

FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO



Visual Odometry System Using Multi-Modal Information in Autonomous Driving Scenarios

Lucas Rebelo Agostinho

Mestrado em Engenharia Eletrotécnica e de Computadores

Supervisor: Prof. Andry Maykol Gomes Pinto

Co-Supervisor: Prof. Aníbal Castilho Coimbra de Matos

June 30, 2022

Resumo

Quer a indústria quer as agências governamentais têm mostrado, nos últimos anos, esforços crescentes no sentido de se caminhar em direção à completa autonomia dos veículos. Segurança, consumo eficiente de combustível e redução de tráfego são alguns dos benefícios associados. Dentro das várias áreas de investigação da condução autónoma, esta dissertação enquadra-se no campo da perceção do ambiente, que é de extrema importância para dotar o veículo com a informação necessária para tomar as devidas ações. Sendo esta uma área em que a segurança humana é crítica, importa que os métodos de perceção possuam um grau elevado de qualidade. O trabalho desenvolvido incide sobre técnicas de odometria visual, que são metodologias que conferem ao veículo a capacidade de se auto-localizar com recurso, por exemplo, a câmaras, LiDAR, RADAR, IMU, GPS, entre outros.

Esta dissertação visa explorar os benefícios de recorrer a diferentes sensores, como os mencionados acima, para construir um sistema de odometria o mais robusto possível. A multimodalidade de dados torna-se importante em cenários de falha individual (recolha de imagem ou disponibilidade de sinal satélite em túneis pouco iluminados, por exemplo), assim como para aumentar a confiança do sistema nas estimativas produzidas. Nesse sentido, foi desenvolvida uma arquitetura de fusão tardia baseada em redes neuronais, não existindo nenhuma abordagem semelhante na literatura atual. O objetivo é gerar a estimativa de movimento de máxima verosimilhança, a partir de estimativas previamente geradas por outras técnicas de odometria que recorrem a diferentes sensores. A rede proposta foi treinada com os dados recolhidos em bases de dados públicas compostos por imagens RGB, nuvens de pontos e medições IMU/GPS. O propósito da rede é mapear a dispersão estatística de cada sinal de entrada, através dos dados, por forma a produzir-se uma ponderação precisa dos dados de entrada. Nesse sentido, recorre-se, para além das estimativas de odometria, a dados estatísticos de cada sinal de entrada, que proporcionam à rede informação sobre a confiabilidade dos mesmos. Esta operação é feita em modo de janela deslizante. Para além disso, são propostas duas arquiteturas estendidas que visam capturar informação no domínio temporal, usando para isso um conjunto de dados de instantes anteriores. Estes modelos recorrem a estruturas LSTM e a uma convolução dos dados numa janela temporal.

Os resultados demonstraram o comportamento que se esperava da rede, nomeadamente uma melhoria de precisão na ordem dos 48.3% dos dados de entrada, em condições normais de trânsito. Devido à baixa representatividade das falhas nos sinais de entrada e ao número reduzido de sinais de entrada (3), a rede apresenta um comportamento não ótimo em situações bastante complexas em que existem múltiplos objetos dinâmicos em cena. Ainda assim, a robustez a falhas como espaços abertos, vegetação densa e passagens em túneis é verificada com bastante sucesso, demonstrando um erro 44.6% inferior relativamente aos sinais de entrada. Uma comparação com técnicas de odometria do estado da arte e com duas versões adaptadas do filtro de Kalman demonstrou uma performance muito competitiva e promissora das capacidades da rede. Por último, as redes com dimensão temporal provaram conseguir produzir as estimativas com a dinâmica mais próxima do real, comparativamente a todos os outros algoritmos avaliados.

Abstract

Both industry and government agencies have shown, in recent years, increasing efforts toward complete vehicle autonomy. Safety, efficient fuel consumption, and reduced traffic are some associated benefits. Within the various areas of autonomous driving research, this dissertation fits into the field of perception, which is extremely important to provide the vehicle with the necessary information to take the appropriate actions. As this is an area where human safety is critical, perception methods must meet the necessary quality. The work developed focuses on visual odometry techniques, which are methodologies that give the vehicle the ability to self-locate using, for example, cameras, LiDAR, RADAR, IMU, and GPS, among others.

This dissertation explores the benefits of using different sensors, like the ones mentioned above, to build an odometry system as robust as possible. The multi-modality of data becomes essential in single fault scenarios (image capturing or satellite signal availability in poorly lit tunnels, for example), as well as to increase system confidence in the estimates produced. In this sense, a late fusion architecture based on neural networks was developed, as there is no similar approach in the current literature. The objective is to generate the maximum likelihood motion estimate from previously generated estimates from odometry techniques of distinct modalities. The proposed network was trained with data collected in public databases composed of RGB images, point clouds, and IMU/GPS measurements. The purpose of the network is to map the statistical dispersion of each input signal from the data and produce an accurate weighting of the inputs. In addition to odometry estimates, statistical data of each input signal is used, which provides the network with information about its reliability. This operation is carried out in sliding window mode. Furthermore, two extended architectures are proposed to capture information in the temporal domain, using a set of data from previous instants. These models use LSTM structures and a multi-kernel convolution of the data in a time window.

The results demonstrated the expected behavior of the network, i.e., an improvement in accuracy of around 48.3% concerning the input signals under normal traffic conditions. Due to the low representation of signal failures in the dataset, and the reduced number of input signals (3), the network presents a non-optimal behavior in very complex situations where there are multiple dynamic objects in the scene. Even so, the robustness to faults such as open spaces, dense vegetation, and tunnel passages is verified quite successfully, demonstrating a 44.6% lower error compared to the input signals. A comparison with state-of-the-art odometry techniques and two adapted versions of the Kalman filter demonstrated a very competitive and promising performance of the network capabilities. Finally, the time-aware networks proved to produce estimates with the closest dynamics to the real vehicle kinematics compared to all other evaluated algorithms.

Agradecimentos

Ao meu orientador, Prof. Andry Pinto, quero agradecer não apenas pela mentoria durante a dissertação, mas pelo acompanhamento incansável, pela dedicação na partilha de conhecimento e pela incrível capacidade de extrair o verdadeiro potencial dos seus alunos, da qual agora se colhem os frutos. À minha co-orientadora Maria Inês Pereira, que tive o enorme prazer de conhecer, agradeço a pronta disponibilidade para ajudar em todas as ocasiões e a abertura para discutirmos e debatermos novas ideias a qualquer altura.

Deixo também um agradecimento aos meus amigos, especialmente ao André Sousa e à Sara Alves, cuja faculdade me deu a sorte dos nossos caminhos se cruzarem; ao David Maia, grande amigo desde que tenho memória e amigo até que um dia a memória me falte; e ao Nuno Ricardo, a pessoa com quem passei, sem dúvida alguma, mais horas nestes últimos cinco anos, mas cuja companhia não teria trocado por nada neste mundo (se o "nada neste mundo" for abaixo de 10 milhões de euros...).

Por último, e não menos, um enorme Obrigado à minha família, em especial à minha mãe e ao meu pai pelo apoio e por me aturarem, mas sobretudo pelo amor, pelos valores, pelo respeito e pelo exemplo que são para mim. Nunca nada será suficiente para retribuir.

Lucas Rebelo Agostinho

Official Acknowledgements

This work was carried out with the support of the host institution: INESC TEC and supervised at the institution by: Maria Inês Rodrigues Pereira.

This dissertation was supported by the European Structural and Investment Funds in the FEDER component, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project nº 047264; Funding Reference: POCI-01-0247-FEDER-047264].

Lucas Rebelo Agostinho

Contents

1	Introduction	1
1.1	Context	1
1.2	Motivation	2
1.3	Objectives	3
1.4	Contributions	4
1.5	Document structure	4
2	State of the Art	5
2.1	Visual Odometry	5
2.1.1	Introduction	5
2.1.2	Relevant works	11
2.2	Point cloud odometry	14
2.3	Multi-Modality	19
2.4	Critical Review	26
3	Deep neural network for late fusion of ego-motion estimates	31
3.1	Introduction	31
3.2	Input signals: data acquisition and augmentation	35
3.3	Data processing	37
3.4	Fusion layer architecture and composition	40
4	Results	45
4.1	Model optimization	46
4.2	Model evaluation in real road driving scenarios	47
4.2.1	Evaluation metrics and test data sequences	47
4.2.2	Regular navigation conditions	49
4.2.3	Multi-modal odometry	51
4.2.4	Challenging scenarios and conditions	54
4.2.5	Data augmentation impact	57
4.2.6	Comparison with state-of-the-art techniques for odometry	59
4.3	LSTM and multi-kernel convolution architectures	61
4.4	Kalman Filter comparison	63
5	Conclusions and future work	69
	References	71

List of Figures

2.1	Visual odometry categorization.	6
2.2	Illustration of epipolar geometry and epipolar constraint.	7
2.3	Feature-based Visual odometry scheme.	7
2.4	Feature and appearance-based visual odometry pipeline.	9
2.5	Illustration of pose graph optimization.	10
2.6	Geometry-based and deep learning-based VO pipelines comparison.	11
2.7	Feature tracking illustration in SOFT.	12
2.8	Structural architecture of DeepVO.	15
2.9	Point cloud odometry common stages.	15
2.10	LOAM block diagram.	17
2.11	LO-Net odometry network diagram.	19
2.12	Autonomous Vehicles <i>Perception Belt</i>	20
2.13	Data fusion processes: early, intermediate and late fusion.	21
2.14	Depth enhanced images in DV-LOAM.	22
2.15	Scan-to-map registration in DV-LOAM.	23
2.16	V-LOAM block diagram.	23
2.17	Point cloud projection into the image plane in LIMO.	24
2.18	Pose graph formulation and tightly-coupled architecture proposed by Wisth <i>et al.</i>	25
3.1	Illustration of the position and orientation components of a given pose	32
3.2	Illustration of the fusion layer's input interface.	33
3.3	Limitation example of the weighted average mean	34
3.4	Input/Output configuration of the late fusion layer	34
3.5	Illustration of the statistical dispersion of some arbitrary odometry inputs	35
3.6	Odometry techniques utilized to generate inputs for the fusion layer	36
3.7	Example of odometry estimates from <i>ORB</i> and <i>MULLS</i>	37
3.8	Data augmentation - synthetic signal errors	38
3.9	Data processing procedure (sliding window)	39
3.10	Full late fusion architecture	41
3.11	Integration of two stacked LSTMs in the original architecture	42
3.12	Illustration of multi-kernel convolution to extract data relations in the time domain	43
4.1	Error evolution using different optimizers	46
4.2	Error evolution using different sliding window lengths	47
4.3	Examples of the prediction capabilities of the fusion network	50
4.4	Box plots of the Euclidean error distribution for the individual odometry estimates in sequences <i>A</i> - and <i>B</i>	51
4.5	Challenging scenarios for visual odometry techniques in sequences <i>C</i> and <i>G</i>	52

4.6	Euclidean error distribution in sequence <i>C</i>	53
4.7	Impact of open spaces and lighting conditions on some of the fusion network inputs.	54
4.8	Reconstruction of the trajectory obtained by the fusion layer and the respective inputs in sequence <i>A</i>	54
4.9	Road traffic scenarios from KITTI-360 containing multiple dynamic objects in the vehicle's surroundings.	55
4.10	Impact of dynamic objects in the inputs and output of the deep fusion layer.	55
4.11	Results obtained in sequence <i>F</i>	56
4.12	Impacts of data augmentation in the network results from challenging scenarios.	58
4.13	Euclidean error distribution for all the evaluated odometry techniques across all selected sequences of KITTI-360.	59
4.14	Comparison between the originally proposed fusion network and the two extended architectures	62
4.15	Distribution of the differences between the motion derivatives and the true motion variations of the vehicle.	63
4.16	Histogram of the Euclidean error distribution for the fusion layer, KF_v , and KF_f on the same three sequences of table 4.9.	65
4.17	Fusion network and KFs outputs when applying white uniform noise to one of their inputs (ORB).	65
4.18	Error distribution of the fusion network and KFs when applying white Gaussian noise to one of their inputs.	66
4.19	Fusion network and KFs outputs when applying a non-changing failure to one of their inputs (MULLS).	67
4.20	Error distribution of the fusion network and KFs when simulating a non-changing failure in one of their inputs.	67
4.21	Fusion network and KFs outputs when applying a systematic deviation to one of their inputs (MULLS).	68
4.22	Error distribution of the fusion network and KFs when applying a systematic deviation to one of their inputs.	68

List of Tables

2.1	TVL-SLAM evaluation on KAIST dataset.	25
2.2	KITTI results of state-of-the-art discussed techniques	27
2.3	Execution time of state-of-the-art discussed techniques	28
3.1	Complementary statistical metrics used as inputs in the fusion layer.	40
4.1	Model parameters of the proposed fully-connected network.	48
4.2	Description of the selected evaluation sequences from KITTI-360 dataset.	49
4.3	Results obtained by the proposed fusion network and the respective input signals in the selected sequences from KITTI-360	52
4.4	Comparison of the results from the fusion network trained with the original dataset ("Fusion") and trained with the data augmented dataset ("Fusion +").	58
4.5	Description of state-of-the-art odometry techniques utilized for results comparison.	59
4.6	Results obtained by several odometry techniques in nine road-driving sequences of KITTI-360, including the proposed fusion network. The results are evaluated under the MAE_{rel} , Med , and MAE_{traj} metrics , expressed in different colors in the table. The techniques marked with an asterisk were used as inputs to the fusion layer, which is labeled as "Fusion". Best results are underlined for each sequence.	60
4.7	Model parameters of FC-LSTM and CNN-FC.	61
4.8	Comparison between the fusion network and the two weighted average approaches in the selected KITTI-360 sequences under the MAE_{rel} and MAE_{traj} metrics (in meters).	64
4.9	Comparison between the fusion network and the two weighted average approaches in the sequences induced with input signal failures. Each column refers to a KITTI-360 sequence, in which one of the inputs (ORB-SLAM2 in the first and MULLS in the last two) is affected with the indicated type of error. In the first, the signal is affected with white noise of null mean and uniform distribution. In the second, the signal starts producing a static value over time, and in the third, the signal is affected by an offset. All sequences present the respective failure regime during half the sequence length. Results are expressed in terms of MAE_{rel} and MAE_{traj} metrics (in meters).	64

Acronyms and Symbols

2D	Two-dimensional
3D	Three-dimensional
AI	Artificial Intelligence
BRISK	Binary Robust Invariant Scalable
CNN	Convolutional Neural Network
DL	Deep Learning
DoF	Degrees of Freedom
DSO	Direct Sparse Odometry
DVSO	Deep Virtual Stereo Odometry
EKF	Extended Kalman Filter
FCNN	Fully-Connected Neural Network
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GRU	Gated Recurrent Unit
ICP	Iterative Closest Point
IMU	Inertial Measurement Unit
INS	Inertial Navigation System
KF	Kalman Filter
LiDAR	Light Detection and Ranging
LO	Laser Odometry
LSTM	Long Short-Term Memory
ML	Machine Learning
MSE	Mean Squared Error
MULLS	Multi-metric Linear Least Square
NN	Neural Network
ORB	Oriented FAST and Rotated BRIEF
PnP	Perspective-n-Point
RANSAC	Random Sample Consensus
RCNN	Recurrent Convolutional Neural Network
ReLU	Rectified Linear Unit
RGB	Red Green Blue
RGB-D	Red Green Blue - Depth
RL	Robot Localization
RMSE	Root of Mean Squared Error
RNN	Recurrent Neural Network
ROS	Robot Operating System
SAD	Sum of Absolute Differences
SGD	Stochastic Gradient Descent

SIFT	Scale-Invariant Feature Transform
SLAM	Simultaneous Localization and Mapping
SLERP	Spherical Linear Interpolation
SONAR	Sound Navigation and Ranging
SURF	Speeded up Robust Features
VO	Visual Odometry
V-SLAM	Visual-Simultaneous Localization and Mapping
WHO	World Health Organization

Chapter 1

Introduction

1.1 Context

In the present days, autonomous driving is undeniably one of the most relevant subjects in technology. The concept itself is vastly broad, covering topics ranging from the latest engineering advances, to socio-cultural, economic, legal and even philosophical aspects. Generally speaking, it is of common interest to several social sectors that progress is made in the autonomous driving area, both for commercial use and industrial purposes. In fact, the existence of partially or completely autonomous vehicles brings a variety of benefits, covered as follows.

Road safety is foreseen to increase as vehicle control gradually shifts from the driver to the vehicle itself. Effectively, the World Health Organization (WHO) suggests that approximately 1.35 million deaths were caused due to road traffic accidents in 2018 [1]. Additionally, multiple road accident researches point out that over 90% of road accidents are caused by human failure [2]. Fully-autonomous vehicles may not yet reach the same driving skills as an attentive human driver, but can increase safety by means of supervision, warnings and driving support. In light of the European Commission's "Vision Zero" program [3], which targets zero fatalities on the road by the year 2050, multiple investments have been made, particularly in technologies that promote vehicle autonomy. Another clear improvement would be on traffic flow. The analysis provided in [2], based on traffic flow models, concludes that the existing transport infrastructure can be used more efficiently while reducing traffic jams and wasted time on the road. Economic concerns can also be pointed out, for example, in unmanned freight transport. To add to the latter, autonomous vehicles allow for reduced fuel consumption, as well as greater comfort and convenience for passengers. Despite the pros, several polls and surveys on public opinion about autonomous vehicles, compiled by *Advocates for Highway and Auto Safety*¹, seem to agree that there is a deep public skepticism about autonomous driving, namely in regard to safety-related issues. While public opinion reveals a general lack of trust and receptiveness towards this scenario, the very phenomenon should be regarded as an incentive to accomplish further progress in this field.

¹<https://saferoads.org/wp-content/uploads/2020/01/AV-Public-Opinion-Polls-7-22-19.pdf>, consulted on 20 November 2019

For any autonomous mobile agent, the ability to self-locate is practically mandatory in every navigation and obstacle avoidance task. Thus, one of the technological fronts on improving autonomous driving systems is focused on the development of techniques and equipment that endow the agent with a certain degree of awareness and perception of its surroundings. It is commonplace to find, in nearly every vehicle today, a GNSS receiver as the primary source of self-localization. However, mass-market receivers provide levels of accuracy and reliability well below those required for use in autonomous vehicles. For example, a typical smartphone GPS receiver has an accuracy of about 5 meters². For this reason, autonomous mobile agents may only loosely depend on satellite data, because of the uncertainty of positioning error, but also of signal delays and quality-of-service issues [4]. This problem further aggravates in urban scenarios due to limited satellite visibility, multipath effect, interferences, and other impairments [4]. Furthermore, these are very dense and populated environments, in which a few meters make a significant difference. There currently exist satellite-based techniques which are able to provide centimeter accuracy, yet they are usually quite expensive [5]. All these limitations are compensated for by drivers' own visual perception capabilities. In the same way, autonomous cars can be equipped with sensors, such as cameras and LiDAR, which provide a similar level of precise relative localization, when utilized with proper techniques. These techniques encompass Visual Odometry (VO) and Laser Odometry (LO) methods, which will be discussed throughout this document.

VO has the particularity of being cost-effective, since generally used cameras are relatively inexpensive, easily found in modern cars, and don't depend on external equipment, unlike GNSS. Furthermore, it is a much more robust alternative to wheel odometry, as it does not suffer from wheel slippage, being also a great complement to global positioning systems and Inertial Measurement Units (IMU) [6]. Its simplicity to produce state estimations, along with the applicability in resource-constrained platforms compose even more arguments in favor of VO [7]. Since the 1980s, VO methods have been the subject of considerable research, and, nowadays, it is possible to achieve trajectory estimates with relative position error ranging from 0.1 to 2% in real time [6], proving the relevance of this set of techniques.

1.2 Motivation

One of the biggest obstacles in autonomous driving, particularly at perception level, lies in the fact that it is very difficult to generalize the environments where vehicles will be deployed. For instance, foggy or sandy landscapes with little texture make the VO task difficult; urban scenes with many dynamic objects or poorly illuminated areas are also a major problem. To make perception systems more robust, it is common to complement VO systems with other sensors, as a consensus grows on equipping vehicles with perception belts composed of camera, LiDAR and RADAR [8]. To obtain a more capable system, the proposal to fuse data, namely images and point clouds, becomes increasingly relevant and promising, given the advances in sensors, computing technologies, and motion estimation algorithms. This VO-applied combination aims to produce

²<https://www.gps.gov/systems/gps/performance/accuracy/>, consulted on 13 January 2022

superior results while improving robustness in scenarios where cameras alone are likely to fail. Although encouraging, this approach seems unattended, as there is substantial research related to the use of cameras and LiDAR but in isolated fashion [9].

Meanwhile, the vast advances in Machine Learning (ML), namely in Deep Learning (DL), are undeniable and its applications extend to the most diverse areas, far beyond Robotics only. Traditional geometry-based techniques are starting to show signs of maturity, while still presenting many shortcomings in complex environments [10]. For instance, these procedures often end up relying on heavily hand-tuned adjustments, showing some lack of adaptability. On the other hand, learning-based approaches offer the advantage of extracting information from data at a higher level, closer to human perception, which guarantees increased reliability in environments where VO faces further limitations. In addition, the existence of a wide range of datasets and the very nature of learning-based techniques allow them to adapt to a wider range of scenarios, presenting a tempting proposal in the generalization problem, where classical methods may fall short [10]. Moreover, the recent stagnation of traditional approaches prompts the trend towards the use of alternative techniques like VO-applied learning. Nevertheless, Deep VO is a quite contemporary tendency, and the results reflect that. In fact, there are still many underexplored subjects, from which some questions arise, such as: *what are the most suitable deep learning structures for odometry? Is it possible to adapt the existing structures or should we think of designing something from scratch? What is the impact of alternative learning techniques like Weakly Supervised Learning? How can deep learning be applied in conjunction with a multi-modal data architecture?* In light of these questions, it is conclusive that Deep VO is still in the early stage of development and its potential needs to be further explored. In addition, the state-of-the-art review of chapter 2 revealed the increased benefits of multi-modal approaches in overall robustness. Yet, in this category too, the use of DL is still very sporadic and underdeveloped, posing again an opportunity to develop a multi-modal fusion architecture using deep learning techniques. As such, proceeding from these motivations, this dissertation proposes a series of objectives to be met that will be detailed in the following section.

1.3 Objectives

This dissertation aims to elaborate the architecture of a multi-modal odometry system, validating the premise that using data generated by different sensors lends more robustness to the overall solution. Deep learning techniques and structures will be studied and applied in the algorithm's design, with particular attention to the generalized nature of the solution, so that it can perform effectively in various types of environment.

The proposed methodology will rely on datasets that include camera, LiDAR, IMU and GPS data in driving scenarios, either to acquire sensitivity towards the problem, train, test and evaluate the presented solution.

Thus, it is expected from this work:

- To compile a dataset composed of odometry estimates from different sources/sensor modalities. Data should be retrieved from commonplace and challenging scenarios to be as representative as possible of the typical road environment;
- To develop a late fusion architecture using the dataset information to predict reasoned ego-motion estimates;
- To base the framework on deep learning structures capable of producing maximum likelihood outputs. The data should allow the network to generate estimates based on the statistical dispersion of each input, their behaviors, and the similarities/cross-divergences between them;
- To provide the network with mechanisms/information to detect individual failure regimes in each input and reduce the impact of the faulty signal on the final estimate;
- To produce coherent motion estimates between timesteps that respect the models of the vehicle kinematics from which the data has been generated.
- To train and evaluate the model by comparing it with the performance of the respective inputs, alternative state-of-the-art odometry techniques, and the Kalman filter.

1.4 Contributions

This dissertation was developed under the THEIA project, a partnership between FEUP and Bosch Portugal for research in the area of autonomous driving. Additionally, the work resulted in a survey of state-of-the-art visual odometry techniques, along with a new benchmark and an in-depth discussion of the limitations of these methods. The article, entitled "A Practical Survey on Visual Odometry for Autonomous Driving in Challenging Scenarios and Conditions", was published in IEEE Access on 7 July 2022.

1.5 Document structure

This document begins by clarifying some concepts and reviewing the state-of-the-art of visual odometry, point cloud-based odometry, and multi-modal odometry. This is part of chapter 2, which also includes a critical review of the current panorama of the area. Then follows chapter 3 which presents the methodology of the work carried out, as well as the solutions proposed for the creation of a multi-modal fusion system for odometry. Chapter 4 analyzes the results obtained by the proposed solutions, comparing them to a set of state-of-the-art techniques. Finally, the conclusions raised by the research work and possibilities for future work are detailed in chapter 5.

Chapter 2

State of the Art

This chapter provides a literature review in the area of visual odometry and applied sensor fusion. It includes an introductory section (2.1.1), where some important concepts that cut across all methods are clarified. Then follows section 2.1.2 where some relevant visual-only methods are discussed, both with traditional and learning-based approaches. Following, in sections 2.2 and 2.3, some works on point cloud odometry and multi-modal odometry are discussed, as well as their key concepts, advantages and disadvantages. Finally, section 2.4 presents a critical reflection on all the presented works, complemented by the respective conclusions arising from the literature study.

2.1 Visual Odometry

2.1.1 Introduction

Odometry is the process of estimating an agent's change in position and orientation over time. Visual Odometry (VO) is the designation given when relying on the input of a single or multiple cameras attached to the agent [6]. VO methodologies consist of reckoning the pose of the robot (or any other agent, e.g. autonomous vehicle) by analyzing subsequent image frames and extracting ego-motion from corresponding points observed from different perspectives in each frame. The underlying goal is to estimate the rigid transformations between successive camera poses and thus, by integration, reconstruct the trajectory described by the agent.

The term "Visual Odometry" was popularized by Nister in [11]. Yet, the concept was introduced by Moravec [12] in the early 1980s, as his work helped to consolidate the general pipeline used in most of VO methods. From that time until the early 21st century, it has been the subject of considerable advances, driven primarily by preparations for NASA's 2004 Mars Mission [13]. Since then, following the development of computing technologies, image capture equipment, and even the fervent rise of Artificial Intelligence (AI), VO has attracted numerous researchers alongside a vast set of proposed techniques. VO is also often regarded as a sub-segment of Visual-SLAM algorithms. The scope of Simultaneous Localization and Mapping (SLAM) is global, i.e. it involves mapping the surrounding area while locating the agent in the overall map, hence, it

naturally includes a module responsible for motion estimation. VO algorithms can be seen as such module, whose focus is the reconstruction of the trajectory, not in a global perspective, but rather relative to previous agent poses (i.e. without loop-closure). Consequently, the study of visual-SLAM techniques can always be considered when addressing visual odometry.

The set of existing VO methodologies can be divided into two distinct groups: knowledge-based approaches and learning-based approaches [7]. The first exploits camera geometrical relations to assess the motion, whereas the others are based on Machine Learning techniques, which rely on considerable amounts of data to provide correct pose estimates. Furthermore, knowledge-based approaches can be categorized into three sub-groups: appearance-based, feature-based and hybrid techniques (figure 2.1).

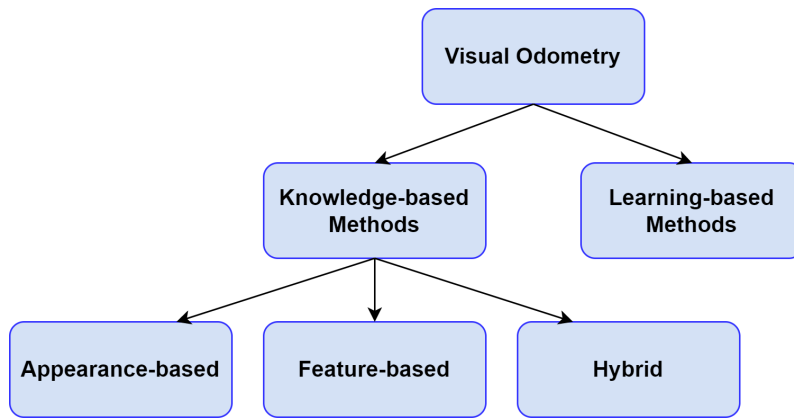


Figure 2.1: Visual odometry categorization.

Feature-based methods focus on the premise that there are points in image frames that can be used as references to correlate sequential images and thus determine camera movement. These key-points consist of image patterns that are distinguishable from their surroundings in terms of intensity, color, or texture, such as corners, edges, lines, and blobs; being more likely to match well in other images [6, 14, 7]. Feature-based VO is considerably robust to geometric distortions and illumination inconsistencies [15]. Yet, by selecting only some points of the image, some valuable information is discarded, besides the fact that it could lead to high computational costs associated with feature extraction/matching and outliers rejection tasks, depending on the number of features being utilized [7].

In the scope of feature-based methods, motion estimation is usually approached in two different ways: *i*) feature-to-feature matching (2D-2D); *ii*) structure-to-feature matching (3D-2D). The former enables the extraction of the Essential Matrix (E), whose decomposition allows for obtaining the Rotation, R , and Translation, t , matrices of the agent's motion. This approach exploits the constraints imposed by epipolar geometry on correspondent features in two different frames (figure 2.2). However, feature-to-feature matching is limited in the sense of recovering the absolute scale of the scene. One of the most commonly utilized techniques to compute the Essential Matrix is by using 5 pairs of correspondences and solving for the motion parameters, proposed by Nister in [16].

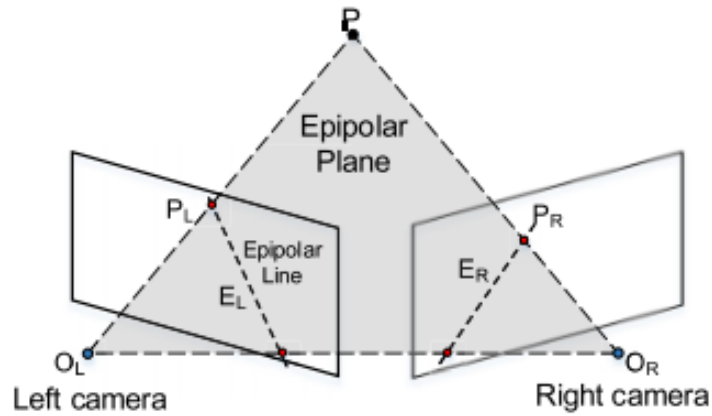


Figure 2.2: Illustration of epipolar geometry and epipolar constraint [15]. P_L and P_R represent the same point observed from two distinct perspectives, left and right respectively. These points relate with the Essential Matrix E by the Epipolar Constraint: $P_L^T E P_R = 0$.

In the 3D-2D case, rotation and translation are calculated by minimizing the reprojection error of 3D landmarks, established in previous frames, into the current frame. This is known as Perspective-n-Point problem (PnP) [7], whose most frequent solution involves using 3 different points (P3P) [17]. Usually 3D-2D motion estimation is faster and performs better compared to 2D-2D. However, the choice should be made according to the type of sensors being used, e.g. monocular or stereo [6]. Figure 2.3 depicts an illustration of the typical feature-based VO scheme, more precisely the concept of tracking selected points.

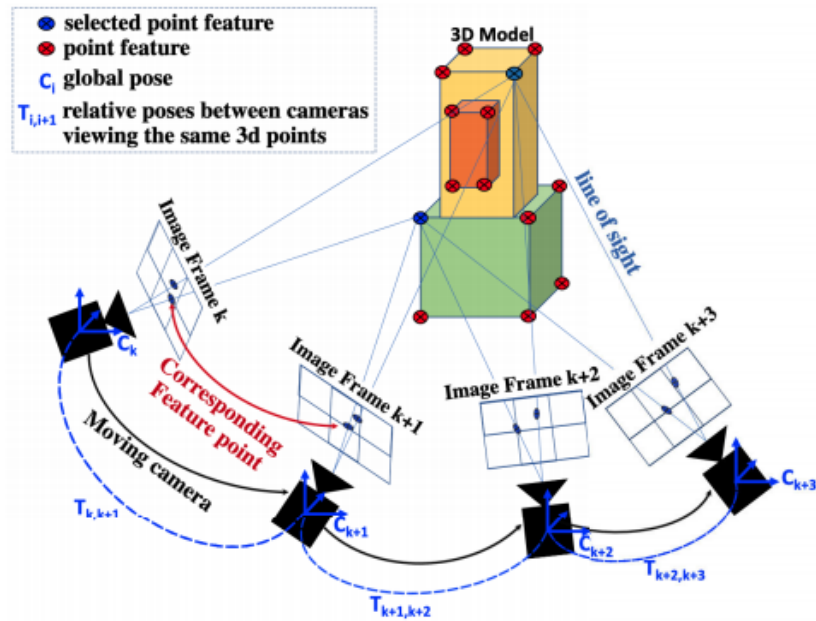


Figure 2.3: Illustration of Feature-based Visual odometry scheme [7]. The two blue points are tracked in every image frame, allowing for the motion to be estimated by the distinct positions occupied by the points in each frame.

The quality of estimations is greatly impacted by outliers in the set of inter-frame feature correspondences. For this reason, it is highly important to resort to mechanisms that effectively detect and remove false matches, to prevent erroneous results. The most widely used outlier rejection method is RANSAC [18], an iterative process based on a random voting scheme, whose principle assumes that there are enough inliers to agree on a model that reliably represents the data set. As a consequence, but also due to the computation time involved, it is important to choose the appropriate feature detectors and descriptors. Currently, the most widely employed are SIFT [19], SURF [20], ORB [21] and BRISK [22], since they are relatively fast as well as invariant to scale and rigid transformations. Nonetheless, the choice of detector must take into account the trade-off between speed and accuracy. Research in recent years has focused on both feature detectors and descriptors when applied to VO, but also on VO techniques that foster a careful post-selection of features before calculating pose transformation. The work presented in [23] conducts an experimental analysis of the most common descriptors addressed in visual odometry. The authors conclude that in the balance between accuracy and computation, SURF and SIFT tend towards higher accuracy, ORB towards lighter computation, and recently proposed A-KAZE presents an intermediate compromise between the two.

In turn, appearance-based VO techniques make use of all the information in the captured frames, as opposed to just using key-points. These methods estimate the camera pose by analyzing the intensity of image pixels, and minimizing the photometric error, which relies on the consistency principle that pixels from one frame maintain their intensity in the second frame, while shifting positions [7]. This way, it is possible to mitigate the aliasing effect caused by repetitive patterns and ensure more robustness in scenes with limited texture where it's harder to detect good features (e.g. foggy or sandy environments). Appearance-based techniques are usually fast to implement [24] while preventing the time required to extract/match features, and run outlier rejection algorithms. On the other hand, they are quite more sensible to illumination variations in the scene and abrupt camera movements, compared to feature-based VO.

Appearance-based methodologies are generally classified as region matching-based or optical flow-based. The former estimates camera motion through the alignment of certain corresponding regions in consecutive frames, but it loses effectiveness significantly in the presence of dynamic objects in the scene, besides being susceptible to local minima solutions. On the other hand, optical-flow-based techniques, as [25, 26, 27], calculate the pixel movement from one frame to the next, resulting in a vector matrix which is then used to estimate the 6 Degrees-of-Freedom (DoF) of camera motion, based on motion models. Figure 2.4 aggregates the different pipelines of feature and appearance-based VO methods. In an effort to mitigate the disadvantages of both feature-based and appearance-based methods, as well as to aggregate the added value of each, it is also possible to resort to different approaches from each domain to elaborate a more robust solution. These are called hybrid techniques [7].

Given the integrative nature of odometry, the deviation from ground-truth trajectory tends to accumulate as errors are incurred in the estimation of each pose. To reduce this problem, it is practically transversal to all geometric approaches to apply an optimization step in the pipelines'

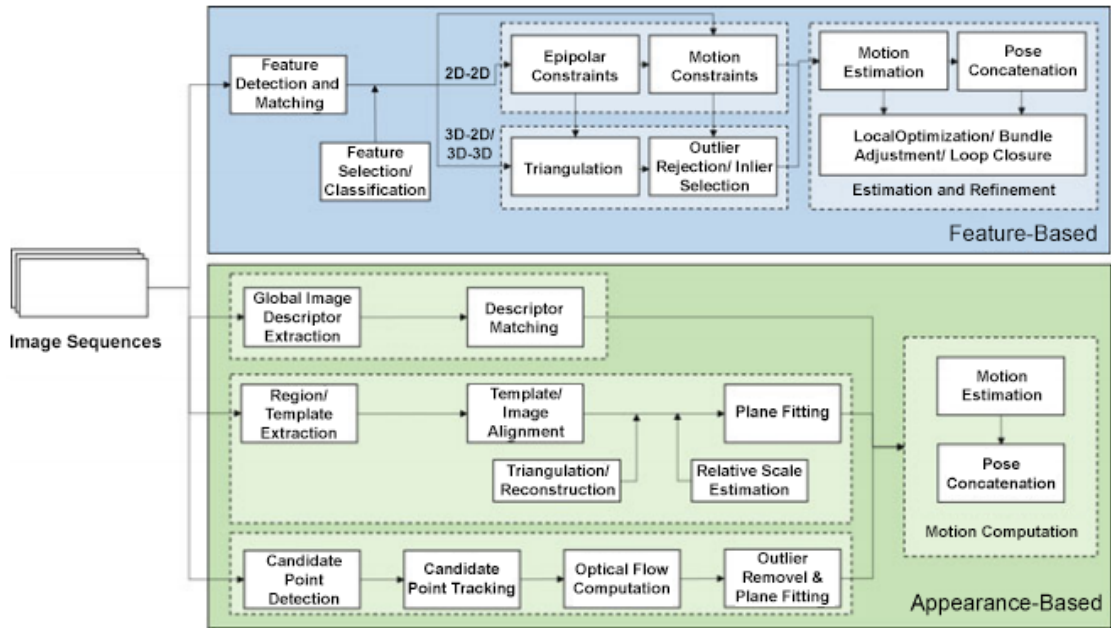


Figure 2.4: Common steps in feature and appearance-based VO pipeline [24].

final stage to correct the drift. Pose graph optimization [28] is a common technique based on the consistency of estimates, i.e. computing the newly pose by the actual and last frames should result in the same pose computed from the last and prior frames, as depicted in figure 2.5. However, in practice, this is not the case, so the pose estimate can be refined by minimizing these discrepancies. Bundle adjustment is another approach for reducing the trajectory error. Whenever a point is tracked during multiple successive image frames, its 3D coordinates should be consistent within each camera pose. Once again, small deviations occur in the odometry estimates, which can be adjusted by these additional geometric constraints, while at the same time, refining also the landmarks' positions. Bundle adjustment differs from loop-closure in the way the landmarks are tracked. For computational reasons, optimization is usually restrained to a limited set of constraints, thus, in Bundle adjustment, the 3D structures are tracked only over a limited window of frames. Loop-closure applies a similar concept, but it takes place when the agent revisits and recognizes a place, reducing the trajectory uncertainty by applying the same geometric principle of consistency. Therefore, this method is applied in SLAM algorithms, since it implies maintaining an accurate map of the environment.

Geometric (or traditional) methods already have well-established foundations, and while these approaches have generated rather reasonable results throughout their evolution, they prove to be fragile in environments with increased complexity. In fact, it becomes challenging to rely solely on this type of techniques since it is extremely difficult to capture the complexity of the real world by hand formulation. As seen in diverse areas, data-driven learning-based approaches are able to acquire a high-level understanding of the scene, without the need of explicit modelling, as long as trained on sufficiently large-scale representative datasets [10]. In addition, camera calibration

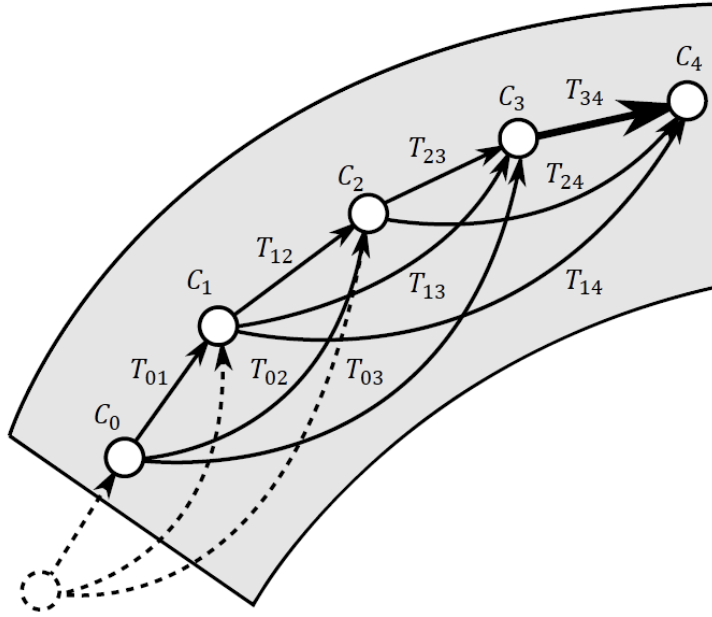


Figure 2.5: Illustration of pose graph optimization [29]. Pose C_4 , for instance, is computed not only from the relation with pose C_3 (by T_{34}), but also from poses C_2 and C_1 (by T_{24} and T_{14}).

parameters do not need to be known *a priori*; translation can be estimated with correct scale, and system’s robustness against image noise increases [24]. As a result, there has been a paradigm shift in VO, leaning towards learning-based methods in recent years. Figure 2.6 illustrates the different architectures of deep learning-based approaches and classical ones. Typical data-driven techniques often consist of multiple sub-nets with distinct functions, such as depth estimation, feature extraction and ego-motion estimation [10]. These are connected in a complementary way and usually trained together as a single network, by confronting the outputs with a supervision signal or by evaluating them against a cost function which depending on the results, readjusts and fine-tunes the network parameters.

The following sections will present a set of works in the field of visual odometry that achieved good state-of-the-art results and/or proposed some valuable approaches with future potential, justifying further research. Moreover, in Section 2.4, those results will be compiled and further discussed. It is commonplace for the majority of proposed techniques, to be tested and report results obtained in KITTI dataset [30], which comprises a total of 22 sequences of visual and LiDAR data recorded on board of a vehicle that transits in common road traffic environments, 11 of which include the respective ground truths. Furthermore, KITTI provides an evaluation tool that allows for comparing multiple algorithms and rank them on a scoreboard¹. The outcomes are frequently evaluated in terms of two metrics that will be recurrent throughout this chapter: *i*) t_{rel} , the average relative translation error over sequence lengths of 100m to 800m, in percentage; *ii*) r_{rel} , the rotation error in deg/100m over the trajectory.

¹Available at: http://www.cvlibs.net/datasets/kitti/eval_odometry.php

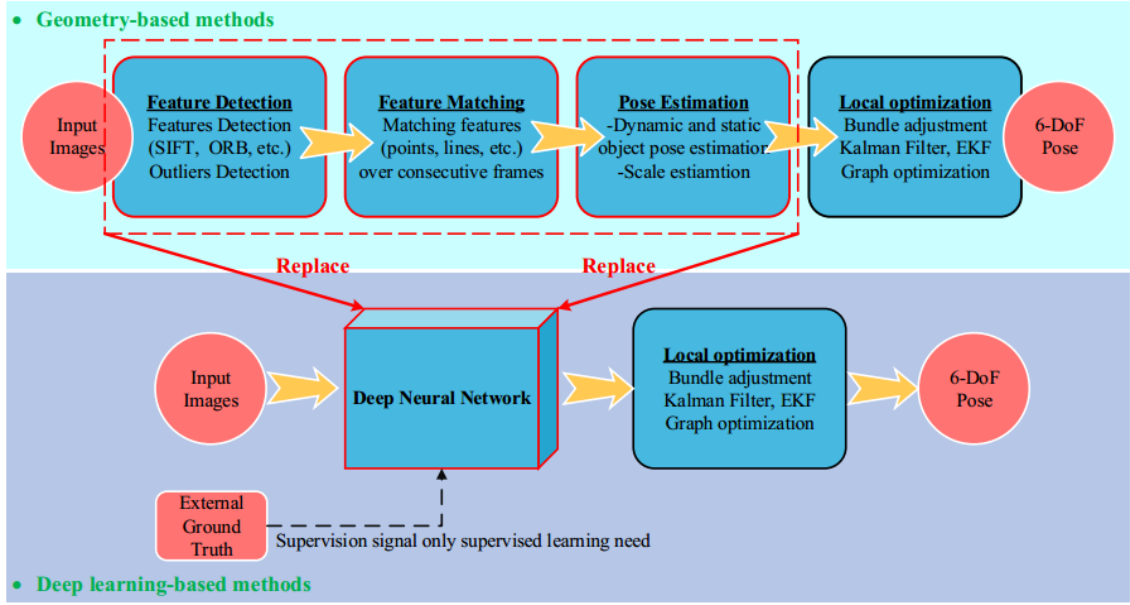


Figure 2.6: Comparison between generalized pipelines followed by geometry-based and deep learning-based VO methods [10]. All the manual steps related with detecting and handling features can be encapsulated/substituted by a neural network architecture or multiple specific sub-nets, reducing the influence of hand-tuned procedures.

2.1.2 Relevant works

ORB-SLAM2 [31] is a popular algorithm among the VO and SLAM communities. It is the extension of ORB-SLAM [32] for monocular, stereo and RGB-D cameras, and while open-source, it's often considered as a benchmark in visual odometry. Being a SLAM technique, this algorithm is composed of 3 threads: tracking and local mapping, which compose the odometry module, and loop-closing. Motion estimation is computed by using ORB features tracked over keyframes and a local map, with special emphasis on multi-step bundle adjustment.

Moreover, the works of Cvišić and Petrović are of special relevance in the field of visual feature-based approaches, since their results bring visual methods' precision closer to the precision levels of LiDAR-based techniques. In [33], the authors proposed a Stereo Odometry algorithm, whose main focus is the careful selection and tracking of features (SOFT), and the impact of these steps on ego-motion estimates. First, corner and blob-like features are extracted on a gradient image. The matches are detected assuming a constant velocity model, using Sum of Absolute Differences (SAD) in a patch around the points, and Circular Matching to mitigate susceptibility to outliers. Then, Normalized Cross Correlation and RANSAC are applied to make an even more thorough selection of quality feature correspondences. The next selection step ensures that points are chosen near and far from the camera and uniformly throughout the image. Some experiments have shown that if the selection is careful enough, adding points only detracts from the performance of the system. Finally, feature tracking is achieved through a feature descriptor which contains, for instance, its age, refined position in the image, and a quality measure, allowing

for robust and reliable tracking (figure 2.7). For the calculation of pose transform, rotation and translation are estimated separately. In the case of rotation, Nister’s 5 Points is used in conjunction with RANSAC (the authors suggest a detailed explanation and some experimental results that validate this choice). This step is further repeated for non-successive frames containing the same features, generating the final result by a Spherical Linear Interpolation (SLERP). The translation is finally calculated using the reprojection error of the closest points to the camera (3D-2D) and the previously calculated rotation.

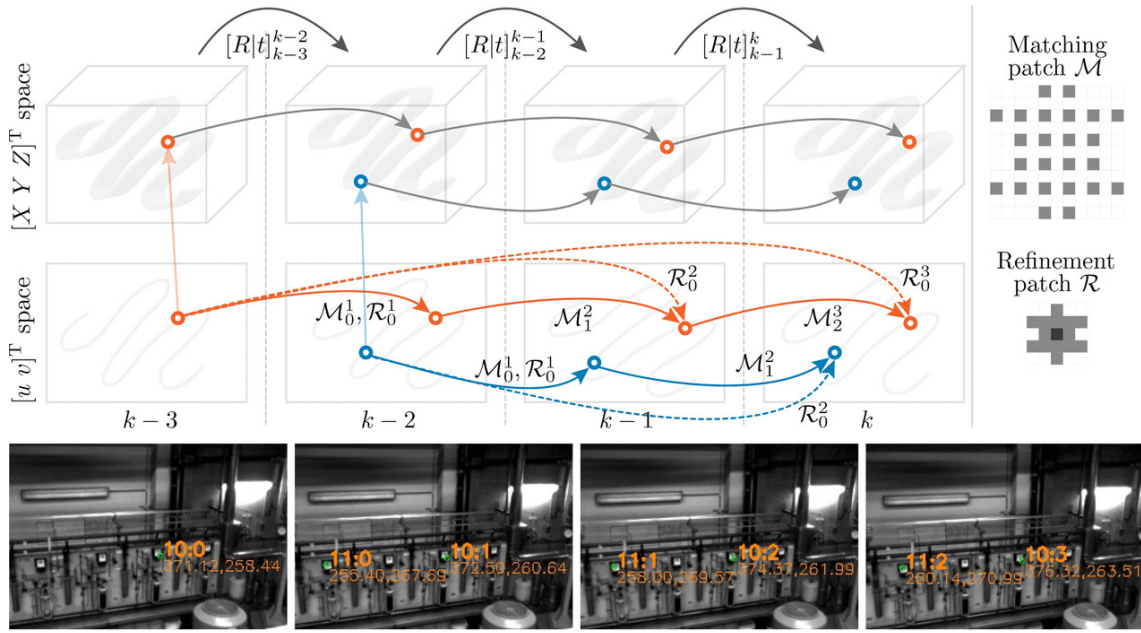


Figure 2.7: Feature tracking illustration in SOFT [34]. The upper row represents the 3D propagation of positions of two features, while the second row represents the propagation in the image plane. Solid arrows stand for the direct matching and dashed arrows for the refinement in the optimization stage, using prior poses.

Later, in 2017, the same authors extended SOFT with an additional mapping thread, giving rise to SOFT-SLAM [34]. This method was originally intended for autonomous unmanned aerial vehicles, with the focus on being computationally efficient. SOFT-SLAM integrates the SOFT visual odometry pipeline and completes it with a mapping module which adds SLAM features such as loop-closure and global consistency. These added capabilities give this technique superior localization accuracy over its state-of-the-art alternatives, such as ORB-SLAM2 and LSD-SLAM [35]. In 2021, Cvišić *et al.* [36] revisited the calibration parameters of the KITTI Odometry dataset. In this last work, the authors propose a new one shot technique for calibrating the parameters of the multi-camera KITTI setup, which in turn results in smaller reprojection errors, directly impacting the accuracy of VO algorithms. The adjusted parameters were applied to a few state-of-the-art methods, such as ORB-SLAM2, with significant improvements in all. By applying the same parameters to a modified version of SOFT, i.e. SOFT2, the authors were able to reach the top of the

KITTI scoreboard ranking² with a t_{rel} of 0.53% and 0.09 deg/100m of average rotation error (r_{rel}), while the mean values for the top 20 places are 0.61% and 0.015deg/100m, respectively.

In 2017, Engel *et al.* [37] proposed Direct Sparse Odometry (DSO). This method is composed of a direct (also known as appearance-based) and sparse scheme, thus not requiring feature detection and matching operations. This approach works by continuously optimizing the photometric error over a finite window of frames, but in contrast to typical direct methods, the optimization occurs for all parameters simultaneously, like an extended bundle adjustment of model parameters, including ego-motion, camera calibration and inverse depth of 3D points. By taking into consideration the full error, as opposed to the error of a particular iteration only, limits the effect of outliers. Moreover, a photometrical model is conceived, which takes into account the image formation process to improve robustness over lighting variations that frequently affect appearance-based approaches. The importance of this work lies on the experimental conclusions that with properly calibrated data, i.e. retrieved by cameras developed specifically for this purpose, which are progressively more common, direct formulations could surpass geometrical/indirect approaches in terms of accuracy, which have been dominating research interest in the past decade.

The following year, DSO was integrated into a deep learning-based framework, leading to Deep Virtual Stereo Odometry (DVSO) [38]. The core of this work is based on extending the capabilities of DSO through a network that produces accurate depth estimates using only one camera, thus reducing scale drift. StackNet, the proposed fully convolutional deep network, receives as input the left image of a stereo system and generates a pair of depth maps that simulate a virtual stereo rig. Training is achieved in a self-supervised manner, i.e. by comparing the back-warped produced outputs, using the transform predictions, with the original inputs. This technique is very common in odometry learning-based methods, avoiding the use of ground truths, which are costly to obtain. Then, StackNet's predicted depth maps are added as additional geometric constraints to the estimated depths in the original DSO general optimization problem. Deep visual odometry methods have so far failed to supplant classical ones. This work, however, goes far beyond the learning-based alternatives, and comes closer to conventional state-of-the-art techniques. The available data suggest a t_{rel} of 0.89% and a r_{rel} of 0.20 deg/100m on KITTI training sequences.

In 2020, Yang *et al.* [39], the same authors of DVSO, presented Deep Depth, Deep Pose and Deep Uncertainty, also known as D3VO. The overall architecture of D3VO is slightly different from that of DVSO and DSO. In D3VO, depth is estimated from a convolutional network called DepthNet, which works similar to StackNet, while also predicting the uncertainty associated with estimates. Uncertainty is particularly important for identifying moving objects in the scene, removing their weight in the estimation of ego-motion. Camera poses, in turn, are estimated from another convolutional deep network called PoseNet, which in addition to transformations, can equalize the illumination of the current and previous frames that together constitute the pair of network inputs, in order to mitigate errors arising from the variation of lighting. Both DepthNet and PoseNet are trained by minimizing the reprojection errors between the inputs and the back-warped outputs, in a self-supervised and joint manner. The pose predictions serve to complement

²On January 2022

a direct tracking front-end module and a global bundle adjustment back-end module based on previous works DVSO and DSO. The results slightly exceed those of DVSO with $t_{rel} = 0.82\%$.

Different from the previous approaches, the work presented by Wang *et al.* [40], DeepVO, is an end-to-end supervised method whose focus goes into learning feature extraction with proper geometric significance and implicitly modelling the motion dynamics over a sequence of frames. To do so, a pair of sequential images is stacked and passed as an input to a Convolutional Neural Network (CNN), generating a compact descriptor of the input pair which is then fed to a Recurrent Neural Network (RNN), that unlike CNNs, keeps memory of its internal states over time, thus allowing the modelling of the sequential properties of VO. This junction gives rise to a RCNN - Recurrent Convolutional Neural Network, as depicted in figure 2.8. Nonetheless, unlike DVSO and D3VO, the results are somewhat unsatisfactory ($t_{rel} = 5.96\%$ and $r_{rel} = 6.12$ deg/100m), yet this approach allows for a non tuning-dependent model that works as a proof of concept for end-to-end DL methodologies. In fact, most of commonly used DL structures in Computer Vision, as CNNs and RNNs are not well suited for VO. This kind of works can serve as starting points and catalyze new VO applied research. Following a similar path, PoseConvGRU [41] is an end-to-end comparable method. PoseConvGRU leverages DeepVO time efficiency by using stacked GRUs instead of LSTMs. LSTMs and GRUs are specific types of RNNs which enable capturing long term relations between successive inputs, by learning and storing internally what is relevant to keep, or not, in the form of internal states. This type of structures is specially important in visual odometry, given the temporal geometrical constraints over a sequence of frames. In PoseConvGRU, GRUs are preferred since they are quite close to LSTMs in performance terms, but achieve similar results with less parameters and time consumed. Zhao *et al.* [42], in turn, predict the optical flow (using FlowNet [43]) to compute the relative transformation of the camera pose and reconstruct a few 3D structures of the scene by triangulation. These structures are then used to align depth predictions coming from a parallel neural network, tackling the problem of scale inconsistency between pose and depth predictions. This is a problem that hinders the learning process and, therefore, the results, so the authors divide the training process of pose and depth branches, taking advantage of the output of both to complement each other and create consistency between them. While achieving slightly better results than DeepVO, these last two approaches present the same error's order of magnitude as DeepVO, thus generating the same conclusions.

2.2 Point cloud odometry

So far, the potential of visual odometry has been acknowledged. Conversely, there are some inherent problems in visual techniques that can hardly be addressed without the use of different sensor modalities. Specifically, performance is highly dictated by: lighting conditions, making it almost impossible to use VO alone at night, for example; scene's appearance, namely featureless environments or ones with very repetitive patterns; and camera(s) calibration precision. A recurring alternative is the use of LiDAR sensors for odometry estimation. Similar to visual techniques,

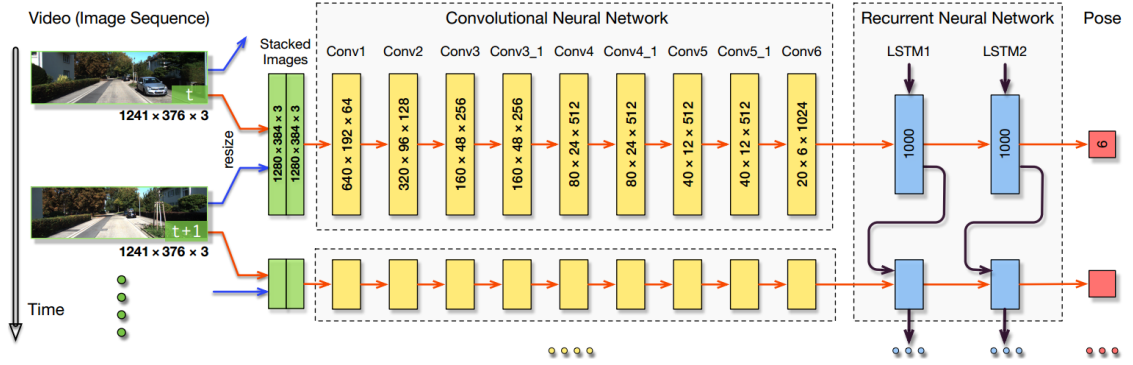


Figure 2.8: Structural architecture of DeepVO [40]. The junction of a CNN with a RNN results in the proposed RCNN, which allows features to be extracted and to capture the sequential relations between image frames.

the goal is to estimate the spatial transformation between two or more LiDAR sweeps by matching point clouds. As in VO, some common steps can be identified in the point cloud odometry (or Laser Odometry - LO) pipeline, such as pre-processing, feature extraction, correspondence searching, transformation estimation, and post-processing, as illustrated in figure 2.9.

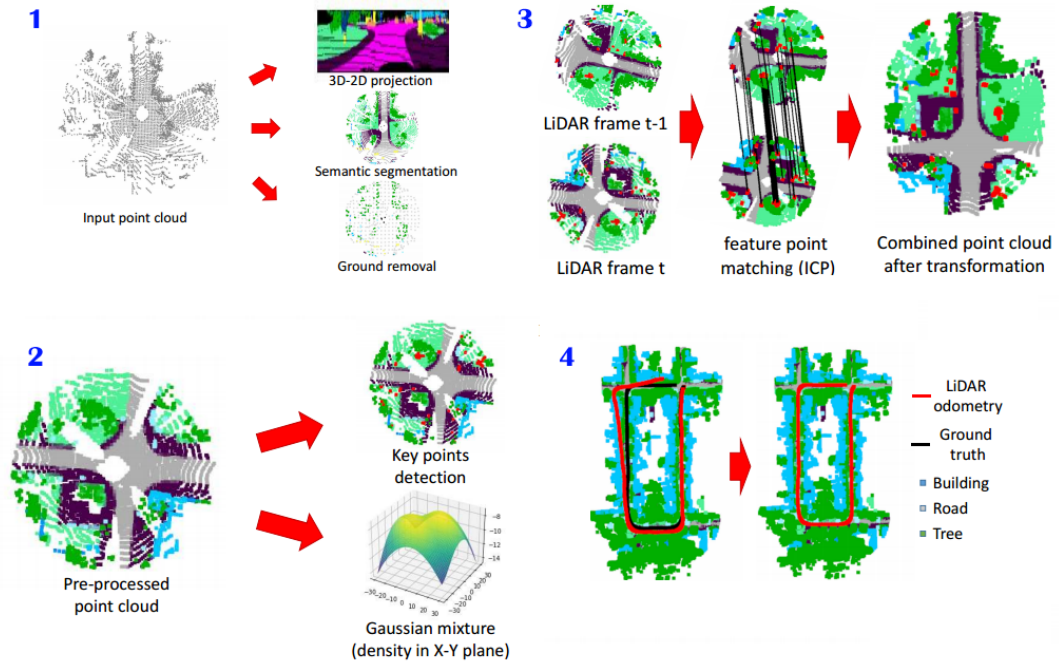


Figure 2.9: Illustration of common stages in point cloud odometry [44]. 1) LiDAR frame pre-processing; 2) Feature extraction; 3) Feature matching and registration; 4) Post-processing (Optimization).

Jonnvithula *et al.* [44] help to break down the laser odometry process as follows. In the first stage, point clouds can be organized, filtered and segmented, so that data can be used efficiently throughout the rest of the process. Unlike images, point cloud data is unstructured by nature,

i.e. it consists of a set of points in which the concept of neighborhood is not explicit, as it is in adjacent pixels. In other words, there is no information about vertex connection or about the surface underneath either. By searching the vicinity of each data point it is possible to retrieve that kind of information and store it in multiple data structures, as meshes, voxels, octrees and K-D trees, which speed up the searching process. With these type of operations it is possible to extract additional information like fitting planes and lines, and normal vectors for example. Many laser-based odometry methods rely on this type of data representation as features of the point cloud in the matching stage, avoiding storing and matching a substantial number of points, which can be costly. Nonetheless, feature extraction can also associate a considerable computational load, therefore it is important to balance both cases. Additional point cloud processing includes projections into 2D image planes, which is particularly attractive when using neural networks. Common correspondence techniques can be of point-to-point type, such as Iterative Closest Point (ICP) [45] and derivatives [46, 47, 48]; distribution matching type, as Normal Distribution Transform (NDT) [49], or even resort to neural networks to carry out the correspondence process. Once the overlap error between successive point clouds is minimized, the transformation is estimated and then adjusted in a final optimization step. As noticeable, the process shares many similarities with visual methodologies, with the advantage of being unaffected by lighting conditions, while measuring depth directly. Like visual approaches, point cloud odometry techniques must be architected to operate in real time, especially if used in an Autonomous Driving context. In recent years, LO has demonstrated superior accuracy over visual odometry, from which the following works stand out by the reported results and/or innovative approaches.

Even though dated from 2014, LOAM, by J. Zhang and S. Singh [50], is still a top-performer amongst point cloud-based odometry algorithms. This method addresses both odometry and surrounding mapping at different frequencies, since these problems have distinct roles and requirements, yet complementary. The odometry function operates at a higher cadence and generates low fidelity estimates, while the mapping module is slower and adjusts the odometry estimates while generating and refining a global map. Figure 2.10 illustrates the block diagram of this method. In the odometry model, point cloud features are extracted in a first stage, selecting and grouping points into sharp edges and planar surface patches, according to well-defined criteria. Along the LiDAR sweep, the captured points are progressively projected onto the previous frame assuming constant angular and linear velocities of the sensor. As features are being extracted and matched, new estimates are produced by a variant of ICP that combines point-to-line and point-to-plane metrics. When the sweep is completed, the coarse estimates allow the new point cloud to align with the previous ones, where the mapping module refines the alignment, adding to the map and producing a pose estimation with increased accuracy, passed on back to the odometry module again. This two-frequencies architecture is very common among SLAM algorithms and some other works in the field of odometry, which will be discussed below. LOAM still maintains state-of-the-art accuracy (t_{rel} of 0.88% on KITTI and ranking #3 on KITTI scoreboard, as of January 2022). In addition, experiments showed the potential of an IMU integrated into the framework. This sensor provided reinforced references to align and remove the point cloud distortions while

increasing pose accuracy. Furthermore, LOAM's contributions served as a landmark for many subsequent works, being frequently used as benchmark within laser-based methods.

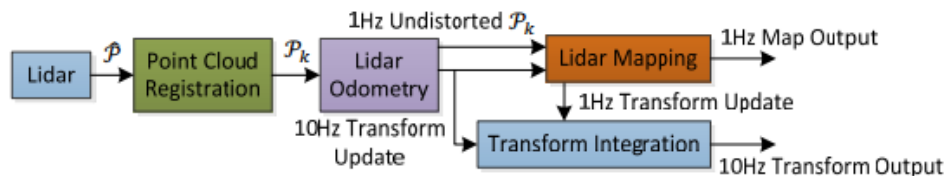


Figure 2.10: LOAM block diagram [50]. By decoupling Odometry and Mapping, LOAM produces outputs at two distinct rates: Pose transforms at 10Hz and refined each second; map outputs at 1Hz.

Recently, F-LOAM [51] tries to reduce the computational burden by transforming LOAM's iterative processes into a two-stage distortion compensation method. It also uses special features such as edges with higher local smoothness and planar features with lower smoothness, which are good for matching. With such efforts, this method achieves a 20Hz cycle frequency on a low-power embedded computing unit. Moreover, Zheng et al. [52] proposed a relevant algorithm focused on computational efficiency for Autonomous Driving scenarios called ELO. It is common for methods like LOAM, for example, to encode point clouds in a tree-based form, which while effective in terms of searching, becomes somewhat limited in large-scale point clouds. In order to maximize efficiency, this method proposes a projection of LiDAR measurements onto a spherical image, restoring the neighborhoods between points in a direct way, promoting a faster search. The problem arises in the fact that in autonomous vehicles, LiDAR sensors capture a significant amount of ground points, which, by the spherical geometry, become too far away in projections. As a way to take advantage of these points and not produce erroneous estimates, the ground points are projected in a top-down bird's-eye-view perspective. With the ground and non-ground points properly segmented and with the proper 2D projections, the application of the search methods for frame matching becomes much more efficient. Besides that, this methods only keeps three local two-dimensional maps for pose optimization, being memory efficient as well. In terms of results, this approach compares favorably to other state-of-the-art techniques, with $t_{rel} = 0.50\%$ and $r_{rel} = 0.18\text{deg}/100\text{m}$. However, ELO's major contribution ends up being on computational efficiency. This method surpasses all available alternatives, while running at 169 frames per second on a common laptop and reducing 95% of LOAM's run-time. Furthermore, ELO was tested on NVIDIA Jetson AGX, commonly used in autonomous vehicles, only needing 27ms of execution time. For these reasons, it is considered as the most suitable LiDAR Odometry method for Autonomous Driving in the review conducted by Jonnavithula *et al.* [44].

Another recent approach that relies on the basics of ICP is MULLS (Multi-metric Linear Least Square) [53] which provides an efficient, low drift 3D SLAM system. This architecture is especially designed to be independent of the LiDAR's specifications, not needing the conversion of the LiDAR data to rings or range images. Firstly, geometric feature points are extracted and encoded, distinguishing between several categories such as ground, facade or pillars. The next step involves the ego-motion estimation by multi-metric linear least square ICP, based on the selected features,

which is modified to increase the accuracy and efficiency. This variation has four essential steps. First, point correspondences are determined within each feature category. After that, weights are calculated for each correspondence considering several factors such as point intensity. Then, according to the point correspondences and calculated weights, the transformation estimation is computed. Finally the authors use statistical metrics to evaluate the quality of the registration procedure.

Because data from range sensors is unordered and sparse, it is more challenging to apply convolution-based DL modules that are typically employed. Taking this fact into account, while trying to leverage the benefits of autonomous learning, Cho *et al.* [54] proposed DeepLO. This method starts by projecting the incoming point clouds into a 2D space. Next, the projection representation is substituted by a pair of 2D maps containing the point cloud vertices and normals, without precision loss. The vertices map from the current instant and the previous one are passed to a fully convolutional network, from which it is extracted a feature vector of the respective pair. The same applies to the normal's maps, which are fed to another similar network in parallel. The two resultant feature vectors are then summed and passed to a third neural network which then predicts the motion parameters. The full network is trained either in a Supervised or Unsupervised way, depending on data availability. Supervised learning is performed by direct comparison with the ground truth, while Unsupervised learning is achieved via an error function that incorporates an adapted version of ICP and field-of-view error on the reprojected data. DeepLO was tested in KITTI dataset, with both learning methodologies. With supervised learning, the network was able to achieve extremely low translation and rotation errors ($t_{rel} = 0.1\%$) in training sequences. Yet, on evaluation, this method performed poorly ($t_{rel} = 9.59\%$), thus illustrating a clear case of overfitting and giving rise to the environment generalization problem. In the unsupervised case, the results were slightly better ($t_{rel} = 6.56\%$), but also far below from other state-of-the-art techniques.

The work of W. Lu *et al.* [55], DeepVCP, presents an end-to-end DL architecture for point cloud registration. This method extracts features from point clouds with some semantic meaning, relying also on a weighting layer to properly segment non-moving points. These are then passed through another layer to extract information about the respective neighborhoods. Due to the sparse nature of point clouds, classical alternatives like ICP lose some accuracy by associating points with points directly. Instead, DeepVCP is able to generate a virtual correspondence of the points from the source to the target, increasing precision. It also reduces the number of iterations of the process to just one, by predicting correspondences instead of searching them. The match between point clouds is then compared with a ground-truth that tunes the network by back-propagation. The results of this algorithm were evaluated in Apollo-SouthBay dataset and the KITTI training sequences, outperforming all DL-based methods previously mentioned in the last paragraphs. Nevertheless, by being a method focused on point cloud registration and not on odometry, this technique has a very high run-time, becoming unfeasible for real-time applications.

Q. Li *et al.* [56] proposed an alternative method for laser-based odometry, LO-Net. This method starts by passing from point cloud format to matrix, by projecting the points with cylindrical coordinates. Then, for each point, its respective normal vector is calculated. Both the matrix

containing the normal vectors from the current instant, and the one from the previous LiDAR scan are passed to a Siamese network (SNN) whose outputs are combined and fed to a convolutional network that estimates the ego-motion parameters. This process is illustrated in figure 2.11. The full network is trained in supervised manner. Furthermore, this algorithm contains a mask estimation module for dynamic objects in the scene, as well as a mapping block to further refine the estimates through scan-to-map matching. Unlike DeepLO, the disparity between training and evaluation sequences was not that pronounced. In fact, in KITTI dataset, mean t_{rel} on evaluation sequences is equal to LOAM's (0.83%). Moreover, without any further training besides the previous on KITTI dataset, LO-Net was tested in FORD dataset, showing promising results by surpassing LOAM by 27% in the same metric. In view of these outcomes, it is reasonable to say that LO-Net is the learning-based technique that better approximates to knowledge-based methods (see Section 2.4).

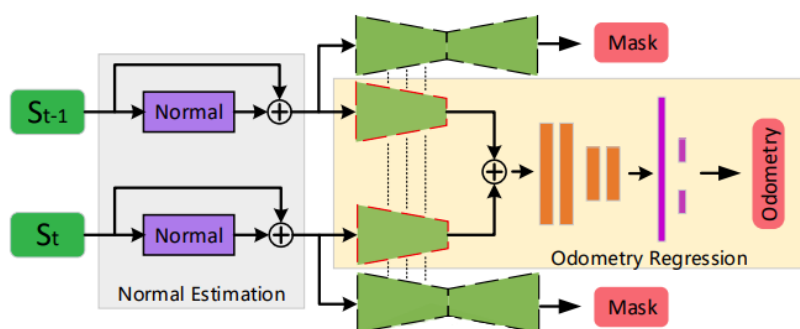


Figure 2.11: LO-Net odometry network diagram [56]. S_{t-1} and S_t denote the LiDAR scans from previous and actual time steps respectively. Upon normal estimation, both normal maps are passed to a Siamese network (dark green), whose parameters are shared, for feature extraction, being later concatenated for odometry regression.

At last, some other interesting methods include CT-ICP, by Dellenbach *et al.* [57], which currently ranks first on the KITTI Odometry leader-board among open-source methods³. This algorithm estimates initial and final positions for each LiDAR scan while performing an elastic scan matching in between. Unlike similar methods, the final pose does not necessarily correspond to the initial pose of the next swipe, providing elasticity and robustness to more abrupt variations in sensor movement. The sweep points are projected into a local map which is then utilized to generate a finer estimate, using an ICP-based technique. In parallel, this method provides a complete SLAM, refining the pose estimates with loop closure.

2.3 Multi-Modality

The environments in which Autonomous Cars move are highly complex. Just like humans, autonomous agents can benefit from an extended and multi-modal perception by combining different

³On January 2022

sensors, such as cameras, LiDARs and IMUs. Most automobile manufacturers today equip their vehicles with a wide array of sensors around them, often called the *Perception Belt*, as can be seen in figure 2.12.

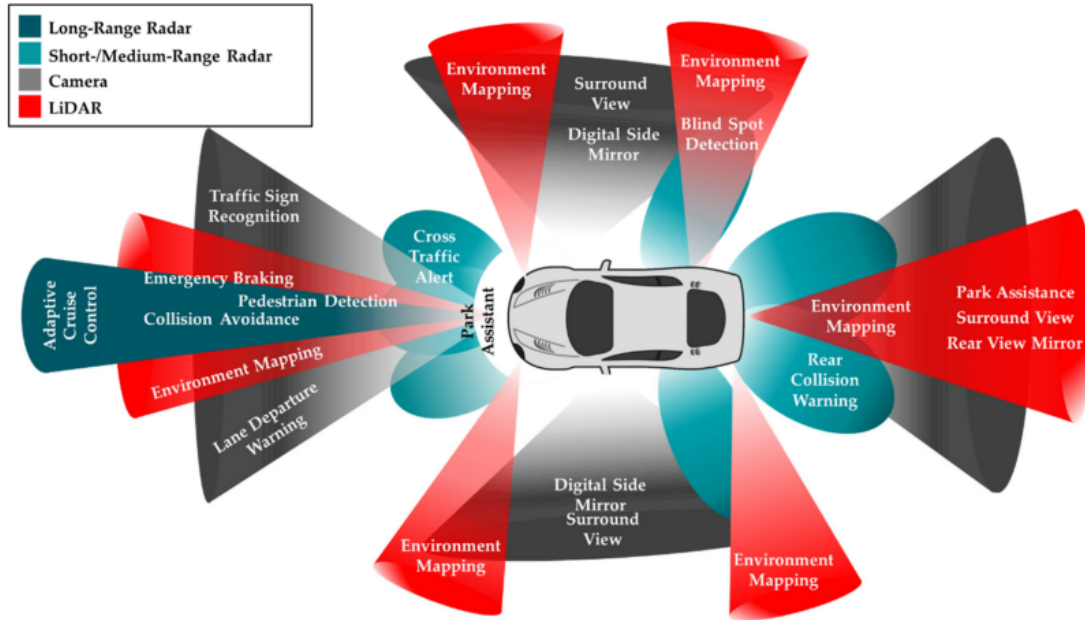


Figure 2.12: An example of used sensors and respective placement and tipology in autonomous vehicles - The *Perception Belt* [58].

Fusion's purpose is that the collected data from distinct sensors will allow to capture more complete and redundant information from surroundings, which will in turn enable less uncertainty in state estimation, leading to better decision taking. But also to strengthen the system by reducing the failure cost of individual sensing modalities. In example, visual odometry may suffer from hard illumination variations, poor lighting conditions and texture-less environments; LiDAR struggles with wide-open spaces, as motorways or long tunnels, and with adverse weather conditions; and IMU's trajectory estimates tend to drift very quickly if not adjusted periodically. Thus, having more than one modality should compensate for eventual short or long term failures.

In the autonomous driving context, most of the literature explores multi-modal systems mainly for object segmentation and classification tasks, frequently associated with DL techniques. Nonetheless, there are some shared aspects with motion estimation, such as fusion categorization. One such classification is based on the instant the merge takes place within the context of the system framework (figure 2.13). In early fusion, data is merged in raw stage, before any pre-processing, usually through projections of one or more sensors to the input space of another. Typically this approach is associated with low computational costs, but is highly dependent on inter-sensor spatial and temporal calibration [58]. Late fusion implies an after-processing merge of data. It entails lower complexity and increased modularity, however, it incurs higher computational expenses and blocks the use of potentially valuable intermediate features of each data type [58]. One of the most significant limitations in late fusion is relying only upon pose estimates, which confers on

the fusion layer an abstraction level that can be limiting in certain situations. Finally, *intermediate fusion* is the more comprehensive technique, as it can be deployed in many possible ways, depending on the system's architecture. Data is fused after some processing, hence being also known as feature-level fusion. In the context of odometry, one can classify how data is used to produce the system's output, namely whether LiDAR measurements complement the imagery and estimate ego-motion with visual odometry or vice versa; or whether the two types of odometry are carried out separately and fuse at a higher abstraction level of the system's framework [59]. The systems who follow the first approach are usually denoted as tightly-coupled, whereas the others are called loosely-coupled [60]. In these circumstances, images and point clouds are usually the primary source of odometry estimates, whereas IMUs and GPS systems provide expedient priors and/or trajectory drift corrections.

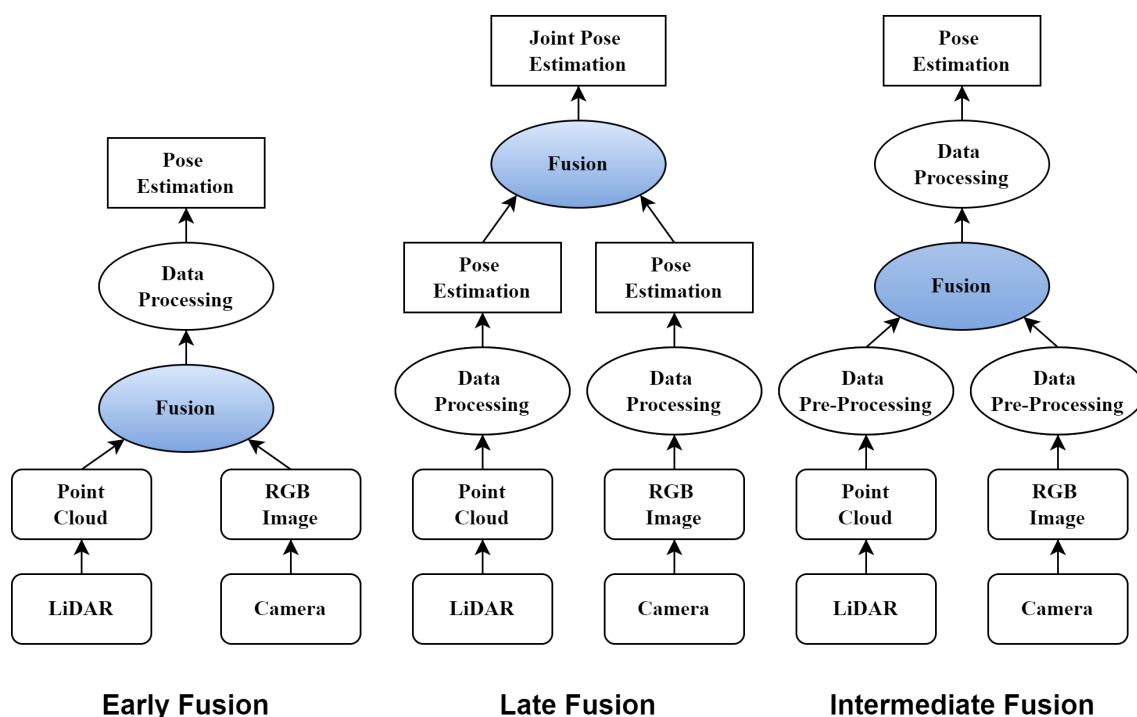


Figure 2.13: Process illustration of each data fusion technique: early, late and intermediate.

The current technology of digital cameras allows them to have a very competitive price and reduced size and weight compared to other sensors. Cameras record high resolution images of the surrounding scenes, by extracting color and texture information, which are especially effective for detecting contours and objects, and, in the case of VO, to identify points of interest that can be easily tracked in sequential frames. However, cameras are extremely dependent on environment conditions and illumination, facing some issues in scale recovering too. In turn, LiDAR sensors can retrieve native depth information and typically offer a more wide-reaching spatial coverage. The number of channels commonly ranges from 16 to 128, and refresh rates may vary in a range from 5 to 20 Hz. Yet, rain, fog and snow can negatively impact LiDARs performance by up to 25% [61], so it is of great importance to take this phenomenon into account. Some studies,

as in [62, 58], demonstrated laser beams' wavelength to impact the adverse effects of weather conditions.

For motion estimation, this pairing can be rewarding since cameras provide valuable information about the scene's texture, complemented by the sparse but precise distance measurements of LiDAR, which also has a wider field-of-view [58]. Furthermore the system can also benefit from the integration of an IMU sensor, which measures the vehicle's relative linear and angular velocities. This type of sensor can, for example, provide pose priors, help identify moving objects alongside camera/LiDAR's input [7], or even contribute in removing point clouds' distortions due to motion. Sensory fusion in odometry is then an appealing opportunity, therefore, some valuable works in this area will be presented and discussed throughout this section.

The work presented by W. Wang *et al.* [59], DV-LOAM, is an example of how to combine LiDAR and camera data at various levels for improved ego-motion detection. Derived from DVL-SLAM [63], and similar to other SLAM methods, DV-LOAM is composed of a front-end and a back-end part. The front-end part is in turn divided into a VO block and a LiDAR mapping module. The first receives sequential images enhanced with sparse depth measures from the LiDAR (figure 2.14), in which the first motion estimates are produced by a direct patch-based method inspired on DSO [37]. The estimates are further refined by applying a sliding window optimization technique, ensuring local consistency and real-time performance. In case the latest frame is a keyframe candidate, features are extracted from the corresponding LiDAR scan and joined to a global map, which then allows for one more refinement step in the scan-to-map process, as depicted in figure 2.15. This step allows the use of the whole LiDAR coverage angle and not only the portion corresponding to the camera perspective. The back-end module is responsible for map maintenance, including loop closure and a pose-graph optimizer, to reduce accumulated drift over long traversals.

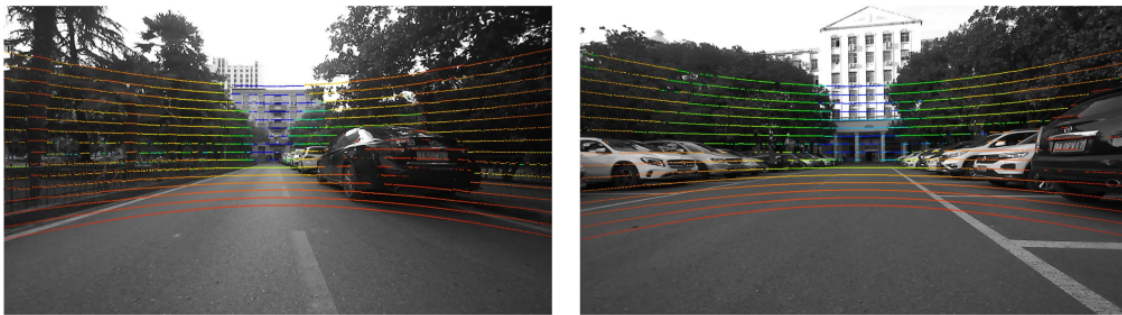


Figure 2.14: Visualization of depth enhanced images of DV-LOAM's front-end odometry module, by point cloud projection. Different colors represent different depth measurements [59].

In the KITTI odometry dataset, DV-LOAM achieves top-level results, more specifically t_{rel} of 0.80% and r_{rel} of 0.38 deg/100m. In addition, this works presents an insightful ablation study, from which some conclusions arise. For example, the visual-LiDAR odometry method applied by DV-LOAM, which is direct and patch-based, presents significant benefits over visual feature-based or laser scan-to-scan matching methods in degraded scenes with few strong features, as

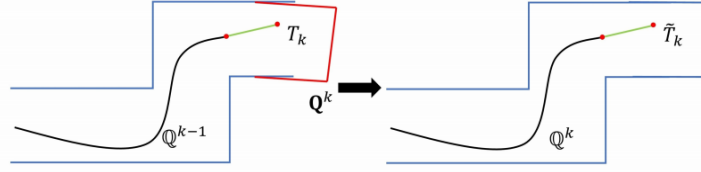


Figure 2.15: Illustration of scan-to-map registration [59]. The new LiDAR scan is adjusted to properly match the map while refining the estimated pose transform T_k . A similar process occurs in SLAM techniques' loop closure.

highways. Nonetheless, feature matching is relatively more stable, showing superior accuracy in both translation and rotation in urban scenarios, for instance.

From the same authors of LOAM, V-LOAM [64] is intended to mitigate LOAM's reliance on smooth motion, which could be compensated by an IMU. The integration of the camera into V-LOAM allows visual odometry to serve as a prior to more refined laser odometry estimations, while handling rapid motion. Analogous to LOAM's architecture, V-LOAM also operates with a dual-frequency model. As seen in figure 2.16, the two models share many similarities. The VO block computes transforms using a feature matching method between frames at a frequency of 60 Hz. It uses image points whose depths are either measured directly by LiDAR or obtained by triangulation. Concomitantly, when each LiDAR sweep is complete once a second, the new point cloud is undistorted assuming constant linear motion and the points are registered in a local map maintained by the laser odometry block, as it is in LOAM. The registration allows to generate an ego-motion estimation that corrects the drift affecting visual estimates. In addition, the LiDAR-camera combination supports operation over short periods of light outbreak. Despite dating back to 2015, V-LOAM still ranks #2 in the KITTI Odometry benchmark, with a translation relative error of 0.75%, as of January 2022.

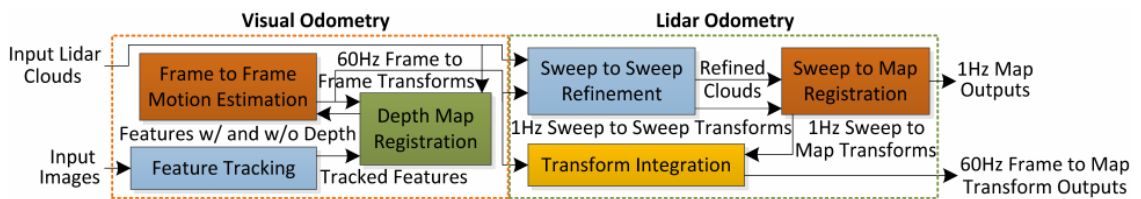


Figure 2.16: V-LOAM block diagram [64]. The two-rates principle inherited from LOAM is still noticeable (Fig. 2.10). Visual and LiDAR modules are properly segmented, as the first runs at 60 Hz and the second at 1Hz, producing Transform estimates with increased fidelity.

Unlike V-LOAM, LIMO, proposed by J. Graeter *et al.* [9], uses LiDAR measurements to supplement the camera images with depth information in a tightly-coupled manner. In a first instance, prominent points are selected from image frames, discarding points lying on cars and pedestrians, for example, to avoid the effect of dynamic objects. This is performed through deep learning semantic labeling. Next the point cloud is projected onto the image and the depth of each feature is computed through plane fitting interpolation in its neighborhood, like in figure 2.17.

The fusion stage also includes a precise set of steps that allow to efficiently segment foreground and ground points. By matching features, a pose transform estimate is generated and adjusted by a refined back-end block that applies Bundle adjustment. In this block, special emphasis is placed on the careful selection of keyframes and landmarks, as well as a robustification section to mitigate the presence of outliers using Trimmed-least-squares algorithm to avoid the effect of large residuals. LIMO achieves state-of-the-art results, with $t_{rel} = 0.93\%$ and $r_{rel} = 0.26$ deg/100m.

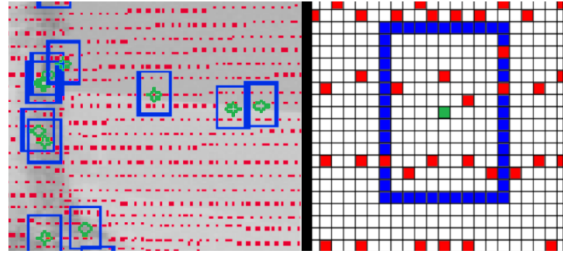


Figure 2.17: Point cloud projection into the image plane in LIMO [9]. The green dots are detected visual features and red ones are the projections of the LiDAR measurements. The blue rectangles illustrate the regions of interest to determine the neighborhood of the visual feature used to estimate the respective depth.

One further approach worth mentioning is Tightly-Coupled Visual-Lidar SLAM (TVL-SLAM), from C. Chou *et al.* [65], in which the visual and LiDAR modules run independently until a certain point in the pipeline when the data from both is merged, thus constituting a mid fusion case. Regarding the Odometry domain, the visual front-end generates a preliminary pose estimate alongside the visual residuals by an implementation of ORB-SLAM2. This estimate contributes to the calculation of the laser front-end residuals by point-to-line and point-to-plane metrics. The fusion occurs in a large scale optimization problem whose inputs are the residuals of both modules. The inter and intra-consistency between the modules is ensured in this last step by a set of constraints that include visual landmark reprojection errors, scan-to-map-registration, and cross-constraints, as both front-ends represent the same environment. In addition, this algorithm also features a technique for extrinsic calibration between sensors, and a multi-step technique for rejecting moving objects. Apart from achieving excellent results in KITTI dataset ($t_{rel} = 0.52\%$), TVL-SLAM was also tested on the KAIST dataset [66], which contains much more complex scenarios with many moving cars and extreme lighting conditions, while using a 16-bin only LiDAR. These latter tests demonstrate the effectiveness of multi-modal sensor coupling, compared to purely visual or LiDAR-only SLAM alternatives, as can be seen in table 2.1. Sequences 2, 3 and 4 are recorded in wide highways scenarios with few moving objects, posing serious difficulties to the LiDAR method. Sequences 5 and 6 present crowded urban scenarios with multiple moving cars, in which LiDAR-based techniques are more resilient. Yet, these results show not only the superiority of this particular algorithm, TVL-SLAM, but the great benefits of multi-modality.

Table 2.1: TVL-SLAM evaluation on KAIST dataset. The values correspond to the root-mean-square error (RMSE), in meters, of the whole trajectories compared with the respective ground-truths. Bold values correspond to the best result for each sequence. Adapted from [65].

No.	Sequence Name	LiDAR SLAM	ORB-SLAM2	TVL-SLAM
1	urban30-Gangnam	75.43m	120.82m	8.58m
2	urban32-Yeouido	43.81m	23.24m	9.71m
3	urban33-Yeouido	112.17m	34.04m	13.06m
4	urban34-Yeouido	377.13m	46.67m	30.95m
5	urban38-Pankyo	29.07m	63.70m	8.70m
6	urban39-Pankyo	30.30m	73.44m	8.14m

Wisth *et al.* [60] recently proposed a tightly-coupled architecture to LiDAR-visual-IMU odometry. The state estimation is formulated as a large scale pose graph optimization problem, as illustrated in figure 2.18 (a). This work tracks multiple features, both visual and LiDAR-related within a limited size temporal window. In the figure, for example, x_2 is the third system's state (encapsulating pose information), which is estimated by the constraints imposed by the tracked features at that time, namely the 3D lines l_0 and l_1 , the visual landmarks m_1 and m_2 , and the IMU measurements. The depth of visual landmarks is calculated through the projection of LiDAR points as in [9], or via stereo matching; point clouds are undistorted to the closest image timestamp to assure temporal synchronization, using the IMU's motion priors; plane/line features are extracted similar to [50], which reduces the number of points in 90% for efficiency purposes. These procedures, typical of a tightly-coupled system, are organized in the architecture of figure 2.18 (b), where the IMU works at 100Hz, the camera at 30Hz and the LiDAR at 10Hz. Although the authors did not conduct any experiences on KITTI, they tested the proposed algorithm in challenging scenarios, including wide spaces, dark tunnels and dense foliage, as well as aggressive motion. In these, the method outperformed the standard LOAM by an average of 38% and 21% on translation and rotation drift respectively. Moreover, despite the tightly-coupled architecture, the pose graph formulation allows the different modalities to impact the system independently, emphasizing performance consistency in failure cases of one sensor.

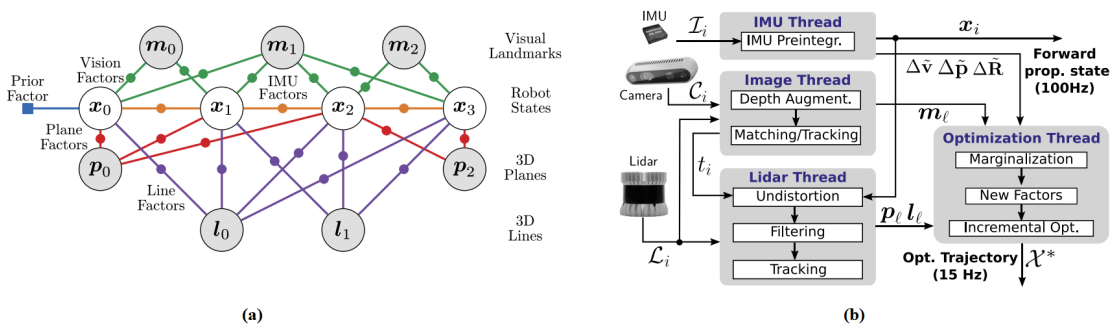


Figure 2.18: (a) Pose graph formulation of the state estimation problem in [60]. (b) Tightly-coupled architecture proposed by Wisth *et al.* [60].

The work of Ramezani *et al.* [67] composes a different class of methods whose primary focus is the estimation of agent odometry via an Inertial Navigation System (INS). The state of the agent's motion is propagated through the use of a multi-state constraint Kalman filter. At the same time, the integration of a stereo rig in the setup allows adding extra constraints to the vehicle motion by 2D and 3D feature matching methods, limiting the drift of the IMU estimates. This sensor is an asset in scenarios where matching successive images or point clouds is more challenging, as well as for reducing the impact of dynamic objects.

Finally, although not so common, there are some late fusion methods that present attractive approaches and are related to the work developed in this dissertation. In [68], the authors present an architecture that uses a wheel encoder, IMU, GPS, and visual odometry estimates for the precise localization of vineyard sprayer robots. Each sensor is filtered taking into account its error distribution and a likelihood ratio test that eliminates distant measurements from the rest, possibly affected by some sort of error. The fusion is performed by maximizing the sum of the intersection probabilities. The results show, however, that the implementation has some flaws, besides not being suitable for real-time applications. Leng *et al.* [69] fuse IMU measurements and visual odometry estimates using an Extended Kalman Filter (EKF) to estimate the location of biped robots. In particular, they rely on a Temporal Convolutional Network (TCN) to estimate the error covariance of the IMU measurements, to be used by the Kalman filter. Finally, although the focus is not odometry, Barreto-Cubero *et al.* [70] use a fully-connected neural network to fuse LiDAR, SONAR, and camera estimates of an object's distance to obtain an accurate map of the navigation platform surroundings.

2.4 Critical Review

So far, some of the most relevant and innovative works in the field of odometry have been briefly described and reviewed, including visual, laser-based and multi-modal methods. Hence, this section will contain a critical analysis over all the exposed techniques, as well as a further analysis on potential research topics and starting points given the current state-of-the-art.

Table 2.2 aggregates the results obtained in KITTI dataset by several algorithms referred in this chapter. Most data was retrieved from the respective publications and from KITTI's website⁴. The same metrics, t_{rel} and r_{rel} , were used to quantify and evaluate the performance of each technique, as both sub-tables are ordered by translation error (t_{rel}), from best to worst. These metrics were obtained by averaging the performance in each sequence where data was available. Furthermore, the evaluation metrics don't consider the training sequences for learning based methods.

By analyzing both tables, it is noticeable that very little separates the top places, regardless of its modalities or technicalities. It may also be observed that conventional techniques predominate when compared to learning-based ones. In fact, during the last years, only small improvements occurred in terms of trajectory accuracy. Eventually, the focus will have changed to robustify even further the odometry systems, and make them more computationally efficient. Unlike the run-time

⁴http://www.cvlibs.net/datasets/kitti/eval_odometry.php

Table 2.2: t_{rel} and r_{rel} metrics obtained in KITTI dataset sequences by some of the works discussed in this chapter. V: Visual; L: Laser; NDA: No data available. * identifies learning-based methods.

KITTI training sequences (00-10)				KITTI test sequences (11-21)			
	Modality	t_{rel} (%)	r_{rel} (°/hm)		Modality	t_{rel} (%)	r_{rel} (°/hm)
TVL-SLAM [65]	V/L	0.47	NDA	SOFT2 [36]	V	0.53	0.09
SOFT2 [36]	V	0.48	0.11	V-LOAM [64]	V/L	0.54	0.13
ELO [52]	L	0.5	0.18	LOAM [50]	L	0.55	0.13
SOFT-SLAM [34]	V	0.68	0.2	TVL-SLAM [65]	V/L	0.56	0.15
ORB-SLAM2 [31]	V	0.72	0.22	CT-ICP [57]	L	0.59	0.14
V-LOAM [64]	V/L	0.75	NDA	SOFT-SLAM [34]	V	0.65	0.14
DV-LOAM [59]	V/L	0.8	0.38	ELO [52]	L	0.68	0.21
D3VO* [39]	V	0.82	NDA	D3VO* [39]	V	0.88	0.21
LO-Net* [56]	L	0.83	0.4	DVSO* [38]	V	0.9	0.21
LOAM [50]	L	0.84	0.5	LIMO [9]	V/L	0.93	0.26
DVSO* [38]	V	0.89	0.2	ORB-SLAM2 [31]	V	1.15	0.27
DeepVO* [40]	V	5.96	6.12				
DeepLO* [54]	L	9.59	3.99				

or the data allocated by an algorithm, quantifying and measuring robustness is not a trivial task. The most common way to put the techniques to the test is by evaluating them in datasets containing challenging scenarios. Although KITTI is vastly used, and therefore suitable for comparing, some other datasets as KAIST [66] include more defiant sequences. In this regard, multi-modal systems are clearly superior, as shown by the experiments of C. Chou *et al.* [65] (TVL-SLAM) in KAIST, and D. Wisth *et al.* [60], which evaluated their respective works against unfavourable conditions targeted at specific sensor modalities, both standing up to the tests and showing no signs of overall degeneration, unlike visual or LiDAR only methods. In terms of efficiency (table 2.3), ELO stands out unrivaled while also performing very competitively. On the other hand, complex methods like TVL-SLAM have an increased computational load, since they rely on multiple modules running in parallel to achieve such precision and robustness, some of which can be a little demanding, such as optimization and mapping modules. One possible solution for this type of architectures would be to enhance individual function blocks, e.g. integrate ELO or a similar approach into the laser odometry front-end.

By reanalyzing the data in table 2.2, it can be seen that the latest improvements in terms of deviation from the trajectory have been residual, regardless of the type of approach taken. In fact, LOAM for example is 8 years old and still performs comparably to its most recent alternatives. It is also noticeable that the increasing complexity in recent works is not being followed by per-

Table 2.3: Execution time of some of the works discussed in this chapter.

ELO [52]	CT-ICP [57]	ORB-SLAM2 [31]	SOFT2 [36]	V-LOAM [64]
0,005s	0,06s	0,06s	0,1s	0,1s
SOFT-SLAM [34]	D3VO [39]	LOAM [50]	LIMO [9]	TVL-SLAM [65]
0,1s	0,1s	0,1s	0,2s	0,3s

formance in the same rate. This phenomenon may point out that conventional knowledge-based approaches start showing signs of maturity and may be converging to a development plateau. The two main barriers to the evolution of these techniques are closely related to the high complexity and unpredictability of real environments. On the one hand, odometry systems need to be as robust as possible to all kinds of adverse conditions, such as lighting and weather, which is directly associated with the physical sensing of the surroundings. On the other, the algorithms should have maximum abstraction possible so that perception, at software level, can withstand every kind of scenario it may encounter, whether wide-open or crowded areas, with many dynamic objects or vegetation for example. This is therefore a joint generalization problem, both at the physical and software domains.

As discussed in section 2.3, sensor fusion is the key solution to the first problem. Yet, the further development of multi-modal approaches is needed and justified in the sense that results are strongly correlated with environment representation. Furthermore, a diverse and complete set of data always provides better insights and reliability. The second problem, however, does not hold an obvious solution. In practice, and regarding the actual sensing options available, it remains very challenging to formulate a reliable real world model by hand, i.e. the problem of odometry is affected by numerous factors that are very difficult to detect and especially to generalize, given the wide variety of scenarios that can be encountered, from urban landscapes with dozens of dynamic objects, to long desert roads with potentially very few landmarks to track, in addition to the very limitations of the sensors and associated noise. Consequently, one viable option is to engineer the capacity of deep learning to capture the more intricate underlying aspects in data, that conventional methods have so far found difficult.

Nevertheless, learning-based techniques still remain a long way short of what is needed, at least for Autonomous Driving scenarios. Also, data availability is still not large enough, and existing DL structures, such as CNNs, which have dominated computer vision in the last years, are quite ineffective in learning sequential data relations. RNNs, in turn, are able to tackle this problem, but fall short on capturing image features the same way CNNs are good at. Therefore, it is frequent to find CNNs followed by RNNs, as common solutions tend to adapt existing structures rather than designing and tailor them from scratch. Alternatively, recent works such as DVSO and D3VO, are starting to approximate conventional methods, not through end-to-end architectures, in which the learning process may be too complex, but through learning-based sub-modules that aim to complement the system chain, such as depth prediction modules, feature extraction, etc. Along this line, as the employment of deep learning is highly flexible, k. Wang *et al.* [10] identify the need to bring new techniques and architectures into visual/laser odometry as a future work

opportunity. Yet, although new approaches are emerging, the authors conclude that we are in the era of weak Artificial Intelligence, but that we still need to advance into the era of strong AI, establishing more robust solutions, suitable for use in real-world cases, since deep learning has already shown promising results and excellent effectiveness in other areas.

Finally, after a detailed literature review, there were very few documented projects that used deep learning in conjunction with a multi-modal architecture in the context of odometry. Therefore, this may be one more option to be considered, justifying once again the work to be developed throughout this dissertation.

Chapter 3

Deep neural network for late fusion of ego-motion estimates

As described in the previous chapter, visual odometry techniques play an important role in the perception-related tasks of autonomous vehicles. Given the diversity of these techniques, it is important to establish common baselines for each category to further guide the work developed during this dissertation. As a part of this work, several open-source state-of-the-art techniques, including camera, LiDAR and IMU-based methods, were compared. This in-depth survey of VO methodologies included the analysis and discussion of published results, as well as a practical benchmarking of the tested techniques, both on regular and challenging scenarios and conditions. Together with the conclusions arising from the previous chapter, the importance of multimodality for the robustness and accuracy of ego-motion and self-localization estimation was again corroborated. This review also reaffirmed the current in-development state of DL-based techniques and the lack of DL approaches applied to sensor fusion in multi-modal architectures, from which the investigation topic of this dissertation arises.

In the following sections, this chapter formalizes the late fusion problem and challenges to overcome; describes how the fusion layer input data is obtained and processed; and details the research work carried out, including some proposed solution architectures and the respective supporting ideas.

3.1 Introduction

As described in chapter 2, late fusion techniques merge data in the final stage of the pipeline. In this case, the main goal is to fuse multiple odometry signals from distinct sources to generate the best possible ego-motion estimate in a given timestep. Odometry estimates are composed of 6 degrees-of-freedom (DoF) position and orientation pairs, usually in the format of $(x, y, z, roll(\phi), pitch(\theta), yaw(\psi))$ or equivalent, in which (x, y, z) describes the position and (ϕ, θ, ψ) the orientation in fixed frames (figure 3.1). Each estimate is relative to the previous pose of the

agent, i.e., it quantifies the change in position from the last timestep in x , y , and z coordinates and how much the agent rotated concerning its last orientation in ϕ , θ , ψ angles.

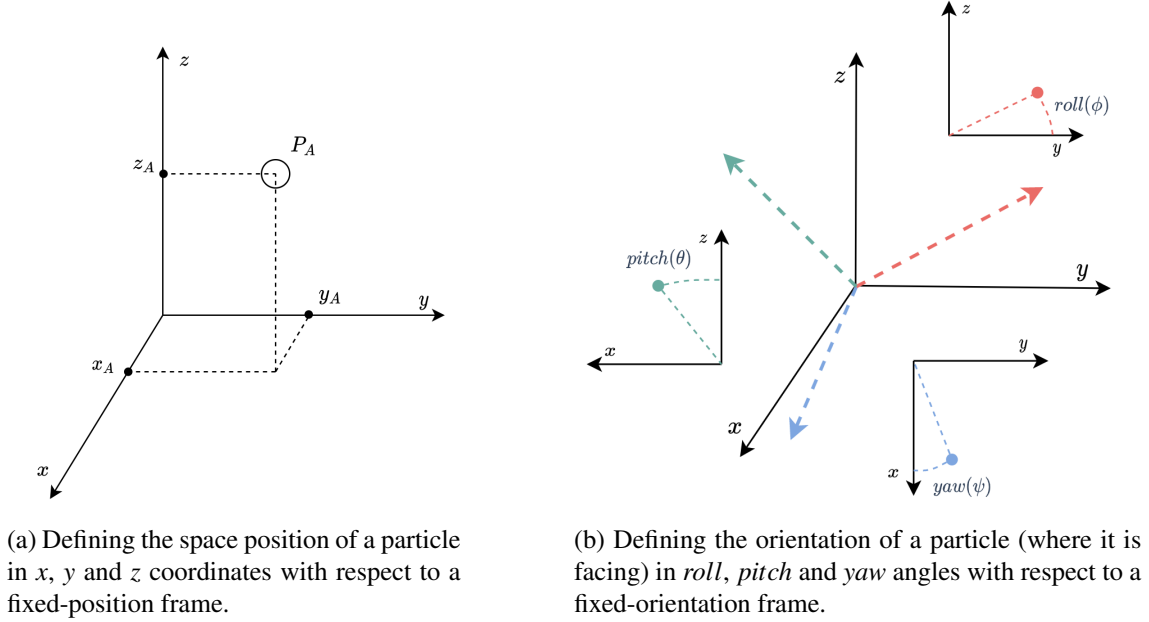


Figure 3.1: Illustration of the position and orientation components of a given pose.

Figure 3.2 illustrates the "point-of-view" of the fusion layer. Within a simple analysis, it is easy to comprehend that this particular task is not trivial, as the layer receives multiple indications as to where the vehicle moved and has to compute the best possible estimate from the received data. In fact, one of the biggest challenges of late fusion is the lack of previous information that is lost in the process. When the odometry estimates reach the fusion stage, every past step in the processing chain is invisible to the fusion module. The fusion layer does not receive information about the surrounding scene of the vehicle nor about other error-inducing conditions, only the resultant outputs. For instance, if the vehicle passes through a dark tunnel, the layer inputs from visual or satellite sources will certainly degrade quality. However, this information is absent at the layer level, as it receives an odometry estimate, just like in regular conditions. Besides heavily degraded signals, as in the previous situation or in case of a sensor failure, every generated signal is subject to a stochastic component that depends upon the sensor, the technique, and the external conditions from which it is rendered. This error source has many distinct origins. Visual methods, for example, may be affected by outliers in feature matching across frames, sudden lighting variations, pixel discretization accuracy, imperfect hardware calibration, occlusions, dynamic objects, and many more error-inducing factors. These are analogous to other sensors and signal sources and, once again, not contemplated in late fusion. For the above reasons, this problem requires additional information about the quality and behaviour of the input signals to address some issues such as: which signals are more reliable and in which circumstances; how can inputs be combined to generate the best output possible; if there are regular patterns in the signals; and several more. These topics will be covered in the following sections.

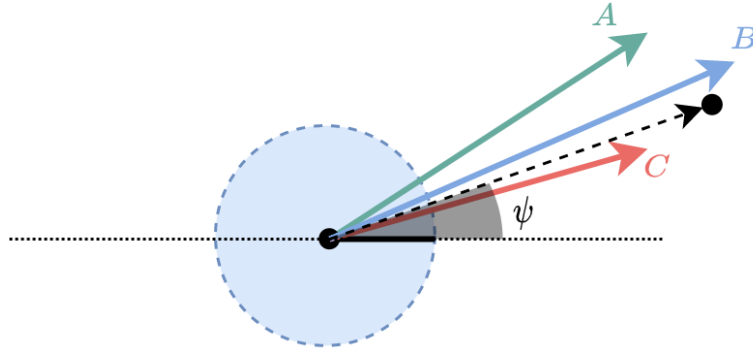


Figure 3.2: Illustration of the fusion layer's input interface (on 2D simplification). The layer receives multiple motion estimations - A, B and C (translation and rotation) and must output a reasonable estimation based on the respective inputs - dashed black line.

The operation circumstances of the fusion layer resemble the functioning principles of the Kalman Filter (KF), which is still the most used technique in late sensor fusion. As additional information, the KF resorts to the covariance matrix of the input signals to rely on more stable and precise inputs. In simple terms, the KF maintains a model of the system's dynamics which is used to predict its next state. Then, through observation of the actual state of the system, which is also prone to observation errors, the estimated system state is updated. This operation is weighted using the covariances of both the model and the observations, while the uncertainty is also propagated. However, the prediction-and-update dynamics of the KF do not directly apply here if concerning only the sensor fusion part. A similar approach can then be obtained by weighting every signal based on its variance. This simplification results in equation 3.1 which illustrates the case of scalar signals fusion.

$$F_k = \frac{\sum_{i=1}^N S_k^i (\delta^{2,i})^{-1}}{\sum_{i=1}^N (\delta^{2,i})^{-1}} \quad (3.1)$$

F_k represents the fusion output at time k , S_k^i the value of signal i in time k and $\delta^{2,i}$ the variance of signal i . Nonetheless, this formulation presents weaknesses in case of input failure, and performance is always conditioned by the quality of the best input signal.

Given the integrative nature of odometry, a robust system should produce accurate estimates to avoid the accumulation of drift in absolute poses or trajectories. As mentioned, the KF approach cannot adjust accuracy if the quality of its inputs is poor, especially if sporadic failures occur in the more accurate signals, as illustrated in figure 3.3. Furthermore, it is also convenient to produce a smooth continuous signal, i.e., with low variance, to reduce the system's upstream signal noise, especially when dealing with hardware. This is also something the simple version of a Kalman filter may not accomplish, depending on its inputs.

Motivated by the prior limitations, the conducted work aims to create a data fusion layer

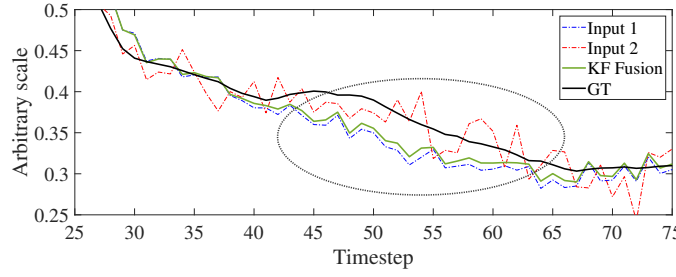


Figure 3.3: Limitation example of the weighted average mean. "Input 1" has an overall lower variance than "Input 2", but presents a small drift, while "Input 2" does not. As a consequence, the generated signal ("KF Fusion") tends to follow "Input 1", drifting from the ground truth ("GT").

that can be modularly integrated into the final segment of any framework in which the output is an agent's odometry, regardless of how that estimate is generated. This particular case intends to use signals generated by different sensor modalities, namely camera, LiDAR, and GPS/IMU, although the fusion layer is agnostic to the signal origin. This allows the overall system to benefit from multimodality, while having no limitations on the number of inputs, except for associated computing power. Figure 3.4 depicts the conceptual input/output architecture of the proposed module.

In this sense, this dissertation intends to develop a learning-based framework able to:

- Map the statistical dispersion of input signals and filter out the stochastic components (figure 3.5);
- Identify systematic errors in input signals and adjust the fusion output accordingly;
- Identify various types of sensor failure and mitigate the respective effects;
- Produce a dynamically coherent output signal as consistent and accurate as possible.

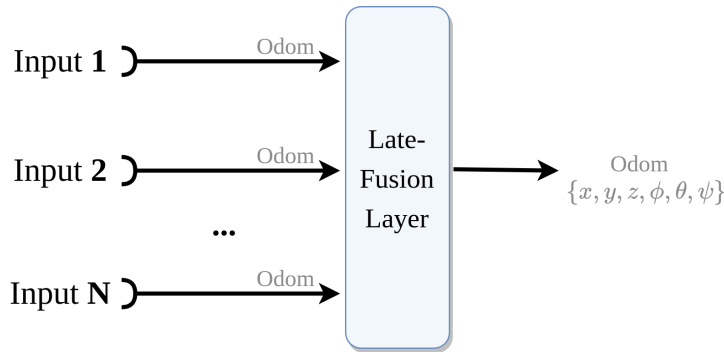


Figure 3.4: Input/Output configuration of the late fusion layer. Inputs can be of any type and number as long as given in the format of odometry estimates, i.e. x, y, z , roll (ϕ), pitch (θ), yaw (ψ) or equivalent. The produced output is in the same format.

The objective is to detect patterns in the input data and infer the same relations to unseen data, thus predicting an accurate output signal, robust to eventual challenging external conditions and

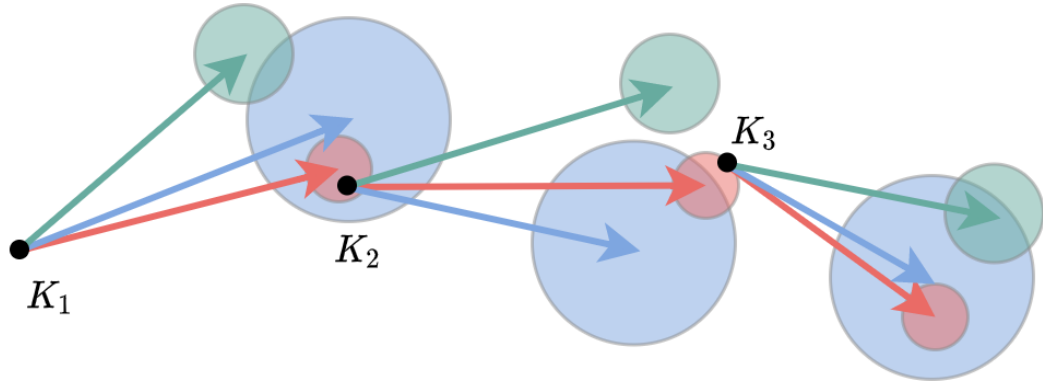


Figure 3.5: Illustration of the statistical dispersion of some arbitrary odometry inputs (green, blue and red). The black circles labeled with K_1 , K_2 and K_3 represent the true position of the agent in successive instants. Wider circles around the arrows correspond to greater dispersions and consequently lower confidence in the estimate of the respective signal. The main goal of the fusion layer is to detect these distributions and predict a reasonable joint estimate. Additionally, although the green signal has a fairly low dispersion, it has a noticeable constant offset, which can also be corrected.

sensor failure. This is the case of a classical supervised learning problem. Consequently, the next section describes all the procedures of data acquisition and pre-processing that enabled the training process of the network.

3.2 Input signals: data acquisition and augmentation

As the fusion layer works with ego-motion estimates, it is necessary to recur to odometry techniques that allow the generation of these signals. For this purpose, the following techniques were implemented:

- ORB-SLAM2 [31] - a stereo odometry method based on the correspondence of visual ORB features in selected keyframes and multi-step bundle adjustment (figure 3.6a). The mapping and loop-closing features were disabled to rely solely on pure frame-to-frame motion estimates;
- MULLS [53] - a LiDAR-based odometry algorithm that extracts and classifies features from point clouds (facade, ground, pillars, etc.) and applies the proposed multi-metric linear least-squares in the iterative closest point algorithm (ICP) for point cloud alignment (figures 3.6b and 3.6c);
- Robot Localization (RL) [71] - A ROS framework that provides non-linear pose estimates from GPS and IMU data.

As the intention is to use different data modalities, it is necessary to resort to datasets that compile camera, LiDAR, and GPS/IMU data in on-road driving scenarios. In this sense, KITTI-360 [72] meets the requirements, providing data corresponding to about 73 km of city and suburbs

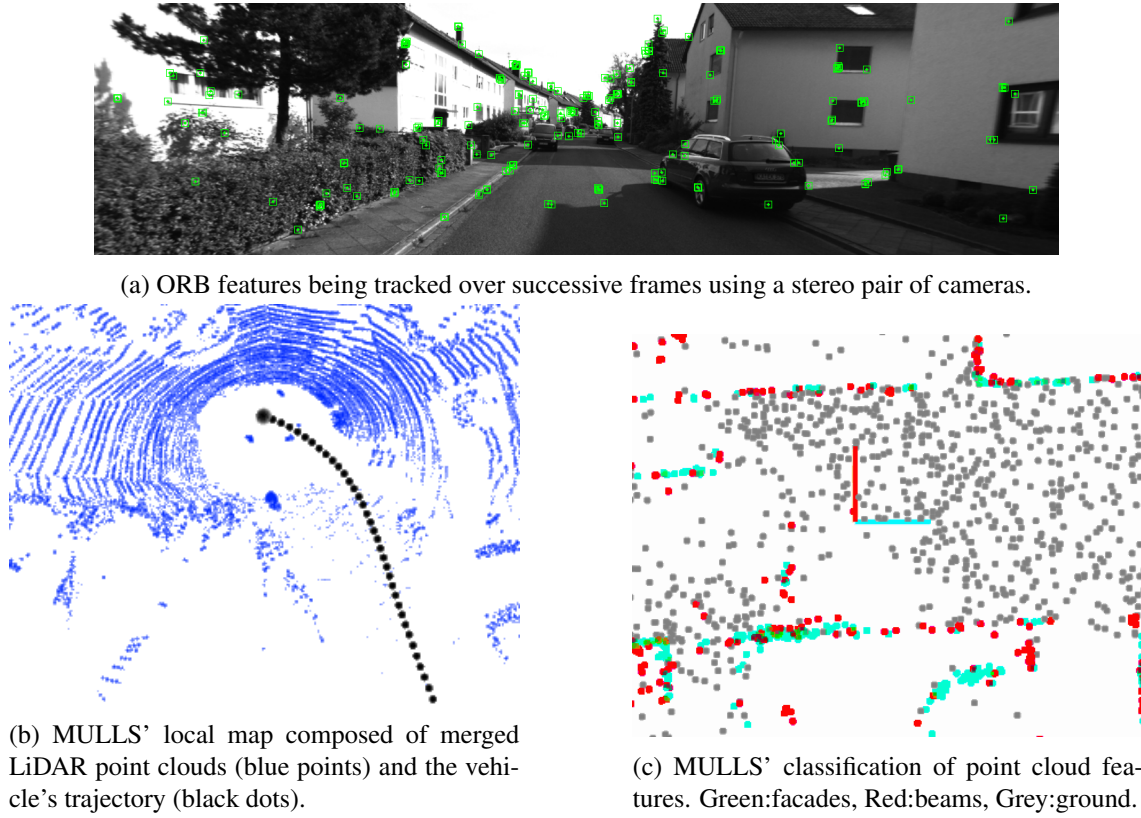


Figure 3.6: Odometry techniques utilized to generate inputs for the fusion layer - ORB [31] and MULLS [53].

traversal, containing from simple to more challenging scenarios. The latter involve the presence of several dynamic objects in the scene, areas with dense vegetation, open spaces with few features, and areas with plenty of lighting variation, such as tunnels. The data was carefully analyzed, extracting a set of sequences representative of all kinds of scenarios and challenges found throughout the dataset. These sequences were used by the aforementioned estimation techniques to generate the odometry data to be used by the network. In addition, the selected sequences were subject to a data augmentation process where various types of faults were added to the odometry signals to simulate real failure cases. The characteristics of these sequences, as well as of the data augmentation will be discussed in the next chapter in greater detail.

Since this dissertation is focused on the autonomous driving context, the 6 DoF problem can be further simplified to reduce the complexity of the fusion layer. Most of the time, with ground vehicles, the interest lies only in the vehicle's location in the horizontal plane and its orientation in the same plane. For this reason, one can consider only 3 DoF, those being x , y and $yaw(\psi)$, assuming the vehicle moves in the xOy plane. By decoupling the position from the orientation, the 3D trajectory points are aligned and projected onto the xOy plane. Analogously, the initial orientation frame must be rotated so that its Z -axis is normal to the trajectory plane (position points in xOy), concentrating most of the Roll, Pitch and Yaw variations in Z -axis (Yaw). Figure 3.7 aggregates some examples of the estimates generated by ORB and MULLS that are used to

feed into the fusion layer.

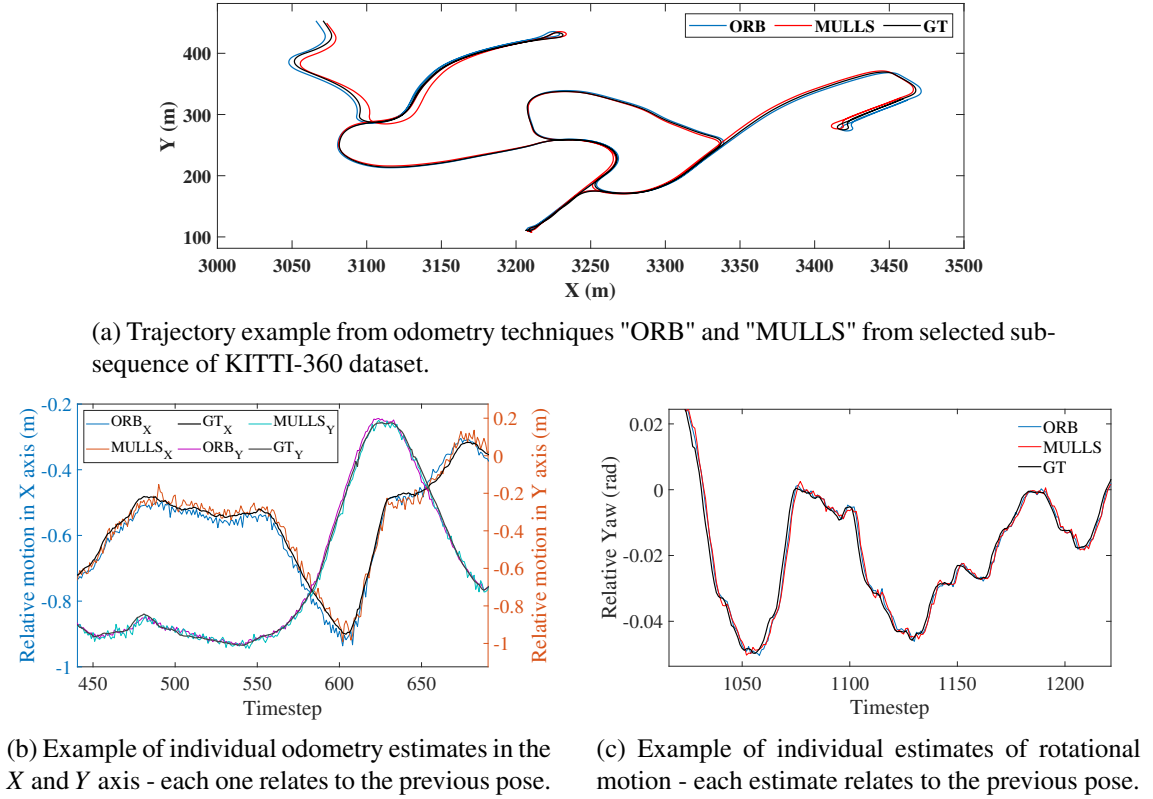


Figure 3.7: Example of odometry estimates from *ORB* and *MULLS*.

Failure cases occupy an almost insignificant portion of the total data. Consequently, a data augmentation step became necessary to prepare the network for eventual real-case situations of sensor malfunction. These modifications encompass the addition of random noise to the input signals. To simulate different types of abnormal behavior, the signals were affected by different types of noise, as described below:

- White noise with zero mean and variable variance uniformly distributed (figure 3.8a). This simulates generic malfunctions, either related to internal or external disturbances;
- Constant signal (figure 3.8b). The signal is static in zero or other arbitrary value and simulates the stoppage of a given input of the fusion layer;
- White noise with non-zero mean (figure 3.8c). This simulates a biased signal coming from a poorly calibrated sensor, for example.

3.3 Data processing

As one of the most significant limitations to this problem is the lack of information contained in the input signals, it becomes crucial to rely on additional measure that help to describe the signals

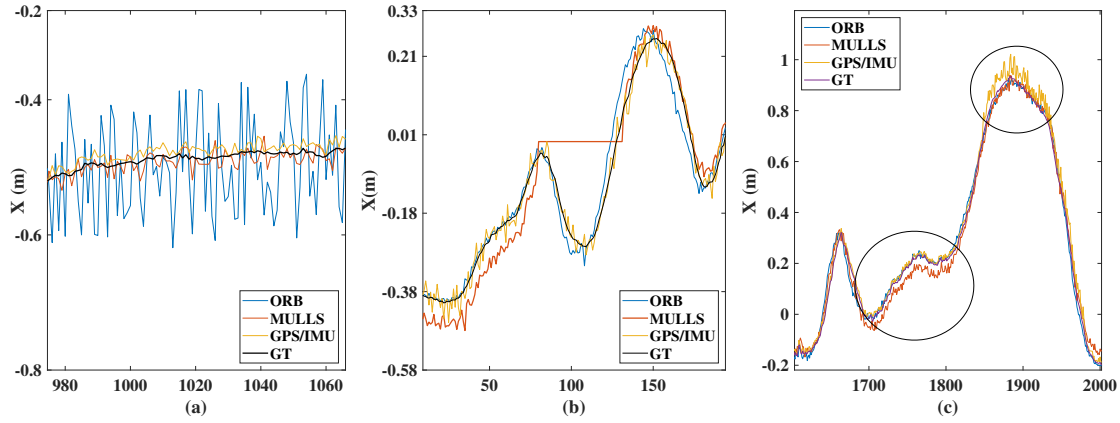


Figure 3.8: Data augmentation - synthetic signal errors.

(a) Example of uniform white noise applied to ORB.

(b) Example of constant-type error applied to MULLS.

(c) Example of non-zero mean type error. Both MULLS and GPS/IMU are pulled down or up from their original state.

in a more comprehensive way. By looking only at the value of the inputs in the current timestep, the temporal behavior of the signals is neglected. This kind of information becomes very relevant for multiple reasons. First, although there is no direct correlation between precision and accuracy, it is assumed that a truthful signal does not have an unstable behavior. If a signal is consistent, i.e., has low variance, it will generally be associated with a higher degree of confidence in its estimates. Secondly, measures that quantify the dispersion of signals and their smoothness help detect any flaws in the input signals if they have values outside the standard range of their regular operation. On the other hand, these measures promote the generation of a consistent signal, which is bounded to a certain extent, avoiding unrealistic "jumps". Finally, adding more measures to the input layer allows the network to create a representation of the data in an extended state space with higher resolution, giving the network tools to more easily detect patterns in the data, either from normal operating conditions, specific maneuvers or signal failure cases.

As it is common in forecasting problems [73], a practical approach is to recur to previous signal values in a sliding window fashion, which in practical terms is the application of a Finite Impulse Response (FIR) filter over the data sequence. This allows retrieving some additional information in the time axis, as depicted in figure 3.9. For this particular case, it is important to compute metrics that will allow the network to learn the statistical distribution of the input signals while filtering out noisy signals with no practical value. Quantifying the dispersion and entropy of the signals is a way to correlate to the respective accuracy and precision. The utilized measures are detailed in table 3.1. This way, for each input signal, composed of the estimates in X, Y and Yaw, three more statistical measures are associated with each signal component: variance, mean and average of its derivatives, calculated over a sliding window of arbitrary length (N), which starts in the actual timestep and captures the previous $N-1$ estimates. In addition, for each pair of inputs of the same component (for example, the X-estimates of ORB-SLAM2 and MULLS), there

is also a covariance measure associated that evaluates the similarity of behavior within the pair. Thus, for the case in hand, where three distinct odometry inputs are used, ORB-SLAM2, MULLS, and RL, the fusion layer utilizes three motion estimates plus nine statistical measures for each. Additionally, nine correlation measures are added, composing a total of 45 inputs to the network, which is described in the next section.

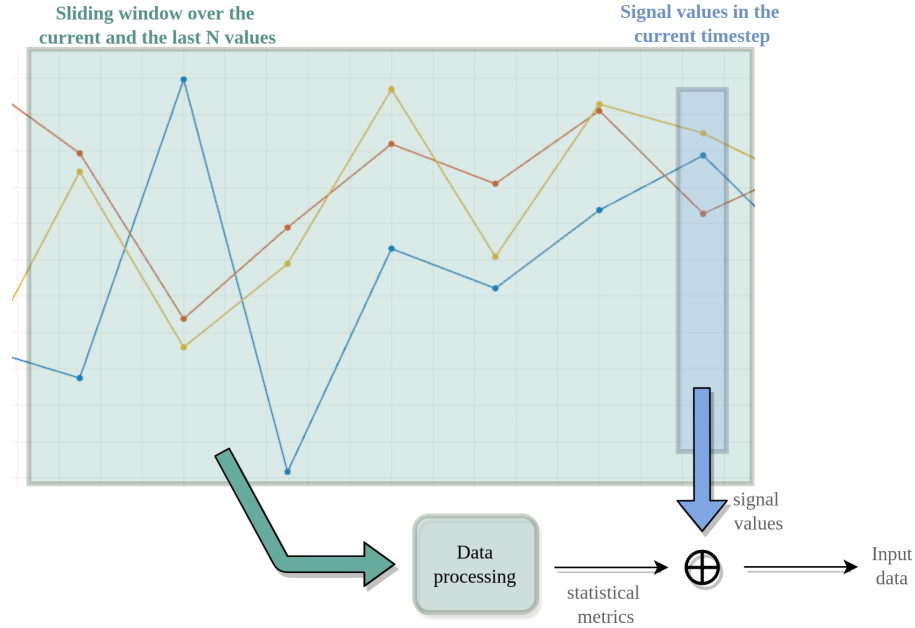


Figure 3.9: Data processing procedure. For each instant, the network receives as inputs the current signal values and multiple statistical metrics calculated over a sliding window of past frames.

Table 3.1: Complementary statistical metrics used as inputs in the fusion layer.

Statistical measure	Description
Variance $\delta(A)_k^2 = \frac{\sum_{i=k-N+1}^k (A_i - \bar{A}_k)^2}{N-1}$	The second statistical moment quantifies the dispersion of a signal and consequently how noisy it is, while helping the network to map the statistical distribution of the inputs. It is also a good indicator of signal failure if the variance presents a value that is off the common limits, either below the under limit, indicating a constant-failure type, or above the upper limit, indicating a noisy signal. Although usually one would measure the variance over the distance to the ground truth, which is not possible during inference, if the sliding window is small enough, it is reasonable to assume that the vehicle movement is constant and thus it is possible to compute the variance over the actual odometry values.
Mean $\bar{X}_k = \frac{\sum_{i=k-N+1}^k X_i}{N}$	The mean is also important to map the statistical distribution of the input values, while providing some guidance related to the motion consistency between timesteps.
Mean of derivatives $md(A_k) = \frac{\sum_{i=k-N+2}^k A_i - A_{i-1}}{N-1}$	The mean of derivatives is an alternative metric to describe the vehicle motion. It measures the average variation in signal values between timesteps in the respective window. This metric provides a sense of the movement amplitude and trend.
Covariance $cov(A_k, B_k) = \frac{\sum_{i=k-N+1}^k (A_i - \bar{A}_k)(B_i - \bar{B}_k)}{N-1}$	The covariance can be seen as a measure of the similarity between two signals. It becomes important to encode signal failures as the covariances between the erroneous signal and the remaining signals decreases towards zero.
$\delta(A)_k^2$: Sample variance of arbitrary signal A in timestep k and window size of N samples. $\bar{\delta}(A)_k$: Mean of arbitrary signal A in timestep k and window size of N samples. $md(A_k)$: Mean of derivatives of arbitrary signal A in timestep k and window size of N samples. $cov(A_k, B_k)$: Cross-covariance of arbitrary signals A and B in timestep k and window size of N samples.	

3.4 Fusion layer architecture and composition

The final step was to design a neuronal architecture capable of capturing and modeling the intended relationships between the data. Since the temporal dimension of the input data is already captured through statistical metrics in a sliding window, it becomes unnecessary to use overly complex architectures. On the other hand, since data fusion is not a problem commonly addressed by learning approaches, there are no standard structures to be applied, such as convolutional networks for computer vision, for example. Therefore, similar to [74] which describes an analogous problem, a fully-connected neural network (FCNN) was proposed. The complete odometry chain is shown in figure 3.10.

The proposed network is composed of 6 hidden layers, the first three having 64 neurons, the following two, 32 neurons, and the last one, 8 neurons. In [74], the presented NNs are composed of 5 to 50 hidden layers, depending on the input data. In the case of this dissertation, 6 hidden layers showed good results without overfitting, providing that the statistical metrics in the input

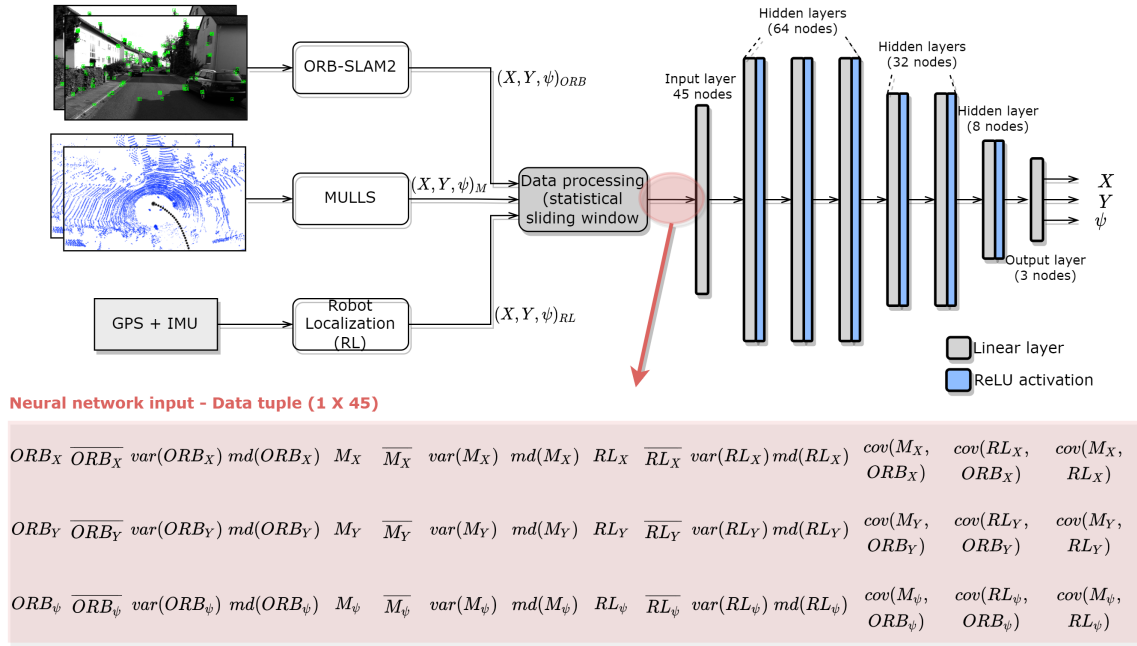


Figure 3.10: Full late fusion architecture. The odometry signals are generated by ORB [31], MULLS [53] and RL [71], using RGB information, point clouds and IMU/GPS data. Over the N most recent estimates from each method, a set of statistical measures is calculated, which together with the ego-motion estimates constitute the inputs to the illustrated fully-connected neural network. At the final end, the net produces an output in the form of odometry.

data take some work away from the NN, hence requiring less complexity. The ReLU activation function, which takes the form $output = \max\{0, input\}$, is used after each hidden layer. Although dropout layers were considered, the best results ended up not needing them. The concept behind this architecture is that the network can expand the representation space of the input data and thus model the behavior patterns of the input signals under different operating conditions, producing at the output the most plausible estimate possible.

To improve the network's ability to capture data patterns, other alternatives were also explored. There are some relationships in the inputs that are unique to the problem at hand and that can be used to provide the fusion layer with better predictive power, namely, it is known that successive odometry estimates must respect a consistency condition that concerns the vehicle dynamics model. This phenomenon should limit the amplitude and shape of the movements that can be taken, but also the relationships between successive movements that must respect the continuous kinematics models (consistency in the acceleration of movements, limited speeds, etc.). For this purpose, the research extended to structures capable of relating data sequentially, going beyond the statistical measures in the sliding window and including all inputs of the current instant plus inputs from previous instants. A temporal approach also opens up the possibility for the network to retain some knowledge about the behavior of the signals during certain maneuvers, e.g. turns or inversions. In this sense, Recurrent Neural Networks (RNN) are memory structures capable of retaining past information and therefore relate inputs in the temporal dimension. The inputs are

no longer a tuple, as in the case of the FCNN in figure 3.10, but at a given instant, a set of stacked tuples of the current instant and the $N-1$ previous ones (for a sequence of N timesteps).

LSTMs are a special type of RNNs that include "forget gates" and cell states. These structures allow the LSTMs to regulate the information that is stored in the cell memory [75], and thus to offer a superior capacity, in theory, of retaining relations in sequential data. These networks work by accumulating information from previous data by self-feedbacking an internal hidden state in every iteration. This way, in a given instant, the network works with the current data and the respective hidden state, which encapsulates past information. LSTMs can operate in a sequence to sequence mode, in which they receive a sequence of data (the order is relevant) and output a sequence with the same length. Figure 3.11 depicts a proposed architecture with two stacked LSTMs, denoted throughout this document as "FC-LSTM". The concept behind this scheme is to recur to the previously trained FCNN (figure 3.10), while dropping the last 2 layers. This allows the LSTM structure to be fed an expanded representation of the odometry inputs, taken directly from the middle layers of the fully-connected NN. Opposed to the first architecture, the inputs are not only the values for the current timestep, but also from previous timesteps, usually composing a sequence length of 3 to 5 poses. The weights in the LSTMs are then updated in order to minimize the distance to the ground truth, regarding the estimates for all timesteps in the current window, forcing the network to generate cohesive outputs. The predicted estimates from previous instants can also be interpolated for trajectory refinement.

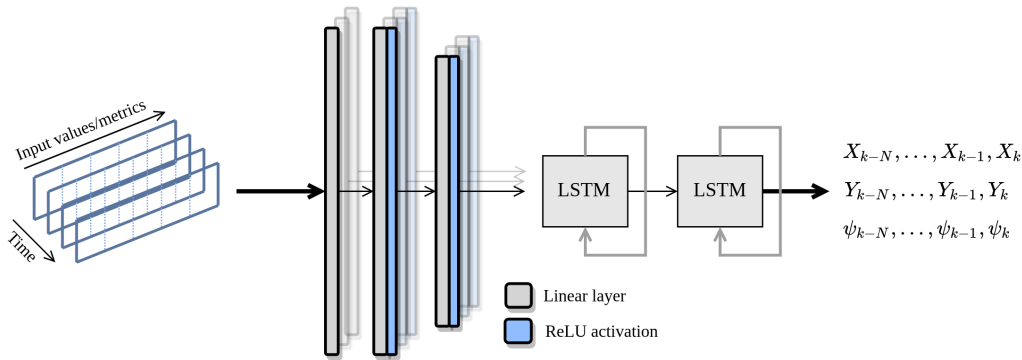


Figure 3.11: Integration of two stacked LSTMs in the original architecture. Unlike the fully-connected network, the inputs are stacked tuples from the current instant and the $N-1$ previous ones. The LSTMs replace the last two layers of the original network, predicting a sequential output for timestep k (actual timestep) and the previous $N-1$. The linear layers' parameters are preserved.

Alternatively, the model illustrated in figure 3.12, which will be denoted as "CNN-FC", tackles the motion/time constraints in a different approach. Analogously to the LSTM scheme, the input data consist of stacked tuples from various timesteps, forming a matrix-like structure. This matrix is subject to multiple convolutions with several kernels with the same size as the matrix, whose parameters are optimized in the training phase of the network. The purpose is to capture the relations among the input values, as the original FCNN of figure 3.10, but also the relations between those values in the time domain. Every kernel generates a numerical descriptor of the

input matrix. These descriptors are then flattened into a 1D vector, and filtered through a max-pool operation. This operation captures the maximum-value features in a non-overlapping sliding window, summarizing the convolution results. The remaining descriptors serve as input to a regular fully-connected network, resulting in odometry estimates from the current and past timesteps, just like in the LSTM case. The training process, namely the loss that is computed against all the ground truth values of the sliding window, is carried out in the same way. Once again, the objective behind the convolutions, like in the LSTM, is to capture the implicit motion dynamics that assure coherence in the sequence and bound the amount of variation between each instant.

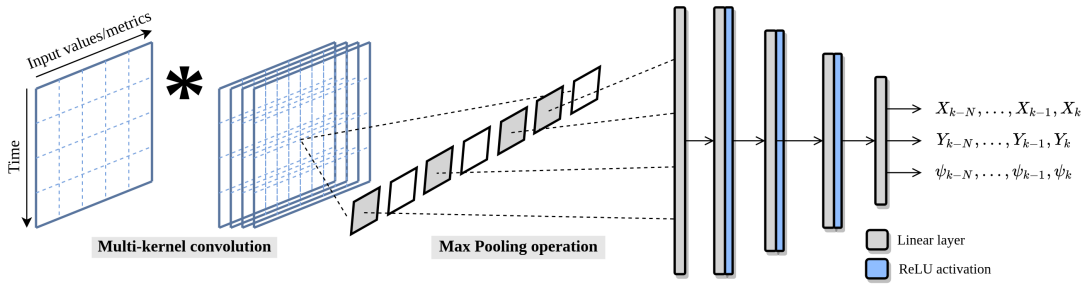


Figure 3.12: Illustration of multi-kernel convolution to extract data relations in the time domain. Unlike the original fully-connected network, the inputs take a matrix form composed of stacked tuples from the current and the previous $N-1$ timesteps. Over the matrix, the network applies a series of convolution operations with different kernels with the same size as the matrix, to capture the relations between the different signals and their evolution over time. These intermediate representations are filtered later in a max pooling stage, and fed to a regular fully-connected NN.

This chapter covered the specific scope of this dissertation, including the problem formulation and respective limitations of fusing odometry estimates in a late stage. It explored how a sliding window of statistical metrics incorporated with deep learning structures constitute the proposed solution, as well as the supporting concepts. Lastly, to enhance the original solution, two model extensions were proposed, whose principles lie in capturing the adjacent motion models of the vehicle to add coherence between sequential estimates.

Chapter 4

Results

The present chapter discusses and analyzes the practical results obtained by the proposed framework for late fusion of odometry signals. Firstly, the impact of some model parameters is evaluated by means of convergence and steady-state error. Then, the fusion network is evaluated on nine data sequences of varying difficulty containing camera, LiDAR, GPS, and IMU data in a road driving context. The results are compared with various state-of-the-art techniques, as well as with other late fusion alternatives, namely the Kalman filter. The discussion extends to the network limitations and robustness to some induced errors in the input signals, which are intended to simulate real sensor failures. This chapter also dissects the results obtained by the two extended versions of the original model that aim to capture the kinematics model of the vehicle. The outcomes are evaluated on the model's ability to produce a consistent output that captures the dynamics and smoothness of the actual vehicle motion.

Throughout the chapter, all results evaluated are from models trained with the same data. The training data comprises 18 selected sequences from the KITTI-360 [72], with an approximate number of 3000 frames per sequence. These were chosen to provide the highest possible representativeness of the dataset. On the sequences, various types of noise were applied directly to the odometry signals, as described in the previous chapter, to provide the models with a greater capacity to operate in challenging scenarios, making a total of 36 training sequences (18 original + 18 with data augmentation). For testing, nine sequences are used (none of them used in training phase), whose detailed description can be found in section 4.2. The ground truth labels are obtained by 3D triangulation of multiple scene points captured by the LiDAR and the cameras attached to the vehicle in a large-scale optimization problem. This offline operation also uses precise GPS/IMU data to generate the most accurate estimates possible. This formulation takes a several ammount of global constraints, which is not feasible for any odometry technique that operates in real-time but guarantees a reliable ground truth signal. The hardware for training the networks was an Nvidia Tensor Core A100 GPU with 40GB of RAM.

4.1 Model optimization

In the previous chapter, a fully-connected network was proposed to address the signal fusion task. Therefore, as it was designed and implemented from scratch, it is useful to perform an analysis of the impact that different optimizers and model parameters can have. The following paragraphs describe the results obtained by Stochastic Gradient Descent (SGD), Adam, and Adadelta in minimizing the Mean Square Error (MSE) of the model against the ground truth values. Each optimizer applies a different update rule in the network weights. SGD works by inferring a probabilistic gradient of the loss function with a sample of the whole data in every iteration. It usually suites the cases when the dataset has some redundancy, which is the case due to the frequency of odometry estimates that is much higher than the velocity of the vehicle and the forward-motion predominance. This optimizer is usually combined with a momentum term which can help the model to escape local minima convergence. Adam is also a common optimizer used in machine learning as it is memory efficient and promotes fast convergence. It can be interpreted as a gradient descent with momentum, weighted along with an exponential moving average of the gradient. In both optimizers, the learning rate, i.e., the amount of change applied to the model weights is crucial, and as the literature suggests, 0.001 is a good value to use in both cases. Although not so common, Adadelta proposes an adaptive method in which the learning rate is dynamically adjusted within the gradient descent.

Figure 4.1 compares the training and validation errors over 75 epochs using the three optimizers. A quick analysis makes it possible to conclude that they all cause the model to converge, with SGD yielding the best results both in training and testing, hence being a suitable optimizer for this task.

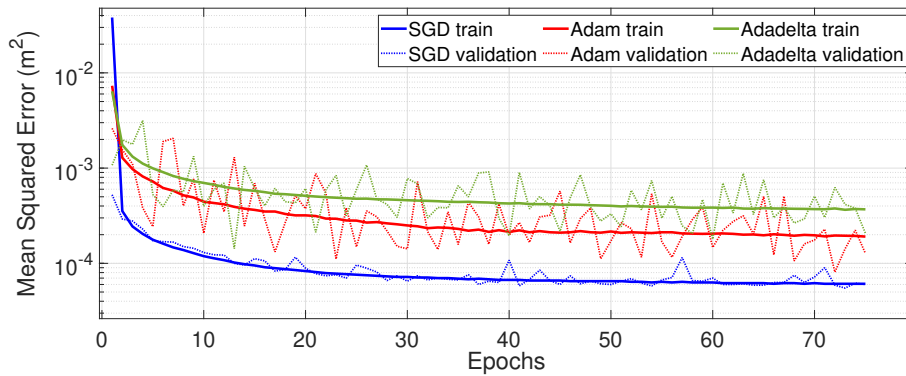


Figure 4.1: Error evolution (MSE) during network's training using SGD, Adam and Adadelta.

Another critical parameter to consider is the sliding window size used to calculate the statistical measurements of the network input signals. Since the update rate of the inputs is much higher than the vehicle dynamics variations, it is possible to assume that the car's movement can be considered constant without significant additional damage between the first and the last window measurements. As the ground truth is not accessible at the time of network inference, the statistical metrics, namely the variance of the input signals, are valuable as long as there are no significant

variations in the signal. Thus, it is necessary to choose a window length that is small enough to maintain the validity of the information retained by the variances but also large enough so that there are a number of samples that make the metrics meaningful.

In this sense, figure 4.2 evaluates the error in training and validation of seven different lengths of the sliding window during 75 training epochs. As can be seen from the graph, there is a slight tendency for the error to increase while increasing the sequence length, which is expected due to the assumptions made. Furthermore, between sequences of lengths 5, 10, and 15, there is no significant difference. For this reason, a length of 10 instants was utilized since it is a good compromise between the significance of metrics and motion linearization.

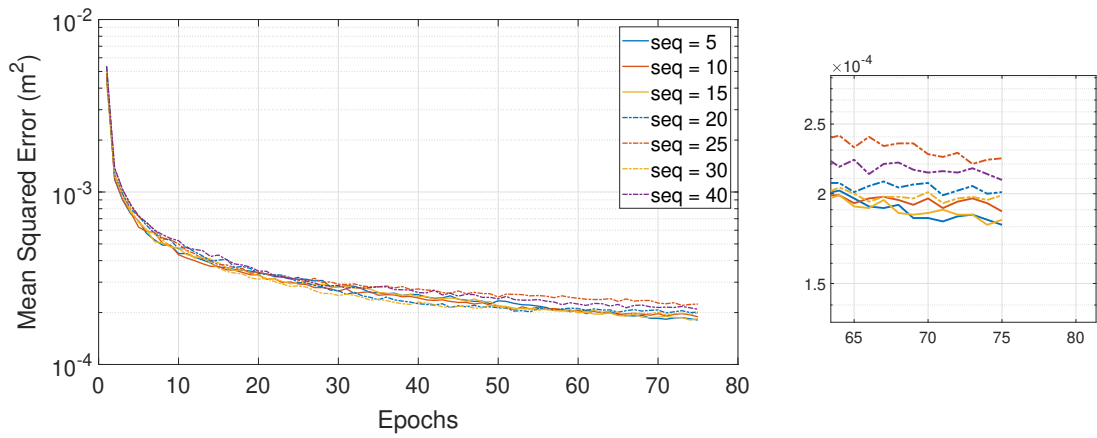


Figure 4.2: Error evolution (MSE) during network's training using different sliding window lengths.

Through the performed tests and various heuristic experiments, the model of the fully-connected network, whose results will be evaluated in the following sections, has the parameters expressed in table 4.1. Although the 3DoF pose is composed of two position parameters (X and Y) and one rotation parameter (Yaw), the following results will be only focused on the position parameters, X and Y, for the sake of simplicity of the model and shorter training periods.

4.2 Model evaluation in real road driving scenarios

4.2.1 Evaluation metrics and test data sequences

The evaluation of results is carried out according to a set of metrics that aims to quantify the deviation of the network predictions with respect to the ground truth. The metrics should assess the NN's outputs in relative terms, that is, quantifying the deviation concerning each instant independently and in terms of the produced trajectory by measuring the accumulated drift. The set of metrics to use is composed by:

Table 4.1: Model parameters of the proposed fully-connected network.

Network Parameter	
Number of signals merged	3
Number of inputs	45
Number of outputs	2
Hidden layers	6
Optimizer	SGD
Activation Function	ReLU
Loss function	MSE
Learning rate (with scheduler)	1e-3
Batch size	8
Sequence length (for statistical metrics)	10
Training epochs	35

- Euclidean mean absolute error of the relative estimates (MAE_{rel}) - provides a sense of the magnitude of the deviation to the ground truth in two-dimensional space, i.e., the accuracy of the estimates. This metric concerns motion estimates from one pose to the next. Ideally, it should tend toward zero.

$$MAE_{rel} = \frac{1}{N} \sum_{i=1}^N \sqrt{(gt_{X,i} - e_{X,i})^2 + (gt_{Y,i} - e_{Y,i})^2} \quad (4.1)$$

N is the number of estimates produced during each sequence, $gt_{X,i}$ and $gt_{Y,i}$ are the ground truth relative values in X and Y in timestep i (how much the vehicle moved in that instant), and $e_{X,i}$ and $e_{Y,i}$ are the relative motion estimates in X and Y in timestep i of an arbitrary algorithm.

- Trajectory's Euclidean mean absolute error (MAE_{traj}) - this metric measures the similarity between the trajectory predicted by the network and the vehicle's actual trajectory by calculating the average drift along the whole path. However, in the case of a single wrong estimate, this metric is highly influenced by the instant in which the error occurs, as if it happens earlier in the vehicle's course, all the remaining estimated trajectory will have degenerated, resulting in a larger error value.

$$MAE_{traj} = \frac{1}{N} \sum_{i=1}^N \sqrt{\left(\sum_{k=1}^i gt_{X,k} - \sum_{k=1}^i e_{X,k}\right)^2 + \left(\sum_{k=1}^i gt_{Y,k} - \sum_{k=1}^i e_{Y,k}\right)^2} \quad (4.2)$$

- Median of Euclidean absolute error (Med) - the trajectory error (measured by the previous metric) can be caused by several small-scale errors that accumulate or by larger errors of momentary nature. The median makes it possible to distinguish these cases since it adds information about the symmetry of the relative error distribution around its mean. Suppose

the evaluated technique has a considerable trajectory error and a low mean relative error but also has a median far from the mean. In that case, it is likely that it has been affected by a large-scale transient error.

To perform the analysis, some road driving sequences were selected from the KITTI-360 dataset [72]. The sequences present varying degrees of difficulty for the perception algorithms, as described in table 4.2.

Table 4.2: Description of the selected evaluation sequences from KITTI-360 dataset.

Name	Type	Description	Difficulty
A -	Suburbs	Normal suburbs traversal	+
A	Suburbs	Long sequence with several turnaround maneuvers	++
B	Country	Wide country road surrounded by vegetation	+
C	Country	Country road surrounded by dense vegetation	+
D -	Road	Normal city traversal	+
D	Road	Very busy road. Heavy trucks that cause sensor blockage	+++
E	Country	Fast moving road, dense vegetation and wide-open fields	++
F	Highway	Highway entrance scenario with many moving vehicles	+++
G	Urban	Urban roads with a tunnel passage	++

As described in the previous chapter, the proposed neural network was trained with the input of three odometry signals from different modalities, namely:

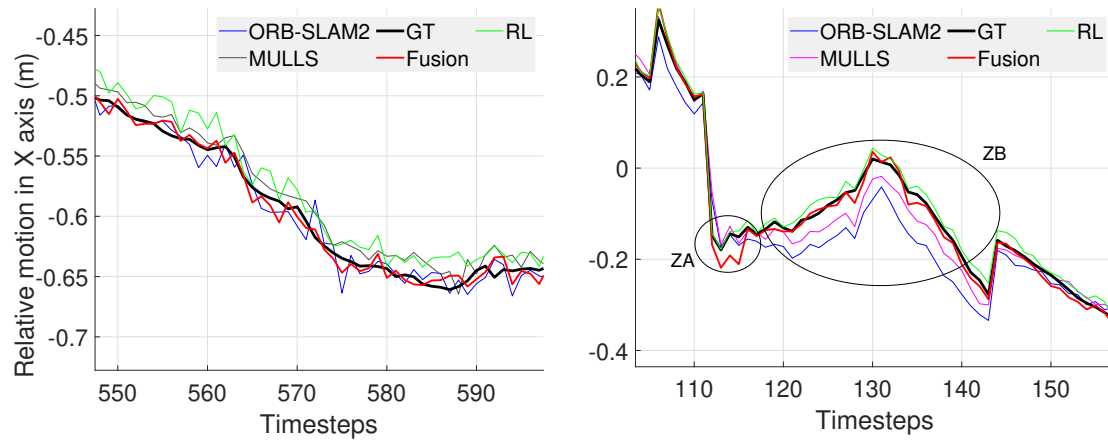
- ORB-SLAM2 [31]: Stereo camera rig (visual);
- MULS [53]: LiDAR (laser)
- ROS Robot Localization [71]: GPS+IMU (satellite + inertial)

This combination aims to create a robust system from the junction of different data modalities by minimizing the probability of situations affecting more than one or even all of the input signals.

4.2.2 Regular navigation conditions

The previous chapter described some of the intended goals of the proposed solution. One basic requirement included the fusion network to map the dispersion of each input and to produce a statistical triangulation of the signals to generate the most likely estimate possible. In this respect, figure 4.3a illustrates a small visual demonstration of the network's effect. In the figure, it can be seen that the network's output is consistently close to the ground truth at almost all times and still has a dispersion around the signal that is more contained than the other inputs or at least similar to the most accurate signal. In turn, figure 4.3b demonstrates the fulfillment of another of the requirements, namely to detect systematic deviations in one or more input signals and adjust the

estimate accordingly so as not to suffer the effect of these deviations. In the zone labeled with "ZB", it can be seen that both ORB-SLAM2 and MULLS show a constant downward deviation. During those instants, the fusion output tends to give RL more confidence. This phenomenon can also be seen by the similarity between the dynamics of the two signals (fusion and RL). In "ZA", although not desirable, the network has a slight delay in actuation, which comes from the sliding window statistical inputs. By comparing the two figures, it is also important to note that the network does not just learn to follow the signal that is generally more accurate across the dataset. In the first graph, it is possible to see, for example, that the RL has a small constant offset, which the net is able to ignore and not get affected by, while in the second graph, the RL is by far the most accurate signal and also the most similar to the NN's output.



(a) Illustration of the triangulation capacity of the neural network in producing an accurate prediction from the presented inputs. In this segment, ORB-SLAM2, which is the closest to the ground truth, presents an average deviation of 0.78 cm, whereas the value for the fusion layer is 0.60 cm.

(b) Illustration of the network's capacity in correcting systematic deviations in the input signals. The mean deviation for RL is 1.40 cm, whereas for the fusion layer is 1.27 cm, even with the off estimates in "ZA".

Figure 4.3: Samples of some prediction capabilities of the proposed fusion network, under normal navigation conditions.

Figure 4.4 also visually demonstrates the effectiveness of the fusion layer. The box plots represent the Euclidean error distribution of the estimates in sequences A- and B, respectively. In these sequences, the fusion estimates' error is consistently lower than the other network inputs. Ignoring some outliers, it is also noticeable that the error is always more contained, which consequently relates to a smaller trajectory error and increased robustness. In sequence B, more than 75% of the fusion layer estimates have an error below the average of the inputs, while in A- this is only not true for ORB-SLAM2. Even then, this percentage drops slightly to around 70%. This data suggests that for easy and intermediate difficulty sequences, the tendency is for the network to outperform its inputs, demonstrating the desired behavior.

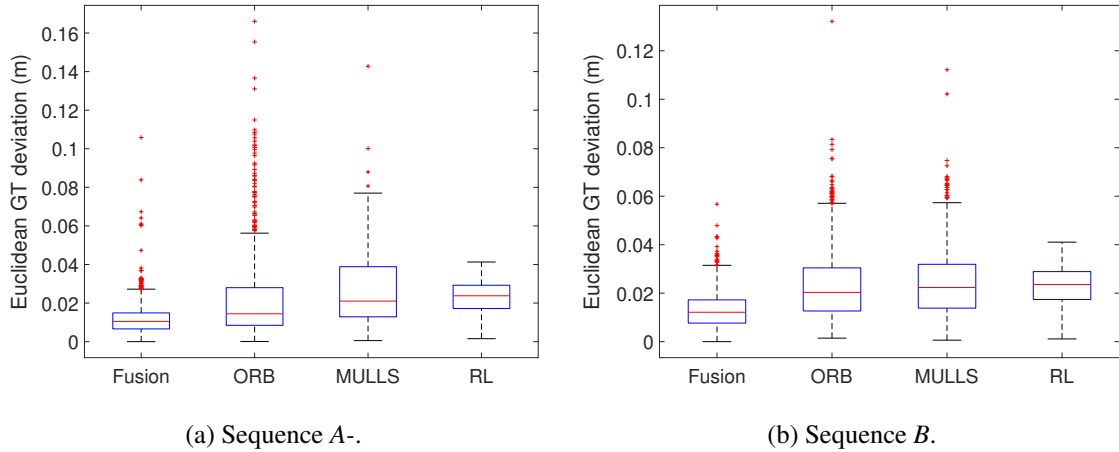


Figure 4.4: Box plots of the Euclidean error distribution for the individual odometry estimates in sequences A- and B. The graph whiskers represent the minimum and maximum registered values, the boxes' limits represent the 25th and 75th percentiles, and the red horizontal lines represent the distribution median. The red dots mark the data set outliers, which are errors whose values are greater than 1.5 times the inter-quantile distance.

Moving on to quantitative analysis, table 4.3 aggregates the MAE_{rel} results in the test sequences for the fusion result and the input signals, as well as the relative improvement for each signal in percent. On average, the fusion layer achieves a relative error (MAE_{rel}) 66.4% smaller than ORB's, 25.8% smaller than MULS's, and 52.7% smaller than RL's. In the sequences where the network is outperformed, except for sequence F, it is always outperformed by only one of the input signals. This phenomenon is again due to the fact that the network has a limited perception of the uncertainty of each input, and because of the reduced number of input signals.

4.2.3 Multi-modal odometry

Another major advantage of the network is that the number and nature of the inputs is not restricted to any type or combination. Therefore, by using signals from different modalities, comprising exteroceptive and proprioceptive sensors, the network can also benefit from the advantages of multi-modal systems. These include greater robustness to targeted failures (like tunnels for visual methods) and increased accuracy due to a more complete and redundant perception of the surrounding environment.

Sequence C, for example, contains roads entirely covered by dense vegetation (figure 4.5a). The vegetation significantly impacts the performance of LiDAR-based methods since the foliage promotes imperfect reflection of the laser beams, producing noisy point clouds which are difficult to align accurately. Recurring to table 4.3, it is possible to see in this sequence, the error of MULS, a method based on LiDAR, is more than double the error of ORB-SLAM2, in addition to the average deviation of the trajectory, which is more than four times higher (1.1302 m for the MULS and 0.2662 m for the ORB). Even so, the MAE_{rel} of the fusion network (0.0147 m) manages to get closer to ORB's (0.0154 m), getting to be even smaller, distancing itself from

Table 4.3: Results obtained by the proposed fusion network and the respective input signals in the selected sequences from KITTI-360. The results are evaluated following the MAE_{rel} metric (in meters). For each technique-sequence pair, it is also indicated the respective improvement on the error from the perspective of the fusion results. A negative value means the fusion operation decreased the result from the technique in question in x%. Best results are underlined for each sequence.

Technique	Sequence				
	<i>A</i> –	<i>D</i> –	<i>A</i>	<i>B</i>	<i>C</i>
ORB-SLAM2	0.0213 (-47.4%)	0.0188 (-41.5%)	0.0208 (-38.5%)	0.0237 (-44.1%)	0.0154 (-4.8%)
MULLS	0.0255 (-56.1%)	0.0103 (6.4%)	0.0514 (-75.1%)	0.0243 (-45.5%)	0.0316 (-53.5%)
RL	0.0230 (-51.3%)	0.0226 (-51.4%)	0.0895 (-85.7%)	0.0229 (42.3%)	0.0210 (-29.9%)
Fusion	0.0112	0.0110	0.0128	0.0132	0.0147
	<i>D</i>	<i>E</i>	<i>F</i>	<i>G</i>	
ORB-SLAM2	0.0398 (-20.3%)	0.0316 (40.6%)	0.5472 (-83.2%)	0.0262 (-27.8%)	
MULLS	0.0138 (129.3%)	0.1342 (-66.9%)	0.0230 (300.6%)	0.0228 (-17.3%)	
RL	0.0876 (-63.8%)	0.0872 (-49.1%)	0.0869 (6.0%)	0.0879 (-78.5%)	
Fusion	0.0317	0.0444	0.0921	0.0189	

MULLS (0.0316 m). Therefore, in this example, the fusion NN is not affected by the vegetation effects on MULLS. The same is also seen in the mean trajectory error. Moreover, figure 4.6 shows that fusion’s error dispersion is more contained than in any other input signal, which means it is more robust and consistent. This example is a practical proof of both the benefits of multi-modality for visual odometry and the added robustness that different sensors provide to the fusion network.



(a) Road surrounded by dense vegetation in sequence C. Foliage promotes faulty reflection of LiDAR laser beams.



(b) Tunnel passage in sequence G. The low visibility degrades the performance of visual odometry techniques.

Figure 4.5: Challenging scenarios (vegetation and tunnel) for visual odometry techniques in sequences C and G.

A similar example occurs in sequence E, which has some wide-open spaces that pose difficulties for point cloud-based methods. In figure 4.7a, MULLS clearly shows a discrepancy concerning the other signals for about 100 frames. However, by tracing the fusion signal it can be seen that

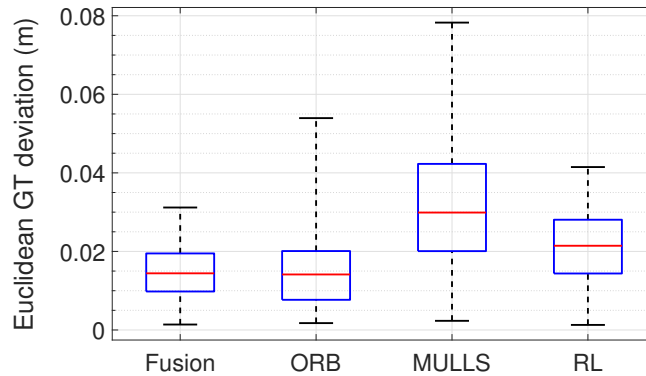


Figure 4.6: Euclidean error distribution in sequence C.

the network identifies the divergence and reduces the influence of MULLS. Although the relative mean error is smaller in ORB (0.0316 m), the fusion error (0.0444 m) is much closer to ORB's than to MULLS' (0.1342 m). This example constitutes a slight systematic deviation, which the network is able to overcome. In cases of larger error, however, it will no longer have the same effectiveness, as discussed later.

Another important example occurs in sequence G, where the vehicle traverses an unlighted tunnel for a few moments, as illustrated in figure 4.5b. The poor lighting of the tunnel and the scarcity of visual features cause the quality of visual techniques to degenerate during these intervals. This effect is not noticeable in the relative error since it lasts only a few instants but is visible in the trajectory error and trace. In figure 4.7b, which shows the trajectory plot of each technique, it can be seen the tunnel's location and that all visual methods present a drift afterward in the curve since they generated inaccurate estimates during the tunnel crossing, which caused the trajectory drift to propagate and accumulate. Regarding the fusion network, the relative error is lower than any of its inputs, and the trajectory error is very similar to MULLS's (LiDAR). It can also be seen from figure 4.7b that the trajectory does not suffer the same deviation as the visual algorithms and keeps track of the ground truth. This is further proof of the advantages of multi-modality in odometry and the correct functioning of the fusion layer in particular. Both of the previous cases demonstrate that the network does not limit itself to following the most accurate signal throughout the dataset. They also show the NN's capability of detecting some input flaws and adjusting its output in the best possible way.

Finally, the benefits of fusion can be seen in the case of sequence A. This sequence is longer than the others and presents a path with several successive curves. In this particular case, MULLS shows a significant degradation in turning maneuvers, whose effect accumulates over the entire trajectory, with an average trajectory deviation of 23.4 m, which is a considerably high value. Once again, the network is able to adaptively predict, giving added weights to the other inputs. This effect is visible in figure 4.8, where it is possible to see the trace of MULLS' trajectory compared to the fusion. Numerically, fusion's average trajectory deviation is only 1.13 m, which is 20.7 times smaller than MULLS'.

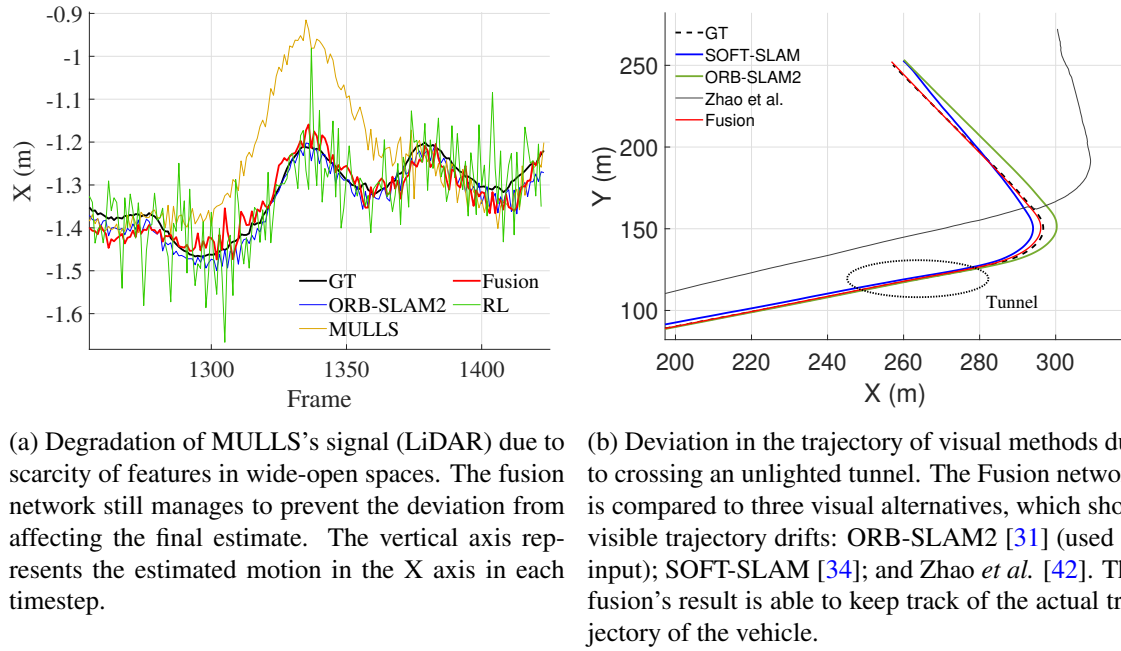


Figure 4.7: Impact of open spaces and lighting conditions on some of the fusion network inputs.

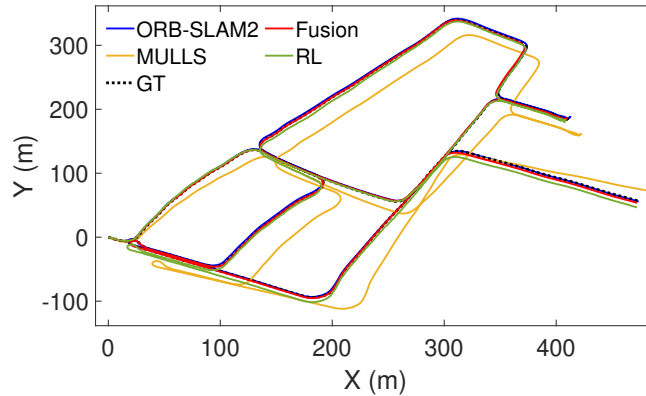


Figure 4.8: Reconstruction of the trajectory obtained by the fusion layer and the respective inputs in sequence A. This plot illustrates the drift in MULLS and the ability of the NN to make consistent predictions. Fusion obtains the best results regarding the mean trajectory drift with only 0.80 m, followed by ORB-SLAM2 with 1.54 m, RL with 4.28 m and MULLS with 19.3 m.

4.2.4 Challenging scenarios and conditions

Even though the network produces satisfactory results, there are still some cases in which its performance falls short, particularly in sequences D and F, which have more challenging characteristics. In both sequences, there are moments in which the field of view is dominated by multiple moving objects, which interferes with the quality of the ego-motion estimates, especially in the visual methods (figure 4.9). In sequence D, a large truck almost completely covers the camera's range for a few moments, leading to the error seen in figure 4.10a. This happens because the visual features captured in moving objects violate the static background assumption, and the remaining

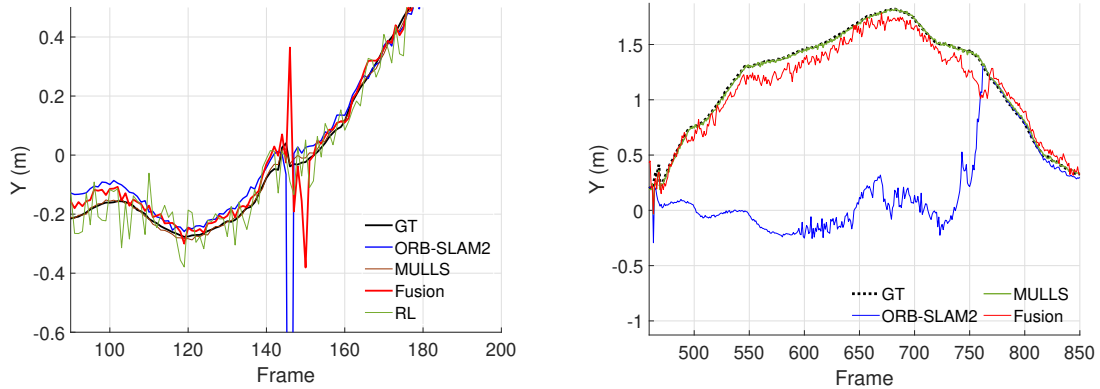
valid features are captured in a very compact area of the scene, due to the occlusions. Ideally, the network should detect that the signal in question (ORB-SLAM2) has a value far away from the remaining inputs, a high variance, and a low cross-covariance concerning the other inputs, discarding this input in the fusion operation. Although this is not entirely verified, it can be noted that the fusion error is much more contained than the error of the visual input, although it is enough to degenerate the trajectory along the rest of the path.



(a) Visual occlusions caused by two large trucks in sequence D. The trucks take up a significant part of the camera's field of view [72].

(b) Start and stop moment with multiple moving vehicles in the surrounding area. As the other vehicles move, while the sensing vehicle is stationary, they cause a false sensation of motion [72].

Figure 4.9: Road traffic scenarios from KITTI-360 containing multiple dynamic objects in the vehicle's surroundings.



(a) Deviation in ORB-SLAM estimates due to a large scale occlusion in the camera. The network is affected by this fault, although in a more contained way. The vertical axis represents the estimated motion in the Y axis in each timestep.

(b) Deviation in ORB-SLAM estimates due to the presence of multiple moving cars in the camera's field of view. Although the network is slightly affected by the failure, it is still noticeable the effort to follow the remaining inputs.

Figure 4.10: Impact of dynamic objects in the inputs and output of the deep fusion layer.

Also, in sequence F the problem is similar, but the failure occurs over a wider time interval. Similar to the previous case, the network can detect that one of the inputs is faulty and place more emphasis on the others, as is noticeable from figure 4.10b. Still, the result is overly affected by the fault. Figure 4.11b shows some areas where ORB-SLAM2, for external reasons in the scene, suffers some sudden momentary deviations, which end up distorting the entire trajectory. The brief deviations of the network, namely in figure 4.10a and the beginning of figure 4.10b are also related to the sliding window, i.e., although the ORB estimates are a bit distinct from the other

inputs, the sliding window metrics, namely, variance, covariance and mean of the derivatives, take a few instants to diverge, since they take into account the last 10 estimates. However, figure 4.11a shows the dispersion of the estimation error for the fusion and respective inputs. The network error still has a margin of progression towards MULLS, even so, it manages to have a distribution that is much closer to MULLS, compared to ORB-SLAM2. The error's 75th percentile is located at 0.1289 m and 0.0314 m for the fusion layer and MULLS respectively, whereas for ORB, the value is much superior, reaching 1.1599 m.

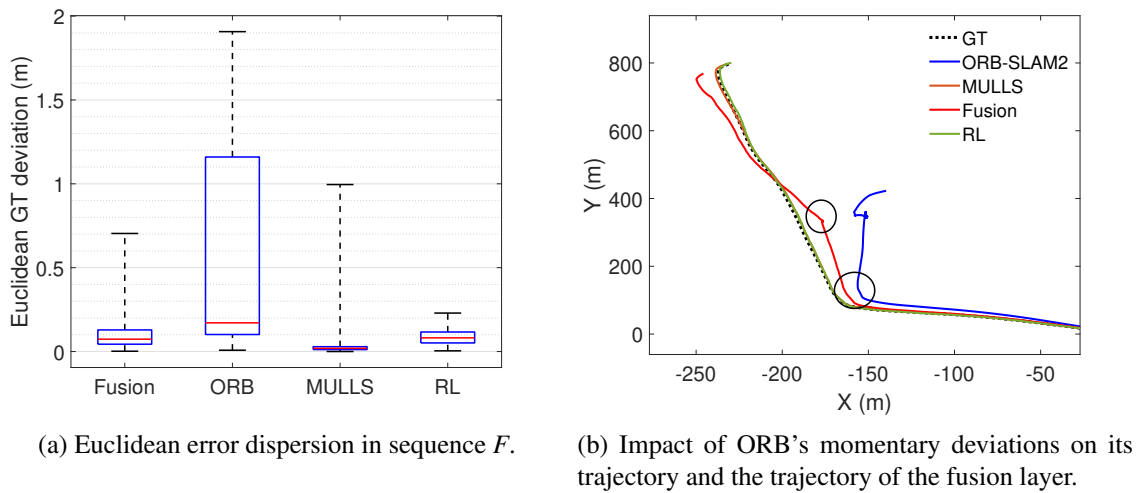


Figure 4.11: Results obtained in sequence F .

This problem is not specific to dynamic objects but is common to any reason that imposes unusual behavior on the input signals. In fact, in sequences D and F, the discrepancy between the median and the mean relative error is more noticeable in the fusion output, suggesting that the network is affected by these momentaneous input failures. The poor results of the network, in sequences D and F, are explained by the low representativeness that this type of error has in the dataset, i.e., these cases occur very sporadically, besides depending on the method used to generate the ego-motion estimate. The considerable randomness inherent to these errors and the lack of representativeness in the dataset hinder the network's ability to detect patterns in the data and thus abstract from noisy signals. Some solutions may involve training the network with more inputs, preferably with good accuracy and reliability. In this case, only three inputs are used, so when one of the signals presents significant failures, the network can only resort to the remaining two, which is a non-optimal quantity for the correct triangulation of the estimates. By taking a higher number of input signals, not only do individual faults become more diluted, but they are also more easily identified by cross-comparison. Moreover, the additional information creates an expanded data representation space that can help decode subtle data patterns, which may allow for better results. Another solution could be to train a parallel network capable of individually detecting faults in signals. This detection network could provide a binary fault indication to the fusion network for each input, alleviating the complexity of the latter.

Moreover, as already mentioned, sequence D has some moments in which the ORB estimates are largely wrong. However, the biggest difficulty of this sequence is the fact that the ORB-SLAM2 signal has a stable behavior but not very accurate, or at least not as accurate as the other inputs, throughout the entire sequence. In fact, from the box chart of figure 4.11a, it can be seen that more than 50 % of the ORB estimates are in a range of errors close to the remaining inputs. This is perhaps the most difficult problem in late fusion, i.e., when the input signals are relatively consistent but point in slightly different directions, and there is no other type of information that can indicate which of the estimates is actually the most likely. In this sense, the objective of the network is to make this discernment possible through the training data, i.e., to know which signals are usually more exact and precise and under what conditions and, therefore, to be able to predict a reasonable result. What happens in this case is that, throughout the training dataset, ORB-SLAM2 tends to produce estimates closer to the ground truth, while MULS and RL are not so consistent. Thus, in the occurrence of these cases, the network tends to produce predictions that are closer to ORB-SLAM2 than to the other inputs, when it should not happen. This phenomenon is easily verified by the fusion's MAE_{rel} (0.0317 m), which is very close to the ORB's (0.0398 m), while MULS' is considerably lower (0.0138 m).

In addition to the solutions already mentioned, namely more inputs and a wider training dataset, which would also apply to this case, it would be interesting to change the network to produce a prediction of the uncertainty associated with its own odometry estimate. In situations similar to this, the estimates would be accompanied by a relatively high level of uncertainty, which would be useful if the network signal, together with additional data, was to be used in a second fusion stage, for example.

4.2.5 Data augmentation impact

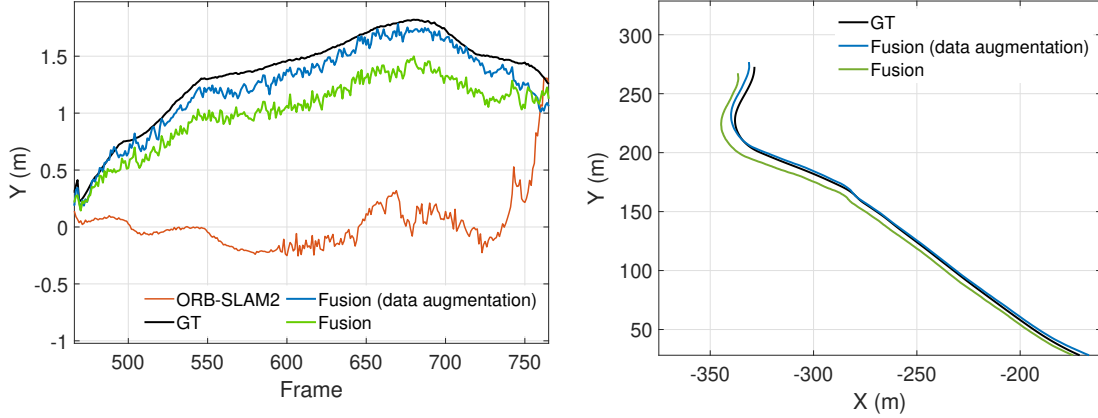
The previous paragraphs have described some of the limitations of the network, notably in the most challenging sequences (D and F). The lack of representativeness of signal failures in the dataset was pointed out as the main reason for the network to present some difficulties in dealing with punctual faults. In this sense, the network was trained not only with the original data, but with sequences in which errors were induced in the network's information sources, in order to simulate failures that may eventually occur. In addition to increasing the number of failure situations in the network's test scenario, this data augmentation process is also important to avoid the phenomenon of overfitting. For this particular problem, given its characteristics, it is important to ensure that the network does not simply mirror one of the input signals without a second criterion. Three different types of errors were then applied to the input signals (only one at a time), and throughout all the training sequences: white noise with zero mean, to simulate any behavior that degrades the precision of the signal; signal offset, to simulate a degradation of accuracy, due to a bad calibration, for example; constant signal error, to simulate that the signal/sensor stopped working. Some graphs illustrating these errors can be found in the previous chapter (figure 3.8). In this way it was possible to observe significant improvements between the models trained with and without

data augmentation. Table 4.4 presents the MAE_{rel} and MAE_{traj} results of the two versions, on the most difficult sequences of the dataset (D,E,F and G).

Table 4.4: Comparison of the results from the fusion network trained with the original dataset ("Fusion") and trained with the data augmented dataset ("Fusion +").

		Sequence			
		D	E	F	G
Fusion	$MAE_{rel}(m)$	0.0533	0.0506	0.1512	0.0508
	$MAE_{traj}(m)$	4.7544	14.975	30.483	3.7254
Fusion +	$MAE_{rel}(m)$	0.0317	0.0444	0.0921	0.0189
	$MAE_{traj}(m)$	3.9483	14.124	14.244	0.7994

Looking at the table, the impact of data augmentation is quite noticeable, as the results are better in all sequences without exception. In terms of relative error (MAE_{rel}), the improvements are on the order of 40.53%, 12.25%, 39.09% and 62.80%, for sequences D, E, F and G respectively. Furthermore, the results can be visualized in the graphs in figure 4.12, where it is clear that with data augmentation, the network is able to get much closer to the reference signal under failure conditions, both in terms of individual estimates and trajectory. On average, the improvement is in the range of 39%, both for the relative and trajectory errors.



(a) Differences in the relative motion estimates between the models trained with and without data augmentation. The plot shows the data-augmented model predictions are much closer to the ground truth, when ORB-SLAM2 presents a divergent behavior.

(b) Comparison between the trajectory reconstructions from the models trained with and without data augmentation. The data-augmented model clearly approximates the real trajectory more efficiently.

Figure 4.12: Impacts of data augmentation in the network results from challenging scenarios.

Although the data augmentation process has a large impact on the network results, a slight overdependence on the ORB-SLAM2 signal is still noticeable. This phenomenon occurs, as mentioned, when the consensus between the input signals is not very pronounced, and there is a slight tendency to give extra weight to ORB. This is also a direct consequence of the small number of

inputs, hence training the network with more inputs and with more data where the ORB signal is affected with noise can have a significant impact on the results.

4.2.6 Comparison with state-of-the-art techniques for odometry

So far, the fusion network has been evaluated from the comparative point of view of its inputs. In this subsection, the network results are exposed along with other state-of-the-art odometry techniques. In this sense, the results were collected for several algorithms, including visual techniques, point cloud-based, multi-modal and learning-based architectures. A summary description of each of these techniques is presented in table 4.5. Table 4.6, in turn, aggregates the obtained results.

Table 4.5: Description of state-of-the-art odometry techniques utilized for results comparison.

Technique	Type	Keypoints
SOFT-SLAM [34]	Visual	- Careful feature selection and tracking (SOFT) - Efficient mitigation of outliers' effect
Zhao <i>et al.</i> [42]	Visual	- Predicts optical flow and scene depth with parallel deep neural networks
CT-ICP [57]	LiDAR	- Adapted real-time ICP - Robust to high-frequency movements (innovative elastic motion formulation)
F-LOAM [51]	LiDAR	- Two-stage point cloud matching instead of traditional iterative method - Low computational cost
Ramezani <i>et al.</i> [67]	Multi-modal (Stereo + IMU)	- The agent's motion is propagated through a Multi-State Constraint Kalman filter (MSCKF) - The stereo rig provides additional pose constraints

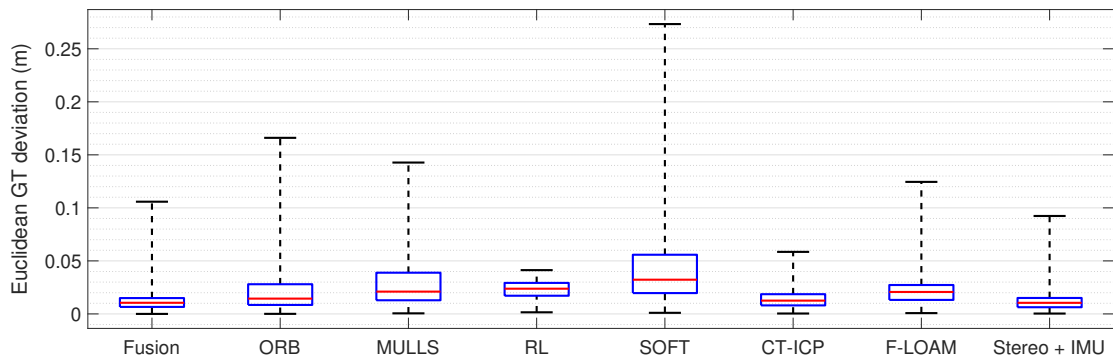


Figure 4.13: Euclidean error distribution for all the evaluated odometry techniques across all selected sequences of KITTI-360.

In terms of relative error (MAE_{rel}), the fusion network obtains, in all sequences, better-than-average results, which is already a demonstration of its efficiency, since this is not the case for any of its inputs. Furthermore, the graph in figure 4.13 helps position the fusion results in the

Table 4.6: Results obtained by several odometry techniques in nine road-driving sequences of KITTI-360, including the proposed fusion network. The results are evaluated under the MAE_{rel} , Med , and MAE_{traj} metrics, expressed in different colors in the table. The techniques marked with an asterisk were used as inputs to the fusion layer, which is labeled as "Fusion". Best results are underlined for each sequence.

Technique	Sequence								
	$A -$	$D -$	A	B	C	D	E	F	G
ORB-SLAM2* [31]	0.0213	0.0188	0.0208	0.0237	0.0154	0.0398	0.0316	0.5472	0.0262
	0.0144	0.0175	0.0145	0.0203	0.0141	0.0287	0.0311	0.1712	0.0249
	1.9218	1.5101	1.9028	2.6691	0.2662	5.5896	7.5060	115.03	2.2322
SOFT-SLAM [34]	0.0547	0.0525	0.0501	0.0383	0.0207	0.0539	0.1124	0.0697	0.0497
	0.0560	0.0191	0.0535	0.0296	0.0184	0.0167	0.1122	0.0451	0.0414
	13.293	7.2436	12.487	2.5541	0.1398	12.116	26.510	6.8950	2.4767
MULLS* [53]	0.0255	0.0103	0.0514	0.0243	0.0316	0.0138	0.1342	0.0230	0.0228
	0.0210	0.0097	0.0456	0.0223	0.0299	0.0111	0.1013	0.0188	0.0174
	3.1749	0.3235	23.458	1.5680	1.1302	1.1905	11.414	1.2512	0.7214
CT-ICP [57]	0.0140	0.0144	0.0142	0.0177	0.0276	0.0161	0.0391	0.0200	0.0192
	0.0125	0.0128	0.0130	0.0159	0.0169	0.0133	0.0383	0.0164	0.0147
	0.9277	0.2598	2.9409	0.3429	0.7102	0.2563	9.4263	1.1266	0.8856
F-LOAM [51]	0.0212	0.0155	0.0297	0.0179	0.0239	0.0162	0.0596	0.2064	0.0158
	0.0207	0.0138	0.0298	0.0157	0.0151	0.0135	0.0454	0.1382	0.0119
	3.9362	0.4932	10.456	1.0559	0.4677	0.6644	16.749	40.121	0.4820
Zhao et al. [42]	-	0.1426	-	0.4015	0.2379	0.1467	0.5724	0.3179	0.4419
	-	0.1220	-	0.3395	0.1855	0.1275	0.4660	0.2260	0.4198
	-	7.5653	-	50.174	4.3649	9.9282	49.452	29.869	34.367
Stereo + IMU [67]	0.0115	0.0139	0.0119	0.0646	0.0072	0.0195	0.0128	0.0235	0.0244
	0.0105	0.0120	0.0111	0.0710	0.0065	0.0132	0.0118	0.0193	0.0219
	1.8414	0.7434	3.6316	1.8653	0.1986	1.0926	2.3420	1.6052	1.0825
RL* [71]	0.0230	0.0226	0.0895	0.0229	0.0210	0.0876	0.0872	0.0869	0.0879
	0.0238	0.0232	0.0836	0.0236	0.0214	0.0816	0.0810	0.0816	0.0800
	12.631	2.5716	4.5716	8.2314	1.2939	1.8726	3.7788	1.3685	1.3883
Fusion	0.0112	0.0110	0.0128	0.0132	0.0147	0.0317	0.0444	0.0921	0.0189
	0.0105	0.0104	0.0105	0.0121	0.0144	0.0278	0.0405	0.0737	0.0173
	1.0132	0.5055	1.1324	1.8469	0.5261	3.9483	14.124	14.244	0.7994
Mean values	0.024	0.021	0.038	0.030	0.021	0.035	0.068	0.140	0.035
	5.390	1.878	8.493	2.612	0.601	3.255	11.104	23.915	1.324
Metrics:	$MAE_{rel}(m)$			$Med(m)$			$MAE_{traj}(m)$		

panorama of all evaluated methods. It can be observed, visually, that the dispersion of the fusion error is clearly concentrated at a lower plateau than most. In terms of average error (MAE_{rel}), for all sequences, the fusion network obtains an error of 1.17 cm, leaving itself surpassed only by the other fusion method of Ramezani *et al.* [67], with 1.15 cm. Still, both considerably below the overall average error of 2.14 cm. Furthermore, the 75th percentile of the fusion error is the lowest of all, meaning that 75% of the estimates generated by the network are more accurate than 75% of the estimates generated by any other technique. It can also be seen that the dispersion of the

fusion error is among the most concentrated, namely the 75% most accurate estimates, despite having a relatively high top-whisker. This means that the network is very accurate, although it produces some outliers as a result of the limitations already discussed above, such as the lack of representative data in the dataset and a limited number of inputs.

4.3 LSTM and multi-kernel convolution architectures

A quick look at the graphs in figures 4.7a and 4.10, for example, shows that the ground truth signal is always smoother than the signals from the odometry techniques analyzed, including the fusion network. Since the graphs refer to the motion relative to the previous position at each instant, this means the actual motion of the vehicle is more restrained. This is because sensors collect the data on board a vehicle that attends to a dynamics model, which restricts its range of motion and ensures consistency between instants regulated by the laws of physics. In light of this, the previous chapter introduced two extended architectures (*FC-LSTM* in figure 3.11 and *CNN-FC* in figure 3.12) whose goal is to capture the model of the car's dynamics by providing the network with more information on the temporal dimension and the means to work with it. Table 4.7 summarizes both networks' parameters.

Table 4.7: Model parameters of FC-LSTM and CNN-FC.

Network Parameter	FC-LSTM	CNN-FC
Number of signals merged	3	3
Number of inputs	45	45
Number of outputs	2	2
Hidden layers	4	6
Sequence length (actual and previous estimates)	5	5
Hidden units (LSTM)	32	-
LSTM cells	2	-
Number of kernels (CNN)	-	1000
No. of inputs to FC (after maxpooling)	-	50
Optimizer	SGD	SGD
Activation Function	ReLU	ReLU
Loss function	MSE	MSE
Learning rate (with scheduler)	1e-3	1e-3
Batch size	4	4
Sequence length (for statistical metrics)	10	10
Training epochs	15	45

In figure 4.14, it is possible to see the results obtained by the two mentioned architectures in the same sequence. Comparing figures (a) and (b) through visual analysis, it can be seen that

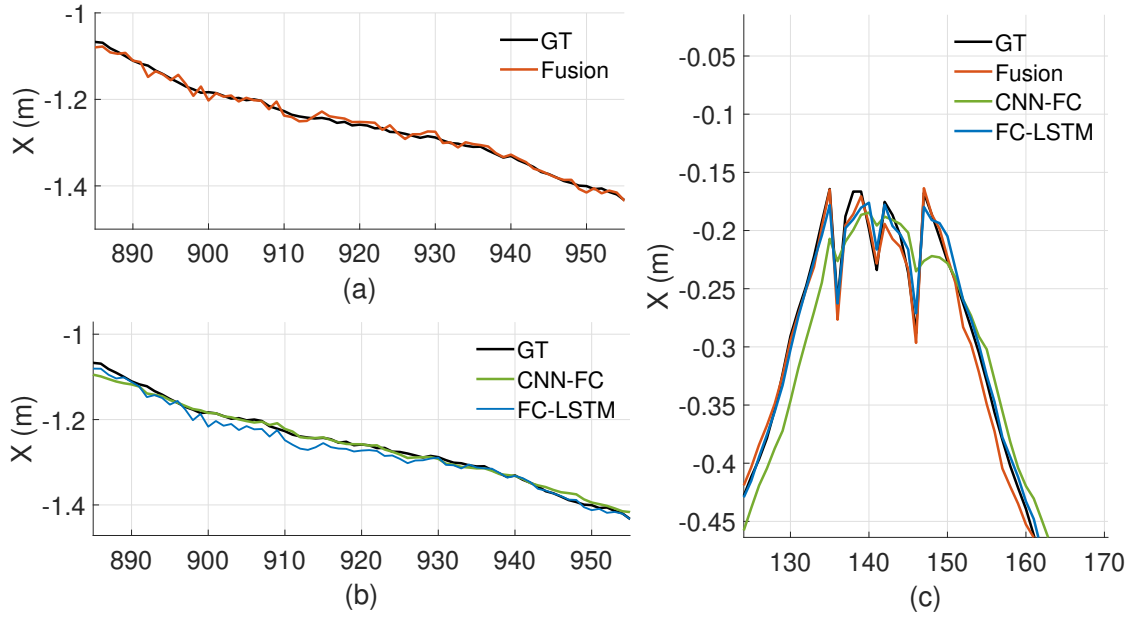


Figure 4.14: Comparison between the originally proposed fusion network and the two extended architectures. All plots refer to relative motion estimates in the same sequence.

both solutions manage to take a small step towards signal smoothing relative to the original fully-connected network. With CNN-FC in particular, the effect is quite noticeable, whereas, with the FC-LSTM, the result is identical to the simpler fully-connected network. Figure 4.14c demonstrates, at the same time, two of the main limitations of the CNN-FC network. One of them is the slight delay in the response, which is related to the time that the sliding window metrics take to become meaningful. The second limitation concerns the over-restriction of movements, in that the network does not seem to react with the necessary amplitude to more abrupt variations in the input signals (timestep 140), behaving similarly to a low-pass filter. In this sequence, the original architecture is still superior, with a MAE_{rel} error of 0.0138 m, compared to the values of 0.0218 m and 0.0178 m of the CNN and LSTM architectures, respectively. Although the results are worse, the graph in figure 4.15 shows that these two solutions, especially the CNN-FC, make the output signal's dynamics closer to the ground truth dynamics. In the graph, the differences in the estimates between successive instants (motion derivatives) are calculated and compared to the ground truth derivatives. The box plot shows the derivatives' dispersion for every architecture. It is noticeable that CNN-FC presents the output in which the dynamics resemble the ground truth dynamics the most. The 75th percentile locates at 0.68 cm, followed by FC-LSTM at 0.81 cm, MULLS at 0.89 cm, and F-LOAM at 0.95 cm. Moreover, regarding the average difference, CNN-FC shows the lowest value: 0.55 cm, compared to 0.63 cm for FC-LSTM and 0.91 cm for the fully-connected. This proves that the CNN results are the closest to the ground truth, making this architecture the most effective in creating a coherent output according to the dynamics of the movement and which better approximates the variations of the vehicle's motion, as was intended with the study of these alternatives. As the complexity of networks increases, it becomes more and more difficult to find

the optimal point for a given task, due to the exponential growth of combinations between the various parameters and variations of architectures. On the other hand, the risk of overfitting is also greater. Still, the results of these two extended frameworks are promising, leading one to believe that with experimental and careful parameter tuning, even better results could emerge, specially concerning the estimates' accuracy.

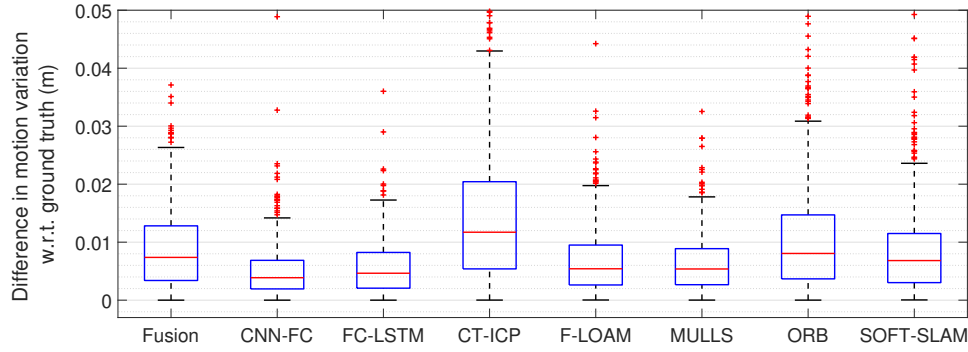


Figure 4.15: Distribution of the differences between the motion derivatives and the true motion variations of the vehicle (GT). The lower the values in the Y-axis, the closer the dynamics of each signal are to the dynamics of the ground truth.

4.4 Kalman Filter comparison

One of the most prevalent techniques in late fusion is the Kalman filter. Although the prediction-update dynamic does not apply directly in this case, it is possible to average the inputs weighted by the inverse of their variance (Equation 3.1), simulating the KF functioning, as elaborated in the previous chapter. To compare the performance of the fusion network and the KF, two different ways of weighting the inputs were considered, disposing of the same data as the neural network. In the original Kalman filter, each signal was to have a variance value associated with its uncertainty. For this purpose, the deviation to the ground truth of each input was calculated for the entire training dataset to obtain the error variance of each signal. In this way, the values of the variances that constitute the weighted average weights are fixed and calculated in advance. The other alternative operates similarly to the fusion layer, calculating the variance in a moving window. Since this is an online operation, the variance must be calculated relative to the signal itself since the error is not accessible. This operation assumes that a more accurate signal also has a more stable behavior, i.e., with lower variance. While discussing the results, the first data fusion technique will be denoted as KF_f (fixed variance) and the second KF_v (variable variance). Results for the two adapted versions of KF were collected using the same data as the fusion network, on the same sequences used to previously assess the NN.

A quick analysis over table 4.8 shows that the fusion layer takes an advantage in both easy and challenging sequences. In sequences where the NN outperforms the best of the KFs, the relative error is on average 42.1% lower, while when KFs outperform, the error is only 24.26%

Table 4.8: Comparison between the fusion network and the two weighted average approaches in the selected KITTI-360 sequences under the MAE_{rel} and MAE_{traj} metrics (in meters).

		Sequence								
		A-	D-	A	B	C	D	E	F	G
Fusion	MAE_{rel}	0.012	0.011	0.013	0.013	0.015	0.032	0.044	0.092	0.019
	MAE_{traj}	1.013	0.505	1.132	1.847	0.526	3.948	14.12	14.24	0.799
KF_v	MAE_{rel}	0.016	0.008	0.031	0.015	0.013	0.020	0.046	0.185	0.021
	MAE_{traj}	4.030	0.377	11.36	2.666	0.351	3.995	5.436	44.35	1.230
KF_f	MAE_{rel}	0.019	0.015	0.020	0.020	0.014	0.034	0.032	0.433	0.024
	MAE_{traj}	1.121	1.076	2.531	1.511	0.376	4.564	6.031	91.44	1.970

lower. Since the results are not significantly far apart, it is safe to conclude that the fusion network can at least match the performance of the Kalman filter. However, more challenging scenarios should be used to test each approach's limitations, advantages, and robustness to compare the results properly. Recurring to some error induced sequences utilized in the validation set of the dataset augmented dataset (as described in section 4.2.5), the performance of the fusion layer was tested against the two KF versions. The results are expressed in table 4.9. From the general point of view, it can already be concluded that the network outperforms both filters by practically surpassing all the results in all the sequences. In addition to the good numerical results, it can also be concluded that in terms of robustness the fusion layer is superior in that it has a consistently good performance across all types of error evaluated. Furthermore, the histogram of figure 4.16 depicts the deviation distribution over the three sequences for the fusion network and the two KFs. The graph shows that the fusion layer and the KF_v have a comparable performance, with a slight tendency for the fusion error to be concentrated on lower values. Moreover, KF_f 's right tail end is more spread across the higher values of the spectrum. The next paragraphs discuss each sequence in more detail.

Table 4.9: Comparison between the fusion network and the two weighted average approaches in the sequences induced with input signal failures. Each column refers to a KITTI-360 sequence, in which one of the inputs (ORB-SLAM2 in the first and MULS in the last two) is affected with the indicated type of error. In the first, the signal is affected with white noise of null mean and uniform distribution. In the second, the signal starts producing a static value over time, and in the third, the signal is affected by an offset. All sequences present the respective failure regime during half the sequence length. Results are expressed in terms of MAE_{rel} and MAE_{traj} metrics (in meters).

		White noise	Constant signal	Offset
Fusion	MAE_{rel}	0.0409	0.0493	0.0151
	MAE_{traj}	6.6616	37.583	1.9381
KF_v	MAE_{rel}	0.0252	0.7610	0.0282
	MAE_{traj}	12.621	511.53	4.3703
KF_f	MAE_{rel}	0.0813	0.0923	0.0181
	MAE_{traj}	23.560	70.424	1.9551

In the first situation, the ORB-SLAM2 estimates are affected by random white noise with null

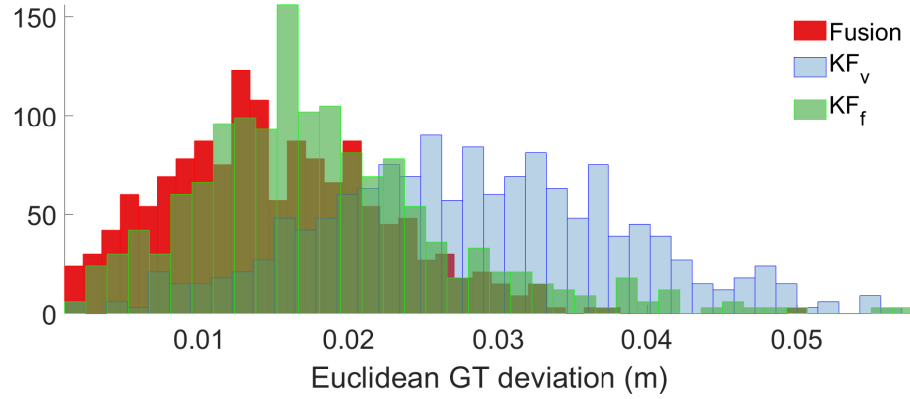


Figure 4.16: Histogram of the Euclidean error distribution for the fusion layer, KF_v , and KF_f on the same three sequences of table 4.9.

mean, arbitrary variance, and uniform distribution. The intention is to evaluate situations where one of the network inputs presents a noisy and unusual behavior for some reason (passing through a tunnel, too much vegetation, etc.), and should be considered carefully. The graphs in figures 4.17 and 4.18 illustrate the described situation and compare the response of the fusion layer with the two adapted versions of the KF. KF_f does not discriminate the noisy signal and performs worse since it weights the inputs according to fixed statistical values. KF_v , on the other hand, by using a sliding window, is able to determine that the variance of the affected signal is much higher, thus giving it a much smaller significance in the weighted average, causing it to be robust to this type of failure. The NN results (4.1 cm), despite not being as satisfactory as KF_v 's (2.5 cm), still manage to reduce the impact of the faulty signal, positioning at an intermediate level between the KFs, which is, nonetheless, much closer to KF_v (KF_f 's is 8.1 cm).

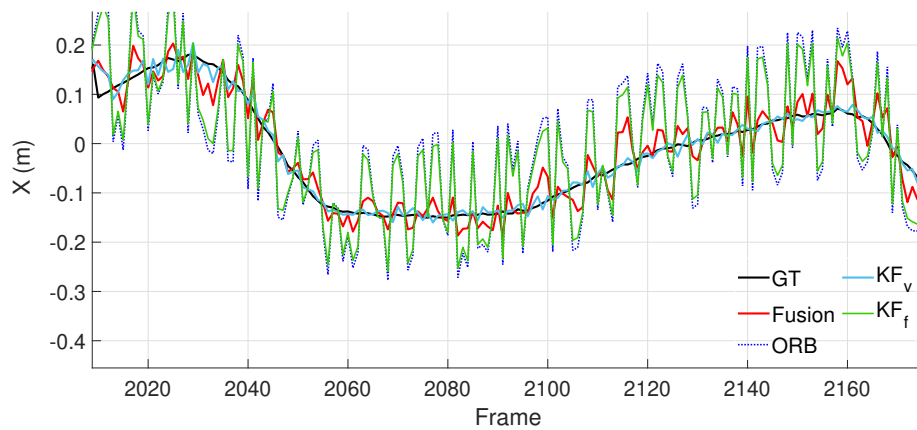


Figure 4.17: Fusion network and KFs outputs when applying white uniform noise to one of their inputs (ORB).

The error dispersion (figure 4.18) also shows more similarities with the error dispersion of KF_v . Since the signal is not noisy throughout the entire sequence, and the fusion layer is superior

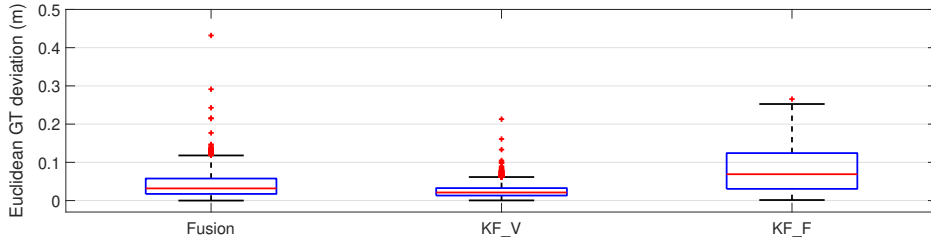


Figure 4.18: Error distribution of the fusion network and KFs when applying white Gaussian noise to one of their inputs.

under normal conditions, it still achieves a better result regarding trajectory error (53% smaller than KF_v 's). This example is particularly challenging for the network since the signal disturbed by the noise is also the signal with the highest precision and accuracy throughout the dataset under normal conditions, so the network tends to give it an added weight, even in conditions that it should not. As discussed in the last section, this was a previously identified limitation, which would not be as expressive if the affected signal was one of the remaining inputs. Again, a richer dataset and more input signals could improve the network's performance under these circumstances, and almost certainly surpass the KF.

In the second case, MULLS starts producing an arbitrary constant value, as if it stopped working (figure 4.19 - timestep 600). As expected, KF_v immediately becomes obsolete since a constant signal has a null variance. The KF_f is also affected by the error, producing in the output a systematic deviation. Under these conditions, the fusion network is superior as it detects the incongruence between the affected signal and the other inputs (table 4.9). This is also noticeable in figure 4.20, in which the network's error dispersion is condensed on a lower plateau than the KFs. In cases similar to this, covariance measures become important to inform the network that one of the input signals behaved distinctively in the last moments. Again, the network could benefit from more signals as input. Looking at figure 4.19 it is possible to see that the network output, ignoring the faulty signal, is only dependent on the remaining two (ORB-SLAM2 and RL). This causes the network to produce outputs with lower confidence while also being more sensitive to possible failures of the two remaining inputs. At the end of the graph (from timestep 750), it can be seen the network producing some undesirable deviations precisely because ORB-SLAM2 also presents some significant divergences from the actual motion of the vehicle. If more inputs were to be used, the network could perhaps resist the occurrence of simultaneous failures, providing the remaining inputs were accurate enough.

In the latter situation, an additional offset (not necessarily constant) is applied to one of the inputs (MULLS), simulating, for example, a poor calibration of the vehicle's sensors. In both versions of the KF, there is no discrimination mechanism other than the variances of the signals. For this reason, both KF_v and KF_f achieve worse results than the fusion layer because the applied offset does not change MULLS's variance (table 4.9). Thus, the signal is still interpreted in the usual way, unlike in the NN. This effect on KF_f is not very noticeable in figure 4.21, because the uncertainty associated with MULLS is higher than ORB's for example, so KF_f already attributes

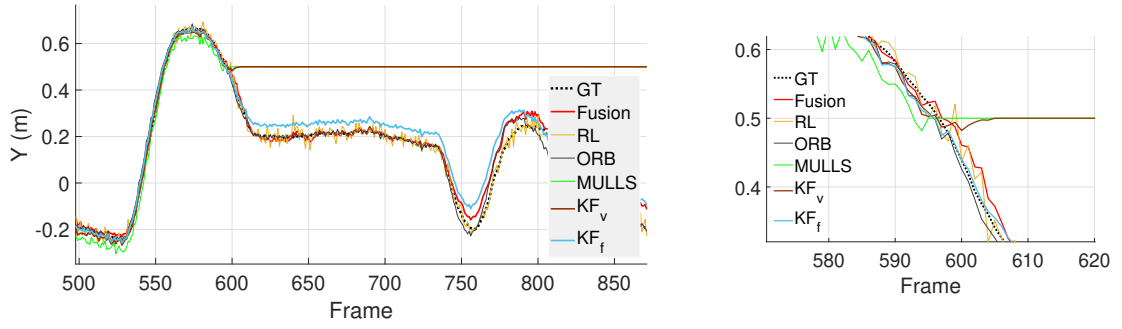


Figure 4.19: Fusion network and KFs outputs when applying a non-changing failure to one of their inputs (MULLS).

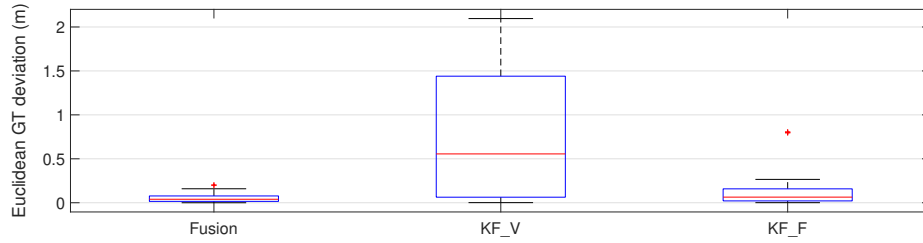


Figure 4.20: Error distribution of the fusion network and KFs when simulating a non-changing failure in one of their inputs.

MULLS a lower weight in the fusion. Nonetheless, the superiority of the neural network can be seen in figure 4.22, which shows the error dispersion of all architectures in this sequence. The error's 75th percentile is located at 0.0200 m for the fusion network, and at 0.0282 m and 0.0225 m for KF_v and KF_f respectively. The discrepancy between MULLS and the remaining inputs allows the network to detect that MULLS's signal is probably undergoing a failure regime, adapting its output accordingly.

These tests show that KF, while achieving very reasonable results, is sometimes limited by the quality of its inputs. On the other hand, the proposed solution has a better ability to abstract from signals that have no value to the final system estimate.

This chapter has shown, through direct comparison and with real data, that the proposed solution can achieve performance on par with state-of-the-art odometry techniques under regular conditions while surpassing the performance of its input signals individually. Furthermore, it also achieves comparable results to Kalman filter fusion. Some limitations of the fusion network and possible solutions to robustify the system were also detected and discussed. The results also revealed the ability of the CNN and LSTM networks to capture the additional inter-pose constraints relative to the car kinematics models by yielding the closest results to the actual vehicle's motion dynamics.

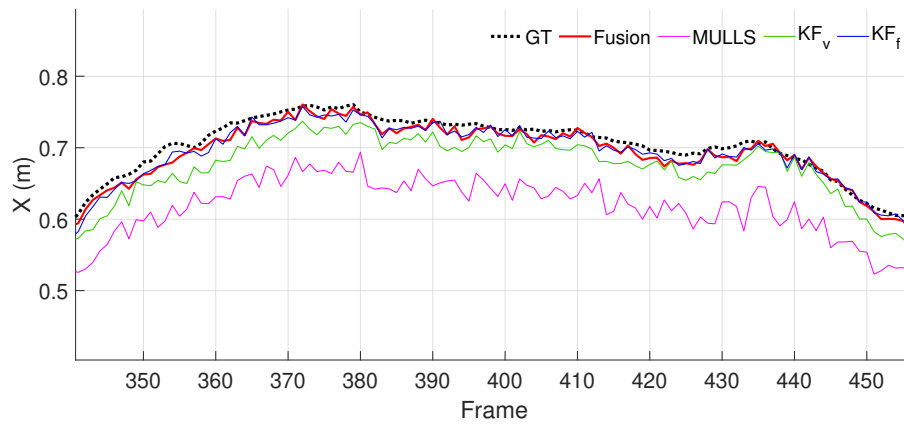


Figure 4.21: Fusion network and KFs outputs when applying a systematic deviation to one of their inputs (MULLS).

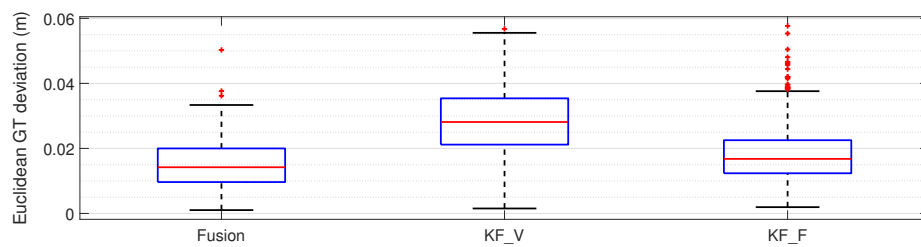


Figure 4.22: Error distribution of the fusion network and KFs when applying a systematic deviation to one of their inputs.

Chapter 5

Conclusions and future work

This dissertation explores the premise that using data from distinct sensor modalities in visual odometry systems increases the capabilities and robustness of the process. In this sense, a solution was proposed to fuse odometry estimates from different algorithms and sensors through a late fusion approach, using a deep neural network. The fusion layer inputs are the ego-motion estimates at each instant and a set of statistical metrics that quantify the quality and uncertainty associated with each input signal. The odometry estimates are generated by state-of-the-art techniques, namely ORB-SLAM2 (visual) [31], MULLS (LiDAR) [53], and ROS Robot Localization (GPS/IMU) [71]. The data used for this effect was retrieved from the KITTI-360 autonomous driving dataset [72] and includes stereo RGB images, point clouds, and IMU/GPS measurements. The purpose of the network is to properly weigh the various estimates taking into account their statistical dispersions, as well as to detect any individual failures and mitigate their effect on the fusion. The main contribution of this work is the creation of a modular odometry late fusion system that can be trained for any combination of inputs, regardless of their origin. Furthermore, there are very few late fusion architectures in the visual odometry state-of-the-art, and none employ deep learning techniques in this extent.

The network's performance was evaluated on nine city/suburb traversal sequences of varying difficulty. In these, the fusion layer is able to improve the respective inputs by 66.4 % (ORB-SLAM2), 25.8 % (MULLS), and 52.7 % (RL), on average, concerning the mean relative motion error. Moreover, the NN's robustness and the benefits of multi-modality were evaluated on specific runs where the scene conditions mainly affect one of the network inputs, such as dim tunnels, open spaces, dense vegetation in the surroundings, and the presence of multiple dynamic objects. In the referred conditions, the network improved the accuracy of the faulty signal by roughly 47% in relative motion estimates and 29% in trajectory estimation. These results demonstrated the network's ability to meet its objectives in real road traffic situations. The evaluation also covered some limitations of the network, especially in quite complex environments where one of the signals is fairly degraded, and the others show inconsistent behavior. Even so, the network could detect the degraded signal and remove some influence from it, although not enough to overcome the results of the other inputs. This problem was due to the low representativeness of failure scenarios

in the training data and the limited number of network inputs. The issue was addressed with a data augmentation step in which the network information sources were affected by various types of noise. Through this operation and network retraining, the relative motion and trajectory estimation errors decreased by about 39%, proving that richer training data could eventually robustify the fusion layer even further.

In addition, the network was also compared with two adapted versions of the Kalman filter, one of the most common techniques in sensor fusion. Once again, the proposed solution demonstrated superior results in regular operation scenarios and under defective conditions in the input signals. The average deviation of the fusion estimates is in the range of 2.78 cm, while for the KFs with fixed and variable variance, it is in the range of 3.94 cm and 6.79 cm, respectively. These results demonstrated that while the KF significantly depends on the quality of its inputs, the network has a more extensive ability to assess the reliability of each input and possibly correct for systematic deviations.

Finally, this dissertation proposes two extended architectures that aim to capture information in the temporal domain as well. These use conventional DL structures such as CNNs and LSTMs to capture the dynamics models of the vehicle in question, learn certain patterns from the data at specific maneuvers, and make the output more physically coherent. The results showed that among all odometry techniques used for benchmarking, the framework with the CNN (CNN-FC) was the closest to the dynamics of the actual agent's motion.

In conclusion, the late fusion deep neural network has shown very effective results in creating consistent, precise and accurate estimates, surpassing its inputs individually on most occasions. The proposed solutions have also demonstrated to be competitive with state-of-the-art alternatives. Some shortcomings in highly complex scenarios justify the following scopes for future work:

- Train the network on more autonomous driving datasets, with a focus on scenarios with various moving obstacles. In parallel, endow the training data with induced errors that better mirror the actual signal failures;
- Train the network with more input signals, preferably with high accuracy and precision, to improve the reliability and robustness of the results;
- Fine-tune the current architectures and incorporate prior knowledge into the network training process. Experiment with more statistical metrics to describe the data and/or different combinations of these;
- Develop a parallel network for the classification of failure regimes in individual signals and pass the information to the leading network;
- Couple the current architecture with a vehicle dynamics model to filter and guide the produced estimates;
- Combine the model into an integrative platform that allows to map the surroundings and correct the trajectory through loop closure.

References

- [1] World Health Organization et al. Global status report on road safety 2018: summary. Technical report, World Health Organization, 2018.
- [2] Markus Maurer, J Christian Gerdes, Barbara Lenz, and Hermann Winner. *Autonomous driving: technical, legal and social aspects*. Springer Nature, 2016.
- [3] European Commission, Directorate-General for Mobility, and Transport. *Next steps towards 'Vision Zero' : EU road safety policy framework 2021-2030*. Publications Office, 2020.
- [4] Davide Margaria, Emanuela Falletti, and Tankut Acarman. The need for gnss position integrity and authentication in its: Conceptual and practical limitations in urban contexts. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 1384–1389. IEEE, 2014.
- [5] Mohammad OA Aqel, Mohammad H Marhaban, M Iqbal Saripan, and Napsiah Bt Ismail. Review of visual odometry: types, approaches, challenges, and applications. *SpringerPlus*, 5(1):1–26, 2016.
- [6] Davide Scaramuzza and Friedrich Fraundorfer. Visual odometry [tutorial]. *IEEE robotics & automation magazine*, 18(4):80–92, 2011.
- [7] Yusra Alkendi, Lakmal Seneviratne, and Yahya Zweiri. State of the art in vision-based localization techniques for autonomous navigation systems. *IEEE Access*, 9:76847–76874, 2021.
- [8] Dominique Gruyer, Valentin Magnier, Karima Hamdi, Laurene Claussmann, Olivier Orfila, and Andry Rakotonirainy. Perception, information processing and modeling: Critical stages for autonomous driving applications. *Annual Reviews in Control*, 44:323–341, 2017.
- [9] Johannes Graeter, Alexander Wilczynski, and Martin Lauer. Limo: Lidar-monocular visual odometry. In *2018 IEEE/RSJ international conference on intelligent robots and systems (IROS)*, pages 7872–7879. IEEE, 2018.
- [10] Ke Wang, Sai Ma, Junlan Chen, Fan Ren, and Jianbo Lu. Approaches challenges and applications for deep visual odometry toward to complicated and emerging areas. *IEEE Transactions on Cognitive and Developmental Systems*. early access. doi:10.1109/TCDS.2020.3038898.
- [11] David Nistér, Oleg Naroditsky, and James Bergen. Visual odometry. In *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, volume 1, pages 652–659. IEEE, 2004.
- [12] Hans Moravec. *Obstacle avoidance and navigation in the real world by a seeing robot rover*. PhD thesis, Stanford University, Stanford, CA, 1980.

- [13] Simon Lacroix, Anthony Mallet, Raja Chatila, and Laurent Gallo. Rover self localization in planetary-like environments. In *Artificial Intelligence, Robotics and Automation in Space*, volume 440, page 433, 1999.
- [14] Friedrich Fraundorfer and Davide Scaramuzza. Visual odometry: Part ii: Matching, robustness, optimization, and applications. *IEEE Robotics & Automation Magazine*, 19(2):78–90, 2012.
- [15] Sherif AS Mohamed, Mohammad-Hashem Haghbayan, Tomi Westerlund, Jukka Heikkonen, Hannu Tenhunen, and Juha Plosila. A survey on odometry for autonomous navigation systems. *IEEE Access*, 7:97466–97486, 2019.
- [16] David Nistér. An efficient solution to the five-point relative pose problem. *IEEE transactions on pattern analysis and machine intelligence*, 26(6):756–770, 2004.
- [17] Laurent Kneip, Davide Scaramuzza, and Roland Siegwart. A novel parametrization of the perspective-three-point problem for a direct computation of absolute camera position and orientation. In *CVPR 2011*, pages 2969–2976. IEEE, 2011.
- [18] Martin A Fischler and Robert C Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of the ACM*, 24(6):381–395, 1981.
- [19] David G Lowe. Distinctive image features from scale-invariant keypoints. *International journal of computer vision*, 60(2):91–110, 2004.
- [20] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool. Surf: Speeded up robust features. In *European conference on computer vision*, pages 404–417. Springer, 2006.
- [21] Ethan Rublee, Vincent Rabaud, Kurt Konolige, and Gary Bradski. Orb: An efficient alternative to sift or surf. In *2011 International conference on computer vision*, pages 2564–2571. IEEE, 2011.
- [22] Stefan Leutenegger, Margarita Chli, and Roland Y Siegwart. Brisk: Binary robust invariant scalable keypoints. In *2011 International conference on computer vision*, pages 2548–2555. IEEE, 2011.
- [23] Hsiang-Jen Chien, Chen-Chi Chuang, Chia-Yen Chen, and Reinhard Klette. When to use what feature? sift, surf, orb, or a-kaze features for monocular visual odometry. In *2016 International Conference on Image and Vision Computing New Zealand (IVCNZ)*, pages 1–6. IEEE, 2016.
- [24] Shashi Poddar, Rahul Kottath, and Vinod Karar. Motion estimation made easy: Evolution and trends in visual odometry. In *Recent Advances in Computer Vision*, pages 305–331. Springer, 2019.
- [25] Andry Maykol Pinto, A Paulo Moreira, Miguel V Correia, and Paulo G Costa. A flow-based motion perception technique for an autonomous robot system. *Journal of Intelligent & Robotic Systems*, 75(3):475–492, 2014.
- [26] Andry Maykol Pinto, Paulo G Costa, Miguel V Correia, Anibal C Matos, and A Paulo Moreira. Visual motion perception for mobile robots through dense optical flow fields. *Robotics and Autonomous Systems*, 87:1–14, 2017.

- [27] Andry Maykol Pinto, Miguel V Correia, A Paulo Moreira, and Paulo G Costa. Unsupervised flow-based motion analysis for an autonomous moving system. *Image and Vision Computing*, 32(6-7):391–404, 2014.
- [28] Edwin Olson, John Leonard, and Seth Teller. Fast iterative alignment of pose graphs with poor initial estimates. In *Proceedings 2006 IEEE International Conference on Robotics and Automation*, ICRA 2006., pages 2262–2269. IEEE, 2006.
- [29] Zana Zakaryaie Nejad. Design and implementation of a localization system for surveying robot based on visual odometry. Master’s thesis, K.N Toosi University of Technology, Faculty of Geodesy and Geomatics Engineering, 2018.
- [30] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *2012 IEEE conference on computer vision and pattern recognition*, pages 3354–3361. IEEE, 2012.
- [31] Raul Mur-Artal and Juan D Tardós. Orb-slam2: An open-source slam system for monocular, stereo, and rgb-d cameras. *IEEE transactions on robotics*, 33(5):1255–1262, 2017.
- [32] Raul Mur-Artal, Jose Maria Martinez Montiel, and Juan D Tardos. Orb-slam: a versatile and accurate monocular slam system. *IEEE transactions on robotics*, 31(5):1147–1163, 2015.
- [33] Igor Cvišić and Ivan Petrović. Stereo odometry based on careful feature selection and tracking. In *2015 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2015.
- [34] Igor Cvišić, Josip Česić, Ivan Marković, and Ivan Petrović. Soft-slam: Computationally efficient stereo visual simultaneous localization and mapping for autonomous unmanned aerial vehicles. *Journal of field robotics*, 35(4):578–595, 2018.
- [35] Jakob Engel, Thomas Schöps, and Daniel Cremers. Lsd-slam: Large-scale direct monocular slam. In *European conference on computer vision*, pages 834–849. Springer, 2014.
- [36] Igor Cvišić, Ivan Marković, and Ivan Petrović. Recalibrating the kitti dataset camera setup for improved odometry accuracy. In *2021 European Conference on Mobile Robots (ECMR)*, pages 1–6. IEEE, 2021.
- [37] Jakob Engel, Vladlen Koltun, and Daniel Cremers. Direct sparse odometry. *IEEE transactions on pattern analysis and machine intelligence*, 40(3):611–625, 2017.
- [38] Nan Yang, Rui Wang, Jorg Stuckler, and Daniel Cremers. Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 817–833, 2018.
- [39] Nan Yang, Lukas von Stumberg, Rui Wang, and Daniel Cremers. 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*, pages 1281–1292, 2020.
- [40] Sen Wang, Ronald Clark, Hongkai Wen, and Niki Trigoni. Deepvo: Towards end-to-end visual odometry with deep recurrent convolutional neural networks. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2043–2050. IEEE, 2017.
- [41] Guangyao Zhai, Liang Liu, Linjian Zhang, Yong Liu, and Yunliang Jiang. Poseconvgru: A monocular approach for visual ego-motion estimation by learning. *Pattern Recognition*, 102:107187, 2020.

- [42] Wang Zhao, Shaohui Liu, Yezhi Shu, and Yong-Jin Liu. Towards better generalization: Joint depth-pose learning without posenet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9151–9161, 2020.
- [43] Eddy Ilg, Nikolaus Mayer, Tonmoy Saikia, Margret Keuper, Alexey Dosovitskiy, and Thomas Brox. FlowNet 2.0: Evolution of optical flow estimation with deep networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2462–2470, 2017.
- [44] Nikhil Jonnavithula, Yecheng Lyu, and Ziming Zhang. Lidar odometry methodologies for autonomous driving: A survey. *arXiv preprint arXiv:2109.06120*, 2021.
- [45] Zhengyou Zhang. Iterative point matching for registration of free-form curves and surfaces. *International journal of computer vision*, 13(2):119–152, 1994.
- [46] Dmitry Chetverikov, Dmitry Svirko, Dmitry Stepanov, and Pavel Krsek. The trimmed iterative closest point algorithm. In *Object recognition supported by user interaction for service robots*, volume 3, pages 545–548. IEEE, 2002.
- [47] A Segal, D Haehnel, and S Thrun. Generalized-icp. robotics: Science and systems. *Seattle, USA*, 2009.
- [48] Steven A Parkison, Lu Gan, Maani Ghaffari Jadidi, and Ryan M Eustice. Semantic iterative closest point through expectation-maximization. In *BMVC*, page 280, 2018.
- [49] Peter Biber and Wolfgang Straßer. The normal distributions transform: A new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, volume 3, pages 2743–2748. IEEE, 2003.
- [50] Ji Zhang and Sanjiv Singh. Loam : Lidar odometry and mapping in real-time. *Robotics: Science and Systems Conference (RSS)*, pages 109–111, 01 2014.
- [51] Han Wang, Chen Wang, Chun-Lin Chen, and Lihua Xie. F-loam : Fast lidar odometry and mapping. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4390–4396, 2021. doi:10.1109/IROS51168.2021.9636655.
- [52] Xin Zheng and Jianke Zhu. Efficient lidar odometry for autonomous driving. *arXiv preprint arXiv:2104.10879*, 2021.
- [53] Yue Pan, Pengchuan Xiao, Yujie He, Zhenlei Shao, and Zesong Li. Mulls: Versatile lidar slam via multi-metric linear least square. *arXiv:2102.03771*, 2021.
- [54] Younggun Cho, Giseop Kim, and Ayoung Kim. Deeplo: Geometry-aware deep lidar odometry. *arXiv preprint arXiv:1902.10562*, 2019.
- [55] Weixin Lu, Guowei Wan, Yao Zhou, Xiangyu Fu, Pengfei Yuan, and Shiyu Song. Deepvcp: An end-to-end deep neural network for point cloud registration. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12–21, 2019.
- [56] Qing Li, Shaoyang Chen, Cheng Wang, Xin Li, Chenglu Wen, Ming Cheng, and Jonathan Li. Lo-net: Deep real-time lidar odometry. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 8473–8482, 2019.

- [57] Pierre Dellenbach, Jean-Emmanuel Deschaud, Bastien Jacquet, and François Goulette. Ct-icp: Real-time elastic lidar odometry with loop closure. *arXiv preprint arXiv:2109.12979*, 2021.
- [58] De Jong Yeong, Gustavo Velasco-Hernandez, John Barry, Joseph Walsh, et al. Sensor and sensor fusion technology in autonomous vehicles: A review. *Sensors*, 21(6):2140, 2021.
- [59] Wei Wang, Jun Liu, Chenjie Wang, Bin Luo, and Cheng Zhang. Dv-loam: Direct visual lidar odometry and mapping. *Remote Sensing*, 13(16):3340, 2021.
- [60] David Wisth, Marco Camurri, Sandipan Das, and Maurice Fallon. Unified multi-modal landmark tracking for tightly coupled lidar-visual-inertial odometry. *IEEE Robotics and Automation Letters*, 6(2):1004–1011, 2021.
- [61] Matti Kutila, Pasi Pyykönen, Werner Ritter, Oliver Sawade, and Bernd Schöufele. Automotive lidar sensor development scenarios for harsh weather conditions. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 265–270. IEEE, 2016.
- [62] Jacek Wojtanowski, Marek Zygmunt, Mirosława Kaszczuk, Z Mierczyk, and Michał Muzal. Comparison of 905 nm and 1550 nm semiconductor laser rangefinders’ performance deterioration due to adverse environmental conditions. *Opto-Electronics Review*, 22(3):183–190, 2014.
- [63] Young-Sik Shin, Yeong Sang Park, and Ayoung Kim. Dvl-slam: sparse depth enhanced direct visual-lidar slam. *Autonomous Robots*, 44(2):115–130, 2020.
- [64] Ji Zhang and Sanjiv Singh. Visual-lidar odometry and mapping: Low-drift, robust, and fast. In *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2174–2181. IEEE, 2015.
- [65] Chih-Chung Chou and Cheng-Fu Chou. Efficient and accurate tightly-coupled visual-lidar slam. *IEEE Transactions on Intelligent Transportation Systems*, 2021.
- [66] Jinyong Jeong, Younggun Cho, Young-Sik Shin, Hyunchul Roh, and Ayoung Kim. Complex urban dataset with multi-level sensors from highly diverse urban environments. *The International Journal of Robotics Research*, 38(6):642–657, 2019.
- [67] Milad Ramezani and Kourosh Khoshelham. Vehicle positioning in gnss-deprived urban areas by stereo visual-inertial odometry. *IEEE Transactions on Intelligent Vehicles*, 3(2):208–217, 2018.
- [68] Guy Zaidner and Amir Shapiro. A novel data fusion algorithm for low-cost localisation and navigation of autonomous vineyard sprayer robots. *Biosystems Engineering*, 146:133–148, 2016.
- [69] Xiaokun Leng, Songhao Piao, Song Wang, Lin Chang, and Zheng Zhu. An improved method for odometry estimation based on ekf and temporal convolutional network. *Physical Communication*, 43:101178, 2020.
- [70] Andres J Barreto-Cubero, Alfonso Gómez-Espinosa, Jesús Arturo Escobedo Cabello, Enrique Cuan-Urquiza, and Sergio R Cruz-Ramírez. Sensor data fusion for a mobile robot using neural networks. *Sensors*, 22(1):305, 2021.

- [71] T. Moore and D. Stouch. A generalized extended kalman filter implementation for the robot operating system. In *Proceedings of the 13th International Conference on Intelligent Autonomous Systems (IAS-13)*. Springer, July 2014.
- [72] Yiyi Liao, Jun Xie, and Andreas Geiger. KITTI-360: A novel dataset and benchmarks for urban scene understanding in 2d and 3d. *arXiv preprint arXiv:2109.13410*, 2021.
- [73] Bryan Lim and Stefan Zohren. Time-series forecasting with deep learning: a survey. *Philosophical Transactions of the Royal Society A*, 379(2194):20200209, 2021.
- [74] Andres J Barreto-Cubero, Alfonso Gómez-Espinosa, Jesús Arturo Escobedo Cabello, Enrique Cuan-Urquizo, and Sergio R Cruz-Ramírez. Sensor data fusion for a mobile robot using neural networks. *Sensors*, 22(1):305, 2021.
- [75] Yong Yu, Xiaosheng Si, Changhua Hu, and Jianxun Zhang. A review of recurrent neural networks: Lstm cells and network architectures. *Neural computation*, 31(7):1235–1270, 2019.