Caractérisation de l'incertitude liée à la position des points d'intérêt dans le contexte de la navigation

Katia Sousa Lillo

UNIVERSITÉ GRENOBLE ALPES

THÈSE

pour obtenir le grade de

DOCTEUR DE L'UNIVERSITÉ DE GRENOBLE ALPES

Spécialité : **SIGNAL IMAGE PAROLE TELECOMS**

Arrêté ministériel : 25 mai 2016

Présentée par
Katia SOUSA LILLO

Thèse dirigée par **Nicolas MARCHAND**, Université Grenoble Alpes et **Simon LACROIX**, Institut National des Sciences Appliquées de Toulouse,
co-encadrée par **Nicolas VERCIER**, Thales AVS, et **Amaury NEGRE**, GIPSA-LAB

préparée au sein du laboratoire
Grenoble Images Parole Signal Automatique
dans l'École Doctorale **Electronique, Electrotechnique, Automatique, Traitement du Signal (EEATS)**

Caractérisation de l'incertitude liée à la position des points d'intérêt dans le contexte de la navigation

Thèse soutenue publiquement le **20 février 2023**,
devant le jury composé de :

Cédric DEMONCEAUX
Professeur des universités, Université de Bourgogne, Rapporteur
Pascal VASSEUR
Professeur des universités, Université de Picardie Jules Verne, Rapporteur
Guillaume ALLIBERT
Maître de conférences, Université de Côte d'Azur, Examinateur
Guillaume CARON
Maître de conférences, Université de Picardie, Examinateur
Denis Pellerin, Président du jury
Professeur des universités, Université Grenoble Alpes, Examinateur
Simon LACROIX
Directeur de recherche, CNRS, co-directeur de thèse
Nicolas MARCHAND
Directeur de recherche, CNRS, Directeur de thèse
Amaury NEGRE
Ingénieur de recherche, CNRS, Invité

UNIVERSITÉ DE GRENOBLE ALPES

ÉCOLE DOCTORALE EEATS

Electronique, Electrotechnique, Automatique, Traitement du Signal

T H È S E

pour obtenir le titre de

docteur en sciences

de l'Université de Grenoble Alpes

Mention : SIGNAL IMAGE PAROLE TELECOMS

Présentée et soutenue par

Katia SOUSA LILLO

Caractérisation de l'incertitude liée à la position des points d'intérêt dans le contexte de la navigation

Thèse dirigée par Nicolas MARCHAND et Simon LACROIX

préparée au laboratoire GIPSA-LAB

soutenue le **20 février 2023**

Jury :

Rapporteurs :	Cédric DEMONCEAUX	-	Université de Bourgogne
	Pascal VASSEUR	-	Université de Picardie Jules Verne
Président :	Denis PELLERIN	-	Université Grenoble Alpes
Examinateurs :	Guillaume ALLIBERT	-	Université de Côte d'Azur
	Guillaume CARON	-	Université de Picardie Jules Verne
Invité :	Amaury NEGRE	-	GIPSA-LAB - CNRS

Encadrement :

Directeurs :	Nicolas MARCHAND	-	GIPSA-LAB - CNRS
	Simon LACROIX	-	LAAS - CNRS
Co-encadrants :	Amaury NEGRE	-	GIPSA-LAB - CNRS
	Nicolas VERCIER	-	Thales AVS

Remerciements

Le travail de thèse est une longue marche. Sur le chemin, des gens formidables et inspirants croisent notre route. Ils nous poussent à nous dépasser, et aller plus loin dans l'analyse. Ces remerciements leurs sont dédiés.

Je souhaite remercier les membres de mon jury pour leur implication, leur disponibilité, et leurs commentaires pertinents et bienveillants. En particulier Pascal VASSEUR et Cédric DEMONCEAUX pour leur relecture minutieuse de mon manuscrit, ainsi que Guillaume CARON pour ses annotations et Guillaume ALLIBERT. Enfin, je tiens à remercier Denis PELLERIN pour sa relecture mais également tous ses conseils bienveillants et son soutien. Je te remercie d'avoir accepté d'endosser le rôle de Président du Jury.

Je tiens également à chaleureusement remercier Simon LACROIX pour avoir cru en moi et avoir permis à mon sujet de recherche d'aller loin. Je suis extrêmement reconnaissante d'avoir partagé ces discussions passionnantes sur le sujet qui m'ont rappelé pourquoi je souhaitais faire de la recherche : l'envie d'apprendre, et d'aller plus loin. Je te remercie d'avoir pris le temps de m'accompagner aussi bien techniquement que humainement. Je suis également reconnaissante de l'accueil reçu par l'équipe RIS du LAAS lors de ma venue à Toulouse.

Un grand merci également à Andrea DE MAIO pour sa disponibilité et son implication dans mon sujet de thèse. Je te remercie pour tes nombreux conseils et toutes les relectures d'articles. Merci pour l'intérêt porté à mes travaux, le partage et le soutien immense apporté.

Je tiens également à remercier Michèle ROMBAUT pour son soutien et son recul pragmatique sur mon travail de thèse. Merci de m'avoir intégré au sein de l'équipe robotique du GIPSA et pour tous les conseils donnés sur mon manuscrit, les articles, et les mots encourageants lorsqu'ils étaient nécessaires. Merci.

Je tiens à transmettre toute ma gratitude Jumana BOUSSEY pour son écoute attentive, sa disponibilité infaillible et son soutien sans limite. Cette thèse n'aurait pas vu le jour sans toi. Je remercie également mes directeur de spécialité Gang FENG et Steeve ZOZOR pour leurs encouragements et leurs soutiens.

Enfin, un grand merci à Amaury NEGRE pour son encadrement et sa disponibilité, et à Nicolas MARCHAND pour son soutien humain même à travers la difficulté.

Cette expérience de vie est rendue possible grâce au soutien des amis qui m'ont permis de rêver grand. Un immense merci à tous les copains Ben, Gautier, Nath, Fab, Fred, François, Matt, Clara, Elsa, Charles, Marc, Mti, Thomas, Romain, Maxime, Gab, Pierre, Christophe, Melek, Esteban, et Mathieu. Merci à Ben pour ces heures de footing à se vider l'esprit, à Gautier, Nath et Fab pour les mots d'encouragements, à Fredo et François de m'avoir formé et d'avoir cru en moi dès le jour 1, à Matt et Clara pour les câlins de réconfort. Un merci tout particulier à Marc pour son soutien sans limite malgré tous les hauts et les bas. Merci à Christophe pour avoir été mon coach mentale de remotivation et d'avoir toujours cru en moi. À Melek ("on l'a fait !"), je te remercie de m'avoir toujours apporté de la force et du réconfort. Enfin, merci à Pierre d'avoir eu la patience de relire tout mon manuscrit et de corriger mes

nombreuses fautes d'orthographes. Vous êtes des personnes exceptionnelles et je vous porte à tous un amour infini.

Mes derniers remerciements vont à ma famille, et en particulier à ma soeur. C'est la source de mon inspiration. Enfin, je remercie ma tatie et mon tonton d'avoir toujours été là pour moi, et d'avoir partagé ce moment unique et magique avec moi. Merci de me donner la force d'avancer. Je vous aime.

À tous les futurs doctorants qui lisent ces quelques lignes, courage à vous !

À ma soeur, à mes amis,

As coisas acontecem no momento certo.
– Tia

Table des matières

Table des sigles et acronymes

GNSS	*Géolocalisation et Navigation par un Système de Satellites*
GPS	*Global Positioning System*
VO	*Visual Odometry*
VIO	*Visual Inertial Odometry*
VINS	*Visual Inertial Navigation System*
SLAM	*Simultaneous Localization And Mapping*
KF	*Kalman Filter*
EKF	*Extended Kalman Filter*
IMU	*Inertial Measurement Unit*
KLT	*Kanade Lucas Tomatsi tracker*
RdN	*Réseau de Neurones*
CNN	*Convolutional Neural Network*
DL	*Deep Learning*
LSTM	*Long Short Term Memory*

Notations

Notation	Définition
x	scalaire
$\boldsymbol{x}$	vecteur
$\mathbf{X}$	matrice
$\mathbf{X}^T$	transposée de la matrice $\mathbf{X}$
$\mathbf{X}^{-1}$	inverse de la matrice $\mathbf{X}$
$det(\mathbf{X}) = \|\mathbf{X}\|$	déterminant de la matrice $\mathbf{X}$
$trace(\mathbf{X})$	trace de la matrice $\mathbf{X}$
$diag(\mathbf{X})$	vecteur contenant les termes diagonaux de la matrice $\mathbf{X}$
$\dot{\boldsymbol{x}}/\ddot{\boldsymbol{x}}$	dérivée temporelle première et seconde de $\boldsymbol{x}$
$\boldsymbol{x}_u(u) = \frac{\partial \boldsymbol{x}(\boldsymbol{u})}{\partial \boldsymbol{u}}$	dérivée partielle de $\boldsymbol{x}$ selon $\boldsymbol{u}$
$\Delta \boldsymbol{x}$	Laplacien de $\boldsymbol{x}$
$\nabla \boldsymbol{x}$	Gradient de $\boldsymbol{x}$
$\hat{\boldsymbol{x}}$	estimée de $\boldsymbol{x}$
$\boldsymbol{x}_k$	vecteur $\boldsymbol{x}$ à l'instant k
$\boldsymbol{\mathcal{R}}_x$	repère cartésien d'origine ${}^xO$ et d'axes $({}^x\boldsymbol{u}, {}^x\boldsymbol{v}, {}^x\boldsymbol{w})$
${}^a\boldsymbol{x}$	vecteur $\boldsymbol{x}$ exprimé dans le repère a
${}^a_b\boldsymbol{R}$	matrice de rotation décrivant la rotation du repère a vers le repère b
${}^a_b\boldsymbol{T}$	matrice de passage du repère a vers le repère b
$[\boldsymbol{x}^\times]$	matrice antisymétrique [1]
$\wedge$	produit vectoriel
$*$	produit de convolution
$\circ$	composition de fonctions
$\|\|.\|\|$	norme euclidienne
$x \in \|[a; b]\|$	x peut prendre toutes les valeurs entières de $\mathbb{N}$ comprises entre a et b (inclus)
$\mathbb{0}_{1\times N}$	matrice nulle de dimension $1 \times N$
$\mathbb{1}_{1\times N}$	matrice unitaire de dimension $1 \times N$
$\mathcal{I}_{N\times N}$	matrice identité de dimension $N \times N$

1. $\left[\boldsymbol{x}^\times\right] = \left[\begin{pmatrix} x_1 \\ x_2 \\ x_3 \end{pmatrix}^\times\right] = \begin{bmatrix} 0 & -x_3 & x_2 \\ x_3 & 0 & -x_1 \\ -x_2 & x_1 & 0 \end{bmatrix}$

Chapitre 1

Introduction

> Ce que nous appelons commencement est souvent la fin. La fin, c'est l'endroit d'où nous partons.
>
> Thomas Stearns Eliot

Motivations

La navigation est le processus d'estimation du mouvement d'un système mobile par rapport à un répère fixe, à partir de mesures capteurs relatives au système. Le terme navigation vient du latin *navigatio*, qui décrit le voyage en bateau, l'action de naviguer. Le besoin de connaître sa position dans le but de calculer le trajet à emprunter nait dès l'Antiquité avec la navigation *côtière*, reposant sur l'observation des côtes par l'identification de points remarquables tels que des *amers* naturels, des balises ou encore l'éclairage des phares. Au-delà des littoraux, c'est la navigation *à l'estime* qui prend le pas. Elle consiste à évaluer au mieux la direction suivie, autrement dit l'orientation du navire, ainsi que la distance parcourue, par l'étude du cap, de la vitesse et du temps de parcours du bâtiment depuis la dernière position connue.

> ***Nota Bene*** Le terme navigation définit la localisation d'un système mobile mais également l'action de mettre en mouvement le système mobile, autrement dit son pilotage. Dans ce manuscrit, nous nous intéressons uniquement à l'estimation du positionnement d'un système mobile. Par abus de language, le terme navigation sera utilisé pour caractériser la localisation d'un système mobile.

Associée à la navigation *astronomique*, la latitude d'un bateau peut être évaluée avec précision en estimant l'élévation de l'étoile polaire la nuit, et l'altitude du soleil la journée. A contrario, la détermination de la longitude, issue de l'intégration de la vitesse, requiert la connaissance de l'heure [Sob10]. Il faudra attendre le XVIII^{e}siècle pour voir apparaître le chronomètre de marine, un cadrant suffisamment précis pour fournir un temps de référence pour le calcul de la longitude. En conséquence, l'incertitude relative à la longitude a longtemps conduit les explorateurs à estimer une position erronée et régulièrement corriger leur trajectoire avec de nouvelles observations.

Parmis les plus célèbres erreurs, on peut citer la découverte des Amériques par *Christophe Colomb* (Fig. 1.1) qui, en 1492, croit fermement avoir atteint le *nouveau monde*, autrement dit les *Indes*, quand en réalité il est aux actuelles Bahamas. La caractérisation de l'incertitude liée à une mesure est donc une information aussi importante que l'information de mesure. L'incertitude caractérise ici la précision d'une mesure. Elle permet d'évaluer la confiance d'une observation et les erreurs qu'elle induit. Dans cette définition de l'incertitude, les mesures considérées sont supposées fiables, autrement dit elles ne correspondent pas à des fautes ou des informations aberrantes.

FIGURE 1.1 – Premier voyage de Christophe Colomb vers les Indes occidentales (credit : alienor.org)

Dans les années 60, l'avènement du programme Apollo (Fig. 1.2) pousse le developpement d'outils de navigation plus précis, tels que les centrales inertielles ou IMU [1]. Il voit naître le premier processus de navigation autonome avec le filtre de Kalman (KF [2]) qui fusionne des mesures issues de capteurs de nature différentes [KO60] afin d'estimer l'état du système mobile, dans notre cas sa position. Il permet par ailleurs d'évaluer l'incertitude sur l'état du système mobile à partir des incertitudes issues des mesures. La caractérisation des incertitudes relatives aux mesures est donc primordiale.

Les capteurs de natures différentes connaissent des limitations. Les capteurs à intégration tels que l'IMU posent des problèmes de synchronisation lors de l'intégration des données et souffrent de dérive. Les capteurs de type géolocalisation, comme le GPS, sont sujets aux pertes de données ou aux fautes de mesures produites par des rebonds. Enfin les capteurs d'observation du mouvement relatif (Caméra, LIDAR [3], RADAR [4] ..) ont besoin d'un point de référence, ainsi que de construire ou d'utiliser une carte de l'environnement [Tak07]. La diversification des combinaisons de capteurs (*e.g.* GPS, IMU, RADAR, LIDAR, Caméra etc ...) permet de contourner les défauts de chaque capteur pris individuellement. En particulier, l'utilisation de capteurs de plusieurs natures est une solution au problème de non disponibilité de certaines données et permet de garantir un apport d'information au système constant.

Au début des années 2000, le programme d'exploration de Mars de la NASA (Fig. 1.3) propose une nouvelle méthode d'estimation du mouvement à partir d'un ou plusieurs capteurs caméras : l'odométrie visuelle (VO [5]). Bien que l'odométrie permette déjà d'obtenir une estimation du mouvement, en intégrant la vitesse angulaire des roues d'un Rover à l'aide d'un modèle cinématique par exemple, celle-ci est sujette aux glissements des roues dus au terrain

1. *Inertial Measurement Unit*
2. *Kalman Filter*
3. LIght Detection And Ranging
4. RAdio Detection And Ranging
5. *Visual Odometry*

FIGURE 1.2 – Mission Apollo (credit : nasa.org)

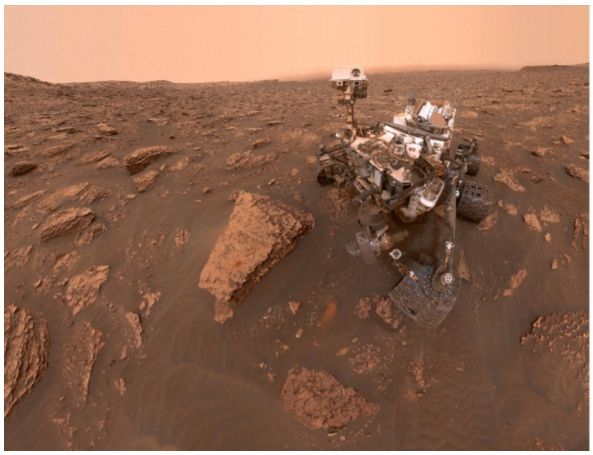


FIGURE 1.3 – Rover curiosity sur Mars (credit : mars.nasa.org)

(sable, roche, etc ...). L'odométrie visuelle, introduite par [Mor80], est une méthode à l'estime qui évalue le mouvement élémentaire entre deux images. La minimisation de l'erreur entre les données observées et les données prédites sur l'ensemble des positions élémentaires issues d'une sequence d'images peut être réalisée avec du *bundle adjustment*, en français l'ajustement de faisceaux [Bro58], une méthode d'optimisation globale.

En quelques années, les caméras sont devenues de plus en plus précises en terme de résolution et peu coûteuses, en contraste avec des IMU de haute performance. Leur compacité leur permet d'être facilement intégrées dans un système de navigation, donnant jour à des sytèmes de VO couplés avec l'inertie VIO [6]. Par ailleurs, de par leur nature, les caméras rendent possible la construction de cartes de l'environnement et la localisation simultanément à travers des techniques de SLAM [7] [LDW91]. De manière analogue, des systèmes couplant de l'inertie et une caméra avec des techniques de filtrage, telles que le KF, donnent le jour à des systèmes de navigation inertie-vision (VINS [8])[Hua19].

On distingue deux approches de fusion de données : l'hybridation *lâche* ou *serrée*. L'hybridation *lâche* intègre les mesures visuelles et inertielles séparément et en amont de la fusion, pour en déduire leurs propres contraintes de mouvement (*i.e.* la position), puis les fusionner. Cette approche est peu coûteuse à mettre en place, néanmoins, elle altère le résultat final, engendrant une perte d'information. À l'inverse, l'hybridation *serrée* fusionne directement les mesures issues de la caméra et de l'IMU au sein d'un seul processus, induisant une plus grande précision, elle est toutefois plus coûteuse à mettre en œuvre.

Comme chaque outil de mesure, la caméra a ses propres limites : des limites physiques, telles que la saturation du capteur d'image par exemple. En effet, la perception de l'environnement peut être biaisée par les conditions de visibilité, les occultations, des déformations optiques, du flou optique induit par une fréquence d'échantillonnage trop faible, ou plus simplement la

6. *Visual Inertial Odometry*
7. *Simultaneous Localization And Mapping*
8. *Visual Inertial Navigation System*

météo. Il est donc nécessaire de caractériser l'incertitude des mesures optiques. En particulier, le but est de prédire l'écart entre les mesures optiques et leur vérité terrain. Dès lors, il faut définir un modèle d'erreur.

Un moyen de se localiser est défini par l'utilisation d'*amers* ou de *points d'intérêt*. En particulier, une méthode possible d'analyse d'images repose sur l'extraction de points saillants dans cette image, et plus précisément leurs positions dans l'image. L'objectif de la localisation est donc de prédire la meilleure estimation de la position d'un système mobile à partir de coordonnées 2-D observées dans l'image et de qualifier l'incertitude résultante sur la position du système. Ainsi, il s'agit de qualifier l'incertitude, c'est-à-dire la précision, sur la position des points d'intérêt.

Comment caractériser la précision des points suivis dans une image dans le but de les intégrer à une hybridation serrée ? En particulier, comment caractériser localement la précision d'un point d'intérêt dans le plan image du capteur ?

Habituellement, l'incertitude sur la localisation du système mobile est estimée sans caractérisation préalable de l'incertitude locale sur la position des points d'intérêt, ce qui ne permet pas une hybridation serrée et précise. L'intérêt de caractériser la précision des points dans le plan image est d'utiliser le maximum d'information utile dans une image, même dans le cas d'une image partiellement bruitée, couramment mise de côté et non exploitée. Par ailleurs, un apport d'information constant est ainsi garanti pour alimenter le filtre ou la méthode d'optimisation choisie. Les approches analytiques populaires ne capturent que partiellement le modèle d'erreur du suivi des points observés. La solution proposée dans ce manuscrit nourrit un algorithme de navigation analytique avec des incertitudes issues d'un modèle d'erreur appris par un réseau de neurones (RdN [9]). Contrairement à des méthodes de linéarisation plus classiques, l'utilisation des RdN permet de proposer un modèle d'estimation des incertitudes applicable à toutes les méthodes d'extraction de points ou d'amers visuels.

Contexte et objectifs

La localisation d'un système mobile en temps réel est l'un des principaux enjeux des systèmes de navigation. Les techniques les plus communément utilisées s'appuient sur les données GPS [10]. Les travaux de thèse s'inscrivent dans les activités de R&T (Recherches et Technlogies), menées par THALES, destinées à étudier les alternatives ou compléments au GNSS [11] (GPS, GALILEO [12], etc ...) pour fusionner et corriger des solutions de navigation inertielle. Les applications envisagées concernent les UAVs [13] en environnement urbain. Ce besoin est motivé par la non disponibilité des mesures GPS ou leurs fortes dégradations. Parmi les effets redoutés on peut citer notamment

* L'indisponibilité en navigation à l'intérieur ;
* L'effet de multi-trajets ;

9. *Réseau de Neurones*
10. *Global Positioning System*
11. *Géolocalisation et Navigation par un Système de Satellites*
12. Système de positionnement par satellites initié par l'Union européenne
13. Unmanned Aerial Vehicles

* L'impact de masquages ;
* Les interférences volontaires ou non qui peuvent conduire à la non disponibilité des mesures de recalage.

> ***Nota Bene*** La dégradation de précision ou la non disponibilité des mesures GPS est d'autant plus préjudiciable que la classe de précision des capteurs inertiels utilisée est faible.

Le contexte des travaux de thèse se place dans une situation de non disponibilité de mesures GPS. Nous souhaitons développer un système de navigation Inertie/Vision (VINS) dans lequel la notion de précision de mesures visuelles est prise en compte d'une manière optimale.

Objectif et contributions de la thèse L'objectif de la thèse est de développer un système capable d'estimer la précision des mesures optiques. La particularité que nous proposons dans ce travail est la caractérisation de la précision locale de la position de chaque point d'intérêt indépendamment les uns des autres. L'apport d'une caractérisation fine et précise est évalué dans un système de navigation VINS dans lequel sont intégrées nos valeurs d'incertitude.

Les contributions de la thèse sont :

- la caractérisation de la précision d'une mesure optique dans le contexte de la navigation, et l'identification d'un modèle d'erreur adapté ;
- le developpement d'une architecture de RdN pour caractériser l'erreur locale sur la position des points d'intérêt dans l'image ;
- l'intégration des estimations de précision dans un système de navigation complet par filtrage serré et l'évaluation de leurs impacts ;
- le developpement d'une méthode d'estimation de l'erreur locale sur la position des points d'intérêt intégrant l'évolution temporelle d'un suivi de point.

Structure du manuscrit

Cette thèse est divisée en trois parties. Les outils et les principes de bases relatifs à la vision sont donnés dans la partie I. La partie II présente nos contributions et ses applications. La dernière partie III propose une discussion du travail ainsi que les pistes et perspectives futures.

PARTIE I

- Dans le **chapitre 2**, les notions et les outils mathématiques sont présentés. Le chapitre définit les principes de vision de l'extraction de l'information jusqu'à la reconstruction du mouvement en passant par le modèle sténopé de la caméra. Une première caractérisation de l'erreur est précisée ainsi que les impacts sur un système de navigation détaillant les leviers d'amélioration possibles.
- Le **chapitre 3** fait un état de l'art des travaux de navigation actuels, en particulier, l'utilisation des covariances.

PARTIE II

- Le **chapitre 4** présente le fonctionnement général d'un réseau de neurones. En particulier, il définit la fonction de minimisation et les métriques pour estimer l'incertitude

locale sur la position des points d'intérêt. Une seconde partie est dédiée à la construction du jeux de données pour l'apprentissage et l'explication des choix opérés.

- Le **chapitre 5** détaille le processus d'estimation des covariances des points d'intérêt dans l'image ainsi que les outils mathématiques relatifs aux réseaux de neurones. L'estimateur DUNE (Deep UNcertainty Estimation) est introduit. La mise en place de métriques pour valider les covariances résultantes du processus d'estimation est réalisée. L'estimateur est évalué sur des images de synthèse issues d'un environnement de simulation ainsi que sur des données réelles. Une validation géométrique est également proposée.
- Le **chapitre 6** intègre les covariances dans un système de navigation complet Inertie/Vision fourni par OpenVINS [Gen+20]. Un nouveau modèle d'erreur est proposé. Les covariances estimées sont comparées à des covariances fixes. L'impact sur la pose est évalué avec des métriques dédiées. Une sélection des points les plus précis à partir de nos covariances estimées est présentée, permettant une sélection active des observations dans le but d'améliorer l'estimation de la pose du système mobile et réduire la dérive.
- Le **chapitre 7** propose d'améliorer le processus d'estimation des covariances en corrélant les observations antérieures d'un même amer offrant ainsi la possibilité d'estimer la covariance d'un point suivi durant une séquence temporelle de taille variable. Cette solution est comparée à notre premier estimateur. Elle offre une meilleure précision et une plus grande flexibilité du format des données d'entrée.

PARTIE III

- Le **dernier chapitre** conclut sur l'ensemble de nos contributions et ouvre la discussion sur les perspectives de travaux futurs.

Première partie

Principes et concepts de Vision

Chapitre 2

Principes fondamentaux de la Vision et de la Navigation

> Tant que les lois mathématiques se réfèrent à la réalité, elles ne sont pas certaines, et tant qu'elles sont certaines, elles ne se réfèrent pas à la réalité.
>
> Albert Einstein

La localisation d'un système mobile est un processus permettant de reconstruire la position du système mobile dans un référentiel donné. Elle définit l'une des tâches principales de la navigation, donnant jour à des systèmes mobiles autonomes capables de suivre leur trajectoire, détecter et éviter correctement des obstacles. L'odométrie visuelle est l'une des techniques habituellement utilisée pour la localisation. Ce chapitre présente de façon détaillée les mesures visuelles, les approches possibles d'extraction et d'analyse de l'information dans une image. En particulier, il montre les limites des méthodes de localisation, telle que l'odométrie visuelle, et l'intérêt d'estimer l'incertitude des mesures visuelles. Dans un second temps, il s'attaque à définir des modèles d'erreurs.

2.1 Odométrie visuelle

L'Odométrie Visuelle (VO) [SF11] est une méthode populaire dans le domaine de la localisation. Elle estime le mouvement d'un système mobile (i.e. drônes, véhicule, Homme ...) à partir des données d'une ou plusieurs caméras embarquées sur système. Elle permet l'estimation incrémentale de la *pose* d'un système mobile, dans un environnement inconnu par rapport à un repère fixe, par l'analyse séquentielle des images issues du signal vidéo d'entrée capturé par le ou les capteurs caméras [YBHH15][Aqe+16].

> ***Nota Bene*** La **pose** d'un système définit la *position* et *l'orientation* de celui-ci dans un environnement.

Bien que la localisation en intérieur par caméra ait fait ses preuves [ESL21], dans un environnement extérieur non statique avec des conditions d'observation changeantes, elle reste un problème délicat et ambitieux. En effet, les variations de terrain (sol non lisse, etc ...) ainsi que les contrastes changeants d'ombre et de lumière, causés par le vent ou l'intensité lumineuse du soleil en fonction de l'heure de la journée, rendent difficile l'observation et la localisation en extérieur [Tak07]. Les conditions optimales de fonctionnement des méthodes

de localisation visuelle sont données par un environnement quasi-statique qui bénéficie d'une bonne luminosité, ni trop forte, pour ne pas saturer le capteur d'image, ni trop faible, pour laisser apparaître suffisament de contraste, avec une scène texturée pour permettre de capturer le mouvement du système mobile [SF11]. Dès lors, les zones ayant peu de textures ou des contrastes doux ne disposent pas de contours saillants à identifier et sont plus susceptibles de capturer un bruit optique ou numérique. Par ailleurs, les ombres, statiques ou dynamiques, issues de l'environnement de la scène peuvent influer sur le calcul du déplacement mesuré sur le plan image. En conséquence, des erreurs importantes apparaissent sur l'estimation du déplacement du système mobile [NVB11]. Par ailleurs, la VO basée sur de la vision monoculaire, c'est-à-dire l'utilisation d'un seul capteur caméra, est sujette aux incertitudes d'échelle [Kit+11] [ZSK14]. Selon [Kit+11], l'estimation du facteur d'échelle peut devenir erronée en cas de changement important de la pente de la route, ce qui peut conduire à une estimation incorrecte de la trajectoire résultante.

Les estimateurs *indirects* de points [Rub+11], aussi appelés estimateurs *sparse*, sont des méthodes basées sur l'identification de caractéristiques dans l'image. L'approche classique consiste à extraire les spécificités telles que l'information de position d'éléments remarquables (points, lignes ou contours) au sein de chaque image du flux vidéo afin de les mettre en correspondance à travers l'utilisation d'un descripteur invariant à certaines transformations. Ainsi, l'évolution des points remarquables est reconstruite sur le plan image permettant de remonter au mouvement du système mobile. L'inconvénient de cette approche est la dépendance à l'égard des seuils relatifs aux algorithmes de détection, de description et de *tracking*[1] ou *matching*[2], engendrant de fausses correspondances. Par ailleurs, la plupart des détecteurs sont peu coûteux en temps de calcul, pour des besoins d'implémentation et d'intégration temps réel, au détriment de leur précision. Les estimateurs *directs* [EKC17] [IA99] évaluent la géométrie de l'environnement et le mouvement du mobile directement à partir des valeurs d'intensité de l'image. Contrairement aux méthodes indirectes, elles exploitent toute l'information contenue dans une image, même celle des zones dont les gradients sont faibles (*i.e.* ayant de faibles contrastes). Elles sont très coûteuses en calcul et en mémoire. Enfin, certaines méthodes *hybrides*, ou *semi-dense*[FPS14], exploitent des portions denses de l'image, mais pas l'image dans son intégralité. Par choix, l'extraction de l'information sera réalisée par des méthodes indirectes (ou *sparse*). Elles ont l'avantage d'être bien moins coûteuse et montrent des résultats plus précis que certaines méthodes denses.

À terme, la VO consiste à retracer la trajectoire d'un système mobile à partir de sa position initale, autrement dit estimer la localisation du système mobile à chaque instant. La concaténation des déplacements relatifs estimés à partir des contraintes géométriques permet de reconstruire la trajectoire complète. Le processus de VO *classique* [Cad+16] permettant la reconstruction du mouvement est donné figure 2.1. À partir d'un modèle de caméra, une matrice de calibration est appliquée aux informations extraites de l'image pour estimer le movement du système mobile.

Toutefois, la VO est soumise à une dérive temporelle [Aqe+16]. L'accumulation de petites erreurs au cours du temps, notamment causées par les approximations de modèles [Zha98]

1. méthode de suivi de points remarquables
2. méthode de mise en correspondance de points remarquables

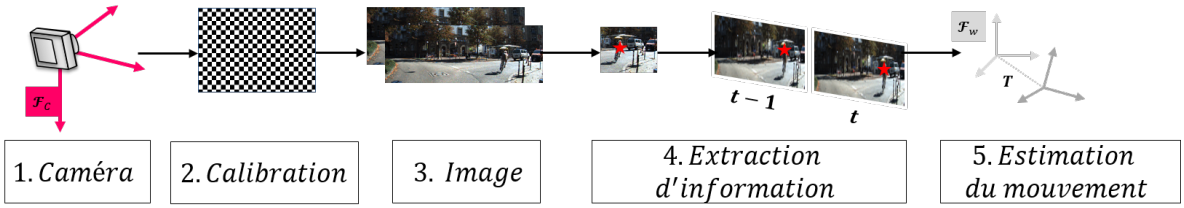


FIGURE 2.1 – Schéma bloc du processus d'odométrie visuelle *classique*.

(modèle de déformation optique, modèle sténopé, etc ...) ainsi que par l'incertitude lié au facteur d'échelle [CPY15], en particulier pour un système monoculaire, créent une imprécision sur la localisation qui s'accumule au fil du temps.

Par ailleurs, les informations extraites des images peuvent être bruitées induisant des biais d'observation, augmentant l'imprécision sur l'estimation de la position. Les sources de ces biais peuvent être de différentes natures, entre autres, la méthode d'extraction de l'information, la saturation du capteur d'image, ou à l'inverse la trop faible luminosité ; le manque de texture de l'environnement ; le flou optique ; etc ... Ces erreurs se cumulent et croîssent au court du temps produisant une déviation grandissante dans l'estimation de la localisation, comme schématisé figure 2.2.

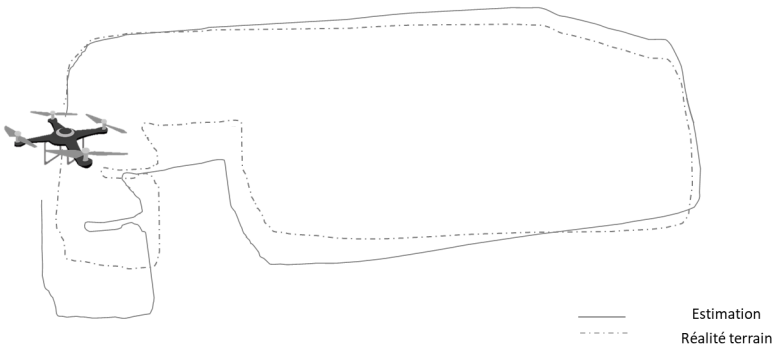


FIGURE 2.2 – Dérive temporelle de l'estimation de la position d'un drone par rapport à la vérité terrain.

L'estimation de l'incertitude est donc primordiale dans le domaine de la localisation. Elle permet de garantir la pose du système mobile avec une confiance donnée. En particulier, l'estimation de la précision des mesures visuelles intégrées dans les méthodes de localisation. En effet, une hypothèse couramment posée suppose constante la précision des mesures visuelles issues d'une même caméra. Autrement dit, la répartition de l'erreur de mesure est uniforme sur le capteur. Nous supposons que chaque mesure visuelle a sa propre précision, fonction de l'environnement capturé (faible luminosité, peu de texture ..) mais également de sa position dans l'image (au centre ou dans les bords). Nos travaux ont pour vocation d'estimer la pré-

cision de chaque mesure visuelle dans une image. Ces mesures visuelles sont caractérisées par des points saillants dans l'image extraite par une méthode d'extraction indirecte. Le système envisagé est un système mobile monoculaire.

2.2 Modèle de caméra et contraintes géométriques

2.2.1 Le modèle sténopé

Les principes de fonctionnement d'une caméra font écho à la *camera obscura*, un instrument optique composée d'une chambre noire dans laquelle vient se projeter la lumière à travers une petite ouverture, comme illustré Fig. 2.3. L'image observée à l'intérieur de la chambre noire est une image inversée de l'environnement extérieur dont sont issus les rayons lumineux.

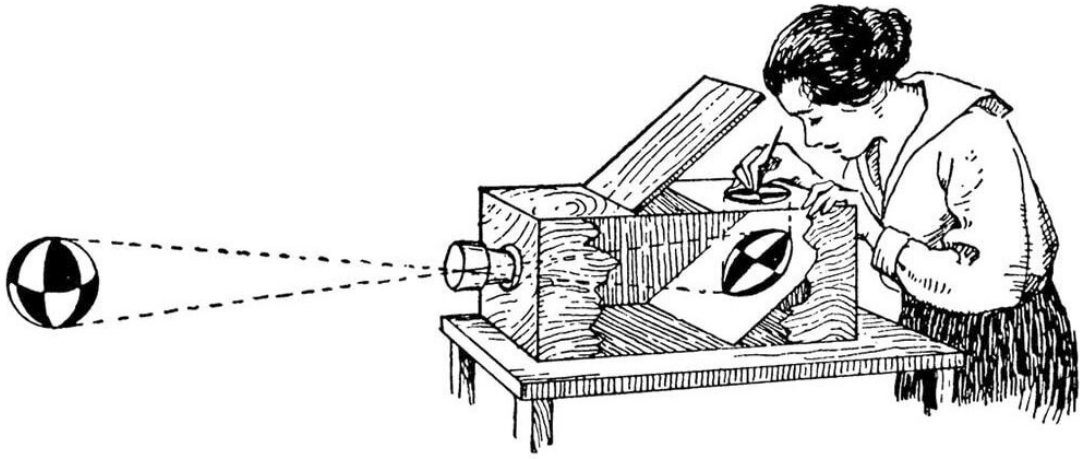


FIGURE 2.3 – Camera Obscura (credit : photoswithphones.com)

Les premières traces écrites de la *camera obscura* remontent au IV^e siècle avant notre ère dans le livre de Mozi [MGL18], philosophe chinois. Il y décrit le modèle par lequel l'image inversée se forme par intersection des rayons à travers un sténopé. Ce modèle, couramment désigné *modèle sténopé* [Stu97] [HZ03] (Fig. 2.4), est encore aujourd'hui la représentation mathématiques la plus utilisée pour décrire le fonctionnement d'un capteur caméra. Le modèle sténopé modélise un capteur caméra par une projection perspective simplifiée de la réalité. Soit un point 3D ${}^{w}\boldsymbol{M} \in \mathbb{R}^3$ de l'environnement, exprimé dans le repère $\boldsymbol{\mathcal{R}}_w$ et ${}^{pix}\boldsymbol{m}$ sa représentation dans $\boldsymbol{\mathcal{R}}_{pix}$ (définit Fig. 2.4). Alors le modèle sténopé décrit la transformation d'un point 3D ${}^{w}\boldsymbol{M}$ de l'environnement projeté en ${}^{uv}\boldsymbol{m}$ sur le plan image formé sur le capteur d'image de la caméra. Cette transformation est représentée par la fonction de projection $\pi(\mathbb{R}^3, SE(3)) \rightarrow \mathbb{R}^3$ (chapitre 4).

Couramment, le repère univers $\boldsymbol{\mathcal{R}}_w$ (Fig. 2.4) caractérisant l'évolution de la position de la caméra est fixe. $\boldsymbol{\mathcal{R}}_C$ a pour origine le centre de projection de la caméra et l'axe ${}^{C}\boldsymbol{z}$ est confondu avec l'axe optique. Le plan formé par les axes ${}^{C}\boldsymbol{x}$ et ${}^{C}\boldsymbol{y}$ est pris parallèle au plan image. Le repère image $\boldsymbol{\mathcal{R}}_{Img}$ prend son origine en ${}^{Img}\boldsymbol{O}$. Les axes ${}^{Img}\boldsymbol{x}$ et ${}^{Img}\boldsymbol{y}$ sont définis comme la projection de ${}^{C}\boldsymbol{x}$ et ${}^{C}\boldsymbol{y}$ en ${}^{Img}\boldsymbol{O}$. Enfin, le repère pixel $\boldsymbol{\mathcal{R}}_{pix}$ caractérise le matriçage de l'image en pixels 2D. L'origine du repère est défini par le coin supérieur gauche de l'image.

Toutes les coordonnées utilisées sont homogènes.

Nota Bene Les coordonnées homogènes sont définies par l'ajout d'une composante unitaire à un vecteur. Soit $\boldsymbol{X} \in \mathbb{R}^3$, alors sa représentation homogène est $\begin{pmatrix} \boldsymbol{X}^T & 1 \end{pmatrix}^T \in \mathbb{R}^4$. Sans homogénéisation, une transformation rigide s'exprime selon (2.1) où $\boldsymbol{R}$ représente la rotation et $\boldsymbol{t}$ la translation du vecteur $\boldsymbol{X}$.

$$ {}^a_b\boldsymbol{T} = {}^a_b\boldsymbol{R} \cdot \boldsymbol{X} + \boldsymbol{t} \tag{2.1} $$

Le modèle homogène permet de simplifier les calculs en incluant l'addition au sein de la multiplication, devenant ainsi (2.2)

$$ {}^a_b\boldsymbol{T} = \begin{pmatrix} {}^a_b\boldsymbol{R} & \boldsymbol{t} \\ \mathbb{0}_{1\times 3} & 1 \end{pmatrix} \cdot \begin{pmatrix} \boldsymbol{X} \\ 1 \end{pmatrix} \tag{2.2} $$

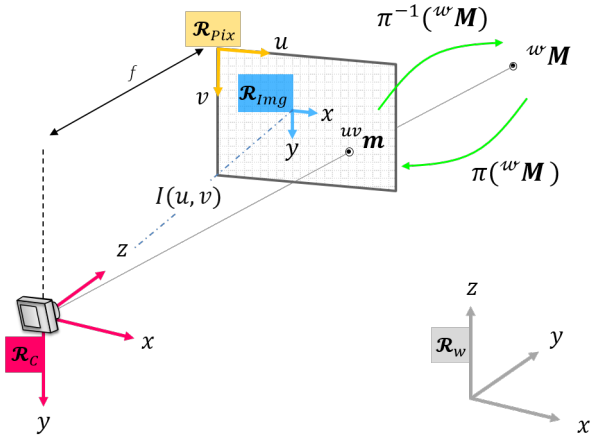


FIGURE 2.4 – Modèle sténopé

La projection d'un point ${}^{w}\boldsymbol{M}$ dans le repère $\boldsymbol{\mathcal{R}}_{pix}$ est définie par ${}^{uv}\boldsymbol{m} = \begin{pmatrix} u & v \end{pmatrix}^T$ où ${}^{uv}\boldsymbol{m}$.

On définit également ${}^{pix}\boldsymbol{m} = \begin{pmatrix} u & v & depth \end{pmatrix}^T$ avec $(u, v) \in \mathbb{R}^2$ les coordonnées de la projection du point ${}^{w}\boldsymbol{M}$ dans $\boldsymbol{\mathcal{R}}_{pix}$. Le troisième paramètre $depth \in \mathbb{R}$ indique l'information de profondeur du point dans l'image, c'est à dire la distance entre ${}^{uv}\boldsymbol{m}$ et ${}^{w}\boldsymbol{M}$. La définition de ${}^{uv}\boldsymbol{m}$ est équivalente à ${}^{pix}\boldsymbol{m}$ sans l'information de profondeur.

Le modèle sténopé met en équation les points observés issus d'un environnement 3D projetés sur le capteur d'image, comme illustré Fig. 2.5. Considérons que le centre de projection du capteur, également définit par l'origine du repère caméra $\boldsymbol{\mathcal{R}}_C$. La distance focale f décrit la distance entre le centre optique ${}^{C}\boldsymbol{O}$ et le plan image donnée par ${}^{Img}\boldsymbol{O}$. Le point ${}^{Img}\boldsymbol{m} = {}^{Img}\begin{pmatrix} x & y \end{pmatrix}^T$ peut être réécrit dans $\boldsymbol{\mathcal{R}}_C$ (2.3)

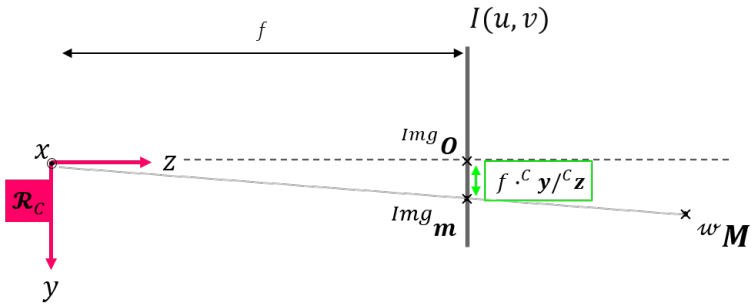


FIGURE 2.5 – Projection d'un point ${}^{w}\boldsymbol{M}$ sur le plan image

$$\begin{aligned} \frac{{}^{Img}x}{f} = \frac{{}^{C}x}{{}^{C}z} &\iff {}^{Img}x = f.\frac{{}^{C}x}{{}^{C}z} \\ \frac{{}^{Img}y}{f} = \frac{{}^{C}y}{{}^{C}z} &\iff {}^{Img}y = f.\frac{{}^{C}y}{{}^{C}z} \end{aligned} \tag{2.3}$$

On peut ainsi décrire sous forme matricielle les coordonnées du point ${}^{Img}\boldsymbol{m}$ selon l'équation (2.4) en passant en coordonnées homogènes.

$${}^{Img}\boldsymbol{m} = \begin{pmatrix} f\,.{}^{C}x/{}^{C}z \\ f\,.{}^{C}y/{}^{C}z \\ 1 \end{pmatrix} \Rightarrow {}^{Img}\boldsymbol{m} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \begin{pmatrix} {}^{C}x \\ {}^{C}y \\ {}^{C}z \\ 1 \end{pmatrix} \tag{2.4}$$

Ainsi ${}^{C}_{Img}\boldsymbol{T}$ la matrice de transformation permettant de projeter un point de de $\boldsymbol{\mathcal{R}}_C$ à $\boldsymbol{\mathcal{R}}_{Img}$ est définie par (2.5)

$${}^{C}_{Img}\boldsymbol{T} = \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \tag{2.5}$$

La fonction de projection (2.4) n'est pas bijective. En effet, ${}^{w}\boldsymbol{M}$ est estimable à partir de ${}^{Img}\boldsymbol{m}$ à un facteur d'échelle près. Une information complémentaire est nécessaire pour pouvoir reconstruire ${}^{w}\boldsymbol{M}$: la profondeur.

Notons $(u_0\, v_0)^T \in \mathbb{R}^2$ les coordonnées de l'origine de $\boldsymbol{\mathcal{R}}_{Img}$, caractérisé par le centre de l'image, dans $\boldsymbol{\mathcal{R}}_{pix}$. Soient $k_u, k_v \in \mathbb{R}^2$ le nombre de pixels par unité de longueur suivant les axes $\boldsymbol{u}$ et $\boldsymbol{v}$ dans $\boldsymbol{\mathcal{R}}_{pix}$. Alors la matrice de transformation ${}^{Img}_{pix}\boldsymbol{T}$ de $\boldsymbol{\mathcal{R}}_{Img}$ à $\boldsymbol{\mathcal{R}}_{pix}$ est définie par (2.6)

$${}^{Img}_{pix}\boldsymbol{T} = \begin{pmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \tag{2.6}$$

Enfin, la matrice de passage ${}^{w}_{C}\boldsymbol{T}$ de $\boldsymbol{\mathcal{R}}_w$ à $\boldsymbol{\mathcal{R}}_C$ décrit une transformation rigide (2.7). Elle définit les *paramètres extrinsèques* à la caméra, c'est-à-dire les paramètres qui ne dépendent

pas directement du capteur mais de l'environnement dans lequel il évolue. ${}^{w}_{C}\boldsymbol{T}$ caractérise les 6 *paramètres extrinsèques* de la caméra : 3 rotations données par les angles d'euler et 3 translations. Son inverse est notée ${}^{w}_{C}\boldsymbol{T}^{-1}$.

$$ {}^{w}_{C}\boldsymbol{T} = \begin{pmatrix} {}^{w}_{C}\boldsymbol{R} & \boldsymbol{t} \\ \mathbb{0}_{1\times 3} & 1 \end{pmatrix} \quad et \quad {}^{w}_{C}\boldsymbol{T}^{-1} = {}^{C}_{w}\boldsymbol{T} = \begin{pmatrix} {}^{w}_{C}\boldsymbol{R}^{T} & -{}^{w}_{C}\boldsymbol{R}^{T}\cdot\boldsymbol{t} \\ \mathbb{0}_{1\times 3} & 1 \end{pmatrix} \tag{2.7} $$

Les différents repères peuvent être schématisés sous forme de combinaison de matrices de passage illustré figure 2.6.

$$ {}^{w}\begin{pmatrix} x & y & z \end{pmatrix} \xrightarrow{{}^{w}_{C}\boldsymbol{T}} {}^{C}\begin{pmatrix} x & y & z \end{pmatrix} \xrightarrow{{}^{C}_{Img}\boldsymbol{T}} {}^{Img}\begin{pmatrix} x & y \end{pmatrix} \xrightarrow{{}^{Img}_{pix}\boldsymbol{T}} {}^{pix}\begin{pmatrix} u & v \end{pmatrix} $$

FIGURE 2.6 – Passage d'un repère à l'autre

Ainsi, le modèle sténopé complet est décrit par la transformation du repère des coordonnées pixelliques $\boldsymbol{\mathcal{R}}_{pix}$ à $\boldsymbol{\mathcal{R}}_{w}$ (2.8) où $f_u = k_u \cdot f$ et $f_v = k_v \cdot f$ et $\boldsymbol{\mathcal{K}}$ définit la *matrice de calibration.*

$$ \begin{aligned} {}^{pix}\begin{pmatrix} u \\ v \\ 1 \end{pmatrix} &= \begin{pmatrix} k_u & 0 & u_0 \\ 0 & k_v & v_0 \\ 0 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} {}^{w}_{C}\boldsymbol{R} & \boldsymbol{t} \\ \mathbb{0}_{1\times 3} & 1 \end{pmatrix} \cdot {}^{w}\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \\ &= {}^{Img}_{pix}\boldsymbol{T} \cdot {}^{C}_{Img}\boldsymbol{T} \cdot {}^{w}_{C}\boldsymbol{T} \cdot {}^{w}\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \\ &= \underbrace{\begin{pmatrix} f_u & 0 & u_0 & 0 \\ 0 & f_v & v_0 & 0 \\ 0 & 0 & 1 & 0 \end{pmatrix}}_{\boldsymbol{\mathcal{K}}} \cdot \begin{pmatrix} {}^{w}_{C}\boldsymbol{R} & \boldsymbol{t} \\ \mathbb{0}_{1\times 3} & 1 \end{pmatrix} \cdot {}^{w}\begin{pmatrix} x \\ y \\ z \\ 1 \end{pmatrix} \end{aligned} \tag{2.8} $$

Nota Bene La *matrice de calibration* $\mathcal{K}$ définit l'ensemble des paramètres intrinsèques au capteur caméra. Ces paramètres dépendent de la nature même du capteur. Lorsque les images sont issues d'un simulateur numérique, autrement dit que les images sont synthètiques, $\mathcal{K}$ est calculée à partir du champ de vue du simulateur, en anglais *Field of View* (FOV), et également de la taille numérique de l'image (en pixel). La figure 2.7 illustre le champ de vue d'un système de vision pour une image de taille $h \times l$. En particulier, la coupe de l'image permet de faire intervenir les différent paramètres f, l, FOV sur un plan.
Ainsi, f_u et f_v sont paramètres de FOV_x et FOV_y respectivement (2.9).

$$\begin{aligned} FOV_x &= 2\arctan(l/(2f_u)) \\ FOV_y &= 2\arctan(h/(2f_v)) \end{aligned} \tag{2.9}$$

Par définition, le point principal est au centre de l'image. Ainsi, ces coordonnées sont définies par 2.10.

$$\begin{aligned} u_0 &= l/2 \\ v_0 &= h/2 \end{aligned} \tag{2.10}$$

d'où

$$\begin{aligned} f_u &= \frac{l/2}{\tan(FOV_x)} = \frac{u_0}{\tan(FOV_x)} \\ f_v &= \frac{h/2}{\tan(FOV_y)} = \frac{v_0}{\tan(FOV_y)} \end{aligned} \tag{2.11}$$

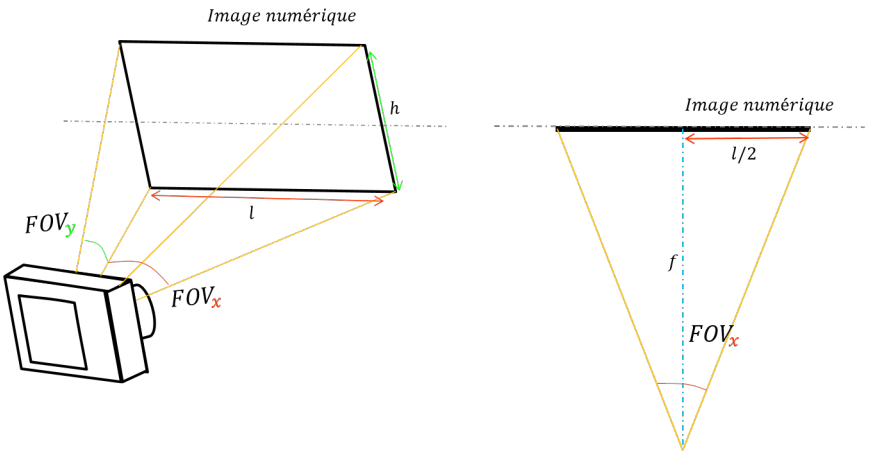


FIGURE 2.7 – Représentation des champs de vue selon les axes $\boldsymbol{x}$ et $\boldsymbol{y}$

2.2.2 Calibrage de la caméra

Le modèle sténopé ne considère pas de déformation optique. En pratique, une lentille est placée au niveau de l'ouverture de la caméra, ouverture laissant entrer les rayons lumineux,

et induit une déformation de leurs trajets. Une phase de calibrage [MR07a] est donc appliquée en amont de tout expérimentation pour déterminer les paramètres intrinsèques de la caméra ainsi corriger les images prises avant tout processus d'extraction d'information ou de fusion de donnée. Les stratégies de calibration sont multiples. Un processus de calibration usuel [Bur16][Zha00] consiste à rectifier la déformation optique dûe à la lentille du capteur, puis à identifier les paramètres de la matrice de calibration $\boldsymbol{\mathcal{K}}$. En complément, les matrices rigides d'homogénéisation de la position de la caméra par rapport à la position de capteurs complémentaires (GPS, LIDAR, IMU, Caméras, etc ...) peuvent être déterminées lors de cette calibration. Dans la suite de notre étude, on suppose le modèle de déformation optique ainsi que les paramètres intrinsèques de la caméra constants au cours d'une expérimentation donnée.

2.2.3 Reconstruction du mouvement

L'objectif de l'odométrie visuelle est déterminer la pose du système mobile à chaque instant et ainsi retracer le mouvement du système mobile. Notre système mobile est un système rigide.

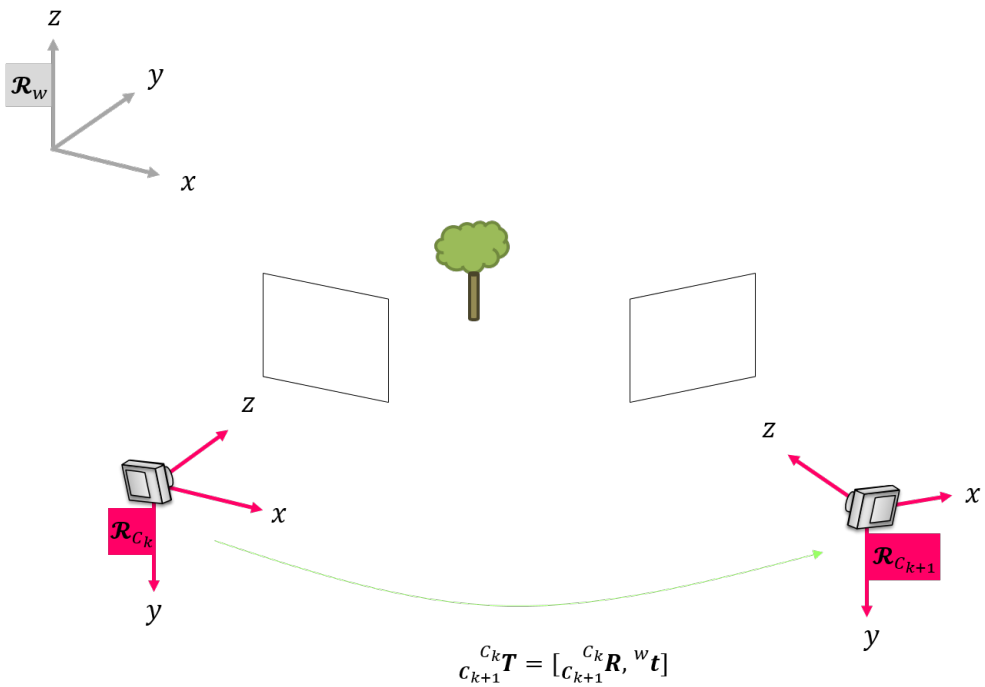


FIGURE 2.8 – Transformation rigide décrivant le mouvement de la caméra à partir des images à k et $k+1$

La transformation rigide qui lie une image capturée à l'instant k à la suivante en $k+1$ se traduit par la composition d'une rotation ${}^{C_k}_{C_{k+1}}\boldsymbol{R} \in \mathbb{R}^{3\times3}$ et d'une translation ${}^{w}\boldsymbol{t} \in \mathbb{R}^{3\times1}$ dans $\boldsymbol{\mathcal{R}}_w$. Dans notre cas, $\boldsymbol{\mathcal{R}}_w$ est fixé par la première position du capteur caméra associée à $\boldsymbol{\mathcal{R}}_C$, le repère caméra, avec lequel il est confondu. Soit ${}^{w}\boldsymbol{p}_{C_k} \in \mathbb{R}^3$, respectivement ${}^{w}\boldsymbol{p}_{C_{k+1}} \in \mathbb{R}^3$, la position de la caméra à l'instant k, respectivement $k+1$ dans le repère $\boldsymbol{\mathcal{R}}_w$. La transformation homogène (Fig. 2.8) décrivant le mouvement de la caméra de $\boldsymbol{\mathcal{R}}_{C_k}$ à $\boldsymbol{\mathcal{R}}_{C_{k+1}}$ s'exprime donc

par ${}^{C_k}_{C_{k+1}}\boldsymbol{T}$ décrit équation (2.12).

$$ {}^{C_k}_{C_{k+1}}\boldsymbol{T} = {}^{C_k}_{C_{k+1}}\boldsymbol{R} \cdot {}^{w}\boldsymbol{p}_{C_k} + {}^{w}\boldsymbol{t} \tag{2.12} $$

où ${}^{C_k}_{C_{k+1}}\boldsymbol{R} \in \mathbb{R}^{3\times3}$ est la rotation de $\boldsymbol{\mathcal{R}}_{C_k}$ à $\boldsymbol{\mathcal{R}}_{C_{k+1}}$, et ${}^{w}\boldsymbol{t} \in \mathbb{R}^{3\times1}$ est la translation associée. La position de la caméra ${}^{w}\boldsymbol{p}_{C_{k+1}}$ à $t+1$ s'exprime alors (2.13) en coordonnées homogènes.

$$ \begin{aligned} \begin{pmatrix} {}^{w}\boldsymbol{p}_{C_{k+1}} \\ 1 \end{pmatrix} &= \begin{pmatrix} {}^{C_k}_{C_{k+1}}\boldsymbol{R} & {}^{w}\boldsymbol{t} \\ \mathbb{0}_{1\times3} & 1 \end{pmatrix} \cdot \begin{pmatrix} {}^{w}\boldsymbol{p}_{C_k} \\ 1 \end{pmatrix} \\ &= \begin{bmatrix} {}^{C_k}_{C_{k+1}}\boldsymbol{R}, & {}^{w}\boldsymbol{t} \end{bmatrix} \cdot \begin{pmatrix} {}^{w}\boldsymbol{p}_{C_k} \\ 1 \end{pmatrix} \end{aligned} \tag{2.13} $$

La fonction de projection $\pi(\mathbb{R}^3, SE(3)) \rightarrow \mathbb{R}^3$, illustrée Fig. 2.4, décrit la projection du point ${}^{pix}\boldsymbol{m}_k$ dans l'image $k+1$ donnée par (2.14).

$$ {}^{pix}\boldsymbol{m}_{k+1} = \pi \left({}^{w}\boldsymbol{M}, {}^{C_{k+1}}_{w}\boldsymbol{T} \right) \tag{2.14} $$

où ${}^{C_{k+1}}_{w}\boldsymbol{T} = {}^{C_{k+1}}_{C_k}\boldsymbol{T} \circ {}^{C_k}_{w}\boldsymbol{T}$ et ${}^{C_k}_{w}\boldsymbol{T}$ est définie (2.7). $\circ$ est l'opérateur de composition des matrices.

2.3 Mesure Visuelle

2.3.1 Analyse d'une image

Une camera numérique génère en sortie une image encodée sur 3 canaux, correspondant aux trois pics de sensibilité de l'oeil : le rouge, le vert et le bleu. Une image est encodée sur $n \times m \times 3$ cases mémoirse. Un pixel est l'ensemble de trois composantes RVB.

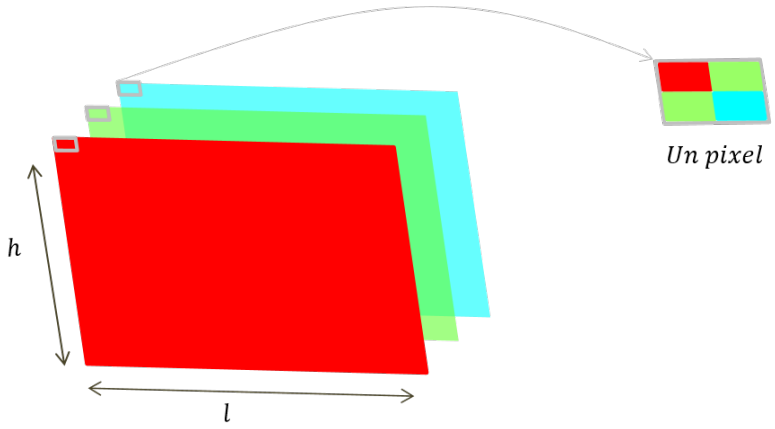


FIGURE 2.9 – Vue explosée de l'encodage RVB d'une image

Un processus courant pour analyser une image issue d'un flux vidéo consiste à extraire de l'information utile puis la mettre en correspondance, c'est-à-dire en relation avec les images suivantes. En particulier, il s'agit de sélectionner des amers ou points saillants, appelés *points d'intérêt*, puis à capturer l'évolution de la position dans $\boldsymbol{\mathcal{R}}_{pix}$ des points d'intérêt sur les images suivantes.

2.3.2 Extraction des points d'intérêt

2.3.2.1 Méthode de détection

La sélection des points d'intérêt dans une image revient à identifier les zones ayant des caractéristiques visuelles (des angles saillants, de la texture, des objets, etc ...). L'utilisation d'un *détecteur* permet d'appliquer cette sélection et d'identifier des points ou des contours remarquables dans une image. De nombreux algorithmes d'extraction de points existent, les plus importants sont identifiés dans le tableau 2.1 selon [Chi+16].

Méthode de détection	Nature	Temps de calcul
Harris [HS88]	Points	++
Shi-Tomasi [JT94]	Points	++
Canny [Joh86]	Contours	++
FAST (Features from Accelerated Segment Test) [RD06]	Points	+
BRIEF (Binary Bobust Bndependent Elementary) [Cal+10]	Points	+
ORB (ORriented FAST and rotated BRIEF) [Rub+11]	Points	+
SURF (Speed up robust feature) [BTG06]	Points	-
SIFT (Scale-Invariant Feature Transform) [Low99]	Points	–
A-KAZE [AS11]	Points	-

TABLE 2.1 – Comparatif des méthodes de détection de points d'intérêt.

Nous avons évalué chaque méthode de détection référencée tableau 2.1. Cette évaluation a été réalisée avec des images réelles issues d'une caméra fisheye, et également des images de synthèse issues d'un simulateur Thales, dans le but d'évaluer la robustesse ainsi que la complexité de chaque méthode. A l'issue de cette étude, nos résultats montrent que les détecteurs de Harris [HS88], Shi-Tomasi [JT94] et Canny [Joh86] sont les méthodes les moins coûteuses en terme de calcul et donc facilement intégrables dans un système en temps réel. FAST [RD06], BRIEF [Cal+10] et ORB [Rub+11] sont les meilleurs compromis entre précision et complexité alors que SURF [BTG06], SIFT [Low99] et A-KAZE [AS11] sont plus robustes mais bien plus coûteux. Parmi les méthodes les moins coûteuses, Shi-Tomasi comme Harris sont équivalents. Dans la suite de notre étude, le détecteur de Harris est utilisé. Son fonctionnement est donc présenté.

Détecteur de Harris Le détecteur de Harris est basé sur le calcul du gradient de l'image qui permet de déterminer la variation spatiale de la luminosité dans l'image. Pour cela, l'image est d'abord passée en niveau de gris. Selon [FS97], le calcul de la dérivée spatiale de l'image dans $\boldsymbol{\mathcal{R}}_{pix}$ peut être approximé par la convolution de l'image $\boldsymbol{I} \in \mathbb{R}^{n \times m}$ avec les opérateurs de Sobel définis par (2.15), où $(n, m) \in \mathbb{R}^2$ représentent les dimensions de l'image. Un exemple figure 2.10 illustre une application de filtre de Sobel pour le cas simple d'un rectange.

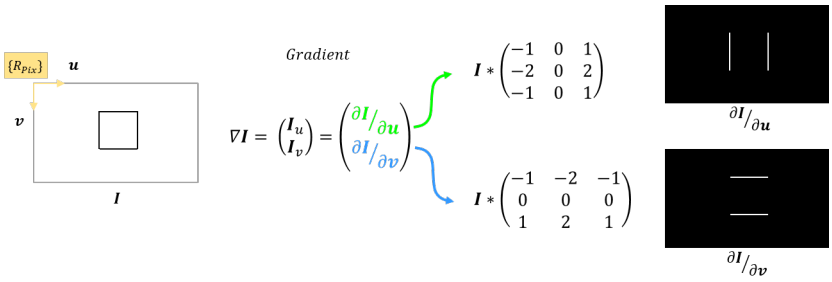


FIGURE 2.10 – Exemple d'application d'un filtre de Sobel sur l'image $\boldsymbol{I}$

$$\begin{aligned} \mathbf{sobel}_u &= \begin{pmatrix} 1 & 2 & 1 \end{pmatrix}^T \\ \mathbf{sobel}_v &= \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}^T \end{aligned} \tag{2.15}$$

Une fois l'estimation de la variation spatiale appliquée à chaque pixel (équation 2.16), Harris [HS88] détermine la matrice Hessienne définie par l'équation 2.17, où $*$ est le produit de convolution.

$$\begin{aligned} \mathbf{I}_u(u,v) &= \mathbf{I}(u,v) * \mathbf{sobel}_u * \mathbf{sobel}_v^T \\ \mathbf{I}_v(u,v) &= \mathbf{I}(u,v) * \mathbf{sobel}_v * \mathbf{sobel}_u^T \end{aligned} \tag{2.16}$$

$$\mathbf{H}(u,v) = \begin{pmatrix} \sum \mathbf{I}_u^2(u,v) & \sum \mathbf{I}_u(u,v)\mathbf{I}_v(u,v) \\ \sum \mathbf{I}_u(u,v)\mathbf{I}_v(u,v) & \sum \mathbf{I}_u^2(u,v) \end{pmatrix} \tag{2.17}$$

Puis [HS88] définit le critère de Harris (2.18) et introduit un seuil permettant de sélectionner les points dont le gradient spatial est le plus important.

$$Harris(u,v) = det(\mathbf{H}(u,v)) - k * trace^2(\mathbf{H}(u,v)) > Seuil \tag{2.18}$$

où $k = 0.04$ selon Harris et Stephens.

Enfin, les points sont ordonnés par ordre croissant, autrement dit les premiers points ont la valeur la plus forte du critère de Harris. Par ailleurs, afin d'éviter la sur-concentration des points sans un voisinage, un critère de distance entre deux points sélectionnés est appliqué. Les points restants définissent les *points d'intérêt*.

Nota Bene Les paramètres de Harris λ_1 et λ_2 sont les valeurs propres de la matrice Hessienne $\mathbf{H}$. En conséquence, $det(Harris(u,v)) = \lambda_1 \cdot \lambda_2$ et $trace(Harris(u,v)) = \lambda_1 + \lambda_2$. Les paramètres de Harris sont un indicateur de la saillance d'un point : se situe-t-il sur une zone de rupture franche ou plutôt *smooth* ? Lorsque λ_1 ou λ_2 est plus grand devant son homologue, le détecteur observe un point le long d'une ligne. A l'inverse, si λ_1 et λ_2 sont du même ordre de grandeur, le point observé se situe sur un coin ou un angle saillant.

Par la suite, l'expression *mesures visuelles* fera référence à la position de ces points remarquables dans l'image dans le repère $\boldsymbol{\mathcal{R}}_{pix}$.

2.3.2.2 Mise en correspondance des données et suivi de points

L'évolution temporelle des mesures visuelles permet la reconstruction du mouvement de la caméra. Les mesures visuelles seules, prises à un instant t, donc issues d'une même détection de Harris, sont décorrélées des observations précédentes. Elles ne contiennent aucune information de mouvement. En conséquence, il est nécessaire de mettre en correspondance les mesures visuelles de façon temporelle afin de capturer le déplacement local des points dans $\boldsymbol{\mathcal{R}}_{pix}$ pour remonter à la trajectoire du système.

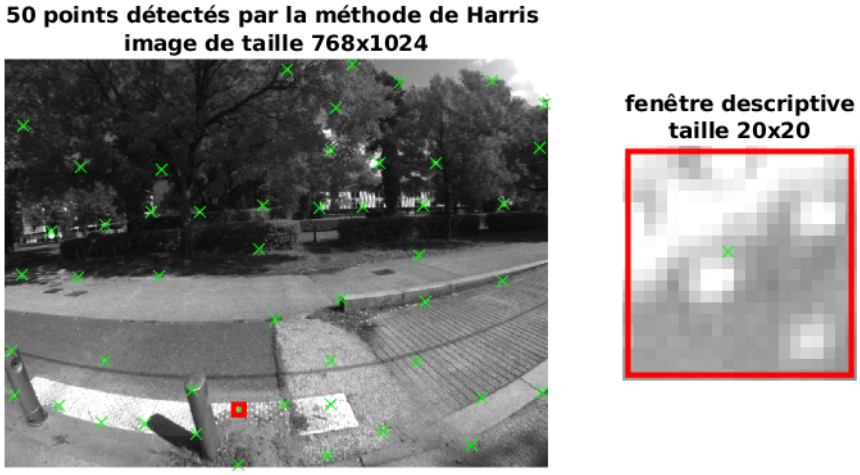


FIGURE 2.11 – Exemple d'un descripteur associé à un point détecté à l'instant. Le descripteur utilisé est une fenêtre de 20×20 pixels. En x, les points d'intérêts détectés et [] la fenêtre descriptive relative à un point

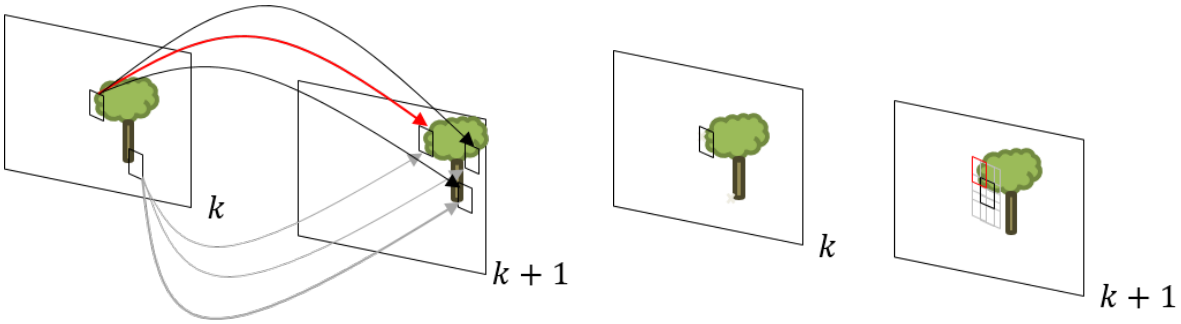
La corrélation des mesures visuelles avec les observations précédentes peut être réalisée par la *matching* de deux *descripteurs* ou par le *suivi de point* d'un *tracker*. Le descripteur décrit le voisinage d'un point à l'issue de sa détection. Le plus généralement cela se traduit par une fenêtre, de taille fixe, au voisinage du point comme représenté figure 2.11. Cette fenêtre est donc une portion de l'image, communément prise en nuance de gris. Ce descripteur est associé à chaque point détecté. Il peut être utilisé lors d'un *matching* (*i.e.* mise en correspondance

de points d'intérêt) ou d'un *tracking* (*i.e.* suivi de points), par exemple. Suivant l'approche de descripteur choisi, celui-ci est robuste à la rotation ou au changement d'échelle lors de l'appariement des points, mais induit un coût de calcul important.

Soit $k \in \mathbb{N}$, l'indice du pas de discrétisation dt du temps t.

Nota Bene Il est essentiel de faire la distinction entre le *matching* et le *tracking*.
Matching : L'opération du *matching* permet de mettre en correspondance deux points d'intérêt détectés à k et $k+1$. Il reconnaît et apparie un point observé à k en $k+1$.
Lors de la première détection, les points saillants sont sélectionnés et un descripteur est associé à chaque point détecté. A l'instant d'après, le détecteur est à nouveau appliqué auxquel on associe le même descripteur. La mise en correspondance, ou matching, des points est réalisée par la comparaison de l'apparence des fenêtres descriptives entre les deux instants considérés, par des méthodes telles que SSD (Somme des différences au carrée) ou ZNCC (Corrélation croisée Normalisée à moyenne nulle)[DSMT05]. À la différence de la méthode de *tracking*, le matching opère sa recherche dans toute l'image. Comme illustré Fig. 2.12a, un point détecté à k est comparé à tous les points détectés à $k+1$ sur l'intégralité de l'image. En conséquence, de mauvais appariments, aussi appelé *mismatch*, se produisent régulièrement lors de redondance visuelle dans l'environnement. Par ailleurs, chaque étape d'analyse nécessite une détection puis une description et enfin une mise en correspondance.
Tracking : À l'inverse du matching de deux descripteurs, le **tracking** permet de suivre l'évolution d'un point. On parle alors de *suivi de points*. Il s'agit d'une méthode de mise en correspondance locale itérative. Sous l'hypothèse de petits déplacements, le tracker suppose qu'un point observé à k restera dans le voisinage de k à $k+1$. Contrairement à l'opération de matching, il n'applique qu'une seule détection à k puis recherche le point dans son voisinage de k sur l'image $k+1$ grâce à une fenêtre descriptive (Fig. 2.12b). Le tracker estime la position du point à l'instant d'après en minimisant le gradient spatial et temporel au voisinage du point courant, grâce à l'hypothèse de petits déplacements.
L'utilisation du tracker permet de limiter les mauvais appariements, en limitant la zone de recherche du point à son voisinage, mais également évite une étape de détection complémentaire à l'instant d'après, puisque la détection est directement réalisée par le tracker lors de la minimisation de l'estimation de la position. Selon [Per+15], avec l'hypothèse de petits déplacements entre deux images capturées, le *tracker* surpasse les performances d'un descripteur en terme de robustesse et de vitesse, comme détaillé dans le tableau 2.2.

Mes travaux de recherche m'ont amené à travailler sur un environnement d'images de synthèses géolocalisées dont les objets (arbres, lampadaires, etc..) ont des occurences identiques au sein d'une même image. Dans ce cas, un descripteur est capable d'associer fausses correspondances en considérant l'occurence d'un point. Contrairement au matching,



(a) (b)

FIGURE 2.12 – Comparaison des méthodes de *matching* et *tracking*. (a) le matching compare la fenêtre descriptive d'un point d'intérêt à k à toutes les fenêtres associés aux points détectés en $k+1$. (b) le tracking concentre la recherche du point en $k+1$ dans le voisinage de la position du point en k.

Propriétés	Matching	Tracking
Robuste aux mouvements importants	✓	-
Robuste aux environnements de synthèse	-	✓
Coût de calcul	-	✓
Performance lors de petits mouvements	-	✓

TABLE 2.2 – Comparatif des méthodes de mise en correspondance et suivi de points

le tracking limite la zone de recherche et montre ainsi de meilleures performances sur les images de synthèse. Le suivi de points est la solution retenue. Par la suite l'utilisation du terme *tracking* ou *tracker* fera référence au suivi des points dans $\boldsymbol{\mathcal{R}}_{pix}$.

Le Tracker Lucas Kanade Tomasi Le tracker Lucas Kanade Tomasi [BK81], (KLT[1]), introduit en 1981 est une méthode itérative de suivi de points. Il pose l'hypothèse d'un mouvement suffisament petit entre deux images consécutives pour qu'il soit considéré comme quasi constant au voisinage d'un point. Il repose sur l'identification d'un point et de son voisinage sur l'image de l'instant d'après en étudiant localement le déplacement des pixels dans le temps (Fig. 2.12b).

L'évolution de la position de la caméra associée à $\boldsymbol{\mathcal{R}}_C$ est exprimée par le pas de temps t. En supposant une cohérence dans la matière de l'environnement observé, la luminosité d'un objet entre deux instances est quasi constante. En effet, l'intensité lumineuse d'un point détecté à l'instant t est alors égale à la l'intensité lumineuse du point à sa nouvelle position dans l'image $t + dt$, comme l'exprime l'équation 2.19. Soit $\boldsymbol{I} \in \mathbb{R}^{n \times m}, (n, m) \in \mathbb{R}$ une image dans $\boldsymbol{\mathcal{R}}_{pix}$.

$$\boldsymbol{I}(u, v, t) = \boldsymbol{I}(u + du, v + dv, t + dt) \tag{2.19}$$

En appliquant un décomposition de Taylor au premier ordre, on obtient alors (2.20).

1. *Kanade Lucas Tomatsi tracker*

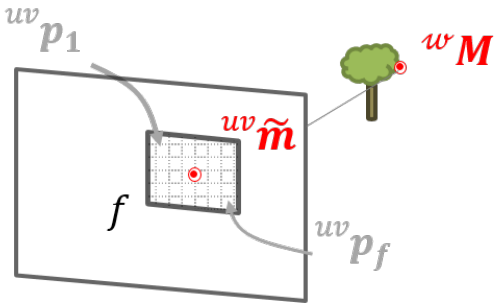


FIGURE 2.13 – Fenêtre f centrée en ${}^{uv}\tilde{\boldsymbol{m}}$ la position du point tracké par KLT issue de ${}^{w}\boldsymbol{M}$. Les points autour de ${}^{uv}\tilde{\boldsymbol{m}}$ contenus dans f sont notés de ${}^{uv}\boldsymbol{p}_1$ à ${}^{uv}\boldsymbol{p}_f, f \in \mathbb{N}$

$$\boldsymbol{I}(u+du, v+dv, t+dt) \approx \boldsymbol{I}(u,v,t) + \boldsymbol{I}_u \cdot du + \boldsymbol{I}_v \cdot dv + \boldsymbol{I}_t \cdot dt \tag{2.20}$$

Puis, en réarrangeant les termes, il en découle (2.22) qui permet d'introduire la variation des déplacements des points suivis au cours du temps $\frac{du}{dt}$ et $\frac{dv}{dt}$ exprimant la vitesse de déplacement des points dans $\boldsymbol{\mathcal{R}}_{pix}$ $\boldsymbol{V}_u$ et $\boldsymbol{V}_v$ ou encore le *flux optique*.

$$\boxed{\boldsymbol{I}(u+du, v+dv, t+dt) - \boldsymbol{I}(u,v,t) = \boldsymbol{I}_u \cdot du + \boldsymbol{I}_v \cdot dv + \boldsymbol{I}_t \cdot dt} \tag{2.21}$$

Dans des conditions idéales, en particulier (2.19) traduit une évolution fixe de l'image au cours du temps, fonction d'un déplacement donné, en supposant l'invariance d'intensité lumineuse. En d'autres termes, on est capable de retrouver exactement la même image à $t+dt$ décalée de du et dv. Donc théoriquement $\boldsymbol{I}(u+du, v+dv, t+dt) - \boldsymbol{I}(u,v,t) = 0$. En réalité, des changements d'intensité lumineuse sont observés.

$$\begin{aligned} 0 &\approx \boldsymbol{I}_u \cdot du + \boldsymbol{I}_v \cdot dv + \boldsymbol{I}_t \cdot dt \\ \iff 0 &\approx \boldsymbol{I}_u \cdot \frac{du}{dt} + \boldsymbol{I}_v \cdot \frac{dv}{dt} + \boldsymbol{I}_t \cdot \frac{dt}{dt} \\ \iff 0 &\approx \boldsymbol{I}_u \boldsymbol{V}_u + \boldsymbol{I}_v \boldsymbol{V}_v + \boldsymbol{I}_t \\ \iff 0 &\approx \nabla \boldsymbol{I} \begin{pmatrix} \boldsymbol{V}_u \\ \boldsymbol{V}_v \end{pmatrix} + \boldsymbol{I}_t \end{aligned} \tag{2.22}$$

Finalement, l'équation 2.23 représente le système d'équation de Lucas-Kanade sur une fenêtre au voisinage du point ${}^{uv}\tilde{\boldsymbol{m}}_k$ issu de ${}^{w}\boldsymbol{M}$ projeté au pas de discrétisation k. Soit ${}^{uv}\tilde{\boldsymbol{m}}_k$ la position du point tracké par KLT, et soit $f \in \mathbb{N}$, $({}^{uv}\boldsymbol{p}_1, \cdots, {}^{uv}\boldsymbol{p}_f) \in \mathbb{R}^2$ les coordonnées des f points autour du point suivi ${}^{uv}\tilde{\boldsymbol{m}}_k$ dans $\boldsymbol{\mathcal{R}}_{pix}$ dans la fenêtre f, illustré (Fig. 2.13).

$$\nabla \boldsymbol{I} \begin{pmatrix} \boldsymbol{V}_u \\ \boldsymbol{V}_v \end{pmatrix} = -\boldsymbol{I}_k$$
$$\begin{pmatrix} \boldsymbol{I}_u({}^{uv}\boldsymbol{p}_1) & \boldsymbol{I}_v({}^{uv}\boldsymbol{p}_1) \\ \vdots & \vdots \\ \boldsymbol{I}_u({}^{uv}\boldsymbol{p}_f) & \boldsymbol{I}_v({}^{uv}\boldsymbol{p}_f) \end{pmatrix} \begin{pmatrix} \boldsymbol{V}_u \\ \boldsymbol{V}_v \end{pmatrix} = - \begin{pmatrix} \boldsymbol{I}_k({}^{uv}\boldsymbol{p}_1) \\ \vdots \\ \boldsymbol{I}_k({}^{uv}\boldsymbol{p}_f) \end{pmatrix} \tag{2.23}$$

On identifie un système linéaire de la forme $\boldsymbol{Ax} = \boldsymbol{b}$ avec 2 inconnues et f équations. En appliquant la méthode de résolution des moindres carrées, la solution au système est donnée par $\boldsymbol{d} = \begin{pmatrix} \boldsymbol{V}_u & \boldsymbol{V}_v \end{pmatrix}^T = (\boldsymbol{A}^T\boldsymbol{A})^{-1} \cdot \boldsymbol{A}^T\boldsymbol{b}$ où $(\boldsymbol{A}^T\boldsymbol{A})$ correspond à la matrice Hessienne définie équation 2.17. Une implémentation est présentée [Bou] en 2001.

Posons $f_{KLT} : (\mathbb{R}^2, \mathbb{R}^{n\times m}, \mathbb{R}^{n\times m}) \to \mathbb{R}^2$, avec $(n, m) \in \mathbb{R}$ les dimensions de l'image $\boldsymbol{I}$, pour représenter le processus de suivi qui estime la position du point $\tilde{\boldsymbol{m}}_{k+1}$ dans $\boldsymbol{I}_{k+1}$ à partir de $\tilde{\boldsymbol{m}}_k$, comme décrit dans [BK81] :

$${}^{uv}\tilde{\boldsymbol{m}}_{k+1} = f_{KLT}({}^{uv}\tilde{\boldsymbol{m}}_k, \boldsymbol{I}_k, \boldsymbol{I}_{k+1}) \tag{2.24}$$

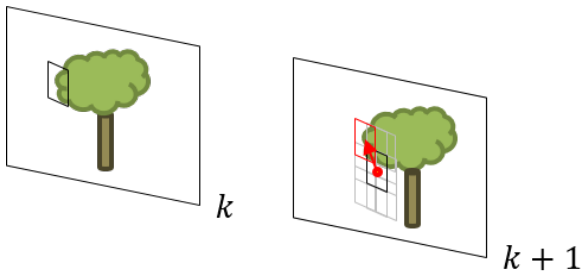


FIGURE 2.14 – Illustration d'un suivi de point par *tracking*. À l'instant k, un point est détecté ; puis à l'instant $k + 1$, le flux optique est évalué, représenté par la flèche rouge.

2.3.3 Imprécision du suivi de points d'intérêt

Nous avons vu jusqu'à présent les étapes d'estimation du mouvement dans l'image. En particulier, nous avons défini un modèle de caméra dont les paramètres intrinsèques sont contenus dans la matrice de calibration, déterminée lors de la phase de calibrage, appliquée aux points d'intérêt détectés (§ 2.3.2.1), puis suivi par *tracking* (§ 2.3.2.2), pour reconstruire le mouvement du système mobile.

Cette capacité de localisation n'est pas absolue. Elle souffre d'une imprécision due à l'accumulation de petites erreurs au cours du temps lors du suivi des points d'intérêt. En particulier, l'imprécision sur l'estimation de la position des points d'intérêt est régie par le niveau de bruit dans l'image capturée ainsi que le constraste et la luminosité de la scène, mais également de l'environnement dans lequel évolue le système (vue uniforme ou distinction d'aspérités), et du modèle de *tracking* ou *matching* considéré. L'ensemble de ces facteurs impactent la précision

du suivi de points d'intérêt, induisant un biais lors de l'estimation du mouvement traduit par la dérive de la trajectoire estimée du système mobile.

Plusieurs approches permettent de limiter cet effet de dérive, parmi lesquelles la corrélation des informations tel que l'apport de données issues d'un capteur extérieur, le recalage par fermeture de boucle, c'est-à-dire le second passage du système mobile à une position déjà observée, ou encore l'utilisation d'une carte de l'environnement. Toutefois, elles requièrent toutes un apport d'information complémentaire.

Enfin, des méthodes de minimisation globale permettent également de réduire l'erreur sur l'estimation de position.

Bien que de nombreuses solution de minimisation de l'erreur de position du système mobile existent, il y aura toujours des erreurs de localisation. Caractériser ces erreurs est peut-être plus important que minimiser l'erreur elle-même.

2.4 Modèle d'erreur

2.4.1 Définition de l'incertitude

L'incertitude définit le doute que soulève une méconnaissance d'une situation. Elle caractérise l'incapacité à trancher sur le choix des événements issus de plusieurs possibilités, complémentaires ou rivales. L'incertitude décrit la situation dans laquelle on ne peut se reposer sur une certitude fondée avec assurance. En mathématiques, elle décrit l'éloignement de l'évenement assuré dont la probabilité est 1, qui peut néanmoins être précisé. L'incertitude renvoie souvent à une forme de risque. On a tendance à la saisir dans son assise pratique : si je suis face à une situation incertaine, je suis exposé à des aléas qui peuvent me blesser et avoir raison de ma destinée. [BCL14] fait la distinction entre risque, dont la probabilité de risques est quantifiable, et l'incertitude, qui soulève une inconnue intangible qui ne peut être anticipée.

> ***Nota Bene*** L'incertitude englobe plusieurs notions qu'il est important de distinguer.
> **Fiabilité** : la fiabilité caractérise la véracité d'une information. Elle indique si une donnée est vraie, fausse ou inexistante. La fiabilité est une notion clé pour les GPS qui souffrent de problèmes de rebonds.
> **Précision** : la précision quantifie la probabilité de l'écart de l'évènement assurée définie par la vérité terrain. Elle décrit l'écart entre la mesure et la réalité.

La notion d'incertitude se réfère ici à la précision accordée aux observations mesurées. Elle induit un indicateur de confiance.

Intervalle de confiance Soit X_G, la vérité terrain. La précision qualifie l'intervalle auquel on peut associer une probabilité telle que la vérité terrain X_G soit contenue dans cet intervalle $[X_L; X_H]$.

Soit $X \in \mathbb{R}$ une variable aléatoire réelle, alors la probabilité sur l'intervalle $[X_L; X_H]$ est donnée par :

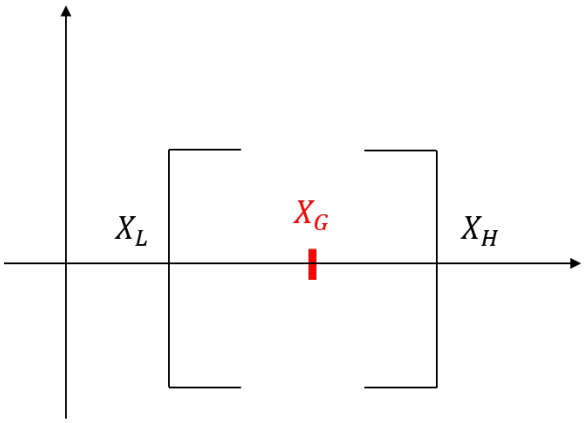


FIGURE 2.15 – Intervalle de confiance $[X_L; X_H]$ encadrant la vérité terrain X_G

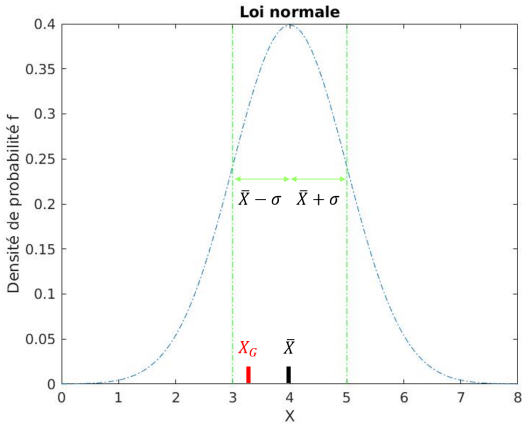


FIGURE 2.16 – Densité de probabilité f de la variable aléatoire X centré en $\bar{X}$

$$P\left(X_L \leq X \leq X_H\right) = \int_{X_L}^{X_H} f(X)dX \tag{2.25}$$

où $f(.)$ est la densité de probabilité associée à X.

L'espérance de X, notée $\mathbb{E}(X)$, est alors définie par :

$$\mathbb{E}(X) = \int_{-\infty}^{+\infty} X \cdot f(X)dX \tag{2.26}$$

D'après le théorème de König-Huyghens, on à la variance de X, $VAR(X)$, donnée par :

$$VAR(X) = \mathbb{E}(X^2) - \mathbb{E}(X)^2 \tag{2.27}$$

Dans le cas d'une loi de probabilité normale sur l'intervalle $[X_L; X_H]$, l'écart entre la valeur moyenne $\bar{X}$ et l'écart-type σ est calculé. Il s'agit d'évaluer si la vérité terrain X_G est

inclue dans l'invertalle $[\bar{X} - \sigma; \bar{X} + \sigma]$, autrement dit il s'agit de calculer la probabilité $P\left(\bar{X} - \sigma \leq X \leq \bar{X} + \sigma\right)$, telle que représentée à la Fig. 2.16. σ est l'écart-type définit par (2.28).

$$\sigma(X) = \sqrt{VAR(X)} \tag{2.28}$$

2.4.1.1 Loi de probabilité normale

La matrice de covariance définit l'écart entre deux variables aléatoires par rapport à leurs espérances. La matrice de covariance, notée $\mathbf{\Sigma}$, indique la confiance accordée à chacune de nos données.

Soit $\boldsymbol{X} = \begin{pmatrix} X_1 \\ \vdots \\ X_n \end{pmatrix} \in \mathbb{R}^n$ un vecteur de taille $n \in \mathbb{R}$. $\boldsymbol{X}$ suit une loi de probabilité normale de valeur moyenne $\boldsymbol{\mu}$ et covariance $\mathbf{\Sigma}$. La loi normale multivariée est notée $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma})$ et sa densité de probabilité s'écrit (2.29).

$$\mathcal{N}(\boldsymbol{\mu}, \mathbf{\Sigma}) = f_{\boldsymbol{\mu}, \mathbf{\Sigma}}(\boldsymbol{X}) = \frac{1}{\sqrt{(2\pi)|\Sigma|}} \exp\left[-\frac{1}{2}(\boldsymbol{X} - \boldsymbol{\mu})^T \mathbf{\Sigma}^{-1}(\boldsymbol{X} - \boldsymbol{\mu})\right] \tag{2.29}$$

où $\boldsymbol{\mu} \in \mathbb{R}^n$ définit le vecteur moyen et $\mathbf{\Sigma} \in \mathbb{R}^{n \times n}$ la matrice de covariance (2.30).

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_{11} & \sigma_{21} & \cdots & \sigma_{n1} \\ \sigma_{12} & \ddots & & \\ \vdots & & \ddots & \\ \sigma_{1n} & & & \sigma_{nn} \end{pmatrix} \tag{2.30}$$

avec $\forall (i,j) \in |[1;n]|$, $\sigma_{ij} = \sigma_{ji} = COV(X_i, X_j)$ et $\forall i = j$, $\sigma_{ii} = VAR(X_i)$. Cette matrice est une matrice semi-définie positive, dont une représentation bidimensionnelle est donnée à la Fig.2.17.

La matrice $\mathbf{\Sigma}$ permet d'exprimer la précision de chaque mesure. Dans le cas d'une variable aléatoire bidimensionnelle, la loi normale peut être représentée par une ellipse. La précision est évaluée par la répartition des points dans l'ellipse à 1σ, 2σ et 3σ où σ est l'écart-type de la distribution définie (2.31) :

$$\sigma = \sum_{i=1}^{n} \sqrt{\sigma_{ii}} = \sum_{i=1}^{n} \sqrt{VAR(Xi)} \tag{2.31}$$

D'après la Fig.2.17, 68% des points sont observés dans l'ellipse à 1σ, 95% à 2σ et 99% à 3σ inclut presque la totalité des points de la distribution. L'estimation est donc adaptée et précise.

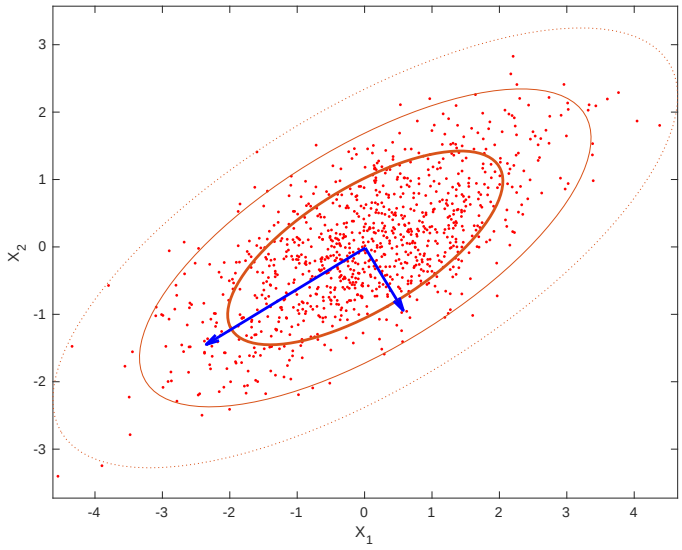


FIGURE 2.17 – Représentation des ellipses à 1σ, 2σ et 3σ d'une covariance bidimensionnelle issue d'une loi normale multivariée

2.4.2 Modèle d'erreur pour le tracking

Un levier important à l'amélioration de la navigation est l'estimation de la précision des informations pour la fusion des données. Dans le cas d'une caméra, l'incertitude associée aux mesures visuelles est souvent prise identique pour toutes les observations issues d'un même capteur. En effet, on observe que la plupart des algorithmes de navigation actuels se basent sur une estimation fixe de la covariance pour toutes les données issues d'un même capteur [MAMT15][HJA10][Str12]. Or, dans une image, la précision sur la position d'un point d'intérêt dépend de la nature de son environnement (flou, peu de texture, faible luminosité, etc ..).

Dès lors, une information bruitée aura le même impact qu'une donnée issue d'un contour net. L'objectif est donc de qualifier de façon adaptée la précision de chacune des observations.

La précision décrit l'écart statistique entre la mesure et la réalité. Dans le cas des mesures optiques, définies par des points d'intérêt (section 2.1), la précision est caractérisée par l'*erreur* entre le point estimé ${}^{uv}\tilde{\boldsymbol{m}}$ lors de l'extraction de l'information et la vérité terrain, représentée à la Fig. 2.18. La vérité terrain est modélisée par la reprojection d'un point ${}^{w}\boldsymbol{M}$, dont la position est connue dans $\boldsymbol{\mathcal{R}}_w$, dans $\boldsymbol{\mathcal{R}}_{pix}$.

Soit ${}^{uv}\boldsymbol{e}_k \in \mathbb{R}^2, k \in \mathbb{N}^2$ l'erreur relative au point ${}^{uv}\tilde{\boldsymbol{m}}_k$ à l'instant k dans $\boldsymbol{\mathcal{R}}_C^k$ (2.32)

$$ {}^{uv}\boldsymbol{e}_k = {}^{uv}\boldsymbol{m}_k - {}^{uv}\tilde{\boldsymbol{m}}_k \tag{2.32} $$

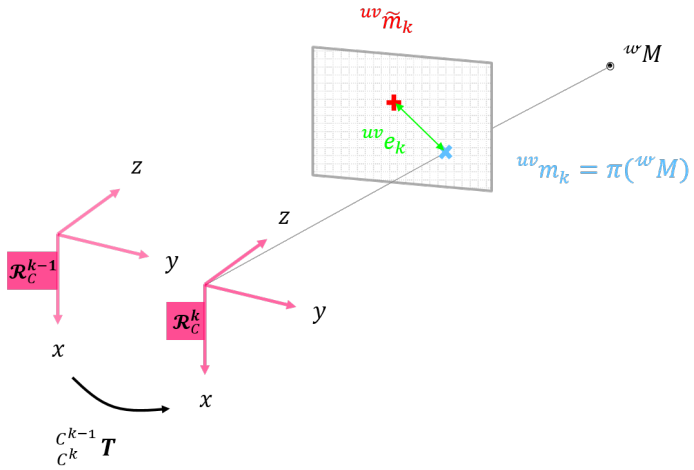


FIGURE 2.18 – Définition de l'erreur représentée par la distance verte entre le point estimé $^{uv}\tilde{\boldsymbol{m}}_k$ en rouge et la vérité terrain $^{uv}\boldsymbol{m}_k$ en bleu, entre deux images consécutives $\boldsymbol{\mathcal{R}}_C^{k-1}$ et $\boldsymbol{\mathcal{R}}_C^k$

Nota Bene Il est important de distinguer la différence entre les valeurs aberrantes, en anglais *outliers*, et les *erreurs*. Les *outliers* sont des fautes qui polluent l'estimation de précision des mesures caractérisées par l'erreur. Cette modélisation de l'erreur évalue la précision d'un point estimé. Elle ne prend pas en compte les fautes de *tracking* ou *matching* qui sont couramment corrigées par des solutions robustes et existantes telle que RANSAC (RANdom SAmple Consensus) [FB81]

Construction de la matrice de covariance des observations Soit $^{uv}\tilde{\boldsymbol{m}}_k \in \mathbb{R}^2$ le point estimé dans l'image $\boldsymbol{\mathcal{R}}_C^k$, et $^{uv}\boldsymbol{e}_k$ l'erreur associée. On suppose que l'erreur $^{uv}\boldsymbol{e}_k$ a un comportement gaussien de valeur moyenne nulle. Alors $^{uv}\boldsymbol{e}_k \sim \mathcal{N}(0, \boldsymbol{\Sigma})$ tel que

$$\boldsymbol{\Sigma} = \begin{pmatrix} \sigma_{uu} & \sigma_{vu} \\ \sigma_{uv} & \sigma_{vv} \end{pmatrix} \tag{2.33}$$

La matrice de covariance $\boldsymbol{\Sigma} \in \mathbb{R}^{2\times 2}$ associée au point $^{uv}\tilde{\boldsymbol{m}}$ est symétrique par définition. Elle est donc caractérisée par trois paramètres.

2.5 Conclusion

Nous avons présenté l'approche classique de l'odométrie visuelle à partir de la détection et du suivi de points d'intérêt dans les images. Cette méthode permet de retracer le positionnement au fil des déplacements de la caméra.

Cependant, elle s'appuie en général sur l'hypothèse d'une incertitude fixe sur la position des points d'intérêt. Autrement dit, l'erreur associée aux points d'intérêt est la même sur

toute l'image et au fil du temps. Bien que l'hypothèse d'une distribution de l'erreur uniforme soit acceptable dans un environnement quasi statique, elle est limitant dans la plupart des scénarios dynamiques. Nous pensons qu'une meilleure modélisation de l'incertitude, adaptée localement à chaque point d'intérêt, permettra une meilleure localisation de la caméra portée par le système mobile.

Néanmoins, il est impossible d'obtenir la distribution réelle des erreurs de mesures issues des capteurs. Il est plus réaliste de travailler directement avec les erreurs de mesures. Comment estimer ces erreurs et évaluer la précision et la confiance des points d'intérêt observés ?

Par la suite, notre étude se concentrera sur la caractérisation et l'estimation locale des covariances des mesures optiques. En d'autres termes, l'objectif est d'associer à chaque mesure de points d'intérêt de $\boldsymbol{\mathcal{R}}_{pix}$ une covariance de dimension 2×2.

Chapitre 3

Modèles d'incertitudes caractérisant les points d'intérêt

> Le premier savoir est le savoir de mon ignorance : c'est le début de l'intelligence.
>
> Socrate

Ce chapitre fait l'état de l'art des modèles d'incertitudes des mesures visuelles.

3.1 Incertitude locale sur le suivi des points d'intérêt

Dans le chapitre précédent, nous avons vu que l'incertitude peut faire référence à la précision et à la fiabilité [ZJM20]. Dans notre travail, nous nous concentrons sur la précision, en supposant que les points d'intérêt non fiables sont filtrés par des méthodes existantes et robustes comme RANSAC [FB81].

Seuls quelques travaux ont été consacrés à l'estimation de l'incertitude locale sur la position des points d'intérêt dans une image [ZMG22]. Dans [KK01], l'effet de la matrice de covariance est évalué et le modèle d'erreur choisi considère une distribution d'erreur uniforme sur toute l'image et à travers le temps. Cette hypothèse suppose que toutes les observations sont non corrélées, ce qui est devenu un postulat courant [Zhu+19a] ; [Zhu+19b]. Par ailleurs, [ZT17] montre que l'estimation adaptée de la covariance associée aux points d'intérêt dans l'image, souvent surdimensionné, améliore de façon significative la navigation d'un système intégrant les données visuelles dans un fitre de kalman.

Certaines approches analytiques sont proposées pour capturer le modèle d'erreur. En particulier, [WM17] repose sur la linéarisation de la fonction de l'algorithme d'appariement du modèle de VO. Dans [WM17], les auteurs estiment l'incertitude sur la position des points suivis par le trackeur KLT [BK81]. L'inconvénient principal de cette approche analytique est la dépendance de l'estimateur à l'algorithme d'appariement spécifié. Une généralisation à toutes les méthodes éparses, comme dense, est difficile à envisager. Par ailleurs, [WM17] fixe une covariance initiale, ce qui borne l'évolution de l'incertitude. [Urf+06] propose une linéarisation de l'erreur de reprojection. Cependant, le bruit dans l'image n'est pas considéré.

Les modèles analytiques permettent de capturer une partie du problème. Toutefois, il est difficile de modéliser entièrement l'erreur de suivi qui englobe à la fois le bruit de l'image, mais également les perturbations de l'environnement, la dynamique de la scène observée ou

encore la nature de la scène [Sze90] (urbain, forêt, etc..). Des techniques d'apprentissage permettent de surmonter ce problème.

3.2 Modèles par apprentissage profonds et incertitudes

L'avènement récent des réseaux neuronaux profonds [LBH15] ; [Qin+22] ; [Hua19] a conduit à l'émergence de nombreux modèles d'erreur qui évaluent l'incertitude de la pose d'un système mobile [Li+22] ; [ZC21] ; [DML20] ; [Yan+20] ; [Wan+18] ; [Liu+18] ; [PK17] ; [Wal+17]. Parmi les premiers modèles d'estimation de l'incertitude de la pose, [Wan+18] se démarque en proposant un système de navigation entièrement conçu avec des méthodes d'apprentissages, de l'extraction des features à la construction de la pose. En particulier, il introduit l'utilisation de la loi normale multivariée pour estimer l'incertitude sur la pose. Cette méthode d'estimation de l'incertitude est repris par la suite par [Liu+18] pour apprendre la matrice de covariance liée au bruit de mesure de la pose du système mobile à partir d'un flux d'images monoculaire. [PK17] s'inspire de ces travaux pour construire un modèle qui estime l'incertitude sur la pose puis corrige la pose calculée afin de réduire la dérive temporelle. [DML20] combine [Liu+18] et [PK17] par l'ajout de complexité dans la construction de l'architecture du réseau de neurones.

Bien que montrant des résultats encourageants, ces modèles d'estimation de l'incertitude sont appliqués à la caractérisation de la précision de la pose et sont donc inadaptés à des solutions d'hybridation serrées. Ils réduisent l'erreur sur la pose appliquée à une hybridation lâche qui est, par définition, sujette à plus d'erreurs de modèle d'approximation que les fusions *serrées*[1].

Dans le domaine de l'apprentissage profond, la plupart des efforts sont consacrés à la résolution de problème à l'échelle des objets [Cui+22] ; [Yan+22]. Certaines fonctions d'appariement locales [DMR18] ; [Han+15] ; [PS21] tenant compte de la position des points d'intérêt sont devenues accessibles au public mais sont rarement couplées à des modèles d'incertitude de leurs prédictions. En parallèle, des travaux sur la caractérisation de l'incertitude et des modèles d'erreurs [Ken19] émergent utilisant des réseaux de neurones. Malgré la disponibilité de tous ces modèles, un problème majeur persiste : la mesure de l'erreur des points d'intérêt.

3.3 Incertitudes visuelles dans le contexte de la navigation

Les sources d'erreur des mesures dans les images peuvent être de différentes natures : issues de l'environnement extérieur, générées lors de la calibration, dues au bruit optique, ou encore à la saturation du capteur. Dans la littérature, nous venons de voir qu'apparaîssent des méthodes d'apprentissage pour l'estimation de covariance dans le cadre d'une navigation par odométrie visuelle, telles que dans [Wan+18], [PK17] et [DML20]. Ces méthodes sont globales, autrement dit elles donnent une information de confiance sur le calcul de la pose du système mobile. [Moh+19] montre qu'une fusion *serrée*, par optimisation ou filtrage, admet de meilleurs résultats qu'une approche *lâche*, comme c'est le cas avec la VO.

L'incertitude sur la position d'un point suivi est nécessaire pour fusionner avec un couplage

1. c'est-à-dire fusionnant les mesures directes et non une information intermédiaire issue des mesures (ie. pseudo-distances vs position absolue pour l'exemle d'un GPS)

serré l'estimation du mouvement. En effet, une fusion serré, c'est-à-dire une fusion appliquée au niveau des mesures, implique l'utilisation de matrice de bruit de mesure notamment dans l'utilisation d'un filtre de Kalman, afin de propoger les états. Ainsi la caractérisation locale des mesures caméras est essentielle.

Une approximation populaire considère une incertitude égale de toutes les observations dans l'image [MAMT15] ; [Tan+22]. En particulier, les observations visuelles sont considérées comme indépendantes dans l'espace et le temps [MR07b]. Cette hypothèse est acceptable dans un environnement statique ou quasi-statique, mais devient inadéquate dans un environnement dynamique. En effet, des observations avec des amplitudes très différentes engendreront une surconfiance ou à l'inverse une sousconfiance lors de la fusion des mesures caméras impactant directement l'estimation de la pose. Une mesure d'erreur d'incertitude non adaptée aura un impact sur la précision de l'estimation du mouvement et augmentera l'incertitude de la pose [MS99] ; [AWD10]. La plupart des solutions existantes aujourd'hui n'estiment pas l'incertitude des caractéristiques locales des mesures visuelles, ce qui limite les méthodes d'optimisation ou de filtrage.

Par ailleurs, lorsqu'une image est bruitée, même partiellement, il est courant de ne pas la prendre en compte dans l'estimation de la pose et de passer à la suivante. L'étude locale des covariances permet d'extraire de l'information utile d'une image bruitée et pondérer ces nouvelles observations avec une précision adaptée. L'apport d'information réduit alors la dérive du système mobile dans le temps.

Certains systèmes incluent un modèle d'erreur visuelle pour minimiser l'erreur de reprojection avec l'erreur IMU [Leu+15] ; [Leu+13] ; [MAT17] mais ne prennent pas en compte les effets visuels courants (par exemple, le flou optique). Or, la précision des mesures visuelles est conditionnée par les contrastes, la luminosité ou le flou. Ainsi, l'incertitude des mesures d'un scénario très dynamique, notamment propice aux effets de flou, sera sous évaluée, induisant un biais dans le calcul de pose. D'autres font évoluer l'incertitude avec un modèle empirique partiel [Wan+20].

3.4 Apprentissage par récurrence

Enfin, l'erreur de suivi d'indices visuels grandit et se cumule dans le temps. Une caractérisation précise de l'incertitude fait appel à une notion d'antériorité. Pour prendre en compte l'évolution temporelle de l'incertitude de suivi sur la position d'un point d'intérêt, des modèles récurents existent et montrent de bonnes performances. En particulier, l'introduction des LSTM (Long-Short Term Memory) [HS97] fait émerger de nouvelles solutions d'estimation de la pose, notamment pose-lstm[Wal+17] et poseNet[KGC15]. A l'échelle du pixel, un nouveau descripteur surf-lstm[Elm+20] intégrant des cellules mémoires minimise l'erreur sur la pose à partir des observations des points d'intérêt. L'utilisation de LSTM dans des systèmes de localisation montre une amélioration de la précision de la pose. Toutefois, d'aucun s'attaque au problème de caractérisation de l'incertitude.

3.5 Conclusion

En vue de réaliser une navigation par hybridation serrée, nous étudions les incertitudes locales de la position des points d'intérêt. Seuls quelques travaux ont été réalisés sur l'estimation de l'incertitude locale des points d'intérêt. Les méthodes analytiques existantes se limitent à une modélisation partielle du problème.

Des solutions par réseau de neurones permettent de modéliser l'incertitude. Elles offrent plus de flexibilité sur les fonctions d'appariement auxquelles elles s'appliquent. Toutefois, la majeure partie d'entre elles se concentrent sur l'estimation de l'incertitude de la pose d'un système mobile.

> ***Nota Bene*** Il est important de noter qu'il est difficile de qualifier l'intégrité d'un réseau de neurones. En conséquence, la solution proposée dans ce manuscrit nourrit un algorithme analytique de navigation. L'estimateur d'incertitude utilisant des méthodes d'apprentissage profond apporte une connaissance de la précision sur les mesures, ce qui peut permettre d'améliorer la précision de la localisation, mais n'impacte pas directement la détermination de la localisation du système mobile. Par ailleurs, la détermination des covariances locales permet d'affiner la précision des covariances des amers 3D [BS06] ; [San+17].

Un problème subsiste : l'incertitude locale n'est pas observable directement. Notre solution devra utiliser une méthode indirecte pour apprendre cette incertitude.

Deuxième partie

Contributions

Chapitre 4

Estimation des covariances locales associées à chaque point d'intérêt

Information is the resolution of uncertainty.

Claude Elwood Shannon

In effect, we have redefined the task of science to be the discovery of laws that will enable us to predict events up to the limits set by the uncertainty principle.

Stephen Hawking

L'estimation de l'incertitude des points d'intérêt est essentielle pour les systèmes basés sur la vision, tel que l'odométrie visuelle. En effet, elle permet de qualifier la précision des observations et ainsi réduire les erreurs issues de l'imprécision des mesures. En particulier, dans le contexte de la navigation visuelle, l'estimation de l'incertitude minimise l'impact de l'effet de dérive au cours du temps. L'objectif de cette section est de caractériser un modèle d'erreur des points d'intérêt en fonction de la nature de l'environnement et des algorithmes utilisés. Dans cette section, nous présentons le fonctionnement de base d'un réseau de neurones (RdN) et montrons comment estimer l'incertitude sur la position d'un point d'intérêt. Nous montrons notamment que les erreurs inhérentes au suivi visuel, en particulier en utilisant le tracker Kanade Lucas Tomasi (KLT), peuvent permettre l'estimation des matrices de covariance de la position de chaque point suivi en utilisant la loi de probabilité multivariée. Dans un second temps, la construction de la base de données d'apprentissage est présentée, à partir d'un dataset d'images de synthèse et d'images réelles.

4.1 Description du système complet

Les points d'intérêt correspondent aux positions des caractéristiques visuelles contenues dans l'image (couleur, profondeur, coins, angles etc ...). La détection et le suivi de points d'intérêt dans une séquence d'images est un processus essentiel pour de nombreuses applications basées sur la vision, telles que la reconstruction de cartes, la navigation, la détection et l'évitement d'obstable, ou encore le suivi d'objets. C'est également l'essence de l'odométrie visuelle [SF11], qui, dans son fonctionnement de base, consiste en l'extraction de caractéristiques visuelles, le suivi de caractéristiques et l'estimation du mouvement de la caméra (section 2.1).

La navigation d'un système mobile repose sur une quantification précise du bruit ou de l'incertitude des capteurs afin de produire des estimations d'état fiables. Cependant, la modélisation de l'incertitude sur la position des points d'intérêt nécessite une caractérisation préalable du modèle d'observation de la caméra. Elle inclut notamment une linéarisation des équations du modèle caméra (§2.1), ce qui peut être délicat et coûteux. En pratique, cette incertitude est souvent fixée et constante [MAMT15][Dav03] pour un capteur et une expérience donnés.

Bien qu'une estimation fixe de l'incertitude puisse parfois être une approximation raisonnable dans certains environnements quasi-statiques, lors des scénarios dynamiques elle aura la fonction de filtre face à de multiples effets visuels, tels que le flou de mouvement, le mouvement des objets ou encore les variations d'éclairage. Dès lors, une partie des observations utiles et disponibles est rejetée par une incertitude à seuil fixe inadaptée. En outre, ces observations inutilisées sont considérées comme valeurs aberrantes engendrant un manque d'information dans l'estimation d'état pouvant impacter les performances de l'estimateur. Par ailleurs, la covariance associée à une mesure joue sur la pondération des données lors de la fusion avec les autres données. Le fait d'avoir une incertitude fixe revient à donner le même poids à une donnée imprécise qu'à une autre donnée très précise, ce qui intuitivement n'est pas idéal. Une bonne estimation de l'incertitude a donc un impact majeur sur la performance de l'application envisagée.

La qualification de l'incertitude, c'est-à-dire, selon la définition donnée section 2.4.1, la quantification de la précision permet de pondérer l'impact des observations, en particulier de réduire l'influence des observations bruitées. Idéalement, les observations issues d'une image buitée ou partiellement bruitée sont exploitées avec un niveau de confiance adapté.

La capacité d'estimer précisément l'erreur du processus de suivi des points d'intérêt permet de fusionner correctement les données visuelles et inertielles [MR07b], et également de sélectionner activement les meilleures caractéristiques pour l'estimation du mouvement.

Notre objectif est d'associer un modèle d'erreur à chaque mesure observée. À partir de la chaîne de fonctionnement classique de la VO (Fig. 2.1), le flux d'images est soumis à l'extraction d'information permettant d'extraire les points d'intérêt qui constituent nos observations. La précision de chaque point d'intérêt est par la suite évaluée en lui associant une matrice de covariance qualifiant l'incertitude de la position du point (section 2.4). Cette quantification de confiance, adaptée à chaque observation, est intégrée à l'estimateur de mouvement, comme illustré figure 4.1.

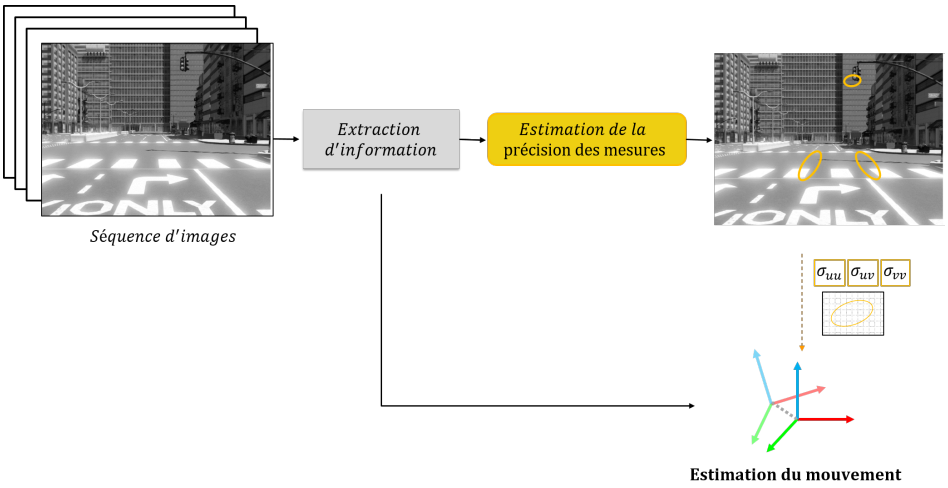


FIGURE 4.1 – Vue d'ensemble du système d'estimation de précision

La plupart des approches d'estimation du mouvement d'un système mobile basé sur l'observation de points d'intérêt supposent que l'incertitude sur la position des points d'intérêt dans l'image est constante [MAMT15]. Cette incertitude est généralement fixée lors de la phase de calibration [Zha00] ou ajustée en post-traitement [Hua19].

Seuls quelques travaux ont été consacrés à l'estimation de l'incertitude des points suivis. Dans [KK01], les auteurs étudient l'effet de la matrice de covariance calculée sur les pixels de l'image sur la précision de la localisation des points caractéristiques. Cependant, le travail n'est pas validé de manière approfondie et les résultats ne sont présentés que sur des points sélectionnés manuellement. À la fin, les auteurs fournissent une valeur de précision par défaut pour chaque caractéristique. Dans [WM17], les auteurs proposent une approche basée sur la linéarisation du tracker [BK81] de Kanade Lucas Tomasi (KLT) pour estimer l'incertitude des points suivis. La méthode est basée sur une approximation incrémentale, initialisée avec une incertitude fixe, qui limite l'évolution de l'incertitude. Il s'agit d'une approche basée sur la dérivation qui dépend du suiveur KLT.

Nous proposons de développer un système d'estimation de l'incertitude basé sur l'utilisation de réseau de neurones (RdN). L'avantage de cette approche est son adaptabilité. En effet, elle ne dépend que des données utilisées lors de l'apprentissage et est donc applicable à toutes les méthodes d'extraction et de matching ou tracking de points d'intérêt. Les données utilisées pour l'apprentissage du réseau sont constituées des points d'intérêt et des valeurs des pixels de l'image autour de chaque point d'intérêt. Nous allons dans un premier temps évaluer comment retrouver l'incertitude de la position d'un point à partir des points d'intérêt. Ce qui nous conduira à définir une fonction de coût pour minimiser l'ensemble des incertitudes de position. Enfin, nous caractériserons ces données d'apprentissage, en particulier leur nature et leur format. Il s'agit de déterminer les informations nécessaires pour constituer la base de données d'apprentissage.

Dans le domaine de l'apprentissage profond, la plupart des efforts sont consacrés à la résolution du problème de suivi au niveau de l'objet [Cui+22] ; [Yan+22]. Certaines approches pour

le suivi de points et la correspondance de patchs sont également devenues accessibles au public [Han+15] ; [PS21] mais sont rarement associées à des modèles d'incertitude de leurs prédictions.

4.2 Modélisation de l'incertitude de la position d'un point

La covariance associée à la position d'un point ${}^{uv}\tilde{\boldsymbol{m}}$ dans le plan focal image n'est pas observable ni mesurable par un capteur. Elle est généralement approximée et constante pour toutes les mesures d'un capteur caméra donné [MAMT15][HJA10][Str12] par estimation de l'erreur de mesure moyenne. En particulier, elle est couramment fixée à 0.5 pixels de confiance pour chaque point d'intérêt observé [KK01]. Dès lors, une information bruitée aura le même impact qu'une donnée saine.

L'estimation de la covariance est réalisée par le biais de l'observation de l'erreur. Nous avons vu section 2.4 la définition de l'incertitude, notamment la distinction entre la notion de fiabilité, qui qualifie des mesures aberrantes pouvant être filtrées par des algorithmes tel que RANSAC[FB81], et de précision. La covariance est un indicateur de la précision d'une mesure, autrement dit elle décrit l'écart entre la mesure et la réalité. Cet écart est formulé par l'*erreur* entre le point estimé ${}^{uv}\tilde{\boldsymbol{m}}$ lors de l'extraction de l'information et la vérité terrain ${}^{uv}\boldsymbol{m}$. Soit ${}^{uv}\boldsymbol{e} \in \mathbb{R}^2$ l'erreur relative au point ${}^{uv}\tilde{\boldsymbol{m}}$ (4.1).

$$ {}^{uv}\boldsymbol{e} = {}^{uv}\boldsymbol{m} - {}^{uv}\tilde{\boldsymbol{m}} \tag{4.1} $$

L'objectif de notre travail est d'estimer une covariance adaptée pour chacune des observations de points d'intérêt. Toutefois, il est impossible d'obtenir la distribution réelle des erreurs de mesures issues des capteurs caméras : la vérité terrain n'existe pas. Il est plus réaliste de travailler directement avec les erreurs de mesures. Comment estimer ces erreurs et évaluer la précision et la confiance des point d'intérêt observés ?

Notre objectif est d'estimer $\hat{\boldsymbol{\Sigma}}$ associée au point ${}^{uv}\tilde{\boldsymbol{m}}$ à partir de la connaissance de la position de ${}^{uv}\tilde{\boldsymbol{m}}$.

FIGURE 4.2 – Fonctionnement du système souhaité où à chaque point ${}^{uv}\tilde{\boldsymbol{m}}_i$ est associé une matrice de covariance prédite $\hat{\boldsymbol{\Sigma}}_i$ par un réseau de neurones (RdN)

4.2.1 Régression par réseau de neurones

Une régression linéaire est définie par un ensemble de N entrées et sorties $(\{\boldsymbol{x}_1, \boldsymbol{y}_1\}, \ldots, \{\boldsymbol{x}_N, \boldsymbol{y}_N\})$ avec $\forall i \in |[1; N]|$, $\boldsymbol{x}_i \in \mathbb{R}^n$ et $\boldsymbol{y}_i \in \mathbb{R}^m$, $(n, m) \in \mathbb{N}^2$, telle que il existe une tranformation linéaire $f(\boldsymbol{x}) = \boldsymbol{x} \cdot \boldsymbol{\mathcal{W}} + \boldsymbol{b}$ où $\boldsymbol{\mathcal{W}} \in \mathbb{R}^{n \times m}$ et $\boldsymbol{b} \in \mathbb{R}^m$. $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ représentent

toutes les combinaisons linéaires possibles. Alors $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ sont fixés par une fonction de minimisation, comme par exemple l'erreur quadratique moyenne telle que $\frac{1}{N}\sum_i ||\boldsymbol{y}_i - (\boldsymbol{x}_i \cdot \boldsymbol{\mathcal{W}} + \boldsymbol{b})||^2$.

En pratique, $\boldsymbol{x}$ et $\boldsymbol{y}$ sont souvent non modélisables par une fonction linéaire. Dans ce cas, une fonction non-linéaire est introduite. Soit $\sigma(.)$ une fonction non linéaire alors

$$f(\boldsymbol{x}) = \sigma(\boldsymbol{x} \cdot \boldsymbol{\mathcal{W}} + \boldsymbol{b}) \tag{4.2}$$

L'équation 4.2 définit le fonctionnement de base d'un réseau de neurones (RdN) dont les principes de fonctionnement seront présentés section 5.1. Finalement, un RdN se résume à un problème de minimisation modélisé par une fonction non-linéaire.

Pour notre étude, nous considérons un réseau de neurones par apprentissage *supervisé* qui, tout comme les régressions linéaires présentées précédemment, applique une fonction de minimisation pour fixer les poids des paramètres du RdN $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ (4.3)

> ***Nota Bene*** L'apprentissage supervisé nécessite la connaissance en amont de la sortie attendue pour une entrée donnée. La relation entre les données d'entrée et de sortie définit le modèle à apprendre. Une fois appris, le modèle est utilisé pour évaluer de nouveaux ensembles de données, non vus, et prédire les résultats.
> À l'inverse, l'apprentissage non-supervisé consiste à construire un modèle à partir d'un ensemble de données dont le résultat attendu n'est pas connu. Il est souvent utilisé pour identifier des modèles et des tendances dans des ensembles de données, pour regrouper des données similaires dans des groupes.

4.2.1.1 Principe et fonctionnement

Dans le cadre d'un apprentissage supervisé, un RdN a deux modes de fonctionnement : l'entrainement et le mode d'inférence. Le mode d'inférence correspond au mode opératoire fonctionnel du réseau. Les poids ont été fixés et ajustés en amont lors de l'entrainement. Pour une entrée donnée, le réseau retourne une prédiction de sortie $\hat{\boldsymbol{y}}$.

Durant l'entrainement, les paramètres $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$, aussi appelés *poids* du réseau, varient et s'ajustent pour associer à une entrée donnée $\boldsymbol{x}$ une sortie attendue $\boldsymbol{y}$. Cette phase compare la prédiction de sortie du réseau $\hat{\boldsymbol{y}}$ avec sa vérité terrain $\boldsymbol{y}$ à travers la fonction de minimisation $\mathcal{L}$, qualifiée de fonction de coût du réseau.

$$\underset{\boldsymbol{\mathcal{W}};\boldsymbol{b}}{\operatorname{argmin}}\, \mathcal{L}(\{RdN(\boldsymbol{x};\boldsymbol{\mathcal{W}};\boldsymbol{b}), \boldsymbol{y}\}) \tag{4.3}$$

La fonction de coût $\mathcal{L}$, donnée par l'équation 4.3, permet de contraindre le RdN et de corriger les paramètres du réseau au cours d'une étape de rétro-propagation à l'issue d'une prédiction. La phase d'entrainement (ou d'apprentissage) est une minimisation itérative.

À l'issue de l'apprentissage, le modèle non linéaire définissant le RdN (5.2) est figé, autrement dit $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ sont connus et fixes. Le réseau peut être évalué en comparant la prédiction $\hat{\boldsymbol{y}}_i$ et la vérité terrain $\boldsymbol{y}_i$.

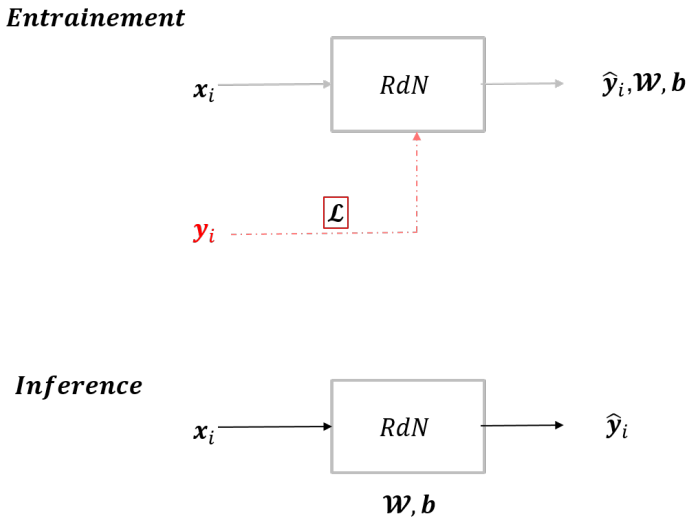


FIGURE 4.3 – Deux modes de fonctionnement du RdN : le mode d'entrainement qui vient minimiser les paramètres du réseau pour le calibrer en comparant la prédiction de sortie $\hat{\boldsymbol{y}}$ avec la vérité terrain $\boldsymbol{y}$ grâce à la fonction de coût $\mathcal{L}$; et le mode d'inférence qui correspond au mode d'utilisation opérationnel du RdN. Il associe une prédiction $\hat{\boldsymbol{y}}$ à une entrée $\boldsymbol{x}$

4.2.1.2 Application

Dans le cadre de l'estimation de l'incertitude de la position des points d'intérêt, la vérité n'est pas connue. En effet, l'incertitude sur la position des points n'est pas observable. Il n'est donc pas possible d'obtenir la distribution réelle des erreurs de mesures associée à un capteur. La fonction de coût $\mathcal{L}$ ne peut donc pas comparer la matrice de covariance prédite $\hat{\boldsymbol{\Sigma}}$ avec la vérité terrain $\boldsymbol{\Sigma}$, tel que illustré Fig. 4.4.

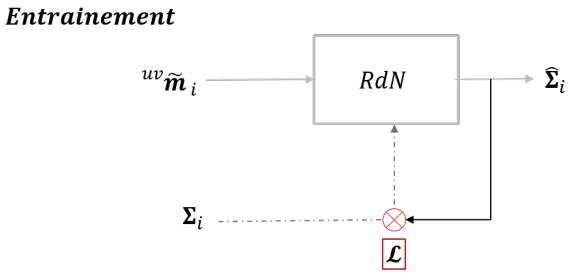


FIGURE 4.4 – RdN minimisant la fonction de coût $\mathcal{L}$ telle que l'écart entre la matrice de covariance prédite $\hat{\boldsymbol{\Sigma}}_i$ et la covariance réelle $\boldsymbol{\Sigma}_i$ soit minimisé.

Contrairement à l'incertitude sur la position d'un point d'intérêt, la mesure de l'erreur sur la position d'un point suivi est observable dans certain cas (voir § 4.2.2). Par ailleurs, l'erreur de position d'un point d'intérêt est représentée par sa matrice de covariance à travers sa densité de probabilité définie (§ 2.4). Ainsi, nous souhaitons définir $\mathcal{L}$ telle que la matrice de covariance prédite $\hat{\boldsymbol{\Sigma}}$ associée au point d'intérêt $^{uv}\boldsymbol{m}$ maximise la probabilité d'obtenir l'erreur mesurée $^{uv}\boldsymbol{e}$ (4.4).

$$\forall i \ , \ \underset{\boldsymbol{\mathcal{W}};\boldsymbol{b}}{\operatorname{argmin}} \mathcal{L}(\{RdN({}^{uv}\boldsymbol{m}_i;\boldsymbol{\mathcal{W}};\boldsymbol{b}),{}^{uv}\boldsymbol{e}_i\}) \tag{4.4}$$

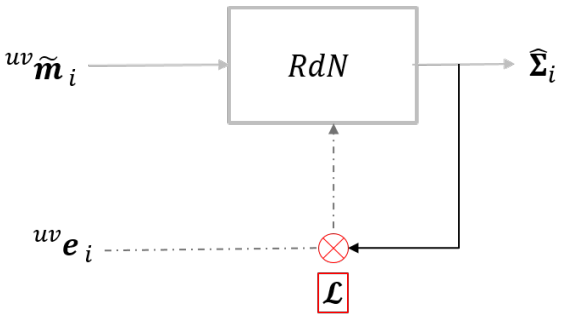


FIGURE 4.5 – RdN minimisant la fonction de coût $\mathcal{L}$ telle que la matrice de covariance prédite $\hat{\boldsymbol{\Sigma}}_i$ associée au point d'intérêt ${}^{uv}\boldsymbol{m}_i$ maximise la probabilité d'obtenir l'erreur ${}^{uv}\boldsymbol{e}_i$.

4.2.2 Mesure de l'erreur

Soit un point 3D ${}^{w}\boldsymbol{M} \in \mathbb{R}^3$ de l'environnement, exprimé dans le repère $\boldsymbol{\mathcal{R}}_w$ (définit Fig. 2.4). Soit ${}^{uv}\boldsymbol{m}_{k+1} \in \mathbb{R}^2$ sa projection dans $\boldsymbol{\mathcal{R}}_{pix}$ à l'instant $k+1$ projetée par la fonction de projection $\pi(\mathbb{R}^3, SE(3)) \rightarrow \mathbb{R}^3$ (§ 2.2.3, eq 4.5). On suppose la position de ${}^{w}\boldsymbol{M}$ connue dans $\boldsymbol{\mathcal{R}}_w$ ainsi que les paramètres intrinsèques et le mouvement de la caméra de k à $k+1$.

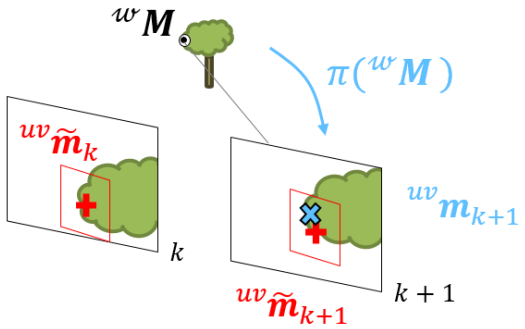


FIGURE 4.6 – Mesure de l'erreur

$$^{pix}\boldsymbol{m}_{k+1} = \pi\left({}^{w}\boldsymbol{M} \ , \ {}^{C_{k+1}}_{C_k}\boldsymbol{T} \circ {}^{C_k}_{w}\boldsymbol{T}\right) \tag{4.5}$$

Soit ${}^{uv}\tilde{\boldsymbol{m}}_k \in \mathbb{R}^2$ le point estimé dans l'image $\boldsymbol{\mathcal{R}}_C^k$. En particulier, l'estimation de la position de ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$ est réalisée par le tracker KLT (Kanade Lucas Tomasi, § 2.3.2.2) (Fig. 2.14). Soit ${}^{uv}\boldsymbol{e}_{k+1} \in \mathbb{R}^2$ (défini eq. 4.1) l'erreur associée à l'écart entre ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$ et ${}^{uv}\boldsymbol{m}_{k+1}$. On suppose que l'erreur ${}^{uv}\boldsymbol{e}_{k+1}$ a un comportement gaussien de valeur moyenne nulle. Soit $\boldsymbol{\Sigma} \in \mathbb{R}^{(2\times 2)}$ la matrice de covariance associée, alors ${}^{uv}\boldsymbol{e}_{k+1} \sim \mathcal{N}(0, \boldsymbol{\Sigma})$. D'après (section 2.4), $\boldsymbol{\Sigma}$ peut s'écrire (4.6) :

$$\mathbf{\Sigma} = \begin{pmatrix} \sigma_{uu} & \sigma_{vu} \\ \sigma_{uv} & \sigma_{vv} \end{pmatrix} \tag{4.6}$$

L'estimation de l'incertitude de la position d'un point revient à minimiser la matrice de covariance $\mathbf{\Sigma}$ associée au point ${}^{uv}\tilde{\boldsymbol{m}}$, où $|\mathbf{\Sigma}|$ est le déterminant de la matrice de covariance $\mathbf{\Sigma}$.

Une solution est de maximiser la vraisemblance d'observer les erreurs de mesure à partir de la distribution sur les erreurs tel que décrit (4.7) :

$$\underset{\mathbf{\Sigma}}{\operatorname{argmax}} P_{\mathbf{\Sigma}}({}^{uv}\boldsymbol{e}) \tag{4.7}$$

On introduit $P_{\mathbf{\Sigma}}({}^{uv}\boldsymbol{e})$ la densité de propabilité de l'erreur définit part (4.8) :

$$P_{\mathbf{\Sigma}}({}^{uv}\boldsymbol{e}) = \frac{1}{\sqrt{(2\pi)|\mathbf{\Sigma}|}} \exp\left[-\frac{1}{2}({}^{uv}\boldsymbol{e})^T \mathbf{\Sigma}^{-1} ({}^{uv}\boldsymbol{e})\right] \tag{4.8}$$

Par définition, la matrice de covariance $\mathbf{\Sigma}$ est symétrique. Elle est également définie positive.

Pour contraindre ce critère, on applique à $\mathbf{\Sigma}$ une décomposition de Cholesky modifiée LDL [HK15] ainsi que la fonction exponentielle à chaque élément de $\boldsymbol{D}$ (4.10). Soit l_1, d_1 et d_2, les paramètres composant les matrices $\boldsymbol{L}$ et $\boldsymbol{D}$ respectivement.

$$\mathbf{\Sigma} = \boldsymbol{L}(l_1)\boldsymbol{D}(d_1, d_2)\boldsymbol{L}(l_1)^T = \begin{pmatrix} 1 & 0 \\ l_1 & 1 \end{pmatrix} \begin{pmatrix} d_1 & 0 \\ 0 & d_2 \end{pmatrix} \begin{pmatrix} 1 & 0 \\ l_1 & 1 \end{pmatrix}^T \tag{4.9}$$

En effet, la fonction exponentielle est définie positive. La décomposition LDL est un produit d'une matrice triangulaire inférieure $\boldsymbol{L}$, et une matrice diagonale $\boldsymbol{D}$, définie par des éléments diagonaux positifs (4.9).

$$\boldsymbol{L}(l_1)\boldsymbol{D}(exp(d_1), exp(d_2))\boldsymbol{L}(l_1)^T = \begin{pmatrix} 1 & 0 \\ l_1 & 1 \end{pmatrix} \begin{pmatrix} e^{d_1} & 0 \\ 0 & e^{d_2} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ l_1 & 1 \end{pmatrix}^T \tag{4.10}$$

Nota Bene l_1, d_1 et d_2 ne sont pas directement égaux aux paramètres σ_{uu}, σ_{uv} et σ_{vv} de la matrice de covariance $\mathbf{\Sigma}$

4.2.3 Définition de la fontion de coût

Considérons $\mathcal{D} = \{ {}^{uv}\boldsymbol{e}_k \,|\, \forall k \in |[1;s]| \}$ l'ensemble des données d'erreurs mesurées issues de s points d'intérêt d'un flux d'images. L'objectif est d'estimer la covariance $\boldsymbol{\Sigma}_k$ à partir de l'erreur mesurée ${}^{uv}\boldsymbol{e}_k$ pour tous $k \in |[1;s]|$.

Sous un modèle de bruit gaussien à moyenne nulle, le maximum de vraisemblance est donné par (4.11)

$$\underset{\boldsymbol{\Sigma}_{1:s} \in \mathbb{R}^{2\times 2}}{\operatorname{argmax}} \sum_{k=1}^{s} P_{\boldsymbol{\Sigma}_k}({}^{uv}\boldsymbol{e}_k) \tag{4.11}$$

où $\Sigma_{1:s}$ représente l'ensemble des matrices de covariance pour le lot de données de 1 à s. L'objectif est de maximiser la probabilité d'obtenir l'erreur ${}^{uv}\boldsymbol{e}_k$. Or, maximiser une fonction est equivalent a maximiser son logarithme et donc minimiser -log [Wan+18].

$$\underset{\boldsymbol{\Sigma}_{1:s} \in \mathbb{R}^{2\times 2}}{\operatorname{argmin}} \sum_{k=1}^{s} -log(P_{\boldsymbol{\Sigma}_k}({}^{uv}\boldsymbol{e}_k)) \tag{4.12}$$

On développe

$$\underset{\boldsymbol{\Sigma}_{1:s} \in \mathbb{R}^{2\times 2}}{\operatorname{argmin}} \sum_{k=1}^{s} -log(\frac{1}{\sqrt{(2\pi)|\boldsymbol{\Sigma}|}} \exp\left[-\frac{1}{2}({}^{uv}\boldsymbol{e})^T \boldsymbol{\Sigma}^{-1} ({}^{uv}\boldsymbol{e})\right]) \tag{4.13}$$

Les termes constants n'interviennent pas dans la minimisation d'une fonction, le logarithme est donc séparé en une somme de deux fonctions logarithmes. On obtient alors :

$$\sim \underset{\boldsymbol{\Sigma}_{1:s} \in \mathbb{R}^{2\times 2}}{\operatorname{argmin}} \sum_{k=1}^{s} -log(\frac{1}{|\boldsymbol{\Sigma}|}) + -log(\exp\left[-({}^{uv}\boldsymbol{e})^T \boldsymbol{\Sigma}^{-1} ({}^{uv}\boldsymbol{e})\right]) \tag{4.14}$$

Finalement,

$$\sim \underset{\boldsymbol{\Sigma}_{1:s} \in \mathbb{R}^{2\times 2}}{\operatorname{argmin}} \sum_{k=1}^{s} log(|\boldsymbol{\Sigma}_k|) + {}^{uv}\,\boldsymbol{e}_k^T (\boldsymbol{\Sigma}_k^{-1}) {}^{uv}\boldsymbol{e}_k \tag{4.15}$$

De façon similaire à [Liu+18], on applique la décomposition en LDL [HK15] décrite eq. (4.10).

$$\boldsymbol{\Sigma}_k = (\boldsymbol{L}(\boldsymbol{l}_k)\boldsymbol{D}(\exp(\boldsymbol{d}_{i,k}))\boldsymbol{L}(\boldsymbol{l}_k))^T) \tag{4.16}$$

Ainsi (4.15) devient (4.17) par simplification du déterminant et de la fonction logarithmique pour un $k \in |[1;s]|$:

$$\begin{aligned} log(|\boldsymbol{\Sigma}|) &= log\left(|\boldsymbol{L}(l_1)\boldsymbol{D}(exp(d_1), exp(d_2))\boldsymbol{L}(l_1)^T| \right) \\ &= log\left(|\boldsymbol{L}(l_1)| \cdot |\boldsymbol{D}(exp(d_1), exp(d_2))| \cdot |\boldsymbol{L}(l_1)^T| \right) \end{aligned} \tag{4.17}$$

Le déterminant de $|\boldsymbol{L}(l_1)|$ vaut 1. En effet, par construction, $\boldsymbol{L}$ est une matrice triangulaire de déterminant 1. Ainsi,

$$\begin{aligned} log(|\boldsymbol{\Sigma}|) &= log\,(\,|\boldsymbol{D}(exp(d_1), exp(d_2))|\,) \\ &= log\,(\,exp(d_1)\,\cdot\,exp(d_2)\,) \\ &= d_1 + d_2 \end{aligned} \tag{4.18}$$

Finalement, la fonction de coût est donnée par Eq. 4.19.

$$\mathcal{L}_{\boldsymbol{\Sigma}_k}({}^{uv}\boldsymbol{e}_k) = \sum_{k=1}^{s}\sum_{i=1}^{2} \boldsymbol{d}_{i,k} + {}^{uv}\boldsymbol{e}_k{}^T \left[(\boldsymbol{L}(\boldsymbol{l}_k)\boldsymbol{D}(\exp(\boldsymbol{d}_{i,k}))\boldsymbol{L}(\boldsymbol{l}_k))^T)^{-1} \right] {}^{uv}\boldsymbol{e}_k \tag{4.19}$$

avec $\boldsymbol{l}_k \in \mathbb{R}^s\,,\ \forall k \in |[1;s]|$ et $\boldsymbol{d}_{i,k} \in \mathbb{R}^{2\times s}\,,\ \forall i \in |[1;2]|\,|\,\forall k \in |[1;s]|$ les paramètres à minimiser tels que les matrices de covariance $\boldsymbol{\Sigma}_k$ maximisent la probabilité d'obtention de l'erreur observée ${}^{uv}\boldsymbol{e}_k$.

En conclusion, la phase d'apprentissage du réseau de neurones pour l'estimation de l'incertitude sur la position des points d'intérêt revient à minimiser $\mathcal{L}_{\boldsymbol{\Sigma}_k}$ (4.20).

$$\forall k\ ,\ \underset{\boldsymbol{\Sigma}_k;\boldsymbol{\mathcal{W}};\boldsymbol{b}}{\operatorname{argmin}}\, \mathcal{L}_{\boldsymbol{\Sigma}_k}(\{RdN({}^{uv}\boldsymbol{m}_k;\boldsymbol{\mathcal{W}};\boldsymbol{b}),{}^{uv}\,\boldsymbol{e}_k\}) \tag{4.20}$$

Nous avons évalué comment retrouver l'incertitude de la position d'un point à partir des points d'intérêt, puis définit une fonction de coût pour minimiser l'ensemble des incertitudes de position. Nous allons à présent caractériser les données d'apprentissage, en particulier leur nature et leur format. Il s'agit de déterminer les informations nécessaires pour constituer la base de données d'apprentissage.

4.3 Base de données d'apprentissage

L'estimation de la covariance n'est pas observable directement par un capteur. Toutefois, elle peut être approximée par l'erreur de mesure directement, qui elle est observable. L'erreur de mesure définit l'écart entre la mesure et la vérité terrain. Comment mesurer cette erreur sans disposer de la vérité terrain d'un point d'intérêt ? C'est l'objet de cette section.

4.3.1 La vérité terrain

Dans les datasets d'odométrie visuelle classique [MAMT15], seule la pose de la caméra est évaluée. En effet c'est la seule vérité terrain disponible. Or, nous souhaitons évaluer l'erreur de suivi des points d'interêts dans les images : il est donc nécessaire de trouver comment obtenir cette information dans les datasets disponibles.

Une première approche consiste à définir la vérité terrain comme étant la projection d'un amer ${}^{w}\boldsymbol{M}$ de $\boldsymbol{\mathcal{R}}_w$ dans $\boldsymbol{\mathcal{R}}_{pix}$. Cette approche suppose de connaître l'environnement dans lequel le système évolue en amont, et en particulier construire une carte dense de profondeur. Le dataset SYNTHIA [Ros+16] fournit des trajectoires enregistrées à partir d'un simulateur. Chaque trajectoire met en association un flux d'images RVB (Rouge-Vert-Bleu) avec des images de profondeur dense. SYNTHIA apporte donc une première solution accessible dans sa mise en œuvre.

4.3.2 SYNTHIA

Présentation SYNTHIA (SYNTHetic collection of Imagery and Annotations) [Ros+16] est une collection d'images de synthèse (Fig. 4.7) accompagnée d'un ensemble de données générées dans le but d'aider à l'apprentissage de contruction de cartes de profondeur, à la segmentation d'objets dans l'image et aux problèmes connexes de compréhension de scènes dans le contexte de scénarios urbains et semi-urbains.

SYNTHIA propose, en particulier, des images denses de cartes de profondeur (Fig. 4.8) associées à chaque image RVB. Un des objectifs est de fournir une vérité terrain à des approches de RdN qui souhaitent construire des cartes de profondeur à partir d'un flux d'images vidéo RVB.

SYNTHIA consiste en un ensemble de scénarios enregistrés d'images réalistes (Fig. 4.7) issues d'un simulateur virtuel. En effet, SYNTHIA évolue dans des environnements urbains et semi-urbains et simule les saisons et les intempéries. Ceci est particulièrement intéressant pour évaluer la robustesse du suivi de points d'intérêt au cours du temps, en injec-

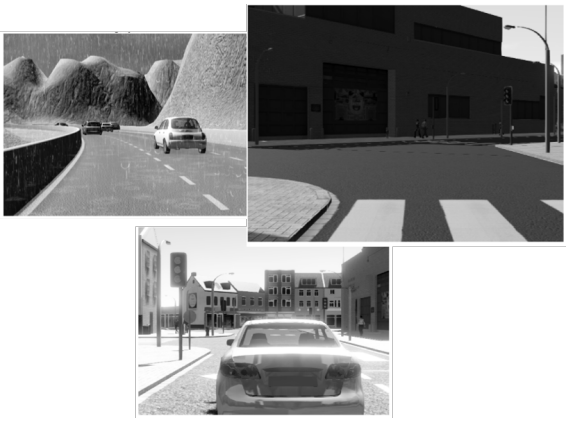


FIGURE 4.7 – SYNTHIA dataset - L'image de gauche montre une scène sous la pluie ; celle du bas simule l'effet d'une surexposition lumineuse [Ros+16]

tant du bruit de façon contrôlée par exemple
et en évaluant l'impact sur le suivi.

La plupart des datasets (*i.e.* KITTI [Gei+13], Euroc [Bur+16], etc ...) utilisés pour des applications de VO sont produits dans des conditions optimales de fonctionnement (ensoleillement, bonne visibilité, texture, en intérieur, etc ..). Par ailleurs, SYNTHIA fournit toutes les caractéristiques de la caméra virtuelle en plus de toutes ses positions et orientations. Dès lors, le déplacement dans $\boldsymbol{\mathcal{R}}_w$ est connu.

FIGURE 4.8 – Image RVB associée à sa carte de profondeur dense

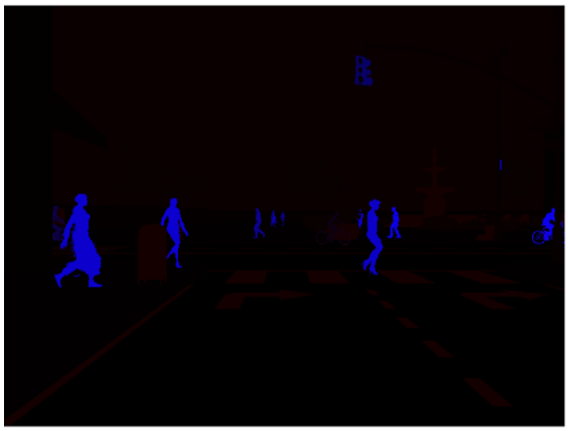


FIGURE 4.9 – Image segmentée des objets dynamiques

Parmi les informations complémentaires associées à chaque image RVB, SYNTHIA offre également des images segmentées (Fig. 4.9) avec des objets identifiés. SYNTHIA évolue dans un environnement dynamique, avec des objets en mouvement. L'identification des objets permet de faire de la segmentation et d'avoir une vérité terrain, mais également de filtrer des erreurs issues de points attachés à des objets en mouvement dont le déplacement ne coïncide pas nécessairement avec le mouvement du système mobile porteur du capteur caméra.

SYNTHIA fournit 13 classes d'objets : terrain, ciel, bâtiment, route, trottoir, clôture, végétation, poteau, voiture, panneau, piéton, cycliste, marquage de voie.

Extraction de l'information L'estimation du mouvement issue de l'odométrie visuelle (VO) repose sur une phase d'extraction et de mise en commun des données entre chaque image d'un flux vidéo (section 2.1). Elle requiert également la connaissance en amont des spécificités

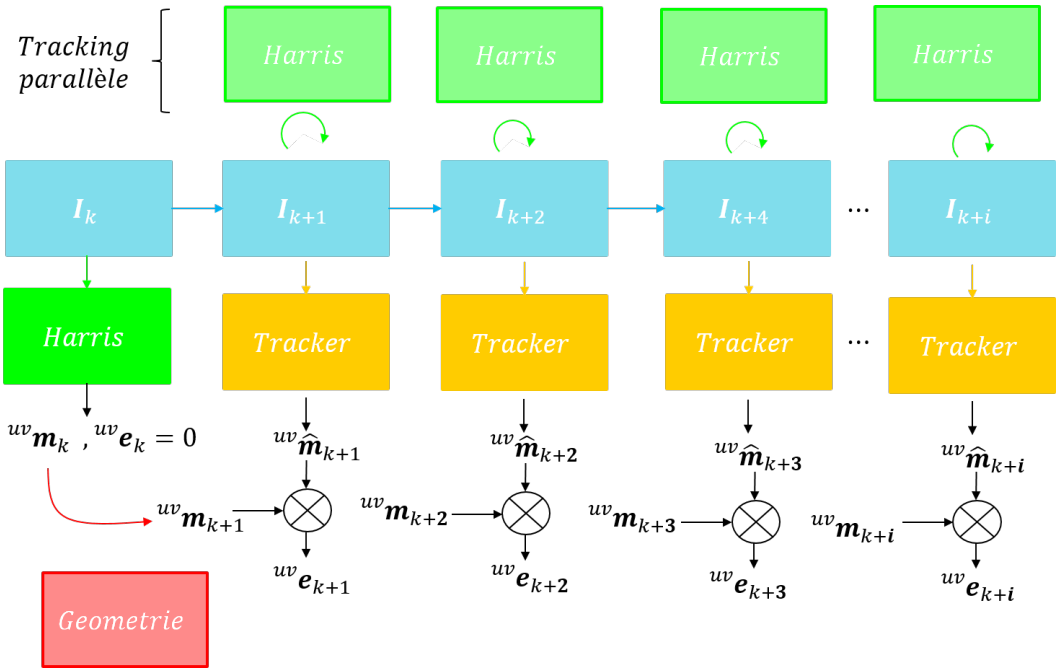


FIGURE 4.10 – Schéma fonctionnelle du procédé de construction de la base de donnée d'apprentissage appliquée au dataset SYNTHIA

de la caméra (*i.e.* les paramètres intrinsèques qui composent la matrice de calibration § 2.2.1). La phase d'extraction est réalisée par un détecteur de points d'intérêt (§ 2.3.2.1). Il existe un grand nombre d'algorithmes de détection pour identifier des points d'intérêt (*i.e.* ORB [MAMT15], SIFT [HMS], etc), parmi lesquels le détecteur de Harris [HS88]. Son faible coût de calcul et ses performances globales font de ce détecteur l'un des plus populaires. Harris se base sur une évaluation du gradient de l'image (§ 2.3.2.1). Il classe le gradient des points saillants dans l'image par ordre croissant et sélectionne les plus élevés. Pour coupler les détecteurs, la littérature fournit des descripteurs appariés (§ 2.3.2.2). Par choix, nous préférons une méthode de suivi basée sur l'étude locale du gradient : le tracker de Kanade Lucas Tomasi (KLT) [BK81]. Selon [Per+15], avec l'hypothèse d'un faible déplacement entre deux images, le tracker dépasse les performances et la vitesse des descripteurs. De plus, il démontre les meilleurs résultats avec des images de synthèse. En effet, la redondance de certains objets de synthèse identiques (*i.e.* arbres, lampadaires, etc...) au sein d'une même image favorise les fautes de correspondance. Dans la suite des travaux, nous utilisons Harris pour sélectionner les points d'intérêt et KLT pour suivre l'évolution du point au cours du temps. On note $f_{KLT}(^{uv}\tilde{\boldsymbol{m}}_k, \boldsymbol{I}_k, \boldsymbol{I}_{k+1})$ (eq. 2.24).

Le processus de suivi d'un point d'intérêt est décrit Fig. 4.10. Soit une image $\boldsymbol{I}_k \in \mathbb{R}^{n \times m}$, avec $(n, m) \in \mathbb{R}$, à un instant k. Chaque point d'intérêt est initialisé par une détection de Harris. Par ailleurs, la position d'un point issue d'une détection de Harris sous-pixellique représente la vérité terrain du point d'intérêt. Ainsi, à $k = 0$, $^{uv}\tilde{\boldsymbol{m}}_k = ^{uv}\boldsymbol{m}_k$ et $^{uv}\boldsymbol{e}_k = ^{uv}\boldsymbol{m}_k - ^{uv}\tilde{\boldsymbol{m}}_k = 0$ (eq. 2.32). $^{uv}\boldsymbol{e}_k$ définit la précision relative au suivi de point KLT, autrement dit la précision de $f_{KLT}(.)$.

Nota Bene En considérant la position d'un point d'intérêt issue du détecteur de Harris comme étant la première mesure de vérité terrain, on suppose que la précision sur la détection est parfaite. En réalité, une incertitude de 0.5 pixel est attribuée au détecteur de Harris. Toutefois, notre étude se concentre sur l'estimation de la précision du suivi de point. La détection de Harris étant réalisée une seule fois lors de l'initialisation d'un nouveau point dans $\boldsymbol{\mathcal{R}}_{pix}$, nous supposons que son impact est moindre contrairement à l'incertitude sur le suivi de point appliqué à chaque itération qui grandit au cours du temps.

Pour $\forall k > 0$, les points d'intérêt initialisés par Harris sont suivis par le tracker KLT à travers la fonction $f_{KLT}(.)$ (eq. 2.24, § 2.3.2.2). Par la suite les points suivis ${}^{uv}\tilde{\boldsymbol{m}}_k$ sont utilisés pour estimer ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$ à travers $f_{KLT}(.)$.

La vérité terrain ${}^{uv}\boldsymbol{m}_k$, initialisée par le détecteur de Harris, est transposée aux images suivantes $\forall k > 0$ grâce à la connaissance de la carte de profondeur, et du mouvement exact de la caméra portée par le système mobile. Pour rappel, on note ${}^{pix}\boldsymbol{m} = \begin{pmatrix} u & v & depth \end{pmatrix}^T$ avec $(u, v) \in \mathbb{R}^2$ les coordonnées du point dans $\boldsymbol{\mathcal{R}}_{pix}$. Le troisième paramètre $depth \in \mathbb{R}$ indique l'information de profondeur du point dans l'image (§ 2.1). Soit la fonction de projection $\pi(\mathbb{R}^3, SE(3)) \rightarrow \mathbb{R}^3$ telle que ${}^{pix}\boldsymbol{m}_k = \pi\left({}^{w}\boldsymbol{M}_k\,,\,{}^{C_{k+1}}_{w}\boldsymbol{T}\right)$ (Fig. 2.4). Dès lors, la reprojection de la vérité terrain observée en k en $k+1$ s'exprime suivant (4.21) noté géométrie sur la Fig. 4.10.

$$ {}^{pix}\boldsymbol{m}_{k+1} = \pi\left(\pi^{-1}({}^{pix}\boldsymbol{m}_k\,,\,{}^{C_k}_{w}\boldsymbol{T})\,,\,{}^{C_{k+1}}_{w}\boldsymbol{T}\right) \tag{4.21} $$

Afin d'alimenter le tracker avec des points d'intérêt tout au long d'une séquence temporelle, un tracking parallèle est appliqué. Il s'agit d'une détection de Harris réalisée en parallèle du tracking décrit précédemment. Les points nouvellements détectés sont ajoutés à la fin du vecteur de points avec un seuil de $N = 1000$ points d'intérêt pour garantir un apport d'information constant.

En vue de générer une base de donnée non polluée par des fautes de détection ou des valeurs aberrantes, tous les points d'intérêt attachés à un objet dynamique ne sont pas pris en compte. Ils sont identifiés par les images segmentées (Fig. 4.9) fournies par le dataset SYNTHIA [Ros+16] et supprimées en appliquant un grossisement autour de l'objet. Cette opération est appliquée après chaque détection, chaque reprojection et chaque tracking KLT. Ainsi lorsqu'un objet en mouvement, tel qu'un vélo, passe devant un point suivi depuis quelques images, le point est supprimé du vecteur de points. Une fois l'objet passé, il peut être de nouveau détecté lors du tracking parallèle.

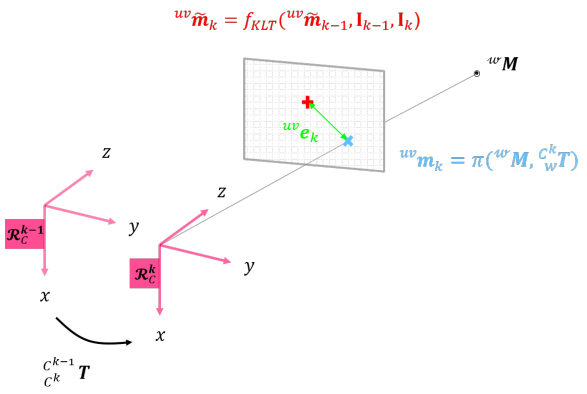


FIGURE 4.11 – Définition de l'erreur

La précision de KLT est évaluée à chaque étape k telle que illustrée sur Fig. 4.11

Format des données Le tracker KLT est appliqué localement au voisinage du point. La fenêtre d'application est couramment choisie de dimension (21×21) [Bra00]. L'objectif est de fournir la précision sur la position du point tracké à partir d'une image $\boldsymbol{I}_k \in \mathbb{R}^{n \times m}$ à l'instant k, avec $(n, m) \in \mathbb{N}^2$, en particuler SYNTHIA génère des images de dimension 640×480. Ainsi pour $N = 1000$ points issus de $\boldsymbol{I}_k$, l'image $\boldsymbol{I}_k$ sera associée à chacun de ces N points indépendamment. En conséquence, elle sera traitée N fois dans l'estimation de la covariance de chaque point induisant un coût de calcul important et proportionnel au nombre de points à évaluer. Une seconde option consiste à regrouper l'estimation de plusieurs points détectés et suivis successivement sur plusieurs images afin de réduire le nombre d'analyse de $\boldsymbol{I}_k$. Ceci soulève un problème d'unicité du format des données d'entrée : Comment garantir un vecteur d'entrée de taille constante ? Outre l'aspect gestion de base de données qui est en effet extrêment coûteux à mettre en place, ce critère de format de données implique de garantir un certain nombre de points suivis sur plusieurs images, sans prendre en compte l'aspect ou la qualité du point. Un point sortant ou apparu après 3 images ne sera pas considéré. Ce critère est très contraignant et filtre de l'information utile. Il s'agit donc de définir une harmonisation du format d'entrée tout en garantissant une indépendance des points d'intérêt observés par image.

La solution retenue est de considérer une fenêtre autour du voisinage du point suivi ${}^{uv}\tilde{\boldsymbol{m}}$ de dimension (21×21). Cette approche a deux avantages : le coût avec l'observation du point dans le voisinage actif de KLT, et l'uniformisation de la nature des données d'entrée. En effet, la fenêtre étant centrée en ${}^{uv}\tilde{\boldsymbol{m}}$, l'information de position du point suivi est inhérente à l'imagette fournie (*i.e.* la fenêtre). Il est ainsi possible de s'affranchir de fournir les coordonnées du point. On suppose alors que la précision du tracker KLT ne dépend que de l'apparence locale de l'image autour du point d'intérêt.

La figure 4.12 illustre le processus de sélection des fenêtres centrées en ${}^{uv}\tilde{\boldsymbol{m}}$. On observe un point détecté par Harris puis tracké sur 191 images (Fig. 4.12(a)). □ indique la position réelle du point ${}^{uv}\boldsymbol{m}$ au cours des 191 images, et × la position estimée du point tracké par KLT ${}^{uv}\tilde{\boldsymbol{m}}$. (Fig. 4.12(c)) montre 10 imagettes associées aux 10 premières positions du points ${}^{uv}\tilde{\boldsymbol{m}}$. Cette représentation des données par des fenêtres (ou imagette) permet de générer des données de même nature et de même dimension.

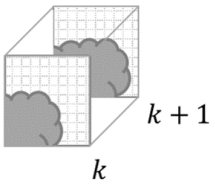
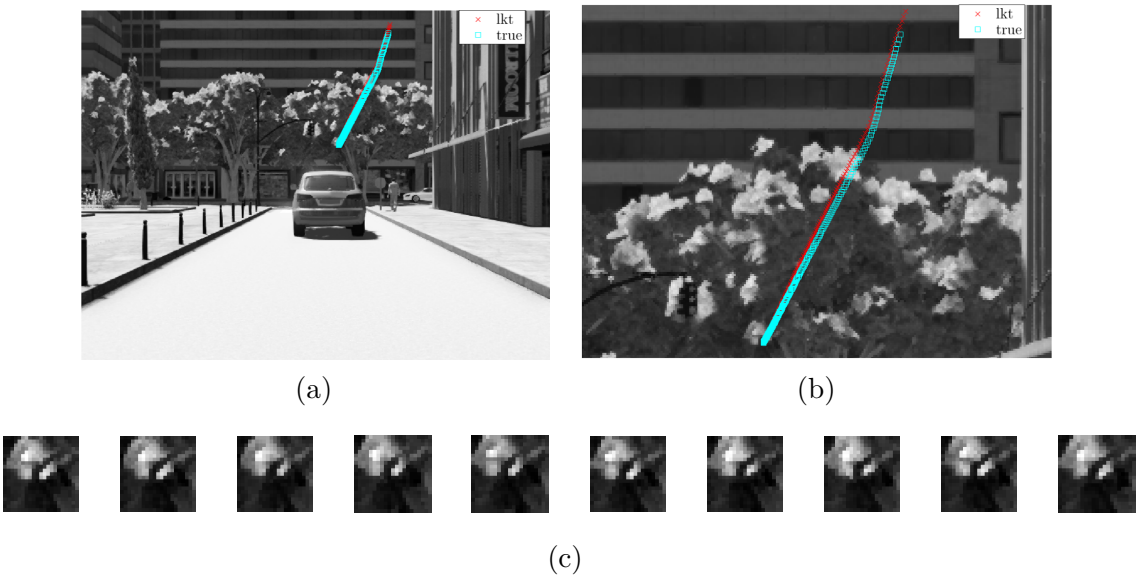


FIGURE 4.13 – Description d'un point suivi en sortie de la base de donnée créé à partir du dataset SYNTHIA

Enfin, le tracking KLT, décrit par $f_{KLT}({}^{uv}\tilde{\boldsymbol{m}}_k, \boldsymbol{I}_k, \boldsymbol{I}_{k+1})$ (eq. 2.24), fait appel à l'image $\boldsymbol{I}_k$ de l'instant courant k et celle $\boldsymbol{I}_{k+1}$ de l'instant suivant $k+1$ pour estimer la position du point ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$. Ainsi pour évaluer la précision de KLT, il est intéressant de mettre en relation les imagettes de l'image courante avec celles de l'image suivante en les concaténant tel que illustré Fig. 4.13.



(a) (b)

(c)

FIGURE 4.12 – Piste de tracking d'un point d'intérêt suivi au cours de X images - (a) Évolution du point dans la scène ; (b) Zoom sur la piste ; (c) Imagettes centrées sur le point suivi ${}^{uv}\tilde{\boldsymbol{m}}$ observées sur les X images

4.3.3 Discussion

L'estimation de la covariance n'est pas observable directement par un capteur. Toutefois, si on ne dispose pas de la vérité terrain, elle peut être approximée par l'erreur de mesure directement, qui elle est observable. À la différence de la position du point dans $\boldsymbol{\mathcal{R}}_w$, la position réelle d'un point dans $\boldsymbol{\mathcal{R}}_{pix}$ n'est pas immédiatemment mesurable. Nous avons proposé une première solution pour la mesure de l'erreur de précision ${}^{uv}\boldsymbol{e}$ en exploitant le dataset SYNTHIA [Ros+16] issu d'un environnement de simulation. En particulier, nous avons vu comment générer une base de données pour évaluer la précision du suivi de points trackés KLT à partir d'image de synthèse dans le but d'estimer la covariance associée à chaque point d'intérêt observé. L'environnement de simulation a plusieurs avantages. Il bénéficie du contrôle des bruits, et des trajectoires appliquées. Il permet également de connaître parfaitement les dimensions de l'environnement, en particulier la carte de profondeur, ainsi que les caractéristiques de la caméra. Par ailleurs, l'utilisation d'un dataset plutôt qu'une expérimentation à l'avantage de permettre la reproduction des résultats.

Pourtant, cette approche n'est pas exploitable sur des données réelles. En effet, elle suppose de connaître l'environnement dans lequel le système évolue en amont, et en particulier de construire ou d'avoir à disposition une carte de profondeur très précise. C'est un processus très couteux à mettre en place dont la précision doit également être estimée. Il contraint la navigation à évoluer dans un environnement connu. L'application n'est donc pas directe et est coûteuse à mettre en place. Outre le coût de la mise en place d'experimentations, les images de synthèse induisent des effets non désirables et non observables sur des données réelles (aliasing, redondance d'objet de synthèse dans l'image .. etc)

Une seconde approche consiste donc à étudier la précision du suivi de points KLT (§ 2.3.2.2)

FIGURE 4.14 – KITTI raw data [Gei+13] - 2011_09_26_*drive*_0002_*sync*

au travers de données réelles issues du dataset KITTI[GLU12].

4.3.4 KITTI Dataset

Présentation du dataset KITTI (Karlsruhe Institute of Technology and Toyota Technological Institute) [GLU12] est un benchmark qui utilise une plateforme automobile équipée de multiples capteurs pour collecter des informations visuelles, entre autres. Ce dataset a pour but de fournir différents données pour les domaines d'applications de l'odométrie visuelle, du flux optique, de la stéréovision ou encore de la détection et du suivi d'objet 3D. Pour chaque cas d'application, une vérité terrain dédiée est fournie.

En particulier, KITTI propose une catégorie de données brutes (*i.e. raw data*) [Gei+13] qui offre des images stéréo synchronisées associées à la position ${}^{w}\boldsymbol{M}$ du système mobile fournies par le GPS couplé avec une IMU [1]. La matrice de calibration ainsi que les matrices d'homogénéisation entre capteurs sont connues. Tous les scénarios sont réalisés dans des zones urbaines ou semi-urbaines de Karlsruhe (Fig. 4.14).

Les scénarios envisagées sont en extérieur et sont sujets au flou optique, à la présence d'objets dynamiques ou encore les déformations optiques. Les conditions d'expérimentation sont optimales et ne proposent pas de scénarios avec des conditions météo perturbées.

Extraction de l'information Précédemment, la mesure de vérité terrain dans $\boldsymbol{\mathcal{R}}_{pix}$ était estimée par la reprojection du point détecté ${}^{uv}\boldsymbol{m}$ au fil des instances k, en considérant comme hypothèse de départ la première détection comme étant la première mesure de vérité terrain. Cette approche implique la connaissance ${}^{w}\boldsymbol{M}$ ou de la profondeur du point ${}^{uv}\boldsymbol{m}$. Une solution pour mesurer le biais induit par le tracking KLT est l'application du tracker sur une série temporelle d'images successives puis de propager le tracker rétro-activement jusqu'à sa position initiale (Fig. 4.15). Idéalement, le point devrait revenir à sa position initiale sans aucun écart de position. Notons ${}^{uv}\boldsymbol{m}_k$ le point détecté à l'instant k par le détecteur de Harris lors de l'initialisation. Il représente la vérité terrain. Notons ${}^{uv}\tilde{\boldsymbol{m}}_k$ le point issu du tracking KLT de k à $k+1$ puis de $k+1$ à k. Alors l'erreur réelle du tracking est définie par la distance entre les deux positions ${}^{uv}\boldsymbol{m}_k - {}^{uv}\tilde{\boldsymbol{m}}_k$.

1. Inertial Measurement Unit ou centrale inertielle

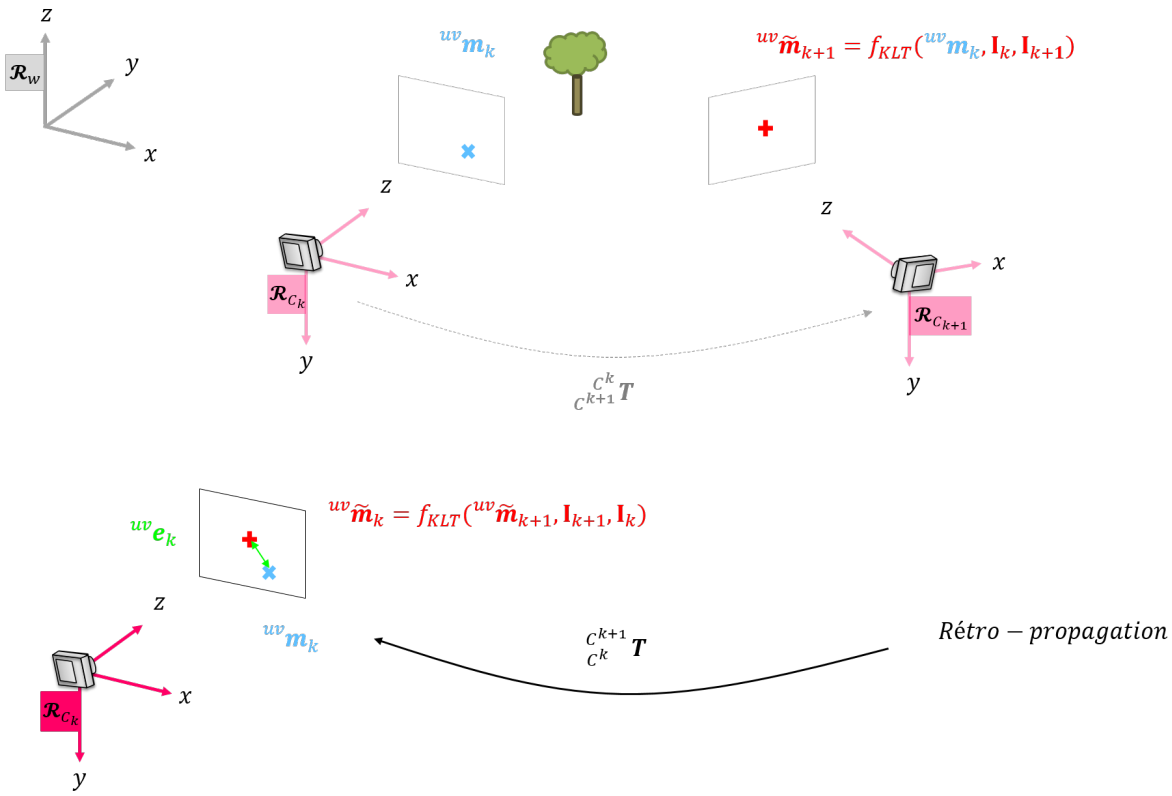


FIGURE 4.15 – Mesure de l'erreur 2D sur des données réelles

Nota Bene Cette erreur représente la précision cumulée d'un aller-retour au cours d'une fenêtre temporelle. Pour obtenir la précision du tracking de l'aller, on divise l'erreur réelle par la taille de la fenêtre temporelle. Dès lors, on suppose une évolution uniforme de l'erreur au cours de la fenêtre temporelle. En conséquence, la taille de la fenêtre temporelle est limitée à un mouvement uniforme et un environnement également uniforme. Un changement important (*i.e.* translation puis rotation brusque) faussera la mesure de l'incertitude.

Dans notre cas (Fig. 4.15), nous appliquons une fenêtre temporelle de 2 images.

4.4 Conclusion

Dans ce chapitre, nous avons proposé d'estimer une incertitude adaptée pour chacun des points d'intérêt. En effet, une incertitude fixe aura pour conséquence le filtrage d'observations. Une incertitude adaptée des points d'intérêt permet d'exploiter le maximum d'informations utiles même dans le cas d'une image bruitée ou partiellement bruitée, améliorant ainsi la performance de l'application envisagée.

L'erreur de mesure d'un point d'intérêt est observable. Elle définit l'écart entre la vérité terrain et l'estimation du point. Par l'application d'une loi normale multivariée, l'estimation de la

matrice de covariance est possible.

Nous avons proposé de générer une base de données pour évaluer la précision du suivi de points trackés KLT à partir d'image de synthèse dans le but d'estimer la covariance associée à chaque point d'intérêt observé. Cette approche comporte de nombreux avantages, notamment la disponibilité des mesures de profondeur et le contrôle de l'environnement de simulation. Toutefois, cette approche n'est pas envisageable en temps réel. De fait, elle suppose de connaître l'environnement dans lequel le système évolue. Par ailleurs, les images de synthèse induisent des effets non désirables et non observables sur des données réelles (aliasing, redondance d'objet de synthèse dans l'image .. etc)

Nous avons envisagé une seconde méthode qui consiste à étudier la précision des points d'intérêt au travers de données réelles issues du dataset KITTI[GLU12]. La mesure de l'erreur est réalisée par l'application du tracker au cours d'une fenêtre temporelle puis faire une rétropropagation du point jusqu'à sa position initiale. Cette méthode est peu coûteuse et facile à mettre en œuvre et convient à tout type de données. Elle donne une mesure précise de l'erreur cumulée au cours d'un aller retour. Néanmoins, elle suppose une évolution uniforme de l'erreur pendant la fenêtre temporelle, limitant la taille de la fenêtre temporelle.

Chapitre 5

Modélisation du réseau de neurones et résultats

L'observation recueille les faits ; la réflexion les combine ; l'expérience vérifie le résultat de la combinaison.

Denis Diderot

Nous avons mis en évidence le lien entre l'estimation de la covariance et l'observation de l'erreur locale d'un point d'intérêt (chapitre 4), et en particulier, comment mesurer cette erreur et définir la représentation de la vérité terrain. Dans ce chapitre, nous abordons les principes de base des réseaux de neurones (RdN). Nous montrons comment construire un modèle fournissant une estimation de matrice de covariance pour chacun des points d'intérêt suivis. Le système proposé est entraîné et évalué sur les données générées précédemment à partir des dataset SYNTHIA[Ros+16] et KITTI[GLU12] mettant en évidence de bons résultats par rapport à l'état de l'art. Une application d'odométrie visuelle est présentée illustrant les bénéfices d'une incertitude adaptée au suivi des points d'intérêt.

5.1 Réseaux de neurones profonds : les outils mathématiques

Dans la suite des travaux, nous abordons des notions de réseaux de neurones (RdN), en particulier le *Deep Leaning* (DL [1]), de l'anglais pour décrire *l'apprentissage profond*. Il convient de présenter les concepts et les outils nécessaires à la compréhension du manuscrit. Pour davantage de détails, le lecteur peut se référer à [GBC16].

5.1.1 Réseau de neurones classique "dense"

Soit $\boldsymbol{x} \in \mathbb{R}^n$ une entrée de dimension n et $\boldsymbol{y} \in \mathbb{R}^m$ la sortie associée, de dimension m, où $(n, m) \in \mathbb{R}^2$. Le fonctionnement de base d'un réseau de neurones (RdN) est modélisé par (4.2)

$$f(\boldsymbol{x}) = \sigma(\boldsymbol{x} \cdot \boldsymbol{\mathcal{W}} + \boldsymbol{b})$$

où $\sigma(.)$ définit une fonction non linéaire. Cette modélisation (4.2) de base décrit une unité ou un étage de réseau de neurones. La composition de plusieurs étages, aussi appelés *couches*, assemblés en cascade créent un modèle de RdN plus complexe.

Soit un ensemble de N entrées et sorties telles que $\forall i \in |[1; N]|$, $\boldsymbol{x}_i \in \mathbb{R}^n$ et $\boldsymbol{y}_i \in \mathbb{R}^m$, $(n, m) \in \mathbb{N}^2$ l'équation (5.1) soit satisfaite.

1. *Deep Learning*

$$\boldsymbol{x}_{i+1} = f_i(\boldsymbol{x}_i) = \sigma(\boldsymbol{x}_i \cdot \boldsymbol{\mathcal{W}}_i + \boldsymbol{b}_i) \tag{5.1}$$

L'équation 5.1 définit la connexion entre deux couches de RdN. Un réseau de neurones est la composition de toutes ces fonctions telle que

$$RdN(\boldsymbol{x}; \boldsymbol{\mathcal{W}}; \boldsymbol{b}) = f_C \circ f_{C-1} \circ \cdots \circ f_1(\boldsymbol{x}) \tag{5.2}$$

$\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ sont les paramètres de chaque couche $C \in \mathbb{N}$ plus communément dénommés *Weighting matrix*, matrice de poids et *bias*, biais. Plus le nombre de couches est important, plus la complexité du modèle est grande ; on parle alors de réseaux de neurones *profonds* ou *Deep Neural Network*.

Nota Bene $\sigma(.)$ décrit la fonction d'activation qui permet le choix de la non-linéarité du modèle. Pour ne citer que quelques exemples, Rectified Linear Unit (5.3), et sa version paramétrable (5.4), ou encore Sigmoïd (5.5).

$$ReLU(\boldsymbol{x}) = max(x, 0) \tag{5.3}$$

$$LeakyReLU(\boldsymbol{x}) = \begin{cases} \alpha \cdot \boldsymbol{x} & if\ \boldsymbol{x} < 0 \\ \boldsymbol{x} & if\ \boldsymbol{x} >= 0 \end{cases} \tag{5.4}$$

$$Sigmoïd(\boldsymbol{x}) = \frac{1}{1 + e^{-\boldsymbol{x}}} \tag{5.5}$$

Une couche telle que définie par (5.1) a la particularité de connecter toutes les entrées d'une couche avec les sorties de la suivante. On parle alors de couche *dense*.

5.1.2 RdN convolutif (CNN)

Un réseau de neurones convolutif (CNN)[RHW85][LeC+89] est un outil très populaire pour le traitement d'image. Il permet, en particulier, d'extraire des caractéristiques visuelles (bruits, texture, etc ..) ce qui est particulièrement intéressant dans le contexte de l'odométrie visuelle. Son fonctionnement repose sur une convolution qui tranforme un signal $x(t)$ en $y(t)$ comme décrit eq. 5.6.

$$y(t) = x(t) * kr(t) = \int_{-\infty}^{+\infty} x(x) kr(t - x)\, \mathrm{d}t \tag{5.6}$$

$kr(t)$ joue ici le rôle d'un filtre, également désigné le kernel ou le noyau du signal. Appliqué à une image, $\boldsymbol{Kr}$ est une matrice de convolution (5.7), communément appelé un *Kernel*.

$$\boldsymbol{Y}(n, m) = \sum_p \sum_q \boldsymbol{I}(n - p, m - q) \boldsymbol{Kr}(p, q) \tag{5.7}$$

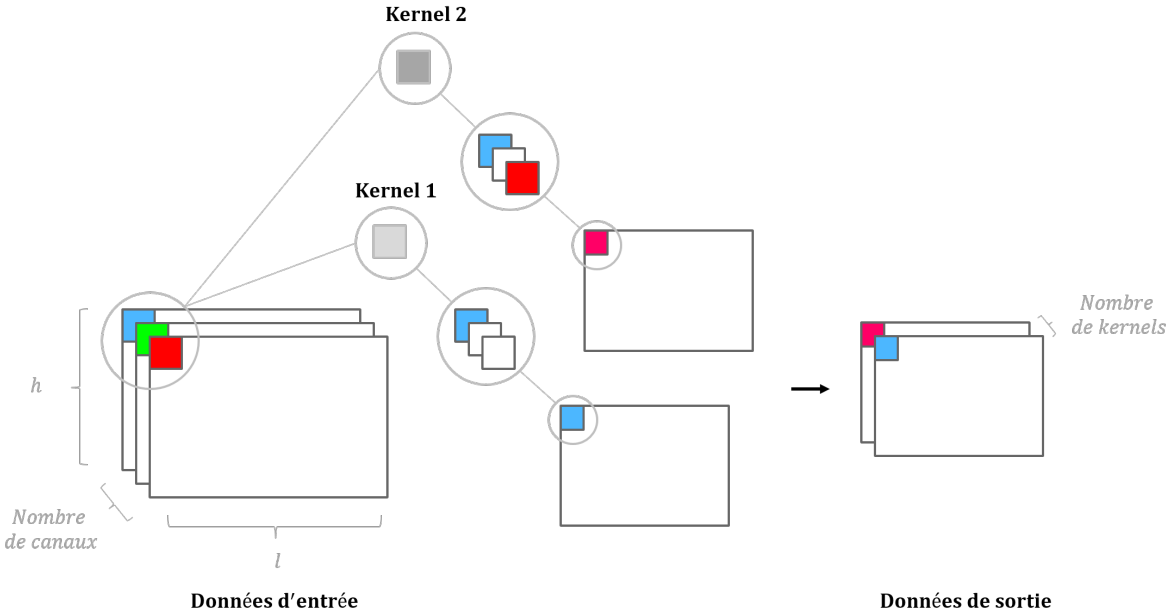


FIGURE 5.1 – Illustration du principe de fonctionnement d'un CNN, avec en entrée une image RVB convolutionnée avec deux filtres de Kernel de dimension 1 pixel. Le premier kernel filtre le rouge et le vert produisant un pixel bleu, le second kernel filtre uniquement le vert et aboutit en un pixel rose. L'image résultante est le résultat de chaque convolution concaténée.

où $\boldsymbol{I} \in \mathbb{R}^{n \times m}$ l'image d'entrée et $\boldsymbol{Y} \in \mathbb{R}^{n \times m}$ le résultat de la convolution. $\boldsymbol{Kr}$ peut être de taille variable (*e.g.* $\boldsymbol{Kr} \in \mathbb{R}^{3 \times 3}$). $\boldsymbol{Kr}$ peut être appliqué à chaque pixel de l'image $\boldsymbol{I}$ ou avec un intervalle spatial de plusieurs pixels, appelé *strides*[GBC16]. Dans le but d'uniformiser la taille du signal de sortie, des techniques de *padding*, de l'anglais *remplissage par* 0, peuvent être mises en place.

L'équation (5.7) décrit le fonctionnement d'une couche CNN. En contraignant la dimension de $\boldsymbol{Kr}$, un CNN peut traiter une image de haute résolution de façon efficace. Par contraste avec une couche *dense* (section 5.1.1) où toutes les composantes d'un signal d'entrée sont interconnectées, le CNN propose une solution moins complexe, donc moins couteuse, où seuls les paramètres composant $\boldsymbol{Kr}$ sont à ajuster. Néanmoins, il réduit les corrélations spatiales possibles au voisinage proche. Dès lors, plusieurs filtres sont appliqués en parallèle de sorte à augmenter le nombre de corrélations spatiales (Fig. 5.1).

5.1.3 Paramétrisation

Le choix des hyperparamètres est essentiel. En effet, il permet au cours de l'entrainement de faire converger la fonction de minimisation.

Nota Bene Les hyperparamètres définissent les paramètres liés à structure du modèle (nombre de couches, taille des noyaux de convolution, etc..). Ils sont à distinguer des paramètres $\boldsymbol{\mathcal{W}}$, $\boldsymbol{Kr}$ et $\boldsymbol{b}$ qui définissent les valeurs internes de chaques couches. L'optimisation permet d'estimer les paramètres (et non les hyperparamètres) qui minimisent la fonction de coût.

Optimisation La descente de gradient est une méthode d'optimisation [LBH15] appliquée à la fonction de minimisation lors de la phase d'apprentissage au cours de la rétro-propagation, en anglais *back-propagation*. Elle détermine le maximum de la fonction de minimisation. À chaque itération, la descente du gradient détermine une direction à suivre dans l'espace des paramètres pour dininuer la fonction à minimiser. En pratique, cette méthode d'optimisation est appliquée à un lot de données sélectionné aléatoirement, un *batch*, sur les données d'entrainement. Cette approximation permet de stabiliser la convergence en moyennant le gradient sur plusieurs échantillons et de réduire le coût de calcul du gradient.

Une limitation de la descente de gradient est la progression de la recherche qui peut ralentir en fonction du gradient de la courbe. Un moment peut être ajouté à la descente de gradient permettant d'adapter le pas d'apprentissage. C'est le principe de fonctionnement de la méthode *Adam* (Adaptive Movement Estimation) [KB14] qui utilise les estimations des premier et deuxième moments du gradient pour adapter le pas d'apprentissage de chaque poids du RdN. *NAdam* (Nesterov-accelerated Adaptive Moment Estimation) [Doz16] est une extension de l'algorithme Adam. Il incorpore le gradient de la nouvelle position estimée à la place de la position actuelle, appelé le moment Nesterov.

Le pas d'apprentissage, ou *learning rate*(lr), peut également être adapté en fonction de l'évolution de la fonction de minimisation. Il permet d'effectuer des grands pas au début pour converger grossièrement, et de faire des pas plus petits vers la fin pour affiner les résultats.

Overfitting L'*overfitting*, ou surajustement des données, définit les paramètres issus de l'apprentissage pour lesquels le dataset d'apprentissage est le seul à fournir de bonnes estimations. Dans ce cas, le modèle est surajusté aux données d'apprentissage et non adapté pour un mode opératoire fonctionnel avec des données nouvelles. Afin de généraliser le modèle à de nouvelles entrées et ainsi éviter l'*overfitting* à un ensemble de données, un *dropout*[Sri+14] est intégré dans le modèle du RdN. Le dropout est une des solutions pour limiter l'overfitting. Il s'agit d'une méthode de régularisation dont le mode de fonctionnement repose sur la mise à zéro aléatoire de certaines entrées de couches, tel que décrit Fig. 5.2. La méthode permet de simplifier la complexité de modèle du RdN envisagé.

Scaling et pooling Lorsque l'on travaille avec un grand nombres d'entrées, il est souvent intéressant de sous-échantillonner la résolution du signal d'entrée pour réduire la complexité du système. Parmis les méthodes de sous-échantillonnage, le *pooling* est couramment appliqué. Il s'agit de moyenner, maximiser ou minimiser les entrées par bloc. Toutefois, les opérations de pooling dégradent l'information spatiale contenue dans des images par exemple. En réponse à ce phénomène, [Han+16][GBC16] montre qu'il est plus intéressant d'utiliser des *strides*,

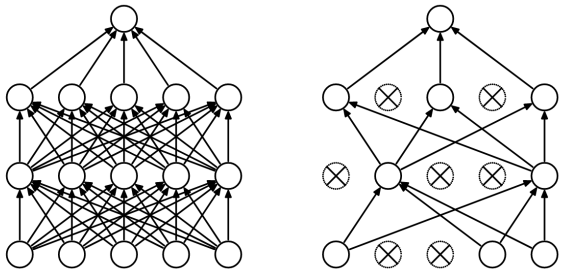


FIGURE 5.2 – *Dropout* aléatoire appliqué aux neurones d'entrée de chaque couche. Figure issue de [Sri+14]

autrement dit des sauts lors du calcul des convolutions, pour sous-échantillonner les données tout en préservant l'information spatiale.

5.2 DUNE

L'estimation de l'incertitude des points d'intérêt, telle que définit dans le chapitre 2.4, permet de qualifier la précision d'une observation. Elle indique la confiance d'une observation et permet la pondération de celle-ci lors de son intégration dans un système de navigation. En particulier, dans le contexte d'une hybridation serrée (§ 6.1), une estimation précise de l'incertitude permet la réduction de la dérive temporelle des estimations de position du système mobile.

Toutefois, l'incertitude locale des points d'intérêt n'est pas observable directement par un capteur. Nous avons montré qu'elle peut être estimée à travers la mesure de l'erreur de position d'un point d'intérêt (chapitre 4).

Par choix, l'extraction de l'information est opérée par une méthode de suivi basée sur l'étude locale du gradient (§ 4.3.2), le tracker de Kanade Lucas Tomasi (KLT) [BK81], qui montre de meilleurs résultats que certains autres descripteurs [Per+15]. Ainsi, l'estimation de l'incertitude d'un point suivi par KLT permet de caractériser les performances du tracker. Notre objectif est donc d'estimer l'incertitude de chacun des points suivis, comme illustré figure 4.1, pour caractériser la précision d'un point localement.

Seuls quelques travaux ont été consacrés à l'estimation locale de l'incertitude des points d'intérêt. En particulier, l'impact de l'estimation d'une covariance adaptée pour estimer avec précision la localisation du système mobile dans $\boldsymbol{\mathcal{R}}_w$ [KK01][WM17]. Des applications de l'odométrie visuelle (VO) basées sur des méthodes d'extraction de l'information indirectes font généralement l'approximation d'une incertitude moyenne fixe et comune à tous les points d'intérêt [MAMT15].

Dans le domaine de l'apprentissage profond, la plupart des efforts des approches modernes sont consacrés à la résolution du problème du suivi au niveau global [Wan+18][Cui+22][Yan+22], autrement dit à l'estimation de l'incertitude résultante de la position du système mobile dans $\boldsymbol{\mathcal{R}}_w$. Certaines approches lient le suivi de points et la correspondance à travers des

patchs[2] [Han+15][PS21] mais sont rarement associées à des modèles d'incertitude de leurs prédictions. L'avantage d'utiliser des patchs d'images est double : il permet d'obtenir un réseau peu profond avec un faible temps d'inférence, et de réduire la quantité de données à traiter, en faisant correspondre les données utilisées par le tracker lui-même.

Nous avons mis au point DUNE (Deep UNcertainty Estimation) (Fig. 5.3), un estimateur de matrice de covariance locale, qui prend en entrée une paire de *patchs* d'images et retourne des matrices de covariances. Considérons deux images successives à k et $k+1$ soumises à une extraction d'information (§ 4.3.2), le système fournit une covariance locale dans $\mathbb{R}^{2\times 2}$ (4.2) pour chaque point d'intérêt suivi entre k et $k+1$. Les paramètres du système minimisent la fonction de coût (§ 4.2) définie équation (4.19).

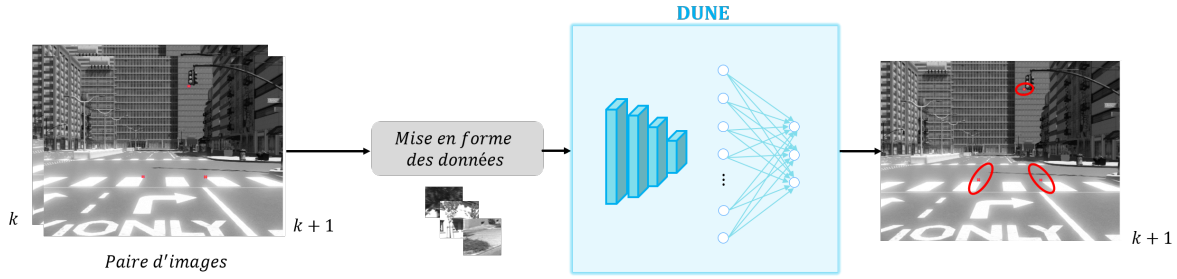


FIGURE 5.3 – Deep UNcertainty Estimation (DUNE) pipeline. Le système prend en entrée une paire d'images, sur laquelle des points d'intérêt sont suivis. L'information est par la suite traitée (§ 4.3) afin d'extraire des imagettes centrées sur la position des points estimée par KLT. Ces imagettes sont fournies au RdN DUNE qui estime les matrices de covariance de chacun des points d'intérêt suivis.

Notre approche est composée de deux étapes : une première met en forme les données (§ 5.2.1) et définit la forme prise par le tenseur en entrée du RdN ; la seconde prédit les matrices de covariances associées à la position des points d'intérêt. DUNE est entraînée et évaluée sur les jeux de données SYNTHIA et KITTI [Ros+16] ; [Gei+13]. En fournissant les incertitudes des points suivis sous la forme d'une matrice de covariance 2D, notre objectif à plus long terme est de permettre une estimation fine des erreurs d'un estimateur visuel de mouvement, couplé ou non à des données inertielles.

5.2.1 Mise en forme des données

Soit $\boldsymbol{W}_k$ la représentation de la fenetre de dimension (21×21) de la position du point suivi ${}^{uv}\tilde{\boldsymbol{m}}_k$ à partir d'une image $\boldsymbol{I}_k \in \mathbb{R}^{n\times m}$ à l'instant k, avec $(n, m) \in \mathbb{N}^2$. Pour rappel, la dimension de la fenêtre $\boldsymbol{W}_k$ est choisie pour coïncider à la dimension de la fenêtre d'étude du tracker KLT (§ 4.3.2) .

Le tracking KLT, décrit par $f_{KLT}({}^{uv}\tilde{\boldsymbol{m}}_k, \boldsymbol{I}_k, \boldsymbol{I}_{k+1})$ (eq. 2.24), fait appel à l'image $\boldsymbol{I}_k$ de l'instant courant k et celle $\boldsymbol{I}_{k+1}$ de l'instant suivant $k+1$ pour estimer la position du point ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$. Ainsi pour évaluer la précision de KLT, il est intéressant de mettre en relation les

2. fenêtres

imagettes de l'image courante avec celles de l'image suivante en les concaténant telle qu'illustré Fig. 4.13

Les données d'entrée sont représentées par la concaténation de deux fenêtres centrées en ${}^{uv}\tilde{\boldsymbol{m}}_k$ le point tracké à k et ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$ le point tracké à $k+1$. Soit $\boldsymbol{W}_k$ et $\boldsymbol{W}_{k+1}$ ces fenêtres composant un échantillon, tel que décrit Fig. 5.4.

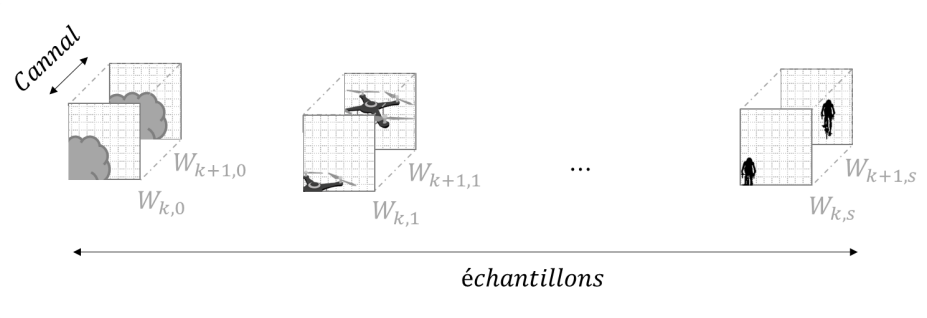


FIGURE 5.4 – Format des données - concatenation des fenêtres centrées en ${}^{uv}\tilde{\boldsymbol{m}}_k$ et ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$. L'ensemble des échantillons s constituent le jeu de données $\mathcal{D}$.

Considérons le jeu de données $\mathcal{D} = \{\ [{}^{uv}\boldsymbol{W}_k;{}^{uv}\boldsymbol{W}_{k+1}]_i\ ,\ {}^{uv}\boldsymbol{e}_{i,k+1}\ \mid\ \forall i \in |[1;s]|\ \mid\ \forall k \in \mathbb{N}\}$ où s définit le nombre d'échantillons dans le jeu de données. On note $\boldsymbol{\Sigma}_i$ l'estimation de la covariance locale résultante du système DUNE associée à l'imagette $\boldsymbol{W}_{k+1}$ de l'échantillon i noté $\boldsymbol{W}_{k+1,i}$. L'erreur associée ${}^{uv}\boldsymbol{e}_{i,k+1}$ pour tout $i \in |[1;s]|$ est évaluée à l'instant $k+1$.

Le problème de minimisation définit § 4.2, se résume à comparer l'erreur d'un échantillon i estimé à l'instant $k+1$ avec la covariance prédite par le RdN $\tilde{\boldsymbol{\Sigma}}_i$ tel que décrit (5.8).

$$\forall i\ ,\ \underset{\boldsymbol{\Sigma}_i;\boldsymbol{\mathcal{W}}_i;\boldsymbol{b}_i}{\operatorname{argmin}}\ \mathcal{L}_{\boldsymbol{\Sigma}_i}(\{RdN([{}^{uv}\boldsymbol{W}_k;{}^{uv}\boldsymbol{W}_{k+1}]_i;\boldsymbol{\mathcal{W}}_i;\boldsymbol{b}_i),{}^{uv}\boldsymbol{e}_{i,k+1}\}) \tag{5.8}$$

Le processus d'entrainement de l'estimateur d'incertitude de la position des points d'intérêt DUNE, représenté Fig. 5.5, est réalisé par la fonction de coût (5.8) minimisant les poids au sein du modèle de RdN afin de satisfaire (4.19). Dans ce cas, la matrice de covariance est apprise par rapport à l'entrée, capturant ainsi l'incertitude hétéroscédastique [3]

5.2.2 Modélisation de DUNE

5.2.2.1 Structure du réseau de neurones

Nous construisons DUNE (Deep UNcertainty Estimation), une méthode d'estimation de l'incertitude basée sur des réseaux de neurones (RdN) profonds (chapitre 5.1) qui vient minimiser la fonction de coût (§ 4.2) définie équation (4.19). DUNE modèlise un estimateur d'incertitude de la position des points d'intérêt observés par une méthode de tracking KLT. Nous supposons que les erreurs induites par le tracking des points d'intérêt dépendent du voisinage des points, en particulier des caractéristiques de l'image (texture, flou, bruit, luminosité, etc...). C'est pourquoi les fenêtres du voisinage des points courant et suivant composent le tenseur

3. décrit une statistique composée de variances hétérogènes.

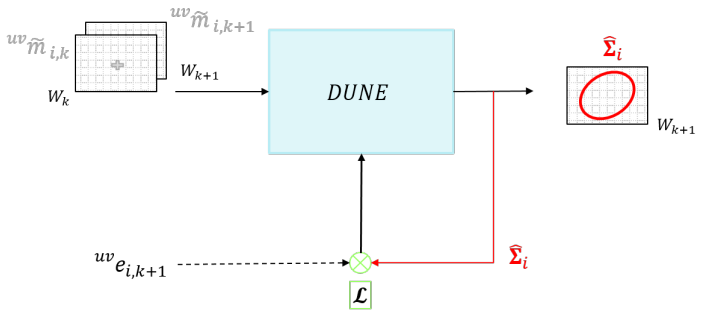


FIGURE 5.5 – Ajustement du réseau de neurones DUNE pendant la phase d'apprentissage - DUNE prend en entrée un échantillon i de deux imagettes concaténées $[^{uv}\boldsymbol{W}_k;^{uv}\boldsymbol{W}_{k+1}]_i$ centrées en $^{uv}\tilde{\boldsymbol{m}}_k$ et $^{uv}\tilde{\boldsymbol{m}}_{k+1}$, et prédit la matrice de covariance $\tilde{\boldsymbol{\Sigma}}_i$ associée à $^{uv}\boldsymbol{W}_{k+1,i}$. La fonction de coût appliquée $\mathcal{L}$ compare l'erreur mesurée à $k+1$ $^{uv}\boldsymbol{e}_{i,k+1}$ avec la matrice de covariance prédite $\tilde{\boldsymbol{\Sigma}}_i$

d'entrée de l'estimateur, considérant ainsi l'environnement du point suivi et l'information sur la position du point.

Inspiré par des méthodes d'estimation de l'incertitude sur la position globale du système mobile utilisant des RdN [Liu+18] [DML20], DUNE a la particularité de considérer uniquement des fenêtres du voisinage des points d'intérêt, aussi appelé *imagettes*, dont les dimensions coïncident avec celles du tracker KLT appliqué. L'avantage d'utiliser des patchs d'images est double : il permet d'obtenir un réseau moins profond, et donc moins complexe, et ainsi limiter le temps d'inférence. Par ailleurs, la quantité de données à traiter est moindre et adaptée à analyser et caractériser le fonctionnement du tracker.

Au vu des résultats récents de [Liu+18], nous basons notre structure DUNE sur un réseau neuronal convolutif (CNN). Un premier essai consistait à adapter le RdN DICE [Liu+18] à l'estimation locale d'incertitude sur la position de chaque point d'intérêt.

Par choix, nous n'utilisons pas la fonction max-pooling afin de préserver l'information spatiale [Han+16] et optons pour les strides, ou sauts (§ 5.1.3). Par ailleurs, le format des données d'entrée, en particulier l'utilisation de fenêtres permet de maximiser l'information au voisinage des points d'intérêt. Tout comme [DML20], nous avons fait le choix d'augmenter le nombre de filtres convolutifs pour l'estimation du modèle d'incertitude.

DUNE est un CNN[4] profond composé de deux blocs principaux (Fig. 5.6) : la structure convolutive profonde, détaillée dans le tableau 5.1, et les couches denses. La structure convolutive est une succession de 4 couches convolutives avec une taille de Kernel $\boldsymbol{Kr}$ (définit § 5.1) fixe de 3×3. Une couche de normalisation et un dropout fixé à 20% sont ajoutés après chaque couche convolutive. La couche de normalisation est utilisée pour assurer la stabilité de l'apprentissage. Un dropout (définit § 5.1.3) est intégré après chaque couche pour éviter le surajustement.

4. *Convolutional Neural Network*

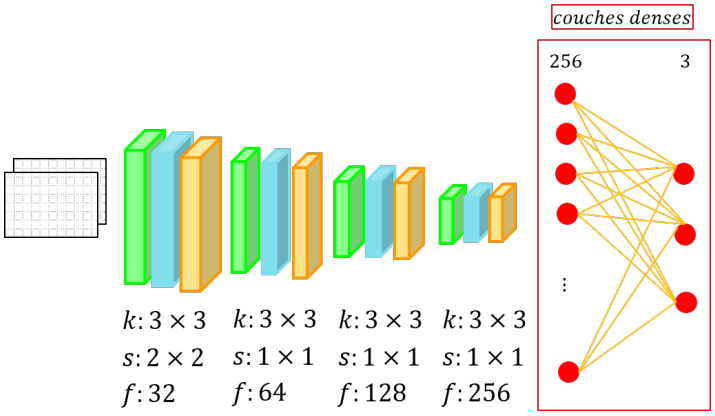


FIGURE 5.6 – Structure de l'estimateur d'incertitude sur la position des points d'intérêt DUNE - k indique la dimension des Kernels ; s la dimension des strides ; et f le nombre de filtres. Les couches vertes représentent les convolutions, les bleues les couches de normalisation et les jaunes les dropouts.

Nota Bene Le dropout intervient uniquement au cours de la phase d'apprentissage pour éviter le surajustement (§ 5.1).

Layer	Kernel	Stride	Padding	Filters
Conv_1	3×3	2×2	None	32
Conv_2	3×3	1×1	None	64
Conv_3	3×3	1×1	None	128
Conv_4	3×3	1×1	None	256

TABLE 5.1 – Structure des couches couvolutives de DUNE

Les couches convolutives sont suivies de deux couches denses (§ 5.1.1), où les entrées et sorties sont interconnectées. Elles sont composées respectivement de 256 et 3 sorties. Chaque couche du RdN DUNE, à l'exception de la dernière, est suivie d'une fonction d'activation *leakyrelu* (§ 5.1.1).

La Fig. 5.6 montre le fonctionnement et la structure de DUNE. DUNE prend en entrée deux fenêtres d'image concaténées centrées sur ${}^{uv}\tilde{\boldsymbol{m}}_k$ et ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$, et renvoie l'incertitude $\boldsymbol{\Sigma} \in \mathbb{R}^{2\times2}$, composée de trois paramètres l_1, d_1 et d_2 (§ 4.2.2), sur les coordonnées du point ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$.

Nota Bene Pour remonter à la matrice de covariance $\boldsymbol{\Sigma}$ à partir des trois paramètres l_1, d_1 et d_2, il suffit d'appliquer (4.10) :

$$\boldsymbol{\Sigma} = \begin{pmatrix} 1 & 0 \\ l_1 & 1 \end{pmatrix} \begin{pmatrix} e^{d_1} & 0 \\ 0 & e^{d_2} \end{pmatrix} \begin{pmatrix} 1 & 0 \\ l_1 & 1 \end{pmatrix}^T$$

5.2.2.2 Corrélation des paramètres de Harris avec la durée du suivi d'un point d'intérêt

Nous avons observé une corrélation entre la valeur des paramètres de Harris (λ_1, λ_2) (§ 2.3.2.2) et la durée de vie du suivi de point dans le temps. Pour rappel, (λ_1, λ_2) définissent les valeurs propres de la matrice hessienne $\boldsymbol{H}$ calculée par le détecteur de Harris [HS88] pendant la phase de sélection des points d'intérêt.

Pour évaluer ce postulat, la distribution des paramètres de Harris (Fig. 5.7) a été représentée en fonction de la durée de vie du tracking d'un point, en d'autres termes, le nombre de *frames* (*i.e.* images) pour lesquelles le point est suivi par le tracker KLT. Au cours d'une séquence vidéo contenant 150 images successives, le suivi de points est divisé en 5 catégories :

- ⋆ Les points suivis par le tracker KLT entre 0 et 5 images consécutives au maximum (Fig. 5.7a)
- ⋆ Les points suivis par le tracker KLT entre 5 et 10 images consécutives (Fig. 5.7b)
- ⋆ Les points suivis par le tracker KLT entre 10 et 20 images consécutives (Fig. 5.7c)
- ⋆ Les points suivis par le tracker KLT sur plus de 20 images consécutives (Fig. 5.7d)
- ⋆ Les points non suivis par le tracker KLT (Fig. 5.8)

Un point ne peut apparaître dans plusieurs catégories. Ainsi, les points ayant une durée de vie sur le suivi de plus de 20 frames ne sont pas inclus dans la première catégorie des points suivis entre 0 et 5 frames. D'après la Fig. 5.7, on observe que la première catégorie est celle qui représente la plus grande proportion en terme de durée de vie d'un point par le tracker KLT. En effet, 50% points détectés sont suivis sur moins de 5 images consécutives. Alors que 28% sont trackés sur plus de 20 frames.

Pour chaque catégorie, on observe également une corrélation entre la valeur des paramètres de Harris et la durée d'observation d'un point au cours du temps. En effet, Fig. 5.7(a) met en évidence que plus de 95% de points suivis entre 0 et 5 images consécutives ont $\lambda_1 \sim 0$ et $\lambda_2 \sim 0$. À partir de Fig. 5.7(c), un shift commence à s'opérer et la distribution atteint son maximum pour λ_1 et λ_2 grands. En particulier, Fig. 5.7(d) les paramètres de Harris associés aux points suivis sur plus de 20 images consécutives, donc ayant une grande duréee de vie en terme de tracking, montre une distribution plus étalée avec un maximum en $\lambda_1 = 0.5$ et $\lambda_2 = 0.2$.

Cette observation est confirmée par la valeur des paramètres de Harris pour des points détectés par le détecteur de Harris mais non suivis par le tracker KLT. En effet, le maximum de cette distribution est observée pour $\lambda_1 \neq 0$ et $\lambda_2 = 0$. Néanmoins, la distribution reste fortement concentrée à $\lambda_1 = 0$ et $\lambda_2 = 0$ qui représente le pic secondaire.

En conclusion, cette étude des paramètres de Harris en fonction de la durée de vie d'un point permet de mettre en avant la corrélation apparente entre la qualité des points et la valeur de λ_1 et λ_2. Ainsi, ils sont des indicateurs de la qualité du tracking et ajoutent de l'information utile quant à la précision sur la position d'un point. L'intégration des paramètres de Harris est donc un levier d'amélioration potentiel de notre méthode d'estimation d'incertitude sur la position des points d'intérêt DUNE.

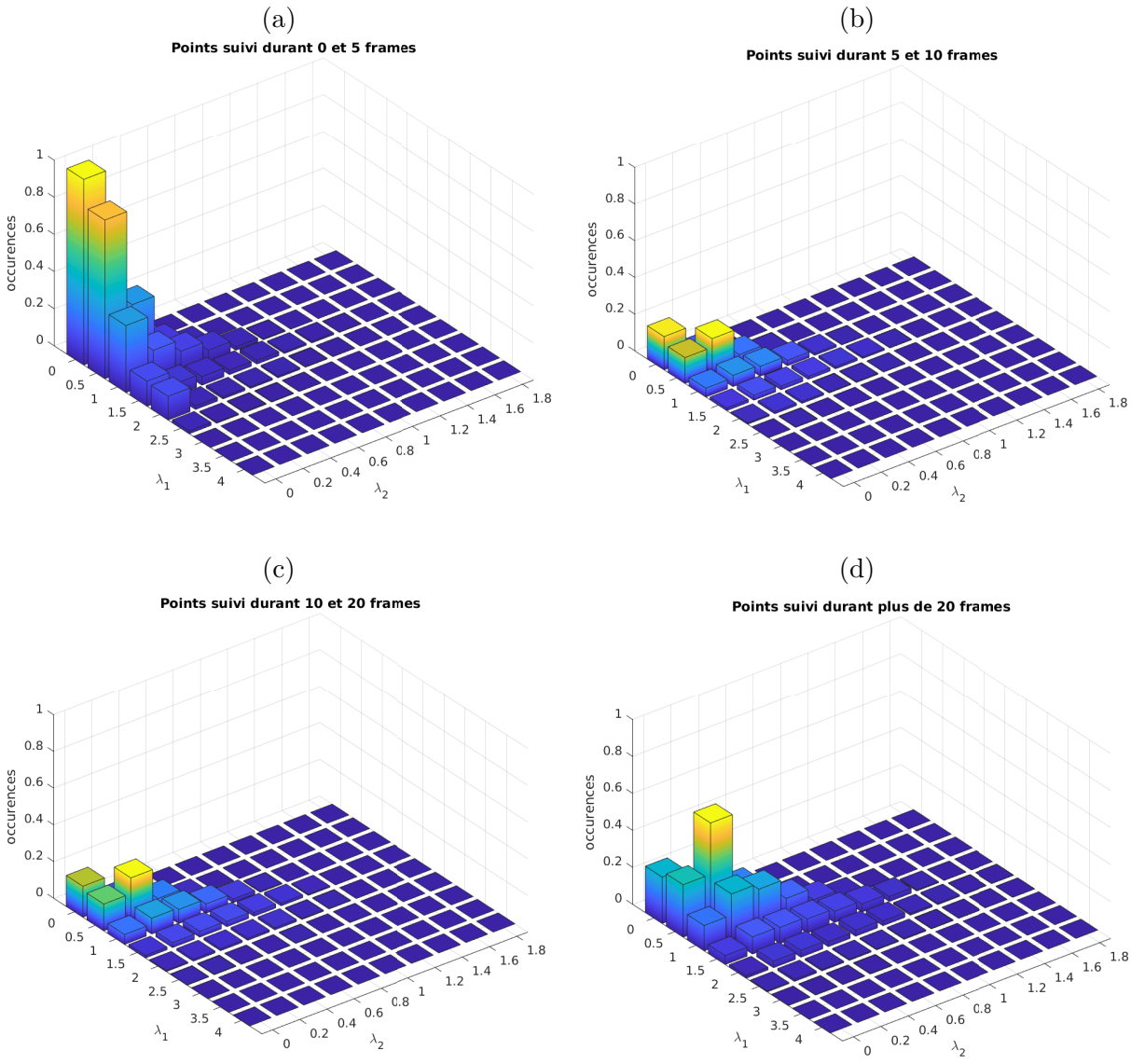
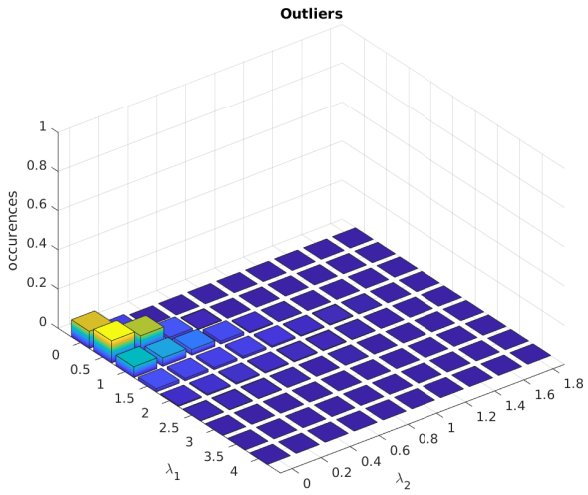


FIGURE 5.7 – Corrélation de la durée de vie du suivi d'un point d'intérêt avec les paramètres de Harris λ_i - (a) points suivis entre 0 et 5 images consécutives ; (b) points suivis entre 5 et 10 images consécutives ; (c) points suivis entre 10 et 20 images consécutives ; (d) points suivis sur plus de 20 images consécutives.

5.2.2.3 Intégration des paramètres de Harris

Comme nous l'avons vu dans le paragraphe précédant, nous avons observé une corrélation entre les paramètres de Harris (λ_1, λ_2), définis par les valeurs propres de la matrice hessienne selon [HS88], et la durée de vie du tracking d'un point, autrement dit le nombre d'images consécutives pendant lesquelles un point est suivi avec succès dans le temps. Nous supposons donc que les paramètres (λ_1, λ_2) sont représentatifs de la précision du suivi de points d'intérêt et nous avons donc choisi d'incorporer cette information directement dans l'arhitecture DUNE.



(e)

FIGURE 5.8 – Corrélation de la durée de vie du suivi d'un point d'intérêt avec les paramètres de Harris λ_i pour les points non suivis ou *outliers*

Nous définissons une variante à la structure de RdN DUNE qui incorpore les paramètres de Harris (λ_1, λ_2). Les paramètres sont concaténés au dernier étage du réseau de neurones. En conséquence, une nouvelle couche dense de 3 sorties est ajoutée, comme le montre la Fig. 5.9. Le choix d'intégrer les paramètres de Harris en bout de chaine permet d'équilibrer l'impact des paramètres de Harris avec toutes les autres entrées. En effet, la concaténation à la couche dense antérieure noie l'information ajoutée avec les *features* en sortie du CNN. L'ajout d'une couche dense supplémentaire permet de maintenir trois informations de sortie correspondant aux paramètres l_1, d_1 et d_2 (§ 4.2.2) constituant l'incertitude $\mathbf{\Sigma} \in \mathbb{R}^{2\times 2}$ sur les coordonnées du point ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$.

Le système proposé est entraîné et évalué sur des données synthétiques, ainsi que sur des données réelles, mettant en évidence de bons résultats par rapport à l'état de l'art. Les avantages des estimations de l'incertitude du suivi sont illustrés pour l'estimation du mouvement visuel.

5.3 Résultats et évaluations

Pour cette section, le détecteur de Harris et le tracker KLT sont choisis. Néanmoins, la méthode DUNE proposée peut être appliquée à toute combinaison d'algorithmes d'extraction et de suivi de points d'intérêt ou d'appariement. Nous allons évaluer DUNE avec des jeux de données issus de SYNTHIA puis KITTI.

> ***Nota Bene*** Les paramètres de Harris (λ_1, λ_2) sont propres aux détecteurs de Harris. De manière analogue, l'utilisation d'autres détecteurs pourra conduire à l'intégration d'autres indicateurs représentatives à la qualité des points pour le détecteur envisagé.

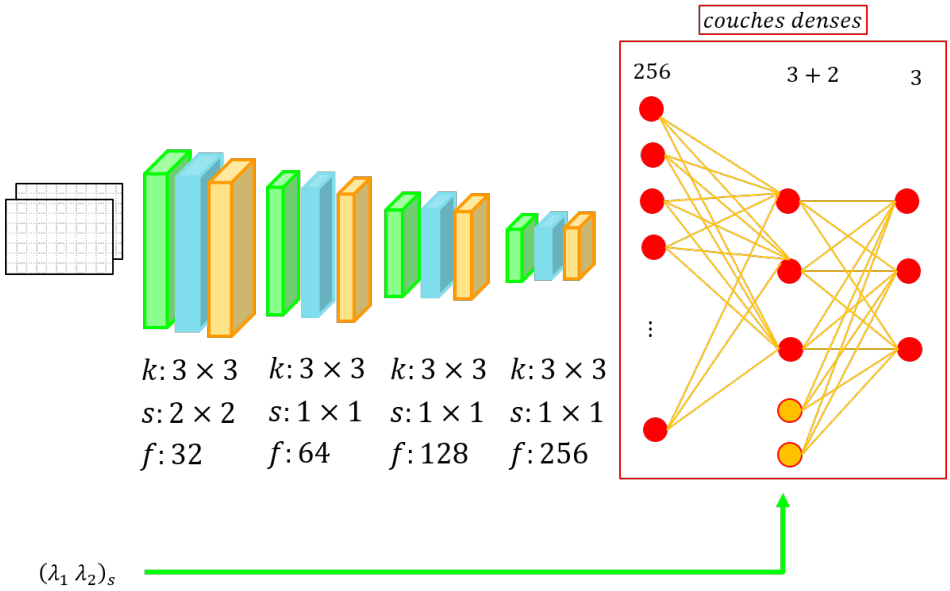


FIGURE 5.9 – Structure de l'estimateur d'incertitude sur la position des points d'intérêt DUNE intégrant les paramètres de Harris (λ_1, λ_2) - k indique la dimension des Kernels ; s la dimension des strides ; et f le nombre de filtres. Les couches vertes représentent les convolutions, les bleues les couches de normalisation et les jaunes les dropouts.

5.3.1 Métriques d'évaluation

L'évaluation des résultats nécessite la mise en place de métriques pour mesurer la validité des matrices de covariance estimées par DUNE.

Soit $s \in \mathbb{N}$ la taille du dataset $\mathcal{D}$, et $i \in |[1; s]|$ un échantillon du jeu de données $\mathcal{D}$ (définit § 5.2.1). L'évaluation est opérée entre l'erreur ${}^{uv}\boldsymbol{e}_i$, représentant la vérité terrain, et l'incertitude prédite par l'estimateur d'incertitude DUNE $\boldsymbol{\Sigma}_i$.

La première métrique est l'erreur quadratique moyenne normalisée, en anglais *Normalized Norm Error*, noté par la suite NNE :

$$NNE = \frac{1}{N} \sum_{i=1}^{s} \sqrt{\frac{||\boldsymbol{e}_i||_2^2}{tr(\boldsymbol{\Sigma}_i)}} \tag{5.9}$$

où $|\boldsymbol{e}_i||_2^2$ est la norme quadratique de l'erreur et $tr(\boldsymbol{\Sigma}_i)$ définit la trace de l'incertitude. Le modèle d'incertitude optimal est obtenu pour $NNE = 1$. Les valeurs inférieures qualifient une estimation pessimiste de l'incertitude, et les valeurs supérieures caractérisent un modèle d'estimation optimiste. NNE est un indicateur sur la précision des variances prédites, mais il ne prend pas en compte les covariances diagonales (définit § 2.4).

Nota Bene Un modèle d'estimation de l'incertitude optimiste aura le rôle d'un filtre : l'estimateur est sous-dimensionné. En conséquence, de l'information utile est filtrée au risque de ne pas avoir suffisament d'information pour entretenir l'application qui l'intègre. À l'inverse, un modèle estimation pessimiste aura tendance a inclure toutes les informations, même les informations bruitées. Le modèle est alors sur-dimensionné.

La deuxième métrique est la distance de Mahalanobis (MD) qui mesure la distance entre la vérité terrain et la mesure normalisée par la matrice de covariance estimée par DUNE. Cette métrique inclut l'évaluation des coefficients diagonaux de la covariance. La MD est donnée par :

$$MD = \frac{1}{N} \sum_{i=1}^{s} \sqrt{\frac{\boldsymbol{e}_i^T {\boldsymbol{\Sigma}_i}^{-1} \boldsymbol{e}_i}{dim(\boldsymbol{e}_i)}} \tag{5.10}$$

Comme pour NNE, MD atteint sa valeur optimale à 1, les valeurs inférieures ou supérieures caractérisant respectivement des modèles d'estimation de l'incertitude pessimistes et optimistes.

Nous supposons que l'erreur ${}^{uv}\boldsymbol{e}_i \sim \mathcal{N}(0, \boldsymbol{\Sigma}_i)$ est gaussienne. On peut donc vérifier l'inégalité suivante (Eq. 5.11) comme indicateur de la dynamique du modèle d'estimation

$$\begin{aligned} -n\sigma_u \geq^{uv} \boldsymbol{e}_u \geq +n\sigma_u \\ -n\sigma_v \geq^{uv} \boldsymbol{e}_v \geq +n\sigma_v \end{aligned} \tag{5.11}$$

où σ (§ 2.4, eq. 2.31) définit l'écart type du modèle prédit, n est le nombre d'écart type et ${}^{uv}\boldsymbol{e}$ les erreurs observées. Pour une distribution normale, les quantiles pour $n = 1$, 2 et 3 devraient inclure respectivement 68%, 95% et 99,7% des erreurs observées ${}^{uv}\boldsymbol{e}$.

5.3.2 Estimateur DUNE appliqué à SYNTHIA

5.3.2.1 Préparation des données

Nous évaluons d'abord DUNE sur le jeu de données SYNTHIA [Ros+16], tiré de la dernière version SYNTHIA-AL [ZB+19]. Nous créons notre jeu de données (§ 4.3.2) avec un total de 56508 échantillons d'entrée provenant de 12 scénarios différents faisant partie du dataset SYNTHIA-AL-Train. Chaque scénario est urbain ou semi-urbain et comprend diverses perturbations visuelles telles que la pluie ou la surexposition. Les séquences vidéo ont une longueur de 250 à 750 images. Un maximum de 1000 points par image est détecté par le détecteur de Harris, les points correspondant à des objets dynamiques sont supprimés. Enfin, les erreurs mesurées des points trackés supérieures à la taille de l'imagette $\boldsymbol{W}$ sont supprimées, afin de ne pas polluer le jeu de données d'entraînement – de telles erreurs pourraient facilement être supprimées au cours d'application de VO (odométrie visuelle) grâce à un processus RANSAC [FB81] par exemple.

5.3.2.2 Paramétrisation de DUNE

70% des données décrites dans la section 5.3.2.1 sont utilisées pour entraîner le réseau neuronal, 15% sont conservées pour la phase de test et 15% pour l'évaluation. Nous effectuons un apprentissage sur 300 epochs avec une taille de bacth de 64. Une *epoch* définit une itération d'apprentissage, et le *batch* représente le lot de données minimisé pour chaque itération. Le pas d'apprentissage initial est de $1 \cdot 10^{-4}$ et diminue lorsqu'un plateau est atteint et persiste sur 8 epochs. Nous choisissons Nadam comme optimiseur (§ 5.1.3) avec ses paramètres fixés à $\beta_1 = 0.9$ et $\beta_2 = 0.99$. Le modèle final est sélectionné comme étant celui qui donne les meilleurs résultats sur les données de test. Cette évaluation est effectuée après chaque epoch.

Nota Bene Le test et l'évaluation consistent à faire une estimation du réseau à un instant donné avec un modèle de RdN dont les poids sont fixes. La phase de test est réalisée au cours de l'apprentissage : à chaque itération de minimisation et rétropropagation, un modèle de RdN est ajusté. Les données de test sont appliquées au modèle intermédiaire et évaluent le RdN pendant la phase d'apprentissage. À l'inverse, la phase d'évaluation consiste à évaluer le modèle de RdN à l'issue de l'apprentissage avec des données d'évaluation (non utilisés pendant l'apprentissage ni le test).

5.3.2.3 Évaluation de l'intégration des paramètres de Harris

Nous avons observé la corrélation entre les paramètres de Harris et la durée de vie du tracking d'un point d'intérêt (§ 5.2.2.2). Afin d'évaluer l'impact de l'intégration des paramètres de Harris dans le RdN DUNE (§ 5.2.2.3) nous effectuons 5 sessions d'apprentissage avec DUNE (§ 5.2.2.1) et DUNE + (λ_1, λ_2) (§ 5.2.2.3). Le tableau 5.2 compare les résultats de DUNE et de DUNE avec la réponse de Harris. Les valeurs indiquées sont obtenues en faisant la moyenne des 5 sessions d'apprentissage.

Méthode / Métrique moyenne	DUNE	DUNE + (λ_1, λ_2)	Valeures idéales
3σ	98.04	96.98	99.7
2σ	93.75	91.28	95
1σ	76.23	69.93	68
MD	0.83	**0.99**	1
NNE	0.77	**0.89**	1

TABLE 5.2 – Influence des paramètres de Harris.

L'évaluation moyenne globale de DUNE sans les paramètres de Harris tend à être pessimiste et met en évidence l'impact des paramètres de Harris sur les estimations de DUNE qui conduit à de meilleurs résultats avec $MD \sim 1$.

5.3.2.4 Résultats de l'apprentissage sur SYNTHIA

Dans cette section, nous présentons les estimations de l'incertitude sur la position des points d'intérêt de deux scénarios sélectionnés, l'un avec un mouvement de rotation de la caméra (Fig. 5.10, 726 points suivis), et l'autre avec un mouvement rectiligne lisse (Fig. 5.11, 949 points suivis). Les Fig. 5.10c et Fig. 5.11c comparent les estimations de la variance avec les erreurs de suivi de la vérité terrain.

Il est intéressant de noter que les estimations de variance sont très différentes entre u et v pour un mouvement de rotation (Fig. 5.10c), alors que pour un mouvement rectiligne (Fig. 5.11c) elles sont plus petites. Dans les deux cas, les prédictions de variance correspondent bien aux erreurs de la vérité de terrain – voir par exemple la vue rapprochée autour du point d'intérêt 280 sur la Fig. 5.11.

DUNE montre sa capacité d'adaptation sur les prédictions de variance en fonction des erreurs de tracking mesurées, indépendamment du type de mouvement. Les estimations issues de DUNE sont bien ajustées à la vérité terrain des erreurs, indiquant que le système est capable de capturer correctement l'incertitude sous l'hypothèse d'une erreur gaussienne.

5.3.2.5 Comparaison à l'état de l'art

Nous comparons nos résultats à la méthode [WM17] basée sur l'estimation incrémentale de l'incertitude avec une valeur initiale constante fixée à 0.5 pixel. Le modèle d'erreur de tracking le plus souvent considéré dans la littérature sur l'estimation du mouvement visuel utilise une covariance fixe (Fix-C) égale à 0.5 pixel (5.12).

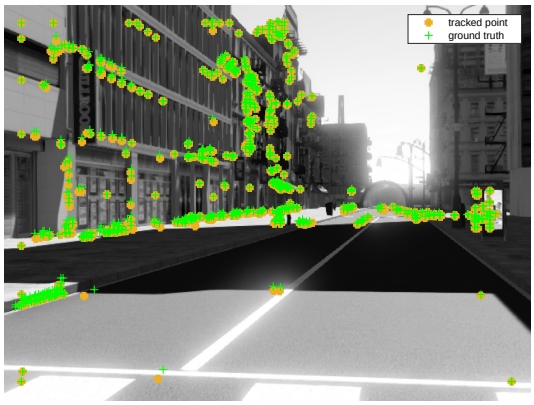
$$Fix - C = \begin{pmatrix} \sigma_{uu} & 0 \\ 0 & \sigma_{vv} \end{pmatrix} = \begin{pmatrix} 0.5 & 0 \\ 0 & 0.5 \end{pmatrix} \tag{5.12}$$

Les résultats sont présentés dans le tableau 5.3.

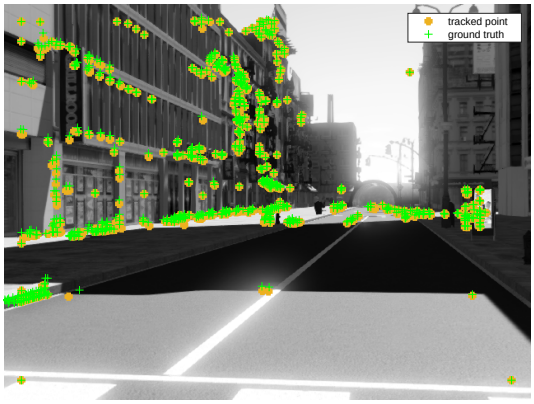
Métriques	DUNE + (λ_1, λ_2)	[WM17]	Fix-C	Valeures idéales
$3\sigma_u$	97.08	97.71	99.95	99.7
$3\sigma_v$	95.75	98.84	99.91	99.7
$2\sigma_u$	91.28	95.21	99.36	95
$2\sigma_v$	90.03	96.80	99.69	95
$1\sigma_u$	71.27	84.46	96.02	68
$1\sigma_v$	70.84	87.74	97.21	68
MD	**1.02**	0.68	0.32	1
NNE	**0.8**	0.58	0.3	1

TABLE 5.3 – Comparaison des résultats de DUNE par rapport à l'état de l'art.

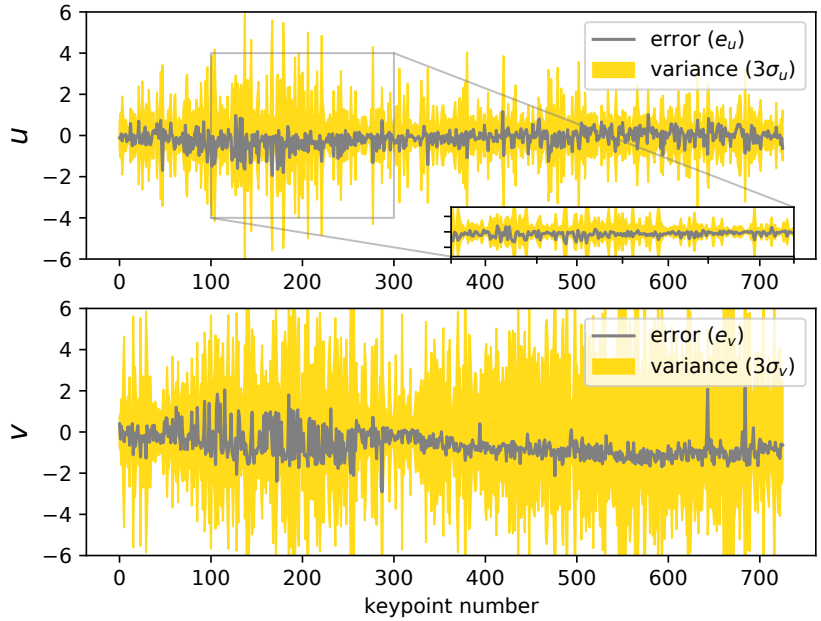
On peut remarquer que la méthode [WM17] et la méthode Fix-C produisent toutes deux des estimations très pessimistes. En effet, $MD < 0.7$ pour [WM17] et $MD < 0.5$ pour Fix-C. Ces résultats sont également visibles à 1σ où Fix-C couvre $\sim 96.6\%$ de l'erreur et



(a) Image k



(b) Image $k+1$

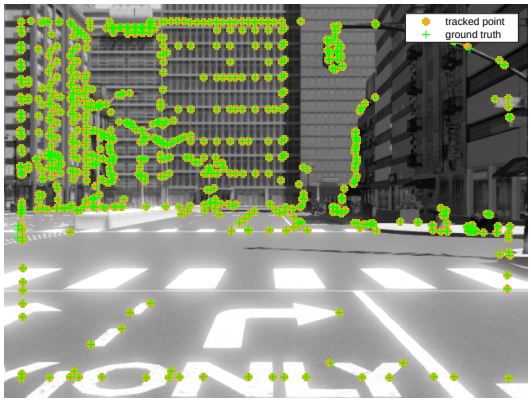


(c) Évolution de l'incertitude prédite, par rapport à l'erreur de tracking

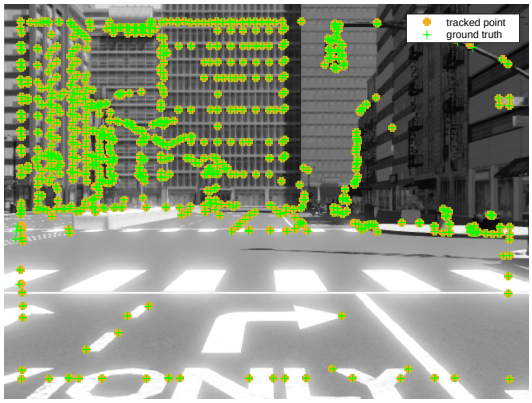
FIGURE 5.10 – Incertitudes estimées en réponse à une scène avec un mouvement de rotation.

[WM17] $\sim 86,1\%$. Les quantiles sont uniquement des indicateurs de la dynamique du modèle prédit, mais n'ont pas pour but d'évaluer la normalité d'un modèle supposé gaussien. Avec $MD = 1.02$, DUNE + (λ_1, λ_2) surpasse les deux approches par une marge significative.

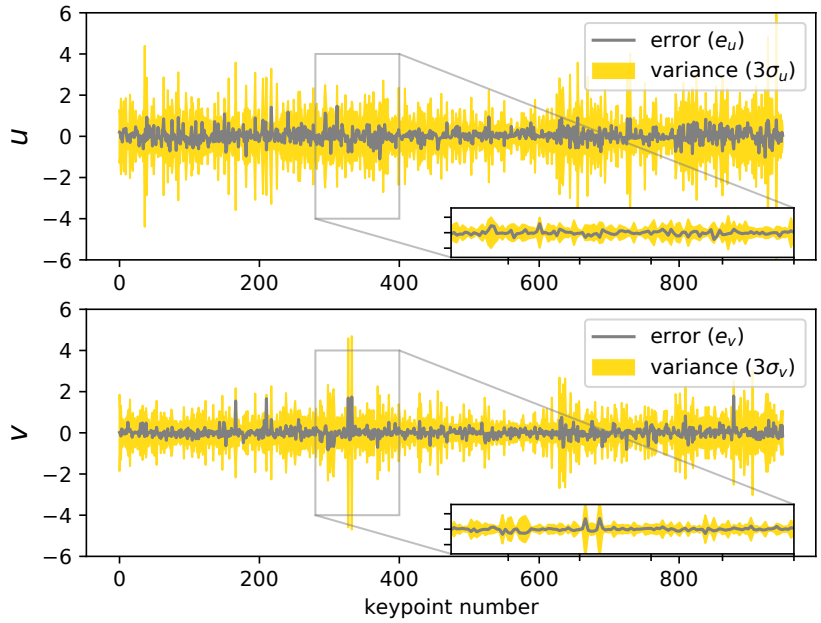
Cette analyse est mise en évidence par la Fig. 5.12, représentant l'incertitude prédite par DUNE + (λ_1, λ_2) par rapport à [WM17] et Fix-C. Il apparaît que DUNE présente une très bonne dynamique et s'adapte le mieux à l'évolution de l'erreur par rapport à [WM17] et à Fix-C. Ceci est particulièrement visible aux points d'intérêt 180 et 380 où l'erreur est très faible.



(a) Image k



(b) Image $k+1$



(c) Évolution de l'incertitude prédite, par rapport à l'erreur de tracking

FIGURE 5.11 – Incertitudes estimées en réponse à une scène avec un mouvement rectiligne uniforme

5.3.2.6 Évaluation des prédictions de DUNE sur l'estimation du mouvement

Nous proposons une validation indirecte pour évaluer la pertinence de notre travail par une simple application d'odométrie visuelle estimant le mouvement entre deux intants k et $k+1$.

Pour chaque image, les points trackés sont répartis en deux catégories : the *best points* (*i.e.* les meilleurs points) et the *worst points* (les pires points). Le premier ensemble regroupe les points vérifiants $\sqrt{tr(\mathbf{\Sigma}_i)} \leq 0,5$, $i \in |[0;s]|$. Il caractérise les meilleurs points (the *best points*). Le second ensemble vérifie $\sqrt{tr(\mathbf{\Sigma}_i)} > 0,5$ (le *points les plus mauvais*).

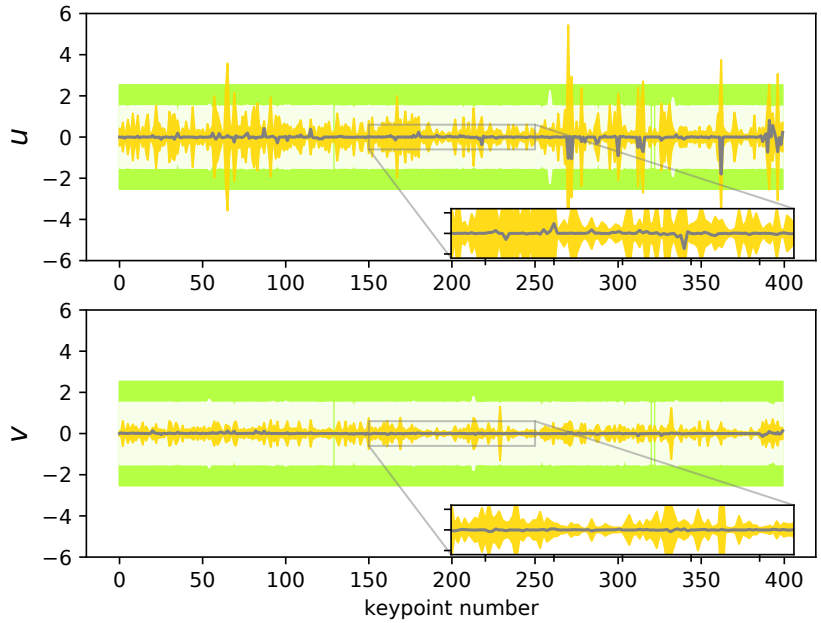


FIGURE 5.12 – Comparaison des prédictions de DUNE à l'état de l'art - La ligne grise représente l'erreur dans les directions u et v d'un scénario avec un mouvement rectiligne, ■ décrit la méthode Fix-C à 3σ, □ fait référence à [WM17] pour 3σ et ■ représente les prédictions d'incertitudes issues de DUNE à 3σ.

Le mouvement de la caméra est estimé pour chaque catégorie indépendamment avec EPnP [LMNF09], en utilisant la position 3D des éléments suivis au temps k fournie par SYNTHIA

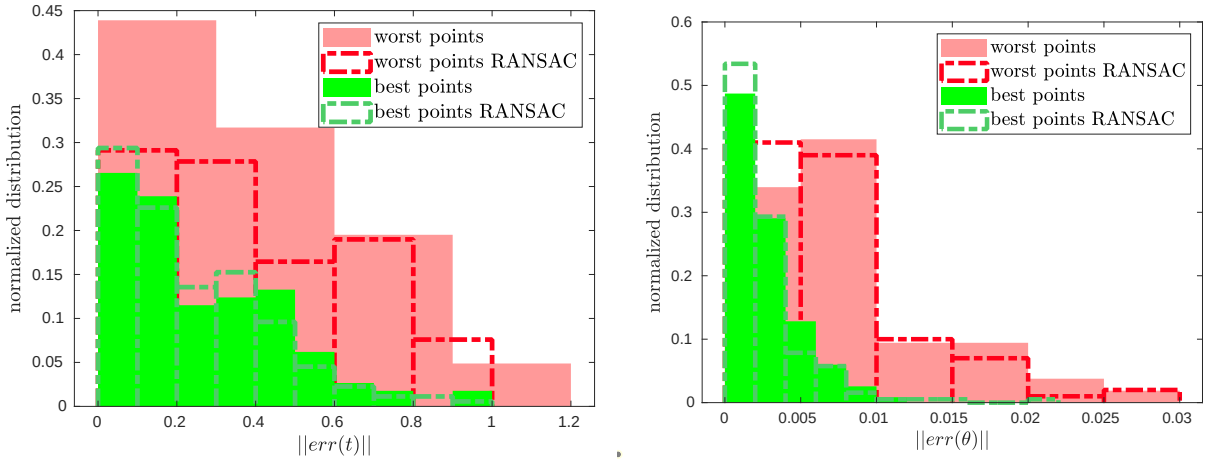
> ***Nota Bene*** En aucun cas ce processus n'est une approche pour la reconstruction de la pose, ni pour le rejet des valeurs aberrantes : le but est d'évaluer les bénéfices apportées par la prédiction de variance issue de DUNE sur l'estimation du mouvement.

En utilisant la vérité terrain de la pose de la caméra pour chaque image fournie par SYNTHIA, nous calculons pour les deux ensembles l'erreur entre le mouvement réel et le mouvement estimé :

$$ {}_{k+1}T_{err} = {}_{k+1}T_{groundtruth} \cdot {}_{k+1}T^{-1}_{estimated} \tag{5.13}$$

où ${}_{k+1}T_{err}$ définit l'erreur de transformation rigide à l'instant $k+1$. Les résultats sont présentés sur la Fig. 5.13. Tous nos résultats sont comparés à une estimation de mouvement intégrant RANSAC [FB81], avec le même seuil. La Fig. 5.13(a) montre que de plus grandes incertitudes, issues des *worse points*, tendent à produire plus d'erreur en translation avec une erreur de translation moyenne de 0.67m ; alors que les meilleurs points ont en moyenne 0.2m d'erreur.

Ce phénomène est accentué sur la Fig. 5.13(b), où l'erreur moyenne pour les meilleurs et les pires points est respectivement de $2 \cdot 10^{-3}$ deg et $8 \cdot 10^{-3}$ deg. Notez que les estimations produites en appliquant RANSAC sur EPnP conduisent à une erreur légèrement plus faible dans les deux cas, ce qui est attendu.



(a) (b)

FIGURE 5.13 – Impact sur l'estimation du mouvement - (a) Erreur sur la translation (en m) ; (b) Erreur sur la rotation (en deg).

5.3.3 Estimateur DUNE appliqué à KITTI

5.3.3.1 Préparation des données

Afin de valider notre approche dans des scénarios réels, nous testons également DUNE sur les données brutes KITTI [Gei+13] (§ 4.3.4) en utilisant les séquences `2011_09_26_0005`, `2011_09_29_0013`, `2011_09_26_0026` et `2011_09_26_0002`. Les séquences `2011_09_26_0005`, `2011_09_29_0013` et `2011_09_26_0026` sont utilisées pour nourrir le modèle pendant la phase d'apprentissage et `2011_09_26_0002` est utilisée pour la phase d'évaluation. Un total de 266905 échantillons est extrait pour l'apprentissage et 30188 échantillons pour l'évaluation.

La structure 3D exacte du monde (*i.e.* carte de profondeur, points géolocalisés, etc ..) n'est pas disponible directement. Elle requiert une analyse préalable de l'environnement nécessaire pour construire précisément la vérité terrain d'un point tracké ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$ à $k+1$, telle que défini précédement comme la reprojection du point détecté ${}^{uv}\boldsymbol{m}_k$ à k (§ 4.3.2).

Toutefois, une vérité terrain disponible directement est donnée par la position d'un point détecté ${}^{uv}\boldsymbol{m}_k$ à un instant k. L'application du tracker en rétropropagation (*i.e.* sur un aller-retour) permet de mesurer le biais induit par la méthode de tracking et en particulier, d'évaluer l'incertitude de position sur ce point. Nous définissons alors une erreur cumulée comme la distance entre la position du point détecté ${}^{uv}\boldsymbol{m}_k$ à k et la position du point tracké sur un aller-retour ${}^{uv}\tilde{\boldsymbol{m}}_k$ tel que décrit (§ 4.3.4). Cette erreur peut également être réécrite ${}^{uv}_{Harris}\boldsymbol{m}_k - {}^{uv}_{KLT}\tilde{\boldsymbol{m}}_{k|k+1}$.

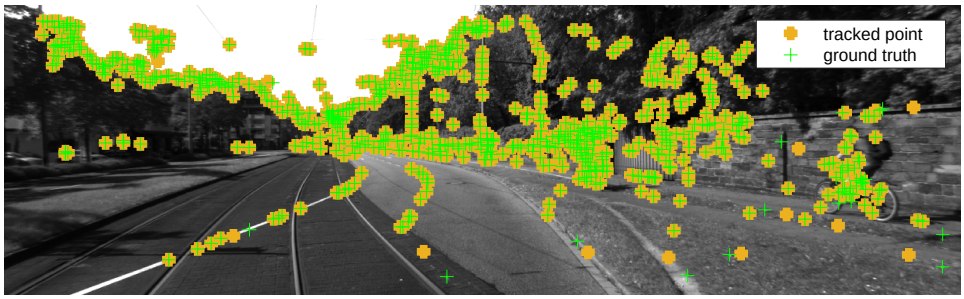


FIGURE 5.14 – Scénario KITTI - +représentent les points ${}^{uv}\boldsymbol{m}_k$ détectés par Harris ; +les points trackés sur un aller-retour ${}^{uv}\tilde{\boldsymbol{m}}_k$.

La limite de cette approche est donnée par la taille de la fenêtre temporelle pour laquelle la dynamique de l'erreur est supposée uniformément répartie.

5.3.3.2 Paramétrisation de DUNE

De façon similaire à l'évaluation sur les données SYNTHIA, présentée § 5.3.2.1, nous effectuons un entraînement sur 500 epoch avec une taille de batch de 64. Le pas d'apprentissage initial est de $1 \cdot 10^{-5}$.

Tout comme précédemment (§ 5.3.2.2), 70% des données sont utilisées pour la phase d'entrainement, 15% pour le test et 15% pour l'évaluation. L'optimiseur Nadam est conservé.

5.3.3.3 Résultats de l'apprentissage sur KITTI

Les résultats sont évalués sur une fenêtre temporelle glissante de taille 2 : à l'instant k, la détection de Harris est appliquée, suivi par le tracker KLT à $k+1$, puis de nouveau KLT à k. Fig. 5.14 représente une image à k avec les points détectés ${}^{uv}\boldsymbol{m}_k$ et les points tracké sur un aller-retour ${}^{uv}\tilde{\boldsymbol{m}}_k$.

La séquence évaluée compte 670 points d'intérêt détectés et trackés. DUNE produit des incertitudes qui s'adaptent au modèle d'erreur mettant en évidence une très bonne dynamique (Fig.5.15). On observe, en particulier, autour du point 100 un pic dans l'erreur de suivi mesurée avec, en réponse, une incertitude sur la position en u et v importante. Puis autour du point 120, une erreur de suivi proche de 0 et en conséquence une incertitude prédite également très petite et proche de 0.

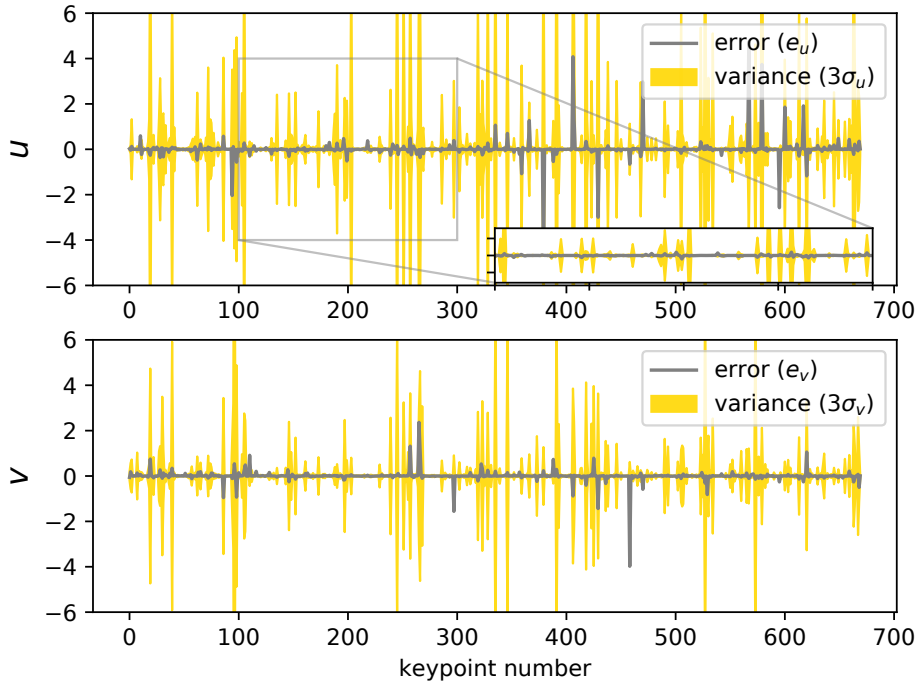


FIGURE 5.15 – Évolution de l'incertitude sur la position des points d'intérêt prédite par DUNE et comparaison à l'erreur de tracking.

5.3.3.4 Comparaison à l'état de l'art

Le tableau 5.4 compare DUNE entraîné sur le jeu de données issu du dataset KITTI avec l'état de l'art. DUNE est mis en relation avec [WM17] et Fix-C (eq. 5.12). La covariance constante de Fix-C est fixée à $0.5 \cdot 2$ pixels. En effet, puisque le réseau de neurone est appris à partir de l'erreur de suivi sur un aller-retour.

Métriques	DUNE+ (λ_1, λ_2)	[WM17]	Fix-C	Valeures idéales
$3\sigma_u$	94.97	99.53	98.92	99.7
$3\sigma_v$	94.83	99.73	99.22	99.7
$2\sigma_u$	91.08	99.34	98.59	95
$2\sigma_v$	91.05	99.55	98.93	95
$1\sigma_u$	79.34	99.86	97.44	68
$1\sigma_v$	79.27	98.17	98.18	68
MD	**1.39**	0.17	0.3	1
NNE	**1.39**	0.10	0.24	1

TABLE 5.4 – Comparaison des résultats de DUNE par rapport à l'état de l'art.

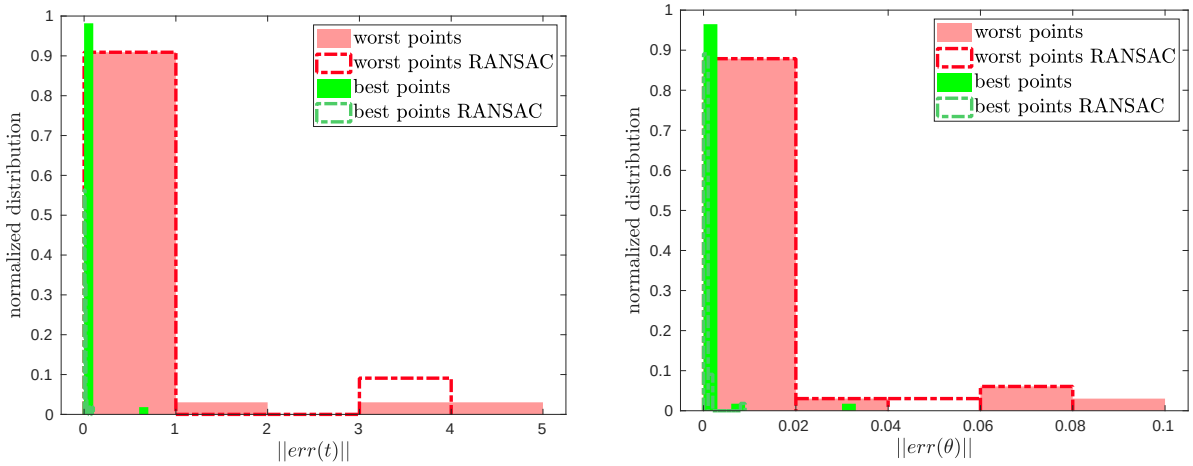
D'après Tab. 5.4, les deux approches [WM17] et Fix-C sont très pessimistes avec $MD = 0.16$ et $MD = 0.3$ respectivement. Le modèle DUNE surpasse les deux méthodes sur des données réelles, avec $MD = 1.39$.

5.3.3.5 Évaluation des prédictions de DUNE sur l'estimation du mouvement

De façon similaire au § 5.3.2.6, nous appliquons une validation géométrique dont la vocation est d'évaluer l'impact des variances sur la reconstruction du mouvement.

Dans le cas actuel, la pose de la caméra n'est pas connue. Néanmoins, l'approche de la retropropagation permet de revenir à l'état initial. Ainsi, la pose du mouvement entre k et k est identique, et la vérité terrain caractérisant la transformation de k à k est définie par la matrice d'identité ${}_{k+1}T_{groundtruth} = \begin{bmatrix}\mathcal{I}_{3\times3} & \mathbb{0}_{3\times1}\end{bmatrix}$. En effet, le mouvement avant-arrière fournit un mouvement connu et théoriquement parfait.

Par ailleurs, la transformation prédite ${}_{k}T_{estimated}$ (5.13) en $3D$ est évaluée par stéréovision lors de la détection initiale de Harris, ce qui permet de trianguler les points de $2D$ en $3D$. L'erreur sur la transformation rigide peut ainsi être calculée.



(a) (b)

FIGURE 5.16 – Impact sur l'estimation du mouvement - (a) Erreur sur la translation (en m) ; (b) Erreur sur la rotation (en deg).

Les résultats sont illustrés Fig. 5.16. L'erreur moyenne de translation est de 0.0128m pour les meilleurs points (*i.e. best points*) sélectionnés, et de 0.3310m pour les plus mauvais (*i.e. worst points*) telle que le montre la Fig. 5.16(a). En ce qui concerne l'erreur sur la rotation, les points d'intérêt issus des *best points* produisent une erreur moyenne de $6 \cdot 10^{-4}$ tandis que les *worst points* atteignent $8 \cdot 10^{-3}$. Dans les deux cas, un facteur 10 différencie les deux collections : *best points* et *worst points*.

5.3.4 Discussion

On peut remarquer la différence entre les résultats de SYNTHIA (Fig. 5.13) et de KITTI (Fig. 5.16). En effet, KITTI met en évidence une amélioration d'un facteur 10 sur l'estimation du mouvement entre les meilleures points (*i.e. best points*) et les pires points (*i.e. worst points*) sélectionnés, alors que SYNTHIA montre un facteur d'environ 3 ou 4.

La force du jeu de données SYNTHIA est la disponibilité directe de la vérité terrain donnée

par la reprojection ${}^{w}\boldsymbol{M}$ dans $\boldsymbol{\mathcal{R}}_{pix}$, notamment à travers la connaissance de la position de ${}^{w}\boldsymbol{M}$ fournie par la carte dense de profondeur. Néanmoins, la précision de la position de ${}^{w}\boldsymbol{M}$ conditionne la précision de la vérité terrain. En particulier, la précision des cartes de profondeur dépend du pas d'échantillonage des cartes. Par ailleurs, les données synthétiques produisent des bruits indésirables tels que l'aliasing.

Les résultats sur le dataset KITTI ont permi de montrer la faisabilité de la méthode appliquée à des données réelles. KITTI a même conduit à de meilleurs résultats. La mesure de la vérité terrain est très précise et ne nécessite aucune étape de reprojection, limitant les approximations de modèle. Toutefois, la distribution de l'erreur est supposée uniforme le long d'une période ΔT limitant la taille de la fenêtre temporelle étudiée. Cette hypothèse suppose un mouvement uniforme et aucun changement brusque dans l'environnement au cours de la période ΔT.

En conclusion, l'approche KITTI fournit une vérité terrain parfaite en appliquant un mouvement en aller-retour. SYNTHIA s'est avéré utile pour prouver et valider notre modèle, et reste une première approximation facile à mettre en œuvre. En effet, l'approche peut être facilement intégrable dans un système de navigation puisqu'elle s'est affranchie de l'information de profondeur pour construire sa mesure de vérité terrain.

5.4 Conclusion

Nous avons présenté une méthode basée sur des CNN qui produit des estimations de l'incertitude sur la position des points d'intérêt suivis par le tracker KLT. Nous avons montré qu'il est possible de capturer la dynamique du modèle d'erreur observé. Le modèle prédictif DUNE a été validé par deux métriques dédiées, et un schéma d'estimation de mouvement montre que les incertitudes estimées peuvent aider à sélectionner les meilleurs points d'intérêt pour l'estimation de mouvement. L'étape suivante consiste à intégrer ces incertitudes estimées dans un système de navigation vision-inertie (VINS) : les incertitudes pourraient d'abord aider à sélectionner les meilleurs points pour l'estimation du mouvement (parmi d'autres critères connus, tels qu'une répartition équilibrée des points sur l'ensemble de l'image), et être utilisées pour estimer une incertitude précise pour les mouvements calculés.

D'autres améliorations de l'architecture de DUNE peuvent également être envisagées, comme la combinaison de DUNE avec un réseau récurrent, exploitant l'ensemble de l'historique de suivi à travers une séquence pour fournir des incertitudes plus précises.

Chapitre 6

Intégration de navigation inertielle et visuelle

La connaissance s'acquiert par l'expérience, tout le reste n'est que de l'information.

Albert Einstein

Nous avons construit un modèle d'erreur DUNE basé sur de l'apprentissage profond capable d'estimer les incertitudes sur la position des points d'intérêt. Cette qualification de la précision des mesures est particulièrement intéressante lors d'une fusion de donnée serrée. En effet, une caractérisation fine de la précision de chaque mesure impliquée lors d'une fusion de donnée réduit l'erreur globale résultante. Dans ce chapitre, nous allons intégrer les incertitudes sur la position des points d'intérêt à un système de navigation VINS [1]. En particulier, nos matrices de covariance sont appliquées à un filtrage serré issu de la plateforme OpenVINS [Gen+20]. L'impact de ces covariances est évalué avec le dataset EUROC [Bur+16] et des leviers d'améliorations sont proposés, notamment la recherche active de points d'intérêt en vue d'améliorer l'estimation de la pose du système mobile.

6.1 Systèmes de Navigation Inertie/Vision (VINS)

Les systèmes de navigation Inertie/Vision (VINS) [Hua19] sont communément utilisés dans le but d'estimer les poses (positions et orientations) de systèmes mobiles (*i.e.* véhicules). Ils sont, en particulier, performants dans des environnements en intérieur là où le GPS ne permet pas de fournir des informations sur la localisation d'un système mobile. Ils sont constitués d'une ou plusieurs centrales inertielles ainsi que d'au moins une caméra.

La plupart des centrales inertielles ou unités de mesure inertielle (IMU) sont équipées de 6 capteurs qui mesurent les accélérations et les vitesses angulaires du système mobile dans les trois directions. La position du système mobile est dès lors estimée par l'intégration de l'accélération et l'orientation par l'intégration des vitesses angulaires. Malheureusement, l'intégration de mesures IMU est corrompue par du bruit et des biais aboutissant souvent à des estimations de pose peu fiables pour la navigation à long terme. Bien qu'il existe des IMU à hautes performances, leurs coûts restent prohibitifs pour un déploiement à grande échelle. Les caméras ont l'avantage d'être compactss, légèrss et économiques. Leur nature permet de fournir une description précise de l'environnement offrant la possibilité à certaines applications

1. Visual Inertial Navigation System

de construire une carte de l'environnement dans lequel le système mobile évolue. Toutefois, la caméra est elle même soumise à des bruits de modèle [Zha98] et des biais induisant une dérive du mouvement avec l'accumulation d'erreur sur la localisation du système mobile conduisant à une navigation biaisée sur le long terme.

La dérive impactant la trajectoire estimée du système mobile est communément corrigée par la réception d'informations ponctuelles de capteurs complémentaires (*i.e.* GPS, ...). Le manque d'information sur la localisation globale du système mobile, les différentes approximations de modèle ainsi que la mauvaise caractérisation du bruit de mesure des capteurs impactent directement la dérive de la position du système mobile qui grandit.

Il existe plusieurs approches de fusion de données dont les méthodes d'optimisation [Tan+22][MAMT15] et les approches par filtrage. Les méthodes basées sur de l'optimisation minimisent l'ensemble des positions prises par le système mobile le long d'une trajectoire, tandis que les méthodes par filtrage prédisent l'état courant du système mobile à partir d'un modèle d'évolution dynamique corrigé par l'apport de nouvelles mesures capteurs.

Nota Bene La réduction de l'erreur sur l'estimation de la pose d'un système mobile dépend de deux choix principaux :

- La méthode de fusion
- Type de couplage

Filtrage VS optimisation [Qin+22] : Le filtre de Kalman (KF) [KO60] permet d'estimer les paramètres d'un système mobile ayant une évolution temporelle à partir de mesures bruitées. Il intègre directement la matrice de covariance dans son modèle et détermine l'erreur moyenne de son estimation. Les méthodes d'optimisation, en revanche, sont fondées sur la résolution d'un problème de moindres carrés non linéaires (*bundle adjustment* en français l'ajustement de faisceaux [Bro58]) sur un ensemble de mesures, permettant la réduction de l'erreur par la relinéarisation [Kum+11] mais avec un coût de calcul élevé.

Globale VS locale (ou lâche VS serrée) [Hua19] : Le couplage *lâche* fusionne le résultat d'un pré-traitement des mesures. Les mesures sont traitées indépendemment les unes des autres et supposées non-corrélées induisant de l'erreur sur l'estimation finale. À l'inverse, le couplage *serré* [MR07b] fusionne directement les mesures observées au sein d'un seul processus, induisant une plus grande précision.

La plupart des solutions existantes [ZMG22] aujourd'hui ne permettent pas d'estimer l'incertitude des mesures visuelles et limitent les méthodes d'optimisation ou de filtrage serrées. Une approximation populaire considère une précision uniforme des observations dans l'image, et constante au cours du temps [MAMT15]. Cette hypothèse est acceptable dans un environnement statique ou quasi-statique, mais devient inadaptée dans un environnement dynamique soumis à des effets visuels (*i.e.* flou optique).

L'avènement récent des réseaux de neurones profonds a conduit à l'apparition de modèles d'erreur évaluant et corrigeant la dérive temporelle de localisation d'un système mobile suite

à une *hybridation*[2] lâche [Li+22][DML20][Liu+18][Wan+18][PK17]. Bien que montrant des résultats encourageant, les approches lâches sont, par définition, soumises à plus d'erreur et sont donc moins précises que les hybridations serrées. L'objectif de ce chapitre est de mettre en œuvre une hybridation serrée intégrant des incertitudes adaptées sur les mesures visuelles et améliorer ainsi la précision finale de la localisation du système mobile.

Dans la suite de ce chapitre, nous présentons un système VINS intégrant une hybridation serrée grâce à la caractérisation de l'incertitude sur la position des points d'intérêt (chapitre 5) et montrons que l'erreur globale de localisation est minimisée. La fusion des données est réalisée par un filtrage de Kalman, détaillé (§ 6.1.1). En particulier, le filtre utilisé est issu de la plateforme OpenVINS [Gen+20] présentée (§ 6.1.2). Les résultats sont comparés entre un filtrage serré intégrant les covariances estimées par DUNE (chapitre 5) et le même filtre utilisant l'approximation populaire d'une incertitude constante pour toutes les observations issues d'un même capteur caméra. L'avantage de cette évaluation est double : elle montre l'impact direct de l'estimation de l'incertitude sur la position des points d'intérêt, et améliore la précision sur la localisation d'un système mobile par une hybridation serrée, l'approche la plus précise existante.

6.1.1 Filtrage de Kalman

Soit $k \in \mathbb{N}$, l'indice du pas de discrétisation du temps t. Le filtre de Kalman (KF) [KO60] est sans doute la technique la plus courante pour l'estimation d'état [YBHH15]. Il se base sur le filtrage bayésien. On note $\mathbf{x}_k \in \mathbb{R}^n$, $n \in \mathbb{R}$ l'état du filtre à l'instant k, alors l'évolution réelle de l'état est supposée dépendante de l'état précédent $\mathbf{x}_{k-1}$ (6.1).

$$\begin{cases} \mathbf{x}_k = f(\mathbf{x}_{k-1}) + \boldsymbol{w}_k \\ \mathbf{z}_k = h(\mathbf{x}_k) + \boldsymbol{v}_k \end{cases} \tag{6.1}$$

avec $\boldsymbol{w}_k \sim \mathcal{N}(0, \boldsymbol{Q}_k)$ l'incertitude de modèle qui est supposée être un bruit gaussien de moyenne nulle non corrélé dont la covariance est $\boldsymbol{Q}_k$, et $\boldsymbol{v}_k \sim \mathcal{N}(0, \boldsymbol{R}_k)$ l'incertitude liée aux observations qui est supposée être un bruit gaussien de moyenne nulle non corrélé dont la covariance est $\boldsymbol{R}_k$. $\mathbf{z}_k \in \mathbb{R}^m, m \in \mathbb{R}$ définit le vecteur des observations, et m le nombre d'observations qui le compose. f est la fonction d'évolution du modèle dynamique et h la fonction d'observation.

> ***Nota Bene*** f et h sont linéaires dans le cas du filtre de Kalman ordinaire

Le fonctionnement du filtre peut être divisé en deux étapes : la prédiction et la correction (ou le *recalage*).

2. Fusionner, combiner

6.1.1.1 Prédiction

L'étape de prédiction consiste à prédire l'état futur du système mobile à partir d'un modèle d'évolution du mouvement dynamique.

$$\hat{\mathbf{x}}_{k|k-1} = f(\hat{\mathbf{x}}_{k-1|k-1}) \tag{6.2}$$

$$\hat{\boldsymbol{P}}_{k|k-1} = \boldsymbol{F}_k \hat{\boldsymbol{P}}_{k-1|k-1} {F_k}^T + \boldsymbol{Q}_k \tag{6.3}$$

où $\hat{\mathbf{x}}$ désigne la prédiction de $\mathbf{x}$ et $\tilde{\mathbf{x}}$ son estimé. $\boldsymbol{F}$ est la matrice d'évolution décrivant le modèle dynamique du système mobile, $\boldsymbol{P}$ est la matrice covariance relative à la précision de l'état $\mathbf{x}$ et $\boldsymbol{Q}$ est la matrice de covariance liée au bruit d'évolution du modèle $\boldsymbol{w}_k$.

6.1.1.2 Mise à jour

La mise à jour permet de *recaler* (ou corriger) la prédiction réalisée par l'intégration de nouvelles observations dans le filtre. Ces observations sont intégrées ainsi que leurs covariances associées (*i.e.* la précision des observations).

$$\begin{aligned}
\tilde{\mathbf{y}}_k &= \mathbf{z}_k - h(\hat{\mathbf{x}}_{k|k-1}) && (6.4)\\
\boldsymbol{S}_k &= \boldsymbol{H}_k \hat{\boldsymbol{P}}_{k|k-1} \boldsymbol{H}_k^T + \boldsymbol{R}_k && (6.5)\\
\boldsymbol{K}_k &= \hat{\boldsymbol{P}}_{k|k-1} \boldsymbol{H}_k^T \boldsymbol{S}_k^{-1} && (6.6)\\
\mathbf{x}_{k|k} &= \hat{\mathbf{x}}_{k|k-1} + \boldsymbol{K}_k \tilde{\mathbf{y}}_k && (6.7)\\
\boldsymbol{P}_{k|k} &= (\mathcal{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \hat{\boldsymbol{P}}_{k|k-1} && (6.8)
\end{aligned}$$

où $\mathbf{z}_k$ est le vecteur des mesures observées à l'instant k, $\boldsymbol{H}_k$ est la matrice d'observation qui relie l'état $\mathbf{x}_k$ à la mesure $\mathbf{z}_k$, et $\boldsymbol{R}_k$ est la matrice de covariance du bruit de mesure $\boldsymbol{v}_k$. $\mathcal{I}$ représente la matrice identité. Le gain de Kalman $\boldsymbol{K}_k$ permet de minimiser la valeur de la covariance $\hat{\boldsymbol{P}}_{k|k-1}$ de l'état $\mathbf{x}_k$. La matrice de covariance $\boldsymbol{P}$ permet la pondération des observations lors de la fusion des données. Il est important de noter que les performances d'un filtre de kalman sont directement affectées par le choix des matrices de covariances $\boldsymbol{Q}$ et $\boldsymbol{R}$ [MS99][AWD10].

6.1.1.3 EKF non linéaire

Une extension non linéaire du filtre existe, le filtre de Kalman étendu (EKF) [Rib04], dans laquelle des approximations linéaires sont obtenues en utilisant un développement en série de Taylor du premier ordre. L'EKF suppose l'hypothèse d'une distribution gaussienne de tous les états initiaux et estimés. Dans ce cas, $\boldsymbol{F}_k$ et $\boldsymbol{H}_k$ sont les matrices jacobiennes décrivant respectivement l'évolution du modèle dynamique de l'état $\mathbf{x}_k$ et l'évolution des observations $\mathbf{z}_k$. Ces matrices permettent de se ramener à un modèle linéaire par linéarisation des fonctions d'évolution non linéaires.

Nota Bene Le filtre de Kalman suppose une évolution linéaire de l'état et de ses observations. Toutefois, en réalité la plupart des systèmes physiques sont non linéaires. L'application d'une linéarisation permet de s'affranchir de cette limitation mais contraint le filtrage à un intervalle de fonctionnement dans lequel l'approximation d'un modèle *linéaire* est acceptable. Au delà de cet intervalle, le filtrage n'est plus *optimal*[3].

[3] Un filtre optimal est défini par des fonctions linéaires et des erreurs gaussiennes

Filtrage de Kalman inertie vision [Ort07] propose un filtrage de kalman étendu qui intègre dans son état la carte de son environnement. Dans un souci de simplification, on suppose un système mobile constitué d'un capteur inertiel et une caméra dont les centres de gravité sont confondus. On note alors ${}^{w}\boldsymbol{p}_{IMU_k} = {}^{w}\boldsymbol{p}_{C_k} = {}^{w}\boldsymbol{p}_k$ la position du système mobile. Soit $\mathcal{X}$ l'état du système mobile, alors :

$$\mathcal{X} = \begin{bmatrix} {}^{w}\boldsymbol{p} & {}^{w}\boldsymbol{v} & {}^{w}\boldsymbol{\theta} & {}^{w}\boldsymbol{b}_a & {}^{w}\boldsymbol{b}_g \end{bmatrix}^T \tag{6.9}$$

où ${}^{w}\boldsymbol{v}$ est la vitesse du système mobile, ${}^{w}\boldsymbol{\theta}$ son orientation, et ${}^{w}\boldsymbol{b}_a$ et ${}^{w}\boldsymbol{b}_g$ constituent les biais accélérométrique et gyrométrique relatifs à l'IMU modélisés comme des processus aléatoires caractérisés par les bruits blancs Gaussiens $\boldsymbol{n}_{b_a}$ et $\boldsymbol{n}_{b_g}$. Soit $\mathcal{MAP}$ la carte de l'environnement de tous les amers 3D ${}^{w}\boldsymbol{M}$ dans $\boldsymbol{\mathcal{R}}_w$ qui sont observés par projection sur la caméra ${}^{uv}\boldsymbol{m}$.

$$\mathcal{MAP} = \begin{bmatrix} {}^{w}\boldsymbol{M}_1 & \cdots & {}^{w}\boldsymbol{M}_i & \cdots & {}^{w}\boldsymbol{M}_L \end{bmatrix}^T \tag{6.10}$$

L définit le nombre de *landmarks* (*i.e.* amers) en mémoire. L'état du filtre $\mathbf{x}_k$ peut alors s'écrire :

$$\mathbf{x}_k = \begin{bmatrix} \mathcal{X}_k & \mathcal{MAP}_k \end{bmatrix}^T \tag{6.11}$$

La phase de prédiction est réalisée par la fonction $f(.)$ du modèle d'évolution dynamique de l'IMU tel que décrit dans [RJM02] et [MR07b]. La vitesse ${}^{w}\boldsymbol{v}$ est obtenue par intégration de l'accélération mesurée par l'IMU, puis la position ${}^{w}\boldsymbol{p}$ par intégration de la vitesse ${}^{w}\boldsymbol{v}$. De même pour l'orientation ${}^{w}\boldsymbol{\theta}$ issue de l'intégration de la vitesse angulaire ${}^{w}\boldsymbol{\Omega}$.

Soit la fonction de projection ${}^{uv}\pi : (\mathbb{R}^3, SE(3)) \rightarrow \mathbb{R}^2$ telle que ${}^{uv}\boldsymbol{m} = {}^{uv}\pi\left({}^{w}\boldsymbol{M}, {}^{C_k}_{w}\boldsymbol{T}\right)$. Alors $h(.)$ le modèle d'observation d'un point ${}^{w}\boldsymbol{M}_i$ s'écrit :

$$h_i(\hat{\mathbf{x}}_{k|k-1}) = {}^{uv}\pi\left({}^{w}\boldsymbol{M}_i - {}^{w}\hat{\boldsymbol{p}}_k, {}^{C_k}_{w}\boldsymbol{T}\right) \tag{6.12}$$

$h_i(\hat{\mathbf{x}}_{k|k-1})$ représente la projection de ${}^{w}\boldsymbol{M}_i$ (cf chapitre 2) à partir de la position prédite ${}^{w}\hat{\boldsymbol{p}}_k = {}^{w}\hat{\boldsymbol{p}}_{C_k}$ de la caméra lors de la phase de prédiction. Les observations $\mathbf{z}$ correspondent à la position des points d'intéret ${}^{uv}\boldsymbol{m}$ dans l'image k.

$$\mathbf{z}_{k,i} =^{uv} \boldsymbol{m}_{k,i} \tag{6.13}$$

La reprojection de ${}^{w}\boldsymbol{M}_i$ est comparée à la position mesurée ${}^{uv}\boldsymbol{m}_{k,i}$ permettant de corriger ${}^{w}\hat{\boldsymbol{p}}_k$.

Dans la suite de ce chapitre, la fusion des données est réalisée par un filtrage de Kalman à contraintes multiples [MR07b], de l'anglais *Multi-State Constraint Kalman Filter*(MSCKF). Le filtrage MSCKF est une variante d'un filtrage de Kalman étendu (EKF), tel que présenté précédemment, dans lequel les amers ne font pas partie de l'état. En revanche l'état est composé d'un ensemble de positions de caméra ${}^{w}\boldsymbol{p}_{C_k}$ et les observations ${}^{uv}\boldsymbol{m}$ des amers ${}^{w}\boldsymbol{M}$ jouent le rôle de contraintes géométriques sur les différentes positions de la caméra. La plateforme OpenVINS [Gen+20] met en œuvre ce filtre.

6.1.2 OpenVINS

Dans cette section, une présentation brève de la plateforme est réalisée pour introduire leur représentation de l'erreur et en particulier leur modèle de covariance sur la position des points d'intérêt. Par la suite, l'intégration de nos covariances estimées par DUNE est décrite.

6.1.2.1 Présentation de la plateforme

OpenVINS est une plateforme opensource offrant un code fonctionnel de base de vision par ordinateur ainsi qu'un estimateur de localisation basé sur un filtrage serré visuel-inertiel. Le filtre principal mis en place par la plateforme est un filtre de Kalman étendu (EKF) qui fusionne les informations inertielles avec les pistes d'évolution de la position des points d'intérêt. Cette fusion est réalisée par un filtre de Kalman à contraintes multiples (MSCKF) [MR07b] appliqué sur une fenêtre glissante. L'utilisation de cette fenêtre glissante permet de mettre à jour l'état du filtre sans nécessairement devoir estimer les états des points d'intérêt. Par ailleurs, le filtre proposé inclut une bibliothèque permettant de moduler la gestion du système et en particulier d'accéder et modifier la covariance de l'état du filtre. Enfin, la plateforme fourni un *bridge* pour travailler directement à partir des datasets de vision les plus utilisés par la communauté (KITTI [GLU12], Euroc [Bur+16], etc ..). Pour davantage de détails, le lecteur peut se référer à [Gen+20] [4].

6.1.2.2 Caractérisation du modèle d'incertitude de OpenVINS

L'erreur peut être observée à plusieurs échelles : localement [Zhu+19a][WM17][Urf+06][Sze90], par l'erreur d'observation qui définit l'écart en pixel de la position d'un point d'intérêt estimé avec sa position réelle, ou globalement [ZMG22][Wan+20][ZT17] avec l'erreur de localisation qui mesure la dérive sur la localisation du système mobile. L'erreur de localisation est immédiate : elle est observable par la superposition de la trajectoire du système mobile avec une carte connue de l'environnement par exemple, ou plus simplement grâce à des informations de géolocalisation (*i.e.* GPS) ou encore l'observation répétée d'un point de vue. Toutefois, la caractérisation de l'erreur de

4. https ://docs.openvins.com/index.html

localisation d'un système mobile n'apporte pas suffisamment de caractérisation des mesures pour effectuer un couplage serré et contraint à un couplage lâche. En effet, l'incertitude sur les mesures n'est pas évaluée. Elle est plus difficile à évaluer que l'incertitude sur l'erreur de localisation. Une solution populaire pour effectuer une hybridation serrée est de considérer une erreur de reprojection fixe sur l'ensemble des mesures visuelles [MAMT15][KK01]. Cette hypothèse est limitante et considère une erreur uniformément répartie sur l'ensemble des images capturées, sans prendre en compte les déformations optiques ou autres effets optiques. L'erreur est également considérée constante dans le temps. Or, l'estimation de la position d'intérêt d'un point à $t+1$ étant issue de l'estimation de la position du point à t, l'erreur mesurée sur la position du point se cumule et grandit. Ainsi, si cette approximation est acceptable dans un environnement statique ou quasi-statique, elle devient limitante dans un environnement dynamique. Des solutions pour évaluer une incertitude adaptée aux mesures ont été proposées chapitre 5.

OpenVINS [Gen+20] propose un filtre MSCKF [MR07b] à couplage serré et suppose que les observations des points d'intérêt d'une image à l'autre sont indépendantes et définissent ainsi chaque matrice de covariance $\boldsymbol{C}$ d'une observation comme le produit d'un vecteur avec la matrice identité :

$$\boldsymbol{C} = \sigma \cdot \mathcal{I}_{2\times 2} \tag{6.14}$$

avec σ constant ce qui rend le bruit isotrope et identique pour tous les points d'intérêt. Cette simplification est pratique d'un point de vue calculatoire mais pas très réaliste. En particulier, elle permet de s'affranchir de la caractérisation de l'incertitude des points d'intérêt dans l'image. Afin d'expliquer les différentes simplifications réalisées et leurs impacts, certaines équations de filtre MSCKF décrites dans l'article [MR07b] sont reprises. Pour davantage de détails, le lecteur peut se référer à [MR07b] et [Gen+20].

[MR07b] est un filtre aux erreurs, c'est-à-dire qu'il estime l'erreur de son état plutôt que l'état lui même. Cette approche s'intéresse à la dynamique de l'erreur et permet notamment de diminuer la fréquence d'un filtre. En effet, la dynamique d'évolution de l'erreur est plus lente que la dynamique du système dynamique. Par ailleurs, la considération des erreurs permet de garantir les conditions pour effectuer la linéarisarion des fonctions d'évolution et d'observation avec une approximation raisonnable, ce qui remplit une condition pour un filtrage de Kalman optimal.

Notons $\boldsymbol{r}_i$ le vecteur résidu de cette linéarisation pour chaque observation i tel que pour un amer j donné, on a :

$$\boldsymbol{r}_i^{(j)} = {}^{uv}\boldsymbol{m}_i^{(j)} - {}^{uv}\hat{\boldsymbol{m}}_i^{(j)} \tag{6.15}$$

Nota Bene Afin d'uniformiser nos notations avec les notations de [MR07b], dans la suite de la section, l'exposant j fera référence à un amer ${}^{w}\boldsymbol{M}_j$ et i l'indice correspondant à la $i^{ème}$ observation ${}^{uv}\boldsymbol{m}_i^{(j)}$ de l'amer j.

${}^{uv}\hat{\boldsymbol{m}}_i^{(j)}$ est obtenu par la projection de l'amer ${}^{w}\boldsymbol{M}_j$ sur la caméra dont la position a été estimée par le filtre ${}^{w}\hat{\boldsymbol{p}}_C^{(i)}$ (6.12), tandis que ${}^{uv}\boldsymbol{m}_i^{(j)}$ est mesuré par le tracker de Kanade Lucas Tomasi (KLT) [BK81] dont OpenVINS propose une implémentation. Par linéarisation, le résidu $\boldsymbol{r}^{(j)}$ relatif à un amer j peut alors s'écrire avec l'équation (24) de [MR07b] que l'on peut réécrire :

$$\boldsymbol{r}^{(j)} \simeq \boldsymbol{H}_{\mathbf{x}}^{(j)}\tilde{\mathbf{x}} + \boldsymbol{H}_f^{(j)} \cdot {}^{w}\tilde{\boldsymbol{M}}_j + \boldsymbol{v}^{(j)} \tag{6.16}$$

avec $\boldsymbol{v}^{(j)} \sim \mathcal{N}(0, \boldsymbol{R}^{(j)})$ le bruit du modèle d'observation supposé gaussien avec $\boldsymbol{R}^{(j)}$ sa matrice de covariance. $\boldsymbol{H}_{\mathbf{x}}^{(j)}$ et $\boldsymbol{H}_f^{(j)}$ sont les matrices jacobiennes respectivement du modèle d'évolution de l'état $\mathbf{x}$ et du modèle d'évolution des observation. $\tilde{\mathbf{x}}$ et ${}^{w}\tilde{\boldsymbol{M}}$ représentent les erreurs des états $\mathbf{x}$ et ${}^{w}\boldsymbol{M}$. Puisque les observations des points d'intérêt sont supposées indépendantes d'une image à l'autre, en reprenant (6.14) on peut décrire le résidu lié à un amer j comme le produit du scalaire σ caractérisant l'intertitude de toutes les observations i avec une matrice identité :

$$\boldsymbol{R}^{(j)} = \sigma \cdot \mathcal{I}_{2i\times 2i} \tag{6.17}$$

> ***Nota Bene*** On remarque qu'en posant $\sigma = 1$, on se ramène à une matrice identité

Dans leur définition du modèle de mesure utilisé par [MR07b], le bruit du résidu doit être indépendant de l'erreur d'état $\tilde{\mathbf{x}}$. Or les amers ${}^{w}\tilde{\boldsymbol{M}}$ sont estimés à partir de l'état $\mathbf{x}$. En considérant $\boldsymbol{A}$ le noyau de $\boldsymbol{H}_f^{(j)}$. Par définition, $\boldsymbol{A}$ est construit telle que $\boldsymbol{A}^T\boldsymbol{H}_f^{(j)} = \mathbf{0}$. On peut réécrire le résidu tel que :

$$\begin{aligned} \boldsymbol{r}_{noyau}^{(j)} &= \boldsymbol{A}^T\left(\boldsymbol{z}^{(j)} - \hat{\boldsymbol{z}}^{(j)}\right) \simeq \boldsymbol{A}^T\boldsymbol{H}_{\boldsymbol{X}}^{(j)}\tilde{\boldsymbol{X}} + \boldsymbol{A}^T\boldsymbol{v}^{(j)} \\ &= \boldsymbol{H}_{noyau}^{(j)}\tilde{\boldsymbol{X}}^{(j)} + \boldsymbol{v}_{noyau}^{(j)} \end{aligned} \tag{6.18}$$

où $\boldsymbol{H}_{noyau}^{(j)} = \boldsymbol{A}^T\boldsymbol{H}_{\boldsymbol{X}}^{(j)}$ et $\boldsymbol{v}_{noyau}^{(j)} = \boldsymbol{A}^T\boldsymbol{v}^{(j)}$. Il convient de mentionner que pour calculer le résidu $\boldsymbol{r}_{noyau}^{(j)}$ (6.18), la matrice A n'a pas besoin d'être explicitement déterminée. Des opérations de permutations, en particulier la factorisation QR décrite dans [GVL13] (page 252, Algorithme 5.2.4), permettent d'exprimer $\boldsymbol{r}_{noyau}^{(j)}$. On note $E\{\boldsymbol{v}_{noyau}^{(j)}\boldsymbol{v}_{noyau}^{(j)T}\}$ la matrice de covariance liée à $\boldsymbol{v}_{noyau}^{(j)}$. Elle définit la matrice de covariance de l'état utilisée pour mettre à jour l'EKF [5].

$$E\{\boldsymbol{v}_{noyau}^{(j)}\boldsymbol{v}_{noyau}^{(j)T}\} = \boldsymbol{A}^T\boldsymbol{R}^{(j)}\boldsymbol{A} \tag{6.19}$$

Comme les observations sont supposées indépendantes, $\boldsymbol{R}^{(j)}$ est approximée par le produit de la matrice identité et de l'incertitude σ fixe (6.17), on a alors :

5. *Extended Kalman Filter*

$$\begin{aligned} E\{\boldsymbol{v}_{noyau}^{(j)}\boldsymbol{v}_{noyau}^{(j)T}\} &= \sigma \cdot \mathcal{I} \cdot \boldsymbol{A}^T\boldsymbol{A} \\ &= \sigma \cdot \boldsymbol{A}^T\boldsymbol{A} \end{aligned} \tag{6.20}$$

De plus $\boldsymbol{A}$ est construit tel que $\boldsymbol{A}^T\boldsymbol{A}$ soit l'identitée, donc :

$$\begin{aligned} E\{\boldsymbol{v}_{noyau}^{(j)}\boldsymbol{v}_{noyau}^{(j)T}\} &= \sigma \cdot \mathcal{I} \\ &= \boldsymbol{R}^{(j)} \end{aligned} \tag{6.21}$$

$E\{\boldsymbol{v}_{noyau}^{(j)}\boldsymbol{v}_{noyau}^{(j)T}\}$ définit la matrice de covariance d'une observation j. Pour remonter à l'état, l'ensemble des amers j est évalué :

$$\begin{aligned} \boldsymbol{R} &= \mathcal{I} \cdot \begin{bmatrix} E\{\boldsymbol{v}_{noyau}^{(0)}\boldsymbol{v}_{noyau}^{(0)T}\} & \cdots & E\{\boldsymbol{v}_{noyau}^{(j)}\boldsymbol{v}_{noyau}^{(j)T}\} \end{bmatrix}^T \\ &= \begin{pmatrix} \boldsymbol{A}^T\boldsymbol{R}^{(0)}\boldsymbol{A} & & \\ & \ddots & \\ & & \boldsymbol{A}^T\boldsymbol{R}^{(j)}\boldsymbol{A} \end{pmatrix} \\ &= \mathcal{I} \cdot \begin{bmatrix} \sigma^{(0)} & \cdots & \sigma^{(j)} \end{bmatrix}^T \\ &= \sigma \cdot \mathcal{I} \end{aligned} \tag{6.22}$$

Or σ est identique pour tous les amers j, donc $\sigma^{(0)} = \sigma^{(j)} = \sigma$. Puis comme précédemment (6.18), $\boldsymbol{A}$ est construit comme le noyau de $\boldsymbol{H}_f$ telle que $\boldsymbol{A}^T\boldsymbol{H}_f = \boldsymbol{0}$ (6.19). Alors la covariance $E\{\boldsymbol{v}_{noyau}\boldsymbol{v}_{noyau}^T\}$ relatif à l'état s'écrit finalement :

$$\begin{aligned} E\{\boldsymbol{v}_{noyau}\boldsymbol{v}_{noyau}^T\} &= \boldsymbol{A}^T\boldsymbol{R}\boldsymbol{A} \\ &= \sigma \cdot \mathcal{I} \end{aligned} \tag{6.23}$$

avec σ constant pour tous les observations i points d'intérêt des amers j. $E\{\boldsymbol{v}_{noyau}\boldsymbol{v}_{noyau}^T\}$ représente la covariance finale de l'état du filtre.

L'hypothèse d'indépendance des observations permet de se ramener à une modélisation de la covariance de l'état du filtre comme étant le produit d'un scalaire avec une matrice identité. Cette représentation simplifie les calculs tout comme l'intégration des variances de chaque observation. Toutefois elle est limitante à une évolution dynamique statique ou quasi-statique. De plus, l'estimation du mouvement par le filtre MSCKF proposée dans [MR07b] est très sensible au bruit visuel et, en particulier lorsque les amers ${}^w\boldsymbol{M}$ observés sont très éloignés, pouvant entraîner des solutions incohérentes.

OpenVINS se place dans le cas où $\forall i, \forall j, \sigma = 1$. Ainsi la covariance de l'état est directement décrite par une matrice identité. Dans cette définition de l'incertitude, tous les points d'intérêt

ont la même précision au sein d'une même image et également au cours du temps. La précision ne dépend ni de l'image, ni de l'environnement ou encore du mouvement. Néanmoins, la qualité d'une mesure est inhérente à la visibilité, la texture, ou encore à la luminosité, donc directement liée à l'environnement observé et au mouvement. En effet, les observations sont réalisées par le tracker KLT et sont soumises aux perturbations optiques, aux bruits de quantification numérique, à la dynamique du mouvement (frequence échantillonnage trop petite vs mouvement rapide) et à l'environnement capturé (texture, luminosité ..). Il convient donc de redéfinir des matrices de covariances adaptées pour chaque observation d'amer et de les intégrer dans le filtre MSCKF proposé par OpenVINS.

6.1.2.3 Intégration des incertitudes issues de DUNE dans OpenVINS

Nous avons montré (chapitre 5) que l'estimateur DUNE produit un modèle d'incertitude adapté à chaque observation. Notre modélisation de l'incertitude sur la position d'un point d'intérêt entre deux instant t et $t+1$ prend en compte les informations et bruits inhérents à l'image. Cette représentation de l'incertitude est intégrée dans le filtre MSCKF de OpenVINS.

Nous avons défini précédemment (chapitre 5) la matrice de covariance d'une observation ${}^{uv}\boldsymbol{m}_i^{(j)}$ telle que

$$\boldsymbol{R}_i^{(j)} = \begin{pmatrix} \sigma_{uu} & \sigma_{uv} \\ \sigma_{uv} & \sigma_{vv} \end{pmatrix}_i^{(j)} \tag{6.24}$$

Ainsi, en reprenant les notations précédentes, on peut réécrire $\boldsymbol{R}^{(j)}$ le résidu relatif à un amer j tel que (6.17) devient :

$$\boldsymbol{R}^{(j)} = \begin{pmatrix} \begin{pmatrix} \sigma_{uu} & \sigma_{uv} \\ \sigma_{uv} & \sigma_{vv} \end{pmatrix}_0^{(j)} & & \\ & \ddots & \\ & & \begin{pmatrix} \sigma_{uu} & \sigma_{uv} \\ \sigma_{uv} & \sigma_{vv} \end{pmatrix}_i^{(j)} \end{pmatrix} \tag{6.25}$$

(6.19) est inchangé et non simplifiable. La matrice de covariance $\boldsymbol{R}$ de l'ensemble des amers j s'écrit alors :

$$\boldsymbol{R} = \begin{pmatrix} \boldsymbol{A}^T\boldsymbol{R}^{(0)}\boldsymbol{A} & & \\ & \ddots & \\ & & \boldsymbol{A}^T\boldsymbol{R}^{(j)}\boldsymbol{A} \end{pmatrix} \tag{6.26}$$

Finalement, la covariance de l'état du filtre dont les observations de points d'intérêt ne sont pas considérés indépendants les uns des autres s'écrit :

$$E\{\boldsymbol{v}_{noyau}\boldsymbol{v}_{noyau}^T\} = \boldsymbol{A}^T\boldsymbol{R}\boldsymbol{A} \tag{6.27}$$

6.1.2.4 Représentation de l'incertitude sur la mesure

DUNE fournit une estimation de la covariance $\mathbf{\Sigma}_i^{(j)}$ sur la position du point mesuré $\mathbf{\Sigma}_i^{(j)}$ par le tracker KLT telle que ${}^{uv}\boldsymbol{e}_i^{(j)} = {}^{uv}\boldsymbol{m}_i^{(j)} - {}^{uv}\tilde{\boldsymbol{m}}_i^{(j)}$ et ${}^{uv}\boldsymbol{e}_i^{(j)} \sim \mathcal{N}(0, \mathbf{\Sigma}_i^{(j)})$.

En reprenant (2.24, cf chap 2), on peut définir le modèle de mesure tel que :

$$\begin{cases} {}^{uv}\boldsymbol{m}_0^{(j)} = {}^{uv}\tilde{\boldsymbol{m}}_0^{(j)} + \boldsymbol{v}_0^{(j)} & \text{si } i = 0 \\ {}^{uv}\boldsymbol{m}_{i+1}^{(j)} = f_{KLT}({}^{uv}\tilde{\boldsymbol{m}}_i^{(j)}, \boldsymbol{I}_i, \boldsymbol{I}_{i+1}) + \boldsymbol{v}_{i+1}^{(j)} & \text{sinon} \end{cases} \tag{6.28}$$

Ainsi, le suivi d'un point par le tracker KLT est caractérisé par une erreur que l'on suppose gaussienne. D'après l'équation (6.15), on a alors :

$$\begin{aligned} \mathbf{r}_i^{(j)} &= \mathbf{z}_i^{(j)} - \hat{\mathbf{z}}_i^{(j)} && (6.29)\\ &= {}^{uv}\boldsymbol{m}_i^{(j)} - {}^{uv}\hat{\boldsymbol{m}}_i^{(j)} && (6.30)\\ &= \left({}^{uv}\tilde{\boldsymbol{m}}_i^{(j)} + \boldsymbol{v}_i^{(j)}\right) - {}^{uv}\hat{\boldsymbol{m}}_i^{(j)} && (6.31)\\ &= \left(f_{KLT}({}^{uv}\tilde{\boldsymbol{m}}_{i-1}^{(j)}, \boldsymbol{I}_{i-1}, \boldsymbol{I}_i) + \boldsymbol{v}_i^{(j)}\right) - {}^{uv}\pi\left({}^{w}\boldsymbol{M}_j - {}^{w}\hat{\boldsymbol{p}}_{C_i}, {}_{w}^{C_i}\boldsymbol{T}\right) && (6.32)\\ &= f_{KLT}({}^{uv}\tilde{\boldsymbol{m}}_{i-1}^{(j)}, \boldsymbol{I}_{i-1}, \boldsymbol{I}_i) - {}^{uv}\pi\left({}^{w}\boldsymbol{M}_j - {}^{w}\hat{\boldsymbol{p}}_{C_i}, {}_{w}^{C_i}\boldsymbol{T}\right) + \boldsymbol{v}_i^{(j)} && (6.33) \end{aligned}$$

avec $\boldsymbol{v}_i^{(j)} \sim \mathcal{N}(0, \boldsymbol{R}_i^{(j)})$ et $\boldsymbol{R}_i^{(j)} \in \mathbb{R}^{2\times 2}$

Nota Bene DUNE estime l'incertitude sur la position d'un point d'intérêt entre deux instants k et $k+1$ (identifiée ici par les observations i et $i+1$). La matrice de covariance sortante $\mathbf{\Sigma}_i^{(j)} = \boldsymbol{R}_{i|i-1}^{(j)}$.

$$\boldsymbol{R}_i^{(j)} = \boldsymbol{R}_0^{(j)} + \sum_{obs=1}^{i} \boldsymbol{R}_{obs|obs-1}^{(j)} \tag{6.34}$$

En reprenant les équations (6.17) à (6.27) du § 6.1.2.3, on obtient la matrice de covariance $E\{\boldsymbol{v}_0\boldsymbol{v}_0^T\}$ (6.27) qui caractérise la précision sur la position des points d'intérêt observés considérés dans l'état.

Une représentation de l'évolution des covariances issues de DUNE et de OpenVINS est proposée Fig. 6.1 dans laquelle un point d'intérêt est observé à trois instants $k-1$, k et $k+1$, capturant des perturbations numériques à l'instant k. L'impact de ces perturbations est évalué, notamment l'adaptabilité des matrices de covariance issue de DUNE et OpenVINS. À chaque itération, un zoom est réalisé sur le point suivi, représenté par $+$.

La première observation $i = 0$ de l'amer (*i.e.* ici le pied inférieur gauche de l'arbre) j est réalisée à l'instant $k-1$. La covariance initiale est donnée par 0.5 pixel [KK01], représentée par le cercle bleu à l'instant $k-1$. Cette covariance initiale correspond à l'incertitude moyenne donnée à la détection de Harris [KK01][MAMT15]. Lors de la première itération, la covariance

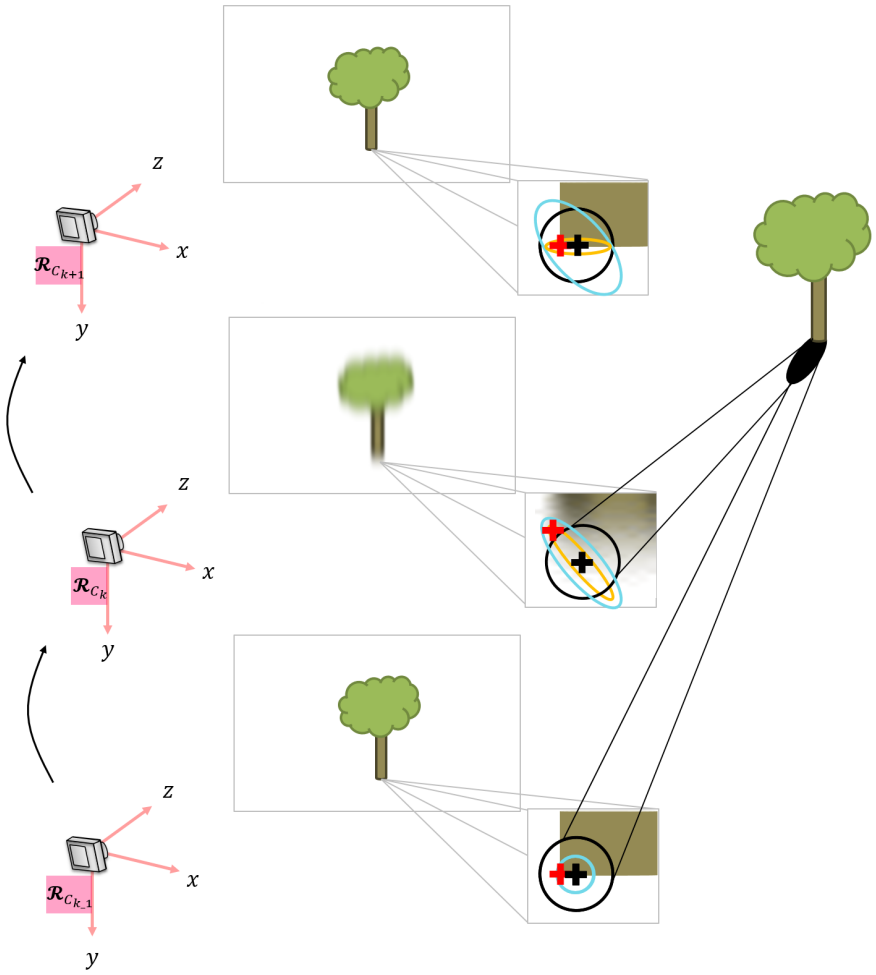


FIGURE 6.1 – Évolution de la covariance proposé par OpenVINS et DUNE entre $k-1$ et $k+1$ - +représente la vérité terrain du point d'intérêt et + le point détecté par KLT ; **O** la covariance associée à OpenVINS fixée à 1 pixel ; O la covariance sortante de DUNE $\boldsymbol{\Sigma}_i^{(j)} = \boldsymbol{R}_{i|i-1}^{(j)}$ et O la covariance cumulée de DUNE $\boldsymbol{R}_i^{(j)}$

$\boldsymbol{\Sigma}_{i=0}^{(j)} = \boldsymbol{R}_{i=0}^{(j)} = \mathcal{I}_{2\times 2} \cdot \begin{pmatrix} \sqrt{0.5} & \sqrt{0.5} \end{pmatrix}^T$: les cercles orange et bleu sont superposés (Fig. 6.1). On observe que la covariance fixe, représentée en noir, issue de OpenVINS est surdimensionnée par rapport à la nature de l'erreur.

À l'instant k, l'image capturée est sujette à du flou directionnel produit par un mouvement brusque vers le haut. La position du point mesuré par KLT est impactée par une forte imprécision. En conséquence, DUNE devrait réaliser une première estimation de l'incertitude $\boldsymbol{\Sigma}_1^{(j)} = \boldsymbol{R}_{1|0}^{(j)}$ représenté par l'ellipse orange (Fig. 6.1). On note que la vérité terrain serait comprise dans l'ellipse orange et en dehors du cercle noir représentant la covariance fixe de OpenVINS : l'estimation de DUNE engloberait bien l'impact du bruit visuel, contrairement à OpenVINS. L'ellipse bleue représente $\boldsymbol{R}_1^{(j)} = \boldsymbol{R}_0^{(j)} + \boldsymbol{R}_{1|0}^{(j)}$.

À l'instant $k+1$, l'image capturée est de nouveau nette. La précision sur la position du point d'intérêt suivi par KLT est ainsi améliorée. La covariance résultante de DUNE $\boldsymbol{\Sigma}_2^{(j)} = \boldsymbol{R}_{2|1}^{(j)}$ devrait être plus petite, et donc l'incertitude sur la position du point serait plus faible, avec une amplitude marquée le long de la ligne, là où le gradient est élevé. L'ellipse bleue illustre $\boldsymbol{R}_2^{(j)} = \boldsymbol{R}_0^{(j)} + \boldsymbol{R}_{1|0}^{(j)} + \boldsymbol{R}_{2|1}^{(j)}$. La covariance fixe proposée par OpenVINS serait de nouveau surdimensionnée pour cette situation.

Tandis que la représentation d'un covariance constante suppose une indépendance entre les observations temporelles et au sein d'une même image, elle définit une variance englobant le maximum de points d'intérêt en considérant la valeur haute de l'estimation de l'incertitude. Elle présente également de nombreux avantages de simplifications des calculs, mais est contrainte à une évolution quasi-statique. Par ailleurs, la représentation de l'incertitude telle que définie par OpenVINS ne prend pas en compte les effets optiques (*i.e.* flou directionnel, faible contraste, forte illumination etc) dans l'estimation de l'incertitude sur la position d'un point.

Notre définition de l'erreur prend désormais en compte les corrélations entre les observations et la nature de l'image, les perturbations numériques [Sze90] ainsi que l'environnement. Par ailleurs, l'incertitude s'ajuste au niveau de bruit et à l'environnement capturé. L'incertitude grandit avec l'évolution de la position du point suivi au cours du temps qui cumule de l'erreur à chaque itération. L'estimateur DUNE fournit la capacité d'avoir des matrices de covariance très petites et précises, donnant de l'importance à l'impact de ces observations lors de l'intégration dans le filtre MSCKF. À l'inverse, les effets optiques sont également caractérisés, limitant l'impact des observations bruitées. Enfin, la définition de l'erreur présentée (6.28) capture également le biais du tracking par KLT.

6.2 Résultats

Dans cette section, les résultats de l'application du filtre MSCKF incorporant les covariances issues de DUNE sont évalués et comparés au fonctionnement optimal du filtre MSCKF proposé par OpenVINS. Toutes les évaluations sont réalisées à partir du dataset Euroc [Bur+16]. Un modèle de DUNE est appris à partir de ce dataset, dont les métriques sont présentés § 6.2.1. Par la suite, de nouvelles métriques sont proposées afin d'évaluer l'impact des incertitudes sur la trajectoire du système mobile. Enfin, une comparaison sur plusieurs critères est réalisée.

6.2.1 Apprentissage sur Euroc

Pour réaliser l'ensemble des évaluations, un nouveau modèle de DUNE est généré sur le dataset Euroc [Bur+16]. De même que pour la construction de la base de donnée d'apprentissage de Kitti (§ 5.3.3), un tracking est effectué en rétropropagation sur un aller retour. Le scénario utilisé pour construire le jeu de données d'apprentissage est $V2\_03\_difficult$. Les hyperparamètres sont identiques à ceux décrits § 5.3.3. Comme précédemment 70% des données ont été utilisées pour ajuster les poids du modèle au cours de l'apprentissage, 15% pour tester le modèle au cours de l'apprentissage et 15% pour évaluer le modèle final à partir des métriques présentées § 5.3.1. Le résultat de cette évaluation est présenté Tab. 6.1.

Métriques	DUNE + (λ_1, λ_2)	[WM17]	Fix-C	Valeures idéales
$3\sigma_u$	93.36	90.23	99.3	99.7
$3\sigma_v$	94.08	86.3	99.8	99.7
$2\sigma_u$	88.50	85.01	97.36	95
$2\sigma_v$	88.32	88.62	98.2	95
$1\sigma_u$	72.85	84.53	95.43	68
$1\sigma_v$	72.01	81.21	98.42	68
MD	**1.17**	1.36	0.3	1
NNE	**1.07**	1.31	0.3	1

TABLE 6.1 – Comparaison des résultats de DUNE par rapport à l'état de l'art

Le modèle, appris sur Euroc, est comparé à [WM17] et Fix-C, avec une variance fixe à $0.5 \cdot 2$ pixels. Comme dans le chapitre 5, DUNE propose des matrices de covariance plus adaptées que [WM17] et Fix-C. En effet, avec $NNE = 0.3$, Fix-C est très pessimiste. cette définition de la covariance fixe englobe quasiment l'intégralité des points d'intérêt dès 1σ. Elle est donc sur-dimensionnée. [WM17] est une solution plus adaptée. Néanmoins, les performances de cette méthode restent en dessous de DUNE, avec $MD = 1.17$ et $NNE \simeq 1$.

6.2.2 Évaluation de DUNE sur OpenVINS

Dans la suite, tous les tests et évaluations sont réalisés sur le scénario $V1\_03\_difficult$ du dataset Euroc.

6.2.2.1 Affichage des matrices de covariance

Les ellipses de covariance à 3σ prédites par DUNE sont présentés Fig. 6.2 en rouge à partir du scénario $V1\_03\_difficult$. Les covariances sont augmentées d'un facteur 10 à des fins de visualisation.

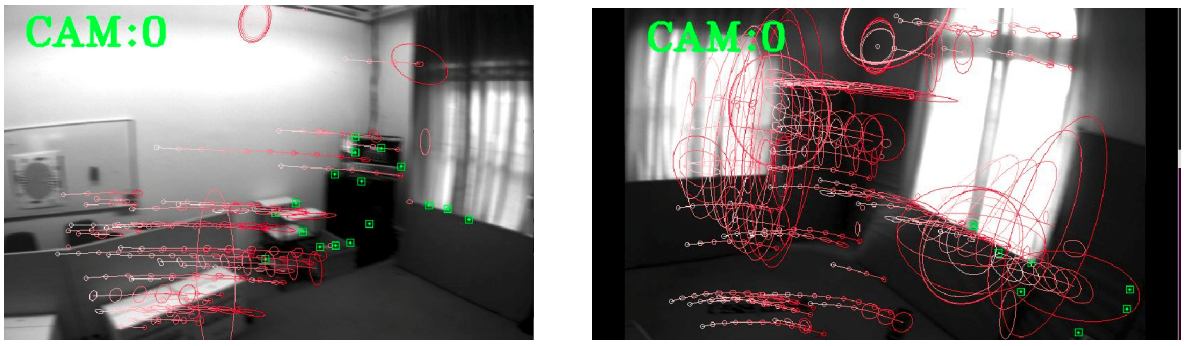


FIGURE 6.2 – Affichage des pistes d'évolution des matrices de covariance issues de DUNE appliquées à un système monoculaire sur deux points de vue d'un scénario Euroc [Bur+16] - Les ellipses sont augmentées d'un facteur 10 à des fins de visualisation

Deux instants sont présentés Fig. 6.2. On observe dans l'image de gauche du flou optique avec un mouvement de rotation rapide, et à droite un flou optique doublé d'une sur-exposition

lumineuse. En conséquence, les covariances sont plus grandes, en particulier sur les bords des fenêtres. Il est intéressant de noter l'évolution des covariances qui grandissent, en particulier quand l'erreur est importante. L'orientation de celle-ci suit bien les lignes et les contours. L'image de droite met en avant ce constat notamment sur le coin bas de droite de sa fenêtre principale. Enfin les carrés vert sont des points d'intérêt tout juste détectés et non suivis. À titre comparatif, une covariance fixe telle que définie § 6.1.2.2 est représentée par un cercle de même dimension pour tous les points et constant dans le temps.

6.2.2.2 Métriques d'évaluation

Afin d'évaluer le bénéfice de l'utilisation de covariances adaptées, nous introduisons de nouvelles métriques dédiées à l'évaluation de la pose du système mobile. Pour cette évaluation, nous bénéficions de la pose réelle du système mobile. Il s'agit donc de comparer la pose estimée par le filtrage MSCKF avec la pose réelle, fournie par le dataset Euroc. Ces métriques sont proposées et intégrées dans OpenVins pour évaluer les performances des essais.

Absolute Trajectory Error (ATE) Tout comme [ZS18], nous appliquons l'erreur absolue sur la trajectoire globale, en anglais *Absolute Trajectory Error*, notée ATE. ATE calcule la différence entre le pose estimée et la pose réelle à chaque instant k de la trajectoire puis en fait la moyenne. Le calcul de l'erreur sur la position (en m) est évalué distinctement de l'erreur sur l'orientation (en deg) du système.

$$ATE(m) = \sqrt{\frac{1}{K}\sum_{k=1}^{K} ||\boldsymbol{p}_k - \hat{\boldsymbol{p}}_k||^2} \tag{6.35}$$

$$ATE(deg) = \sqrt{\frac{1}{K}\sum_{k=1}^{K} ||{}_k\boldsymbol{R} \cdot {}_k\hat{\boldsymbol{R}}^{-1}||^2} \tag{6.36}$$

> ***Nota Bene*** Il est important de noter que les matrices de rotation sont exprimées en degrés à partir les trois angles de rotation ϕ, θ et ψ indiquant le roulis, le tangage et le lacet. Toutefois, l'orientation est exprimée en quaternion dans le vecteur d'état du filtre.

Posons $\boxminus$ l'opérateur indiquant la différence entre l'état réel et l'estimé, exprimant le produit des matrices de rotations ou la différence des positions tel que décrit ci-contre :

$$\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k = \begin{cases} \boldsymbol{p}_k - \hat{\boldsymbol{p}}_k \\ {}_k\boldsymbol{R} \cdot {}_k\hat{\boldsymbol{R}}^{-1} \end{cases} \tag{6.37}$$

Idéalement, l'écart entre la pose estimée et la vérité terrain est nul, donc la moyenne sur cet écart au court d'un scénario est également nulle : $ATE = 0$.

Relative Pose Error (RPE) L'erreur n'est pas uniformément répartie durant un scénario. Certains évenements dynamiques peuvent perturber l'estimateur. Pour observer localement l'erreur, nous utilisons RPE. Inspiré de ATE, l'objectif est de calculer l'ecart moyen entre la pose réelle et la pose estimée mais sur des segments $D = \{d_1, \cdots, d_i, \cdots, d_v\}$ de la trajectoire.

Posons $\tilde{\mathbf{x}}_r$ l'écart entre la pose initiale et finale du segment courant :

$$\tilde{\mathbf{x}}_r = \mathbf{x}_k \boxminus \mathbf{x}_{k+d_i} \tag{6.38}$$

Alors, RPE évalue l'erreur moyenne sur la position (en m) et sur l'orientation (en deg) sur ce segment :

$$RPE = \sqrt{\frac{1}{D}\sum_{k=1}^{D} ||\tilde{\mathbf{x}}_r \boxminus \hat{\tilde{\mathbf{x}}}_r||} \tag{6.39}$$

Tout comme ATE, idéalement, l'écart entre l'estimée et la vérité terrain est nulle, donc $RPE = 0$.

Normalized Estimation Error Squared (NEES) Enfin l'estimation de l'erreur normalisée (NEES) [Che+18] définit la distance entre l'erreur de la pose et la matrice de covariance $\boldsymbol{P}_k$ de l'état associée à la pose estimée par le filtre. L'erreur normalisée est calculée à chaque instant k puis moyennée sur l'ensemble de la trajectoire :

$$NEES = \frac{1}{K}\sum_{k=1}^{K} (\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k)^T \boldsymbol{P}_k (\mathbf{x}_k \boxminus \hat{\mathbf{x}}_k) \tag{6.40}$$

Comme précédemment, le calcul de l'erreur sur l'orientation (en deg) est effectué séparément du calcul de l'erreur sur la position. L'écart parfait est nul, donc $NEES = 0$.

6.2.2.3 Préparation des données

Une implémentation du tracker KLT [BK81] est proposée par OpenVINS [Gen+20]. Les observations sont donc initialisées avec un détecteur de Harris puis trackées avec un critère de répartition. Il s'agit d'une distance minimale définie par une fenêtre de taille 20×20 au voisinage du point suivi, et un rayon minimum de 15 pixels entre deux points d'intérêt. L'intégration des covariances de DUNE est réalisée en amont, en post-traitement. Une première passe est réalisée pour détecter les points d'intérêt et les suivre sans filtrage. La position des points est enregistrée dans un fichier annexe, associée à chaque image. Puis, l'estimateur DUNE, dont le modèle a été appris (§ 6.2.1), est appliqué et fournit les covariances pour chaque point entre deux instants k et $k+1$. Comme détaillé au paragraphe 6.1.2.4, les incertitudes sont cumulées, avec une variance initiale de 0.5 pixel [KK01], caractérisant la précision du détecteur de Harris. Les covariances estimées par DUNE sont ajoutées dans le fichier annexe. OpenVins a été adapté pour venir lire ce fichier, avec la position de l'observation et les covariances estimées, et les intégrer dans le filtrage MSCKF. Par ailleurs, le filtre a été modifié pour accueillir des covariances de formats différents d'une matrice diagonale.

Toutes les comparaisons réalisées ont été évaluées avec les paramètres optimaux de fonctionnement du système MSCKF mis en place par OpenVINS.

6.2.2.4 Influence du nombre d'observations optimal intégrées dans le filtre

Le premier résultat consiste à évaluer l'influence du nombre de points suivis. Intuitivement, le meilleur résultat est issu du plus grand nombre de points, tant que l'on dispose de points de qualité (*i.e.* précis, non bruités). Notre évaluation compare dans un premier temps les métriques ATE et NEES, puis s'intéresse à l'erreur relative RPE.

ATE/NEES Le tableau 6.2 est estimé avec l'application de matrice de covariance fixe de variance 1 pixel. En parallèle, le tableau 6.2 présente les résultats avec l'application des covariances cumulées estimées par DUNE. Pour chaque cas, les mêmes points sont observés pour réaliser la fusion des données.

Métriques / Méthode	**ATE (deg/m)**	**NEES (deg/m)**
$OpenVINS\_20\_points$	3.577 / 0.149	265.993 / 22.030
$OpenVINS\_50\_points$	**3.470** / 0.167	**251.512** / 35.539
$OpenVINS\_80\_points$	4.469 / 0.261	331.366 / 102.382
$OpenVINS\_100\_points$	4.349 / 0.290	355.436 / 99.281
$OpenVINS\_150\_points$	4.416 / 0.197	342.642 / 46.548
$OpenVINS\_200\_points$	3.646 / 0.186	331.220 / 50.789

TABLE 6.2 – Évaluation de l'influence du nombre de points dans OpenVINS

Métriques / Méthode	**ATE (deg/m)**	**NEES (deg/m)**
$DUNE\_20\_points$	**3.109** / **0.144**	**249.496** / **23.882**
$DUNE\_50\_points$	3.721 / **0.141**	283.362 / **32.897**
$DUNE\_80\_points$	**3.076** / **0.112**	**284.489** / **26.860**
$DUNE\_100\_points$	**3.156** / **0.103**	**302.179** / **25.682**
$DUNE\_150\_points$	**2.875** / **0.123**	**292.804** / **23.227**
$DUNE\_200\_points$	**3.013** / **0.187**	**289.888** / **48.861**

TABLE 6.3 – Évaluation de l'influence du nombre de points avec l'intégration des covariances estimées par DUNE

Les valeurs en gras indiquent le meilleur résultat entre les covariances fixe de OpenVINS et nos covariances estimées par DUNE. On observe que les covariances impactent la pose finale du système mobile. En particulier, l'apport d'information sur la précision des mesures permet d'améliorer la précision sur la pose du système. En effet, pour chaque cas, sauf à 50 points, l'erreur de trajectoire moyenne sur la position et l'orientation est plus faible avec une différence maximale de $\Delta NEES \simeq 50$deg et $\Delta NEES \simeq 75$m pour 100 points. On a en moyenne une amélioration $ATE \simeq 0.60$deg et $ATE \simeq 0.07$m. L'erreur cumulée observée avec

NEES est fortement impactée indiquant le bénéfice d'estimer des covariances adaptées sur la position des points d'intérêt.

RPE Comme précédemment, le tableau 6.4 est estimé avec l'application de matrice de covariance fixe de variance 1 pixel et le tableau 6.5 avec l'application des covariances cumulées estimées par DUNE. L'erreur de pose moyenne RPE est estimée sur 6 segments $D = \{8.0, 16.0, 24.0, 32.0, 40.0, 48.0\}$.

Points	**D=8m**	**D=16m**	**D=24m**	**D=32m**	**D=40m**	**D=48m**
20	1.303 / 0.142	1.928 / 0.170	2.395 / 0.209	2.856 / 0.202	3.527 / 0.206	4.239 / 0.243
50	1.240 / 0.154	1.682 / 0.183	1.869 / 0.202	1.779 / 0.227	1.952 / 0.242	2.391 / 0.263
80	1.313 / 0.184	1.690 / 0.240	1.592 / 0.270	1.576 / 0.299	1.645 / 0.322	1.670 / 0.327
100	1.249 / 0.202	1.613 / 0.236	1.576 / 0.266	1.539 / 0.288	1.402 / 0.304	1.655 / 0.324
150	1.231 / 0.168	1.612 / 0.221	1.520 / 0.268	1.488 / 0.286	1.499 / 0.276	1.822 / 0.267
200	1.278 / 0.156	1.583 / 0.210	1.603 / 0.239	1.529 / 0.241	1.389 / 0.240	1.520 / 0.242

TABLE 6.4 – Évaluation de $RPE(deg/m)$ sur l'influence du nombre de points dans OpenVINS

Points	**D=8m**	**D=16m**	**D=24m**	**D=32m**	**D=40m**	**D=48m**
20	1.287 / 0.137	1.726 / 0.150	1.931 / 0.197	2.189 / 0.206	2.619 / 0.212	3.091 / 0.226
50	1.307 / 0.134	1.828 / 0.171	2.058 / 0.199	1.970 / 0.219	2.180 / 0.221	2.489 / 0.228
80	1.089 / 0.124	1.470 / 0.152	1.551 / 0.172	1.605 / 0.180	1.845 / 0.177	1.938 / 0.167
100	1.006 / 0.115	1.254 / 0.138	1.338 / 0.151	1.448 / 0.168	1.756 / 0.179	2.060 / 0.180
150	1.094 / 0.130	1.415 / 0.154	1.353 / 0.167	1.328 / 0.172	1.386 / 0.174	1.667 / 0.162
200	1.133 / 0.153	1.302 / 0.187	1.268 / 0.209	1.200 / 0.215	1.162 / 0.213	1.486 / 0.215

TABLE 6.5 – Évaluation de $RPE(deg/m)$ sur l'influence du nombre de points avec l'intégration des covariances estimées par DUNE

Les résultats sont également représentés Fig. 6.3 et Fig. 6.4. Cette représentation permet de montrer la répartition de l'erreur pour chaque segment. L'encadré englobe deux quartiles $2Q$ des erreurs, avec la médiane représentée par le trait coupant l'encadré en deux. Les deux limites indiquent l'erreur maximale (par la tête $\top$) et l'erreur minimum (par le pied $\perp$).

Le nombre de points impacte peu l'erreur RPE sur les différents segments. On note toutefois que l'erreur sur l'orientation croît à chaque segment pour 20 points (Fig. 6.3). En moyenne, les résultats sont comparables avec 100, 150 et 200 points.

À l'inverse Fig. 6.4, on constate une amélioration dans l'amplitude de la répartition de l'erreur avec l'ajout de points. En particulier pour 80, 100 et 150 points. Par ailleurs, l'erreur moyenne sur la position est plus faible de 0.1m.

Comme attendu, plus on dispose de points, meilleurs sont les résultats. En contre partie, la répartition de l'erreur est plus importante, notamment pour 200 points (Fig. 6.4). Par ailleurs, le coût des calculs est plus important et difficile à envisager dans une application en temps réel. On observe qu'avec 100 points, les résultats obtenus peuvent parfois se montrer les meilleurs. Ce constat est notamment visible sur les premiers segments (Fig. 6.4 et Fig.

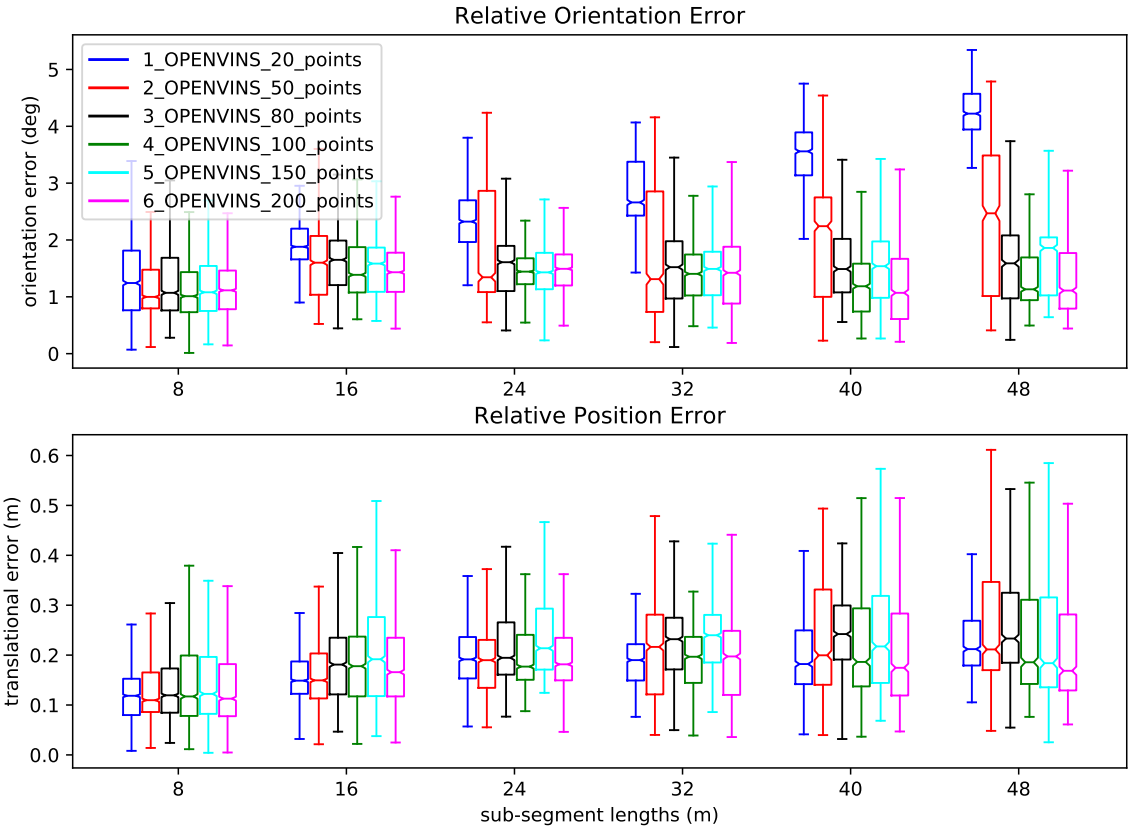


FIGURE 6.3 – Évaluation de $RPE(deg/m)$ sur l'influence du nombre de points dans OpenVINS

6.3), en particulier sur la position. Puis les résultats sur 150 points dépassent ceux de 100 points sur les derniers segments, avec une répartition relativement petite.

On note par ailleurs que OpenVINS est meilleur avec 50 points. Au delà de 50 points, des points de moins bonne qualité (*i.e.* moins précis) sont intégrés avec les mêmes poids que les points de bonne qualité et que ça dégrade donc l'estimation. Pour vérifier cette affirmation et l'impact d'observations bruitées, l'influence de la dynamique des scénarios est évaluée par la suite.

Dans la suite de notre étude, on fixe le nombre de points à 150.

6.2.2.5 Influence de la difficulté des scénarios

L'impact de la difficulté est ici évalué, en particulier, l'adaptabilité des covariances en réponse à la dynamique du système mobile. Le tableau 6.6 met en évidence la dégradation de l'estimation de la pose en fonction de la difficulté du scénario. On observe en particulier une erreur comparable sur le scénario *V1_01_easy*. Puis sur le scénario *V1_03_difficult*, l'erreur mesurée est doublée entre l'hybridation intégrant les covariances estimées par DUNE et les covariances fixe de OpenVINS.

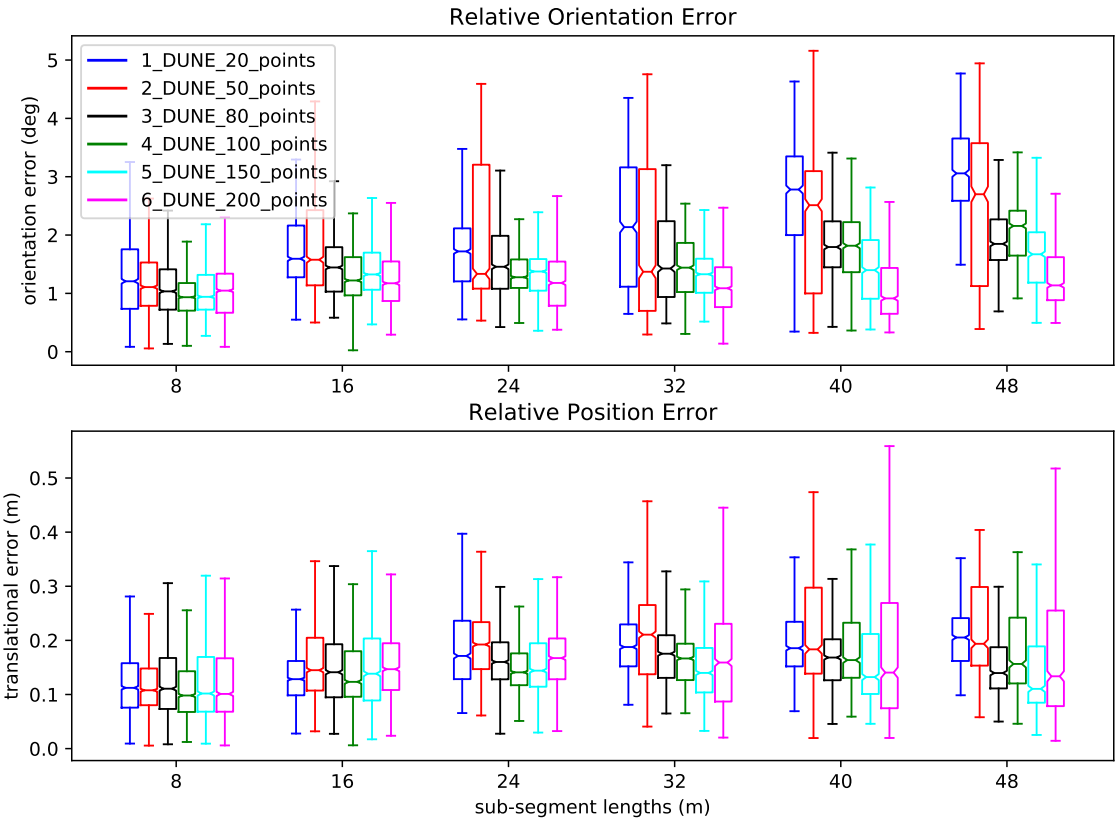


FIGURE 6.4 – Évaluation de $RPE(deg/m)$ sur l'influence du nombre de points avec l'intégration des covariances estimées par DUNE

ATE(deg/m) / Méthode	**V1_01_easy**	**V1_03_difficult**	**Average**
DUNE	**0.969 / 0.063**	**2.875 / 0.123**	**1.922 / 0.093**
OPENVINS	0.982 / 0.064	4.416 / 0.197	2.699 / 0.131

TABLE 6.6 – Évaluation de l'erreur moyenne $ATE(deg/m)$ sur deux niveaux de difficulté de scénario

La répartition de l'erreur RPE moyenne des deux scénarios sur les segments `D={8.0, 16.0, 24.0, 32.0, 40.0, 48.0}`. est présentée Fig. 6.5. L'erreur RPE sur l'orientation est comparable entre les deux approches. Toutefois, OpenVINS montre les maximums plus étalés. Ce constat est également observable avec l'erreur RPE sur la position. L'impact des covariances est particulièrement visible sur l'estimation de la position, avec une meilleure estimation moyenne sur chaque segment.

DUNE montre de bons résultats est semble surpasser OpenVINS. Ce phénomène est d'autant plus flagrant que le scénario est difficile et est sujet aux bruits et effets visuels (*i.e.* flou).

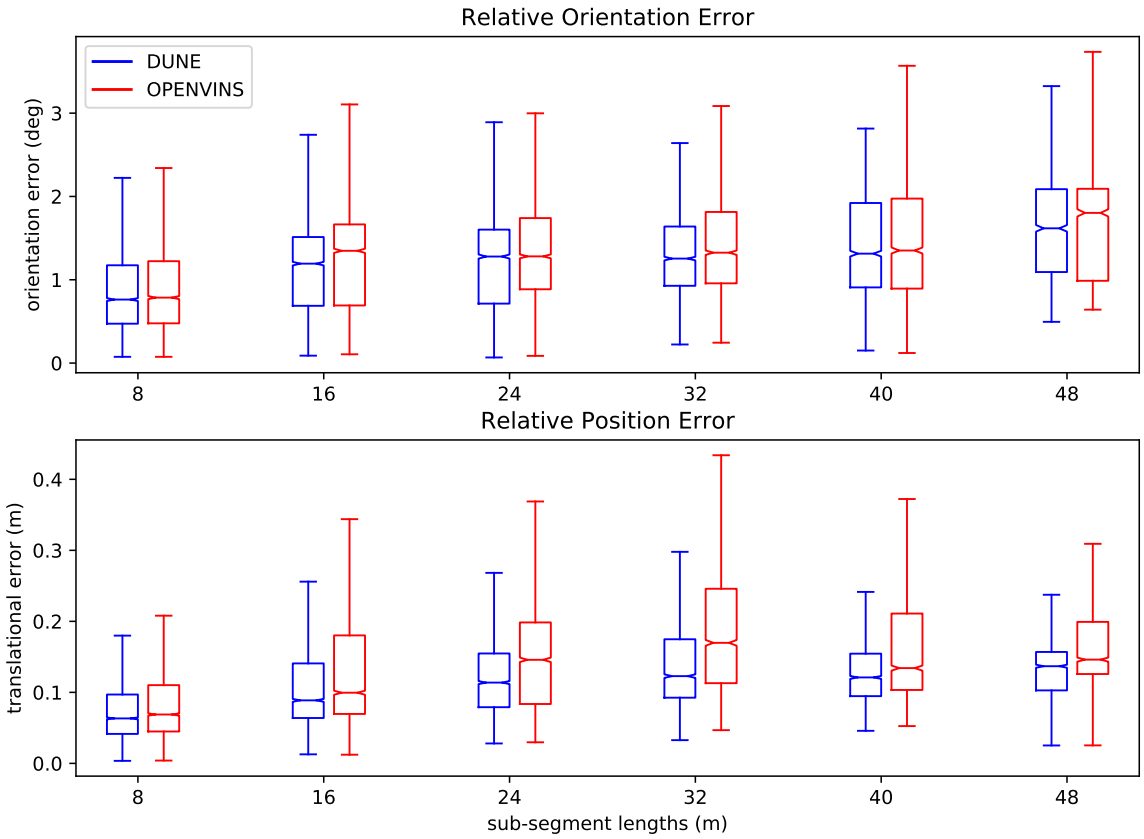


FIGURE 6.5 – Évolution de l'erreur moyenne RPE sur des segments de trajectoire.

6.2.2.6 Sélection des meilleurs points

Un des avantages majeur de qualifier la précision des mesures est de pouvoir pondérer les résultats, en particulier pouvoir sélectionner les meilleures observations à partir d'un grand nombre de détections dans le but d'améliorer l'estimation de la pose. Pour effectuer cette sélection, le critère du nombre de points détectés puis suivis par KLT dans OpenVINS est augmenté à 250. La précision de ces points est évaluée par DUNE, puis un seuil à 3 pixels est appliqué tel que

$$\sqrt{\sigma_{uu} + \sigma_{vv}} <= seuil \tag{6.41}$$

Métrique / Méthode	**ATE (deg/m)**	**NEES (deg/m)**
DUNE_best	**2.855** / **0.057**	353.112 / **9.209**
DUNE_150_points	2.875 / 0.123	**292.804** / 23.227
OpenVINS_150_points	4.416 / 0.197	342.642 / 46.548

TABLE 6.7 – Évaluation de la sélection des meilleurs points

Une fois qu'un point dépasse le seuil à k, toutes ces occurences à $k+1 \cdots k+i$ sont supprimées.

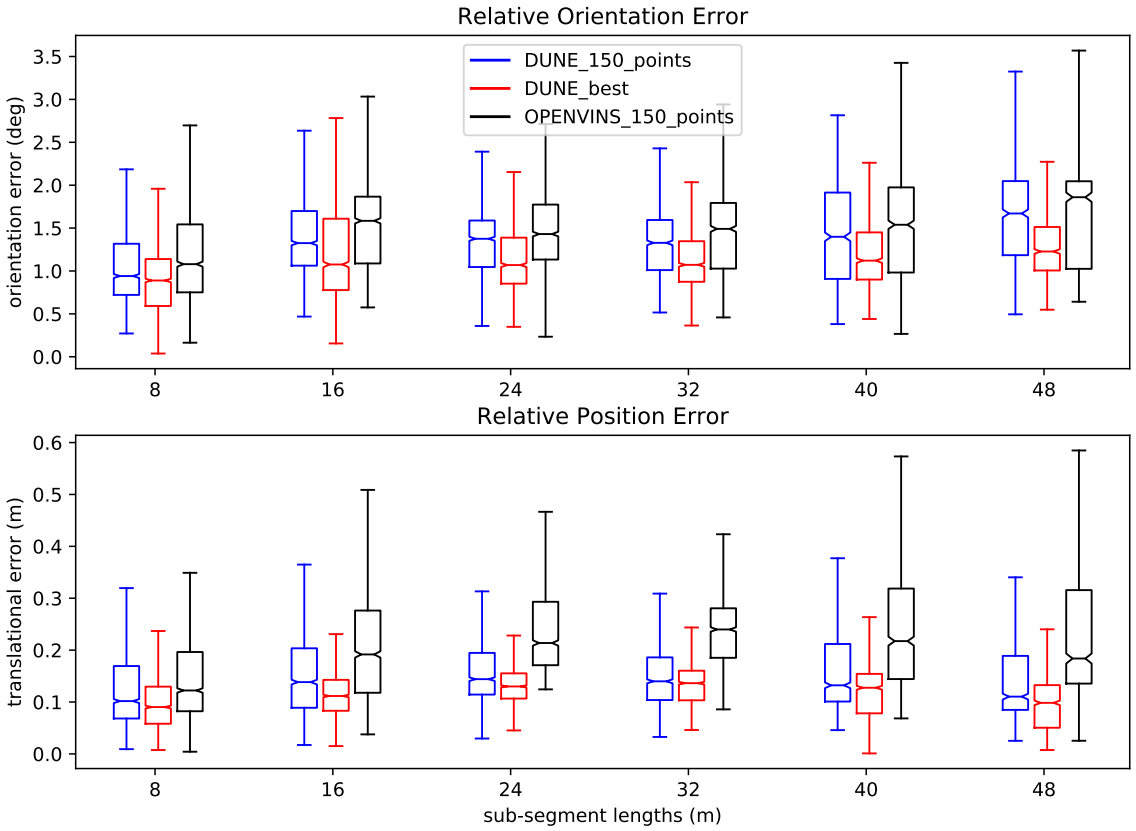


FIGURE 6.6 – Évolution de l'erreur RPE pour la sélection des meilleurs points

Les résultats sont présentés Tab. 6.7 sur le scénario *V1_03_difficult*. La sélection de points montre de meilleurs résultats, notamment sur l'estimation de la position. À partir du Tab. 6.7, les résultats sur l'orientation semblent comparables entre $DUNE\_150\_points$ et la sélection des 150 meilleurs points $DUNE\_best$ issus de 250 points aléatoires.

> ***Nota Bene*** Les points pour évaluer $OpenVINS\_150\_points$ et $DUNE\_150\_points$ sont identiques et choisis de façon aléatoire. À l'inverse, $DUNE\_best$ compte 150 points sélectionnés à partir de 250 points aléatoires.

La Fig. 6.6 met en évidence la performance et l'impact de sélectionner les meilleurs points sur chaque segment `D={8.0, 16.0, 24.0, 32.0, 40.0, 48.0}`. En effet, Fig. 6.6 montre que la sélection de points permet d'améliorer l'estimation de la position relative et également de l'orientation relative.

Quantiles à 3σ de ATE Afin d'avoir plus de précision sur la répartition de l'erreur ATE, la distribution des erreurs à 3σ est présentée Tab. 6.8, Tab. 6.9, et Tab. 6.10.

L'impact de la sélection des meilleurs points est particulièrement visible sur l'erreur de position, avec un écart type de 0.026m et une erreur moyenne de 0.051m. Ces performances

ATE(deg)	ATE(m)
$rmse\_ori = 2.875$	$rmse\_pos = 0.123$
$mean\_ori = 2.799$	$mean\_pos = 0.091$
$min\_ori = 1.280$	$min\_pos = 0.021$
$max\_ori = 4.204$	$max\_pos = 0.521$
$std\_ori = 0.657$	$std\_pos = 0.082$

TABLE 6.8 – Répartition de l'erreur ATE pour l'estimation $DUNE\_150$

ATE(deg)	ATE(m)
$rmse\_ori = 2.855$	$rmse\_pos = 0.057$
$mean\_ori = 2.798$	$mean\_pos = 0.051$
$min\_ori = 1.963$	$min\_pos = 0.002$
$max\_ori = 4.773$	$max\_pos = 0.115$
$std\_ori = 0.570$	$std\_pos = 0.026$

TABLE 6.9 – Répartition de l'erreur ATE pour l'estimation $DUNE\_best$

ATE(deg)	ATE(m)
$rmse\_ori = 4.416$	$rmse\_pos = 0.197$
$mean\_ori = 4.349$	$mean\_pos = 0.142$
$min\_ori = 2.836$	$min\_pos = 0.019$
$max\_ori = 5.698$	$max\_pos = 0.744$
$std\_ori = 0.764$	$std\_pos = 0.138$

TABLE 6.10 – Répartition de l'erreur ATE pour l'estimation $OpenVINS\_150$

dépassent les performances initiales du filtre proposées par OpenVINS (Tab. 6.10). L'orientation est également améliorée, avec un écart type de 0.5deg contre 0.6deg à partir d'une sélection aléatoire de 150 points (Tab. 6.8).

La trajectoire totale du scénario *V1_03_difficult* est présentée Fig. 6.7. Les trajectoires superposées des estimations réalisées à partir de DUNE_best, DUNE_150 et OpenVINS.

6.3 Conclusion

Après avoir introduit une méthode DUNE d'estimation des covariances locales dans le plan image sur la position des points d'intérêt, celle-ci a été intégrée à une hybridation serrée par un filtre MSCKF proposé par la plateforme OpenVINS. Les résultats de notre estimateur DUNE sont compétitifs et montrent une amélioration des performances, mettant en évidence la limite de l'hypothèse d'une erreur sur les observations indépendante et uniformément répartie. L'impact des covariances adaptées aux mesures lors d'une fusion de données est bénéfique. Par ailleurs, la sélection active des meilleurs points permet d'améliorer la précision de l'hybridation.

Toutefois, le modèle proposé estime les incertitudes réalisées entre deux instants k et $k+1$ et n'utilise pas les informations antérieures pour affiner son estimation. La mise en place d'une architecture récurrente permettrait probablement de surmonter ce problème.

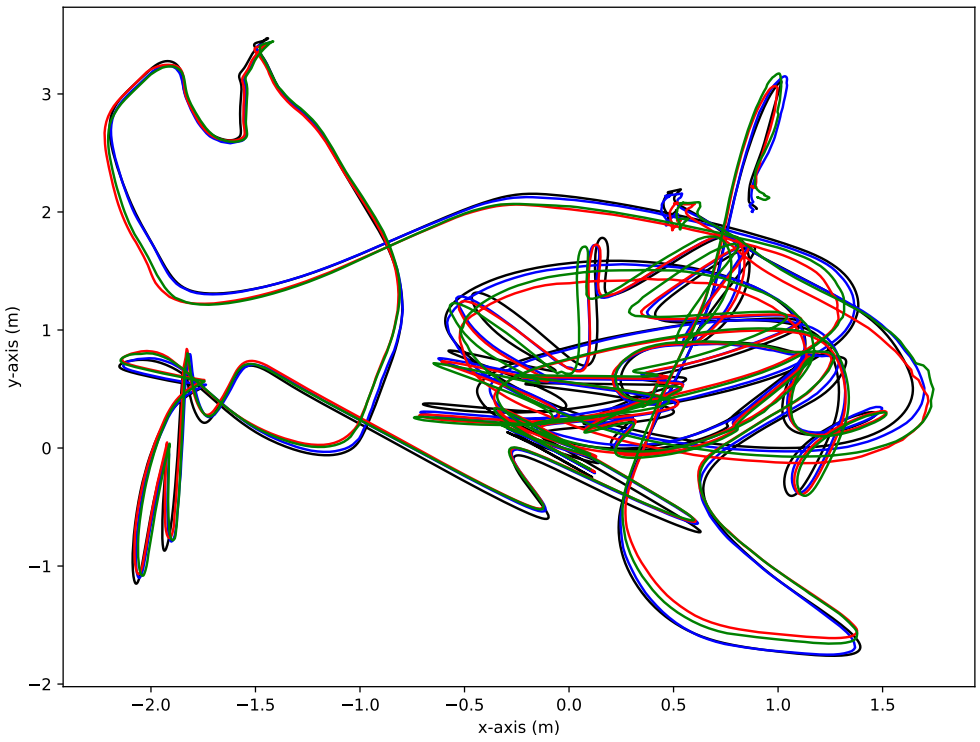


FIGURE 6.7 – Trajectoire globale du scénario *V1_03_difficult* - en noire la vérité terrain ; en bleu DUNE_best ; en rouge DUNE_150 et en vert OpenVINS

Chapitre 7

Intégration d'une mémoire à l'estimateur DUNE

Thus we may have knowledge of the past but cannot control it ; we may control the future but have no knowledge of it.

Claude Elwood Shannon

On ne construit du solide que sur le passé.

Thomas Stearns Eliot

L'estimateur DUNE fournit une incertitude sur la position des points d'intérêt entre deux instants t et $t+1$. Il ne prend pas en compte l'évolution du point dans le temps. Or, l'évolution du mouvement d'un point d'intérêt est liée aux positions antérieures du point. L'intégration d'un historique permet par conséquent de préciser l'estimation de l'incertitude sur la position des points d'intérêt. Dans ce chapitre, nous introduisons la notion de réseau de neurones récurrent, en particulier la notion de LSTM[1] est présentée. Un modèle amélioré de DUNE est présenté intégrant de la mémoire, DUNE-M. Ce modèle est évalué à partir des dataset SYNTHIA[Ros+16] et KITTI[GLU12]. Une comparaison à DUNE est réalisée mettant en évidence de bons résultats. Une application d'odométrie visuelle est présentée illustrant les bénéfices de l'utilisation d'une mémoire.

7.1 Présentation des outils mathématiques

Dans la suite des travaux, nous abordons des notions de réseaux de neurones récurrent (RNN[2]). Il convient de présenter les concepts et les outils nécessaire à la compréhension du manuscrit. Pour davantage de détails, le lecteur peut se référer à [GBC16], [Ola15] et [AA19].

7.1.1 Réseau de neurones récurrent

Les réseaux de neurones récurrents (RNN) [RHW85][Wer88] sont des cas particuliers de réseaux de neurones profonds caractérisés par la récurrence d'utilisation des couches du RdN.

1. *Long Short Term Memory*
2. Recurrent Neural Network

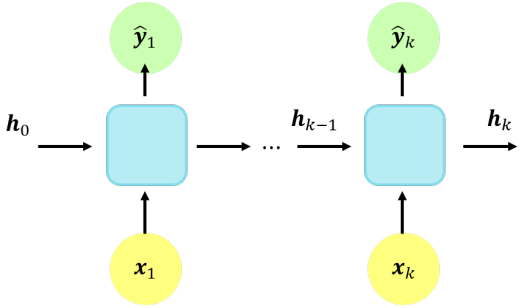
En effet, les différentes couches (*i.e.* étages) du réseau de neurones sont appelées de façon répétée afin d'itérer la même fonction sur des entrées en différé. La force des RNNs est d'exploiter le comportement temporel dynamique d'une application.

Les RNN sont des modèles puissants qui affichent d'excellentes performances dans de nombreuses tâches, notamment le traitement vidéo, la génération de sous-titre ou encore la traduction [KB13]. Par souci de concision, nous nous concentrerons sur des modèles RNN génériques. Soit $\boldsymbol{x} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \ldots, \boldsymbol{x}_T]$ une séquence temporelle d'entrées de taille $T \in \mathbb{R}$ avec $\forall k \in \mathbb{R}, \boldsymbol{x}_k \in \mathbb{R}^n$ une entrée de dimension n, et $\boldsymbol{y} = [\boldsymbol{y}_1, \ldots, \boldsymbol{y}_k, \ldots, \boldsymbol{y}_T]$ une séquence temporelle de sorties associées avec $\forall k, \boldsymbol{y}_k \in \mathbb{R}^m$ la sortie de dimension m associée à $\boldsymbol{x}_k$, où $(n, m) \in \mathbb{N}^2$. Le formalisme générique d'un RNN est issu de l'application répétée de la fonction f_h définie (7.1) où σ est la fonction d'activation *Sigmoïd* définie § 5.1.1.

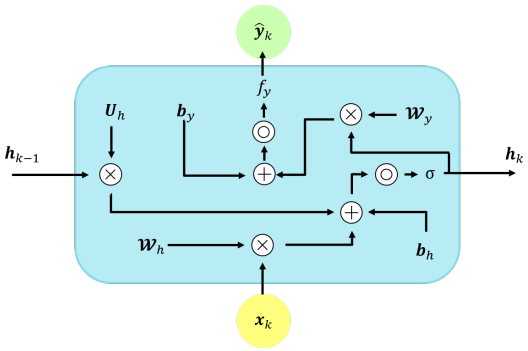
$$\boldsymbol{h}_k = f_h(\boldsymbol{x}_k, \boldsymbol{h}_{k-1}) = \sigma(\boldsymbol{x}_k \boldsymbol{\mathcal{W}}_h + \boldsymbol{h}_{k-1} \boldsymbol{U}_h + \boldsymbol{b}_h) \tag{7.1}$$

La récurrence de la fonction f_h génère un état intermédiaire h_k à l'instant k, couramment appelé *hidden state* (de l'anglais pour désigner un *état caché*). Cet état intermédiaire propage le résultat des observations antérieures à l'application courante, permettant de produire une estimation $\hat{\boldsymbol{y}}_k$ (7.2) résultant des expériences passées, tel que illustré Fig. 7.1a.

$$\hat{\boldsymbol{y}}_k = f_y(\boldsymbol{h}_k) = \boldsymbol{h}_k \boldsymbol{\mathcal{W}}_y + \boldsymbol{b}_y \tag{7.2}$$



(a) Réseau de neurones récurrent



(b) Fonctionnement d'un bloc RNN reprenant les équations 7.1 et 7.2

FIGURE 7.1 – Fonctionnement d'un réseau de neurones récurrent générique

La figure 7.1a montre la génération des états intermédiaires h_k en fonction de l'échantillon d'entrée $\boldsymbol{x}_k$ à l'instant k. En particulier, l'application répétée de la couche RNN illustré par le bloc bleu. Le fonctionnement interne du bloc RNN est donné par la figure 7.1b. $\boldsymbol{\mathcal{W}}_h$, $\boldsymbol{U}_h$ et $\boldsymbol{\mathcal{W}}_y$ définissent les poids du RNN et $\boldsymbol{b}_h$ et $\boldsymbol{b}_y$ sont les biais (§ 5.1.1). Ces coefficients sont identiques le long d'une séquence temporelle T. f_y et f_h sont des fonctions d'activation permettant la non-linéarité de l'application modélisée.

Toutefois, plus le nombre d'itération est élevé, moins les observations lointaines ont d'impact sur l'état courant du RNN. Les RNN génériques n'ont donc pas la capacité de garder en

mémoire un état dans le temps.

Parmi les réseaux de neurones récurrents, les réseaux GRU (Gated Reccurent Unit) [Cho+14] et LSTM (Long Short Term Memory) [HS97] se démarquent. Ces deux types de RNN intègrent un état interne qui joue le rôle de mémoire. Des travaux récents [Elm+20][Liu+17] ont montré les performances des RNN, en particulier les performances des couches LSTM pour corréler temporellement des données qui dépassent celles des couches GRU. Contrairement aux couches GRU, les LSTM gardent en mémoire plus d'information et sont capables de corréler des informations observées sur du long terme. Par la suite, nous détaillons le fonctionnement d'un réseau LSTM choisi pour intégrer une mémoire à l'estimateur DUNE.

7.1.2 Long Short Term Memory (LSTM)

Les réseaux Long-Short Term Memory ou *LSTM* sont une variation des réseaux RNN capable d'accumuler et d'oublier de l'information dans son état interne. Un LSTM est composé de quatre portes $\mathbf{\Gamma}$, ou *gate* en anglais, défini par (7.3). Ces portes sont des fonctions ayant des rôles bien précis au sein du LSTM. Tout comme précédemment, σ définit la fonction d'activation *Sigmoïd* (§ 5.1.1).

$$\mathbf{\Gamma} = \sigma(\boldsymbol{x}_k \boldsymbol{\mathcal{W}} + \boldsymbol{h}_{k-1} \boldsymbol{U} + \boldsymbol{b}) \tag{7.3}$$

La porte de mise a jour $\mathbf{\Gamma}_m$ détermine si les observations courantes doivent modifier le contenu de la mémoire. La porte de pertinence $\mathbf{\Gamma}_p$ fait le tri sur les observations antérieures et abandonne les moins pertinentes. Elle vient modérer le poids des observations passées à conserver en mémoire et celles à remplacer. La porte d'oubli $\mathbf{\Gamma}_o$ est capable d'effacer la totalité du contenu d'une cellule mémoire. Enfin la porte de sortie $\mathbf{\Gamma}_s$ définit l'influence du contenu de la cellule sur la sortie du neurone. Chaque porte est caractérisée par une matrice de poids et un biais. On note ω l'ensemble des paramètres tel que $\omega = \{\boldsymbol{\mathcal{W}}_m, \boldsymbol{U}_m, \boldsymbol{\mathcal{W}}_p, \boldsymbol{U}_p, \boldsymbol{\mathcal{W}}_o, \boldsymbol{U}_o, \boldsymbol{\mathcal{W}}_s, \boldsymbol{U}_s\}$. Le fonctionnement des portes au sein d'un LSTM est décrit Fig. 7.2. Tout comme les RNN décris précédemment, les LSTM possèdent des état intermédiaire h_k, ou *état caché*, relatif à chaque itération k de la séquence temporelle T.

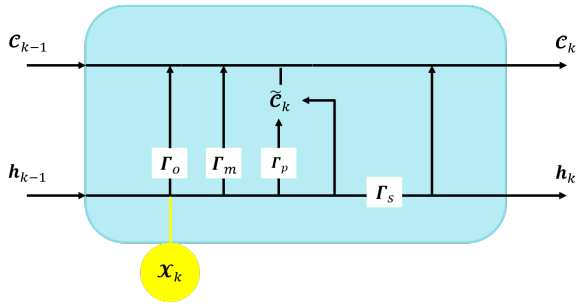


FIGURE 7.2 – Schéma bloc d'une cellule mémoire LSTM

Soit $\boldsymbol{\mathcal{C}}_k$ l'état interne d'un LSTM, également dénommé la *cellule* du LSTM, et $\tilde{\boldsymbol{\mathcal{C}}}_k$ son état interne intermédiaire. $\tilde{\boldsymbol{\mathcal{C}}}_k$ intervient dans la sélection des informations antérieures à conserver par l'application de $\mathbf{\Gamma}_p$ (eq. 7.4) et les informations courantes à retenir avec $\mathbf{\Gamma}_m$ (eq. 7.5).

$$\tilde{\boldsymbol{\mathcal{C}}}_k = \tanh\left(\boldsymbol{\mathcal{W}}_{\mathcal{C}}\left[\boldsymbol{\Gamma}_p \star \boldsymbol{h}_{k-1} \quad \boldsymbol{x}_k\right]^T + \boldsymbol{b}_{\mathcal{C}}\right) \tag{7.4}$$

où $\boldsymbol{\mathcal{W}}_{\mathcal{C}}$ est la matrice de poids relative à la cellule $\boldsymbol{\mathcal{C}}$ et $\boldsymbol{b}_{\mathcal{C}}$ son biais.

$$\boldsymbol{\mathcal{C}}_k = \boldsymbol{\Gamma}_m \star \hat{\boldsymbol{\mathcal{C}}}_k + \boldsymbol{\Gamma}_o \star \boldsymbol{\mathcal{C}}_{k-1} \tag{7.5}$$

L'opérateur $\star$ décrit la multiplication terme à terme. L'état caché $\boldsymbol{h}_k$ définit l'influence du contenu de la cellule mémoire sur la sortie du neurone (7.6).

$$\boldsymbol{h}_k = \boldsymbol{\Gamma}_s \star \boldsymbol{\mathcal{C}}_k \tag{7.6}$$

En conclusion, le rôle d'une couche LSTM est de prédire les états futurs en fonction des entrées actuelles et des états passés dans la mémoire interne.

7.1.3 Mode de fonctionnement des RNN

Les réseaux récurrents prennent en entrée une séquence temporelle permettant d'appliquer de façon itérative une fonction à une séquence d'entrées les unes après les autres. Soit $T_x \in \mathbb{R}$ la taille de la séquence temporelle des entrées $\boldsymbol{x}$ et $T_y \in \mathbb{R}$ la taille de la séquence temporelle des sorties $\hat{\boldsymbol{y}}$ du RNN. Dépendamment du problème envisagé, plusieurs modes de fontionnement sont possibles :

- Le mode *one-to-one* décrit le fonctionnement classique d'un RdN. Il considère une séquence d'entrée $T_x = 1$ et fournit une seule sortie associée $T_y = 1$.
- Le mode *one-to-many* associe à une entrée $T_x = 1$ plusieurs estimations de sortie $T_y > 1$. Chaque sortie à $k-1$ devenant l'entrée de l'application du RNN à k.
- Le mode *many-to-one* considère une séquence d'entrée $T_x > 1$ et prédit une seule estimation $\hat{\boldsymbol{y}}$ sortante $T_y = 1$, tel que décrit Fig. 7.3
- Le mode *many-to-many* associe autant de sorties qu'il existe d'entrées $T_x = T_y = T > 1$
- Enfin, le mode *many-to-many* associe une séquence de sorties à une séquence d'entrée dont le nombre d'entrées T_x et de sorties T_y diffère, $T_x \neq T_y$ (*e.g.* notamment utilisé pour la traduction)

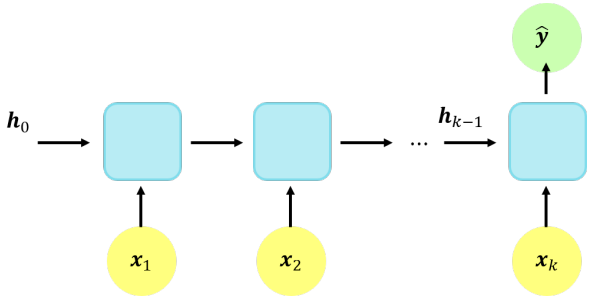


FIGURE 7.3 – Type de RNN *many to one* avec $T_x > 1$ et $T_y = 1$

L'estimateur DUNE-M présenté dans la suite de ce chapitre applique un fonctionnement *many to one* tel que présenté Fig. 7.3. Il prend en entrée une séquence temporelle d'images et

fournit une estimation d'incertitude sur la position des points d'intérêt. La section suivante est consacrée à définir le fonctionnement de DUNE-M, en particulier définir le format des entrées et des sorties ainsi que l'architecture du RdN.

7.2 DUNE-M

L'estimateur d'incertitude sur la position des points d'intérêt DUNE présenté chapitre 5 permet de qualifier la précision des observations entre deux instants k et $k+1$. En particulier, l'estimation précise de l'incertitude sur la position des points d'intérêt a permi de montrer la réduction de l'erreur sur la position d'un système mobile lors de son intégration dans un système de navigation VINS (chapitre 6). Toutefois, l'incertitude locale des points d'intérêt est estimée de façon indépendante de l'évolution de la position des points au cours du temps au delà de $k+1$. DUNE suppose donc que l'incertitude sur la position des points de $k+1$ à $k+2$ est indépendante et décorrélée de l'incertitude sur la position des points de k à $k+1$.

L'évolution de la position d'un point étant dépendante des positions précédentes du point, l'évolution de l'incertitude sur les positions d'un point dépend également de l'incertitude sur les positions précédentes du point. Nous montrons que l'intégration d'une mémoire dans notre estimateur DUNE permet d'affiner la précision sur l'estimation de l'incertitude de la position des points de suivi au cours du temps. Nous introduisons DUNE-M, l'estimateur DUNE augmenté d'une mémoire (DUNE-Mémoire), comme illustré figure 7.4. Ainsi, l'ensemble de l'historique de suivi de la position d'un point est exploité pour fournir des incertitudes plus précises.

Quelques travaux ont été consacrés à l'intégration de la mémoire, en particulier des couches LSTM, dans l'estimation de la position d'un système mobile [Wal+17]. Pose-LSTM [Wal+17] estime la pose d'un système mobile à partir des poses précédentes prises par le système mobile. En parallèle, des études sur la position des points étudient les LSTM, notamment SURF-LSTM [Elm+20] qui propose un descripteur robuste SURF [BTG06] intégrant des cellules mémoires LSTM.

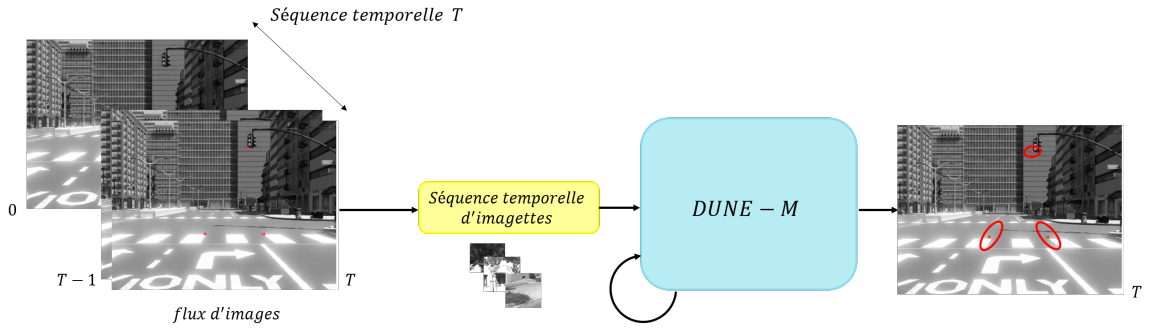


FIGURE 7.4 – Deep UNcertainty Estimation avec Mémoire (DUNE-M). Le système prend en entrée une séquence temporelle d'imagettes (§ 4.3.2) envoyées itérativement à DUNE-M pour prédire la covariance sur la position des points d'intérêt relative à l'image finale de la séquence temporelle T.

L'objectif de ce chapite est de caractériser l'évolution de l'incertitude sur la position de chacun des points suivis en fonction des positions antérieures des points d'intérêt (Fig. 7.4).

7.2.1 Format des données

Nous introduisons DUNE-M (Fig. 7.4), un estimateur de matrice de covariance local basé sur des réseaux de neurones récurrents (RNN). DUNE-M prend en entrée une séquence temporelle d'imagettes centrées sur un point tracké (définies § 4.3.2) et retourne une matrice de covariance relative à la dernière image de la séquence temporelle T.

Par choix, DUNE-M utilise des paires d'images successives. La structure est inspirée de DUNE et permet d'évaluer directement l'impact de l'utilisation d'une mémoire LSTM.

Considérons une séquence d'images successives $\boldsymbol{I} = [\boldsymbol{I}_0, \ldots, \boldsymbol{I}_k, \ldots, \boldsymbol{I}_T]$. Les images sont regroupées par paires telles que $[\{\boldsymbol{I}_0, \boldsymbol{I}_1\}, \ldots, \{\boldsymbol{I}_k, \boldsymbol{I}_{k+1}\}, \ldots, \{\boldsymbol{I}_{T-1}, \boldsymbol{I}_T\}]$. DUNE-M fournit alors une covariance dans $\mathbb{R}^{2\times2}$ (4.2) sur la position des points d'intérêt suivis à T.

Soit $\boldsymbol{W}_k$ (§ 5.2.1) la représentation de l'imagette de dimension (21×21) de la position du point suivi ${}^{uv}\tilde{\boldsymbol{m}}_k$ à partir d'une image $\boldsymbol{I}_k \in \mathbb{R}^{n\times m}$ à un instant $k \in |[0;T]|$ de la séquence temporelle, avec $(n, m) \in \mathbb{R}^2$. Pour rappel, la dimension de la fenêtre $\boldsymbol{W}_k$ est choisie pour coïncider à la dimension de la fenêtre d'étude du tracker KLT (§ 4.3.2).

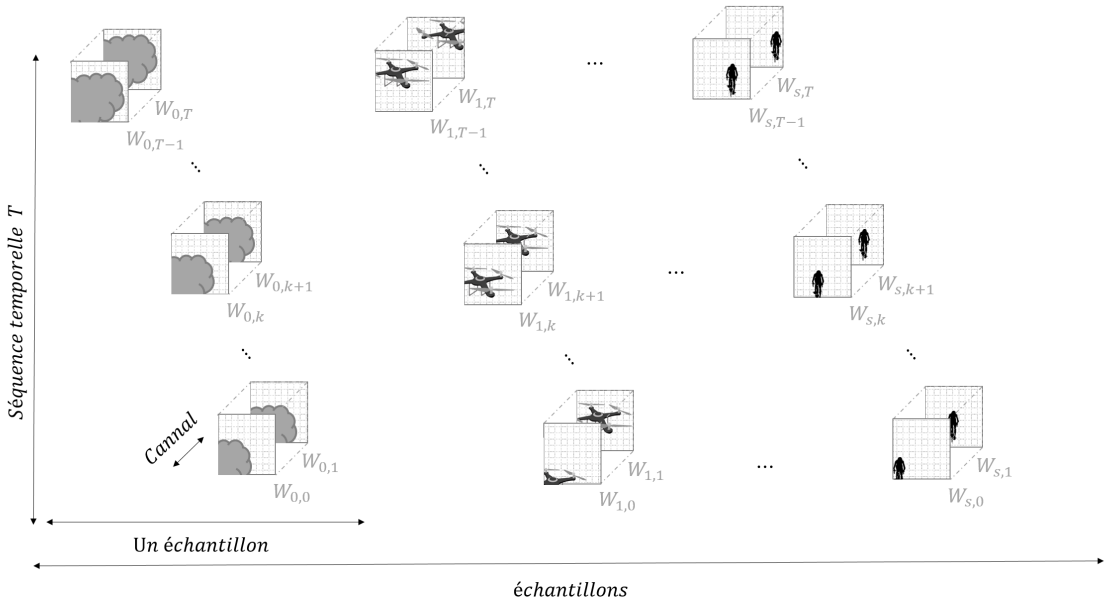


FIGURE 7.5 – Format des données de DUNE-M - concatenation des fenêtres centrées en ${}^{uv}\tilde{\boldsymbol{m}}_k$ et ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$, $\forall k \in |[0;T]|$. Un échantillon est composé de l'ensemble des paires de fenêtres d'une séquence temporelle relative au suivi d'un même point dans le temps. L'ensemble des échantillons s constituent le jeu de données $\mathcal{D}$.

Pour rappel, les données d'entrée de DUNE sont des stacks de deux imagettes centrées en ${}^{uv}\tilde{\boldsymbol{m}}$ le point tracké entre deux instants k et $k+1$. Les données d'entrées de DUNE-M sont

représentées par la concaténation de l'ensemble des stacks de deux imagettes entre deux instants $\forall k \in |[0;T]|$ de la séquence temporelle, tel que décrit Fig. 7.5.

Considérons le jeu de données $\mathcal{D} = \{ {}^{uv}\boldsymbol{W}_{i,k}\ , {}^{uv}\boldsymbol{e}_{i,T} \mid \forall i \in |[1;s]| \mid \forall k \in |[0;T]|\}$ où s définit le nombre d'échantillons dans le jeu de données et T la longueur de la séquence temporelle. On note $\boldsymbol{\Sigma}_i$ la covariance locale résultante du système DUNE associée à l'imagette $\boldsymbol{W}_T$ de l'échantillon i noté $\boldsymbol{W}_{i,T}$. L'erreur associée ${}^{uv}\boldsymbol{e}_{i,T}$ pour tout $i \in |[1;s]|$ définit l'erreur de tracking sur la séquence temporelle de $|[0;T]|$.

Le processus d'entrainement de l'estimateur DUNE-M d'incertitude de la position du suivi des points d'intérêt d'une séquence temporelle est représenté Fig. 7.6. Les paires d'imagettes sont itérées dans le réseau de neurones récurrent (RNN) DUNE-M. La prédiction $\boldsymbol{\Sigma}_i$ issue de l'estimateur est comparée à l'erreur de suivi sur un échantillon ${}^{uv}\boldsymbol{e}_{i,T}$ pour minimiser les poids du modèle $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ par la fonction de coût $\mathcal{L}$ définie § 4.2.

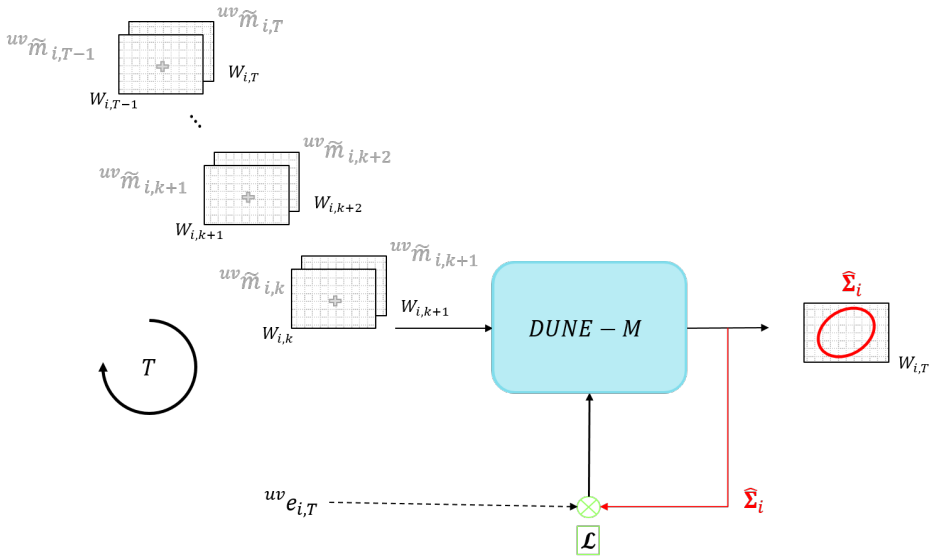


FIGURE 7.6 – Ajustement des poids du RNN DUNE-M pendant la phase d'apprentissage - DUNE-M prend en entrée un échantillon i décrit par une séquence imagettes concaténées $[{}^{uv}\boldsymbol{W}_{i,k}\ ; {}^{uv}\boldsymbol{W}_{i,k+1}]$ centrées en ${}^{uv}\tilde{\boldsymbol{m}}_{i,k}$ et ${}^{uv}\tilde{\boldsymbol{m}}_{i,k+1}$ $\forall k \in |[0;T]|$, et prédit la matrice de covariance $\tilde{\boldsymbol{\Sigma}}_i$ associée à ${}^{uv}\boldsymbol{W}_{i,T}$. La fonction de coût appliquée $\mathcal{L}$ compare l'erreur mesurée à T ${}^{uv}\boldsymbol{e}_{i,T}$ avec la matrice de covariance prédite $\tilde{\boldsymbol{\Sigma}}_i$

La fonction de coût $\mathcal{L}$ compare l'erreur issue d'un échantillon $i \in s$ sur la séquence temporelle de taille T avec la covariance prédite par DUNE-M $\tilde{\boldsymbol{\Sigma}}_i$ relative à ${}^{uv}\tilde{\boldsymbol{m}}_{i,T}$. DUNE-M est composé d'un RNN qui itère des paires d'imagettes $\forall k \in |[0;T]|$. Alors $\mathcal{L}$ est décrit par (7.7).

$$\forall i\ ,\ \underset{\boldsymbol{\Sigma}_i;\boldsymbol{\mathcal{W}}_i;\boldsymbol{b}_i}{\operatorname{argmin}}\ \mathcal{L}_{\boldsymbol{\Sigma}_i}(\{\mathcal{R}\overset{T}{\underset{k=0}{\mathcal{N}}}\mathcal{N}([{}^{uv}\boldsymbol{W}_{i,k}\ ; {}^{uv}\boldsymbol{W}_{i,k+1}];\boldsymbol{\mathcal{W}}_i;\boldsymbol{b}_i), {}^{uv}\boldsymbol{e}_{i,T}\}) \tag{7.7}$$

7.2.2 Structure du réseau de neurones DUNE-M

De même structure que DUNE (définie § 5.2.2), DUNE-M est un réseau de neurones composé d'un bloc convolutif, détaillé dans le tableau 5.1, et de couches denses. La structure convolutive est une succession de 4 couches convolutives avec une taille de Kernel $\boldsymbol{Kr}$ (définie § 5.1) fixe de 3×3 suivies d'une fonction d'activation *leakyrelu* (§ 5.1.1). Une couche de normalisation et un dropout fixé à 20% sont ajoutés après chaque couche convolutive.

> ***Nota Bene*** Le dropout intervient uniquement au cours de la phase d'apprentissage pour éviter le surajustement (§ 5.1).

Layer	Kernel	Stride	Padding	Filters
Conv_1	3×3	2×2	None	32
Conv_2	3×3	1×1	None	64
Conv_3	3×3	1×1	None	128
Conv_4	3×3	1×1	None	256

TABLE 7.1 – Structure des couches convolutives de DUNE-M

La construction de DUNE-M a été réalisée en intégrant une couche LSTM à la sortie du bloc convolutif et en entrée des couches denses (Fig. 7.7). L'objectif est de capturer les caractéristiques sortantes des CNN et de corréler ces caractéristiques temporellement. L'hypothèse de base est que l'évolution de l'incertitude sur la position d'un point de suivi dépend de ses positions antérieures. L'utilisation d'une mémoire LSTM est donc le moyen d'observer l'évolution des caractéristiques relatives à un point suivi le long d'une séquence temporelle et, en particulier, de pouvoir apprendre l'incertitude sur la position d'un point suivi à partir de l'historique complet de l'évolution du point depuis sa détection. Pour rappel, le LSTM permet des corrélations sur de la mémoire à long-terme, et montre de meilleures performances que des réseaux GRU [Elm+20] [Liu+17] (§ 7.1.1).

Soit une entrée $\boldsymbol{x}_k = [{}^{uv}\boldsymbol{W}_k\,;\,{}^{uv}\boldsymbol{W}_{k+1}]$ issue de la séquence $\boldsymbol{x}_i = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \ldots, \boldsymbol{x}_T]_i$, avec i un échantillon. i désigne également l'indice d'un point ${}^{uv}\tilde{\boldsymbol{m}}_i$ suivi sur la séquence temporelle $T \in \mathbb{R}$, et $\boldsymbol{x}_i$ représente la piste du point ${}^{uv}\tilde{\boldsymbol{m}}_i$ (*e.g.* Fig. 4.12).

Par choix, le bloc LSTM est positionné en sortie des CNN afin de collecter toutes les caractéristiques issues des CNN. L'évolution des caractéristiques à travers la séquence temporelle d'images (*i.e.* du flou optique, du bruit, de la texture etc..) est directement liée à l'incertitude sur la position d'un point d'intérêt. Ainsi, la capture et la corrélation de l'évolution de ces caractéristiques permet de fournir une estimation plus fine de l'incertitude. Afin de collecter toutes les caractéristiques issues des CNN pour chaque entrée $\boldsymbol{x}_k$, le bloc CNN est mis en récurrence et applique les mêmes fonctions, le long de la séquence temporelle T. Autrement dit, $\forall k \in T$, chaque entrée $\boldsymbol{x}_k$ de la séquence temporelle est soumise aux mêmes filtres kernels $\boldsymbol{Kr}$ du bloc CNN. En particulier, pour une séquence donnée i, les fonctions CNN seront identiques (*i.e.* des poids identiques $\boldsymbol{\mathcal{W}}$ et $\boldsymbol{b}$ sont appliqués pour chaque échantillon i). Un tenseur de sortie concatène le résultat séquentiel de chaque entrée $\boldsymbol{x}_k$ à travers les couches CNN.

Nota Bene En pratique, l'itération des couches convolutives est réalisées par une fonction *TimeDistributed* qui applique un délai entre chaque entrée $\boldsymbol{x}_k$.

À l'issue de l'extraction des caractéristiques de chaque entrée par récurrence du bloc CNN, le tenseur est collecté par 256 cellules mémoire d'une couche LSTM suivi d'un droptout de 50%.

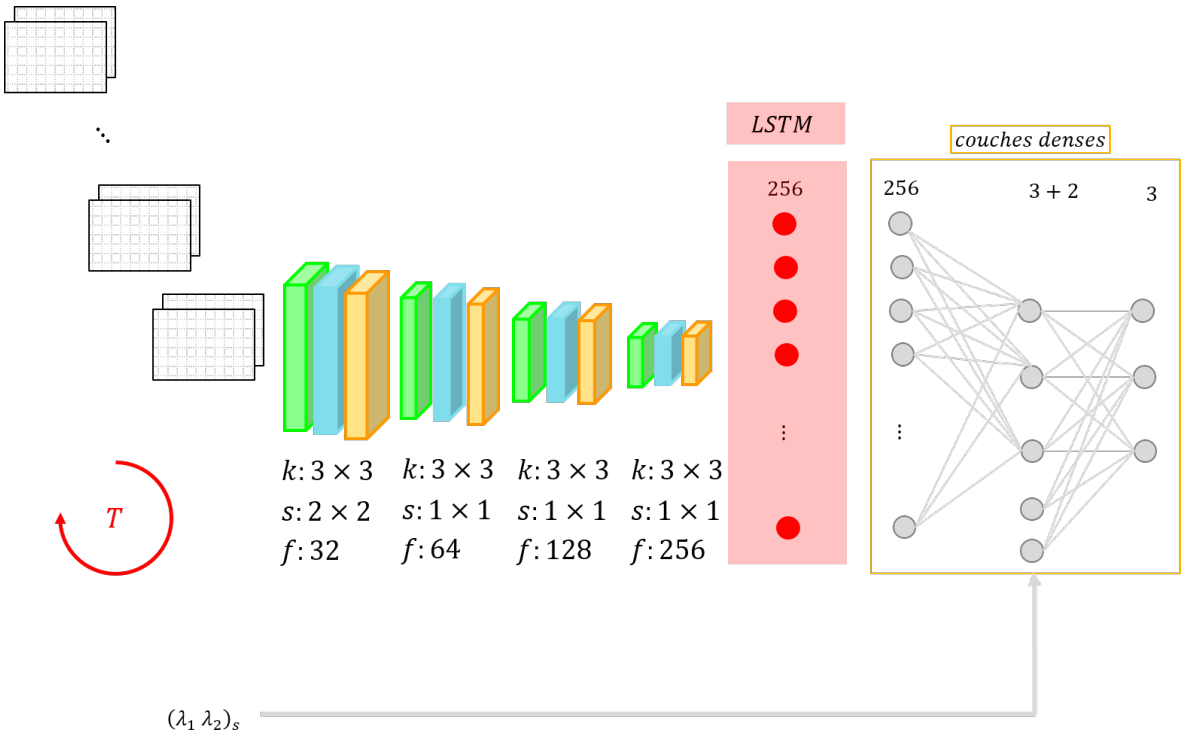


FIGURE 7.7 – Structure de l'estimateur DUNE-M d'incertitude de la position du suivi des points d'intérêt d'une séquence temporelle intégrant les paramètres de Harris $(\lambda_1, \lambda_2)_i$ - T est la taille de la séquence temporelle, i un échantillon, k indique la dimension des Kernels ; s la dimension des strides ; et f le nombre de filtres. Les couches vertes représentent les convolutions, les bleues les couches de normalisation et les jaunes les dropouts. Chaque couche du RdN, à l'exception de la dernière, est suivie d'une fonction d'activation *leakyrelu*

La sortie des couches convolutives et des cellules mémoire est suivie de trois couches denses intégrant les paramètres de Harris (§ 5.2.2.3). Elles sont composées respectivement de 256, 5 et 3 sorties. Chaque couche du RdN DUNE-M, à l'exception de la dernière, est suivie d'une fonction d'activation *leakyrelu* (§ 5.1.1).

La Fig. 7.7 montre le fonctionnement et la structure de DUNE-M. À l'image des différents types de fonctionnement de RNN présentés § 7.1.3, DUNE-M est un RNN *many-to-one*. L'estimateur prend en entrée une séquence temporelle $\boldsymbol{x}_i$ et renvoie l'incertitude $\boldsymbol{\Sigma}_i \in \mathbb{R}^{2\times2}$, composée de trois paramètres l_1, d_1 et d_2 (§ 4.2.2), sur les coordonnées du point ${}^{uv}\tilde{\boldsymbol{m}}_{i,T}$. La force de notre architecture DUNE-M est qu'elle n'a pas modifié le fonctionnement de base de la structure DUNE, permettant une amélioration sans modifier le format originel. L'impact

de la mémoire est donc le résultat de la comparaison des performances des deux estimateurs DUNE et DUNE-M.

7.3 Évaluations et résultats

Dans la suite de cette section, nous fixons la taille des séquences temporelles à $T = 5$ par souci de simplicité. Les séquences d'entrées à taille variable sont abordées § 7.4. Tout comme §5.3, les résultats sont évalués avec le dataset SYNTHA et KITTI. Les métriques utilisées pour réaliser cette évaluation sont les mêmes que celles présentées § 5.3.1.

7.3.1 Estimateur DUNE-M appliqué à SYNTHIA

7.3.1.1 Préparation des données

Nous évaluons d'abord DUNE-M sur le jeu de données SYNTHIA [Ros+16], tiré de la dernière version SYNTHIA-AL [ZB+19].

Au cours de chaque scénario, les points sont détectés et trackés (chapitre 4). L'erreur est mesurée entre la position du point suivi ${}^{uv}\tilde{\boldsymbol{m}}_{k+1}$ à $k+1$ et la reprojection ${}^{uv}\boldsymbol{m}_{k+1}$ du point ${}^{w}\boldsymbol{M}$ issu des cartes de profondeur denses fournies par SYNTHIA (7.8).

$$ {}^{pix}\boldsymbol{m}_{k+1} = \pi\left(\pi^{-1}({}^{pix}\boldsymbol{m}_k\,,\,{}^{C_k}_{w}\boldsymbol{T})\,,\,{}^{C_{k+1}}_{w}\boldsymbol{T}\right) \tag{7.8} $$

Un seuil d'erreur de 3 pixels entre deux instants k et $k+1$ est fixé pour maintenir le jeu de données intègre pour l'apprentissage. Les points détectés ou suivis sur des objects dynamiques sont également éliminés.

Le jeu de données (§ 4.3.2) créé est produit à partir de 12 scénarios différents faisant partie du dataset SYNTHIA-AL-Train. Chaque scénario est urbain ou semi-urbain et comprend diverses perturbations visuelles telles que la pluie ou la surexposition. Les séquences vidéo ont une longueur de 250 à 750 images.

7.3.1.2 Paramétrisation de DUNE

70% des données décrites dans la section 5.3.2.1 sont utilisées pour entraîner le réseau neuronal, 15% sont conservés pour la phase de test et 15% pour l'évaluation. Nous effectuons un apprentissage sur 50 epochs avec une taille de bacth de 64. Le pas d'apprentissage initial est de $1 \cdot 10^{-04}$ et diminue lorsqu'un plateau est atteint et persiste sur 8 epochs. Nous choisissons Nadam comme optimiseur (§ 5.1.3) avec ses paramètres fixés à $\beta_1 = 0.9$ et $\beta_2 = 0.99$. Le modèle final est sélectionné comme étant celui qui donne les meilleurs résultats NNE et MD sur les données de test. Cette évaluation est effectuée après chaque epoch.

7.3.1.3 Résultats

Les résultats sont évalués sur des séquences temporelles $\boldsymbol{x}_i = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \ldots, \boldsymbol{x}_T]_i$ fixes de taille $T = 5$ où $\boldsymbol{x}_k = [{}^{uv}\boldsymbol{W}_k\,;\,{}^{uv}\boldsymbol{W}_{k+1}]$. À l'instant $k = 1$, le détecteur de Harris est appliqué,

FIGURE 7.8 – Scénario SYNTHIA - +représentent la piste des points ${}^{uv}\boldsymbol{m}$ reprojetés ; +la piste des points trackés ${}^{uv}\tilde{\boldsymbol{m}}$.

suivi $\forall k > 1$ par le tracker KLT. Fig. 7.8 met en avant les pistes des i points trackés ${}^{uv}\tilde{\boldsymbol{m}}_i$, ainsi que la reprojection de ${}^{w}\boldsymbol{M}$ dans $\boldsymbol{\mathcal{R}}_{pix}$ en ${}^{uv}\boldsymbol{m}_i$.

Fig. 7.9 230 séquences $\boldsymbol{x}_i$ sont évaluées, relatives aux 230 points affichés Fig. 7.8. Les incertitudes prédites par DUNE-M sur la position des points ${}^{uv}\tilde{\boldsymbol{m}}_{i,T=5}$ sont représentées Fig. 7.9.

On observe peu d'erreurs, dûes au scénario quasi-statique. On note néanmoins quelques erreurs identifiables le long des lignes du trottoir à gauche de la voiture Fig. 7.8. En réponse, on mesure des erreurs identifiées keypoints 110, 160 et 210 Fig. 7.9 avec une prédiction de covariance appropriée.

L'erreur moyenne sur l'ensemble du jeu de données d'évaluation est -0.0549 pixel avec une erreur maximum de 13.4125 pixels.

7.3.1.4 Comparaison à l'état de l'art

DUNE-M est évalué à partir des données d'évaluations. Les incertitudes prédites sont analysées à 1σ, 2σ et 3σ. Tout comme dans le chapitre 5, nous comparons nos résultats (Tab. 7.2) à la méthode [WM17] basée sur l'estimation incrémentale de l'incertitude sur la séquence temporelle $T = 5$ avec une valeur initiale constante fixée à 0.5 pixel, ainsi qu'au modèle de covariance fixe (Fix-C) égale à $0.5 \cdot 5$ pixel (7.9).

$$Fix - C = \begin{pmatrix} \sigma_{uu} & 0 \\ 0 & \sigma_{vv} \end{pmatrix} \tag{7.9}$$

Les résultats (Tab. 7.2) montrent que la méthode Fix-C est très pessimiste. En effet, $MD < 1$ ainsi que $NNE < 1$. Par ailleurs, plus de 95% des erreurs observées sont inclues à 1σ. [WM17]

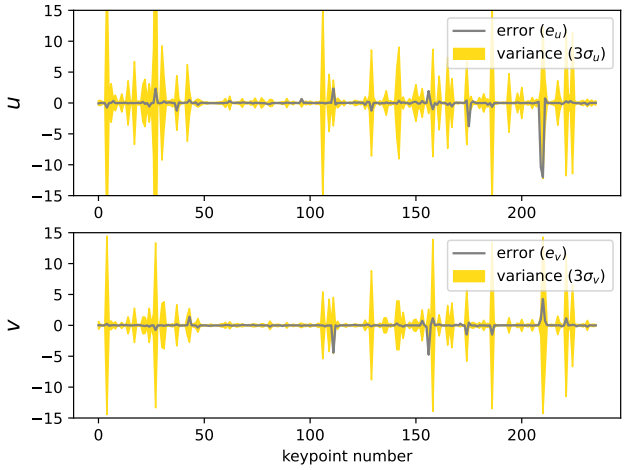


FIGURE 7.9 – Évolution de l'incertitude sur la position des points d'intérêt prédite par DUNE et comparaison à l'erreur de tracking.

Métriques	DUNE-M + (λ_1, λ_2)	[WM17]	Fix-C	Valeures idéales
$3\sigma_u$	96.50	88.21	99.87	99.7
$3\sigma_v$	97.12	90.72	99.97	99.7
$2\sigma_u$	91.54	81.19	99.26	95
$2\sigma_v$	92.91	80.86	99.77	95
$1\sigma_u$	76.90	65.38	95.41	68
$1\sigma_v$	81.03	60.89	97.42	68
MD	**1.11**	0.96	0.29	1
NNE	**0.96**	0.92	0.29	1

TABLE 7.2 – Comparaison des résultats de DUNE-M par rapport à l'état de l'art

montre de meilleurs performances, avec une distance de Mahalanobis proche de 1, de même que $NNE \sim 1$. Toutefois, moins de 95% des erreurs observées sont inclues à 3σ. DUNE-M montre des performances qui surpassent les méthodes Fix-C et [WM17]. Le modèle évalué a une distance de mahalanobis de 1.1 ainsi que $NNE \sim 1$. De plus, $\sim 95\%$ des erreurs observées sont comprises dans 3σ.

7.3.1.5 Comparaison à DUNE

Afin de comparer l'impact et les bénéfices de l'utilisation d'une mémoire LSTM, nous comparons notre méthode DUNE (présentée chapitre 5) et DUNE-M. Le modèle utilisé pour l'évaluation de DUNE est entrainé selon les spécifications chapitre 5. Pour chaque séquence temporelle $\boldsymbol{x}_i = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \ldots, \boldsymbol{x}_T]_i$, DUNE retourne T prédictions d'incertitude de $\boldsymbol{x}_k = [{}^{uv}\boldsymbol{W}_k\,;\,{}^{uv}\boldsymbol{W}_{k+1}]$, $\forall k \in |[1, T]|$. La prédiction finale de la séquence $\boldsymbol{x}_i$ est donnée par la somme des T prédictions d'incertitude de $\boldsymbol{x}_k$ fournies par DUNE. Les résultat sont mis en relation Tab. 7.3 avec les prédictions issues de DUNE-M, estimant directement la prédiction

de la séquence $\boldsymbol{x}_i$.

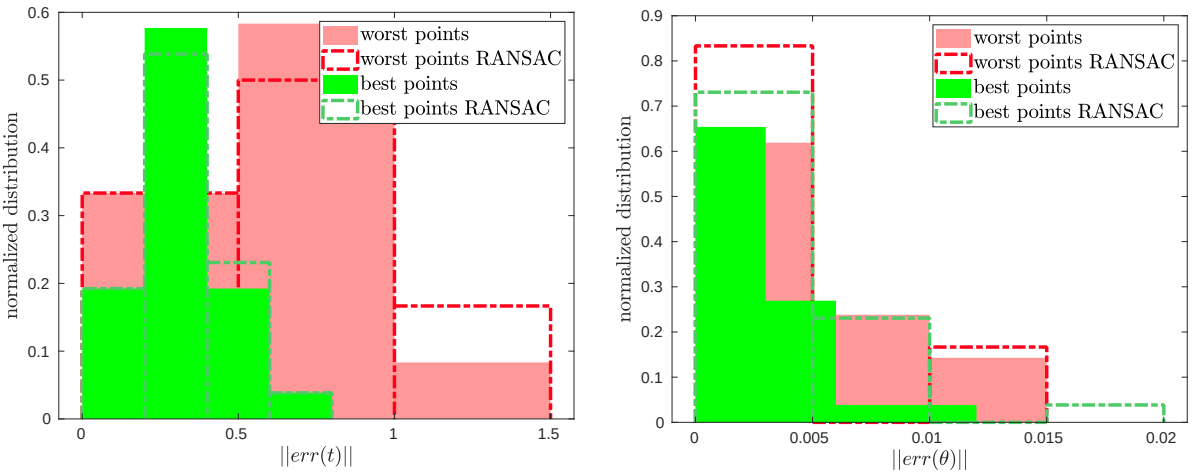
Métriques	DUNE + (λ_1, λ_2)	DUNE-M + (λ_1, λ_2)	Valeures idéales
$3\sigma_u$	94.68	96.50	99.7
$3\sigma_v$	92.20	97.12	99.7
$2\sigma_u$	88.26	91.54	95
$2\sigma_v$	84.74	92.91	95
$1\sigma_u$	71.46	76.90	68
$1\sigma_v$	65.34	81.03	68
MD	1.21	**1.11**	1
NNE	1.07	**0.96**	1

TABLE 7.3 – Comparaison des résultats de DUNE et DUNE-M

DUNE montre de bon résultat avec $NNE \sim 1$ et une distance de mahalanobis $MD = 1.2$. Toutefois, $DUNE - M$ montre une amélioration avec un modèle un peu moins optimiste : $MD = 1.1$ et $NNE \sim 1$.

7.3.1.6 Évaluation des prédictions de DUNE sur l'estimation du mouvement

Nous appliquons la méthode de validation présentée § 5.3.2.6 entre $k = 0$ et $k = T$. Les meilleurs points (*i.e. best points*) sont sélectionnés à partir du critère $\sqrt{tr(\boldsymbol{\Sigma}_i)} \leq 0.5 * T$, $i \in |[0; s]|$ et les pires points (*i.e. worst points*) vérifient $\sqrt{tr(\boldsymbol{\Sigma}_i)} > 0.5 * T$.



(a) (b)

FIGURE 7.10 – Impact sur l'estimation du mouvement - (a) Erreur sur la translation (en m) ; (b) Erreur sur la rotation (en deg).

Le mouvement de la caméra est estimé avec EPnP [LMNF09] à partir de ${}^{w}\boldsymbol{M}$ fourni par synthia. L'erreur sur la pose est mesurée avec :

$$ {}_{k+1}T_{err} = {}_{k+1}T_{groundtruth} \cdot {}_{k+1}T^{-1}_{estimated} \tag{7.10} $$

où ${}_{k+1}T_{err}$ définit l'erreur de transformation rigide à l'instant $k+1$.

Les résultats (Fig. 7.10) montre une erreur moyenne de 0.3255 m en translation pour les meilleurs points et 0.6073 pour les pires points. Ces métriques surpassent légèrement celle issues de RANSAC avec une erreur moyenne en translation de 0.3273 m pour les meilleurs points et 0.6195 pour les pires (Fig. 7.10a). Ces résultats sont également visibles sur les erreurs de rotation (Fig. 7.10b) avec une erreur moyenne de 0.0015 deg pour les meilleurs points et 0.0025 deg pour les pires ; avec RANSAC on estime une erreur moyenne de 0.0025 deg pour les meilleurs points et 0.0025 deg pour les pires.

Les résultats présentés par DUNE sur la Fig. 5.13 mettent en évidence une erreur plus grande en translation et comparable en rotation. L'erreur issue des meilleurs points a diminué et est passée d'un maximum de 1m à 0.75m.

7.3.2 Estimateur DUNE appliqué à KITTI

7.3.2.1 Préparation des données

DUNE-M a également été évalué sur KITTI [Gei+13] afin de valider notre approche dans des scénarios réels. Le modèle a été appris à partir du scénario `2011_09_26_0002`. `2011_09_26_0005` a été utilisé pour tester le modèle au cours de l'apprentissage. Enfin, le scénario `2011_09_26_0113` est adopté pour la phase d'évaluation de DUNE-M.

Au cours de chaque scénario, les points sont détectés et trackés (chapitre 4). Tout comme dans le chapitre 5, la structure 3D exacte du monde (*i.e.* carte de profondeur, points géolocalisés, etc ..) n'est pas disponible directement. L'application du tracker en rétropropagation (*i.e.* sur un aller-retour) permet de mesurer le biais induit par la méthode de tracking et en particulier, d'évaluer l'incertitude de position sur ce point. En particulier, on se place à $T = 5$, ainsi on opère un aller retour sur un total de 10 images (5 allers et 5 retours).

Nous mesurons l'erreur sur la séquence

$$\left[{}^{uv}\boldsymbol{m}_1, \ldots, {}^{uv}\,\tilde{\boldsymbol{m}}_k, \ldots, {}^{uv}\,\tilde{\boldsymbol{m}}_T, {}^{uv}\,\tilde{\boldsymbol{m}}_{T-1}, \ldots, {}^{uv}\,\tilde{\boldsymbol{m}}_k, \ldots, {}^{uv}\,\tilde{\boldsymbol{m}}_1\right]_i \tag{7.11}$$

où $T = 5$. Elle est définie par la distance entre la position du point détecté ${}^{uv}\boldsymbol{m}_1$ et la position du point tracké sur un aller-retour ${}^{uv}\tilde{\boldsymbol{m}}_1$ tel que décrit (§ 4.3.4). Dès lors, on suppose une répartition de l'évolution uniforme sur la fenêtre $|[1;T]|$

Un seuil d'erreur de 3 pixels entre deux instants k et $k+1$ est fixé pour maintenir le jeu de données intègre pour l'apprentissage. Les points détectés ou suivis sur des objects dynamiques sont également éliminés.

7.3.2.2 Paramétrisation de DUNE

Comme précédemment, 70% des données sont utilisées pour entraîner le réseau neuronal, 15% sont conservées pour la phase de test et 15% pour l'évaluation. Nous effectuons un apprentissage sur 150 epochs avec une taille de bacth de 64. Le pas d'apprentissage initial est identique ainsi que la méthode d'optimisation Nadam. Le modèle final est sélectionné comme étant celui qui donne les meilleurs résultats MD et NNE sur les données de test.

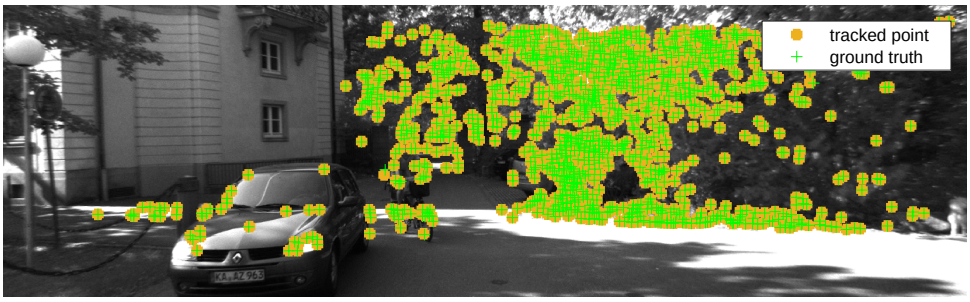


FIGURE 7.11 – Scénario KITTI - +représentent les points ${}^{uv}\boldsymbol{m}_k$ détectés par Harris ; +les points trackés sur un aller-retour ${}^{uv}\tilde{\boldsymbol{m}}_k$.

7.3.2.3 Résultats

Les résultats sont évalués sur des séquences temporelles $\boldsymbol{x}_i = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \ldots, \boldsymbol{x}_T]_i$ fixes de taille $T = 5$ où $\boldsymbol{x}_k = [{}^{uv}\boldsymbol{W}_k \, ; \, {}^{uv}\boldsymbol{W}_{k+1}]$. À l'instant $k = 1$, le détecteur de Harris est appliqué, suivi $\forall k > 1$ par le tracker KLT jusqu'à $k = T$. Puis le tracker est appliqué en rétropropagation de $k = T - 1$ à $k = 1$. L'apprentissage est effectué seulement sur l'aller afin d'estimer l'incertitude relative à la position du point ${}^{uv}\tilde{\boldsymbol{m}}_{i,k=T}$ à l'instant T avec l'erreur de tracking sur l'aller-retour. En conséquence, la covariance résultant liée à l'erreur doit être divisée par deux. La figure 7.8 montre les i points trackés ${}^{uv}\tilde{\boldsymbol{m}}_i$ sur l'aller-retour, ainsi que les i points ${}^{uv}\boldsymbol{m}_i$ issus de la détection de Harris à $k = 1$.

La portion du scénario `2011_09_26_0113` illustrée Fig. 7.11 est issue d'un mouvement de rotation dynamique. L'erreur moyenne sur l'ensemble du jeu de donnée d'évaluation est 0.0057 pixel avec une erreur maximum de 14.9996 pixels.

7.3.2.4 Comparaison à l'état de l'art

DUNE-M est évalué à partir des données d'évaluations issues du scénario `2011_09_26_0113`. Les incertitudes prédites sont analysées à 1σ, 2σ et 3σ et référencées dans le tableau Tab. 7.4.

Tout comme dans le chapitre 5, nous comparons nos résultats (Tab. 7.2) à la méthode [WM17] et à un modèle de covariance fixe (Fix-C) égale à $0.5 \cdot 5 \cdot 2$ pixel (7.9).

Contrairement à Tab. 7.2, la méthode [WM17] donne des résultats trop optimistes avec une distance de mahalanobis $MD \sim 1.5$, validée également par $NNE \sim 1.4$. Ce résultat est visible dans la répartition des quantiles entre 84% et 90% des erreurs observées à 1σ. À l'inverse, la méthode Fix-C est pessmiste comme précédemment avec une distance de mahalanobis $MD \sim 0.3$ et $NNE \sim 0.3$. DUNE-M montre les meilleurs résultats notamment avec

Métriques	DUNE-M + (λ_1, λ_2)	[WM17]	Fix-C	Valeures idéales
$3\sigma_u$	90.03	90.63	98.3	99.7
$3\sigma_v$	90.76	96.0	99.9	99.7
$2\sigma_u$	88.33	95.21	96.46	95
$2\sigma_v$	88.26	88.7	99.5	95
$1\sigma_u$	83.56	84.56	94.43	68
$1\sigma_v$	82.80	91.23	98.5	68
MD	**1.17**	1.46	0.26	1
NNE	**1.07**	1.37	0.26	1

TABLE 7.4 – Comparaison des résultats de DUNE-M par rapport à l'état de l'art

$MD \sim 1.2$ et $NNE \sim 1$. 90% des erreurs observées sont comprises dans 3σ. Ces résultats sont très satisfaisants et constituent les meilleurs résultats disponibles dans la littérature sur l'estimation d'incertitude sur la position des points d'intérêt.

7.3.2.5 Comparaison à DUNE

De façon similaire à 7.3.1.5, nous comparons notre méthode DUNE (présentée chapitre 5) et DUNE-M. Pour chaque séquence temporelle $\boldsymbol{x}_i = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_k, \ldots, \boldsymbol{x}_T]_i$, DUNE retourne T prédictions d'incertitude de $\boldsymbol{x}_k = [{}^{uv}\boldsymbol{W}_k\,;\,{}^{uv}\boldsymbol{W}_{k+1}]$, $\forall k \in |[1,T]|$. La prédiction finale de la séquence $\boldsymbol{x}_i$ est la somme des T prédictions d'incertitude de $\boldsymbol{x}_k$ fournies par DUNE. Les résultats sont mis en relation Tab. 7.5 avec les prédictions issues de DUNE-M, estimant directement la prédiction de la séquence $\boldsymbol{x}_i$.

Métriques	DUNE + (λ_1, λ_2)	DUNE-M + (λ_1, λ_2)	Valeures idéales
$3\sigma_u$	93.46	90.03	99.7
$3\sigma_v$	96.16	90.76	99.7
$2\sigma_u$	91.63	88.33	95
$2\sigma_v$	94.73	88.26	95
$1\sigma_u$	88.4	73.56	68
$1\sigma_v$	90.93	72.80	68
MD	0.70	1.17	1
NNE	0.68	1.07	1

TABLE 7.5 – Comparaison des résultats de DUNE et DUNE-M

Le modèle d'estimation issu de DUNE, comme étant la somme des incertitudes des entrées de la séquence temporelle $\boldsymbol{x}_i$, est pessimiste. En effet, $NNE \sim 0.7$, de même que la distance de mahalanobis $MD = 0.7$. $DUNE - M$ montre de meilleurs résultats, notamment avec $NNE \sim 1$ et $MD = 1.17$.

7.3.2.6 Évaluation des prédictions de DUNE sur l'estimation du mouvement

De façon similaire au § 7.3.1.6, nous appliquons une validation géométrique dont la vocation est d'évaluer l'impact des variances sur la reconstruction du mouvement.

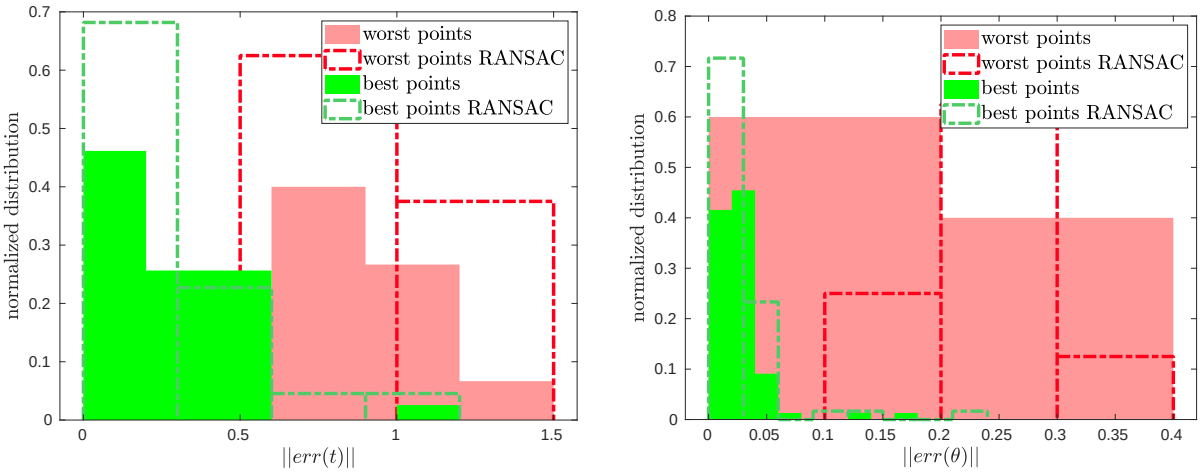
Comme dans le chapitre 5, la pose de la caméra n'est pas connue. Toutefois, l'approche de la rétropropagation permet de revenir à l'état initial permettant de caractériser la vérité terrain du mouvement de la caméra par la transformation ${}_{k+1}T_{groundtruth} = \begin{bmatrix} \mathcal{I}_{3\times3} & \mathbb{0}_{3\times1} \end{bmatrix}$.

Par ailleurs, la transformation prédite ${}_{k}T_{estimated}$ (5.13) en $3D$ est évaluée par stéréovision lors de la détection initiale de Harris, ce qui permet de trianguler les points de $2D$ en $3D$. L'erreur sur la transformation rigide peut ainsi être calculée avec :

$$ {}_{k+1}T_{err} = {}_{k+1}T_{groundtruth} \cdot {}_{k+1}T^{-1}_{estimated} \tag{7.12} $$

où ${}_{k+1}T_{err}$ définit l'erreur de transformation rigide à l'instant $k+1$.

Les meilleurs points (*i.e. best points*) sont sélectionnés à partir du critère $\sqrt{tr(\mathbf{\Sigma}_i)} \leq 0.5 * T$, $i \in |[0; s]|$ et les pires points (*i.e. worst points*) vérifient $\sqrt{tr(\mathbf{\Sigma}_i)} > 0.5 * T$. Les résultats sont présentés Fig. 7.10.



(a) (b)

FIGURE 7.12 – Impact sur l'estimation du mouvement - (a) Erreur sur la translation (en m) ; (b) Erreur sur la rotation (en deg).

Les résultats (Fig. 7.10) montre une erreur moyenne de 0.3017 m en translation pour les meilleurs points et 0.7604 pour les pires points. Ces métriques sont comparables à celles issues de RANSAC avec une erreur moyenne en translation de 0.3017 m pour les meilleurs points et 0.8521 pour les pires (Fig. 7.10a). La sélection des meilleurs ayant une meilleure incertitude sur sa position met en évidence une erreur moindre lors de l'estimation de la pose d'un système mobile et montre de meilleurs résultats. Ces résultats sont également visibles sur les erreurs de rotation (Fig. 7.10b) avec une erreur moyenne de 0.0298 deg pour les meilleurs

points et 0.1627 deg pour les pires ; avec RANSAC on estime une erreur moyenne de 0.0272 deg pour les meilleurs points et 0.2321 deg pour les pires.

7.4 Séquence à taille variable

Les RNN sont des modèles puissants. Leurs forces résident en la capacité de traiter les données en différé pour une séquence donnée. En particulier, la conception d'un réseau en récurrence RNN rend possible l'utilisation de séquence à taille variable en entrée comme en sortie du réseau. Dans notre cas, nous avons définis un RNN *many-to-one* associant une séquence d'entrée à une prédiction en sortie. Cette caractéristique des RNN intégre l'évolution dynamique d'un point dans le temps : elle permet de suivre l'évolution d'un point d'intérêt suivi indépendamment de l'âge de ce point.

En pratique, l'apprentissage des données est réalisé par paquets (*i.e.* par batch) contraignant les entrées d'un même lot de données à avoir une même taille. Parmi les solutions existantes populaires, deux approches se distinguent :

— Uniformiser les séquences d'entrées
— Traiter les données avec un batch= 1

7.4.1 Uniformisation des données

La première approche consiste à uniformiser les données pour retrouver un format de données ayant des séquences de taille fixe. Une première solution repose sur le *zero padding*, ou l'ajout de zéros en français. Il s'agit de sélectionner la taille de la séquence temporelle la plus longue parmi le jeu de données d'entrainement et définir cette taille T comme la dimension des séquences pour tous les échantillons $i \in s$. Puis de compléter toutes les séquences dont la longueur est inférieure à T avec des entrées nulles (*i.e.* des matrices de zéros), tel qu'illustré Fig. 7.13.

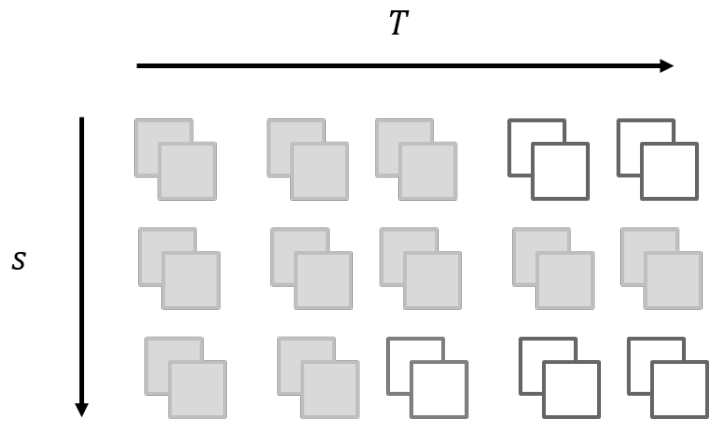


FIGURE 7.13 – Séquence à taille variable à partir de *zero padding*

7.4.2 Utilisation d'un batch unitaire

L'utilisation d'un batch unitaire consiste à traiter chaque séquence $i \in s$ une à une. Cette approche permet de se libérer de la contrainte du format des séquences de données d'entrée. Néanmoins, il rend l'apprentissage bien moins stable, puisque le traitement des données par lot permet la stabilité de l'apprentissage du modèle. Par ailleurs, le traitement des données en mode d'inférence, comme au cours de la phase d'apprentissage, est fortement impacté.

> ***Nota Bene*** La mise en place d'un système intégrant des séquences à taille variable dépend uniquement du format des données d'entrée et n'impacte pas l'architecture DUNE-M. La structure de DUNE-M, présentée Fig. 7.7, est conçue pour accueillir des données à taille variable ainsi que des séquences à taille fixe, comme présentée § 7.3.

7.5 Discussion

On peut remarquer la différence entre les résultats de KITTI avec DUNE (Fig. 5.16) et de KITTI avec DUNE-M (Fig. 7.12). En effet, on observe un facteur 10 sur l'erreur d'estimation du mouvement issue des meileurs points (*i.e. best points*) et des pires points (*i.e. worst points*) sélectionnés. La limite de l'approche d'évaluation de l'erreur à partir d'une retropropagation est fixée par la taille de la fenêtre temporelle pour laquelle la dynamique de l'erreur est supposée uniformément répartie. Cette hypothèse peut être vraie dans un environnement statique ou quasi-statique, mais montre ces failles dans un environnement dynamique. Toutefois, entre deux instants k et $k+1$, elle reste raisonnable. En conséquence, les résultats issus de KITTI avec DUNE (Fig. 5.16) admettent une plus faible erreur.

À l'inverse des résultats sur KITTI, le jeu de données SYNTHIA a été construit de façon indépendante à la taille de la séquence temporelle. Ainsi, l'erreur sur l'estimation de la position entre DUNE (Fig. 5.13) et DUNE-M (Fig. 7.10) est comparable.

Les avantages et les inconvénients des deux architectures DUNE et DUNE-M sont analysés Tab. 7.6. Cette analyse est réalisée à partir des critères de flexibilité des données d'entrée, d'adaptablité de la structure du RdN, du coût de calcul, de l'exploitation des données passées et de stabilité du RdN au cours de l'apprentissage.

La structure de DUNE-M offre l'utilisation de données avec des séquences à taille variable, la rendant particulièrement flexible. Elle exploite les positions passées d'un point d'intérêt pour prédir une meilleure estimation d'incertitude de l'état courant et converge en quelques epoch (~ 50). DUNE-M montre de nombreux avantages, mais reste toutefois plus coûteux en calculs. Par ailleurs, la taille des séquences des données d'entrée est contrainte par l'hypothèse d'une évolution uniforme de l'erreur pour une application sur des données réelles (*i.e.* mesure de l'erreur par retropopagation).

Critères	DUNE-M	DUNE
Flexibilité	Applicable à des entrées de taille variable	Applicable à des entrées de taille fixe
Adaptabilité	Architecture indépendante de la taille des séquences d'entrée	Architecture dépendante de la taille des entrées
Coût	-	+
Mémoire	Les états antérieurs sont intégrés	Les états antérieurs ne sont pas pris en compte
Hypothèse	Évolution de l'erreur uniformément répartie sur $T > 2$	Évolution de l'erreur uniformément répartie entre t et $t + 1$
Stabilité	Stable pendant l'apprentissage et rapide dans la convergence	Stable au cours de l'apprentissage, mais nécessite 2 fois plus d'epoch pour converger à un modèle

Table 7.6 – Comparaison DUNE-M et DUNE

7.6 Conclusion

Nous avons présenté une méthode de RNN qui produit des estimations de l'incertitude sur la position des points d'intérêt suivis par le tracker KLT intégrant des corrélations temporelles avec les positions antérieures des points. Nous avons montré qu'il est possible de capturer l'évolution temporelle dynamique d'un point suivi. Le modèle prédictif DUNE-M a été comparé, notamment à DUNE présentée chapitre 5, et validé par deux métriques dédiées.

Tout comme DUNE (chapitre 6), l'estimateur DUNE-M peut également être intégré à un système VINS. En particulier, le format de données avec des séquences à taille variable offre plus de souplesse dans le choix de la sélection de points d'intérêt, ce qui constitue la plus grande force de l'architecture mise en place avec DUNE-M. Par ailleurs, les covariances issues de DUNE-M permettent d'estimer précisément l'incertitude sur la position d'un point et évite de cumuler les T covariances prédites entre deux instants.

En conclusion, DUNE-M reprend le fonctionnement et les performances de DUNE. L'architecture offre d'avantage de flexibilité et une plus grande précision sur des séquences de données.

Troisième partie

Conclusion

Chapitre 8

Conclusion et perspectives

Nous n'arrêtons jamais d'explorer, et le terme de toute exploration sera le retour au point de départ.

Thomas Stearns Eliot

Dans la vie, rien n'est à craindre, tout est à comprendre.

Marie Curie

8.1 Synthèse des contributions

Cette thèse caractérise et évalue l'incertitude des mesures visuelles observées dans une image dans le contexte de la navigation. L'objectif était de déterminer la précision des points d'intérêt suivis afin d'améliorer la navigation d'un système mobile, en particulier lors d'un couplage serré des données avec une centrale inertielle. Les motivations principales de ce travail étaient de proposer une solution pour garantir une localisation précise à partir d'un système mobile basé sur de la vision, pour avoir la possibilité de le coupler par la suite avec d'autres capteurs (*i.e.* IMU). Les apports sont les suivantes :

- caractériser chaque point d'intérêt individuellement et indépendamment les uns des autres pour s'affranchir d'une erreur uniformément répartie sur l'ensemble de chaque image ;
- prendre en compte le niveau de bruit dans l'image, la dynamique de la scène, et l'environnement observé ;
- sélectionner les points les plus précis pour l'estimation de la pose avec une incertitude adaptée ;
- améliorer la disponibilité d'observations même dans le cas d'une image partiellement bruitée.

Nous avons proposé une méthode de caractérisation de l'incertitude sur la position des points d'intérêt. En effet, l'incertitude n'est pas directement observable, il est donc nécessaire de l'estimer à travers une métrique mesurable. Nous avons montré que l'erreur d'observation, caractérisée par l'écart entre le point observé et la vérité terrain, permet d'estimer une matrice de covariance adaptée à chaque observation.

Dans un second temps, cette caractérisation de l'incertitude a été intégrée dans un estimateur basé sur de l'apprentissage profond. En effet, il n'existe pas de modèle analytique englobant

toutes les sources possibles d'incertitude, nous appliquons donc une technique par apprentissage. La solution proposée a été évaluée sur des données de synthèse ainsi que sur des données réelles. Une validation géométrique a permi de confirmer le concept proposé.

Puis, les matrices de covariance adaptées à chaque point d'intérêt observé issues de l'estimateur d'incertitude ont été intégrées dans un système de navigation complet qui intègreles données visuelles et inertielles selon un shéma d'hybridation serrée. La solution proposée est particulièrement performante dans des scénarios dynamiques et bruités, là où l'hypothèse d'une covariance constante pour toutes les observations est peu adaptée. La sélection active des meilleurs points à partir des incertitudes produites par notre estimateur améliore encore l'estimation de la pose du système mobile.

Enfin, une amélioration de notre estimateur prenant en compte l'évolution temporelle d'un point d'intérêt est proposée : elle offre la possibilité d'avoir des pistes de suivi de points de différentes tailles, permettant une estimation précise de la précision d'un point en fonction de tous ses états antérieurs.

8.2 Discussions et perspectives

Notre dernier axe de recherche nécessiterait davantage d'analyse pour évaluer l'impact de l'ajout des états antérieurs. En effet, l'impact et les limites des pistes de suivi de point à taille variable n'a pas été comparé à des pistes de taille fixe. L'intégration de l'estimateur dans un système de navigation complet montrerait le bénéfice direct de ce nouveau modèle sur l'estimation de la pose d'un système mobile.

En parallèle, plusieurs études seraient intéressantes à envisager, notamment l'exportation et l'intégration de nos estimateurs dans d'autres applications (*i.e.* des applications de réalité augmentée, ou encore de l'*eye tracking*, en français *suivi du regard*). En effet, la caractérisation locale de l'incertitude des observations visuelles peut permettre d'améliorer la précision des cartes en trois dimension, ou tout application exploitant des points d'intérêt. Par ailleurs, il serait également pertinent d'appliquer nos estimateurs à d'autres méthodes de mise en correspondance et de suivi des points d'intérêt (*e.g.* SIFT).

Plusieurs axes de recherche pourraient permettre d'améliorer l'estimation de la précision de la pose d'un système mobile :

- Caractériser de façon précise l'incertitude de détection, indispensable pour une utilisation répétée du détecteur de Harris, notamment lors de méthode de matching de points.
- Intégrer les données inertielles en entrée de notre estimateur pour améliorer la précision des matrices de covariance produites. En effet, les données inertielles indiquent la dynamique du mouvement globale et apportent une information sur le bruit susceptible d'affecter l'image.
- La sélection active des points en fonction de leur précision est fixée avec un seuil. En fonction de l'évolution dynamique d'un scénario, ce seuil fixe peut être limitant. La classification des points avec des méthodes d'apprentissages peut permettre de surmonter ce problème.

Enfin, un modèle d'apprentissage auto-supervisé serait une solution élégante pour améliorer la mesure de l'erreur, notamment sur des données réelles. En effet, la solution proposée de mesure de l'erreur est limitée à une fenêtre temporelle relativement petite, notamment pour des scénarios dynamiques, puisqu'elle considère que l'erreur est uniformément répartie dans le temps. Un modèle estimant une carte de profondeur dense et précise, avec une incertitude connue, permettrait de disposer de la position des amers 3D. Dès lors, par reprojection la mesure de la vérité est disponible et s'affranchit de l'hypothèse d'une fenêtre temporelle dans le cas d'une application sur des données réelles. Des solutions performantes [Li+20] permettent déjà de construire des cartes denses à partir d'images réelles grâce à des modèles d'apprentissages de compression puis décompression de l'image. Une étude en amont de caractérisation de la précision des cartes de profondeur serait toutefois nécessaire.

Bibliographie

[AA19] Afshine Amidi et Shervine Amidi. *Recurrent Neural Networks cheatsheet.* `https://stanford.edu/~shervine/teaching/cs-230/cheatsheet-recurrent-neural-networks`. 2019 (cf. p. 107).

[Aqe+16] Mohammad OA Aqel et al. "Review of visual odometry : types, approaches, challenges, and applications". In : *SpringerPlus* 5.1 (2016), p. 1-26 (cf. p. 9, 10).

[AS11] Pablo F Alcantarilla et T Solutions. "Fast explicit diffusion for accelerated features in nonlinear scale spaces". In : *IEEE Trans. Patt. Anal. Mach. Intell* 34.7 (2011), p. 1281-1298 (cf. p. 19).

[AWD10] Ali Almagbile, Jinling Wang et Weidong Ding. "Evaluating the performances of adaptive Kalman filter methods in GPS/INS integration". In : *Journal of Global Positioning Systems* 9.1 (2010), p. 33-40 (cf. p. 35, 86).

[BCL14] Yannick Barthe, Michel Callon et Pierre Lascoumes. *Agir dans un monde incertain.* 2014 (cf. p. 26).

[BK81] D. Lucas Bruce et Takeo Kanade. "An Iterative Image Registration Technique with an Application to Stereo VIsion". In : 1981 (cf. p. 23, 25, 33, 41, 51, 63, 90, 98).

[Bou] Jean-Yves Bouguet. "Pyramidal Implementation of the Lucas Kanade Feature Tracker Description of the algorithm". en. In : (), p. 9 (cf. p. 25).

[Bra00] G. Bradski. "The OpenCV Library". In : *Dr. Dobb's Journal of Software Tools* (2000) (cf. p. 53).

[Bro58] Duane C Brown. *A solution to the general problem of multiple station analytical stereotriangulation.* D. Brown Associates, Incorporated, 1958 (cf. p. 3, 84).

[BS06] Christian Beder et Richard Steffen. "Determining an initial image pair for fixing the scale of a 3d reconstruction from an image sequence". In : *Joint Pattern Recognition Symposium.* Springer. 2006, p. 657-666 (cf. p. 36).

[BTG06] Herbert Bay, Tinne Tuytelaars et Luc Van Gool. "Surf : Speeded up robust features". In : *European conference on computer vision.* Springer. 2006, p. 404-417 (cf. p. 19, 111).

[Bur16] Wilhelm Burger. "Zhang's camera calibration algorithm : in-depth tutorial and implementation". In : *HGB16-05* (2016), p. 1-6 (cf. p. 17).

[Bur+16] Michael Burri et al. "The EuRoC micro aerial vehicle datasets". In : *The International Journal of Robotics Research* (2016). eprint : `http://ijr.sagepub.com/content/early/2016/01/21/0278364915620033.full.pdf+html` (cf. p. 50, 83, 88, 95, 96).

[Cad+16] Cesar Cadena et al. "Past, Present, and Future of Simultaneous Localization and Mapping : Toward the Robust-Perception Age". en. In : *IEEE Transactions on Robotics* 32.6 (déc. 2016), p. 1309-1332 (cf. p. 10).

[Cal+10] Michael Calonder et al. “Brief : Binary robust independent elementary features”. In : *European conference on computer vision*. Springer. 2010, p. 778-792 (cf. p. 19).

[Che+18] Zhaozhong Chen et al. “Weak in the NEES ? : Auto-tuning Kalman filters with Bayesian optimization”. In : *2018 21st International Conference on Information Fusion (FUSION)*. IEEE. 2018, p. 1072-1079 (cf. p. 98).

[Chi+16] Hsiang-Jen Chien et al. “When to use what feature ? SIFT, SURF, ORB, or A-KAZE features for monocular visual odometry”. en. In : *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. Palmerston North, New Zealand : IEEE, nov. 2016, p. 1-6 (cf. p. 19).

[Cho+14] Kyunghyun Cho et al. “Learning phrase representations using RNN encoder-decoder for statistical machine translation”. In : *arXiv preprint arXiv :1406.1078* (2014) (cf. p. 109).

[CPY15] Sunglok Choi, Jaehyun Park et Wonpil Yu. “Simplified epipolar geometry for real-time monocular visual odometry on roads”. In : *International Journal of Control, Automation and Systems* 13.6 (2015), p. 1454-1464 (cf. p. 11).

[Cui+22] Yutao Cui et al. “MixFormer : End-to-End Tracking with Iterative Mixed Attention”. In : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, p. 13608-13618 (cf. p. 34, 41, 63).

[Dav03] Andrew J Davison. “Real-time simultaneous localisation and mapping with a single camera”. In : *Computer Vision, IEEE International Conference on*. T. 3. IEEE Computer Society. 2003, p. 1403-1403 (cf. p. 40).

[DML20] Andrea De Maio et Simon Lacroix. “Simultaneously Learning Corrections and Error Models for Geometry-Based Visual Odometry Methods”. In : *IEEE Robotics and Automation Letters* 5.4 (2020), p. 6536-6543 (cf. p. 34, 66, 85).

[DMR18] Daniel DeTone, Tomasz Malisiewicz et Andrew Rabinovich. “Superpoint : Self-supervised interest point detection and description”. In : *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2018, p. 224-236 (cf. p. 34).

[Doz16] Timothy Dozat. “Incorporating nesterov momentum into adam”. In : (2016) (cf. p. 62).

[DSMT05] Luigi Di Stefano, Stefano Mattoccia et Federico Tombari. “ZNCC-based template matching using bounded partial correlation”. In : *Pattern recognition letters* 26.14 (2005), p. 2129-2134 (cf. p. 22).

[EKC17] Jakob Engel, Vladlen Koltun et Daniel Cremers. “Direct sparse odometry”. In : *IEEE transactions on pattern analysis and machine intelligence* 40.3 (2017), p. 611-625 (cf. p. 10).

[Elm+20] Ahmed Elmoogy et al. “Surf-lstm : A descriptor enhanced recurrent neural network for indoor localization”. In : *2020 IEEE 92nd Vehicular Technology Conference (VTC2020-Fall)*. IEEE. 2020, p. 1-5 (cf. p. 35, 109, 111, 114).

[ESL21] Naser El-Sheimy et You Li. “Indoor navigation : State of the art and future trends”. In : *Satellite Navigation* 2.1 (2021), p. 1-23 (cf. p. 9).

[FB81] Martin A Fischler et Robert C Bolles. “Random sample consensus : a paradigm for model fitting with applications to image analysis and automated cartography”. In : *Communications of the ACM* 24.6 (1981), p. 381-395 (cf. p. 30, 33, 42, 72, 77).

[FPS14] Christian Forster, Matia Pizzoli et Davide Scaramuzza. “SVO : Fast semi-direct monocular visual odometry”. en. In : *2014 IEEE International Conference on Robotics and Automation (ICRA)*. Hong Kong, China : IEEE, mai 2014, p. 15-22 (cf. p. 10).

[FS97] Hany Farid et Eero P. Simoncelli. “Optimally rotation-equivariant directional derivative kernels”. en. In : *Computer Analysis of Images and Patterns*. Sous la dir. de Gerhard Goos et al. T. 1296. Series Title : Lecture Notes in Computer Science. Berlin, Heidelberg : Springer Berlin Heidelberg, 1997, p. 207-214 (cf. p. 19).

[GBC16] Ian Goodfellow, Yoshua Bengio et Aaron Courville. *Deep Learning*. `http://www.deeplearningbook.org`. MIT Press, 2016 (cf. p. 59, 61, 62, 107).

[Gei+13] Andreas Geiger et al. “Vision meets Robotics : The KITTI Dataset”. In : *International Journal of Robotics Research (IJRR)* (2013) (cf. p. 50, 55, 64, 78, 120).

[Gen+20] Patrick Geneva et al. “OpenVINS : A Research Platform for Visual-Inertial Estimation”. In : *Proc. of the IEEE International Conference on Robotics and Automation*. Paris, France, 2020 (cf. p. 6, 83, 85, 88, 89, 98).

[GLU12] Andreas Geiger, Philip Lenz et Raquel Urtasun. “Are we ready for Autonomous Driving ? The KITTI Vision Benchmark Suite”. In : *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012 (cf. p. 55, 57, 59, 88, 107).

[GVL13] Gene H Golub et Charles F Van Loan. *Matrix computations*. JHU press, 2013 (cf. p. 90).

[Han+15] Xufeng Han et al. “Matchnet : Unifying feature and metric learning for patch-based matching”. In : *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, p. 3279-3286 (cf. p. 34, 42, 64).

[Han+16] Ankur Handa et al. “gvnn : Neural network library for geometric computer vision”. In : *European Conference on Computer Vision*. Springer. 2016, p. 67-82 (cf. p. 62, 66).

[HJA10] Haule Strasdat, J. M. M. Montiel et Andrew J. Davison. “Real-time Monocular SLAM : Why Filter ?” In : IEEE Intl. Conf. on Robotics et Automation (ICRA), 2010 (cf. p. 29, 42).

[HK15] Humphrey Hu et George Kantor. “Parametric covariance prediction for heteroscedastic noise”. In : *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE. 2015, p. 3052-3057 (cf. p. 46, 47).

[HMS] S Heymann, K Müller et A Smolic. “SIFT Implementation and Optimization for General-Purpose GPU”. en. In : (), p. 6 (cf. p. 51).

[HS88] C. Harris et M. Stephens. “A Combined Corner and Edge Detector”. In : *Procedings of the Alvey Vision Conference 1988*. 1988 (cf. p. 19, 20, 51, 68, 69).

[HS97] Sepp Hochreiter et Jürgen Schmidhuber. “Long short-term memory”. In : *Neural computation* 9.8 (1997), p. 1735-1780 (cf. p. 35, 109).

[Hua19] Guoquan Huang. “Visual-inertial navigation : A concise review”. In : (2019), p. 9572-9582 (cf. p. 3, 34, 41, 83, 84).

[HZ03] Richard Hartley et Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003 (cf. p. 12).

[IA99] Michal Irani et P Anandan. “About direct methods”. In : *International Workshop on Vision Algorithms*. Springer. 1999, p. 267-277 (cf. p. 10).

[Joh86] John CANNY. “A computational Approach to Edge Detection”. In : t. VOL. PAMI-8. IEEE TRANSACTIONS ON PATTERN ANALYSIS et MACHINE INTELLIGENCE, nov. 1986 (cf. p. 19).

[JT94] Jianbo Shi et Tomasi. “Good features to track”. en. In : *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition CVPR-94*. Seattle, WA, USA : IEEE Comput. Soc. Press, 1994, p. 593-600 (cf. p. 19).

[KB13] Nal Kalchbrenner et Phil Blunsom. “Recurrent continuous translation models”. In : *Proceedings of the 2013 conference on empirical methods in natural language processing*. 2013, p. 1700-1709 (cf. p. 108).

[KB14] Diederik P Kingma et Jimmy Ba. “Adam : A method for stochastic optimization”. In : *arXiv preprint arXiv :1412.6980* (2014) (cf. p. 62).

[Ken19] Alex Guy Kendall. “Geometry and uncertainty in deep learning for computer vision”. Thèse de doct. University of Cambridge, UK, 2019 (cf. p. 34).

[KGC15] Alex Kendall, Matthew Grimes et Roberto Cipolla. “Convolutional networks for real-time 6-DOF camera relocalization. CoRR abs/1505.07427 (2015)”. In : *arXiv preprint arxiv :1505.07427* (2015) (cf. p. 35).

[Kit+11] Bernd Manfred Kitt et al. “Monocular visual odometry using a planar road model to solve scale ambiguity”. In : (2011) (cf. p. 10).

[KK01] Y. Kanazawa et K. Kanatani. “Do we really have to consider covariance matrices for image features ?” In : *Proceedings Eighth IEEE International Conference on Computer Vision. ICCV 2001*. T. 2. 2001, 301-306 vol.2 (cf. p. 33, 41, 42, 63, 89, 93, 98).

[KO60] Rudolph Emil Kalman et Others. “A new approach to linear filtering and prediction problems”. In : *Journal of basic Engineering* 82.1 (1960), p. 35-45 (cf. p. 2, 84, 85).

[Kum+11] Rainer Kummerle et al. “G^2o : A general framework for graph optimization”. en. In : *2011 IEEE International Conference on Robotics and Automation*. Shanghai, China : IEEE, mai 2011, p. 3607-3613 (cf. p. 84).

[LBH15] Yann LeCun, Yoshua Bengio et Geoffrey Hinton. “Deep learning”. In : *nature* 521.7553 (2015), p. 436-444 (cf. p. 34, 62).

[LDW91] J.J. Leonard et H.F. Durrant-Whyte. “Simultaneous map building and localization for an autonomous mobile robot”. In : (1991), 1442-1447 vol.3 (cf. p. 3).

[LeC+89] Yann LeCun et al. “Backpropagation applied to handwritten zip code recognition”. In : *Neural computation* 1.4 (1989), p. 541-551 (cf. p. 60).

[Leu+13] Stefan Leutenegger et al. “Keyframe-based visual-inertial slam using nonlinear optimization”. In : *Proceedings of Robotis Science and Systems (RSS) 2013* (2013) (cf. p. 35).

[Leu+15] Stefan Leutenegger et al. “Keyframe-based visual–inertial odometry using nonlinear optimization”. In : *The International Journal of Robotics Research* 34.3 (2015), p. 314-334 (cf. p. 35).

[Li+20] Shunkai Li et al. “Self-supervised deep visual odometry with online adaptation”. In : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, p. 6339-6348 (cf. p. 131).

[Li+22] Shaopeng Li et al. “Overview of deep learning application on visual SLAM”. In : *Displays* (2022), p. 102298 (cf. p. 34, 85).

[Liu+17] Qingshan Liu et al. “Bidirectional-convolutional LSTM based spectral-spatial feature learning for hyperspectral image classification”. In : *Remote Sensing* 9.12 (2017), p. 1330 (cf. p. 109, 114).

[Liu+18] Katherine Liu et al. “Deep Inference for Covariance Estimation : Learning Gaussian Noise Models for State Estimation”. In : *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, p. 1436-1443 (cf. p. 34, 47, 66, 85).

[LMNF09] Vincent Lepetit, Francesc Moreno-Noguer et Pascal Fua. “EPnP : An accurate O(n) solution to the PnP problem”. In : *International Journal of Computer Vision* 81 (fév. 2009) (cf. p. 77, 119).

[Low99] D.G. Lowe. “Object recognition from local scale-invariant features”. en. In : *Proceedings of the Seventh IEEE International Conference on Computer Vision*. Kerkyra, Greece : IEEE, 1999, 1150-1157 vol.2 (cf. p. 19).

[MAMT15] Raul Mur-Artal, Jose Maria Martinez Montiel et Juan D Tardos. “ORB-SLAM : a versatile and accurate monocular SLAM system”. In : *IEEE transactions on robotics* 31.5 (2015), p. 1147-1163 (cf. p. 29, 35, 40-42, 49, 51, 63, 84, 89, 93).

[MAT17] Raúl Mur-Artal et Juan D Tardós. “Visual-inertial monocular SLAM with map reuse”. In : *IEEE Robotics and Automation Letters* 2.2 (2017), p. 796-803 (cf. p. 35).

[MGL18] D. Mo, A. Ghiglione et S. Li. *Livre de Mozi*. Collection Études d’histoire et de culture chinoises. Presses de l’Université Laval, 2018 (cf. p. 12).

[Moh+19] Sherif A. S. Mohamed et al. “A Survey on Odometry for Autonomous Navigation Systems”. en. In : *IEEE Access* 7 (2019), p. 97466-97486 (cf. p. 34).

[Mor80] Hans Peter Moravec. “Obstacle Avoidance and Navigation in the Real World by a Seeing Robot Rover”. AAI8024717. Thèse de doct. Stanford, CA, USA, 1980 (cf. p. 3).

[MR07a] Christopher Mei et Patrick Rives. “Single view point omnidirectional camera calibration from planar grids”. In : *Proceedings 2007 IEEE International Conference on Robotics and Automation*. IEEE. 2007, p. 3945-3950 (cf. p. 17).

[MR07b] Anastasios I. Mourikis et Stergios I. Roumeliotis. “A Multi-State Constraint Kalman Filter for Vision-aided Inertial Navigation”. In : *Proceedings 2007 IEEE International Conference on Robotics and Automation*. 2007, p. 3565-3572 (cf. p. 35, 40, 84, 87-91).

[MS99] AH Mohamed et KP Schwarz. “Adaptive Kalman filtering for INS/GPS”. In : *Journal of geodesy* 73.4 (1999), p. 193-203 (cf. p. 35, 86).

[NVB11] Navid Nourani-Vatani et Paulo Vinicius Koerich Borges. “Correlation-based visual odometry for ground vehicles”. In : *Journal of Field Robotics* 28.5 (2011), p. 742-768 (cf. p. 10).

[Ola15] Christopher Olah. *Understanding LSTM Networks*. `https://colah.github.io/posts/2015-08-Understanding-LSTMs/`. 2015 (cf. p. 107).

[Ort07] Joan Solà Ortega. “Towards visual localization, mapping and moving objects tracking by a mobile robot : a geometric and probabilistic approach”. In : *Ecole Doctorale Systemes, Docteur de lInstitut National Politechnique de Toulouse, PhD theses* (2007) (cf. p. 87).

[Per+15] Amila Perera et al. “Feature Point Tracking Algorithm Evaluation for Augmented Reality in Handheld Devices”. In : *Computer Vision - ACCV 2014 Workshops*. Springer International Publishing, 2015, p. 275-288 (cf. p. 22, 51, 63).

[PK17] Valentin Peretroukhin et Jonathan Kelly. “Dpc-net : Deep pose correction for visual localization”. In : *IEEE Robotics and Automation Letters* 3.3 (2017), p. 2424-2431 (cf. p. 34, 85).

[PS21] Mostafa Parchami et Saif Iftekar Sayed. “Deep Feature Tracker : A Novel Application for Deep Convolutional Neural Networks”. In : *arXiv preprint arXiv :2108.00105* (2021) (cf. p. 34, 42, 64).

[Qin+22] Jiangying Qin et al. “A Survey on Visual Navigation and Positioning for Autonomous UUVs”. In : *Remote Sensing* 14.15 (2022), p. 3794 (cf. p. 34, 84).

[RD06] Edward Rosten et Tom Drummond. “Machine Learning for High-Speed Corner Detection”. en. In : *Computer Vision – ECCV 2006*. Sous la dir. d’Aleš Leonardis, Horst Bischof et Axel Pinz. T. 3951. Berlin, Heidelberg : Springer Berlin Heidelberg, 2006, p. 430-443 (cf. p. 19).

[RHW85] David E Rumelhart, Geoffrey E Hinton et Ronald J Williams. *Learning internal representations by error propagation*. Rapp. tech. California Univ San Diego La Jolla Inst for Cognitive Science, 1985 (cf. p. 60, 107).

[Rib04] Maria Isabel Ribeiro. “Kalman and extended kalman filters : Concept, derivation and properties”. In : *Institute for Systems and Robotics* 43 (2004), p. 46 (cf. p. 86).

[RJM02] Stergios I Roumeliotis, Andrew E Johnson et James F Montgomery. “Augmenting inertial navigation with image-based motion estimation”. In : *Proceedings 2002 IEEE International Conference on Robotics and Automation (Cat. No. 02CH37292)*. T. 4. IEEE. 2002, p. 4326-4333 (cf. p. 87).

[Ros+16] German Ros et al. “The SYNTHIA Dataset : A Large Collection of Synthetic Images for Semantic Segmentation of Urban Scenes”. In : *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. Juin 2016 (cf. p. 49, 52, 54, 59, 64, 72, 107, 116).

[Rub+11] Ethan Rublee et al. “ORB : An efficient alternative to SIFT or SURF”. en. In : *2011 International Conference on Computer Vision*. Barcelona, Spain : IEEE, nov. 2011, p. 2564-2571 (cf. p. 10, 19).

[San+17] Wojciech Sankowski et al. “Estimation of measurement uncertainty in stereo vision system”. In : *Image and Vision Computing* 61 (2017), p. 70-81 (cf. p. 36).

[SF11] Davide Scaramuzza et Friedrich Fraundorfer. “Visual Odometry [Tutorial]”. In : *IEEE Robotics Automation Magazine* 18.4 (2011), p. 80-92 (cf. p. 9, 10, 39).

[Sob10] D. Sobel. *Longitude : The True Story of a Lone Genius Who Solved the Greatest Scientific Problem of His Time*. Bloomsbury Publishing, 2010 (cf. p. 1).

[Sri+14] Nitish Srivastava et al. “Dropout : a simple way to prevent neural networks from overfitting”. In : *The journal of machine learning research* 15.1 (2014), p. 1929-1958 (cf. p. 62, 63).

[Str12] Hauke Strasdat. “Local accuracy and global consistency for efficient visual SLAM”. Thèse de doct. Department of Computing, Imperial College London, 2012 (cf. p. 29, 42).

[Stu97] Peter Sturm. “Vision 3D non calibrée : contributions à la reconstruction projective et étude des mouvements critiques pour l’auto-calibrage”. Thèse de doct. Institut National Polytechnique de Grenoble-INPG, 1997 (cf. p. 12).

[Sze90] Richard Szeliski. “Bayesian modeling of uncertainty in low-level vision”. In : *International Journal of Computer Vision* 5.3 (1990), p. 271-301 (cf. p. 34, 88, 95).

[Tak07] Takeshi Takahashi. “2D localization of outdoor mobile robots using 3D laser range data”. In : *Robotics Institute Carnegie Mellon University Pittsburgh, Pennsylvania* 15213 (2007) (cf. p. 2, 9).

[Tan+22] Yang Tang et al. “Perception and Navigation in Autonomous Systems in the Era of Learning : A Survey”. In : *IEEE Transactions on Neural Networks and Learning Systems* (2022) (cf. p. 35, 84).

[Urf+06] Onay Urfalioglu et al. “Error Analysis of Camera Parameter Estimation based on Collinear Features”. In : *The 3rd Canadian Conference on Computer and Robot Vision (CRV'06)*. IEEE. 2006, p. 32-32 (cf. p. 33, 88).

[Wal+17] Florian Walch et al. “Image-based localization using lstms for structured feature correlation”. In : *Proceedings of the IEEE International Conference on Computer Vision*. 2017, p. 627-637 (cf. p. 34, 35, 111).

[Wan+18] Sen Wang et al. “End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks”. In : *The International Journal of Robotics Research* 37.4-5 (2018), p. 513-542 (cf. p. 34, 47, 63, 85).

[Wan+20] Shizhuang Wang et al. “Feature-based visual navigation integrity monitoring for urban autonomous platforms”. In : *Aerospace systems* 3.3 (2020), p. 167-179 (cf. p. 35, 88).

[Wer88] Paul J Werbos. “Generalization of backpropagation with application to a recurrent gas market model”. In : *Neural networks* 1.4 (1988), p. 339-356 (cf. p. 107).

[WM17] Xue Iuan Wong et Manoranjan Majji. “Uncertainty Quantification of Lucas Kanade Feature Track and Application to Visual Odometry”. In : *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*. 2017, p. 950-958 (cf. p. 33, 41, 63, 74, 75, 77, 80, 88, 96, 117, 118, 121, 122).

[Yan+20] Nan Yang et al. “D3vo : Deep depth, deep pose and deep uncertainty for monocular visual odometry”. In : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, p. 1281-1292 (cf. p. 34).

[Yan+22] Bin Yan et al. “Towards Grand Unification of Object Tracking”. In : *ECCV*. 2022 (cf. p. 34, 41, 63).

[YBHH15] Khalid Yousif, Alireza Bab-Hadiashar et Reza Hoseinnezhad. “An overview to visual odometry and visual SLAM : Applications to mobile robotics”. In : *Intelligent Industrial Systems* 1.4 (2015), p. 289-311 (cf. p. 9, 85).

[ZB+19] Javad Zolfaghari Bengar et al. “Temporal Coherence for Active Learning in Videos”. In : *The IEEE International Conference in Computer Vision, Workshops (ICCV Workshops)*. 2019 (cf. p. 72, 116).

[ZC21] Bingbing Zhuang et Manmohan Chandraker. “Fusing the old with the new : Learning relative camera pose with geometry-guided uncertainty”. In : *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2021, p. 32-42 (cf. p. 34).

[Zha00] Z. Zhang. “A flexible new technique for camera calibration”. In : *IEEE Transactions on Pattern Analysis and Machine Intelligence* 22.11 (2000), p. 1330-1334 (cf. p. 17, 41).

[Zha98] Zhengyou Zhang. “Determining the epipolar geometry and its uncertainty : A review”. In : *International journal of computer vision* 27.2 (1998), p. 161-195 (cf. p. 10, 84).

[Zhu+19a] Chen Zhu et al. “Feature error model for integrity of pattern-based visual positioning”. In : *Proceedings of the 32nd International Technical Meeting of the Satellite Division of The Institute of Navigation (ION GNSS+ 2019)*. 2019, p. 2254-2268 (cf. p. 33, 88).

[Zhu+19b] Chen Zhu et al. “Six degrees-of-freedom dilution of precision for integrity of camera-based localization”. In : *32nd International Technical Meeting of the Satellite Division of the Institute of Navigation, ION GNSS+ 2019*. 2019 (cf. p. 33).

[ZJM20] Chen Zhu, Mathieu Joerger et Michael Meurer. “Quantifying feature association error in camera-based positioning”. In : *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*. IEEE. 2020, p. 967-972 (cf. p. 33).

[ZMG22] Chen Zhu, Michael Meurer et Christoph Günther. “Integrity of Visual Navigation—Developments, Challenges, and Prospects”. In : *NAVIGATION : Journal of the Institute of Navigation* 69.2 (2022) (cf. p. 33, 84, 88).

[ZS18] Zichao Zhang et Davide Scaramuzza. “A Tutorial on Quantitative Trajectory Evaluation for Visual(-Inertial) Odometry”. en. In : *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. Madrid : IEEE, oct. 2018, p. 7244-7251 (cf. p. 97).

[ZSK14] Ji Zhang, Sanjiv Singh et George Kantor. “Robust monocular visual odometry for a ground vehicle in undulating terrain”. In : *Field and service robotics*. Springer. 2014, p. 311-326 (cf. p. 10).

[ZT17] Zhen Zhu et Clark Taylor. “Conservative uncertainty estimation in map-based vision-aided navigation”. In : *IEEE Transactions on Aerospace and Electronic Systems* 53.2 (2017), p. 941-949 (cf. p. 33, 88).

Résumé — Dans cette thèse, on s'intéresse à la caractérisation de l'incertiude de localisation des mesures optiques dans une image. Autrement dit, il s'agit d'estimer localement la précision de la position de chaque point d'intérêt observé et suivi au cours du temps. L'application envisagée est un système (VINS - Visual Inertial Navigation System) de navigation Inertie/Vision couplé par une méthode de filtrage. Une hypothèse populaire consiste à considérer l'incertitude fixe et égale de tous les points issus d'une même image, et plus généralement d'un même capteur pour une application donnée. Cette approximation est raisonnable dans un environnement statique ou quasi-statique, mais aura le rôle d'un filtre appliquée à un environnement dynamique. Dans ce cas, une partie de l'information utile est perdue induisant un manque d'information lorsque les images capturées sont bruitées, même partiellement. Une caractérisation précise de l'incertitude sur les mesures permet la pondération des observations, ainsi qu'une alimentation constante d'observations même bruitées à l'application visée. L'intérêt de qualifier la précision des mesures est de pouvoir appliquer un filtrage serré à la navigation et ainsi réduire le biais sur la localisation du système mobile. Pour réponde à cette problématique, nous proposons une méthode DUNE (Deep UNcertainty Estimation) d'estimation de la covariance pour chaque point d'intérêt à partir de la position des ces points. Notre approche, basée sur des techniques d'apprentissage profond, propose une modélisation d'erreur locale. Elle évalue en particulier le tracker KLT (Kanade Lucas Tomatsi). La méthode proposée a l'avantage de ne pas être dépendante du tracker KLT, et peut s'exporter à différentes méthodes de suivi de points (ORB, SIFT ..). Cette approche originale est évaluée à travers des métriques appropriées, une validation géométrique et une intégration dans un système de navigation VINS.

Mots clés : Incertitude, point d'intérêt, tracking, KLT, réseau de neurones, LSTM, navigation, VINS.

Abstract — In this thesis, we are interested in the characterization of the localization uncertainty of optical measurements in an image. In other words, the aim is to estimate locally the accuracy of the position of each point of interest observed and tracked over time. The application considered is a system (VINS - Visual Inertial Navigation System) of Inertial/Vision navigation coupled by a filtering method. A popular assumption is to consider the fixed and equal uncertainty of all points from the same image, and more generally from the same sensor for a given application. This approximation is reasonable in a static or quasi-static environment, but will have the role of a filter applied to a dynamic environment. In this case, part of the useful information is lost inducing a lack of information when the captured images are noisy, even partially. A precise characterization of the uncertainty on the measurements allows the weighting of the observations, as well as a constant supply of observations even noisy to the targeted application. The interest of qualifying the accuracy of the measurements is to be able to apply a tight filtering to the navigation and thus reduce the bias on the localization of the mobile system. To address this issue, we propose a DUNE (Deep UNcertainty Estimation) method to estimate the covariance for each point of interest from the position of these points. Our approach, based on deep learning techniques, proposes a local error model. It evaluates in particular the KLT (Kanade Lucas Tomatsi) tracker. The proposed method has the advantage of not being dependent on the KLT tracker, and can be exported to different point tracking methods (ORB, SIFT ..). This original approach is evaluated through appropriate metrics, geometric validation and integration into a VINS navigation system.

Keywords : Uncertainty, keypoints, tracking, KLT, neural network, LSTM, navigation, VINS.

GIPSA-LAB, Grenoble-INP, UGA
Grenoble