dernière partie de l'état de l'art, nous avons présenté les réseaux de neurones convolutionnels appliqués à la segmentation sémantique, leurs architectures et les applications liées au contexte de localisation.

Dans le second chapitre, nous avons exploité une information supplémentaire définie par la hauteur des points-clés par rapport au sol Z pour rendre le système de localisation plus performant. Cette information est obtenue par triangulation sur une paire d'images requête et par reconstruction 3D en utilisant l'algorithme SfM ou SLAM pour les images clés. Nous avons testé deux utilisations principales de l'information Z dans notre thèse : (1) augmenter le nombre des vrais appariements

entre les descripteurs 2D et les points 3D par l'intégration de Z lors de la phase de correspondance ; (2) augmenter la robustesse des vocabulaires en ajoutant l'information Z à la phase de la construction des mots visuels. Notre contribution est valable uniquement lorsque la direction verticale est connue et la hauteur de la caméra est fixe. Nous avons testé notre approche sur deux bases de données différentes correspondantes à un environnement d'intérieur (musée) et à un environnement urbain (Oxford). Les expérimentations montrent la performance de notre approche en terme de temps et précision.

Aussi dans le troisième chapitre, nous avons exploité une information supplémentaire extraite par l'apprentissage profond. Cette information est obtenue en appliquant un réseau de neurones convolutionnel de segmentation sémantique sur une image. La sortie de ce réseau est une carte 2D qui conserve généralement la même taille que l'image en entrée. Pour chaque élément dans la carte 2D une étiquette correspond à un pixel dans l'image. En plus, chaque étiquette est caractérisée par un vecteur nommé SoftMax. Ce vecteur de taille N correspond au nombre de classes sur lesquelles le réseau a été entraîné et leurs probabilités. En utilisant ce vecteur nous avons pu affecter aux points-clés qui se retrouvent aux frontières sémantiques au moins deux classes distinctes (label). Une autre idée a été proposée dans cette thèse une approche basée sur la concaténation du vecteur de probabilités et le descripteur 2D. Par conséquent, la description finale d'un point clé est obtenu par la combinaison des informations sémantiques avec les descripteurs visuels. L'utilisation des GPU nous a permis d'accélérer le temps d'exécution. La base de données utilisée pour segmenter les images est Ade20k. Cette base est capable de fournir 150 classes sémantiques au maximum par image. La puissance discriminative de la combinaison entre la sémantique et le descripteur visuel nous a donné une bonne performance en terme de précision. Les résultats ont été établis sur les deux bases de données publiques de localisation Rome et Dubrovnik.

Dans le dernier chapitre, nous avons pensé à améliorer la méthode indirecte par un algorithme puissant capable de chercher puis trouver les images clés les plus semblables à image requête fournie par l'utilisateur. En rappelant que la méthode indirecte est basée sur la recherche d'images clés de la carte 3D puis sur la comparaison de leurs descripteurs 3D avec les descripteurs 2D de l'image requête pour calculer la pose. Notre proposition est divisée en deux parties principales : (1) identifier la zone de visibilité de l'image requête dans la carte 3D en se basant sur son contenu sémantique (2) filtrer les images clés par zone puis sélectionner les plus pertinentes pour passer à la phase de correspondance 2D-3D. Les deux signatures proposées sont basées sur la segmentation sémantique. La première signature est composée de deux éléments principaux : codage binaire et l'information sémantique. La deuxième signature est une combinaison entre les descripteurs visuels et l'information sémantique. La performance de notre approche a été testée sur deux bases de données de localisation (Rome, Dubrovnik).

Ces travaux de recherche et les différentes thématiques dans ce travail de thèse ouvrent différentes perspectives de recherche.

Aujourd'hui, l'apprentissage profond est une technologie prometteuse pour la vision par ordinateur et la tendance dans les domaines suivant : la reconnaissance automatique, la robotique, le suivi, etc. Ces algorithmes présentent des alternatives plus performantes que les approches classiques. Liées au contexte de la thèse, ces approches sont capables de réaliser les tâches suivantes :

— Détection puis extraction des points clés
— Reconstruction 3D
— Estimation de la pose
— Segmentation sémantique de la carte 3D
— Similarité entre deux images

Dans la littérature, les points clés sont détectées dans l'image par un détecteur comme Harris ou Hessien. Puis, l'extraction est faite en utilisant un descripteur visuel comme SIFT. La même tâche aujourd'hui peut être effectuée par des approches basées sur l'apprentissage profond avec plus de robustesse aux changement d'échelle, d'orientation, et à des modifications de l'éclairage. Par conséquent, l'obtention des vrais appariement (inliers) 2D-3D ou 3D-2D sont plus probables avec ces nouveaux descripteurs. Cela influe positivement sur la précision de la pose à récupérer. De manière identique pour la construction 3D, les réseaux de neurones conventionnels sont capables de construire une carte 3D comme celles qui sont construites par les algorithmes : SLAM et SfM. Le réseau prend en entrée des images RGB et donne à la sortie un nuage de points 3D. Chaque point 3D de la carte est caractérisé par ses coordonnées 3D et une classe label. Donc, en exploitant les réseaux de neurones profonds pour obtenir une carte labélisée et des descripteurs robustes, on peut espérer augmenter la précision de la pose et rendre le système de localisation beaucoup plus robuste aux problèmes tels que les structures répétitives et les changements mineurs dans l'image.

L'inconvénient principal ici est que la majorité des bibliothèques des réseaux de neurones profond tel que Pytorch, tensorflow sont implémentées pour des GPUs. C'est un processeur graphique coûteux qui effectue des calculs mathématiques rapides, principalement pour le rendu d'images. La majorité des systèmes tels que les voitures autonomes, les casques de la réalité virtuelle/augmentée intègrent dans leur circuit les GPUs. Sachant que, il existe parfois des systèmes compatibles avec les CPUs pour des taches de faible complexité mais d'accès limité et pose parfois des problèmes de compatibilités.

La thématique d'estimation de la pose reste un sujet de recherche à approfondir dans le domaine de la vision par ordinateur car il existe de nombreux défis à résoudre. Même si les approches proposées dans cette thèse permettent d'améliorer l'estimation de la pose, il nous reste toujours plusieurs pas à franchir pour réussir à obtenir une pose idéale en termes de temps de calcul et de précision.

Bibliographie

[1] www.analyticsvidhya.com/blog/2020/02/learn-image-classification-cnn-convolutional-neural-networks-3-datasets/.

[2] www.becominghuman.ai/computer-vision-applications-in-self-driving-cars-610561e14118.

[3] www.e-consystems.com/3d-usb-stereo-camera.asp ,.

[4] www.fr.mathworks.com/matlabcentral/fileexchange/26649-kdtree-implementation-in-matlab.

[5] www.fr.mathworks.com/matlabcentral/fileexchange/62232-bag-of-visual-words.

[6] www.fr.wikipedia.org/wiki/epipolar.

[7] www.medium.com/@garimanishad/thank-you-for-pointing-that-out-c048deed0380.

[8] www.pinterest.com/janicedrew/collages/.

[9] www.researchgate.net/profile/marcpollefeys.

[10] www.theia-sfm.org/sfm.html.

[11] www.wikiagri.fr/articles/agriculture-de-precision-agriculture-du-futur/20066.

[12] David Acuna, Amlan Kar, and Sanja Fidler. Devil is in the edges : Learning semantic boundaries from noisy annotations. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 11075–11083, 2019.

[13] Sameer Agarwal, Yasutaka Furukawa, Noah Snavely, Ian Simon, Brian Ctitleess, Steven M Seitz, and Richard Szeliski. Building rome in a day. *Communications of the ACM*, 54(10) :105–112, 2011.

[14] Pablo Fernández Alcantarilla, Adrien Bartoli, and Andrew J Davison. Kaze features. In *European Conference on Computer Vision*, pages 214–227. Springer, 2012.

[15] Relja Arandjelovic, Petr Gronat, Akihiko Torii, Tomas Pajdla, and Josef Sivic. Netvlad : Cnn architecture for weakly supervised place recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5297–5307, 2016.

[16] Clemens Arth, Daniel Wagner, Manfred Klopschitz, Arnold Irschara, and Dieter Schmalstieg. Wide area localization on mobile phones. In *2009 8th ieee international symposium on mixed and augmented reality*, pages 73–82. IEEE, 2009.

[17] Mathieu Aubry, Bryan C Russell, and Josef Sivic. Painting-to-3d model alignment via discriminative visual elements. *ACM Transactions on Graphics (ToG)*, 33(2) :14, 2014.

[18] Vijay Badrinarayanan, Alex Kendall, and 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.

[19] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf : Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.

[20] Holger Caesar, Jasper Uijlings, and Vittorio Ferrari. Coco-stuff : Thing and stuff classes in context. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1209–1218, 2018.

[21] Song Cao and Noah Snavely. Graph-based discriminative learning for location recognition. In *Proceedings of the ieee conference on computer vision and pattern recognition*, pages 700–707, 2013.

[22] David Caruso, Jakob Engel, and Daniel Cremers. Large-scale direct slam for omnidirectional cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 141–148. IEEE, 2015.

[23] Tommaso Cavallari, Stuart Golodetz, Nicholas A Lord, Julien Valentin, Luigi Di Stefano, and Philip HS Torr. On-the-fly adaptation of regression forests for online camera relocalisation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4457–4466, 2017.

[24] Jacob Chan, Jimmy Addison Lee, and Kemao Qian. F-sort : An alternative for faster geometric verification. In *Asian Conference on Computer Vision*, pages 385–399. Springer, 2016.

[25] Siddharth Choudhary and PJ Narayanan. Visibility probability structure from sfm datasets and applications. In *European conference on computer vision*, pages 130–143. Springer, 2012.

[26] Ondrej Chum and Jiri Matas. Matching with prosac-progressive sample consensus. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 220–226. IEEE, 2005.

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

[28] David Crandall, Andrew Owens, Noah Snavely, and Dan Huttenlocher. Discrete-continuous optimization for large-scale structure from motion. In *CVPR 2011*, pages 3001–3008. IEEE, 2011.

[29] Gabriella Csurka, Christopher Dance, Lixin Fan, Jutta Willamowski, and Cédric Bray. Visual categorization with bags of keypoints. In *Workshop on statistical learning in computer vision, ECCV*, volume 1, pages 1–2. Prague, 2004.

[30] Andrew J Davison, Ian D Reid, Nicholas D Molton, and Olivier Stasse. Monoslam : Real-time single camera slam. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (6) :1052–1067, 2007.

[31] Frank Dellaert, Steven M Seitz, Charles E Thorpe, and Sebastian Thrun. Structure from motion without correspondence. In *Proceedings IEEE Conference on Computer Vision and Pattern Recognition. CVPR 2000 (Cat. No. PR00662)*, volume 2, pages 557–564. IEEE, 2000.

[32] Konstantinos G Derpanis. The harris corner detector. *York University*, pages 2–3, 2004.

[33] Daniel DeTone, Tomasz Malisiewicz, and Andrew Rabinovich. Superpoint : Self-supervised interest point detection and description. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 224–236, 2018.

[34] Raúl Díaz and Charless C Fowlkes. Cluster-wise ratio tests for fast camera localization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, pages 19–28, 2017.

[35] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam : Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.

[36] Jakob Engel, Jörg Stückler, and Daniel Cremers. Large-scale direct slam with stereo cameras. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1935–1942. IEEE, 2015.

[37] Youji Feng, Lixin Fan, and Yihong Wu. Fast localization in large-scale environments using supervised indexing of binary features. *IEEE Transactions on Image Processing*, 25(1) :343–358, 2015.

[38] Martin A Fischler and Robert C Bolles. Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6) :381–395, 1981.

[39] Swarnendu Ghosh, Nibaran Das, Ishita Das, and Ujjwal Maulik. Understanding deep learning techniques for image segmentation. *ACM Computing Surveys (CSUR)*, 52(4) :1–35, 2019.

[40] Ben Glocker, Jamie Shotton, Antonio Criminisi, and Shahram Izadi. Real-time rgb-d camera relocalization via randomized ferns for keyframe encoding. *IEEE transactions on visualization and computer graphics*, 21(5) :571–583, 2014.

[41] Albert Gordo, Jon Almazan, Jerome Revaud, and Diane Larlus. End-to-end learning of deep visual representations for image retrieval. *International Journal of Computer Vision*, 124(2) :237–254, 2017.

[42] Abner Guzman-Rivera, Pushmeet Kohli, Ben Glocker, Jamie Shotton, Toby Sharp, Andrew Fitzgibbon, and Shahram Izadi. Multi-output learning for camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1114–1121, 2014.

[43] F Walch1 C Hazirbas, L Leal-Taixé1 T Sattler, S Hilsenbeck, and D Cremers. Image-based localization with spatial lstms.

[44] Yuan Hu, Yunpeng Chen, Xiang Li, and Jiashi Feng. Dynamic feature fusion for semantic edge detection. *arXiv preprint arXiv :1902.09104*, 2019.

[45] Forrest N Iandola, Song Han, Matthew W Moskewicz, Khalid Ashraf, William J Dally, and Kurt Keutzer. Squeezenet : Alexnet-level accuracy with 50x fewer parameters and< 0.5 mb model size. *arXiv preprint arXiv :1602.07360*, 2016.

[46] Arnold Irschara, Christopher Zach, Jan-Michael Frahm, and Horst Bischof. From structure-from-motion point clouds to fast location recognition. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 2599–2606. IEEE, 2009.

[47] Hervé Jégou, Matthijs Douze, Cordelia Schmid, and Patrick Pérez. Aggregating local descriptors into a compact image representation. In *2010 IEEE computer society conference on computer vision and pattern recognition*, pages 3304–3311. IEEE, 2010.

[48] Hailin Jin, Paolo Favaro, and Stefano Soatto. A semi-direct approach to structure from motion. *The Visual Computer*, 19(6) :377–394, 2003.

[49] Hyo Jin Kim, Enrique Dunn, and Jan-Michael Frahm. Predicting good features for image geo-localization using per-bundle vlad. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 1170–1178, 2015.

[50] Alex Kendall, Matthew Grimes, and Roberto Cipolla. Posenet : A convolutional network for real-time 6-dof camera relocalization. In *Proceedings of the IEEE international conference on computer vision*, pages 2938–2946, 2015.

[51] Khalil Khan, Massimo Mauro, and Riccardo Leonardi. Multi-class semantic segmentation of faces. In *2015 IEEE International Conference on Image Processing (ICIP)*, pages 827–831. IEEE, 2015.

[52] Bongjoe Kim, Hunjae Yoo, and Kwanghoon Sohn. Exact order based feature descriptor for illumination robust image matching. *Pattern Recognition*, 46(12) :3268–3278, 2013.

[53] Georg Klein and David Murray. Improving the agility of keyframe-based slam. In *European Conference on Computer Vision*, pages 802–815. Springer, 2008.

[54] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[55] John Lafferty, Andrew McCallum, and Fernando CN Pereira. Conditional random fields : Probabilistic models for segmenting and labeling sequence data. 2001.

[56] John Lambert, Liu Zhuang, Ozan Sener, James Hays, and Vladlen Koltun. MSeg : A composite dataset for multi-domain semantic segmentation. In *Computer Vision and Pattern Recognition (CVPR)*, 2020.

[57] Mans Larsson, Erik Stenborg, Lars Hammarstrand, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. A cross-season correspondence dataset for robust semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 9532–9542, 2019.

[58] Jianfeng Li, Xiaowei Wang, and Shigang Li. Spherical-model-based slam on full-view images for indoor environments. *Applied Sciences*, 8(11) :2268, 2018.

[59] Yunpeng Li, Noah Snavely, Dan Huttenlocher, and Pascal Fua. Worldwide pose estimation using 3d point clouds. In *European conference on computer vision*, pages 15–29. Springer, 2012.

[60] Yunpeng Li, Noah Snavely, and Daniel P Huttenlocher. Location recognition using prioritized feature matching. In *European conference on computer vision*, pages 791–804. Springer, 2010.

[61] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco : Common objects in context. In *European conference on computer vision*, pages 740–755. Springer, 2014.

[62] Liu Liu, Hongdong Li, and Yuchao Dai. Efficient global 2d-3d matching for camera localization in a large-scale 3d map. In *Computer Vision (ICCV), 2017 IEEE International Conference on*, pages 2391–2400. IEEE, 2017.

[63] Zhaoliang Liu, Ling-Yu Duan, Jie Chen, and Tiejun Huang. Depth-based local feature selection for mobile visual search. In *2016 IEEE International Conference on Image Processing (ICIP)*, pages 276–280. IEEE, 2016.

[64] David G Lowe. Method and apparatus for identifying scale invariant features in an image and use of same for locating an object in an image, March 23 2004. US Patent 6,711,293.

[65] Will Maddern, Geoffrey Pascoe, Chris Linegar, and Paul Newman. 1 year, 1000 km : The oxford robotcar dataset. *The International Journal of Robotics Research*, 36(1) :3–15, 2017.

[66] Achref Ouni Eric Royer Marc Chevaldonne, Michel Dhome. Leveraging semantic segmentation for improved image based localization. In *under examination in ITS2021*. ieee, 2021.

[67] Achref Ouni Eric Royer Marc Chevaldonne, Michel Dhome. A new cbir model using semantic segmentation and fast spatial binary encoding. In *under examination in Machine learning journal*. Springer, 2021.

[68] Daniela Massiceti, Alexander Krull, Eric Brachmann, Carsten Rother, and Philip HS Torr. Random forests versus neural networks—what's best for camera localization ? In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 5118–5125. IEEE, 2017.

[69] Colin McManus, Ben Upcroft, and Paul Newmann. Scene signatures : Localised and point-less features for localisation. 2014.

[70] Lili Meng, Jianhui Chen, Frederick Tung, James J Little, and Clarence W de Silva. Exploiting random rgb and sparse features for camera pose estimation. In *BMVC*, 2016.

[71] Krystian Mikolajczyk and Cordelia Schmid. An affine invariant interest point detector. In *European conference on computer vision*, pages 128–142. Springer, 2002.

[72] Anastasiia Mishchuk, Dmytro Mishkin, Filip Radenovic, and Jiri Matas. Working hard to know your neighbor's margins : Local descriptor learning loss. In *Advances in Neural Information Processing Systems*, pages 4826–4837, 2017.

[73] Marius Muja and David Lowe. Flann-fast library for approximate nearest neighbors user manual. *Computer Science Department, University of British Columbia, Vancouver, BC, Canada*, 2009.

[74] Marius Muja and David G Lowe. Fast approximate nearest neighbors with automatic algorithm configuration. *VISAPP (1)*, 2(331-340) :2, 2009.

[75] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam : a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5) :1147–1163, 2015.

[76] Gerhard Neuhold, Tobias Ollmann, Samuel Rota Bulo, and Peter Kontschieder. The mapillary vistas dataset for semantic understanding of street scenes. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 4990–4999, 2017.

[77] Kai Ni, Anitha Kannan, Antonio Criminisi, and John Winn. Epitomic location recognition. *IEEE transactions on pattern analysis and machine intelligence*, 31(12) :2158–2167, 2009.

[78] Yuki Ono, Eduard Trulls, Pascal Fua, and Kwang Moo Yi. Lf-net : learning local features from images. In *Advances in neural information processing systems*, pages 6234–6244, 2018.

[79] Achref Ouni, Eric Royer, Marc Chevaldonné, and Michel Dhome. A hybrid approach for improved image similarity using semantic segmentation. In *International Symposium on Visual Computing*, pages 647–657. Springer, 2020.

[80] Achref Ouni, Eric Royer, Michel Dhome, et al. Geometric-visual descriptor for improved image based localization. In *2020 IEEE International Conference on Visual Communications and Image Processing (VCIP)*, pages 289–293. IEEE, 2020.

[81] Mattis Paulin, Matthijs Douze, Zaid Harchaoui, Julien Mairal, Florent Perronin, and Cordelia Schmid. Local convolutional features with unsupervised training for image retrieval. In *Proceedings of the IEEE international conference on computer vision*, pages 91–99, 2015.

[82] Florent Perronnin and Christopher Dance. Fisher kernels on visual vocabularies for image categorization. In *2007 IEEE conference on computer vision and pattern recognition*, pages 1–8. IEEE, 2007.

[83] James Philbin, Ondrej Chum, Michael Isard, Josef Sivic, and Andrew Zisserman. Object retrieval with large vocabularies and fast spatial matching. In *2007 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8. IEEE, 2007.

[84] Charles R Qi, Hao Su, Kaichun Mo, and Leonidas J Guibas. Pointnet : Deep learning on point sets for 3d classification and segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 652–660, 2017.

[85] Xiaozhi Qu, Bahman Soheilian, Emmanuel Habets, and Nicolas Paparoditis. Evaluation of sift and surf for vision based localization. *International Archives of the Photogrammetry, Remote Sensing & Spatial Information Sciences*, 41, 2016.

[86] Xiaozhi Qu, Bahman Soheilian, and Nicolas Paparoditis. Vehicle localization using mono-camera and geo-referenced traffic signs. In *2015 IEEE Intelligent Vehicles Symposium (IV)*, pages 605–610. IEEE, 2015.

[87] Filip Radenović, Giorgos Tolias, and Ondřej Chum. Fine-tuning cnn image retrieval with no human annotation. *IEEE transactions on pattern analysis and machine intelligence*, 41(7) :1655–1668, 2018.

[88] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net : Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention*, pages 234–241. Springer, 2015.

[89] Edward Rosten and Tom Drummond. Machine learning for high-speed corner detection. In *European conference on computer vision*, pages 430–443. Springer, 2006.

[90] Torsten Sattler, Michal Havlena, Konrad Schindler, and Marc Pollefeys. Large-scale location recognition and the geometric burstiness problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1582–1590, 2016.

[91] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Fast image-based localization using direct 2d-to-3d matching. In *2011 International Conference on Computer Vision*, pages 667–674. IEEE, 2011.

[92] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Improving image-based localization by active correspondence search. In *European conference on computer vision*, pages 752–765. Springer, 2012.

[93] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Towards fast image-based localization on a city-scale. In *Outdoor and Large-Scale Real-World Scene Analysis*, pages 191–211. Springer, 2012.

[94] Torsten Sattler, Bastian Leibe, and Leif Kobbelt. Efficient & effective prioritized matching for large-scale image-based localization. *IEEE transactions on pattern analysis and machine intelligence*, 39(9) :1744–1756, 2016.

[95] Davide Scaramuzza, Agostino Martinelli, and Roland Siegwart. A flexible technique for accurate omnidirectional camera calibration and structure from motion. In *Fourth IEEE International Conference on Computer Vision Systems (ICVS'06)*, pages 45–45. IEEE, 2006.

[96] Dominik Schlegel, Mirco Colosi, and Giorgio Grisetti. Proslam : Graph slam from a programmer's perspective. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 1–9. IEEE, 2018.

[97] Jamie Shotton, Ben Glocker, Christopher Zach, Shahram Izadi, Antonio Criminisi, and Andrew Fitzgibbon. Scene coordinate regression forests for camera relocalization in rgb-d images. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 2930–2937, 2013.

[98] Abhinav Shrivastava, Tomasz Malisiewicz, Abhinav Gupta, and Alexei A Efros. Data-driven visual similarity for cross-domain image matching. In *ACM Transactions on Graphics (ToG)*, volume 30, page 154. ACM, 2011.

[99] Josef Sivic and Andrew Zisserman. Video google : A text retrieval approach to object matching in videos. In *null*, page 1470. IEEE, 2003.

[100] Noah Snavely, Steven M Seitz, and Richard Szeliski. Photo tourism : exploring photo collections in 3d. In *ACM Siggraph 2006 Papers*, pages 835–846. 2006.

[101] Elena Stumm, Christopher Mei, Simon Lacroix, Juan Nieto, Marco Hutter, and Roland Siegwart. Robust visual place recognition with graph kernels. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4535–4544, 2016.

[102] Peter Sturm and Bill Triggs. A factorization based algorithm for multi-image projective structure and motion. In *European conference on computer vision*, pages 709–720. Springer, 1996.

[103] Shinya Sumikura, Mikiya Shibuya, and Ken Sakurada. Openvslam : A versatile visual slam framework. In *Proceedings of the 27th ACM International Conference on Multimedia*, pages 2292–2295. ACM, 2019.

[104] Ke Sun, Yang Zhao, Borui Jiang, Tianheng Cheng, Bin Xiao, Dong Liu, Yadong Mu, Xinggang Wang, Wenyu Liu, and Jingdong Wang. High-resolution representations for labeling pixels and regions. *arXiv preprint arXiv :1904.04514*, 2019.

[105] Niko Sünderhauf, Sareh Shirazi, Feras Dayoub, Ben Upcroft, and Michael Milford. On the performance of convnet features for place recognition. In *2015 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 4297–4304. IEEE, 2015.

[106] Linus Svarm, Olof Enqvist, Magnus Oskarsson, and Fredrik Kahl. Accurate localization and pose estimation for large 3d models. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 532–539, 2014.

[107] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander A Alemi. Inception-v4, inception-resnet and the impact of residual connections on learning. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[108] Richard Szeliski and Philip HS Torr. Geometrically constrained structure from motion : Points on planes. In *European Workshop on 3D Structure from Multiple Images of Large-Scale Environments*, pages 171–186. Springer, 1998.

[109] Yurun Tian, Bin Fan, and Fuchao Wu. L2-net : Deep learning of discriminative patch descriptor in euclidean space. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 661–669, 2017.

[110] Carl Toft, Erik Stenborg, Lars Hammarstrand, Lucas Brynte, Marc Pollefeys, Torsten Sattler, and Fredrik Kahl. Semantic match consistency for long-term visual localization. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 383–399, 2018.

[111] A. Torii, R. Arandjelović, J. Sivic, M. Okutomi, and T. Pajdla. 24/7 place recognition by view synthesis. In *CVPR*, 2015.

[112] Akihiko Torii, Relja Arandjelovic, Josef Sivic, Masatoshi Okutomi, and Tomas Pajdla. 24/7 place recognition by view synthesis. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1808–1817, 2015.

[113] Akihiko Torii, Josef Sivic, and Tomas Pajdla. Visual localization by linear combination of image descriptors. In *2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops)*, pages 102–109. IEEE, 2011.

[114] Julien Valentin, Matthias Nießner, Jamie Shotton, Andrew Fitzgibbon, Shahram Izadi, and Philip HS Torr. Exploiting uncertainty in regression forests for accurate camera relocalization. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4400–4408, 2015.

[115] Limin Wang, Sheng Guo, Weilin Huang, and Yu Qiao. Places205-vggnet models for scene recognition. *arXiv preprint arXiv :1508.01667*, 2015.

[116] Tobias Weyand, Ilya Kostrikov, and James Philbin. Planet-photo geolocation with convolutional neural networks. In *European Conference on Computer Vision*, pages 37–55. Springer, 2016.

[117] William J. Wolfe, Donald Mathis, C Weber Sklair, and Michael Magee. The perspective view of three points. *IEEE Transactions on Pattern Analysis & Machine Intelligence*, (1) :66–73, 1991.

[118] Zhiding Yu, Chen Feng, Ming-Yu Liu, and Srikumar Ramalingam. Casenet : Deep category-aware semantic edge detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5964–5973, 2017.

[119] Zhiding Yu, Weiyang Liu, Yang Zou, Chen Feng, Srikumar Ramalingam, BVK Vijaya Kumar, and Jan Kautz. Simultaneous edge alignment and learning. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 388–404, 2018.

[120] Amir Roshan Zamir and Mubarak Shah. Accurate image localization based on google maps street view. In *European Conference on Computer Vision*, pages 255–268. Springer, 2010.

[121] Amir Roshan Zamir and Mubarak Shah. Image geo-localization based on multiplenearest neighbor feature matching usinggeneralized graphs. *IEEE transactions on pattern analysis and machine intelligence*, 36(8) :1546–1558, 2014.

[122] Bernhard Zeisl, Torsten Sattler, and Marc Pollefeys. Camera pose voting for large-scale image-based localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 2704–2712, 2015.

[123] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.

[124] Hang Zhang, Kristin Dana, Jianping Shi, Zhongyue Zhang, Xiaogang Wang, Ambrish Tyagi, and Amit Agrawal. Context encoding for semantic segmentation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7151–7160, 2018.

[125] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[126] Zhuoyao Zhong, Lianwen Jin, and Zecheng Xie. High performance offline handwritten chinese character recognition using googlenet and directional feature maps. In *2015 13th International Conference on Document Analysis and Recognition (ICDAR)*, pages 846–850. IEEE, 2015.

[127] Bolei Zhou, Hang Zhao, Xavier Puig, Sanja Fidler, Adela Barriuso, and Antonio Torralba. Scene parsing through ade20k dataset. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 633–641, 2017.

[128] Yan Zhou, Xianwei Zheng, Ruizhi Chen, Hanjiang Xiong, and Sheng Guo. Image-based localization aided indoor pedestrian trajectory estimation using smartphones. *Sensors*, 18(1) :258, 2018.