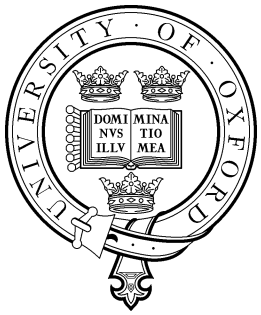
Learning Methods for Robust Localization



Changhao Chen

Pembroke College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Hilary 2020

Acknowledgements

I never expected that I can experience such wonderful four years at Oxford. This long journey towards a Ph.D. is challenging but also fun. I feel thankful to many people I met. Firstly, the most of my gratefulness goes to my supervisors, Professor Niki Trigoni and Professor Andrew Markham, for their delicate guidance and helpful support. Niki and Andrew provide me with the best platform and opportunities that allow me to explore the boundary of limits and unknowns in the frontier area. They have given me massive advice to shape my research taste, trained me to find novel and promising ideas, and helped me to systematically develop research skills, so that I am able to grow up as an independent researcher.

Secondly, I am thankful to a number of colleagues, whom I feel lucky enough to learn from and collaborate with. They are (in alphabetical order): Prince Abudu, Ronald Clark, Pedro Gusmao, Qingyong Hu, Shuyu Lin, Yishu Miao, Stefano Rosa, Risqi Saputra, Johan Wahlstrom, Bing Wang, Sen Wang, Wei Wang, Zhihua Wang, Hongkai Wen, Linhai Xie, Bo Yang and Peijun Zhao. I really appreciate their interesting and fruitful views on a wide range of research fields, that benefit this work. I am especially indebted to Chris Xiaoxuan Lu, who always offers deep insights and generous helps on solving various real-world problems encountered in our joint works.

Thank you to Department of Computer Science, and Pembroke College at the University of Oxford for hosting me with an enjoyable and creative environment over the past four years. I also would like to thank Prof. Alex Rogers, Prof. Nicholas Lane, Prof. Phil Blunsom, Prof. Yarin Gal, and Dr. Edward Jones for their constructive feedback and suggestions on my work during all the examinations I had.

Finally, my deepest gratitude goes to my parents, and all of my friends for keeping me energetic and enthusiastic throughout this journey. Without their support, I cannot image how to survive from a number of paper submission deadlines and finish this dissertation.

Abstract

Location awareness is a fundamental need for intelligent systems, such as self-driving vehicles, delivery drones, and mobile devices. Given their on-board sensors (e.g. camera, inertial sensor and LIDAR), previous researchers have developed a variety of localization systems, by building hand-crafted models and algorithms. Under ideal conditions, these sensors and models are able to accurately estimate system states without time bound. However, in real-world environments, many issues such as imperfect sensor measurements, inaccurate system modelling, complex environmental dynamics and unrealistic constraints, degrade the accuracy and reliability of localization systems. Therefore, this thesis aims to leverage machine learning approaches to overcome the intrinsic problems of the human-designed localization models.

This research presents learning methods to estimate self-motion using multimodal sensor data to achieve accurate and robust localization. Firstly, we exploit inertial sensor, a completely ego-centric and relatively robust sensor, to develop Inertial Odometry Neural Network (IONet) that learns motion transformation from raw inertial data, and reconstructs accurate trajectories. This inertial only solution shows impressive performance in locating people and wheeled objects without being influenced by environmental issues. IONet was further refined as L-IONet, a lightweight framework, to reduce the computational burden of model training and testing, and enable real-time inference on low-end devices. As a first trial in this direction, we collected and released Oxford Inertial Odometry Dataset (OxIOD) with a very large amount of inertial motion data collection containing 158 sequences totalling 42 km, to train and comprehensively evaluate our proposed models. Secondly, we present a novel generic framework to learn selective sensor fusion in enabling more robust and accurate odometry estimation and localization in real-world scenarios. Two fusion strategies are proposed: soft fusion, implemented in a deterministic fashion; and hard fusion, which introduces stochastic noise and intuitively learns to keep the most relevant feature representations, while discarding useless or misleading information. Both are trained in an end-to-end fashion, and

can be applied to complimentary pairs of sensor modalities, e.g. RGB images, inertial measurements, depth images, and LIDAR point clouds. We offer a visualization and interpretation of fusion masks to give deeper insights into the relative strengths of each stream. Finally, we leverage deep generative models to propose Sequential Invariant Domain Adaptation (SIDA) to mitigate the domain shift problem of the deep neural network based localization models. This framework works well on long continuous sensor data. Its key novelty is to use a shared encoder to convert the input sequence into a domain-invariant hidden representation, to encourage the useful semantic features obtained, whilst discarding the domain specific features. We employ proposed SIDA on deep learning based inertial odometry and human activity recognition to demonstrate its effectiveness in improving the generalization ability in new domains. We show that SIDA is able to transform raw sensor data into an accurate trajectory in new unlabelled domains, benefiting from the knowledge transferred from the labelled source domain. Through extensive experiments, all our proposed methods demonstrate their effectiveness and potential in achieving accurate and robust localization in real-world environments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Motivation

Location awareness is a fundamental need for human and mobile agents. As a particular motivating example, humans are able to perceive their self-motion via multimodal sensory perception, and rely on this awareness to locate and navigate themselves in a complex three-dimensional space [1], seemingly without conscious thought. Furthermore, this ability plays a vital role in developing perception, cognition, and motor control [2].

In a similar vein, artificial agents should be able to perceive the environment and estimate their system states using on-board sensors, such as cameras, radar, inertial sensor and Global Positioning System (GPS) [3]. These agents could be robots, e.g. self-driving vehicles, delivery drones or home service robots, autonomously sensing their surroundings and making decisions [4]. As another example, emerging Augmented Reality (AR) or Virtual Reality (VR) interfaces can enhance or entirely supplant the natural world, and the ability to maintain accurate perceptual awareness is key for optimal human-machine interaction. Further applications include mobile/smart devices, such as smartphones, wristbands or Internet-of-Things (IoT) devices, providing users with a wide range of location-based-services such as pedestrian navigation [5], sports/activity monitoring [6] or first-responder navigation [7]. Enabling a high level of autonomy for these and other digital agents requires robust and accurate egomotion awareness, with the ability to operate accurately and robustly in any environment. This challenge serves as the central overarching theme of this dissertation.

The process of localization can be summarized as determining self-motion, i.e. the relative change in pose, both in terms of translation and rotation, between two or more frames of sensor data, followed by integrating these pose changes with respect to an initial state to derive global pose in terms of location and orientation. Due to its importance for intelligent agents, exploiting sensor measurements for egomotion estimation and localisation has

been studied for decades. So far researchers have developed a variety of intricate analytical models - they formulate localization as a state estimation problem under the Markovian assumption: the current system states (e.g. orientations, velocities, and locations) are related to sensor observations and previous system states [8]. For example, Strapdown Inertial Navigation Systems (SINS), widely applied in flying and submersible robots, integrate inertial measurements directly to obtain locations [9]. For pedestrian navigation, Pedestrian Dead Reckoning (PDR) was proposed to infer a trajectory via step length and direction estimation using inertial sensors [5]. In robotics, models such as Visual Odometry (VO) [10, 11, 12], Visual-Inertial Odometry (VIO) [13, 14, 15, 16], Lidar Odometry (LO) [17], Simultaneous Localization and Mapping (SLAM) [18, 11, 19] were developed to exploit the on-board camera, IMU and Lidar for tracking robot pose and building maps. Under ideal conditions, these sensors and models are capable of accurately estimating system states without time bound. However, in real-world environments, imperfect sensor measurements, inaccurate system modelling, complex environmental dynamics and unrealistic constraints impact both the accuracy and reliability of localization systems. For example, the bias and noise of low-cost IMU sensors result in huge drifts of the inertial navigation systems (i.e. SINS) during long-term operation. Similar real-world challenges, including camera occlusion, changes in illumination, and missing or corrupted sensor data, degrade the performance of visual robotic systems. A natural question then arises: instead of crafting models by hand, can we build a data-driven model with the aid of machine learning to overcome the intrinsic problems of the human-designed approaches?

Recent advances in machine learning have achieved extensive and notable successes in various fields [20], from complex games, such as Go [21], to machine translation [22] and image recognition [23]. Compared to the sheer volume of research in other deep learning fields in computer vision and robotics, egomotion estimation has not been thoroughly explored. It is challenging to apply previous deep learning models designed for image or language understanding directly to the multimodal localization problem. In particular:

- Current motion estimation models mainly rely on visual observations but rarely focus on other more robust sensor data, for example, inertial sensors.
- Furthermore, models trained in one domain are hard to generalise to a new domain with a different data distribution, the well known distribution shift issue.
- There is a lack of effective fusion methods for learning from multiple sensor modalities, although classical sensor fusion has a long history.

- DNN based methods are criticised as being a 'black box' due to a lack of model interpretability.

Based on these issues, the main motivation of this thesis is: *can we develop learning methods to estimate self-motion using multimodal data to achieve accurate and robust localization, without relying on hand-crafted models?*

1.2 Research Challenges

This motivation brings with it a number of high-level challenges that will be addressed as part of this work. Although existing methods have tackled localization problem with either analytical models or deep learning, these approaches are fragile, and have not been scaled to the satisfactory level of deployment in practical applications. This research aims to overcome real issues faced in complex real-world environments to achieve robust and ubiquitous localization. These high-level challenges include:

- **Robustness**

  Both accuracy and robustness are essential in achieving usable localization for practical applications. The robustness of a localization system indicates its ability to tolerate perturbations, and provide accurate pose estimation consistently and reliably across a wide range of dynamic scenarios. The majority of existing methods have utilized visual observations to offer impressive results in particular scenarios. However, these vision-based models are confronted by a number of real factors, e.g. camera occlusions, low light conditions, environmental dynamics, that degrade their performance. In contrast, usage of another sensor modality that is more robust to environmental dynamics is far less explored in the localization problem. A good choice is an inertial sensor, which is completely ego-centric and insensitive to environmental perturbations. Overall, robustness is a key issue in system safety and reliability, allowing the system to localize under any operation condition. Learning from inertial data is a promising but underexplored research direction.

- **Effective sensor fusion**

  Another major challenge is to design an effective sensor fusion strategy for learning models. Generally, multiple sensors are available on autonomous systems: for example, self-driving vehicles are normally equipped with Lidar, camera and GPS/IMU as their motion perception modules. Effective sensor fusion is necessary for

constructing a robust and accurate motion perception system. This is due to the fact that different sensors are complementary to each other, and a single sensor modality may not be able to offer full observations for estimating system states. Fusing multimodal sensor data is also a way to improve robustness, i.e. by taking advantage of the complementary properties of different sensors. Developing a multimodal system will reduce the influences of the degradation of each modality, when input data are corrupted or missing. In classical analytical algorithms, a number of methods, such as filtering and smoothing, have been developed to combine information from different sources. However, there is a lack of systematic research into fusion strategies for deep learning models, especially in the problems of localization and motion estimation. Previous learning models do not consider possible sensor degradation or the complementary properties of different sensors. Thus, these learning models will suffer from degraded performance or even failures when deployed in complex real world. To date learning from multimodal sensor data for localization has not been adequately investigated.

- **Adaptation into new domains**

  The ability to generalize to a new domain is always a concern for data-driven methods. In situations where learning models are trained in one domain but tested in a different domain, the model performance will degrade. In practical applications, a number of influential factors, from sensor bias/noise to environmental/self-motion dynamics, will cause the test sensor data to be different from training data. Current DNN models need to be trained and evaluated with large labelled datasets. However, collecting labelled egomotion data requires a high-precision motion capture system (e.g. Vicon System) to generate ground-truth labels, together with a huge investment in effort and time, so it is impossible to collect labelled data for every possible test condition. Therefore, the challenge is to explore an effective way that enables the trained DNN models to automatically adapt into new domains without the need of labelled data and human effort.

1.3 Research Questions

In this dissertation, we consider how best to use and design machine learning approaches for localisation to tackle the above challenges. The main research questions that we investigate are as follows:

- **Question 1: How to learn to localize using noisy inertial sensor data?**

  To achieve robust localization, inertial motion sensing is a suitable choice, as it only relies on a self-contained sensor, requires no external physical infrastructure, and is insensitive to the operating environmental dynamics. However, low-cost inertial sensors, commonly found in robots and smartphones, are plagued by high sensor measurement bias and noise, leading to unbounded growth in system error. The inertial sensor biases indicate the offset of the average signal output, including constant bias error and bias instability, while inertial sensor noises are random measurement error corrupted on sensors, e.g. sensor white noises. It is not possible to apply deep learning models as used in image and language processing to the task of inertial tracking directly: compared with images, inertial data is highly temporally dependent and contains only self-motion information; in contrast to natural language, inertial data is a long continuous sequence, and the prediction target is a continuous real number rather than a discrete category. Moreover, there is no public dataset to train and evaluate the proposed learning based inertial tracking model. Overall, this research component will investigate how to design a suitable structure to learn from noisy inertial data for egomotion estimation and localization.

- **Question 2: How to conduct effective sensor fusion for deep learning models?**

  Deep neural networks are effective at extracting high-level features representative of egomotion from sensor modalities. Intuitively, the features from each modality offer different strengths. Very few works have considered the problem of effective feature fusion from multiple sensor modalities in the context of egomotion estimation. They either feed all features directly into other modules without any additional operation, or simply concatenate the features into one vector. These direct approaches do not consider the relative importance of features with respect to different sensor modalities, and environmental/measurement dynamics. Moreover, in real world operation, data corruption or loss, will result in large errors if all features are simply considered to be correct without a mechanism for filtering or selection. Intuitively, the features from different sensor modalities are not equally important, i.e. offer different strengths for motion estimation task. Therefore, this research investigates in a solution that effectively learns feature selection for robust sensor fusion.

- **Question 3: How to exploit unlabelled data to reconstruct accurate trajectories in a new domain?**

Training a deep learning model on a specific domain will cause the model to overfit to the given data distribution, and hence it will likely perform poorly in a new domain. Labelling data from all domains is not always possible, as it is time-consuming and costly. How to exploit unlabelled sensor data to improve generalization ability in a new environment is a practical and important question for real world usage. Specifically, the task of turning sensor measurements into pose and odometry estimates is hugely complicated by the fact that different placements and orientations of the devices, different gaits/motions, and biomechanics, lead to significantly different sensor observations. In the absence of labels in testing domains, the question becomes, how can the model adapt to an arbitrary domain in an unsupervised or self-supervised manner?

1.4 Contributions

This section summarizes the contributions of this dissertation as below:
Contribution 1: *Question 1* is addressed by casting inertial tracking as a sequential learning problem in *Chapter 3*. We first propose Inertial Odometry Neural Network (IONet), a novel deep neural network framework to learn self-motion and locate users using noisy inertial sensor. IONet is then refined as a Lightweight Inertial Odometry Neural Network (L-IONet), that reduces the training/testing time and computation, enabling real-time inference on low-end devices. In addition, we released the Oxford Inertial Odometry Dataset (OxIOD), a large-scale first-of-its-kind dataset for training and evaluating learning-based inertial odometry. These contributions are published in:

- **Changhao Chen**, Xiaoxuan Lu, Andrew Markham, Niki Trigoni. "IONet: Learning to Cure the Curse of Drift in Inertial Odometry", In the 32nd AAAI Conference on Artificial Intelligence (AAAI), 2018.
- **Changhao Chen**, Peijun Zhao, Chris Xiaoxuan Lu, Wei Wang, Andrew Markham, Niki Trigoni. "Deep Learning based Pedestrian Inertial Navigation: Methods, Dataset and On-Device Inference", Accepted by IEEE Internet of Things Journal.
- **Changhao Chen**, Chris Xiaoxuan Lu, Johan Wahlstrom, Niki Trigoni, Andrew Markham. "Deep Neural Network Based Inertial Odometry Using Low-cost Inertial Measurement Units", Accepted by IEEE Transactions on Mobile Computing.

Contribution 2: *Question 2* is addressed by introducing SelectFusion, a generic framework to learn sensor fusion from multimodal data, enabling more robust and accurate odometry estimation and localization in *Chapter 4*. This framework is implemented and evaluated in three different learning based tasks, i.e. visual-inertial odometry, depth-vision relocalization, and LIDAR-vision odometry, showing that it can selectively utilize suitable features for the problem at hand. The fusion mask can be visualized to indicate the system behaviours. These contributions are described in the following publications:

- **Changhao Chen**, Stefano Rosa, Yishu Miao, Chris Xiaoxuan Lu, Wei Wu, Andrew Markham, Niki Trigoni. "Selective Sensor Fusion for Neural Visual-Inertial Odometry", In International conference on Computer Vision and Pattern Recognition (CVPR), 2019.

- **Changhao Chen**, Stefano Rosa, Chris Xiaoxuan Lu, Niki Trigoni, Andrew Markham. "SelectFusion: A Generic Framework to Selectively Learn Multisensory Fusion", In submission.

Contribution 3: *Question 3* is addressed by proposing Sequential Invariant Domain Adaptation (SIDA), that operates over long continuous sensor data to transfer the learned knowledge from one domain to another domain in *Chapter 5*. Our framework reduces the effort in converting the raw sensor data into an accurate trajectory, as no labelled or paired data is required to achieve motion transformation in new domains. These contributions are published in:

- **Changhao Chen**, Yishu Miao, Chris Xiaoxuan Lu, Linhai Xie, Phil Blunsom, Andrew Markham, Niki Trigoni. "MotionTransformer: Transferring Neural Inertial Tracking Between Domains", In the 33rd AAAI Conference on Artificial Intelligence (AAAI), 2019.

During my D.Phil. study, I contributed to other research publications listed below. Given that these were partial contributions in related areas, but not at the heart of my D.Phil. research, they are not further discussed in this thesis.

- Bing Wang, **Changhao Chen**, Chris Xiaoxuan Lu, Peijun Zhao, Niki Trigoni, Andrew Markham. "AtLoc: Attention Guided Camera Localization", In the 34th AAAI Conference on Artificial Intelligence (AAAI), 2020.

- Chris Xiaoxuan Lu, Stefano Rosa, Peijun Zhao, Bing Wang, **Changhao Chen**, Niki Trigoni, Andrew Markham. "milliMap: Robust Indoor Mapping with Low-cost mmWave Radar", In the 18th ACM International Conference on Mobile Systems, Applications, and Services (MobiSys), 2020.

- Muhamad Saputra, Pedro Gusmao, Chris Xiaoxuan Lu, Yasin Almalioglu, Stefano Rosa, **Changhao Chen**, Johan Wahlstrom, Wei Wang, Andrew Markham, Niki Trigoni. "DeepTIO: A Deep Thermal-Inertial Odometry with Visual Hallucination", In the IEEE Robotics and Automation Letters, 2020.

- Peijun Zhao, Chris Xiaoxuan Lu, Bing Wang, **Changhao Chen**, Mengyu Wang, Andrew Markham, Niki Trigoni. "Heart Rate Sensing with a Robot Mounted mmWave Radar", In the International Conference on Robotics and Automation (ICRA), 2020.

- Linhai Xie, Yishu Miao, Sen Wang, Phil Blunsom, Zhihua Wang, **Changhao Chen**, Andrew Markham, Niki Trigoni. "Learning with Stochastic Guidance for Robot Navigation", In the IEEE Transactions on Neural Networks and Learning Systems, 2020.

- Wei Wang, Muhamad Saputra, Peijun Zhao, Pedro Gusmao, Bo Yang, **Changhao Chen**, Andrew Markham, Niki Trigoni. "DeepPCO: End-to-End Point Cloud Odometry through Deep Parallel Neural Network". In IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2019.

- Chris Xiaoxuan Lu, Xuan Kan, Bowen Du, **Changhao Chen**, Hongkai Wen, Andrew Markham, John A Stankovic. "Autonomous Learning for Face Recognition in the Wild via Ambient Wireless Cues". In The Web Conference (WWW), 2019.

- Peijun Zhao, Chris Xiaoxuan Lu, Jianan Wang, **Changhao Chen**, Andrew Markham, and Niki Trigoni. "mID: Privacy-Preserving Tracking and Identification with Millimeter Wave Radar". In International Conference on Distributed Computing in Sensor Systems (DCOSS), 2019.

- Chris Xiaoxuan Lu, Yuanbo Xiangli, Peijun Zhao, **Changhao Chen**, Niki Trigoni, and Andrew Markham. "Automatic Speaker Recognition and WiFi Geofence Adaptation via Cross-modal Labelling". IEEE Internet of Things Journal, 2019.

- Chris Xiaoxuan Lu, Yang Li, Peijun Zhao, **Changhao Chen**, Linhai Xie, Hongkai Wen, Rui Tan, Niki Trigoni. "Simultaneous Localization and Mapping with Power

Network Electromagnetic Field". In Annual International Conference on Mobile Computing and Networking (MobiCom), 2018.

1.5 Thesis Outline

The outline of this thesis is presented as follows: Chapter 2 overviews the background of this research; Chapter 3 proposes learning-based inertial odometry models; Chapter 4 studies sensor fusion strategies for deep learning models; Chapter 5 presents a sequence domain adaptation solution; Chapter 6 summarizes the overall conclusions and suggests directions for future work.

Chapter 2

Background

This chapter provides a comprehensive overview of *conventional model based localization solutions*, recent advances in *deep learning*, and existing *learning based localization approaches*. In addition, we offer basic concepts relevant to this research.

2.1 Sensors and Systems

Given sensing data, the aim of a localization system is to track the position and orientation of mobile agents. Possible sensing modalities are GPS/radio signals, inertial, visual and LIDAR point cloud data, as mobile agents are often equipped with a combination of these sensors. Figure 2.1 shows four representative intelligent systems: (a) smartphones (b) virtual-reality devices (c) self-driving vehicles and (d) drones. Generally, those systems are deployed in complex dynamic environments, and hence impose high requirements on achieving ubiquitous and robust localization with their onboard sensors.

A self-driving vehicle (e.g. the Oxford RobotCar project in Figure 2.1 (c) [24][1]), equipped with LIDAR, GPS, IMU and cameras, rely on these sensors to locate and navigate itself in dynamic environments with moving vehicles and pedestrians. It should work well both in open areas and challenging cluttered spaces such as tunnels and city areas with high buildings. Consumer drones (e.g. DJI Mavic Pro camera drone in Figure 2.1 (d) [2]) have successfully exploited GPS, IMU and cameras for large-scale localization, navigation and obstacle avoidance. The situations where drones are deployed for emergency rescue and product delivery, require their localization systems to be robust and accurate even under

[1]The Oxford RobotCar: https://robotcar-dataset.robots.ox.ac.uk/
[2]DJI Mavic Pro: https://www.dji.com/uk/mavic

(a) iPhone 11 Pro

(b) Oculus VR Device

(c) Oxford RoboCar Self-driving Vehicle

(d) DJI Mavic Drone

Figure 2.1: An overview of common intelligent systems with a need of robust localization. (a) smartphone - iPhone 11 Pro (b) virtual-reality (VR) device: Oculus (c) self-driving vehicle - Oxford RobotCar [24] (d) drones - DJI Mavic Camera Drone.

pertubations, such as extreme weather or illumination changes. Beyond the robotic systems mentioned above, cameras, IMU and GPS are available on smartphones (e.g. iPhone 11 Pro in Figure 2.1 (a) [3]) and Virtual Reality (VR) devices (e.g. Oculus VR device in Figure 2.1 (b) [4]), to track a user's motion and enable personal applications. As people spend a majority of their time indoors, it is more challenging to provide accurate locations in GPS-denied areas.

2.2 Model Based Localization Solutions

Global Positioning System (GPS) solves most large-scale positioning problems in outdoor areas. However, GPS is not able to provide orientation information, and suffers from serious attenuation or multi-path effects in and around buildings. Numerous radio-based techniques were developed to solve indoor positioning, including RFID [25, 26, 27], WiFi

[3]iPhone 11 Pro: https://www.apple.com/uk/iphone-11-pro/
[4]Oculus VR: https://www.oculus.com/

[28, 29, 30, 31, 32, 33], Bluetooth [34, 35], and Ultra-wideband (UWB) [36] based positioning systems. These methods are highly dependent on infrastructure that has been installed and setup before use, limiting their mobility and scalable usage. In contrast, tracking mobile agents with self-contained sensors offers great flexibility and mobility, and hence has attracted attention from industry and academia over the past decades. Previous work typically builds analytical models to describe and estimate motion dynamics.

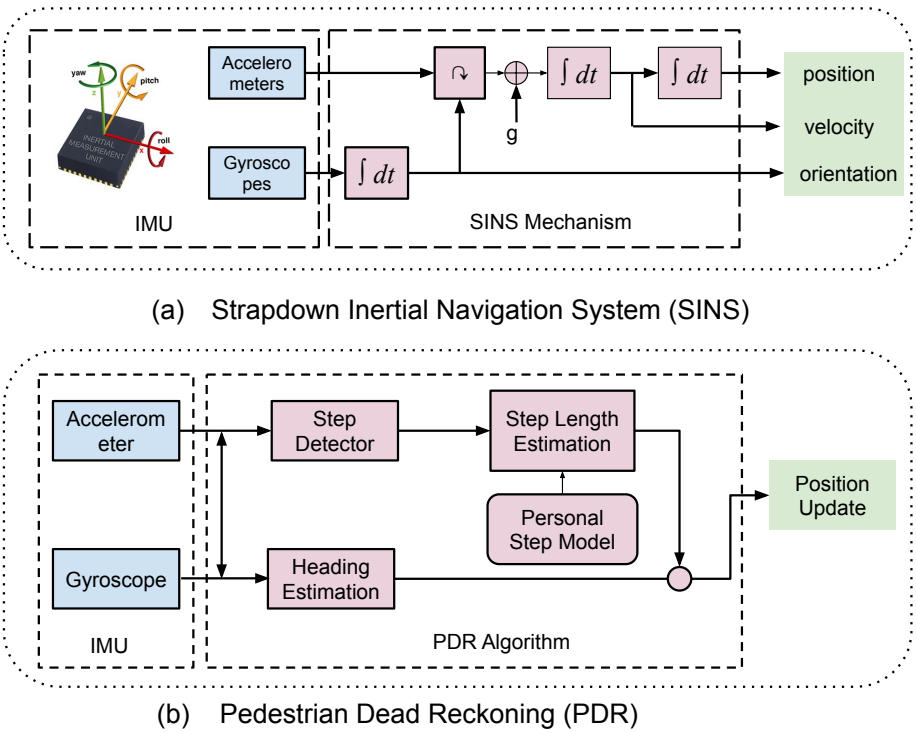
This dissertation mainly studies an usage of common on-board inertial, visual and LIDAR sensors to locate intelligent systems. Positioning techniques with other sensor modalities including radio-based [37], acoustic-based [38] and magnetic-based approaches [39, 40, 41], are not the main target of this research, so they are not covered in this section. To provide background information relevant to this thesis, we discuss the existing mainstream inertial-based, visual-based and LIDAR-based positioning approaches as below.

2.2.1 Inertial Positioning Systems

Early inertial positioning systems were developed as the core components in control and navigation systems for aircraft, submarines, and spacecraft, relying on expensive, heavy and high-precision inertial measurement units (IMUs). Based on Newtonian mechanics, the Strapdown Inertial Navigation Systems (SINS) integrate inertial measurements directly into orientation, velocity and location [42, 9]. Recent advances within micro-electro-mechanical (MEMS) technology has enabled the production of IMUs with significantly lower cost, size and energy consumption. As a result, MEMS IMUs are deployed on robots, unmanned aerial vehicle (UAV) and mobile devices. However, the accuracy of a MEMS IMU is very limited, and hence the SINS algorithms are hard to realize on low-cost inertial sensors. Via open integration, the high sensor noise and bias of low-cost IMUs cause exponential system error propagation such that the inertial navigation systems will collapse within seconds.

The inertial measurements from low-cost MEMS IMUs are normally corrupted with various error sources, such as sensor white noise, random walking noise, thermo-mechanical noise, scale factor, and axis misalignment [43]. These sensor errors are mainly categorized into two parts: the deterministic terms, e.g. constant bias, axis misalignment, can be removed via a certain calibration method; the stochastic terms, e.g. random white noises, are hard to be removed directly, but modelled as stochastic processes [44]. The error model of inertial sensor can be formulated as:

$$\hat{\omega} = \omega + b_\omega(t) + n_\omega(t) \tag{2.1}$$



(a) Strapdown Inertial Navigation System (SINS)

(b) Pedestrian Dead Reckoning (PDR)

Figure 2.2: An overview of two existing inertial navigation models: (a) strapdown inertial navigation system (SINS) (b) pedestrian dead reckoning (PDR).

$$\hat{a} = a + b_a(t) + n_a(t), \tag{2.2}$$

where $(\hat{\omega}, \hat{a})$ are the angular rates and acceleratations measured by gyroscope and accelerometer. They contain the real motion values (ω, a), a random noise term that changes very fast $n(t)$, and a bias term that varies slowly $b(t)$ [45]. In practice, this stochastic noise is modelled with a zero-mean, independent, white Gaussian noise, while the bias term is modelled as a Bownnian motion or random walk process. To determine the parameters of error model, a popular choice is using Allan Variance to analyse and calibrate an inertial sensor [46, 43].

To mitigate the unbounded error drift, domain specific knowledge e.g. human motion constraints have been incorporated to enhance the accuracy of the inertial navigation system in the context of pedestrian navigation. One solution is to perform a so-called zero-velocity update (ZUPT) whenever the foot is detected to be at standstill [47]. Essentially, this means that the navigation system uses the fact that the foot will be intermittently stationary to mitigate error growth. Unfortunately, this approach relies on the assumption that the inertial sensor is firmly attached to the foot, and the pedestrian user adheres to a standard periodic walking motion. Another solution is step-based pedestrian dead reckoning (PDR), which

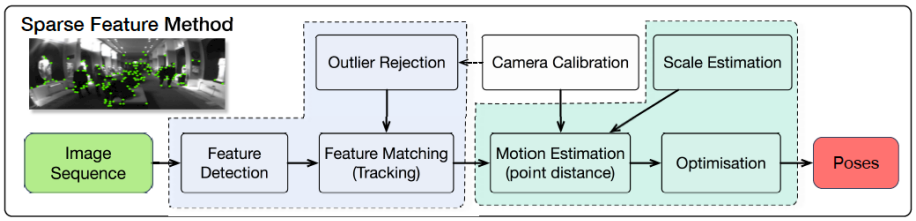
infers positional and rotational displacement quantities by detecting steps, estimating step length and direction, and then updating the position estimates accordingly [5]. Generally, the models used for estimating step length and step direction can be said to contain implicit motion models. Consequently, a PDR system sometimes has slower error growth than a strapdown inertial navigation system. However, the performance of the dynamic step estimation may be degraded by sensor noise, variations in the user's walking habits, and changes in the phone attachment [48, 49]. Moreover, in many navigation situations of interest, no steps can be detected. For example, if a phone is placed on a baby stroller or a shopping trolley, the assumption of periodicity, exploited by step-based PDR, breaks down. Recent attempts focused on fusing PDR with external references, such as a floor plan [50], WiFi fingerprinting [51] or geomagnetic fields [52] to correct system error, but still leaves the fundamental problem of rapid inertial system drift unsolved.

The architecture of two existing mainstream algorithms, i.e. SINS and PDR, is illustrated in Figure 2.2. In summary, both ZUPT-aided inertial navigation and step-based PDR are limited by assumptions on motion dynamics and sensor attachment that prevent widespread adoption in daily life.

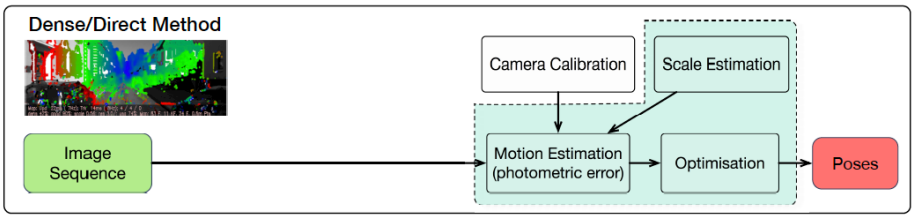
2.2.2 Visual Odometry

Visual odometry (VO) estimates the ego-motion of a camera by exploiting geometric constraints from consecutive images. Geometry-based VO can be further categorized into sparse feature based and dense/direct methods, according to whether salient features are used or not. Sparse feature based methods follow a process of extracting salient features of raw images, matching the correspondence of feature points, and determining the camera pose according to the geometric relation of these feature points among consecutive images [10]. The performance of sparse feature methods depend on the quality of feature extraction and matching, which is also computationally expensive. The other disadvantage is that only salient features participate in estimating motion, without exploiting the rich appearance and semantic information of whole scene.

In contrast, dense/direct methods abandon hand-designed feature detection algorithm, but instead enforce photometric consistency of all pixels between images for pose estimation [53]. Dense/direct methods show better performance than sparse feature methods, especially in featureless scenarios, due to the raw pixels used in the photometric optimization. However they still suffer from the problem of inaccurate photometric alignment. A comparison of sparse feature and dense/direct methods is illustrated in Figure 2.3.



(a) Sparse feature based method.



(b) Dense/Direct method.

Figure 2.3: The structure of conventional sparse feature visual odometry and direct visual odometry [54]

2.2.3 Visual SLAM and Relocalization

The intrinsic issues inside existing VO systems, such as outliers and noise, result in inevitable and accumulative error drifts over time. In order to constrain this visual system drift, the visual Simultaneous Localization and Mapping (SLAM) systems build a world map simultaneously along side pose estimation. With the aid of the created map, a loop closure detection module is added to recognize places that have been previously visited, and jointly optimize pose estimation and map building. MonoSLAM [18] is the first monocular SLAM system, which incorporates sparse features into Kalman filtering as system states. Keyframe based SLAM, e.g. Parallel Tracking and Mapping (PTAM) [55], treats mapping and pose tracking as two individual procedures, and operates in two parallel threads, allowing real-time inference. ORB-SLAM [19] is one of the state-of-the-art SLAM systems, which is feature based and operates in real-time. ORB-SLAM detects and builds a map of sparse ORB features, which are robust to motion clutter, and allows precise self-motion tracking, mapping, relocalization, and loop closure. Large-Scale Direct Monocular SLAM (LSD-SLAM) [11] is a direct SLAM system, tracking camera pose and building maps by using keyframe based photometric alignment and pose graph backend.

Relocalization is a desired property when the system is found to be 'lost' (the famous 'kidnapped' robot problem [56]). When a 3D map is available from visual SLAM or its counterpart structure-from-motion (SfM), the camera pose can be retrieved by exploiting 3D-to-3D matching [57] or 2D-to-3D matching [58] between the query images and the map. These can be viewed as metric-based methods. Appearance-based localization clas-

sifies images according to a number of known scenarios and produces corresponding locations. For example, FAB-MAP [59] adopts the SIFT features from images to probabilistically recognize places based on bag-of-words model.

2.2.4 Visual-Inertial Odometry

Integrating visual and inertial information in the form of Visual-Inertial Odometry (VIO) enables ubiquitous mobility for mobile agents by providing robust and accurate pose information. Both cameras and inertial sensors are relatively low-cost, power-efficient and widely found in ground robots, smartphones, and drones. Single cameras are able to capture the appearance and structure of a 3D scene. However, they are scale-ambiguous, and not robust to most challenging scenarios, e.g. strong lighting changes, lack of texture and high-speed motion. In contrast, IMUs are completely ego-centric, scene-independent, and can provide absolute metric scale, but inertial measurements are corrupted by process noise and biases. Conventional visual-inertial approaches exploit the complementary properties of two sensors to achieve accurate pose estimation.

Traditionally, visual-inertial approaches can be roughly segmented into three different classes according to their information fusion methods: filtering approaches [13], fixed-lag smoothers [14] and full smoothing methods [15]. Under the Markov assumption, filtering based VIOs [60, 61, 13] infer system states using the latest states instead of full past states, and hence enable efficient pose estimation. Due to its computational efficiency, early VIO systems were developed within this category, e.g. the Multi-State Constraint Kalman filter (MSCKF) [60]. However, on one hand, linearization error and other system drifts are hard to mitigate in the process of odometry estimation, because the system uses the current states and measurements only once, and discards the older states permanently. On the other hand, the linearization approximation is always a problem for filtering methods, due to the fact that VIO is a highly nonlinear system. To address these, fixed-lag smoother methods [62, 63, 14] propose to optimize state estimation within a given time window rather than only over the latest time step. Generally by incorporating multiple observations, fixed-lag smoother approaches are more accurate and robust than filtering based VIO, but at the expense of higher computation. Full smoothing methods [64, 65, 66, 15] estimate states conditioned on the entire history of system states. Theoretically, full smoothing methods achieve the best performance. However, this process solves a complex nonlinear optimization problem. Its complexity grows cubically with respect to state dimension, making the system hard to scale. Several approaches here proposed to address this complexity problem by introducing incremental smoothing frameworks [66, 15].

In classical VIO models, features are handcrafted, for example, OKVIS [14] presented a keyframe-based approach that jointly optimizes visual feature reprojections and inertial error terms. Semi-direct [67] and direct [68] methods have been proposed in an effort to move towards feature-less approaches, removing the feature extraction pipeline for increased speed. IMU-preintegration [15] provides a theoretical proof of how to avoid continuous preintegration of inertial measurements, thus improving computational speed. Recently, VINS-Mono [16] was proposed as a fast, tightly-coupled, sliding window-based optimization approach for VIO. Although VIO systems can provide quite accurate poses by effectively exploiting visual and inertial information, they will be influenced by the quality of sensor data and the motion/environmental conditions. In some cases, for example, if two sensors are not tightly time syncronized, or the change of lighting conditions impacts visual feature detection, their performance will be degraded.

2.2.5 LIDAR Odometry

LIDAR sensors provide high-frequency range measurements, with the benefits of working consistently in complex lighting conditions and optically featureless scenarios. LIDAR odometry can exploit the high accuracy of LIDAR sensor data to produce high-precision 6-DoF pose. LIDAR Odometry and Mapping (LOAM) [69] splits the problem of odometry and mapping into a high frequency/low accuracy process for motion estimation and a low frequency/high accuracy pose refinement process, as illustrated in Figure 2.4. Both processes use a feature detector to extract structural points, i.e. the points on edges and planes, and match them to edge line segments and planar surfaces. The odometry estimation algorithm performs faster to find the correspondence of these feature points for motion estimation, while the mapping algorithm runs at a lower frequency to register fine point clouds. To the same extent, [70, 71] further extend LOAM to use the fusion of LIDAR, monocular camera and IMU for achieving higher accuracy and robustness.

The performance of LIDAR odometry is sensitive to the point cloud registration errors due to non-smooth motion. And the data quality of LIDAR measurements is also affected by extreme weather conditions, for example, heavy rain or low hanging clouds.

2.2.6 Discussion

The localization solutions discussed above can all be summarized as crafting a mapping function to describe the relation between the input sensor data (e.g. visual, inertial, and LIDAR data), and the output target values (e.g. location, velocity and orientation).

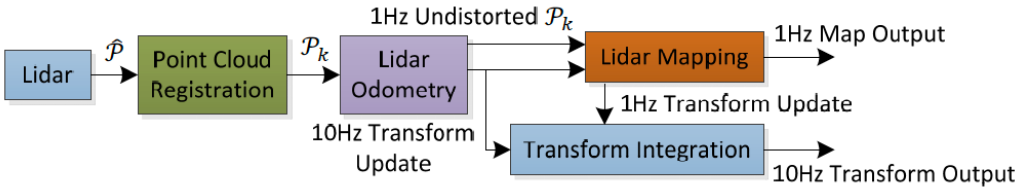


Figure 2.4: An illustration of LIDAR odometry and mapping system, consisting of point cloud registration, LIDAR odometry estimation, mapping, and transform integration [69]

In classical model-based localization solutions, this mapping function is built by hand-designed models and algorithms, as shown in Figure 2.5. Although they are able to provide accurate pose information in some scenarios, they are still limited due to the complexity of real-world applications. Their performance will be influenced by sensor noise and bias, imprecise system modelling, environmental factors, and unrealistic assumptions. For inertial solutions, the bias and noise of low-cost inertial sensor cause inevitable system drifts during long-term operation in strapdown inertial navigation algorithm, while Pedestrian Dead Reckoning (PDR) assumes the body properties and periodic motion of users, limiting their potential usages. Visual solutions either depend on the salient features extracted by hand-designed detectors (feature based visual odometry) or the photometric alignment (direct visual odometry) to estimate motion. They are sensitive to environmental dynamics and lighting conditions. By fusing visual and inertial sensor as visual-inertial odmetry (VIO), more accurate pose can be provided. However, possible sensor degradation will reduce the performance of VIO systems, such as incorrect system initialization and calibration, image occlusions, or missing data. Similarly, the performance of LIDAR odometry is sensitive to some real issues, for example, they will not work well in heavy rain or under non-smooth motion.

These limitations motivate us to design data-driven models for solving the task at hand. The advantage of learning based methods is their potential robustness to lack of features, dynamic lightning conditions, motion blur, accurate camera calibration, which are hard to model by hand [4]. Next section will discuss several deep learning methods that are relevant to this research.

2.3 Deep Learning Methods

Recent advances in deep learning have enabled great breakthroughs in processing raw images, video, speech and audio for solving a variety of tasks [20]. Deep learning has pow-

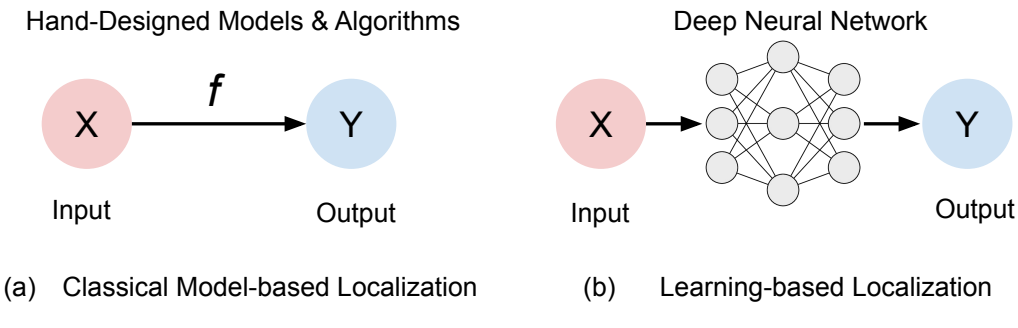


Figure 2.5: (a) Conventional localization solutions are built by hand-designed models and algorithms. (b) Data-driven solutions exploit deep neural network to learn self-motion or locations directly from input data.

ered many research areas and achieved state-of-the-art performance, from image recognition [72, 23], machine translation [73, 74], to recommender systems [75]. Deep learning techniques leverage deep neural networks (DNNs) to automatically discover and extract high-level representations from high-dimensional raw data, that are necessary for the task at hand, i.e. representation learning. Deep neural networks are extremely expressive models that are composed of multiple processing layers of neurons with learnable weights and biases. The DNN framework is typically trained and optimized via backpropagation [76] and stochastic gradient descent (SGD) [77] with respect to the end goal, usually a loss function between model predictions and groundtruth labels. Instead of building analytical algorithmic models as in conventional ways, deep learning approaches are data-driven, require little hand-engineering and hence can take advantage of increases in both computation power and data volumes. In this section, we discuss several common model structures and their corresponding learning methods that are relevant to the research in this thesis.

2.3.1 Convolutional Neural Networks

Convolutional Neural Networks (ConvNets) aim to process data with a known, grid-like structure [78]. For example, an image is a regular array of pixel intensities in 3 dimensions: width, height, and depth (the number of channels). ConvNets are effective at processing images to extract high-level representations for visual tasks.Normally a ConvNet model consists of convolutional layers, pooling layers and fully-connected layers, transforming raw images to layers of activation and finally to target values. Convolutional layer performs convolution operations on local regions of the input from each layer. After the

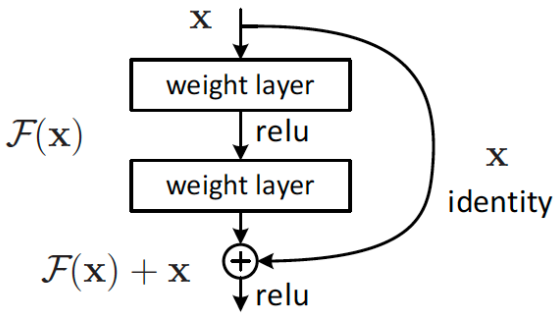


Figure 2.6: The residual learning in ResNet model: instead of mapping $F(x) + x$ directly using stacked layers, residual learning fits the residual part $F(x)$ [23].

convolutional layers, pooling layer is added to downsample the neuron outputs. In the end, Fully-connected (FC) layers map the features from previous layers to the final target values.

ConvNets have brought great success in image understanding, achieving the state-of-art performance in a wide range of visual tasks, e.g. the detection, segmentation, and recognition of objects and regions in images. AlexNet [72] is the first deep ConvNet work, that significantly outperformed other competitors in the ImageNet ILSVRC Challenge, and hence attracted the attention of computer vision community towards the adoption of deep ConvNet model. GoogleNet [79] introduces an Inception Module that dramatically reduces the number of parameters in the deep neural network. The work of VGGNet [80] includes very small convolutional filters into the model, and proves that the depth of neural network is a key point for improving model performance. ResNet [23] presents a residual learning framework to address the gradient vanishing problem of training very 'deep' neural networks. Instead of learning a mapping function directly using stacked layers, the residual module uses the layers to learn a residual function, as shown in Figure 2.6. Due to its impressive performance, ResNet has become a default choice when using ConvNet in practice.

ConvNets have been successfully applied to extract features from raw images for camera relocalization [81, 82, 83, 84], visual odometry [85, 86, 87, 88] and visual-inertial odometry [89]. Inspired by their work, we also use ConvNets to extract useful features from RGB images, depth images, and LIDAR data for proposed selective sensor fusion strategies in Chapter 5.

2.3.2 Recurrent Neural Networks

Recurrent Neural Networks (RNNs) are designed to process sequential data for exploiting the temporal dependencies among data. Recurrent models maintain hidden states over time,

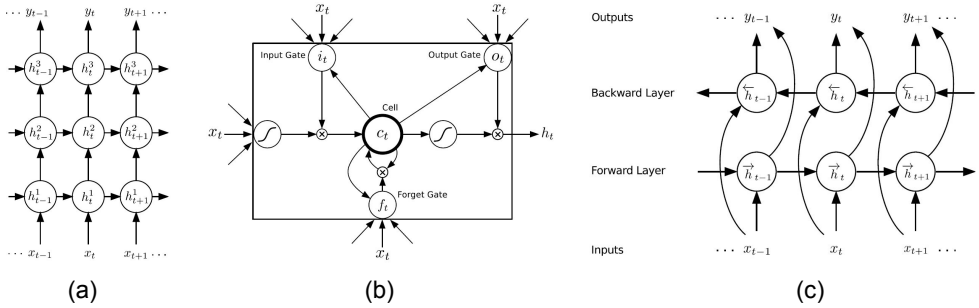


Figure 2.7: The architectures of recurrent neural networks: (a) standard Recurrent Neural Network (RNN) (b) LSTM (c) Bidirectional Recurrent Neural Networks (Bi-RNN) [90]

that contain the history information about all the past states of sequence.

Given an input sequence $\mathbf{x} = (\mathbf{x}_1, ..., \mathbf{x}_T)$ with T frames of data, an RNN model updates its hidden vectors $\mathbf{h} = (\mathbf{h}_1, ..., \mathbf{h}_T)$ and produces output vectors $\mathbf{y} = (\mathbf{y}_1, ..., \mathbf{y}_T)$ via the Equations (2.3):

$$\begin{aligned} \mathbf{h}_t &= \mathcal{H}(\mathbf{W}_{ih}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}_{t-1} + \mathbf{b}_h) \\ \mathbf{y}_t &= \mathbf{W}_{ho}\mathbf{h}_t + \mathbf{b}_o, \end{aligned} \tag{2.3}$$

where t is the time step of a sequence, $\mathbf{W}$ and $\mathbf{b}$ denote the weight matrices, and bias vectors of neural network. $\mathcal{H}$ is the activation function of hidden layer, which is usually an elementwise implementation of a sigmoid function.

However, due to the well-known exploding and vanishing gradient problems [91], recurrent models are hard to train, especially in processing long sequences. To address these, the Long Short-Term Memory (LSTM) framework [92] was proposed. The central idea behind LSTM is to use a memory cell for maintaining its hidden state over time, and regulate the information flow into and out of the cell. The hidden state $\mathbf{h}_t$ at the time step t is updated by iterating the Equations (2.4) [90]:

$$\begin{aligned} \mathbf{i}_t &= \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{W}_{ci}\mathbf{c}_{t-1} + \mathbf{b}_i) \\ \mathbf{f}_t &= \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{gf}\mathbf{h}_{t-1} + \mathbf{W}_{cf}\mathbf{c}_{t-1} + \mathbf{b}_f) \\ \mathbf{c}_t &= \mathbf{f}_t\mathbf{c}_{t-1} + \mathbf{i}_t\text{tanh}(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \\ \mathbf{o}_t &= \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{W}_{co}\mathbf{c}_t + \mathbf{b}_o) \\ \mathbf{h}_t &= \mathbf{o}_t\text{tanh}(\mathbf{c}_t), \end{aligned} \tag{2.4}$$

where σ is the logistic sigmoid function, $\mathbf{i}, \mathbf{f}, \mathbf{o}, \mathbf{c}$ are the input gate, forget gate, output gate and cell activation vectors respectively.

Similarly, the Gated Recurrent Unit (GRU) [93] is a lightweight recurrent model to control the information flow. The hidden state of GRU model $\mathbf{h}$ is updated via the Equations

(2.5):

$$\begin{aligned}
\mathbf{r}_t &= \text{sigm}(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1} + \mathbf{b}_t) \\
\mathbf{z}_t &= \text{sigm}(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{W}_{hz}\mathbf{h}_{t-1} + \mathbf{b}_z) \\
\tilde{\mathbf{h}}_t &= \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h) \\
\mathbf{h}_t &= \mathbf{z}_t \odot \mathbf{h}_{t-1} + (1 - \mathbf{z}_t) \odot \tilde{\mathbf{h}}_t,
\end{aligned} \tag{2.5}$$

where $\mathbf{r}$ and $\mathbf{z}$ are GRU cells, $\mathbf{W}$ and $\mathbf{b}$ are the weight matrices and bias vectors, and $\odot$ denotes the elementwise multiplication operation.

Traditional recurrent models only use previous context information. In some scenarios, both previous and future states are crucial to updating current states. Therefore, a variant of the recurrent model, i.e. bidirectional recurrent neural networks (Bi-RNNs) [94] have been introduced to exploit the dynamic information context in both sides. This bidirectional architecture processes input data in both past and future directions by including two separate hidden layers, which represents to handle data forward and backward. A concatenation layer then combines the outputs from the forward and backward layer, and updates the hidden states $\mathbf{h}_t$ at time step t via the Equations 2.6:

$$\mathbf{h}_t = [\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t], \tag{2.6}$$

where $\overrightarrow{\mathbf{h}}$, $\overleftarrow{\mathbf{h}}$ denote the hidden states from the forward and backward layer.

The architectures of three recurrent neural networks, i.e. standard RNN, LSTM and Bidirectional RNN, are illustrated in Figure (2.7) [90].

Recurrent neural networks have been successfully applied for solving various tasks with time-series data, e.g. speech recognition [95], machine translation [96], and video description [97]. In the context of localization, previous work exploited recurrent neural networks to model the temporal dependencies of consecutive data for solving visual odometry [85] and visual-inertial odometry [89]. To the best of our knowledge, RNNs have never been used to achieve inertial only localization. Chapter 3 will show that our proposed Inertial Odometry Neural Network (IONet) framework can learn self-motion from inertial data, and achieve robust and accurate localization using recurrent neural networks.

2.3.3 Generative Adversarial Networks

Generative adversarial networks (GANs) [98] are a kind of deep generative model in which a generator network competes against a discriminator network. The generator G generates fake samples $\mathbf{x} = G(\mathbf{z}, \theta_G)$ conditioned on a hidden representation $\mathbf{z}$. The discriminator D distinguishes between the real samples from the training set and the fake samples generated by the generator. The discriminator network $D(\mathbf{x}, \theta_D)$ produces a probability value that

describes the belief in whether the sample is real or fake, i.e. either from the training set or the generator. The training procedure of GAN can be viewed as a zero-sum game: the discriminator D is trained to maximize the probability that the discriminator successfully classifies the samples from the real or fake category; meanwhile, the generator G is trained to increase the possibility that the discriminator is 'fooled' (i.e. not able to distinguish the samples). This process can be summarized as a two-layer minimax game between the generator and discriminator with an optimal objective:

$$\min_G \max_D \{\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))]\}, \tag{2.7}$$

where the sample $\mathbf{x}$ from dataset follows the real data distribution $\mathbf{x} \sim p_{data}(\mathbf{x})$, the hidden representation $\mathbf{z}$ follows a prior distribution $\mathbf{z} \sim p_z(\mathbf{z})$. Therefore, the object of the generator is to minimize the following function i.e. GAN loss, while the discriminator is to maximize it:

$$L(\theta) = \{\mathbb{E}_{\mathbf{x}\sim p_{data}(\mathbf{x})}[\log D(\mathbf{x})] + \mathbb{E}_{\mathbf{z}\sim p_z(\mathbf{z})}[\log(1 - D(G(\mathbf{z}))]\}. \tag{2.8}$$

The principle idea behind GAN is to ease the requirement of designing an objective function by hand that determines the quality of data generation. DCGAN [99] extends the basic GAN model to scale to high resolution images by including deep convolutional layers into the framework. Wasserstein GAN [100] was proposed to enable the convergence of GAN models to be more stable by incorporating a Wasserstain loss into the learning process. CycleGAN [101] introduces a cycle-consistent loss to ensure that the generated fake data can be mapped back to the real data by an inverse function.

GAN and its variants have achieved great success in image generation [102], unsupervised representation learning [99] and domain adaptation [103]. The adoptaion of GANs for time-series data and localization is far less to be explored. In Chapter 4, our proposed Sequential Invariant Domain Adaptation (SIDA) is based on GANs to learn domain invariant features, and achieve domain adaptation for localization problem, that improves the generalization ability of a learned inertial odometry model in new domains.

2.3.4 Autoencoders

Another important family of generative modelling approaches are autoencoders. Autoencoders have been widely applied in dimensionality reduction and representation learning. An autoencoder model consists of an encoder $\mathbf{h} = f(\mathbf{x})$ that encodes the input $\mathbf{x}$ into a hidden representation $\mathbf{h}$, and a decoder $\mathbf{y} = g(\mathbf{h})$ that reconstructs the output, whilst forcing the input and output to be equal. As the dimension of the hidden state is usually smaller

than that of the input and output, useful data properties i.e. a good representation can be learned during the optimization of $g(f(\mathbf{x})) = \mathbf{x}$. The objective of training an autoencoder is to minimize the following reconstruction loss:

$$L(\mathbf{x}, g(f(\mathbf{x}))), \tag{2.9}$$

where $\mathbf{x}$ is input data, f and g are encoder and decoder function, that can be parameterized by deep neural networks.

Denoising autoencoder (DAE) [104] is one of the most widely applied autoencoders. By adding noise to the input, a denoising autoencoder (DAE) is able to learn more robust representation by minimizing the following reconstruction error:

$$L(\mathbf{x}, g(f(\hat{\mathbf{x}}))), \tag{2.10}$$

where $\mathbf{x}$ is input data, and $\hat{\mathbf{x}}$ is corrupted data with some form of noise. The encoder f and decoder g learn to remove the noise, and hence can discover useful features that capture the intrinsic structure of original data. Variational autoencoder (VAE) [105] is another important family of autoencoders, that introduces variational inference into autoencoder, to approximate the intractable inference distribution. VAE provides a probabilistic tool to represent the latent space of autoencoder.

Autoencoders have been applied to learn compact and useful scene representation for visual SLAM [106]. In Chapter 4, we will investigate the adoption of autoencoders in domain adaptation for time-series sensor data, to keep semantic meaningful representation for inertial tracking.

2.3.5 Attention Mechanism

The attention mechanism was originally conceived to address the problem of sequence-to-sequence (seq2seq) model for machine translation [73]. In the seq2seq model, the encoder maps the input sequence into a context vector, that captures a summary of the input, while the decoder generates a sequence output conditioned on this context vector [22]. The attention model is embedded into the seq2seq model to generate an alignment score matrix for the input and output, allowing the decoder to focus on the important parts of context vector and input sequence. Figure 2.8 shows an illutration of the attention module.

Instead of building correlation between two sequences, the self-attention model learns a score matrix to relate different positions inside a single sequence [74]. Self-attention mechanism has been successfully applied in language generation. [107] applies self-attention for image caption generation, by proposing a soft-attention that generates a matrix of alignment weights to softly reweight context vector, and a hard-attention that uses only some

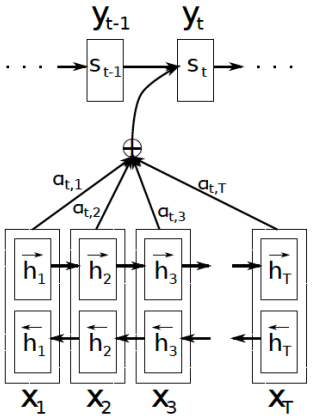


Figure 2.8: An illustration of attention mechanism [73]. $\mathbf{x}_i$ and $\mathbf{y}_i$ are the input and output sequence, $\mathbf{h}_i$ and $\mathbf{s}_i$ is the hidden states of encoder and decoder, α is the attention element that aligns the decoder with the input sequence.

part of context vector for sequence generation. This soft attention is differentiable and can be deterministically learned via backpropagation, while hard attention is non-differentiable and requires to be trained by reinforcement learning. Recently, self-attention was further developed into the Transformer model [74], that allows seq2seq modelling without using a recurrent model. The Transformer framework has achieved the state-of-the-art performance in a wide range of language learning tasks [108].

The attention mechanism has also been extended to solve computer vision problems. A popular example is the non-local neural network [140], which uses non-local style attention to capture the long dependencies of images for video analysis. Few work explored to incorporate attention mechanisms into the learning based localization. Similar to self-attention, our work on selective sensor fusion is capable of selecting useful features from multiple sensor modalities, to improve both accuracy and robustness of deep odometry estimation and localization models in Chapter 5.

2.4 Deep Learning Based Localization Solutions

Recent data-driven approaches to localization have gained a lot of attention. This section overviews existing deep-learning-based localization models, to provide background information about this fast evolving research area, as illustrated in Table 2.1. Note that some of the work listed in Table 2.1 appeared later than our work. A detailed discussion of previous work related to our contributions can be found in each chapter respectively.

PoseNet [81] first tackled camera relocalization from single RGB images with a ConvNet in an end-to-end manner to regress 6-DoF global pose. PoseNet has attracted great attention towards developing more accurate deep learning based relocalization. PoseNet was extended with a learned scale factor to balance camera position and orientation loss [82]. VidLoc [83] leverages LSTM along with ConvNet to exploit the temporal dependencies of consecutive images. VLocNet [113] jointly optimized a relocalization network and visual odometry network, and achieved impressive results. MapNet [84] incorporates geometric constraints between camera pose pairs to learn a general map representation for accurate relocalization. Other work further improves the localization accuracy [110, 133, 134], enables camera relocation to scale to large-scale outdoor environment [109, 141, 124, 132], and enhances the capacity of generalization across different scenes [131].

Previous learning-based work also tackled localization problems by predicting ego-motion from visual observations using deep neural networks instead of applying geometric theory. Deep learning methods are capable of extracting high-level feature representation from large datasets, and hence provide an alternative to solve the visual odometry (VO) problem. They can be mainly categorized into supervised methods, e.g. DeepVO [85] and unsupervised methods, e.g. SfmLearner [86]. In supervised approaches, DeepVO [85] utilizes a combination of convolutional neural network (ConvNet) and recurrent neural network (RNN) to capture the temporary dependencies of consecutive images. Figure 2.9 shows the architecture of this RNN+ConvNet based VO system, which extracts visual features from pairs of images via ConvNet, and passes features through RNNs to model the temporal correlation of features. DeepVO reports impressive results on estimating pose of driving vehicles. This framework has been extended to improve its generalization ability [121] by incorporating curriculum learning (i.e. the model is trained by increasing the data complexity) and geometric loss constraints. Knowledge distilling (i.e. a large model is compressed by teaching a smaller one) is applied into the DeepVO framework to largely reduce the parameter number of network [137]. Recently, there has been growing interest in exploring unsupervised learning framework to jointly learn depth and camera ego-motion from video sequences [86]. This is achieved in an unsupervised manner by utilizing view synthesis as a supervisory signal, showing competitive performance over classical VO systems. Based on this concept, [112, 87, 88, 116, 117, 122, 123, 135, 136] extended this framework to achieve better performance. In these approaches, ConvNets are used as a mapping function that regresses relative pose from pairs of images. However, the global scale of pose estimation is not consistent in these unsupervised approaches, and their performance is not yet competitive with supervised approaches.

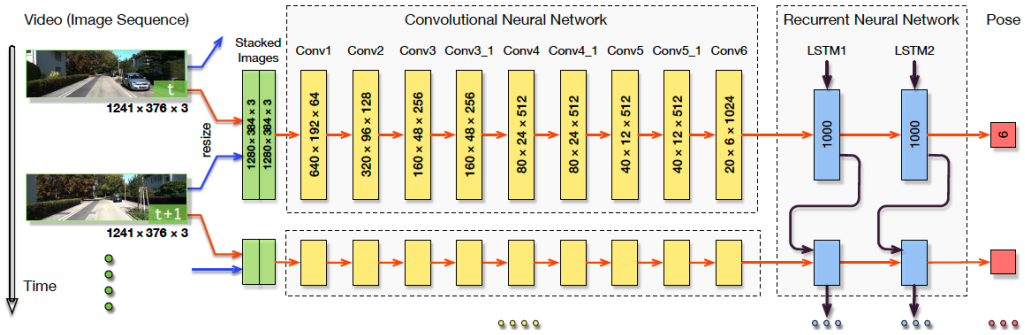


Figure 2.9: An architecture of a RNN+ConvNet based VO system.

Beyond deep visual odometry, VINet [89] processes sequences of images and inertial measurements to construct deep visual-inertial odometry (VIO) with a combination of LSTM and ConvNet models. Furthermore, VIOLearner [120] proposes an online error correction module for training DNN based VIO in an unsupervised manner. In addition, deep learning approach shows capability of learning a compact but dense scene representation that is suitable for keyframe-based monocular SLAM [106]. BA-Net [119] presents a differentiable bundle adjustment module, that imposes geometry constraints to optimize the learned camera pose and scene depth. Besides images, LIDAR point cloud data can also be processed by deep learning to construct deep LIDAR odometry [127, 128].

2.5 Discussion

The existing deep learning approaches show their potential for solving localization problems. However, they are still limited by issues, such as robustness, effective sensor fusion and adaptation into new domains, as discussed in Section 1.2 and 1.3.

- Robustness: A majority of work depends on visual sensors to construct learning-based relocalization [81, 83, 82, 109, 110, 113, 84, 124], visual odometry [85, 86, 112, 87, 88, 116, 117, 121, 122, 123] and visual SLAM [106, 119]. This is because deep learning techniques first powered the research in computer vision community, e.g. ConvNets for image recognition, and it is natural to extend this research to visual localization. Although images can offer rich scene features to aid motion estimation, visual solutions are easily affected by environmental issues, e.g. camera occlusion, weather and lighting conditions, and moving objects. Little work has considered the use of a more robust sensor, i.e. an inertial sensor, which is completely egocentric, and suffers far less from environmental factors. Chapter 3 proposes Inertial

Odometry Neural Network (IONet) to learn self-motion only using an inertial sensor, and reconstruct accurate trajectories without the constraint of sensor attachment, no matter that the sensor is inside a pocket, a bag or on a trolley. To the best of our knowledge, this is the first work that achieves learning-based inertial positioning. After our work, researchers started to work on learning based inertial approaches for pedestrian navigation [115] and car localization [125].

- Effective sensor fusion: Recently, there is growing interest in adopting a combination of multiple sensors to provide more accurate and robust pose, such as depth-vision re-localization [83], vision-inertial odometry [89, 120, 138], and LIDAR-vision odometry [139]. However, they simply concatenate features from multiple modalities, without considering the complementary properties of different sensors and potential sensor degradation. Effective sensor fusion strategies have never been thoroughly studied for deep learning models. How to effectively incorporate features extracted by deep neural network from various sensor modalities and under different dynamics is a central problem for constructing robust localization in practical applications. We propose SelectFusion, a generic sensor fusion strategy to select useful features for odometry estimation and relocalization, which enables models to be more accurate and robust, and offers interpretability in Chapter 4.

- Adaptation into new domains: Previous work normally trained and tested their models in one domain. For example, PoseNet [81] and its variants [83, 110, 84] used 7-Scene dataset as a common evaluation benchmark, that train and evaluate models in the same scenario. The domain shift problem (i.e. the training and testing data are from different data distribution), is always a concern in deep learning, but has far less been explored in the localization context. There is growing need to solve domain adaptation, that will enable DNN based localization models to generalize well in complex and ever-changing real-world environment. Chapter 5 addresses this problem by proposing Sequential Invariant Domain Adaptation (SIDA) that operates on long continuous sensor data (inertial data in our case) to transfer the learned knowledge from domain to another.

In the next sections, we aim to overcome these challenges and develop novel learning methods, that enable the learning-based localization to scale to complex real-world environments.

Table 2.1: Evolving deep learning techniques for localization

Model	Year/Month*	Modality	Application
PoseNet [81]	2015/12	Vision	Relocalization
VINet [89]	2017/02	Vision + Inertial	Visual Inertial Odometry
DeepVO [85]	2017/05	Vision	Visual Odometry
VidLoc [83]	2017/07	Vision	Relocalization
PoseNet+ [82]	2017/07	Vision	Relocalization
SfmLearner [86]	2017/07	Vision	Visual Odometry
Naseer et al. [109]	2017/09	Vision	Relocalization
Walch et al. [110]	2017/10	Vision	Relocalization
IONet (Ours) [111]	2018/02	Inertial Only	Inertial Odometry
UnDeepVO [112]	2018/05	Vision	Visual Odometry
VLocNet [113]	2018/05	Vision	Relocalization, Odometry
MapNet [84]	2018/06	Vision	Relocalization
CodeSLAM [106]	2018/06	Vision	Visual SLAM
GeoNet [87]	2018/06	Vision	Visual Odometry
Zhan et al. [88]	2018/06	Vision	Visual Odometry
Brachmann et al. [114]	2018/06	Vision	Relocalization
RIDI [115]	2018/09	Inertial Only	Inertial Odometry
Yang et al. [116]	2018/09	Stereo Vision	Visual Odometry
Zhao et al. [117]	2018/10	Vision	Visual Odometry
SIDA (Ours) [118]	2019/01	Inertial Only	Domain Adaptation
BA-Net [119]	2019/04	Vision	Bundle Adjustment
VIOLearner [120]	2019/04	Vision+Inertial	Visual Inertial Odometry
Saputra et al. [121]	2019/05	Vision	Visual Odometry
GANVO [122]	2019/05	Vision	Visual Odometry
Li et al. [123]	2019/05	Vision	Visual Odometry
Piasco et al. [124]	2019/05	Vision	Relocalization
Brossard et al. [125]	2019/05	Inertial Only	Inertial Odometry
SelectFusion(Ours) [126]	2019/06	Vision+Inertial+LIDAR	VIO and Sensor Fusion
LO-Net [127]	2019/06	LIDAR	LIDAR Odometry
L3-Net [128]	2019/06	LIDAR	LIDAR Odometry
Xue et al. [129]	2019/06	Vision	Visual Odometry
Wang et al. [130]	2019/06	Vision	Visual Odometry
SANet [131]	2019/10	Vision	Relocalization
Huang et al. [132]	2019/10	Vision	Relocalization
Xue et al. [133]	2019/10	Vision	Relocalization
CamNet [134]	2019/10	Vision	Relocalization
Li et al. [135]	2019/10	Vision	Visual Odometry
Sheng et al. [136]	2019/10	Vision	Visual Odometry
Saputra et al. [137]	2019/10	Vision	Knowledge Distilling, VO
DeepVIO [138]	2019/11	Vision+Inertial	Visual Inertial Odometry
Valente et al. [139]	2019/11	Vision+LIDAR	Sensor Fusion

- Year/Month refers to paper publication date.

Chapter 3

Learning to Localize using Inertial Data

This chapter introduces learning approaches to estimate self-motion and locate users using noisy inertial sensor data. This work first proposes Inertial Odometry Neural Network (IONet), a novel deep neural network (DNN) framework to learn motion transformation from raw IMU data [111, 142]. To the best of our knowledge, this is the first technique for learning to localize using inertial data only. This inertial solution shows good performance in locating people and wheeled objects without being influenced by environmental factors. IONet was further refined as L-IONet, a lightweight IONet model, which greatly reduces the computation burden for training and testing, and also memory, enabling real-time inference on low-end devices [143]. For a first trial in this direction, a very large inertial motion dataset containing 158 sequences totalling 42 km in distance was collected to train and evaluate the framework, with different device attachments, motion modes, users, consumer devices, and large-scale localization experiments. We released this large scale dataset as Oxford Inertial Odometry Dataset (OxIOD)[1], to both boost the adoption of data-driven methods and provide a common benchmark for the task of pedestrian inertial navigation.

3.1 Introduction

Although global positioning system (GPS) is an adequate solution to most outdoor positioning problems, satellite signals are blocked or suffer from serious attenuation and multipath effects in and around buildings, and therefore cannot be used for indoor positioning [5]. There is an emerging need to provide ubiquitous location information in GPS-denied areas for applications such as health/activity monitoring, smart retail, public places navigation, human-robot interaction, and augmented/virtual reality. One of the most promising

[1] Our dataset can be found at http://deepio.cs.ox.ac.uk/

approaches is to perform dead reckoning using inertial sensor data. These methods have attracted great attention from both academia and industry [144], thanks to their mobility and flexibility. Unlike other commonly used sensor modalities, such as GPS, radio or vision, inertial measurements are entirely egocentric and independent of both pre-deployed infrastructure as well as factors such as line-of-sight and visibility.

Recent advances of MEMS (Micro-electro-mechanical systems) sensors enable inertial measurement unit (IMU) small and cheap enough to be deployed on mobile devices. However, the low-cost inertial sensors on smartphones are plagued by high sensor noise, leading to unbounded system drifts. Based on Newtonian mechanics, traditional strapdown inertial navigation systems (SINS) integrate IMU measurements directly. They are hard to realize on accuracy-limited IMU due to exponential error propagation through integration. To address these problems, step-based pedestrian dead reckoning (PDR) has been proposed. This approach estimates trajectories by detecting steps, estimating step length and heading, and updating locations per step [145]. Instead of double integrating accelerations into locations, a step length update mitigates exponential increasing drifts into linear increasing drifts. However, dynamic step estimation is heavily influenced by sensor noise, user's walking habits and phone attachment changes, causing unavoidable errors to the entire system [48]. In some scenarios, no steps can be detected, for example, if a phone is placed on a baby stroller or shopping trolley, the assumption of periodicity, exploited by step-based PDR would break down. Therefore, the intrinsic problems of SINS and PDR prevent widespread use of inertial localization in daily life.

The emerging deep neural networks have proved their impressive performance in solving machine learning tasks, mainly in the fields of images, audio and speech processing [20]. When applied in tracking and localization, recent deep approaches [89, 85] demonstrate that deep neural networks (DNNs) are capable of extracting high-level motion representations from raw data, while providing the state-of-art results compared with traditional model based techniques in terms of accuracy and robustness. But few prior research has exploited raw sequential measurements from low-cost noisy inertial sensors to learn deep tracking.

To constrain the error drifts of the inertial system, we present a general framework - **I**nertial **O**dometry Neural **Net**work (IONet) that learns motion transformation and reconstructs accurate trajectories from raw inertial measurements. Instead of directly integrating inertial data into system states, we propose to break the cycle of continuous error propagation, and reformulate inertial tracking as a sequential learning problem. Our proposed model is able to predict motion transformation and provide a trajectory for indoor users from raw data without the need of any hand-engineering. In addition, it outputs uncertainty

along with motion predictions, representing the belief in model outputs. Furthermore, IONet is further developed as **L**ightweight **I**nertial **O**dometry Neural **Net**work (L-IONet), a novel lightweight framework to allow more efficient inference on low-end devices.

This chapter contributes as follows:

- We cast the inertial tracking problem as a sequential learning problem.
- We propose IONet, the first deep neural network (DNN) framework that learns location transforms in polar coordinates from raw IMU data, and constructs inertial tracking regardless of IMU attachment.
- Our framework is capable of estimating uncertainties along with the pose prediction, providing a metric for the model prediction confidence.
- We collected a large dataset for training and evaluation, and conducted extensive experiments across different attachments, motion modes, users/devices and new environment, whose results outperform traditional SINS and PDR mechanisms.
- We demonstrate that our model can generalize to a more general motion without regular periodicity, e.g. trolley or other wheeled configurations, and work in highly dynamic motion patterns, e.g. human running and mixed velocity motion.
- IONet is further developed as L-IONet, a lightweight deep neural network framework to efficiently infer inertial odometry on low-end devices.

3.2 Related Work

Strapdown inertial navigation systems (SINS) have been studied for decades [42]. Previous inertial systems heavily relied on expensive, heavy, high-precision inertial measurement units, hence their main application had to be constrained on moving vehicles, such as automobiles, ships, aircraft, submarines and spacecraft. Recent advances of MEMS technology enable low-cost MEMS IMU to be deployed on robotics, UAV, and mobile devices [144]. However, restricted by size and cost, the accuracy of a MEMS IMU is extremely limited, and has to be integrated with other sensors, such as visual inertial odometry [14]. Another solution is to attach an IMU on user's foot in order to take advantage of heel strikes for zero-velocity update to compensate system error drifts [47]. These inconveniences prevent inertial solution on consumer grade devices [5].

Unlike SINS's open-loop integration of inertial sensors, PDR uses inertial measurements to detect steps, estimate stride length and heading via an empirical formula [5]. System errors still quickly accumulate, because of incorrect step displacement segmentation and inaccurate stride estimation. In addition, a large number of parameters have to be carefully tuned according to a user's walking habits. Recent research mainly focused on fusing PDR with external references, such as a floor plan [50], geomagnetic fields [52], or electromagnetic fields [146], still leaving the fundamental problem of rapid system drift unsolved. Compared with prior work, we abandon step-based approach and present a new general framework for inertial odometry. This allows us to handle more general tracking problems, including trolley/wheeled configurations, which step-based PDR cannot address.

Previous deep learning based work has tackled localization problems, by employing visual odometry [86, 85, 83] and visual inertial odometry [89]. Data-driven methods have been explored to track human motion and gesture using inertial sensor [147, 148, 149, 150]. But these approaches either rely on hand-designed features or enhance the existing models with learned parameters, to analyse human motion rather than locate users. Compared with them, our model is able to achieve accurate long-term localization in large-scale environments through deep learning. To the best of our knowledge, our proposed IONet is the first deep neural network framework to achieve inertial odometry using inertial data only.

3.3 Problem Formulation

This section introduces the background of inertial navigation mechanisms, and a derivation of sequence based physical model. We discuss the limitations of this model-based method, and show how the sequence based formulation paves the way to our proposed learning based approach.

3.3.1 The Principles Of Inertial Navigation

The principles of inertial navigation are based on Newtonian mechanics. They allow tracking the position and orientation of an object in a navigation frame given an initial pose and measurements from accelerometers and gyroscopes.

Figure 2.2 illustrates the basic mechanism of inertial navigation algorithms. The three-axis gyroscope measures angular velocities of the body frame with respect to the navigation frame, which are integrated into pose attitudes in Equations (3.1-3.2). To represent the orientation, the direction cosine $\mathbf{C}_b^n$ matrix is used to represent the transformation from

the body (b) frame to the navigation (n) frame, and is updated with a relative rotation matrix $\mathbf{\Omega}(t)$. The 3-axis accelerometer measures proper acceleration vectors in the body frame, which are first transformed to the navigation frame and then integrated into velocity, discarding the contribution of the gravity force $\mathbf{g}$ in Equation (3.4). The locations are updated by integrating velocity in Equation (3.5). Equations (3.1-3.5) describe the attitude, velocity and location updates at any time stamp t [42, 9]. In our application scenarios, the effects of earth rotation and the Coriolis accelerations are ignored.

The Attitude Update is given by

$$\mathbf{C}_b^n(t) = \mathbf{C}_b^n(t-1) * \mathbf{\Omega}(t) \tag{3.1}$$

$$\mathbf{\Omega}(t) = \mathbf{C}_{b_t}^{b_{t-1}} = \boldsymbol{I} + \frac{\sin(r)}{r}[\boldsymbol{r}\times] + \frac{1-\cos(r)}{r^2}[\boldsymbol{r}\times]^2, \tag{3.2}$$

$$\boldsymbol{r} = \boldsymbol{\omega}(t)dt \tag{3.3}$$

the Velocity Update is given by

$$\mathbf{v}(t) = \mathbf{v}(t-1) + ((\mathbf{C}_b^n(t-1)) * \boldsymbol{a}(t) - \mathbf{g}_n)dt, \tag{3.4}$$

and the Location Update is given by

$$\mathbf{L}(t) = \mathbf{L}(t-1) + \mathbf{v}(t-1)dt, \tag{3.5}$$

where $\boldsymbol{a}$ and $\boldsymbol{\omega}$ are accelerations and angular velocities in body frame measured by IMU, $\mathbf{v}$ and $\mathbf{L}$ are velocities and locations in navigation frame, r is the norm of $\boldsymbol{r}$, and $\mathbf{g}$ is gravity.

Under ideal condition, SINS sensors and algorithms can estimate system states for all future times. High-precision INS in military applications (aviation and marine/submarine) uses highly accurate and costly sensors to keep measurement errors very small. They also require a time-consuming system initialization including sensor calibration and orientation initialization. However, these requirements are inappropriate for pedestrian tracking. Realizing a SINS mechanism on low-cost MEMS IMU platform suffer from the following two problems

- The measurements from IMUs embedded in consumer phones are corrupted with various error sources, such as scale factor, axis misalignment, thermo-mechanical white noise and random walking noise [43]. From attitude update to location update, the INS algorithm sees a triple integration from raw data to locations. Even a tiny noise will be highly exaggerated through this open-loop integration, causing the whole system to collapse within seconds.

- A time-consuming initialization process is not suitable for everyday usage, especially for orientation initialization. Even small orientation errors lead to the incorrect projection of the gravity vector. For example, a 1 degree attitude error will cause an additional 0.1712 m/s^2 acceleration on the horizontal plane, leading to 1.7 m/s velocity error and 8.56 m location error within 10 seconds.

3.3.2 Sequence-based Physical Model

To address the problems of error propagation, our insight is to break the cycle of continuous integration, and segment inertial data into independent windows. This is analogous to resetting an integrator to prevent windup in classical control theory [151].

However, windowed inertial data is not independent, as Equations (3.1-3.5) clearly demonstrate. This is because key states (namely attitude, velocity and location) are *hidden* - they have to be derived from previous system states and inertial measurements, and propagated across time. Unfortunately, errors are also propagated across time, cursing inertial odometry. It is clearly impossible for windows to be truly independent. However, we can aim for pseudo-independence, where we estimate the *change* in navigation state over each window. Our problem then becomes how to constrain or estimate these latent system states over a window. Following this idea, we derive a sequence-based physical model from basic Newtonian Laws of Motion, and reformulate it into a learning model.

The unobservable or latent system states of an inertial system consist of orientation $\mathbf{C}_b^n$, velocity $\mathbf{v}$ and position $\mathbf{L}$. In a traditional model, the transformation of system states could be expressed as a transfer function/state space model between two time frames in Equation (3.6), and the system states are directly coupled with each other

$$[\mathbf{C}_b^n \quad \mathbf{v} \quad \mathbf{L}]_t = f([\mathbf{C}_b^n \quad \mathbf{v} \quad \mathbf{L}]_{t-1}, [\boldsymbol{a} \quad \boldsymbol{\omega}]_t). \tag{3.6}$$

We first consider displacement. To separate the displacement of a window from the prior window, we compute the change in displacement $\Delta\mathbf{L}$ over an independent window of n time samples, which is simply

$$\Delta\mathbf{L} = \sum_{t=0}^{n} \mathbf{v}(t) \cdot dt. \tag{3.7}$$

We can separate this out into a contribution from the initial velocity state, and the accelerations in the navigation frame

$$\Delta\mathbf{L} = n\mathbf{v}(0)dt + [(n-1)\mathbf{s}_1 + (n-2)\mathbf{s}_2 + \cdots + \mathbf{s}_{n-1}]dt^2 \tag{3.8}$$

where

$$\mathbf{s}(t) = \mathbf{C}_b^n(t-1)\boldsymbol{a}(t) - \mathbf{g} \tag{3.9}$$

is the acceleration in the navigation frame, comprising a dynamic part and a constant part due to gravity.

Then, Equation (3.8) is further formulated as

$$\begin{aligned}\Delta\mathbf{L} = n\mathbf{v}(0)dt + [(n-1)\mathbf{C}_b^n(0) * \boldsymbol{a}_1 + (n-2)\mathbf{C}_b^n(0)\boldsymbol{\Omega}(1) \\ * \boldsymbol{a}_2 + \cdots + \mathbf{C}_b^n(0)\prod_{i=1}^{n-2}\boldsymbol{\Omega}(i) * \boldsymbol{a}_{n-1}]dt^2 \\ - \frac{n(n-1)}{2}\mathbf{g}dt^2\end{aligned} \tag{3.10}$$

and simplified to become

$$\Delta\mathbf{L} = n\mathbf{v}(0)dt + \mathbf{C}_b^n(0)\mathbf{T}dt^2 - \frac{n(n-1)}{2}\mathbf{g}dt^2 \tag{3.11}$$

where

$$\mathbf{T} = (n-1)\boldsymbol{a}_1 + (n-2)\boldsymbol{\Omega}(1)\boldsymbol{a}_2 + \cdots + \prod_{i=1}^{n-2}\boldsymbol{\Omega}(i)\boldsymbol{a}_{n-1}. \tag{3.12}$$

In our work, we consider the problem of indoor positioning i.e. tracking objects and people on a horizontal plane. This introduces a key observation: in the navigation frame, there is no long-term change in height[2]. The mean displacement in the z axis over a window is assumed to be zero and thus can be removed from the formulation. We can compute the absolute change in distance over a window as the L-2 norm i.e. $\Delta l = \|\Delta\mathbf{L}\|_2$, effectively decoupling the distance traveled from the orientation (e.g. heading angle) traveled, leading to

$$\begin{aligned}\Delta l &= \|n\mathbf{v}(0)dt + \mathbf{C}_b^n(0)\mathbf{T}dt^2 - \frac{n(n-1)}{2}\mathbf{g}dt^2\|_2 \\ &= \|\mathbf{C}_b^n(0)(n\mathbf{v}^b(0)dt + \mathbf{T}dt^2 - \frac{n(n-1)}{2}\mathbf{g}_0^b dt^2)\|_2.\end{aligned} \tag{3.13}$$

Because the rotation matrix $\mathbf{C}_b^n(0)$ is an orthogonal matrix i.e. $\mathbf{C}_b^n(0)^T\mathbf{C}_b^n(0) = \mathbf{I}$, the initial unknown orientation has been successfully removed from, giving us

$$\begin{aligned}\Delta l &= \|\Delta\mathbf{L}\|_2 \\ &= \|n\mathbf{v}^b(0)dt + \mathbf{T}dt^2 - \frac{n(n-1)}{2}\mathbf{g}_0^b dt^2\|_2.\end{aligned} \tag{3.14}$$

[2]This assumption can be relaxed through the use of additional sensor modalities such as a barometer to detect changes in floor level due to stairs or elevator.

Hence, over a window, the horizontal distance traveled can be expressed as a function of the initial velocity, the gravity, and the linear and angular acceleration, all in the body frame

$$\Delta l = f(\mathbf{v}^b(0), \mathbf{g}_0^b, \boldsymbol{a}_{1:n}, \boldsymbol{\omega}_{1:n}). \tag{3.15}$$

To determine the change in the user's heading, we consider that a user's proper accelerations and angular rates $(\boldsymbol{a}_{1:n}, \boldsymbol{\omega}_{1:n})$ are also latent variables of IMU raw measurements $(\hat{\boldsymbol{a}}_{1:n}, \hat{\boldsymbol{\omega}}_{1:n})$, and on the horizontal plane, only the heading attitude is essential in our system. From Equations (3.2-3.3) the change in the heading $\Delta\psi$ is expressed as a function of the raw data sequence. Therefore, we succeed in reformulating traditional model as a polar vector $(\Delta l, \Delta\psi)$ based model, which is only dependent on inertial sensor data, the initial velocity and gravity in the body frame

$$(\Delta l, \Delta\psi) = f_\theta(\mathbf{v}^b(0), \mathbf{g}_0^b, \hat{\boldsymbol{a}}_{1:n}, \hat{\boldsymbol{\omega}}_{1:n}). \tag{3.16}$$

To derive a global location, the starting location $\mathbf{L} = (L_0^x, L_0^y)$ and heading ψ_0 and the Cartesian projection of a number of windows can be written as

$$\begin{cases} L_n^x = L_0^x + \Delta l \cos(\psi_0 + \Delta\psi) \\ L_n^y = L_0^y + \Delta l \sin(\psi_0 + \Delta\psi). \end{cases} \tag{3.17}$$

Our task now becomes how to implicitly estimate this initial velocity and the gravity in body frame, by casting each window as an estimation problem.

This section serves as an overview showing the transition from the traditional model-based method to the proposed neural-network-based method. It takes the traditional state-space-model described in Equations (3.1-3.5), which converts raw data to poses in a step-by-step manner, to a formulation where a window of raw inertial data is processed in a batch to estimate a displacement and an angle change. In both formulations, the final output depends on the initial attitude and velocity. As a result, in both cases, the curse of error accumulation will not be avoided if using the model-based integration approach. Note that the explicit sequence based physical model we derived in this section is not used in our proposed learning based inertial odometry model, but it indicates that it is not possible to calculate polar vector (the change of location and heading) by hand via Equation (3.16) without the initial velocity and orientation. However, our sequence based formulation paves the way to our proposed neural network approach. We will show how to formulate this as a learning problem in next section, by learning polar vectors from independent windows of inertial data, that avoids providing the initial states of sequence.

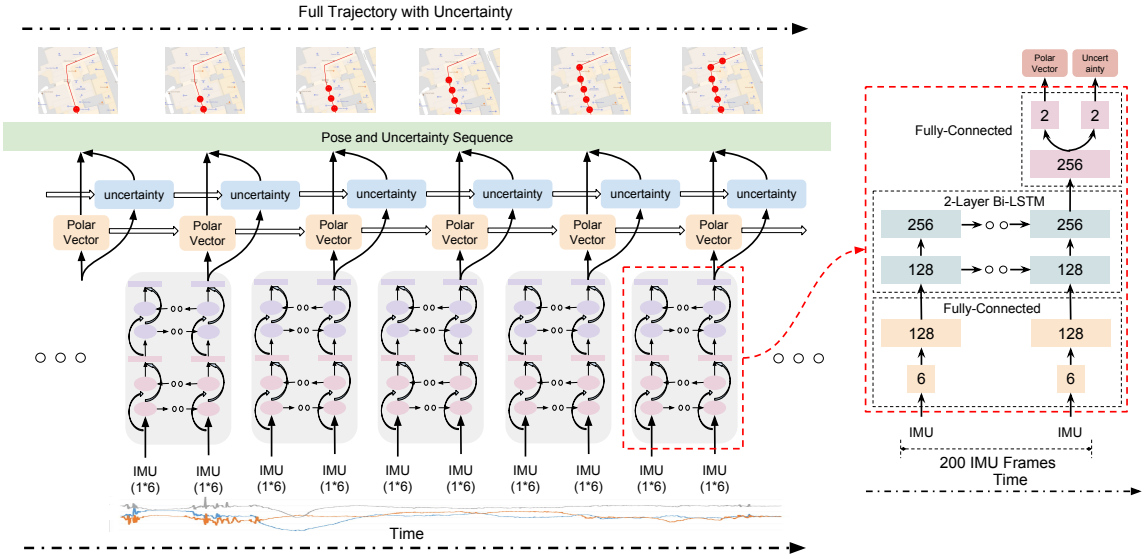


Figure 3.1: Overview of IONet framework. Inertial measurements are segmented into independent windows. A 2-layer bi-LSTM is used to estimate the change in heading and displacement (polar vector) as well as the estimation uncertainties.

3.4 Inertial Odometry Neural Networks

Estimating the initial velocity and orientation in the body frame explicitly using traditional techniques is an extremely challenging problem. Rather than trying to determine the two terms, we instead treat Equation (3.16) as an estimation problem, where the inputs are the observed sensor data and the output is the polar vector. The unobservable terms simply become latent states of the estimation. Intuitively, the motivation for this relies on the regular and constrained nature of pedestrian motion. Over a window, which could be a few seconds long, a person walking at a certain rate induces a roughly sinusoidal acceleration pattern. The frequency of this sinusoid relates to the walking speed. In addition, biomechanical measurements of human motion show that as people walk faster, their strides lengthen [152]. Similarly, vehicle speed can also be estimated using only raw IMU measurements. This was demonstrated in [153], which used an accelerometer to track the vibrations of the vehicle chassis. The idea relies on the fact that the vibrations have a fundamental frequency that is proportional to the vehicle speed. Moreover, the gravity in body frame is related to the initial yaw and roll angle, determined by the attachment/placement of the device, which can be estimated from the raw data [154]. Therefore, we propose the use of deep neural networks to learn the relationship between raw acceleration data and the polar delta vector, as illustrated in Figure 3.1.

Input data are independent windows of consecutive IMU measurements, strongly tem-

poral dependent, representing body motion. To recover latent connections between motion characteristics and data features, a deep recurrent neural network (RNN) is capable of exploiting these temporal dependencies by maintaining hidden states over the duration of a window. Note however that latent states are not propagated between windows. Effectively, the neural network acts as a function f_θ that maps sensor measurements to polar displacement over a window

$$(\boldsymbol{a}, \boldsymbol{\omega})_{200\times 6} \xrightarrow{f_\theta} (\Delta l, \Delta\psi)_{1\times 2}, \tag{3.18}$$

where a window length of 200 frames (2 seconds) is used here[3].

In the physical model, orientation transformations impact all subsequent outputs. We adopt a Long Short-Term Memory (LSTM) to handle the exploding and vanishing problems of vanilla RNN, as it has a much better ability to exploit the long-term dependencies [90]. In addition, as both previous and future frames are crucial in updating the current frame a bidirectional architecture is adopted to exploit dynamic context.

Equation (3.16) shows that modeling the final polar vector requires modeling some intermediate latent variables, e.g. initial velocity and gravity. Therefore, to build up higher representation of IMU data, it is reasonable to stack 2-layer LSTMs on top of each other, with the output sequences of the first layer supplying the input sequences of the second layer. The second LSTM outputs one polar vector to represent the transformation relation in the processed sequence. Each layer has 128 hidden nodes. To increase the output data rate of polar vectors and locations, IMU measurements are divided into independent windows of 200 frames (2s) with a stride of 10 frames (0.1s).

The optimal parameter θ^* inside the proposed deep RNN architecture can be recovered by minimizing a loss function on the training dataset $\mathbf{D} = (\mathbf{X}, \mathbf{Y})$ as

$$\theta^* = \arg\min_\theta \mathcal{L}(f_\theta(\mathbf{X}), \mathbf{Y}). \tag{3.19}$$

The loss function is defined as the sum of Euclidean distances between the ground truth $(\Delta l, \Delta\psi)$ and estimated value $(\Delta\tilde{l}, \Delta\tilde{\psi})$

$$\mathcal{L} = \sum \|\Delta l - \Delta\tilde{l}\|_2^2 + \kappa\|\Delta\psi - \Delta\tilde{\psi}\|_2^2 \tag{3.20}$$

where κ is a factor to regulate the weights of location displacement Δl and heading change $\Delta\psi$.

[3]We experimented with a window size of 50, 100, 200 and 400 frames, and selected 200 as an optimal parameter regarding the trade-off between accumulative location error and predicted loss.

3.5 Uncertainty Estimation

In this section, we aim to determine the uncertainty of deep inertial navigation, which represents the confidence in IONet model output. Since deep neural networks are hard to interpret [155], uncertainty estimation allows us to understand to what extent we should trust model predictions [156]. Moreover, quantifying uncertainty is essential in enhancing inertial navigation with sensor fusion and graph SLAM [157]. For example, the inertial sensor can be better integrated with GPS to form a GPS/IMU systems [158], or with a camera to form visual inertial odometry [14], with the aid of uncertainty.

In our work, the uncertainty is first estimated on motion transformation, e.g. polar vector transformation $(\Delta l, \Delta\psi)$ defined in Equation 3.16, and then we will show how to infer the uncertainty of absolute heading attitude and the location. Unlike the values of polar vector, which are learned using supervisory labels provided by high precision optical motion tracking system, the hand-crafted labels for uncertainties are impossible to obtain. This is due to the fact that there is no direct measurement method for real uncertainty. Therefore, we propose to estimate the uncertainty of inertial tracking in an unsupervised manner by extending the framework in Section 3.4 to a Gaussian model.

The uncertainty for the polar vector is assumed to be normally distribution. Our IONet architecture proposed in Section 3.4 is trained by minimizing the mean square loss between the predicted polar vector $f_\theta(\boldsymbol{x})$ and its corresponding labels $\boldsymbol{y}$. Their outputs can be regarded as the mean of conditional distribution: $\mathcal{N}(f_\theta(\boldsymbol{x}), \boldsymbol{\sigma}^2)$. The likelihood of predicting the real motion transformation is defined as a Gaussian distribution with the model prediction and its variance $\boldsymbol{\sigma}^2$ [159]:

$$\begin{aligned} p(\boldsymbol{y}|f_\theta(\boldsymbol{x})) &= \mathcal{N}(f_\theta(\boldsymbol{x}), \boldsymbol{\sigma}^2) \\ &= \frac{1}{\sqrt{2\pi\boldsymbol{\sigma}^2}} \exp\left(- \frac{(\boldsymbol{y} - f_\theta(\boldsymbol{x}))^2}{2\boldsymbol{\sigma}^2} \right). \end{aligned} \tag{3.21}$$

We alter the deep neural network framework to predict both the motion transformation and its variance. The variance $\boldsymbol{\sigma}$ represents the probabilistic distribution over the model output, termed Aleatoric uncertainty [160]. The aim is to optimize the neural network weights θ by performing a MAP (maximum a posteriori probability) inference. This is equivalent to finding an optimal value θ^* for the model parameters

$$\begin{aligned} \theta^* &= \arg\max_\theta p(\boldsymbol{y}|f_\theta(\boldsymbol{x})) \\ &= \arg\min_\theta -\log p(\boldsymbol{y}|f_\theta(\boldsymbol{x})) \\ &= \arg\min_\theta \frac{1}{2\boldsymbol{\sigma}^2}\|\boldsymbol{y} - f_\theta(\boldsymbol{x})\|^2 + \frac{1}{2}\log\boldsymbol{\sigma}^2. \end{aligned} \tag{3.22}$$

The minimizing objective of loss is defined as

$$\mathcal{L} = \frac{1}{2}\boldsymbol{\sigma}^{-2}\|\boldsymbol{y} - \tilde{\boldsymbol{y}}\|^2 + \frac{1}{2}\log\boldsymbol{\sigma}^2 \tag{3.23}$$

where $\tilde{\boldsymbol{y}}$ and $\boldsymbol{\sigma}$ are the model predicted mean and variance. In practice, our framework is designed to predict the log variance $\boldsymbol{s}_i = \log\boldsymbol{\sigma}^2$, considering that regressing $\boldsymbol{s}_i$ is more stable than directly predicting $\boldsymbol{\sigma}^2$ [160]

$$\mathcal{L} = \frac{1}{2}\exp(-\boldsymbol{s}_i)\|\boldsymbol{y} - \tilde{\boldsymbol{y}}\|^2 + \frac{1}{2}\boldsymbol{s}_i. \tag{3.24}$$

Based on the propagation rules of uncertainty, the uncertainty of absolute heading attitude and the location can be inferred. As the absolute heading is the accumulation of the heading displacement, its variance $\boldsymbol{\sigma}^2_{\psi_t}$ at t time step is the sum of the previous variance $\boldsymbol{\sigma}^2_{\psi_{t-1}}$ and the variance of delta heading $\boldsymbol{\sigma}^2_{\Delta\psi_t}$, estimated by our deep neural network i.e.

$$\boldsymbol{\sigma}^2_{\psi_t} = \boldsymbol{\sigma}^2_{\psi_{t-1}} + \boldsymbol{\sigma}^2_{\Delta\psi_t}. \tag{3.25}$$

From Equation (3.17), both variances of the delta location and the absolute heading are considered in order to infer the variance of delta location in the navigation frame - East (x) and North (y) axes:

$$\begin{cases} \boldsymbol{\sigma}^2_{\Delta L^x} = \cos(\psi)^2 * \sigma^2_{\Delta l} + \Delta l^2[\sin(\psi) * \sigma_\psi]^2 \\ \boldsymbol{\sigma}^2_{\Delta L^y} = \sin(\psi)^2 * \sigma^2_{\Delta l} + \Delta l^2[\cos(\psi) * \sigma_\psi]^2 \end{cases}. \tag{3.26}$$

Similar to the uncertainty of the absolute heading, the uncertainty of the absolute location is the accumulation of the variance of the location change

$$\boldsymbol{\sigma}^2_{L_t} = \boldsymbol{\sigma}^2_{L_{t-1}} + \boldsymbol{\sigma}^2_{\Delta L_t}. \tag{3.27}$$

3.6 Lightweight Inertial Odometry Neural Networks

In this section, we introduce the **L**ightweight **I**nertial **O**dometry Neural **Net**work (L-IONet), a lightweight framework to learn inertial tracking. L-IONet is more efficient in resource and computational consumption than the previous LSTM-based IONet approach. Figure 3.2 compares the structures of three inertial navigation models, i.e. classical PDR, IONet, and L-IONet.

Although deep learning solutions demonstrate great potential to solve sensing problems, e.g. IONet in inertial navigation, their huge computational and memory requirement

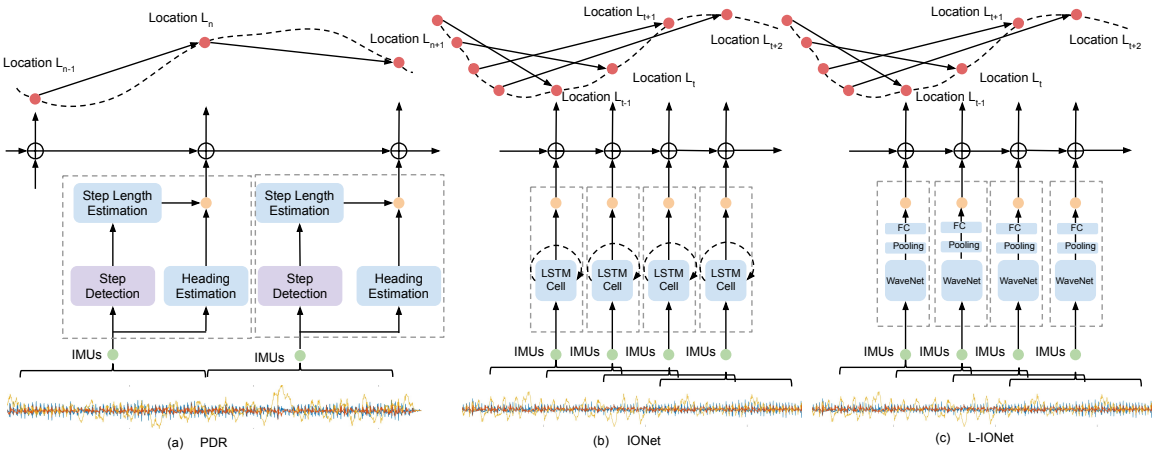


Figure 3.2: The framework illustration of three inertial navigation models: (a) Pedestrian Dead Reckoning (b) Inertial Odometry Neural Network (IONet) (c) Lightweight Inertial Odometry Neural Network (L-IONet).

restricts the deployment of DNN models onto low-end devices [161]. Compared with deploying machine learning models on the cloud, computation at the edge reduces bandwidth usage, cloud workload and latency. In addition, it can help to protect a users' privacy, as all raw sensor data remain on the user's device rather than uploading to the cloud [162]. Therefore, it is necessary to design an efficient and fast DNN model to enable the inference of IONet on low-end devices, e.g. mobile phone, smartwatch, Raspberry Pi.

The main bottleneck of the IONet framework is the LSTM module. This is because recurrent models update the hidden states through very complex nonlinear operators, and condition on all the past history of inputs. When training and inferring on the entire IONet model, the LSTM model is not easy to optimise. It takes long training time and huge resources to converge, especially when working above long sequential data, for example, 200 frames of inertial data in one single sequence in our problem.

We propose to replace the recurrent model with an autoregressive model to produce outputs using the recent frames of a sequence. A good example is WaveNet, a generative causal autoregressive model, widely applied in processing speech and voice signals for synthesis tasks [163]. Inspired by WaveNet, we propose an autoregressive model based L-IONet to process the long continuous signals of inertial sensors to predict polar vectors, which are further connected with previous states to reconstruct trajectories. Because L-IONet has no recurrent module and fewer complex nonlinear operations, this feed-forward model is easier to train in parallel, and much faster at state inference.

The basic module of our proposed framework is the causal dilated convolution layer. It can be viewed as a convolutional neural network (ConvNet) with a sliding window, but

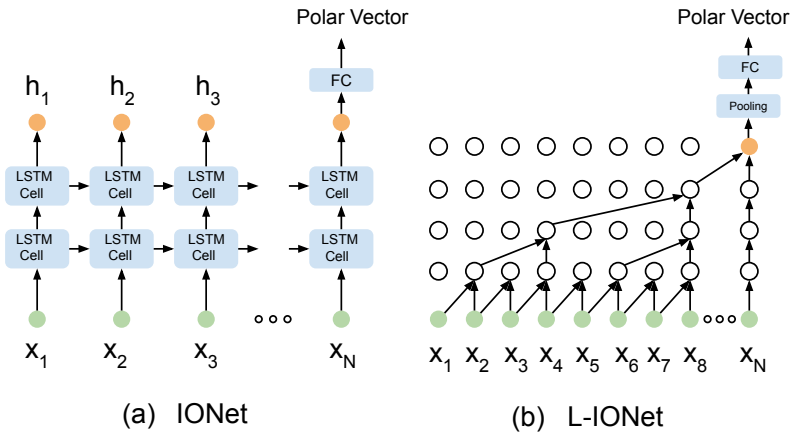


Figure 3.3: A comparison of LSTM-based IONet and WaveNet-based L-IONet

is a specific type of ConvNet that works well on long sequential data. Compared with a regular ConvNet, the causal convolution inside our model is a 1-dimensional filter that convolves on the elements of current and previous timesteps from the preceding layer, to prevent using future states, as shown in Figure 3.3 (b). The stacked layers of dilated convolutions allow the receptive area of convolution operation to be made very large by using the convolution that skips input values with a certain distance. Hence the model is able to capture dependencies over a long sequence of data without requiring a large number of parameters [164].

Each causal dilated convolution performs via a gated activation unit:

$$\mathbf{z} = \tanh(\mathbf{W}_{f,k} * \mathbf{x}) \odot \sigma(\mathbf{W}_{g,k} * \mathbf{x}) \tag{3.28}$$

where $\mathbf{W}$ denotes the weights of the convolutional filters, f and g represent filters and gates, k is the layer index, $*$ is the convolution operator, and $\odot$ is an element-wise multiplication operator. Meanwhile, residual and skip connection modules are adopted to enable a deeper structure, and improve the model's non-linearity in regression.

In our L-IONet framework, several layers of dilated causal convolutions are stacked to increase the receptive areas of the inputs. They skip the inputs with a specified stride, and their dilation doubled for every layer. In our case, an 8-layers model with a dilation of 1, 2, 4, 8, 16, 32, 64, 128 for each layer respectively, is enough to process a sequence of 200 frames of inertial data.

The original WaveNet is designed for audio generation, which quantizes the real data into discrete values, and reconstructs from the quantized data using the softmax function. Instead of learning classification possibility, our framework replaces the softmax function with a pooling layer and a fully-connected layer to map the extracted features $\mathbf{z}$ into the

2-dimensional polar vectors:

$$(\Delta l, \Delta\psi) = \text{FC}(\text{Pool}(\mathbf{z})). \tag{3.29}$$

Similar to IONet model, the predicted polar vectors are further connected with previous system states to produce current locations via Equation 3.17.

In order to increase the output rate of neural network predictions, we present a sliding window method. As Figure 3.2 (b) and (c) demonstrated, the inertial sensor readings are segmented into independent sequences by using a fixed-size sliding window. In our problem, we choose n the window size of the sequence as 200 frames (2 seconds), with a stride for sliding the window as 10. The polar vector is predicted by the deep neural network from each sequence, and connected by a merging module to generate locations, as Equation (3) describes. Note that the current location is updated by the location at the timestep of 200 frames before current timestep rather than the location at the timestep of 10 frames before. With the predictions from the overlapping windows, the output rate is increased to 10 Hz. Low pass filters are further used upon the polar vectors and locations to smooth the predictions for trajectory reconstruction.

3.7 Oxford Inertial Odometry Dataset

This section introduces the Oxford Inertial Odometry Dataset (OxIOD), a data collection of inertial measurements for training and evaluating deep learning based inertial odometry models. To reflect sensor readings under everyday usage, the data were collected with IMUs with various attachments (handheld, in the pocket, in the handbag and on a trolley/stroller), motion modes (halting, walking slowly, walking normally, and running), four types of off-the-shelf consumer phones and five different users, as illustrated in Table 3.2. Our dataset has 158 sequences, and the total walking distance and recording time are 42.5 km, and 14.72 h (53022 seconds). We argue that the existing public datasets are not approapriate for training deep inertial odometry, and our proposed OxIOD dataset can be used to train robust and accurate deep learning models for inertial tracking.

3.7.1 Existing Inertial Navigation Datasets

Table 3.1 shows representative datasets that include inertial data for the purpose of navigation and localisation. In KITTI [165], Oxford RobotCar [24] and EuRoC MAV datasets [166], the sensors are rigidly fixed to the chassis of a car, which is suitable for studying

Table 3.1: Comparison of datasets with IMU and ground truth

Dataset	Year	Environment	Attachment	IMU Type	Sample Rate	Groundtruth	Accuracy	Data Size
KITTI Odometry	2013	Outdoors	Car	OXTS RT3003	10 Hz	GPS/IMU	10 cm	22 seqs, 39.2 km
EuRoC MAV	2016	Indoors	MAV	ADIS 16488	200 Hz	Motion Capture	1 mm	11 seqs, 0.9 km
Oxford RobotCar	2016	Outdoors	Car	NovAte SPAN	50 Hz	GPS/IMU	Unknown	1010.46 km
TUM VI	2018	In/Outdoors	Human	BMI 160	200 Hz	Motion Capture	1 mm	28 seqs, 20 km
ADVIO	2018	In/Outdoors	Human	InvenSense 20600	100 Hz	Other Algorithms	Unknown	23 seqs, 4.5 km
OxIOD (Ours)	2018	Indoors	Human	InvenSense 20600	100 Hz	Motion Capture	0.5 mm	**158 seqs**, **42.587 km**

vehicle movements, but not directly useful for studying human movement. The TUM VI dataset [167] was collected to evaluate visual-inertial odometry (VIO), with a pedestrian holding the device in front of them. The ground truth in TUM VI is provided at the beginning and ending of the sequences, while during most of the trajectories there is no ground truth. Similarly, in ADVIO [168], the dataset is rather short (4.5 km) and only offers pseudo ground truth generated by a handcrafted inertial odometry algorithm.

There are several datasets focusing on human gait and activities, which are somewhat similar to our dataset, but do not concentrate on localisation. Some of these datasets measure human activities, such as USC-HAD [169], CMU-MMAC [170], and OPPORTUNITY [171]. Though these datasets have inertial data with accurate poses as ground truth, they cannot be used to train and test odometry/localisation, since the participants did not move much during the experiments. Some other datasets, such as MAREA [172], focus on human gait recognition and collected inertial data while carriers were walking or running. However, these datasets lack solid ground truth and thus limit their usage in training and testing odometry models.

As we can see from Table 3.1, our OxIOD dataset has a large amount of data from 158 sequences, leading to a total distance of 42.587km. The data size of OxIOD is larger than most other inertial navigation datasets, and hence is suitable for deep neural network methods, which require large amounts of data and high accuracy labels. It should be noted that the total length of the dataset even exceeds some of those collected by vehicles. Meanwhile, our dataset can better represent human motion in everyday conditions and thus has a greater diversity.

3.7.2 Sensor Setup

The data were collected by the on-board sensors of consumer phones, recording accelerations and angular rates from 6-axis IMUs, and magnetic fields from 3-axis magnetometers (only inertial data are used in our experiments). The sensor types of IMUs and magnetometers employed in our adopted mobile phones are listed in Table 3.3. A Vicon motion capture

Table 3.2: Oxford Inertial Odometry Dataset

	Type	Seqs	Time (s)	Distance (km)
Attachments (iPhone 7P/User 1) (Normally Walking)	Handheld	24	8821	7.193
	Pocket	11	5622	4.231
	Handbag	8	4100	3.431
	Trolley	13	4262	2.685
Motions	Slowly Walking	8	4150	2.421
	Normally Walking	-	-	-
	Running	7	3732	4.356
Devices	iPhone 7P	-	-	-
	iPhone 6	9	1592	1.381
	iPhone 5	9	1531	1.217
	Nexus 5	8	4021	2.752
Users	User 1	-	-	-
	User 2	9	2928	2.422
	User 3	7	2100	1.743
	User 4	9	3118	2.812
	User 5	10	2884	2.488
Large Scale	floor 1	10	1579	1.412
	floor 2	16	2582	2.053
Total		158	53022	42.587

system [173] was deployed to produce high-precise groundtruth values of the object motion, i.e. its orientation, velocity and position. The large-scale collection was conducted on two office floors, where we used a Google Tango Tablet [174] as pseudo groundtruth.

Table 3.3: Sensor Setup of OxIOD Dataset

Mobile Phone	IMU	Magnetometer
iPhone 7 Plus	InvenSense ICM-20600	Alps HSCDTD004A
iPhone 6	InvenSense MP67B	AKM 8963
iPhone 5	STL3G4200DH	AKM 8963
Nexus 5	InvenSense MPU-6515	Asahi Kasei AK8963

IMU: The majority of data were collected with an iPhone 7 Plus device. The IMU inside iPhone 7 Plus is InvenSense ICM-20600, a 6-axis motion tracking sensor. It combines a 3-axis gyroscope and a 3-axis accelerometer. 16-bit ADCs are integrated in both gyroscope and accelerometer. The sensitivity error of the gyroscope is 1%, while the noise is $4\text{mdps}/\sqrt{\text{Hz}}$. The accelerometer noise is 100 $\mu g/\sqrt{\text{Hz}}$.

Magnetometer The Alps HSCDTD004A embedded in iPhone 7 Plus is a 3-axis geomagnetic sensor, which is mainly used for electronic compass. It has a measurement range of ± 1.2mT and an output resolution of 0.3 μT/LSB.

Vicon System We deployed 10 Bonita B10 cameras in the Vicon Motion Tracker system [173], encircling an area where we conducted data collection experiments. Each Bonita B10 camera has a frame rate of 250 fps, and resolution of 1 megapixel (1024*1024). The lens operating range of Bonita B10 can be up to 13 m. These features enable us to capture motion data with a precision down to 0.5 mm, making the ground truth very accurate and reliable. The software used in the Vicon system is *Vicon Tracker 2.2*. We connected Vicon Tracker to Robot Operating System (ROS) with *vicon_bridge*, and recorded the data stream with rostopic. The map size of our experimental setup in Vicon Room is $5m \times 8m$.

Time Synchronisation The IMU and magnetometer are integrated in the mobile phone, sharing the same time stamp. Vicon data recorded with ROS is saved with UTC timestamp. Before each experiment, we synchronised the time of iPhone 7 Plus and ROS with UTC, and thus all time stamps recorded along with sensor data will be synchronised.

3.7.3 Data Collection

Attachments The phone based IMUs will experience distinct motions when attached in different places. In the context of pedestrian navigation, a natural use of mobile phone leads to an unconstrained placement of inertial sensors, and therefore we selected four common situations to study, i.e. the device is in the hand, in the pocket, in the handbag or on the trolley. In our data collection, a pedestrian (named *User 1*) walked naturally inside the Vicon room, carrying a phone in four attachments. Figure 3.4 shows in which way the devices were held during the experiments.

Motion Modes Humans move in different motion modes in their everyday activities. We selected and collected data from four typical motion models: halting, walking slowly, walking normally and running. The experiments with different motion modes were performed by *User 1* with iPhone 7Plus in hand, to reduce the influences from user walking habits or sensor properties. The velocities of participants are around 0.5 m/s, 1 m/s, and 1.5 m/s during slow walking, normal walking and running. Our experiments indicate that the user speeding can be directly recovered from raw inertial data via deep neural networks, even under a mixed of motion modes.

Devices and Users Both sensors properties and the walking habits of users throw impacts on the performance of inertial navigation systems. In order to ensure inertial odometry invariant across devices and users, we collected data from several types of devices and different users. Four off-the-shelf smartphone were chosen as experimental devices:

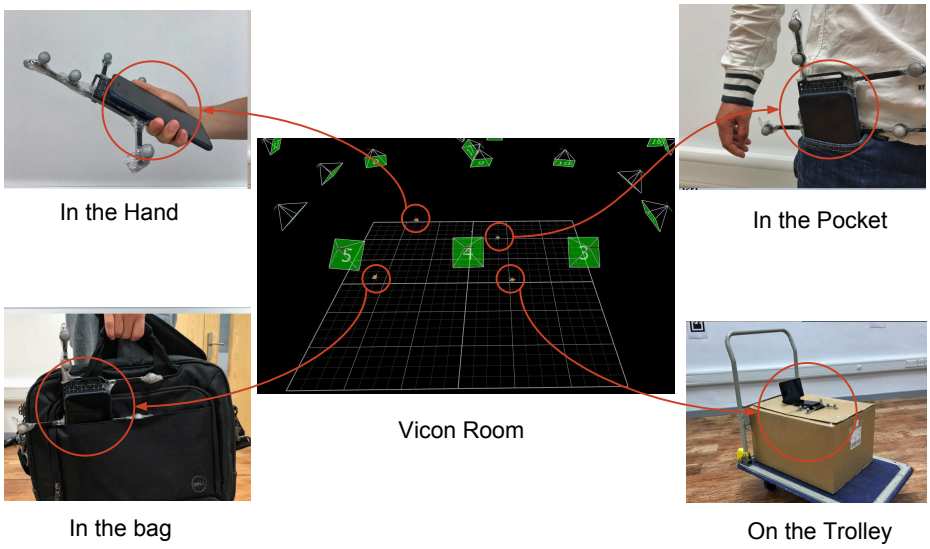


Figure 3.4: Inertial data from OxIOD dataset are collected from a smartphone in four different attachments: handheld (left above), pocket (right above), handbag (left below), trolley (right below). The high-precise motion labels are provided by the Vicon System.

iPhone 7 Plus, iPhone 6, iPhone 5, and Nexus 5, listed in Table 3.3. Five participants were recruited to perform experiments with phone in the hand, pocket and handbag respectively. The mixed data from various devices and users can also be applied in the identification of devices and users.

Large-scale localization Besides the extensive data collection inside the VICON Room, we also conducted large-scale tracking in two environments. Without the help of Vicon system, Google Tango device was chosen to provide pseudo ground truth. Participant was instructed to walk freely in an office building on two separate floors (about 1650 m^2 and 2475 m^2). The smartphones were placed in the hand, pocket and handbag respectively, while the Tango device was attached on the chest of the participant to capture precise trajectories.

3.8 Experiments

In this section, we evaluate our proposed model in terms of accuracy and robustness. We will first describe the training and testing details of the neural network models. Then we evaluate the accuracy of IONet and test its ability to generalize across different users, devices, motion modes, and environments, followed by uncertainty estimation, and further apply our model to trolley tracking. L-IONet is evaluated for pedestrian localization in

Table 3.4: Number of generated subsequences from the OxIOD dataset

Dataset Domain	Training Subsequences	Testing Subsequences
Handheld (Normal)	45544	3812
Handheld (Slow)	36242	3095
Handheld (Run)	31161	3007
Pocket	53631	2385
Handbag	36410	4430
Trolley	29001	2718

different attachments and large-scale environments. Lastly, we provide a systematic study into the model performance on low-end devices.

3.8.1 Training Details

Dataset: There is no public dataset for indoor localization using phone-based IMU. We therefore used our own OxIOD dataset[4] for training and evaluating deep learning models. The training dataset uses the inertial measurements from an IMU sensor on an off-the-shelf iPhone 7 Plus. The smartphone is attached at different positions with the same pedestrian, including hand-held, in pocket and in handbag and on trolley. Note that, this phase of training data collection only needs one pedestrian (User 1) to participate. The raw inertial readings are segmented into sequences with a window size of 200 frames (2 seconds). The number of generated subsequences for training and testing from our OxIOD dataset is given in Table 3.4. In order to test our model's ability to generalize across different users, we invited 3 new participants and made further evaluations on two additional phones: an iPhone 6 and an iPhone 5.

Training: We implemented our model on the Pytorch platform [175], and train the model on a NVIDIA TITAN X GPU. During training, we used Adam, a first-order gradient-based optimizer [176] with a learning rate of 0.0001. On average, the training converges after 100 iterations. To avoid overfitting, we gathered data with abundant moving characteristics inside, and adopted Dropout [177] in each LSTM layer, randomly dropping 25% units from neural networks during training. This method significantly reduces overfitting, and proves to perform well on new users, devices and environments.

Testing: We also found that a separate training on every attachment shows better performance in prediction than training jointly, hence we implemented the prediction model of 2-layer Bi-LSTM trained on separate attachments in our following experiments. In a practical deployment, existing techniques can be adopted to recognize different attachments

[4]Our Dataset can be found at http://deepio.cs.ox.ac.uk

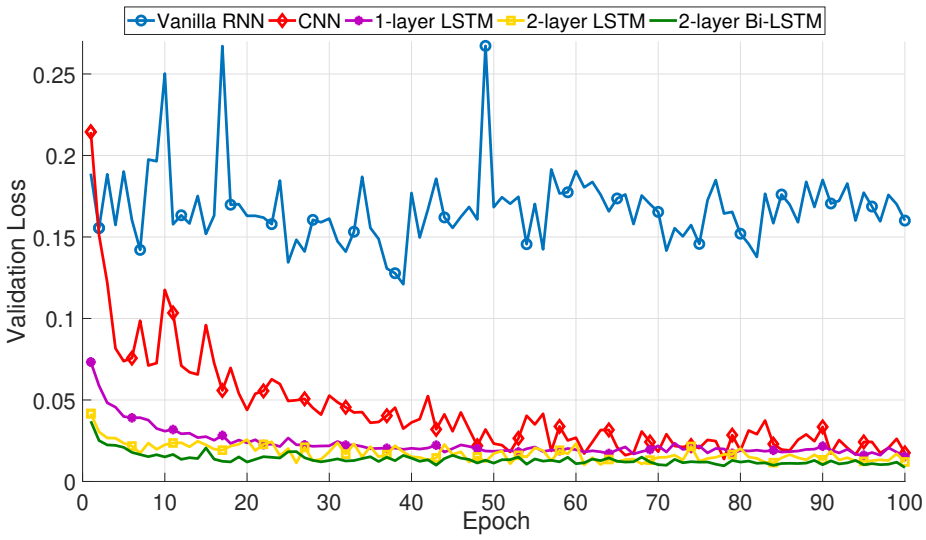


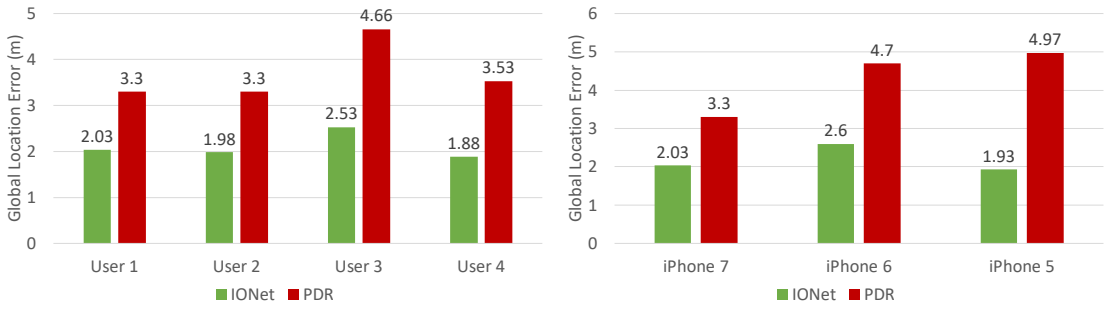
Figure 3.5: The losses of adopting various frameworks demonstrate that our proposed IONet framework with 2-layer Bi-LSTM descends more steeply, and stays lower and more smoothly during the training than all other neural networks.

from pure IMU measurements [48], providing the ability to dynamically switch between trained models.

Baselines: Two traditional methods are selected as baselines, pedestrian dead reckoning (PDR) and strapdown inertial navigation system (SINS) mechanism [42], to compare with our prediction results. PDR algorithms are seldom made open-source, especially a robust PDR used in different attachments, so we implement code ourselves according to [48] for step detection and [49] for heading and step length estimation.

3.8.2 Comparison with Other DNN Frameworks

To evaluate our assumption of adopting a 2-layer Bidirectional LSTM for polar vector regression, we compare its validation results with various other DNN frameworks, including frameworks using vanilla RNN, vanilla Convolutional Neural Network (CNN), 1-layer LSTM and 2-layer LSTM without Bi-direction. The training data are from all attachments. Figure 3.5 shows their validation loss lines. For a fair comparison, the dimension of hidden states is chosen the same for all recurrent neural networks architectures (vanilla RNN, 1-layer LSTM, 2-layer LSTM, and 2-layer Bi-directional LSTM). Our proposed framework with 2-layer Bi-LSTM descends more steeply, and stays lower and more smoothly during the training than all other neural networks, supporting our assumption, while vanilla RNN suffers from vanishing gradient problems, and CNN does not seem to capture the temporal dependencies well.



(a) Multiple Users (b) Multiple Phones

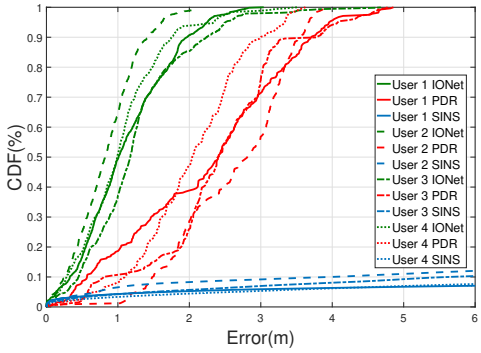
Figure 3.6: The maximum position error of IONet stayed around 2 meters within 90% of the testing time, seeing 30%- 40% improvement compared with traditional PDR in multiple users (a) and multiple phones (b).

3.8.3 Tests Involving Multiple Users and Devices

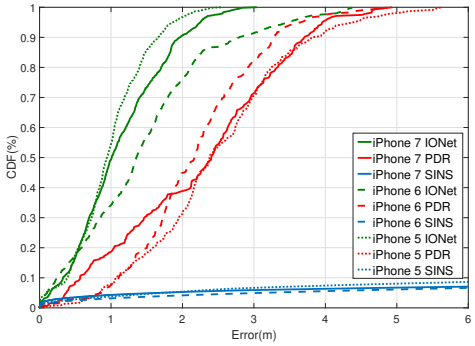
A series of experiments were conducted inside a large room with new users and phones to show our neural network's ability to generalize. The Vicon system provides a highly accurate reference to measure the location errors.

The first group of tests include four participants, walking randomly for two minutes with the phone in different attachments, e.g. in hand, pocket and handbag respectively, covering everyday behaviors. Note that the training dataset is taken from only one of these participants. The performance of our model is measured as cumulative error distribution function (CDF) against Vicon ground truth and compared with conventional PDR and SINS. Figure 3.7a, Figure 3.7c and Figure 3.7e illustrate that our proposed approach outperforms the competing methods in every attachment. If raw data is directly triply integrated by SINS, this results in cubic error growth. The maximum position error of IONet stayed around 2 meters within 90% of the testing time, seeing 30%- 40% improvement compared with traditional PDR in Figure 3.6a.

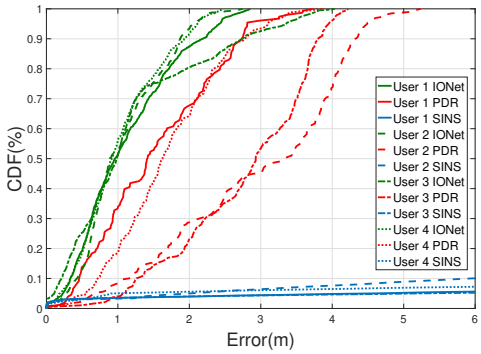
Another group of experiments is to test the performance across different devices, shown in Figure 3.7b, Figure 3.7d and Figure 3.7f. We choose another two very common consumer phones, iPhone 6 and iPhone 5, whose IMU sensors, InvenSense MP67B and ST L3G4200DH, are quite distinct from our training device, iPhone 7 (IMU: InvenSense ICM-20600). Although intrinsic properties of different sensors influence the quality of inertial measurements, our neural network shows good robustness.



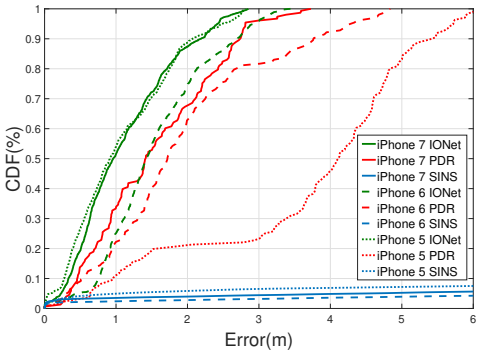
(a) Handheld (multi users)



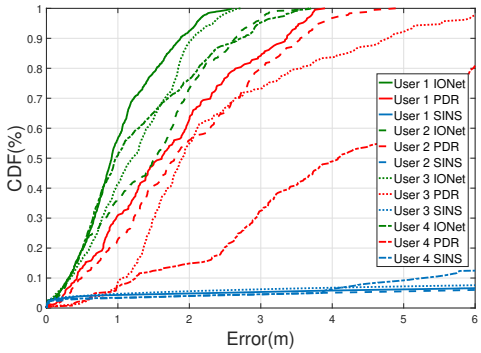
(b) Handheld (multi devices)



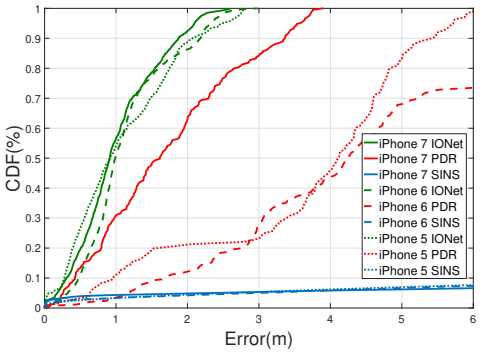
(c) In Pocket (multi users)



(d) In Pocket (multi devices)

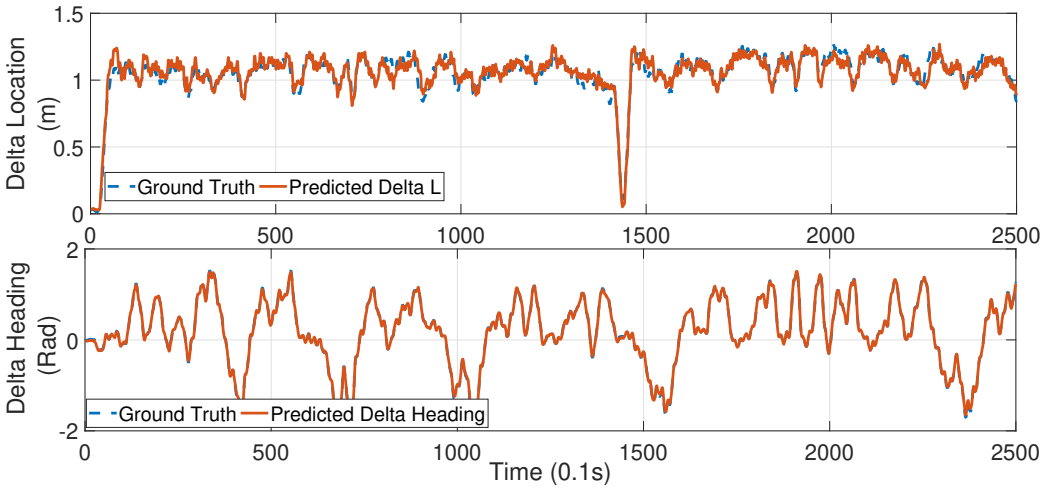


(e) In Handbag (multi users)

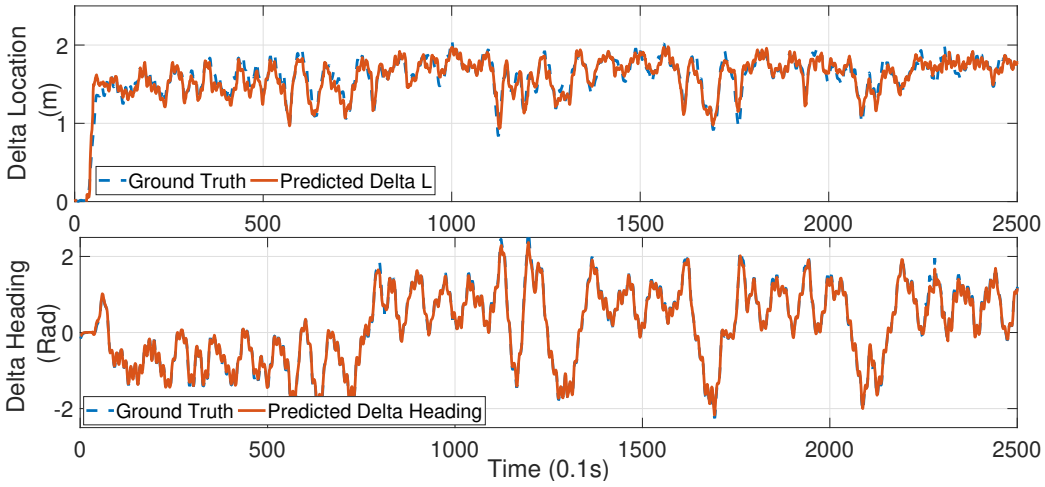


(f) Handbag (multi devices)

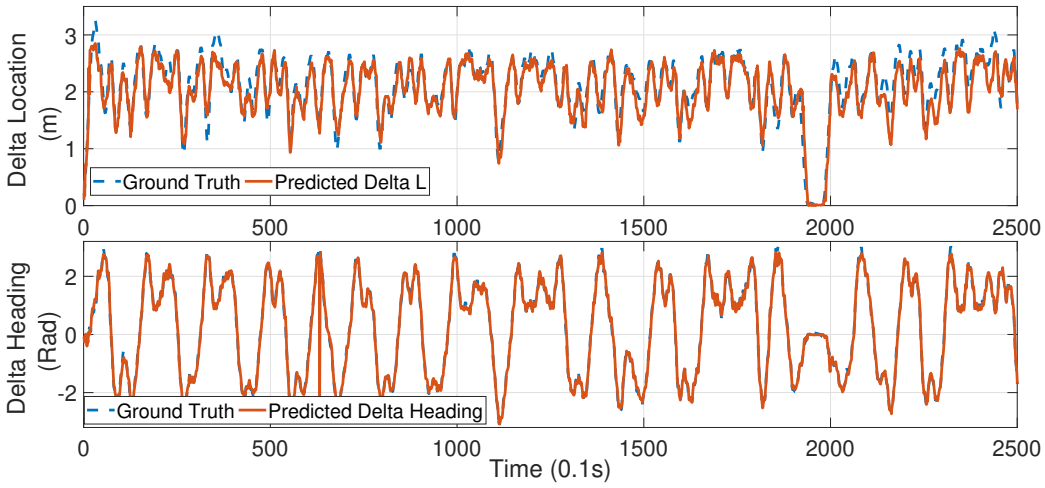
Figure 3.7: The performance of our proposed IONet is compared with SINS and PDR. In the experiments involving multiple users, our learning model trained on User 1 is tested directly on other users without further fine-tuning in three attachments: handheld (a), pocket (c) and bag (e), to show the generalization ability across different devices. In the experiments involving multiple devices, our model trained on iPhone 7 is tested directly on other devices without further fine-tuning in three attachments: handheld (d), pocket (d) and bag (f) to show its generalization ability across different devices.



(a) Walking slowly with handheld phone

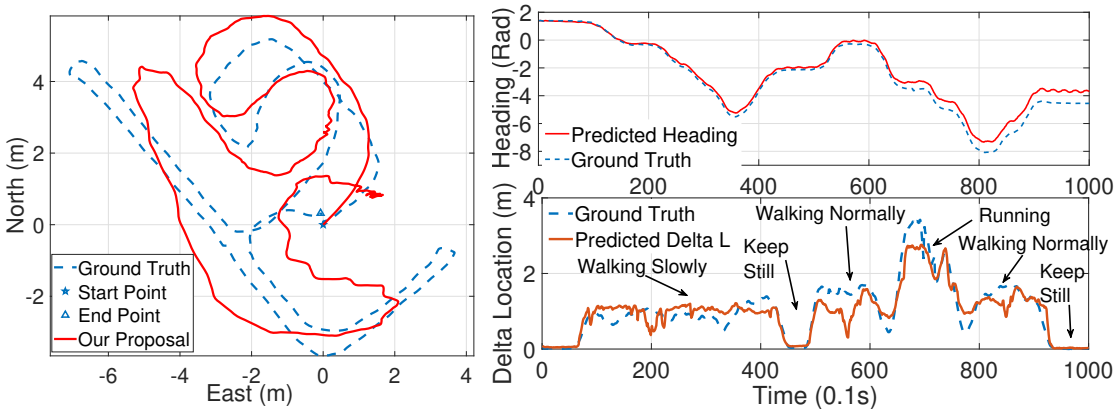


(b) Walking normally with handheld phone



(c) Running with handheld phone

Figure 3.8: The predicted polar vector are very close to the ground truth with handheld phone no matter whether (a) slowly walking (b) normally walking and (c) running, showing the effectiveness of our proposed framework on polar vector regression from raw inertial data.

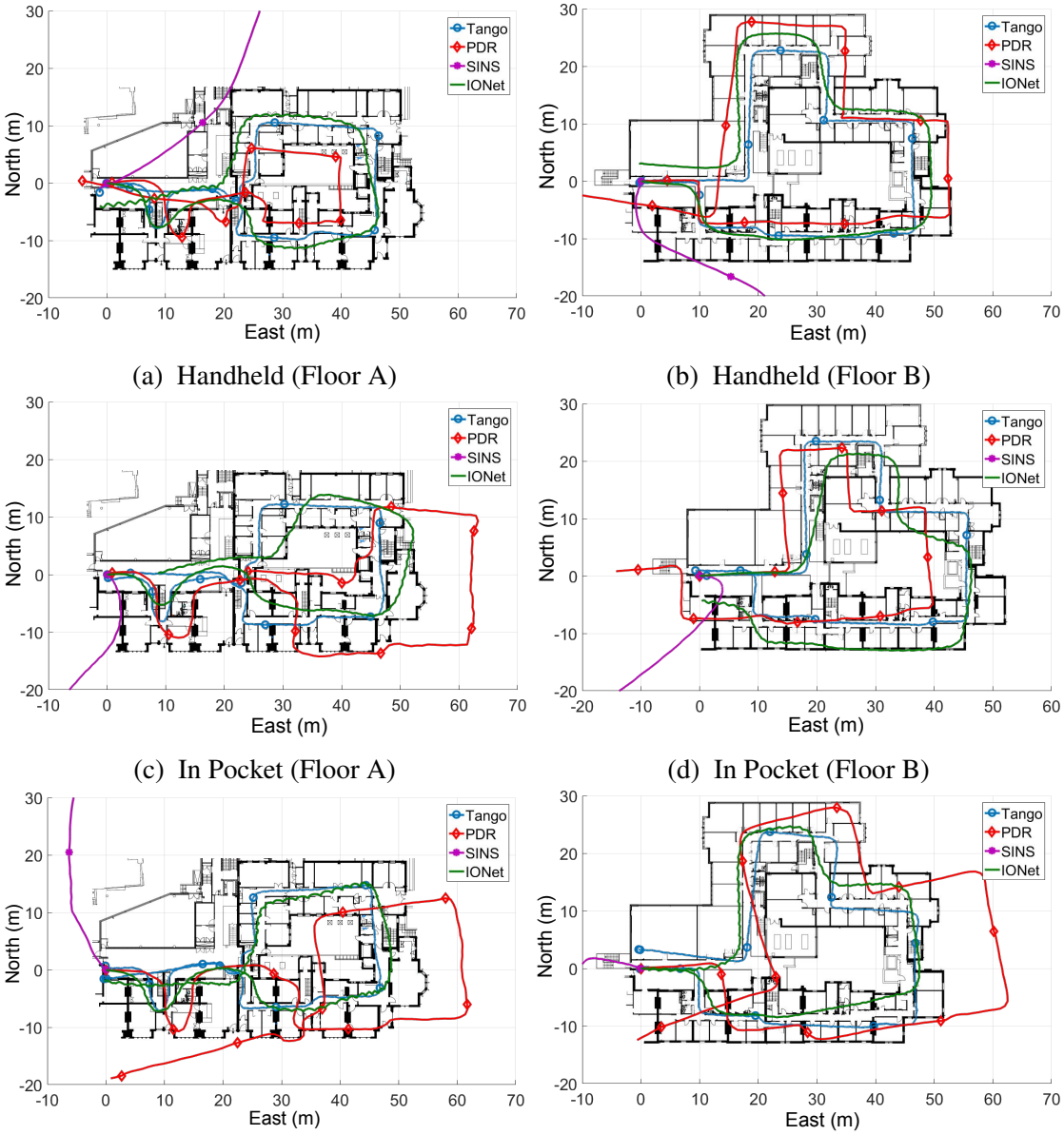


(a) Trajectories (b) Heading and Delta Location

Figure 3.9: The performance of IONet is evaluated on an challenging experiment with varying motion modes including waking slowly, halting, walking normally and running for estimating (a) trajectory and (b) heading and location displacement. IONet can learn an extensive amount of information about the idiosyncratic motion behavior associated with different motion modes.

3.8.4 Tests Involving Multiple Motion Modes

To evaluate the ability of IONet to generalize across different motion modes, several experiments were conducted. Figure 3.8 displays estimated and ground truth polar vectors when training and testing individually with three different motion types: slow walk, normal walk, and running. As can be seen from the figures, both relative distance and heading can be estimated with great accuracy over an extended period for all motion modes. This demonstrates that the neural network can achieve excellent performance over a broad range of motion types. By contrast, Figure 3.9 illustrates the performance on a more varied data set where all of the three previously studied motion modes are used, and where the training data included all of the three motion modes. As seen in Figure 3.9 (b), the heading and distance estimates are of good quality through most of the data set. However, the overall error is higher than when considering the different motion modes separately as in Figure 3.8. Hence, taken together, Figure 3.9 and Figure 3.8 make it possible to conclude that the neural network can learn an extensive amount of information about the idiosyncratic motion behavior associated with different motion modes. For practitioners, this means that motion type classification and individual calibration for different motion types potentially can lead to significant improvements in performance. Despite the fact that quickly varying movement characteristics may be more challenging for IONet, Figure 3.9 (a) still demonstrates good ability to reconstruct the overall motion pattern of a trajectory with mixed motion modes.



(a) Handheld (Floor A)

(b) Handheld (Floor B)

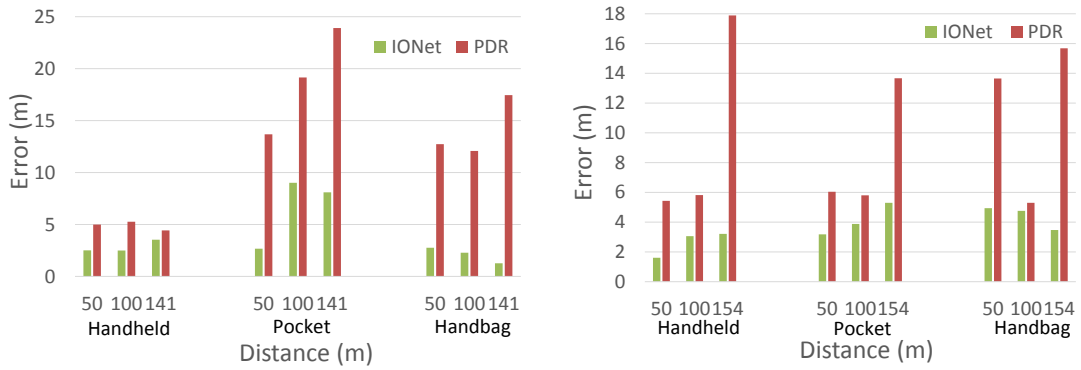
(c) In Pocket (Floor A)

(d) In Pocket (Floor B)

(e) In Handbag (Floor A)

(f) In Handbag (Floor B)

Figure 3.10: Our proposed IONet can generate more accurate trajectories in large-scale localization experiments on two office floors - Floor A (1650 m^2) and Floor B (2475 m^2) in three attachments - Handheld, In Pocket, and In Handbag, compared with SINS and PDR. Note that our learning model is trained inside one room (Vicon Room), but generalize well to outside places without further training.



(a) Floor A

(b) Floor B

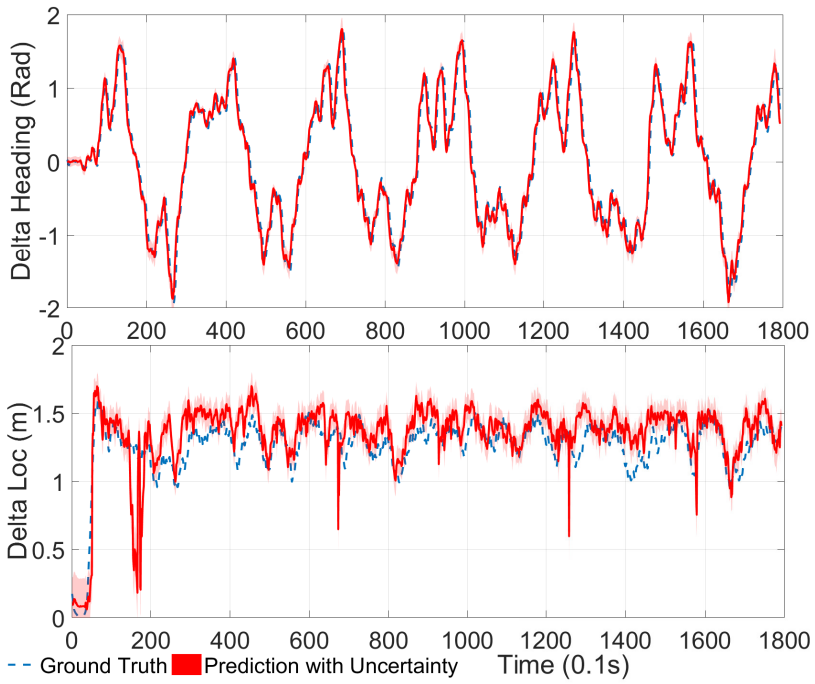
Figure 3.11: In large-scale indoor localization, absolute position errors of IONet is calculated at a distance of 50m, 100m and the end point on Floor A (a) and Floor B (b), showing competitive performance over traditional PDR.

3.8.5 Large-scale Indoor Localization

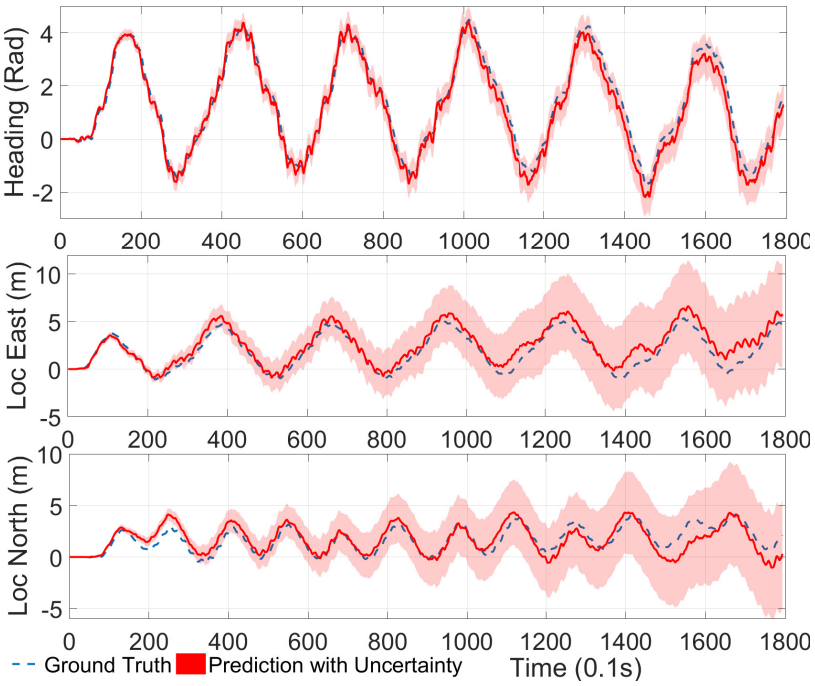
Here, we apply our model on a more challenging indoor localization experiment to present its performance in a new environment. Our model *without training outside* Vicon room, is directly applied to six large-scale experiments conducted on two floors of an office building. The new scenarios contained long straight lines and slopes, which were not contained in the training dataset. Lacking the high precision reference from Vicon, we take Google Tango Tablet [174], a well-known visual-inertial device, as pseudo ground truth.

The floor maps are illustrated in Figure 3.10a (about 1650 m^2 size) and Figure 3.10b (about 2475 m^2). Participants walked normally along corridors with the phone in three attachments respectively. The predicted trajectories from our proposal are closer to Tango trajectories, compared with the two other approaches in Figure 3.10. The continuously propagating error of SINS mechanism caused trajectories with cubic error growth. Impacted by incorrect step detection or inaccurate step stride and heading estimation, PDR accuracy is limited. We calculate absolute position error against pseudo ground truth from Tango at a distance of 50m, 100m and the end point in Figure 3.11. Our IONet shows competitive performance over traditional PDR and has the advantage of generating a continuous trajectory at 10 Hz, although its heading attitude deviates from true values occasionally.

To summarize the generalizability of IONet, Section 3.8.1 stated that IONet gives significantly better performance when applied to different attachments separately. On the other hand, Sections 3.8.3 and 3.8.5 demonstrated that the neural network generalizes well across different users, devices, and test environments. As discussed in Section 3.8.4, IONet can perform well when simultaneously trained and evaluated on multiple motion modes,

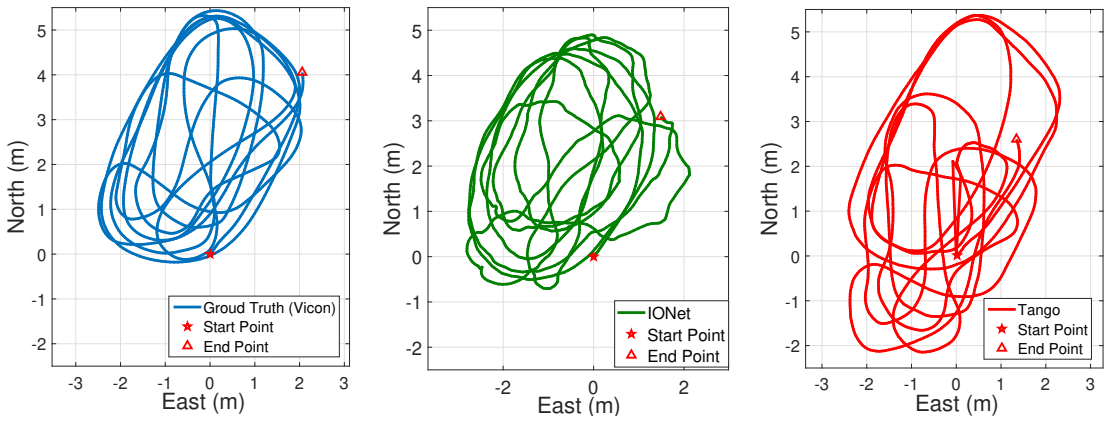


(a) The uncertainty of the polar vector.



(b) The uncertainty of heading and location

Figure 3.12: The predicted mean values of IONet are shown together with their corresponding uncertainty and the ground truth for (a) polar vector and (b) absolute heading and locations.



(a) Ground Truth (b) IONet (c) Tango

Figure 3.13: The trolley tracking trajectories of our proposed (b) IONet are compared with (a) Ground Truth, and (c) Tango. Our proposed IONet shows almost the same accuracy as Tango, and even better robustness, because our pure inertial solution suffers less from environmental factors.

but may give better performance if the network is trained individually for each motion type.

3.8.6 Uncertainty Estimation

We tested the Bayesian model proposed in Section 3.5 to evaluate its ability for uncertainty estimation. The handheld attachment in normal walking was taken as an example. The model was trained on the data collected by iPhone 7Plus, but tested on an Android smartphone Nexus 5, whose IMU has distinct different properties from the iPhone 7Plus, in order to show the model confidence in a different input distribution.

The predicted polar vector (delta location and delta heading) is shown together with its corresponding uncertainty (standard deviation $\boldsymbol{\sigma}$) and the ground truth in Figure 3.12a. From the variance of motion transformation, the uncertainties of absolute heading and locations are inferred according to Equations 3.25 - 3.27, and their results are presented in Figure 3.12b. For heading estimation, the uncertainty captures to what extent the predicted values deviate from the real ones. Although we never provide any labels for training the uncertainty, it automatically learns to represent the likelihood of the predicted results, even with a new input distribution. The corresponding variances of the whole trajectory on the North (y) and East (x) axes increase dramatically. This is because the location sees an integration from the location change in two axes, and it is inferred from both the variances of absolute heading and the delta L.

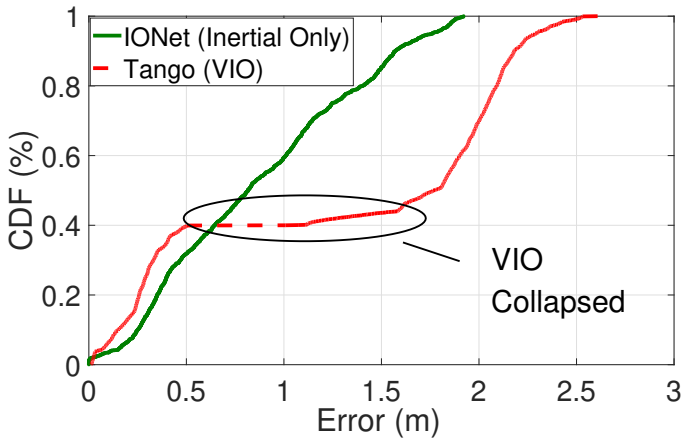


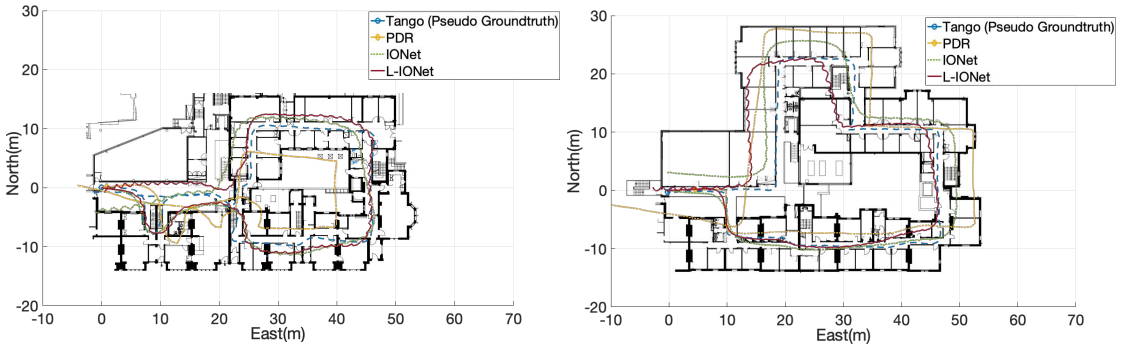
Figure 3.14: The error Cumulative Distribution Function (CDF) of absolute locations predicted by IONet is compared with Tango (visual inertial odometry) in trolley tracking.

3.8.7 Trolley Tracking

We consider a more general problem without periodic motion, which is hard for traditional step-based PDR or SINS on a limited quality IMU. Tracking wheel-based motion, such as a shopping trolley/cart, robot or baby-stroller is highly challenging and hence under-explored. Current approaches to track wheeled objects are mainly based on visual odometry [12, 11] or visual-inertial odometry (VIO) [13, 16]. However, they fail when the device is occluded or operating in low light environments, such as placed in a bag. Moreover, their high energy- and computation-consumption also constrain further application. Here, we apply our model on a trolley tracking problem *using only inertial sensors*. Due to a lack of comparable techniques, our proposal is compared with the state-of-art visual-inertial odometry Tango.

Our experiment devices, namely an iPhone 7 and the Google Tango are attached on a trolley, pushed by a participant. Detailed experiment setup and results could be found in supplementary video [5]. High-precision motion reference was provided by Vicon. The VIO of Tango device is based on Kalman filtering to fuse the inertial navigation system with the visual features extracted from images. However, in VIO the image features are extracted by hand-designed algorithms. In some scenarios, it is hard to obtain enough visual features to estimate the geometry structure of a scene, for example, when cameras are in front of a blank wall, no useful features can be extracted. In our experiments, the VIO collapsed due to the lost features in the structureless and featureless wall of the experimental room, which further breaks down the entire system. From the trajectories from Vicon, our IONet and Tango in Figure 3.13 our proposed approach shows almost the same accuracy as Tango, and

[5] Video is available at: https://youtu.be/L5LtE-PQuHk



(a) Office Floor 1 (b) Office Floor 2

Figure 3.15: The largescale localisation experiments were conducted on (a) Office Floor 1 and (b) Office Floor 2. The trajectories were generated from the IONet, L-IONet and PDR. The pseudo ground truth was provided by Google Tango device.

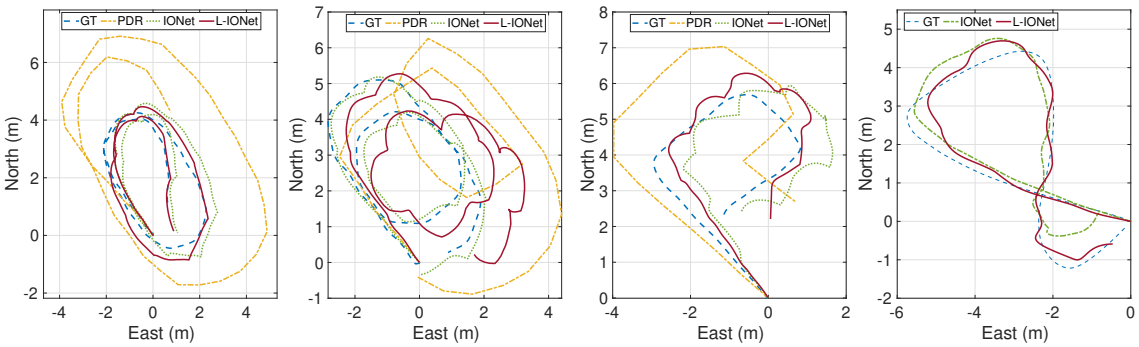
even better robustness, because our pure inertial approach suffers less from environmental factors. With the help of visual features, VIO (Tango) can constrain error drift by fusing visual transformations and the inertial system, but it will collapse when capturing erroneous features or no features, especially in open spaces. This happened in our experiment, shown in Figure 3.14. Although VIO can recover from the collapse, it still left a large distance error.

3.8.8 Lightweight IONet Evaluation

We show how to solve the pedestrian inertial navigation problem using our proposed L-IONet model with above the OxIOD dataset. IONet and L-IONet models can reconstruct trajectories from raw IMU data, and provide users with their accurate locations. A 1-layer Bidirectional LSTM with 128 dimensional hidden states was adopted for IONet, while the WaveNet with 32 filters was used in L-IONet for the evaluation. Both models were trained with the above detailed split training sets from the four attachment categories, i.e. the handheld (20 sequences), pocket (10 sequences), handbag (7 sequences) and trolley (12 sequences). Two sets of experiments were conducted to evaluate our proposed models.

The first set of tests involved tracking a pedestrian with the phone in different attachments. In this experiment, the participant carrying the smartphone was asked to walk normally inside the Vicon room. The IMU data [6] were collected and feed into the IONet and L-IONet to predict the participant's motion. The installed motion capture systems can provide highly precise trajectories as groundtruth. Note that these walking trajectories are not present in the training dataset. A basic PDR algorithm was implemented as a baseline, and

[6]The test data can be found at the 'test' fold of our OxIOD dataset

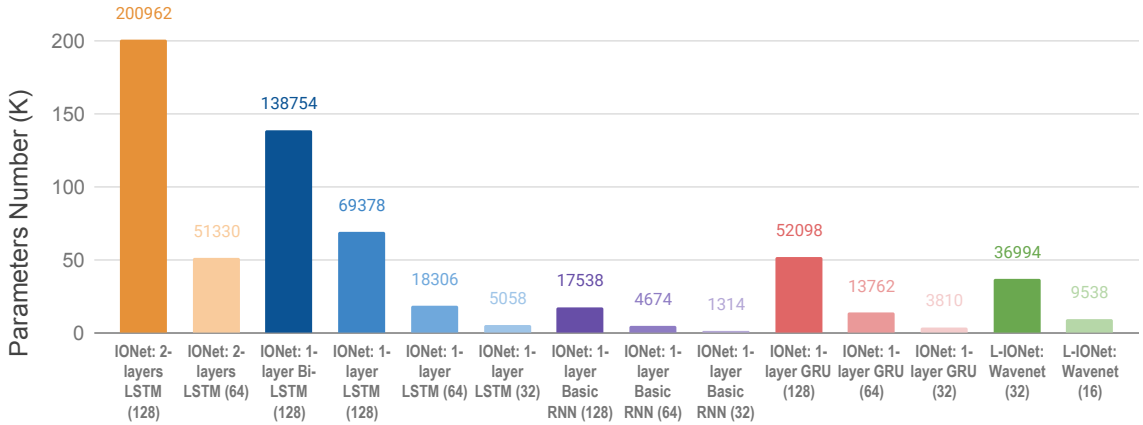


(a) Handheld (b) In Pocket (c) In Handbag (d) On Trolley

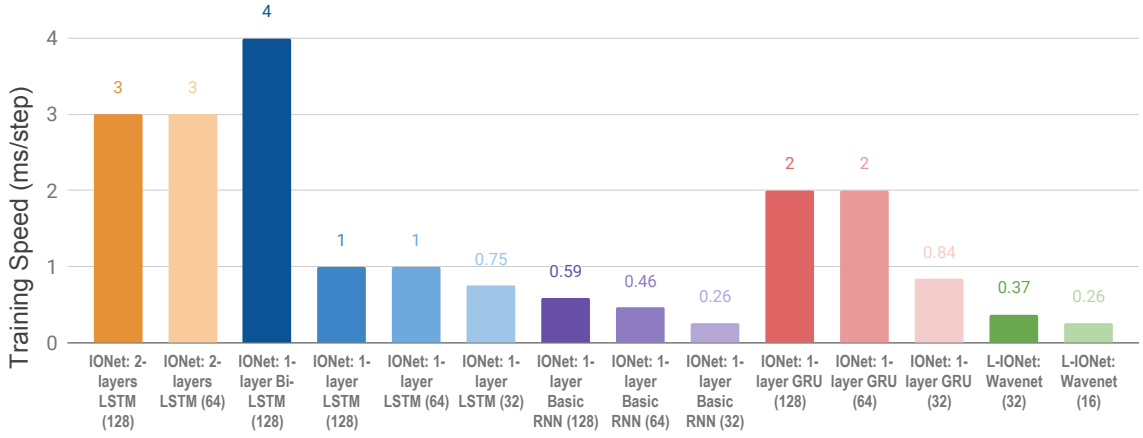
Figure 3.16: The trajectories reconstruction for pedestrian tracking with device in four attachments: a) in the hand, b) in the pocket, c)in the handbag, and d) on the trolley respectively. The trajectories were generated from IONet, L-IONet and a basic PDR algorithm. PDRs do not work when the device was placed on the trolley, as no step can be detected in this situation. The ground truth values are provided by the Vicon System.

we show that our dataset can also be used as a benchmark for conventional PDR algorithms. Figure 3.16 demonstrates the trajectories generated from the groundtruth (blue line), PDR (green line), IONet (orange line) and L-IONet (red line). It indicates that the deep learning based methods outperformed the model-based PDR when the phone was placed either in the hand, pocket or handbag. The trolley tracking is a difficult problem for PDR algorithms, as no step (periodicity pattern) can be detected in this case, and hence a handcrafted model is hard to build for this wheeled motion. In contrast, the learning based approaches are still able to generalise to this general motion, and reconstruct physically meaningful trajectories, while the PDR algorithm fails in this task. The L-IONet model produced results even closer to the groundtruth compared with IONet, especially in the handheld and trolley domains.

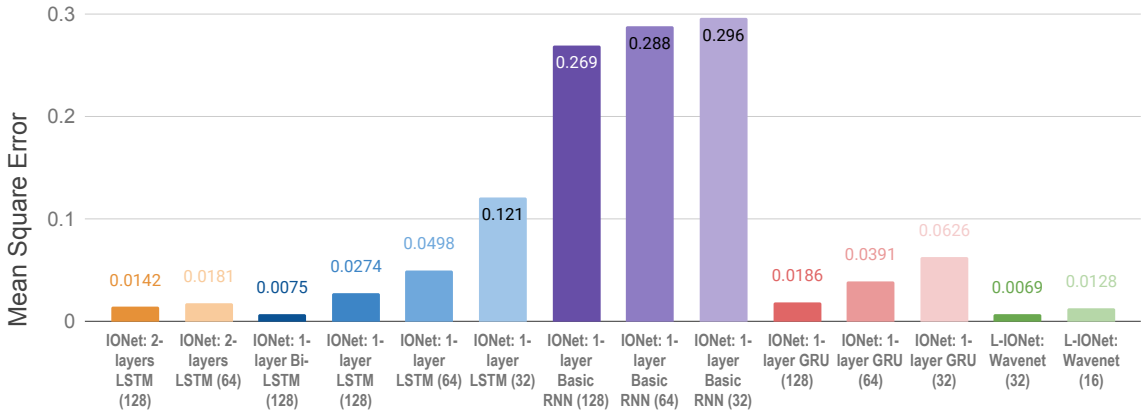
The other set of experiments is to perform large-scale localisation on two floors of an office building. Although DNN models were trained with inertial data collected inside the Vicon room, we show that the models can be used to predict the pedestrian motion outside the room directly. This is due to the fact that inertial data are not sensitive to environments, and hence the proposed DNN models can generalize to a new environment easily. Figure 3.15 demonstrates that both IONet and L-IONet models achieve good localisation results, although the two models never saw any data outside the Vicon room. This experiment shows the generalisation ability of the deep learning based models towards new environments.



(a) Parameters Number



(b) Training Speed



(c) Mean square error

Figure 3.17: A comparison of IONet and L-IONet models in terms of their (a) number of parameters, (b) training (convergence) speed and (c) test accuracy. L-IONet shows competitive performance to IONet, but requires less memory and a quicker training time.

3.8.9 Model Performance on Low-End Devices

To evaluate the performance of our proposed models on low-end devices, we chose three levels off-the-shelf consumer smartphones, i.e. Huawei Mate 8, Nexus 6, HTC One M8, and one consumer smartwatch, i.e. Sony Smartwatch 2. Huawei Mate 8 is equipped with octa-core (4x2.3 GHz and 4x1.8 GHz) CPU and 4 GB RAM. Nexus 6 is equipped with quad-core 2.7 GHz CPU and 3 GB RAM. HTC One M8 is equipped with quad-core 2.5 GHz CPU and 2 GB RAM. Sony Smartwatch 2 is equipped with 1 core 180 MHz CPU and 256 MB RAM. Our IONet and L-IONet models were first trained with the Keras framework on a TITAN X GPU, further converted into Tensorflow Lite models, and then deployed on the low-end devices to test their online inference speed.

We conducted systematic research into the performance of DNNs models for inertial tracking at the edge. The LSTM-based IONet is compared with our proposed WaveNet style L-IONet, with different hyperparameters chosen to demonstrate their impacts on the model performance, which are the number of layers, whether LSTMs are bi-directional or not, the number of hidden states for LSTMs, and the number of convolutional filters for WaveNet. Moreover, we replaced the LSTM module in IONet with Gated Recurrent Units (GRUs, a more lightweight recurrent model) and Basic RNNs as baselines to show the trade-off between model accuracy and efficiency.

Table 3.5: The execution time (ms) of the deep neural networks models on the low-end devices.

Models	Huawei Mate 8	Nexus 5	HTC One M8	Sony SW2
IONet: 2-layers LSTM (128)	38.13	38.65	88.13	342.61
IONet: 2-layers LSTM (64)	11.42	14.74	33.62	109.41
IONet: 1-layer Bi-LSTM (128)	27.23	31.08	71.02	261.08
IONet: 1-layer LSTM (128)	12.7	16.15	37.38	130.85
IONet: 1-layer LSTM (64)	4.65	7.08	16.69	48.9
IONet: 1-layer LSTM (32)	1.32	2.25	3.49	18.7
IONet: 1-layer Basic RNN (128)	2.4	3.13	4.63	31.2
IONet: 1-layer Basic RNN (64)	0.86	1.7	2.69	14.06
IONet: 1-layer Basic RNN (32)	0.46	1.13	1.94	8.78
IONet: 1-layer GRU (128)	7.29	12.92	14.72	81.8
IONet: 1-layer GRU (64)	3.02	6.21	8.00	35.03
IONet: 1-layer GRU (32)	1.76	4.24	5.68	21.54
L-IONet: WaveNet (32)	3.7	6.47	13.74	56.78
L-IONet: WaveNet (16)	1.27	3.58	8.43	27

Figure 3.17 compares different model configurations of IONet (LSTM), IONet (GRU),

IONet (Basic RNN) and L-IONet (WaveNet), in terms of their number of parameters, training speed and mean square error (MSE) of predicted polar vectors. It is clear to see that the L-IONet with 32 filters i.e. WaveNet (32), achieves the highest accuracy, with a prediction error of 0.0069, even slightly lower than that of IONet with 1-layer Bi_LSTM (128), i.e. 1-layer Birectional LSTM with 128 hidden states. In contrast, the number of parameters in the L-IONet with WaveNet (32) is only one quarter that of the IONet with 1-layer Bi-LSTM (128). Meanwhile, L-IONet with WaveNet (32) is around 10 times faster than IONet with 1-layer Bi-LSTM when training on a Tesla K80 GPU. This indicates that L-IONet shows competitive performance in accuracy over IONet, while still superior in the speed and resource consumption.

Table 3.5 illustrates the execution time of different IONet (LSTM, GRU, Basic RNN) and L-IONet (WaveNet) models when deployed on Huawei Mate 8, Nexus 5, HTC One M8 and Sony SW2 respectively. The execution time (milliseconds, ms) is the average inference time of these models at the low-end devices. The L-IONet models, i.e. WaveNet (32) and WaveNet (16) performed faster inference than the LSTM-based IONet models. Even at the swartwatch device equipped with very limited CPU and memory, our proposed L-IONet is capable of realising real-time inference, producing outputs within only 56.78 ms (WaveNet (32)) and 27 ms (WaveNet (16)) for each step. The inference speed of IONet with 1-layer LSTM (64) is similar to WaveNet (32), but its prediction error is almost 8 times higher than WaveNet (32). We further compare LSTM (32) with WaveNet (32) and find that the prediction error of LSTM (32) increases to 17.5 times higher than WaveNet (32), although LSTM (32) is with faster inference speed. It is interesting to see that GRUs are more lightweight compared with both LSTM and WaveNet models. However, the prediction accuracy of GRU models are not satisfying with larger prediction errors, i.e. 0.0186, 0.0391, and 0.0626 for GRU(128), GRU(64) and GRU (32) respectively, around 3, 6, 9 times higher than WaveNet (32). The Basic RNNs (128) (64) (32) have fewer parameters, and faster inference speed on low-end devices than our WaveNet-based L-IONet, but they almost learned nothing from inertial data with huge test errors (i.e. 0.268, 0.288 and 0.296), nearly 40 times larger than the WaveNet models. Therefore, our WaveNet based L-IONet models show better trade-off between the prediction accuracy and on-device inference efficiency. It is worth noticing that the Wavenet-based L-IONet owns the advantages of faster training speed over Basic RNNs, LSTMs, and GRUs, as shown in Figure 3.17 (b). This is because feed-forward models are easier to train and optimize than recurrent models.

3.9 Summary

This chapter presented IONet, a novel deep neural network framework to learn and estimate motion displacements over a given time window using low-cost inertial sensors. Specifically, inertial measurements and ground truth labels were input to this model to learn the transformation between the raw measurements and the movement of the sensor. The method makes no assumption about either sensor placement or user motion and is therefore able to circumvent fundamental limitations of existing methods for pedestrian inertial navigation, e.g. pedestrian dead reckoning. The performance of the method was evaluated through extensive experiments including not only typical pedestrian motions such as walking and running at different speeds, but also wheeled configurations, e.g. trolley tracking. Performance evaluations demonstrated our proposed IONet outperformed competing algorithms based on standard inertial double-integration systems and handcrafted step detection algorithms in several scenarios.

We present and release the Oxford Inertial Odometry Dataset (OxIOD), a first-of-its-kind public dataset for deep learning based inertial navigation research, with a large amount of pedestrian, multi-attachment sensor data (158 sequences, totalling more than 42 km in distance), and high-precise labels. In order to capture human motion that accurately reflects everyday usage, this dataset was collected with a high degree of diversity, across different attachments, motion modes, users, types of device and places. OxIOD is able to be used to train robust and accurate deep learning models for inertial navigation, and we evaluate both the classical algorithms (PDRs) and data-driven models on OxIOD as a common benchmark.

To enhance the online efficiency of DNN models on low-end devices, we further developed Lightweight Inertial Odometry Neural Networks (L-IONet), a lightweight deep neural network framework to learn inertial tracking, more efficient at training and inference than the LSTM-based IONet model.

Chapter 4

Selective Sensor Fusion

Autonomous vehicles and mobile robotic systems are typically equipped with multiple sensors to provide redundancy. By integrating the observations from different sensors, these mobile agents are able to perceive the environment and estimate system states, e.g. locations and orientations. Although deep learning approaches for multimodal odometry estimation and localization have gained traction, they rarely focus on the issue of robust sensor fusion - a necessary consideration to deal with noisy or incomplete sensor observations in the real world. This chapter proposes SelectFusion [126], an end-to-end selective sensor fusion module which can be applied to complimentary pairs of sensor modalities such as monocular images and inertial measurements, depth images and LIDAR point clouds. In particular, we propose two fusion modules based on different attention strategies: deterministic soft fusion and stochastic hard fusion, and we offer a comprehensive study of the new strategies compared to trivial direct fusion. We evaluate all fusion strategies in both ideal conditions and on progressively degraded datasets that present occlusions, noisy and missing data and time misalignment between sensors, and we investigate the effectiveness of the different fusion strategies in attending to the most reliable features, which in itself, provides insights into the operation of the various models.

4.1 Introduction

Mobile agents are often outfitted with multiple sensors. For example, a self-driving vehicle is equipped with a combination of GPS, IMUs, monocular or stereo video cameras, LIDAR. Making such mobile agents fully autonomous and intelligent requires the ability of fusion, a method that can effectively exploit the individual strengths of distinct sensors and coherently estimate the system states. Multimodal sensor fusion has long been a central

problem in robotics and computer vision [56], with application to a variety of tasks such as perception, planning and controlling. Despite different applications, the rationale of sensor fusion is more or less the same: many system state variables cannot be always observable by a single sensor modality, while different sensors can be complementary to one another. Conventional sensor fusion methods resort to handcrafted designs that heavily relies on human experience and domain knowledge. Consequently, the developed fusion methods are often modality-specific and/or task-specific.

A typical example is integrating visual and inertial sensors in the form of Visual-Inertial Odometry (VIO) [13, 14, 15, 16], which enables ubiquitous mobility for mobile agents by providing robust and accurate pose information. These two sensors are relatively low-cost, light-weight, power-efficient and widely found in robots, smartphones, and VR/AR wearable devices. Single cameras are able to capture the appearance and structure of a 3D scene. However, they are scale-ambiguous, and not robust to most challenging scenarios, e.g. strong lightning changes, lack of textures and high-speed motions. In contrast, IMUs are completely ego-centric, scene-independent, and can provide absolute metric scale, but inertial measurements are corrupted by sensor noise and biases. Existing VIO approaches generally follow a standard pipeline that involves the fine-tuning of two modules, feature detection and tracking, and of the sensor fusion strategy. These methods rely on handcrafted features, and the fusion strategy takes the form of Bayesian filtering [13], fixed-lag smoothers or full smoothers [14, 15, 16].

Recently, there has been growing interest in applying deep neural networks (DNNs) for *learning to estimate system states* in an end-to-end manner, for example, solving visual-odometry (VO) [85, 86], visual-inertial odometry (VIO) [89, 120] or camera relocalization [81, 83]. Instead of building analytical models by hand, this is achieved by learning complex mappings directly from raw sensor data to target values. These end-to-end approaches are appealing due to the capability of deep learning in automatic feature extraction from high-dimensional raw data. However, despite the long history of classical sensor fusion algorithms, there is a lack of effective fusion strategies over deep feature space, especially in the tasks of localization and odometry estimation. These previous learning-based methods do not explicitly model the sources of degradation in real-world usage. Without considering possible sensor errors, all features are directly fed into other modules for further pose regression in [84, 83, 81], or simply concatenated as in [89]. These factors can potentially impact the accuracy and safety of neural systems, when the input data are corrupted or missing. Moreover, features from different modalities are considered equally important in these methods, although the complementary property of different modalities require sys-

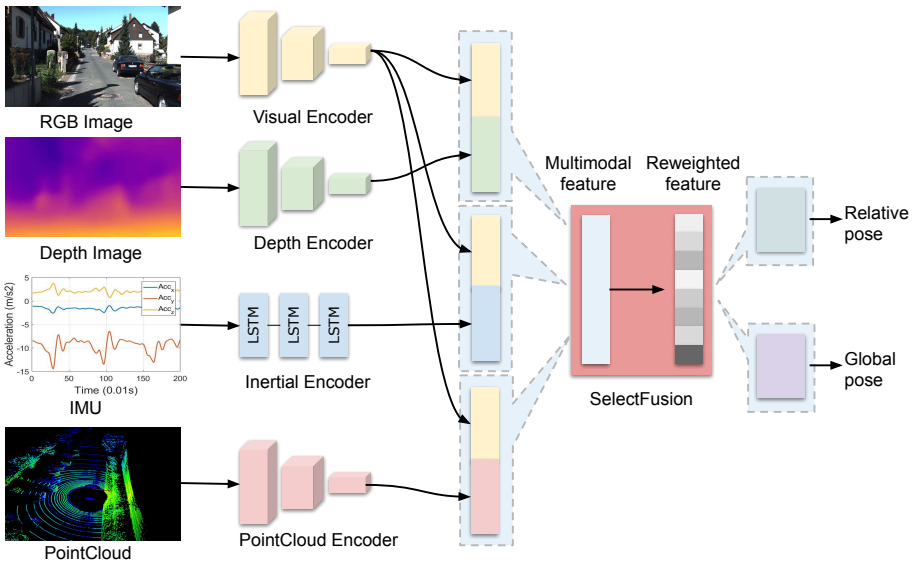


Figure 4.1: An overview of the general framework to learn system states from multiple sensor modalities. Our framework can selectively utilize the suitable features for solving problems to improve both the accuracy and robustness. In our example, the network inputs a pair of sensor modalities from RGB image, depth image, inertial measurements, or point cloud data, and outputs relative pose or global locations.

tems to utilise deep features with regard to observation uncertainties or self/environmental dynamics.

For this reason, we present a generic framework that models feature selection for robust sensor fusion, as illustrated in Figure 4.1. In this work, we mainly consider the problem of using a pair of sensor modalities, although it can be extended naturally to three or more modalities. As a case study, two tasks - learning global localization and ego-motion estimation, are chosen to demonstrate the effectiveness of our proposed selective sensor fusion. Our system is not restricted to specific modality, performing feature selection from four different sensor data, i.e. RGB-images, inertial measurements, LIDAR point clouds and depth images. The selection process is conditioned on the measurement reliability and the dynamics of both self-motion and environment. Two alternative feature weighting strategies are presented: soft fusion, implemented in a deterministic fashion; and hard fusion, which introduces stochastic noise and intuitively learns to keep the most relevant feature representations, while discarding useless or misleading information. Both architectures are trained in an end-to-end fashion.

By explicitly modelling the selection process, we are able to demonstrate the strong correlation between the selected features and the environmental/measurement dynamics by visualizing the sensor fusion masks, as illustrated in Figure 4.7. In the case of estimating visual-inertial odometry, our results show that features extracted from different modalities

(i.e., vision and inertial motion) are complementary in various conditions: the inertial features contribute more in presence of fast rotation, while visual features are preferred during large translations (Figure 4.10). Thus, the selective sensor fusion provides insights into the underlying strengths of each sensor modality, guiding future multimodal system design. We also demonstrate how incorporating selective sensor fusion makes neural models robust to data corruption typically encountered in real-world scenarios.

This chapter presents a generic framework for selective sensor fusion in multimodal deep pose estimation. To summarise, the novel contributions of this work are as follows:

- We present a novel generic framework to learn selective sensor fusion enabling more robust and accurate odometry and localization in real-world scenarios.
- We show how our selective sensor fusion can be incorporated into a uniform framework, not restricted by specific modality or task, by learning odometry estimation or relocalization by fusing a pair of modalities from vision, depth, inertial and LIDAR data.
- Our selective sensor fusion masks can be visualized and interpreted, providing deeper insight into the relative strengths of each stream, and guiding further system design.
- We create challenging datasets on top of current public datasets by considering seven different sources of sensor degradation, and conduct a new and complete study on the accuracy and robustness of deep sensor fusion in presence of corrupted data.

4.2 Related Work

Deep Learning for Localization We discuss several learning based localization models relevant to this chapter, i.e. deep learning for visual-inertial odometry, relocalization and lidar odometry. VINet [89] used a neural network to learn visual-inertial odometry, by directly concatenating visual and inertial features. We observe that this previous method has not properly addressed the problem of learning a meaningful sensor fusion strategy, but simply concatenated visual and inertial features in the latent space. We argue that a gap between deep architectures and traditional model estimation techniques currently lies in a careful design of the fusion strategy. VIOLearner [120] presents an online error correction module for deep visual-inertial odometry that estimates the trajectory by fusing RGB-D images with inertial data. DeepVIO [138] recently proposed a fusion network to

fuse visual and inertial features. This network is trained with a dedicated loss. However, this way of learning sensor fusion does not expose the behaviour of the fusion module, while in our approach we propose the use of an interpretable mask, that offers insight into the usefulness of the input at any time.

Deep approaches have also been devoted to visual relocalization. Posenet [81] was the first work to use Convolutional Neural Networks (CNNs) for 6-DoF pose regression from monocular images. PoseNet has been further improved by combining CNNs and LSTMs [83], or by adding additional co-visibility constraints based on local maps and the estimated odometry [84].

Learning LIDAR odometry has been explored by LO-Net [127, 128], which exploits geometric consistency for scan-to-scan motion estimation, while also learning pose correction similarly to deep SLAM approaches, and can achieve accuracy comparable to traditional approaches [69]. Fusion of LIDAR and visual information has been investigated in [178], which proposes to fuse LIDAR and visual information, but in their work the learning is limited to training a model for removal of moving objects rather than localization. We provide a generic framework for deep features fusion, and outperformed the direct fusion in different scenarios.

Multimodal Learning Multimodal learning aims to solve machine learning problems involving multiple data modalities. The success of multimodal learning has been demonstrated in a wide range of applications, e.g. audio-visual speech classification [179] and recognition [180], face recognition [181], manipulation [182], and autonomous navigation [183]. However, there is a lack of systematic study into the sensor fusion for deep state estimation, especially in learning based localization and pose estimation.

Attention Mechanism Our proposed selective sensor fusion is particularly related to attention mechanisms, that have been widely applied in neural machine translation [74], image caption generation [107], and video description [184]. Limited by the fixed-length vector in embedding space, these attention mechanisms compute a focus map to help the decoder, when generating a sequence of words. This is different from our design intention that the features selection works to fuse multimodal sensor fusion for deep pose estimation, and cope with more complex error resources, and self-motion dynamics.

Interpretability On the other hand, interpretability has become a desirable property for learned models, in particular for applications in which such models are used to inform critical decisions in the real world (e.g, navigation of autonomous vehicles). In these instances,

black-box models are not adequate [185]. For this reason, interpretable attentive models are gaining traction [186].

4.3 Learning Multimodal State Estimation

In this section, we present a uniform framework to learn multimodal representation for state estimation, which lays the foundation for our proposed selective sensor fusion. Figure 4.2, 4.3 and 4.4, show a modular overview of the architecture, consisting of feature encoders (i.e. visual, depth, inertial, and pointcloud encoder), feature fusion, temporal modelling and task solver (i.e. odometry estimation or relocalization). Our model takes in a sequence of raw sensor data, and generates their corresponding system states, i.e. relative poses or global locations. With the exception of our novel feature fusion, the pipeline can be any generic deep state estimation system. In the Feature Fusion component, we propose two different selection mechanisms (soft and hard) and compare them with direct (i.e. a uniform/unweighted mask) fusion, as shown in Figure 4.5.

4.3.1 Feature Encoders

4.3.1.1 Visual Feature Encoders

As visual feature encoders are used in both global relocalization and odometry estimation, they are designed with respect to the property of each task for better feature extraction and utilization.

For a relative pose (odometry) estimation, latent representations are extracted from a set of two consecutive monocular images $\mathbf{x}_V$. Ideally, we want our visual encoder f_{vision} to learn geometrically meaningful features rather than features overfitted with appearance or context. For this reason, instead of using a PoseNet model [81], as commonly found in other DL-based VO approaches [86, 88, 87], we use a FlowNet-style architecture, i.e. FlowNetSimple [187] as our feature encoder. Flownet provides features that are suited for optical flow prediction, which highly contributes to the motion detection. The network consists of nine convolutional layers. The size of the receptive fields gradually reduces from 7×7 to 5×5 and finally 3×3, with stride two for the first six. Each layer is followed by a ReLU nonlinearity except for the last one, and we use the features from the last convolutional layer $\mathbf{a}_V$ as our visual feature. We initialize the visual encoder with the weights of a model that was pre-trained on the FlyingChairs dataset[1], since training from scratch

[1] https://lmb.informatik.uni-freiburg.de/resources/datasets/FlyingChairs.en.html

would require larger amounts of data compared with our dataset size. The Visual Encoder (FlowNet) is employed to learn visual-inertial odometry and LIDAR-vision odometry, as shown in Figure 4.4 and 4.3.

For a global relocalization task, we instead use Residual Neural Network (ResNet) [188] to extract features from a set of single images. Both structure and appearance features contribute to the retrieval of absolute poses in the 3D scene that has been visited before. Hence, visual features should best capture the entire scene. We adopt ResNet18, consisting of 18 layers convolutional layers with skip connections, and modify it by introducing an average pooling layer and a full-connected layer at the end, that map the features after ResNet18 to a d dimension visual feature $\mathbf{a}_V$. The Visual Encoder (ResNet) is used in the depth-vision based relocalization, as illustrated in Figure 4.2.

In summary, given a set of images $\mathbf{x}_V$, we are able to extract visual features $\mathbf{a}_V \in \mathbb{R}^d$ suitable for the task via the Visual Encoder (FlowNet) or (ResNet) f_{vision}:

$$\mathbf{a}_V = f_{\text{vision}}(\mathbf{x}_V). \tag{4.1}$$

4.3.1.2 Inertial Feature Encoder

Inertial data streams have a strong temporal component, and are generally available at higher frequency ($\sim$100 Hz) than images ($\sim$10 Hz). In order to model the temporal dependencies of the consecutive inertial measurements, we use a two-layer Bi-directional LSTM with 128 hidden states as the Inertial Feature Encoder f_{inertial}. In the deep VIO model, as shown in Figure 4.4, a window of inertial measurements $\mathbf{x}_I$ between each two images is fed to the inertial feature encoder in order to extract the d dimensional feature vector $\mathbf{a}_I \in \mathbb{R}^d$:

$$\mathbf{a}_I = f_{\text{inertial}}(\mathbf{x}_I). \tag{4.2}$$

4.3.1.3 Depth Feature Encoder

In our work, the depth image is exploited to solve the task of vision-depth based relocalization, as shown in Figure 4.2. Similar to the visual encoder designed for relocalization, we also use ResNet18 as the depth feature encoder, but replace the first layer of ResNet model with a 1-channel convolutional network, considering that the depth image is 1-channel rather than 3-channels. Hence, the input is a set of 1-channel depth images $\mathbf{x}_D$, and transformed into a d dimensional features vector $\mathbf{a}_D \in \mathbb{R}^d$ via the depth encoder f_{depth}:

$$\mathbf{a}_D = f_{\text{depth}}(\mathbf{x}_D). \tag{4.3}$$

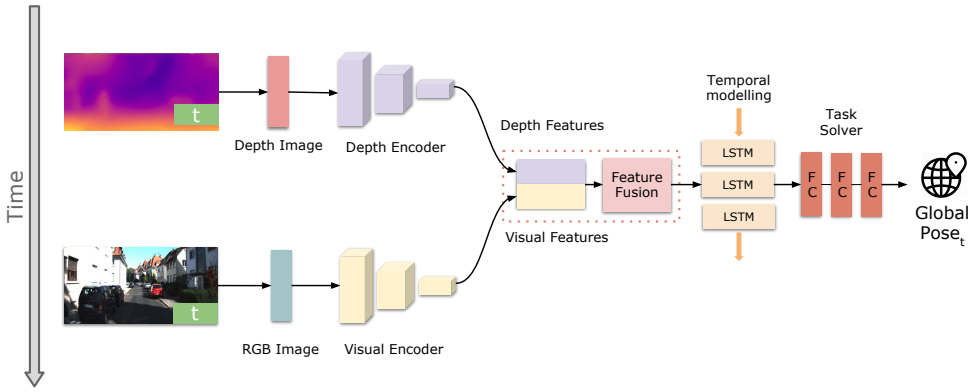


Figure 4.2: An overview of our depth-vision relocalization (**Task 1**) architecture with proposed selective sensor fusion, consisting of depth and visual encoders, feature fusion, temporal modelling and task solver (global pose estimation).

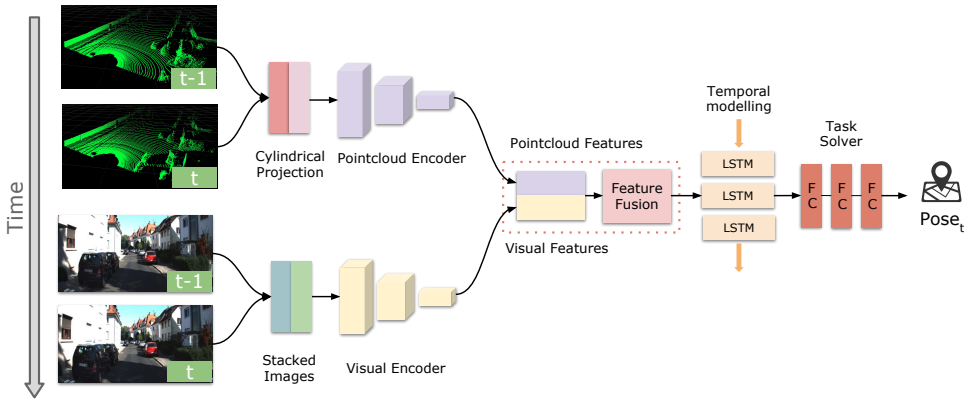


Figure 4.3: An overview of our neural LIDAR-visual odometry (**Task 2**) architecture with proposed selective sensor fusion, consisting of visual and LIDAR encoders, feature fusion, temporal modelling and task solver (relative pose regression).

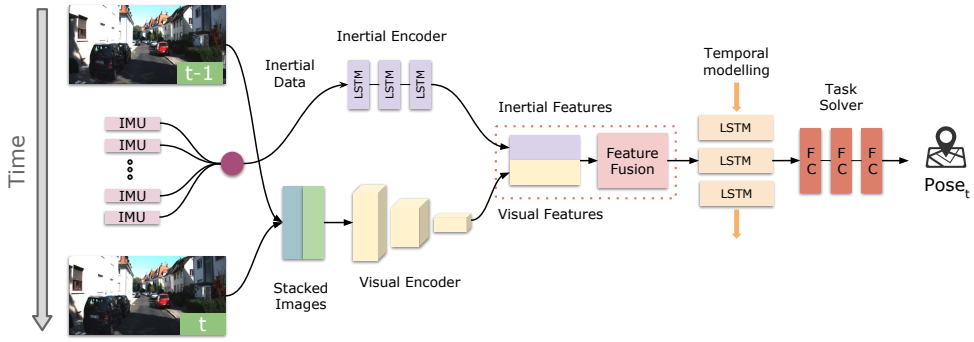


Figure 4.4: An overview of our neural visual-inertial odometry (**Task 3**) architecture with proposed selective sensor fusion, consisting of visual and inertial encoders, feature fusion, temporal modelling and task solver (relative pose regression).

4.3.1.4 Pointcloud Feature Encoder

The point clouds are a set of data in Cartesian coordinates, representing 3D structure in space. They are produced normally by LIDAR devices. The sparse structure and irregular format of point cloud data make them challenging to be processed directly by neural networks. To allow convolutional neural networks to effectively process point cloud data, we convert them into a regular point cloud matrix via the cylindrical projection [189, 127]:

$$\alpha = \text{arctan}(y/x)/\Delta\alpha \tag{4.4}$$

$$\beta = \text{arcsin}(z/\sqrt{x^2 + y^2 + z^2})/\Delta\beta \tag{4.5}$$

where (x, y, z) are original coordinates in LIDAR coordinate system, and (α, β) are new coordinates in the point cloud matrix. The new point cloud matrix is with a size of $H \times W \times C$. The position (α, β) of matrix is filled with the range value $r = \sqrt{x^2 + y^2 + z^2}$ from the position (x, y, z) of original point cloud.

In this work, the point cloud data are used to learn vision-LIDAR odometry, as shown in Figure 4.3 and hence we also use the FlowNet visual encoder to transform the input matrix $\mathbf{x}_P$ into a d dimensional point cloud feature $\mathbf{a}_P \in \mathbb{R}^d$:

$$\mathbf{a}_P = f_{\text{pointcloud}}(\mathbf{x}_P). \tag{4.6}$$

4.3.2 Fusion Function

We now combine the high-level representation produced by each feature encoder from raw data sequences, with a fusion function g that combines information from a pair of sensor modalities to extract the useful combined feature $\mathbf{z}$ for a regression task:

$$\mathbf{z} = g(\mathbf{a}_1, \mathbf{a}_2), \tag{4.7}$$

where $(\mathbf{a}_1, \mathbf{a}_2)$ is any pair of sensor modality features from visual $\mathbf{a}_V$, inertial $\mathbf{a}_I$, depth $\mathbf{a}_D$, and point cloud $\mathbf{a}_P$ channels. In this work, we specifically investigate the problem of fusing two sensor modalities for better demonstration, although our framework can extend naturally to exploit three or more modalities.

There are several different ways to implement this fusion function. The current approach is to directly concatenate the two features together into one feature space (we call this method direct fusion g_{direct}). However, in order to learn a robust sensor fusion model, we propose two fusion schemes – deterministic soft fusion g_{soft} and stochastic hard fusion

g_{hard}, which explicitly model the feature selection process according to the current environment dynamics and the reliability of the data input. Our selective fusion mechanisms re-weights the concatenated inertial-visual features, guided by the concatenated features themselves. The fusion network is another deep neural network and is end-to-end trainable. Details will be discussed in Section 4.4.

4.3.3 Temporal Modelling and Task Solvers

The fundamental tenet of state estimation requires modelling temporal dependencies to derive accurate system states, e.g. relative poses. In the past, a state-space-model (SSM) describes this temporal relation and evolution of system states. Similarly, in our learning model, a recurrent neural network, i.e. Long Short-Term Memory (LSTM) network takes in the input combined feature representation $\mathbf{z}_t$ at time step t and its previous hidden states $\mathbf{h}_{t-1}$ and models the dynamics and connections between a sequence of features. The hidden states $\mathbf{h}_t$ contains the history of the features relevant to the task. After the recurrent network, a fully-connected layer serves as the regressor, mapping the features to a system state $\mathbf{y}_t$, i.e. pose transformation or global pose, representing the motion transformation over a time window or a global location.

Hence, the relation between the final system states $\mathbf{y}_t$ and the input features $\mathbf{z}_t$ can be described via the recurrent neural network and the previous hidden states $\mathbf{h}_{t-1}$:

$$\mathbf{y}_t = \mathbf{RNN}(\mathbf{z}_t, \mathbf{h}_{t-1}). \tag{4.8}$$

We implemented three tasks above this multimodal representation learning framework to estimate key system states from pairs of raw sensory data.

4.3.4 Task 1: Learning Vision-Depth Relocalization

The first task is to exploit monocular RGB images and depth images to perform global relocalization in the scenarios that have been visited before. As illustrated in Figure 4.2, depth and RGB images are encoded into features by the Depth Encoder and Visual Encoder (ResNet), fused as new features through Feature Fusion modules, and converted into global poses via temporal modelling and task regression modules. The global pose $\mathbf{y} = [\mathbf{p}, \mathbf{q}]$ is presented by a 3-dimensional position vector $\mathbf{p} \in \mathbb{R}^3$ and a 4-dimensional quaternion based orientation vector $\mathbf{q} \in \mathbb{R}^4$. The objective is to minimize the L1 distance between the groundtruth values $[\hat{\mathbf{p}}, \hat{\mathbf{q}}]$ and predicted values $[\mathbf{p}, \mathbf{q}]$ with the loss function:

$$L(\theta)_1 = |\hat{\mathbf{p}} - \mathbf{p}| + \lambda_1 \left| \hat{\mathbf{q}} - \frac{\mathbf{q}}{||\mathbf{q}||} \right|, \tag{4.9}$$

where λ_1 is a balance factor, which we choose as $\lambda_1 = 10$ in our experiment. Here, L1 loss is chosen rather than L2 loss, because L1 loss performs better and more stable [82].

4.3.5 Task 2: Learning Lidar-Vision Odometry

The second task is to learn lidar-vision odometry. Different from global relocalization, odometry estimation produces relative poses between two frames of images, which can adapt to new scenarios. Global pose is achieved by integrating pose transformations. As shown in Figure 4.3, the framework consists of Pointcloud Encoder and Visual Encoder (FlowNet) that extract features from lidar pointcloud data and RGB images, Feature Fusion that combines lidar and visual features as a new feature vector, and Temporal Modelling and Task Solver modules to transform features as system states. The network outputs relative poses $\mathbf{y} = [\mathbf{p}, \mathbf{r}]$, consisting of a 3-dimensional translation vector $\mathbf{p} \in \mathbb{R}^3$, and a 3-dimensional Euler rotation vector $\mathbf{r} \in \mathbb{R}^3$. The objective is to minimize the Mean Square Error (MSE) of the relative poses to recover optimal neural networks parameters θ:

$$L(\theta)_2 = ||\hat{\mathbf{p}} - \mathbf{p}||_2 + \lambda_2 ||\hat{\mathbf{r}} - \mathbf{r}||_2, \tag{4.10}$$

where $[\hat{\mathbf{p}}, \hat{\mathbf{r}}]$ are groundtruth values, and λ_2 is a scale factor to balance between translational error and rotational error. λ_2 is chosen as 100 in our experiment.

4.3.6 Task 3: Learning Visual-Inertial Odometry

The third task is to learn visual-inertial odometry, providing accurate pose estimation by using visual and inertial sensors, which are widely deployed in mobile robotics, self-driving vehicles and drones. Similar to lidar-vision odometry, our model outputs the relative poses between two frames of images. Figure 4.4 shows that visual and inertial features are extracted from consecutive monocular images, and a sequence of inertial data between two frames of images by FlowNet based Visual Encoder and LSTM based Inertial Encoder. The features are combined as new features via Feature Fusion, and converted into system states through Temporal Modelling and Task Regressor. The network produces pose transformation $\mathbf{y} = [\mathbf{p}, \mathbf{r}]$ with a 3-dimensional translation vector $\mathbf{p} \in \mathbb{R}^3$, and a 3-dimensional rotation vector $\mathbf{r} \in \mathbb{R}^3$. By minimizing the MSE of the predicted relative poses, the optimal parameters θ are recovered via:

$$L(\theta)_3 = ||\hat{\mathbf{p}} - \mathbf{p}||_2 + \lambda_3 ||\hat{\mathbf{r}} - \hat{\mathbf{r}}||_2, \tag{4.11}$$

where $[\hat{\mathbf{p}}, \hat{\mathbf{r}}]$ are true relative poses, $[\mathbf{p}, \mathbf{r}]$ are predicted values, and λ_3 is a scale factor to balance between translational error and rotational error. In our case, we choose λ_3 as 100.

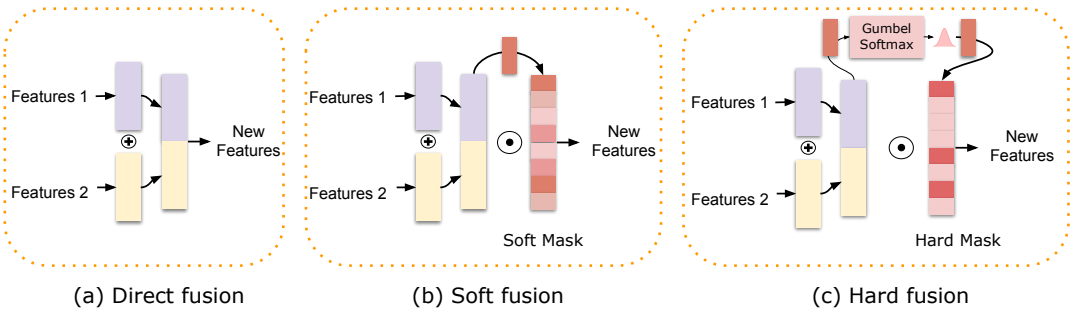


Figure 4.5: An overview of three fusion methods: (a) direct fusion, (b) soft fusion and (c) hard fusion.

4.4 Sensor Fusion Module

In this section, we propose SelectFusion, a generic framework to selectively learn multisensory representation from raw data. Intuitively, the features from each modality offer different strengths for the task of state estimation. For example, in the case of visual-inertial odometry, visual and inertial inputs contribute complementarily to the pose regression. Our perspective is that simply considering all features as though they are equally important and correct, without any consideration of degradation and self/environmental dynamics, is unwise and will lead to unrecoverable drifts and errors. Therefore, we propose two different selective sensor fusion schemes for explicitly learning the feature selection process: soft (deterministic) fusion, and hard (stochastic) fusion, as illustrated in Figure 4.6. In addition, we also present a straightforward sensor fusion scheme – direct fusion – as a baseline model for comparison.

4.4.1 Direct Fusion

A straightforward approach for implementing sensor fusion consists in the use of Multi-Layer Perceptrons (MLPs) to combine the features from the two sensor modality channels. Ideally, the system learns to discriminate relevant features for prediction in an end-to-end fashion. Hence, direct fusion is modelled as:

$$g_{\text{direct}}(\mathbf{a}_1, \mathbf{a}_2) = [\mathbf{a}_1; \mathbf{a}_2] \tag{4.12}$$

where $[\mathbf{a}_1; \mathbf{a}_2]$ denotes an operation function that concatenates features $\mathbf{a}_1$ and $\mathbf{a}_2$, which are extracted from the Modality One and Two respectively.

4.4.2 Soft Fusion

We now propose a soft fusion scheme that explicitly and deterministically models feature selection. Similar to the widely applied attention mechanism [74, 107, 184], this function re-weights each feature by conditioning on both sensor modality channels, allowing the feature selection process to be jointly trained with other modules. The function is deterministic and differentiable.

Here, a pair of continuous masks $\mathbf{s_1}$ and $\mathbf{s_2}$ is introduced to implement soft selection of the extracted feature representations, before these features are passed to temporal modelling and task solver:

$$\mathbf{s}_1 = \text{Sigmoid}(\text{MLP}_1([\mathbf{a}_1; \mathbf{a}_2])) \tag{4.13}$$

$$\mathbf{s}_2 = \text{Sigmoid}(\text{MLP}_2([\mathbf{a}_1; \mathbf{a}_2])) \tag{4.14}$$

where $[\mathbf{a}_1; \mathbf{a}_2]$ denotes an operation function that concatenates features $\mathbf{a}_1$ and $\mathbf{a}_2$. MLP is multilayer perceptron, a feedforward neural network that maps features to fusion mask space. The Sigmoid function makes sure that each of the features will be re-weighted in the range $[0, 1]$. This process is deterministically parameterised by the neural networks, conditioned on both the features $\mathbf{a}_1$ and features $\mathbf{a}_2$. $\mathbf{s_1}$ and $\mathbf{s_2}$ represent soft masks applied to the features extracted from Modality One and Modality Two respectively.

Then, the visual and inertial features are element-wise multiplied with their corresponding soft masks as the new re-weighted vectors. The selective soft fusion function is modelled as

$$g_{\text{soft}}(\mathbf{a}_1, \mathbf{a}_2) = [\mathbf{a}_1 \odot \mathbf{s}_1; \mathbf{a}_2 \odot \mathbf{s}_2]. \tag{4.15}$$

4.4.3 Hard Fusion

In addition to the soft fusion introduced above, we propose a variant of the fusion scheme – hard fusion. Instead of re-weighting each feature with a continuous value, hard fusion learns a stochastic function that generates a binary mask that either propagates the feature or blocks it. This mechanism can be viewed as a switcher for each component of the feature map, which is a stochastic layer implemented by a parametrised Bernoulli distributions.

However, the stochastic layer cannot be trained directly by back-propagation, as gradients will not propagate through discrete latent variables. To tackle this, the REINFORCE algorithm [190, 191] is generally used to construct the gradient estimator. In our case, we propose to employ a more lightweight method – Gumbel-Softmax resampling [192, 193] to infer the stochastic layer of hard fusion, so that our hard fusion module can be trained in an end-to-end fashion as well.

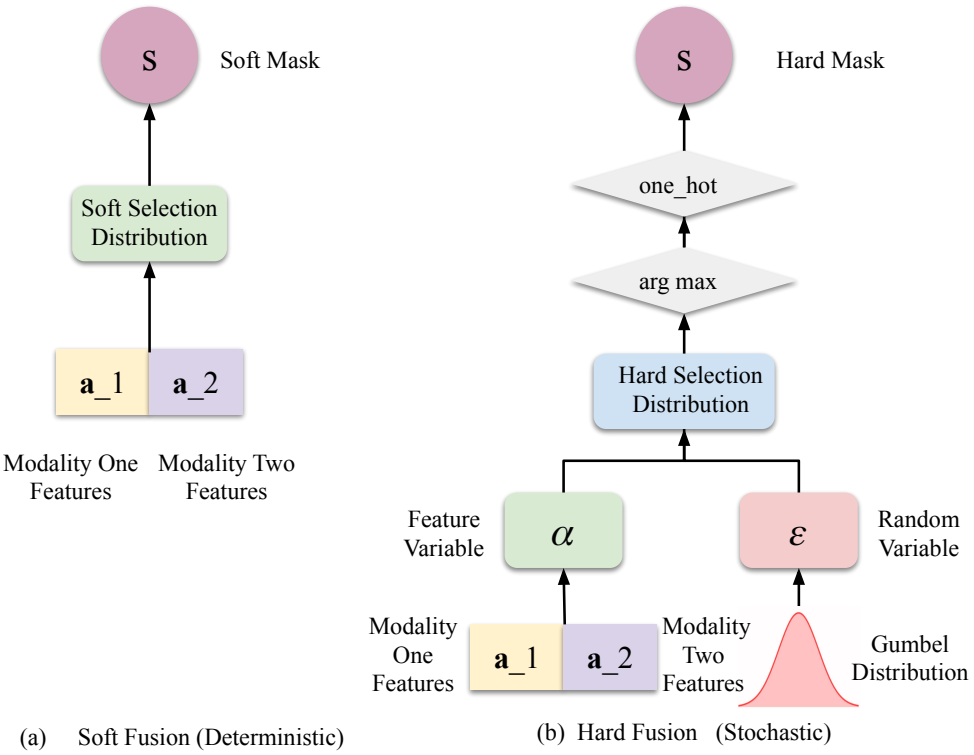


Figure 4.6: An illustration of our proposed soft (deterministic) and hard (stochastic) feature selection process.

Instead of learning masks deterministically from features, hard masks $\mathbf{s}_1$ and $\mathbf{s}_2$, representing the binary mask for the features from two modalities, are re-sampled from a discrete Binomial distribution. This discrete distribution is parameterized by $\boldsymbol{\alpha}$, which is learned by deep neural networks and conditioned on features but with the addition of stochastic noise:

$$\mathbf{s}_1 \sim p(\mathbf{s}_1|\mathbf{a}_1, \mathbf{a}_2) = \text{Binomial}(\boldsymbol{\alpha}) \tag{4.16}$$

$$\mathbf{s}_2 \sim p(\mathbf{s}_2|\mathbf{a}_1, \mathbf{a}_2) = \text{Binomial}(\boldsymbol{\alpha}), \tag{4.17}$$

where each mask $\mathbf{s} = [\mathbf{s}^{(1)}, ..., \mathbf{s}^{(n)}]$ is a n-dimensional binary vector $\mathbf{s}^{(i)}$. Each element of hard mask $\mathbf{s}^{(i)}$ is a 2-dimensional categorical variable, deciding whether to select the ith feature or not. The total number of features is n. The element $\mathbf{s}^{(i)}$ can be viewed as resampling from a Bernoulli distribution:

$$\mathbf{s}^{(i)} \sim \text{Bernoulli}(\boldsymbol{\alpha}^{(i)}). \tag{4.18}$$

Similar to soft fusion, the features from the two modalities are element-wise multiplied with their corresponding hard masks as the new reweighted vectors. The stochastic hard fusion function is modelled as

$$g_{\text{hard}}(\mathbf{a}_1, \mathbf{a}_2) = [\mathbf{a}_1 \odot \mathbf{s}_1; \mathbf{a}_2 \odot \mathbf{s}_2]. \tag{4.19}$$

Now we come to solve the problem of inferring this discrete distribution in order to generate hard mask $\mathbf{s}$. We apply the so-called Gumbel-Softmax trick to convert the non-continuous function into a continuous approximation by using the fact that the distribution of a discrete random variable $P(x = k)$ can be reparameterized by a random variable π_k and a Gumbel random variable ϵ_k via

$$x = \arg\max_k(\log \pi_k + \epsilon_k). \tag{4.20}$$

In practical, it is simple to implement this reparameterization trick into our model. Figure 4.6 (b) shows the detailed workflow of our proposed Gumbel-Softmax resampling based hard fusion. The Gumbel-softmax trick [194] allows us to efficiently draw a hard mask $\mathbf{s^{(i)}}$ from a categorical distribution given the class vector $\pi_k^{(i)}$ and a Gumbel random variable $\epsilon_k^{(i)}$, and then an one-hot encoding performs 'binarization' of the category:

$$\mathbf{s}^{(i)} = \text{one\_hot}(\arg\max_k[\epsilon_k^{(i)} + \log \pi_k^{(i)}]), \tag{4.21}$$

where $i \in [1, .., n]$ is the index of feature, $k \in [1, 2]$ is the index of class vector for each feature. In this case, there are only two categories, indicating whether to select a particular feature or not. This can be viewed as a process of adding independent Gumbel perturbations to the discrete class variable. In practice, the random variable $\boldsymbol{\epsilon}$ is sampled from a Gumbel distribution, which is a continuous distribution on the simplex that can approximate categorical samples:

$$\epsilon = -\log(-\log(u)), u \sim \text{Uniform}(0, 1). \tag{4.22}$$

In Equation 4.21 the argmax operation is not differentiable, so softmax function is used as an approximation:

$$h^{(i)} = \frac{\exp((\log(\pi_k^{(i)} + \epsilon_k^{(i)})/\tau)}{\sum_{j=1}^{2} \exp((\log(\pi_k^{(i)}) + \epsilon_k^{(i)})/\tau)}, k = 1, 2 \tag{4.23}$$

where $\tau > 0$ is the temperature that modulates the re-sampling process. Finally, $h^{(i)}$ is transformed into binary mask $\mathbf{s}^{(i)}$ through the one_hot function.

The $\pi_k^{(i)}$ is jointly learned by deep neural networks in our models, and formulated as the parameters $\boldsymbol{\alpha} = (\pi_k^i)|_{k=1,2}^{i=1..n}$, conditioned on the concatenated feature vectors $[\mathbf{a_1}; \mathbf{a_2}]$ from two modalities:

$$\boldsymbol{\alpha} = \text{ReLU}(\text{FC}([\mathbf{a_1}; \mathbf{a_2}])), \tag{4.24}$$

where FC is full-connected layer, to map concatenated features into $k * 2$ dimensional class vectors, and ReLU is to impose nonlinearity and ensure the class vectors to be nonnegative.

In our approach, we find that modulating the temperature with respect to the training procedure can enable better performance in selective sensor fusion. This is because the temperature determines the samples and gradients: when the temperature is high, the variance of the gradients is small, while the samples are more smooth; at low temperatures, the variance of the gradients is high, while the samples are more discrete, which means it will fit well into the discrete distribution of the fusion mask. Thus we start the temperature from a higher value, i.e. 1 in our case, and gradually decrease it towards 0.5 over each epoch of the training process.

4.4.4 Discussions on Neural and Classical Sensor Fusion

In essence, soft fusion gently re-weights each feature in a deterministic way, while hard fusion directly blocks features according to the environment and its reliability. In general, soft fusion is a simple extension of direct fusion that is good for dealing with the uncertainties in the input sensory data. By comparison, the inference in hard fusion is more difficult, but it offers a more intuitive representation. The stochasticity gives the multimodal system better generalisation ability and higher tolerance to imperfect sensory data. The stochastic mask of hard fusion acts as an inductive bias, separating the feature selection process from prediction, which can also be easily interpreted by corresponding to uncertainties of the input sensory data.

Classical sensor fusion strategies normally rely on the handcrafted physical models and algorithms. For example, in the case of visual-inertial odometry, filtering methods update their belief based on the past state and current observations of visual and inertial modalities [60, 13, 195, 61]. ”Learning” within these methods is usually constrained to gain and covariances [196]. This is a deterministic process, and noise parameters are hand-tuned beforehand. Deep leaning methods are instead fully learned from data and the hidden recurrent state only contains information relevant to the regressor. Our approach models the feature selection process explicitly with the use of soft and hard masks. Loosely, the proposed soft mask can be viewed as similar to tuning the gain and covariance matrix in classical filtering methods, but based on the latent data representation instead.

4.5 Experiments

Our proposed selective sensor fusion is employed on three different tasks: vision-depth based global relocalization (task 1), deep LIDAR-vision odometry (task 2), and deep visual-

inertial odometry (task 3). We show that our proposed framework is not restricted into specific modality or task. Moreover, we investigate the robustness of neural models under data degradation by generating data degradation above public dataset. In addition, our selective sensor fusion offers an interpretation of the fusion process via a visualization of the soft/hard fusion mask.

4.5.1 Experimental Setups

We conducted extensive experiments above four well-known public datasets to learn from different pairs of sensor modalities: the 7-Scenes dataset [197] for vision-depth based relocalization, the KITTI odometry dataset [165] for vision-point cloud based odometry estimation, the KITTI raw dataset [165] and the EuRoC MAV dataset [166] for vision-IMU based odometry estimation. Our frameworks were implemented with PyTorch and trained on a NVIDIA Titan X GPU.

As the main focus of this work is a study of the general multimodal fusion problem, we want to investigate the performance of the two selective sensor fusion strategies compared to pre-existing models. In each task, we always choose a deep vision-only model and a deep multimodal model with direct fusion as the *common baselines*. Additionally, specific representative works were chosen as *task baselines*, according to each specific task.

All of our networks including common baselines were trained with a batch size of 16 using the Adam optimizer, with a learning rate lr $= 1e^{-4}$, for a fair comparison. All model were trained for 100 epochs, and the sequence length is chosen as 5.

4.5.1.1 Common Baselines

Common baselines share the same basic framework as our proposed SelectFusion framework. For a fair comparison, the hyper-parameters of our proposed network and of the common baselines are identical, including batch size, learning rate, and the dimension of hidden states. The vision-only model is composed of the same visual encoder, temporal modelling and task solver modules as our framework. The multimodal model with direct fusion uses the same structure as our proposed framework, except for the fusion component, which is a simple concatenation of the multimodal features. The single modality model and multimodal model with direct fusion can be viewed as ablated variants of our proposed approach.

4.5.1.2 Vision-Depth Relocalization

7-Scenes Dataset (vision+depth): The 7-Scenes dataset [197] contains RGB images and depth data captured by a handheld Microsoft Kinect camera from seven indoor scenarios. Each scene provides several sequences, and each sequence is with 500-1000 frames of colour and depth images. It has been widely used as a common benchmark for camera relocalization. We follow the official data split to train and test our models above this dataset for global pose estimation. This dataset is used to learn relocalization from vision and depth images.

Task Baselines: Our SelectFusion model is built as an end-to-end relocalization model, and thus we compared with PoseNet [81], LSTM-Pose [110] and VidLoc [83], which are representative techniques within the category of learning techniques. We also compare with DSO (Direct Sparse Odometry) [198], to show how relocalization, not being affected the problem of cumulative error drift, compares to visual odometry approaches.

4.5.1.3 LIDAR-Vision Odometry

KITTI Odometry Dataset (vision+LIDAR) The KITTI Odometry dataset [165] provides 11 sequences (00-10) with visual images, LIDAR point cloud and groundtruth. It has been extensively adopted as VO/SLAM benchmark. We use this dataset to fuse the visual and point cloud data to estimate relative pose (odometry) and reconstruct trajectory. Sequences *00, 01, 02, 03, 04, 06, 08, 09* are used for training DNN models, while the rest Sequences *05*, *07*, and *10* are relatively long and used for evaluation. The images are resized to 512×256. The challenges with this dataset are the relatively low frame rate (10Hz) for image data, the presence of many dynamic objects, high car speeds of up to 90 km/h and changing lightning conditions due to strong shadows cast by buildings and trees. We use this dataset to train deep vision-LIDAR odometry.

Task Baselines: We used three representative works that have been evaluated and widely adopted on the KITTI odometry benchmark, as our task baselines, i.e. VISO2_M [199], LOAM (LIDAR Odometry and Mapping)[69]. VISO2_M is a monocular VO algorithm, in which a fixed camera height, (i.e. a predefined 1.7 meters in the KITTI dataset) is given to recover the absolute scale of generated trajectories. As V-LOAM is no longer open-sourced, we adopted LOAM[69], the state-of-the-art LIDAR odometry algorithm.

4.5.1.4 Visual-Inertial Odometry

KITTI RAW dataset (visual+inertial) The KITTI Raw dataset [165] contains the raw data collection from car-driving scenarios. High-frequency inertial data (100 Hz) is only

available in the raw unsynchronized data package. We manually synchronized inertial data and images according to their timestamps, in order to exploit the visual and inertial data to learn odometry estimation. We used Sequences *00, 01, 02, 04, 06, 08, 09* for training and tested the network on Sequences *05*, *07*, and *10*, excluding sequence 03 as the corresponding raw file is unavailable. The images and ground-truth provided by GPS are collected at 10 Hz, while the IMU data is at 100 Hz. This dataset is adopted to learn visual-inertial odometry.

EuRoC MAV dataset (visual+inertial) The EuRoC dataset [166] contains tightly synchronized video streams from a Micro Aerial Vehicle (MAV), carrying a stereo camera and an IMU, and is composed by 11 flight trajectories in two environments, exhibiting complex motion. We used Sequence *MH_05_difficult* for testing, and left the other sequences for training. We downsampled the images and IMUs to 10 Hz and 100 Hz respectively. This dataset is used for training deep visual-inertial odometry model.

Task Baselines: We chose three state-of-the-art VIO pipelines, i.e. MSCKF [195], OKVIS [200] and mono-VINS[16] as task baselines to compare with our deep VIOs: MSCKF [195] is an Extended Kalman Filter based solution; OKVIS [200] is a keyframe based VIO with sliding window nonlinear optimization; mono-VINS[16] uses sliding window nonlinear optimization and IMU preintegration.

4.5.2 Task 1: Global Relocalization using Vision and Depth

We first employ selective sensor fusion to combine visual and depth information for a global localization task in indoor scenarios. The features are extracted from RGB and depth images using the visual and depth feature encoders discussed in Section 4.3. Our models, including the common baseline (direct fusion), are trained and evaluated on the 7-Scenes dataset. For each scene, we follow the official data split to train and test our models, and report for each model the median position and orientation error, according to the convention of prior works [81, 110, 83]. Note that relocalization indicates the ability to localize in a scene that has been visited before, so the training and testing scenes are overlapped, but the testing data have never been seen in the training set.

Table 4.1 shows the results for the common baseline (Direct Fusion) and our proposed SelectFusion frameworks, i.e. soft fusion (Soft (Ours)) and hard fusion (Hard (Ours)). For a fair comparison, the only difference in the three models is the feature fusion part. Clearly, our proposed SelectFusion strategies outperform the common baseline (direct fusion), and in particular hard fusion further improves the performance of the direct fusion with a gain

Table 4.1: The results of vision-depth relocalization (Task 1) on the 7-Scenes dataset, reported in position error (m) and orientation error (°)

Scene	PoseNet	LSTM-Pose	DSO	VidLoc (V)	VidLoc(V+D)	Direct Fusion	Soft (Ours)	Hard (Ours)
Chess	0.32 m, 8.12	0.24 m, 5.77	0.17 m, 8.13	0.18 m, NA	0.16 m, NA	0.16 m, 5.30	0.15 m, 5.46	0.14 m, 5.02
Fire	0.47 m, 14.4	0.34 m, 11.9	0.19 m, 65.0	0.21 m, NA	0.19 m, NA	0.26 m, 10.2	0.28 m, 10.3	0.26 m, 9.80
Heads	0.29 m, 12.0	0.21 m, 13.7	0.61 m, 68.2	0.14 m, NA	0.13 m, NA	0.16 m, 12.5	0.15 m, 12.1	0.15 m, 12.4
Office	0.48 m, 7.68	0.30 m, 8.08	1.51 m, 16.8	0.26 m, NA	0.24 m, NA	0.24 m, 6.78	0.22 m, 6.79	0.23 m, 6.39
Pumpkin	0.47 m, 8.42	0.33 m, 7.00	0.61 m, 15.8	0.36 m, NA	0.33 m, NA	0.22 m, 5.10	0.21 m, 4.97	0.21 m, 4.93
Red Kitchen	0.59 m, 8.64	0.37 m, 8.83	0.23 m, 10.9	0.31 m, NA	0.28 m, NA	0.25 m, 6.41	0.26 m, 6.36	0.25 m , 6.76
Stairs	0.47 m, 13.8	0.40 m, 13.7	0.26 m, 21.3	0.26 m, NA	0.24 m, NA	0.37 m, 11.8	0.35 m, 11.9	0.30 m, 11.3
Average	0.44 m, 10.4	0.31 m, 9.85	0.26 m, 29.4	0.25 m, NA	0.23 m, NA	0.24 m, 8.30	0.23 m, 8.27	0.22 m, 8.08

of 8.33% in the position and 2.65% in the orientation. This shows the effectiveness of SelectFusion in learning multimodal representation for global relocalization.

In addition to the common baseline, we also choose four representative visual localization approaches as task baselines, i.e. PoseNet [81], LSTM-Pose [110], DSO [198] and VidLoc [83]. In Table 4.1, we report the results of task baselines from their original works for a more convincing comparison. VidLoc can be viewed as a simple direct fusion, but it uses full-size images, and different feature encoders. Clearly, our proposed hard fusion model outperform these learning based relocalization models (i.e. PoseNet [81], LSTM-Pose [110], and VidLoc [83]) on the common benchmark. Our methods also shows better performance than DSO [198], indicating that relocalization model is more suitable than visual odometry approach when applied in a familiar place. Although the hyperparameters in task baselines are different from our models, it still demonstrates that our models can achieve competitive performance over previous works using only the uniform framework and proposed fusion strategies

4.5.3 Task 2: Deep LIDAR-Vision Odometry

We now focus on the problem of learning LIDAR-vision odometry in a car-driving scenario. It is achieved by extracting effective features from point cloud data and RGB images for relative pose estimation. The pointcloud feature encoder and visual feature encoder (FlowNet-style) are used to process raw pointcloud data and RGB images respectively. Our proposed selective sensor fusion framework can be naturally extended to this task to automatically select useful features. The models are trained on the KITTI Odometry dataset and tested on three new sequences, i.e. Sequence 05, 07 and 10. Then the relative poses produced by the neural networks are integrated into global trajectories, which are further evaluated according to the official KITTI VO/SLAM evaluation metrics [165]. This metric is calculated by averaging the Root Mean Square Errors (RMSEs) of the translation and rotation for all the subsequences of lengths (100,..., 800) meters.

Table 4.2: The results of lidar-vision odometry (Task 2) on the KITTI Odometry dataset

Seq.	VISO2_M		LOAM		Vision Only		LIDAR Only		Ours (V+L)		Ours (Soft)		Ours (Hard)	
	$t_{rel}(\%)$	$r_{rel}(^\circ)$	$t_{rel}(\%)$	$r_{rel}(^\circ)$	$t_{rel}(\%)$	$r_{rel}(^\circ)$	$t_{rel}(\%)$	$r_{rel}(^\circ)$	$t_{rel}(\%)$	$r_{rel}(^\circ)$	$t_{rel}(\%)$	$r_{rel}(^\circ)$	$t_{rel}(\%)$	$r_{rel}(^\circ)$
05	19.22	17.58	0.75 (0.57)	0.38	4.74	1.89	9.55	3.60	4.73	1.82	4.65	1.83	4.25	1.67
07	23.61	29.11	0.69 (0.63)	0.50	8.27	3.30	8.63	3.75	4.31	2.34	4.36	2.19	4.46	2.17
10	41.56	32.99	1.51 (0.79)	0.57	9.18	1.89	15.59	4.77	5.92	1.73	8.35	2.01	5.81	1.55
Ave.	28.13	26.66	0.98 (0.66)	0.48	7.40	2.36	11.26	4.04	4.99	1.96	5.78	2.01	4.84	1.80

- $t_{rel}(\%)$ is the average translational RMSE drift (%) on lengths of 100m-800m.
- $r_{rel}(^\circ)$ is the average rotational RMSE drift (°/100m) on lengths of 100m-800m.
- The Vision-Only, Lidar Only, Ours (V+L), Ours (Soft), and Ours (Hard) models are trained on Sequence 00, 01, 02, 03, 04, 06, 08 and 09 with same hyperparameters for a fair comparison.

Table 4.2 shows the results of our deep LIDAR-vision odometry on the KITTI Odometry dataset. Vision Only and LIDAR Only models represent the model using only vision or LIDAR data to estimate ego-motion. Compared with them, fusing vision and LIDAR features (Ours (V+L)) contributes to a large improvement no matter in translation or rotation. Ours (V+L), Ours (Soft) and Ours (Hard) are our frameworks with a naive direct fusion, soft fusion and hard fusion. Our proposed hard fusion is capable of improving the performance over the naive fusion model about 3.0% in translation and 8.2% in orientation. Note that these models are built on the same modules, except the feature fusion part for a fair comparison. Meanwhile, two classical methods, i.e. VISO_M (Monocular Visual Odometry) [199] and LOAM (LIDAR Odometry and Mapping) [69] are chosen to compare with our data-driven approaches. As we can see, the learning based methods greatly outperform monocular visual odometry, but still have a certain gap with the state-of-the-art LIDAR odometry (LOAM). The model based methods are tailored to the specific visual odometry or LIDAR odometry problem: LOAM is built on scene geometry information and quite accurate with the good-quality pointcloud data; the monocular visual odometry (VISO_M) relies on hand-crafted features and is quite challenging above high-dimensional images. In comparison, the data-driven models can automatically extract suitable features, which means that they are not restricted into specific sensor modality or task, leaving potentials to explore an universal framework for deep states estimator.

4.5.4 Task 3: Deep VIO on UAV and self-driving scenarios

Finally, we come to evaluate our proposed model on the KITTI raw dataset (self-driving scenario) and EuRoC MAV dataset (UAV scenario) on learning visual-inertial odometry (VIO), a fundamental research problem in the robotic community. These two datasets are challenging, as some real-world sensor degradations are contained in the original data: in

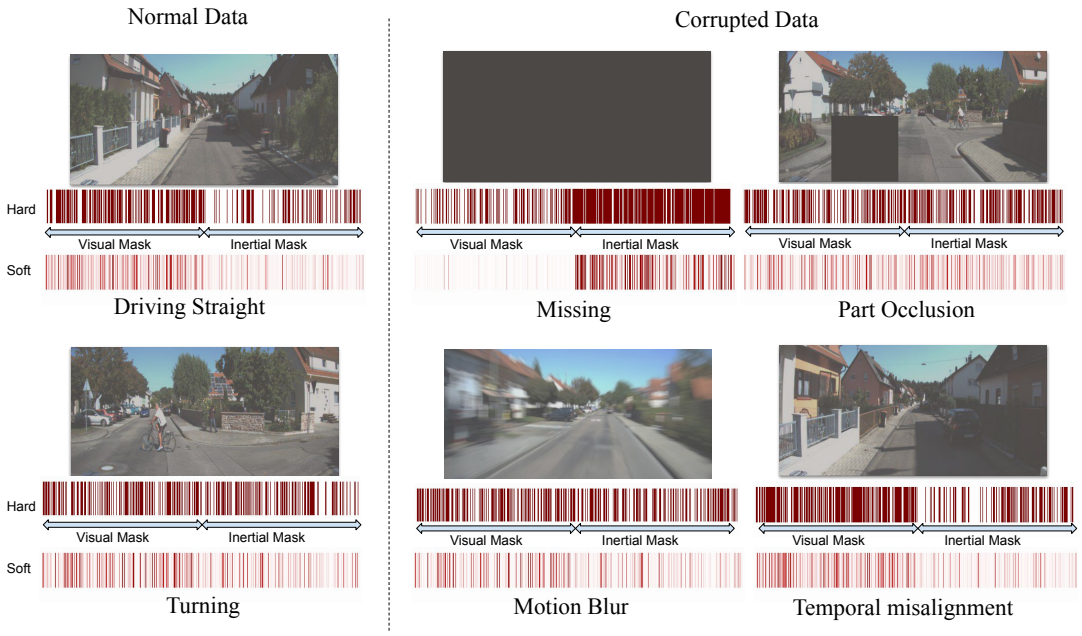


Figure 4.7: Visualization of the learned hard and soft fusion masks under different conditions for Task 3 Deep VIO on self-driving scenarios (left: normal data; middle and right: corrupted data). The number (hard) or weights (soft) of selected features in the visual and inertial sides can reflect the self-motion dynamics (increasing importance of inertial features during turning), and data corruption conditions.

the KITTI dataset, some IMU data are missing for a number of timesteps; IMU and camera streams are not tightly time-synchronized, which causes temporal sensor degradation; in addition, there are moving vehicles which act to partially occlude the camera; also in the Euroc dataset, there is significant motion blur and camera occlusion. Except the real sensor degradations, we also generate synthetic data degradations above the public datasets to study the robustness of learning models.

4.5.4.1 Synthetic Data Degradation

In order to provide an extensive study of the effects of sensor data degradation and to evaluate the performances of the proposed approach, we generate a degraded dataset, as shown in Figure 4.7, by adding various types of noise and occlusion to the original data, as described in the following.

1) **Vision Degradation.** In order to simulate the effects of occlusions, motion blur and missing frames that commonly affect video streams, we corrupt input images in three ways:

Occlusions: we overlay a mask of dimensions 128×128 pixels on top of the sample images, at random locations for each sample. Occlusions can happen due to dust or dirt on the sensor or stationary objects close to the sensor [201].

Motion Blur: we introduce motion blur to represent the camera blur caused by fast ego-motion or fast object movements. This motion blur is generated by estimating the relative motion of the scene, and producing corresponding blur above original images. Motion blur can happen when the camera or the light condition changes substantially [202].

Missing data: we randomly remove 10% of the input images. This can occur when packets are dropped from the bus due to excess load or temporary sensor disconnection. It can also occur if we pass through an area of very poor illumination e.g. a tunnel or underpass.

2) **IMU Degradation.** In order to simulate the effect of large unmodelled noise in inertial data, as well as missing samples, we degrade the inertial data in the following ways:

Noise+bias: on top of the already noisy sensor data we add additive white noise to the accelerometer data and a fixed bias on the gyroscope data. This can occur due to increased sensor temperature and mechanical shocks, causing inevitable thermo-mechanical white noise and random walking noise [203].

Missing data: we randomly remove windows of inertial samples between two consecutive random visual frames. This can occur when the IMU measuring is unstable or packets are dropped from the bus.

3) **Cross-Sensor Degradation.** We also model the two misalignment issues that commonly affect visual-inertial systems:

Spatial misalignment: we randomly alter the relative rotation between the camera and the IMU, compared to the initial extrinsic calibration. This can occur due to axis misalignment and the incorrect sensor calibration [13]. We uniformly model up to 10 degrees of misalignment .

Temporal misalignment: we apply a time shift between windows of input images and windows of inertial measurements. This can happen due to relative drifts in clocks between independent sensor subsystems [204].

4.5.4.2 Experiment on the EuRoC Dataset

In the experiment of EuRoC dataset, we report the root mean squared error (RMSE) of the absolute translation error (ATE) of our models and baselines. This evaluation metric is commonly adopted by previous classical VIO works[195, 200, 16], so that our proposed frameworks can be compared with them directly. Table 4.3 reports the performance of learning models (i.e. Vision-Only (DeepVO), VIO Direct (VINet)), and classical algorithms (i.e. MSCKF[195], OKVIS[200] and mono-VINS[16]) on the trajectory

Table 4.3: The results (m) of deep visual-inertial odometry (Task 3) on the EuRoC dataset (UAV scenario).

	Original	Vision Degrad.	All Degrad.
MSCKF	0.48	30.37	fail
OKVIS	0.47	1.42	fail
mono-VINS	0.35	fail	fail
Vision Only	2.42	2.44	1.99
VIO Direct	0.99	1.14	1.15
VIO Soft	1.06	1.18	1.21
VIO Hard	0.84	1.04	1.12

- The results (m) are reported the root mean squared error (RMSE) of the absolute translation error (ATE).
- The Vision-Only, VIO Direct, VIO Soft, and VIO Hard models are trained on the sequences except MH_05_difficult of EuRoC MAV dataset [166] with same hyperparameters for a fair comparison, and tested on Sequence MH_05_difficult.

MH_05_Difficult in presence of normal data, all combined visual degradation (10% occlusion, 10% motion blur, and 10% missing data) and all combined visual+inertial degradation (5% for each). Note that learning models share the same framework and hyper-parameters, while the only difference is the fusion strategy. Details of data degradations can be found at the Section 4.5.4.1.

We can see that hard fusion consistently outperforms other learning models in all three scenarios, demonstrating the effectiveness of our proposed fusion strategy. In the normal set, hard fusion improves the direct fusion (our common baseline) by 15.15% in ATE. Another notable point is that OKVIS only shows a large performance decrease in the vision degradations, and fails on the all degradations, while mono-VINS fails on both degradation scenarios. In contrast, learning models all can work on degradation scenarios. This indicates that learning models are capable of overcoming sensor degradations to perform more robustly. Learning models still can not compete with the classical algorithms in normal set. This is due to two reasons: 1) in the EuRoC dataset, the groundtruth values of motion tracking (from a Vicon system) and sensor data (from a UAV) are collected from two time systems, and hence the training of learning models is limited because of the probable errors on ground-truth labels; 2) this deep VIO framework is still a basic framework, while extensions and constraints relevant to specific properties of visual and inertial sensors can be added onto it to further enhance the performance, e.g. Bundle adjustment.

Table 4.4: The results of visual-inertial odometry (Task 3) on the KITTI Raw dataset (car-driving scenario)

Seq.	MSCKF	Vision Only	VIO (Direct)	VIO (Soft)	VIO (Hard)
05	19.0%, 82.5°	6.14%, 2.84°	4.18%, 1.57°	4.44%, 1.69°	4.11%, 1.49°
07	89.9%, 126°	6.41%, 2.76°	3.39%, 1.79°	2.95%, 1.32°	3.44%, 1.86°
10	42.0%, 134°	6.93%, 2.97°	2.80%, 1.69°	2.85%, 1.22°	1.51%, 0.91°
Ave.	50.3%, 114°	6.49%, 2.85°	3.45%, 1.69°	3.41%, 1.41°	3.02%, 1.42°

- $t_{rel}(\%)$ is the average translational RMSE drift (%) on lengths of 100m-800m.
- $r_{rel}(^\circ)$ is the average rotational RMSE drift (°/100m) on lengths of 100m-800m.
- The Vision-Only, VIO Direct, VIO Soft, and VIO Hard models are trained on Sequence 00, 01, 02, 04, 06, 08 and 09 of KITTI raw dataset [165] with same hyperparameters for a fair comparison, and tested on Sequence 05, 07 and 10.

4.5.4.3 Experiment on the KITTI Dataset

On the KITTI dataset, we used the official KITTI VO/SLAM evaluation metrics[2] to evaluate our proposed VIO models and other baselines, the same as the evaluation in LIDAR-vision odometry. This metric is to calculate the averaged RMSE of the translation and rotation for all the sub-sequences of lengths (100, ..., 800) meters, which can reflect both the global and local drifts of odometry estimation.

Table 4.4 reports the performance of our proposed hard fusion and soft fusion on the trajectories 05, 07 and 10 of normal dataset, together with a classical VIO algorithm i.e. MSCKF and two learning models, i.e. vision only (DeepVO) and VIO direct (VINet). Our proposed selective sensor fusion (hard) further improves the averaged performance of the direct fusion by 12.46% in translation and 15.98% in orientation, while soft fusion shows an improvement of 1.16% and 16.57% in translation and rotation. Due to the real issue of bad time synchronization between images and IMUs, OKVIS and mono-VINS failed on the KITTI raw dataset. We then compare with a MSCKF implementation based on [195][3]. This approach, differently from OKVIS or VINS-Mono, is based on a trifocal tensor matching between triplets of successive frames, in order to get an ego-motion estimate, which is then fused with inertial information via a multi-state constraint Kalman filter to refine the estimates of the camera poses for each triplet. This sliding approach arguably makes it more robust to IMU de-synchronisation. It is clear to see that the learning based VIO models, including VIO (direct), VIO (soft) and VIO (hard) outperform MSCKF by a large margin. This is because MSCKF is limited by the bad time synchronization of two sensors,

[2]http://www.cvlibs.net/datasets/kitti/eval_odometry.php

[3]The code can be found at https://uk.mathworks.com/matlabcentral/ fileexchange/43218-visual-inertial-odometry

Table 4.5: The results of deep visual-inertial odometry (Task 3) on the KITTI dataset (autonomous driving scenario)

	Original	Vision Degrad.	All Degrad.
Vision Only	6.49%, 2.85°	11.8%, 3.53°	8.06%, 3.18°
VIO Direct	3.45%, 1.69°	5.06%, 1.29°	3.62%, 1.28°
VIO Soft	3.41%, 1.41°	4.39%, 1.84°	3.49%, 1.40°
VIO Hard	3.02%, 1.42°	4.76%, 1.12°	3.27%, 1.29°

- $t_{rel}(\%)$ is the average translational RMSE drift (%) on lengths of 100m-800m.
- $r_{rel}(^\circ)$ is the average rotational RMSE drift (°/100m) on lengths of 100m-800m.
- The Vision-Only, VIO Direct, VIO Soft, and VIO Hard models are trained on Sequence 00, 01, 02, 04, 06, 08 and 09 of KITTI raw dataset [165] with same hyperparameters for a fair comparison, and tested on Sequence 05, 07 and 10.

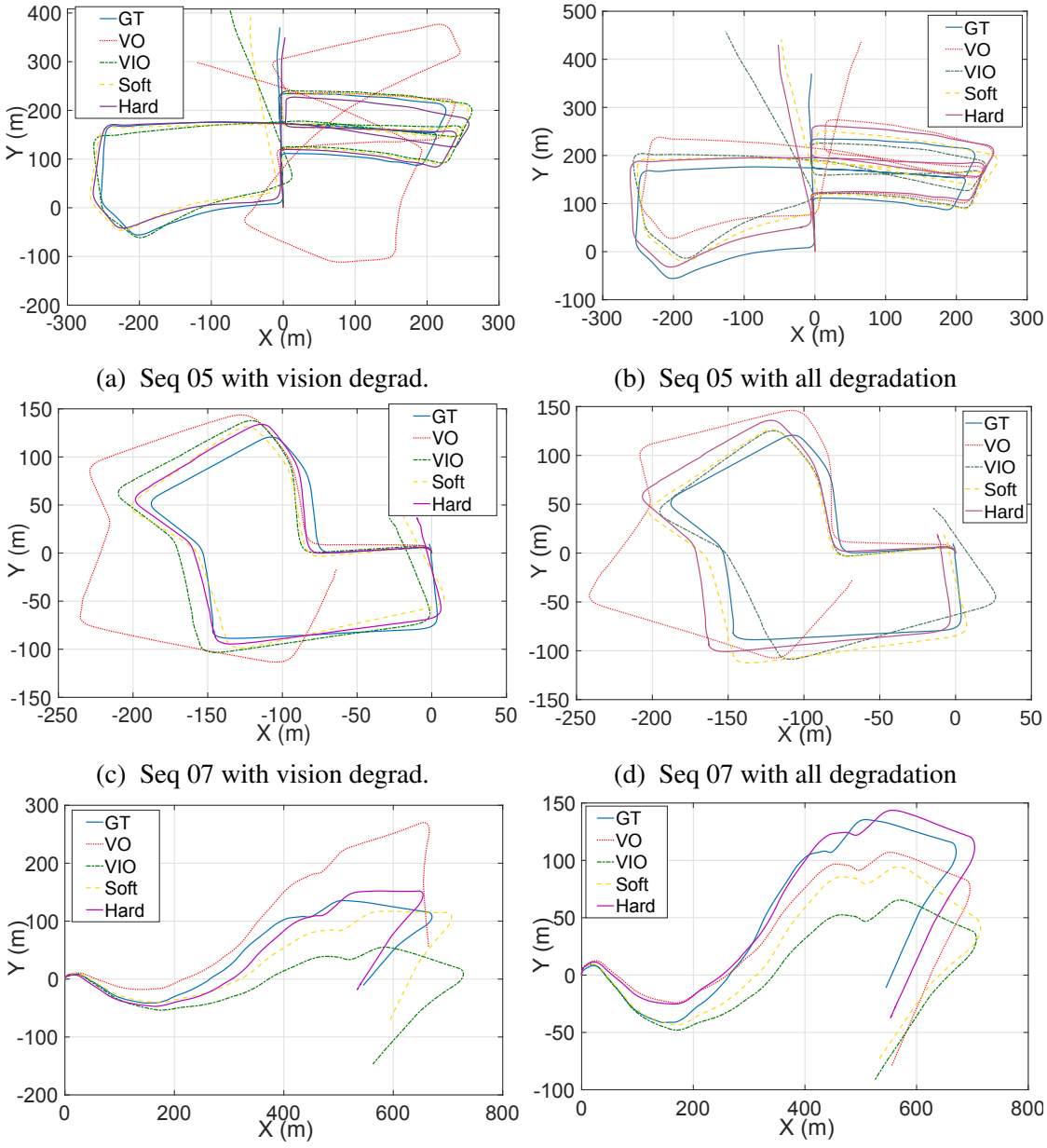
Table 4.6: Effectiveness of different sensor fusion strategies in presence of different kinds of sensor data corruption for deep VIO

	Vision Degradation			IMU Degradation		Sensor Degradation	
Model	Occlusion	Blur	Missing	Noise and bias	Missing	Spatial	Temporal
Vision Only	7.23%, 2.81°	7.76%, 2.59°	27.6%, 9.20°	6.49%, 2.85°	6.49%, 2.85°	6.49%, 2.85°	6.49%, 2.85°
VIO Direct	4.24%, 1.77°	4.28%, 1.85°	5.61%, 1.32°	3.74%, 1.30°	3.59%, 1.74°	4.12%, 2.00°	3.27%, 1.55°
VIO Soft (Ours)	3.85%, 1.59°	3.82%, 1.48°	6.42%, 2.02°	3.72%, 1.20°	3.50%, 1.59°	3.45%, 1.46°	3.43%, 1.72°
VIO Hard (Ours)	3.77%, 1.74°	3.75%, 1.33°	5.45%, 1.26°	3.16%, 1.64°	3.18%, 1.47°	3.22%, 1.57°	3.20%, 1.31°

- $t_{rel}(\%)$ is the average translational RMSE drift (%) on lengths of 100m-800m.
- $r_{rel}(^\circ)$ is the average rotational RMSE drift (°/100m) on lengths of 100m-800m.
- The Vision-Only, VIO Direct, VIO Soft, and VIO Hard models are trained on Sequence 00, 01, 02, 04, 06, 08 and 09 of KITTI raw dataset [165] with same hyperparameters for a fair comparison, and tested on Sequence 05, 07 and 10.

whilst the learning models are generally more robust to overcome such data degradations caused by real-world issues (data collection).

Table 4.5 and 4.6 show the performance of the proposed data fusion strategies, compared with the common baselines on the KITTI raw dataset. In particular, we compare with a DeepVO [85] (Vision-Only) implementation, and finally with an implementation of VINet [89] (VIO Direct), which uses a naïve fusion strategy by concatenating visual and inertial features. In the vision degraded set the input images are randomly degraded by adding occlusion, motion blurring and removing images, with 10% probability for each degradation. In the full degradation set, images and IMU sequences from the dataset are corrupted by all seven types of degradation with a probability of 5% each. Table 4.5 reports the results of deep VIO models in the presence of combined visual degradations, and all degradations. As we can see, our proposed selective sensor fusion, especially hard fusion,



(a) Seq 05 with vision degrad.

(b) Seq 05 with all degradation

(c) Seq 07 with vision degrad.

(d) Seq 07 with all degradation

(e) Seq 10 with vision degrad.

(f) Seq 10 with all degradation

Figure 4.8: Estimated trajectories on the KITTI dataset for Task 3 deep visual-inertial odometry (VIO). Left column: dataset with vision degradation (10% occlusion, 10% blur, and 10% missing data); right column: data with all degradation (5% for each). Here, GT, VO, VIO, Soft and Hard mean the ground truth, neural vision-only model, neural visual inertial models with direct, soft, and hard fusion. The neural VIO models with our proposed soft and hard selective masks showed better performance in terms of accuracy and robustness, compared with direct fusion and vision only model.

achieved better performance than the learning models without our proposed fusion module in these sensor degradation scenarios. In each data degradation, as illustrated in Table 4.6, our proposed selective sensor fusion, especially hard fusion consistently outperforms other baselines. This demonstrates to what extent our proposed SelectFusion can tolerant the perturbations of each data degradation, showing that SelectFusion is more robust to the issues caused by data degradations compared with other baselines.

Figure 4.8 shows a visual comparison of the resulting three test trajectories (Seq 05, Seq 07, Seq 10) in presence of visual and combined kinds of degradation. We compare the two VO and vanilla VIO baselines with the proposed soft and hard fusion strategies. It can be noticed how, while at start VIO performs as well as soft and hard fusion, on average, over time the proposed selective fusion strategies outperform the vanilla fusion, since the increased robustness reduces error accumulation. This is particularly visible in the most challenging Seq 05. As expected, a VO approach heavily underperforms in presence of large amounts of angular rotations (Seq 05, Seq 07). Another interesting result is how VO performs slightly better in presence of IMU degradation and camera-IMU synchronization (Figure 4.8b and 4.8f). That shows how a vanilla VIO fusion is unable to deal with these issues, to the point of underperforming compared to vision-only approaches. This result further corroborates the fact that in deep learning-based approaches explicitly learning the belief on the different components makes the estimation more robust, while stacking sensors without a sensible fusion strategy can lead to catastrophic fusion, similarly to traditional approaches. Catastrophic fusion happens when the single components of the system before fusion significantly outperform the overall system after fusion.

4.5.5 Interpretation of Selective Fusion

Incorporating the hard mask into our framework enables us to quantitatively and qualitatively interpret the fusion process. Firstly, we analyse the contribution of each individual modality in different scenarios for deep visual-inertial odometry (Task 3). Since hard fusion blocks some features according to their reliability, in order to interpret the ”feature selection” mechanism we simply compare the ratio of the non-blocked features for each modality. Figure 4.9 shows that visual features dominate compared with inertial features in most scenarios. Non-blocked visual features account for more than 60%, underlining the importance of this modality. We see no obvious change when facing small visual degradation, such as image blur, because the FlowNet extractor can deal with such disturbances. However, when the visual degradation becomes stronger the role of inertial features becomes significant. Notably, the two modalities contribute equally in presence of occlusion.

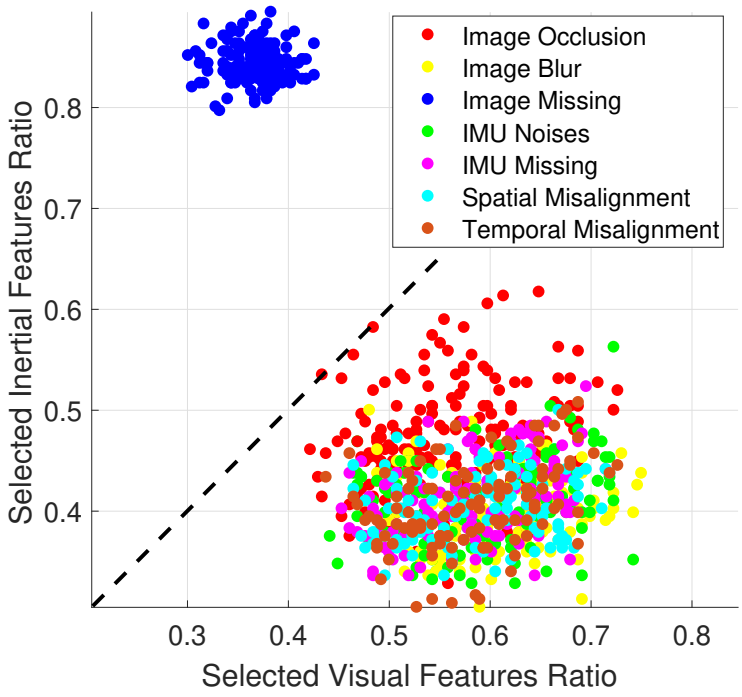


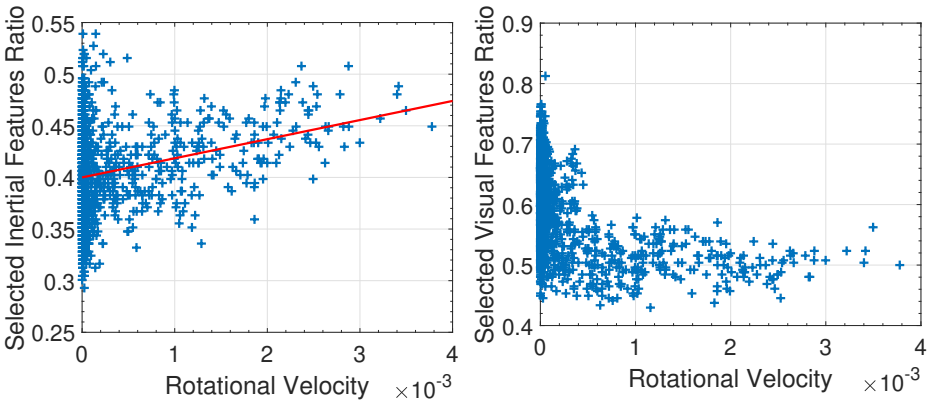
Figure 4.9: A comparison of visual and inertial features selection rate in seven data degradation scenarios for Task 3.

As it would be expected, inertial features dominate (by more than 90%) with missing images.

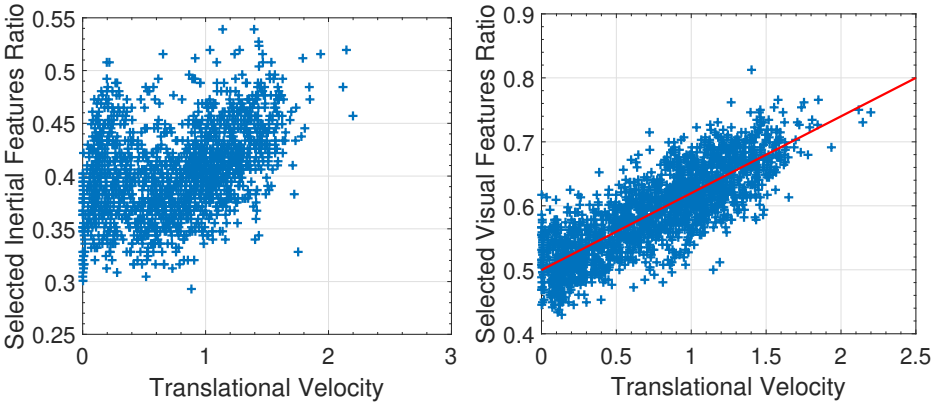
In Figure 4.10 we analyze the correlation between amount of linear and angular velocity and the selected features. These results also show how the belief on inertial features is stronger in presence of large rotations, e.g. turning, while visual features are more reliable with increasing linear translations. It is interesting to see that at low translational velocity (0.5m / 0.1s) only 50% to 60% visual features are activated, while at high speed (1.5m / 0.1s) 60 % to 75 % visual features are used.

4.6 Summary

We present a generic multimodal sensor fusion framework for deep state estimation, in support of odometry estimation and global relocalization tasks. Motivated by the need for robust interpretable sensor fusion in real-world applications, we proposed two variants of selective fusion modules, i.e. a deterministic soft fusion and a Gumbel-softmax based hard fusion, that can be integrated in different neural network frameworks. The proposed model is not restricted to a specific modality or task. It can learn to perform sensor fusion on



(a) Inertial-Rotation (b) Visual-Rotation



(c) Inertial-Translation (d) Visual-Translation

Figure 4.10: Task 3: Correlations between the number of inertial/visual features and amount of rotation/translation show that the inertial features contribute more with rotation rates, e.g. turning, while more visual features are selected with increasing linear velocity.

feature space from pairs of different modalities, e.g. vision-depth, vision-lidar and vision-inertial data, conditioned on the input data itself. Extensive experiments illustrate that our proposed models outperform single modality and multimodal models with direct fusion baselines, and also show competive performance over other classical approaches. In order to investigate the performance in various data degradation conditions, we extended two public datasets to include degraded and misaligned data streams, and study the influence of different modalities under different degradation and self-motion/environmental circumstances. In addition, we are able to provide insightful interpretations of fusion process by visualizing the learned masks.

Chapter 5

Sequential Invariant Domain Adaptation

Supervised learning normally requires a large amount of labelled data. Training deep neural networks on the data from a specific domain will likely cause the model to be overfitted in the given data distribution, and hence it typically performs poorly in a new domain. This is especially the case of learning based localization approaches, as a variety of issues, such as environmental factors and the changes of self-dynamics, alter the sensor measurements. Labelling data from all domains is not always possible, as it is time-consuming and costly. How to exploit an unlabelled stream of sensor data and improve generalization ability in a new environment is a practical question for the real world deployment of DNN-based localization models. This chapter introduces Sequential Invariant Domain Adaptation (SIDA), a generic domain adaptation framework to process long continuous sensory (or time series) data [118]. The intuition is to use a shared encoder to transform the sequence data into a domain-invariant hidden representation, with the aid of deep generative models to encourage the semantic representation obtained, whilst discarding the domain specific features. Our framework dramatically reduces the effort in converting the raw sensor data from another new domain into an accurate trajectory without the requirement of labelled data in new domains. To demonstrate its generality, we also show how it can be used for human activity recognition (HAR) tasks.

5.1 Introduction

Developing accurate inertial tracking is thus of key importance for robot/pedestrian navigation and for self-motion estimation [5]. Unlike other commonly used sensor modalities, such as GPS, radio and vision, inertial measurements are completely egocentric and as such

are far less environment dependent e.g. they work equally well in an unlit underground tunnel as in open spaces. In emergency rescue scenarios, it can help track firefighters and other first-responders to enhance the safety and efficiency of their operations without the need of any bespoke positioning infrastructure. However, the task of turning inertial measurements into pose and odometry estimates is hugely complicated by the fact that different placements (e.g. carrying a device in a pocket or in the hand) and orientations lead to significantly different inertial data in the sensor frame. It is clearly infeasible to collect labelled data from every possible domain (e.g. attachment), as this requires specialized motion capture systems e.g. VICON and a high degree of effort. Therefore, this chapter is aimed at proposing a robust domain adaptation network for sequential inertial data, which is able to directly learn to locate in unlabelled domains without using any paired sequences.

Prevailing inertial tracking methods, e.g. Strapdown Inertial Navigation System (SINS) [42] and Pedestrian Dead Reckoning (PDR) [154], are mostly based on delicate handcrafted models. These model-based approaches can obtain plausible performance in general scenarios, but their lack of generalisation ability yields poor performance in complex real world applications. Our work on neural inertial tracking discussed in Chapter 3 has demonstrated that deep neural networks are capable of extracting high level motion representations (displacement and heading angle) from raw IMU sequence data, and providing accurate trajectories. However, the data-driven method that requires substantial labeled data for training, and a model trained on a single domain-specific dataset may not generalise well to new domains [205]. As shown in Figure 5.1, the uncertainties of phone placements, the corresponding motion dynamics, and the projection of gravity significantly alter the inertial measurements acquired from different domains (sensor frames), while the actual trajectories in the navigation frame are identical.

We note that it is possible to train end-to-end deep neural networks when presented with large amounts of labelled data. The question becomes, how can we generalize to an arbitrary attachment (domain) in the absence of labels or a paired/time-synchronized sequence? Although from the observation the raw inertial data for each domain is very different, and the resulting odometry trajectories are also unrelated to one another, the underlying statistical distribution of odometry pose updates must be similar, if derived from a common agent (e.g. human motion). Our intuition is to decompose the raw inertial data into a domain-invariant semantic representation, learning to discard the domain-specific motion sequence transformation.

To overcome the challenges of generalising inertial tracking across different motion domains, we propose the **S**equential **I**nvariant **D**omain **A**daptation (**SIDA**) framework with

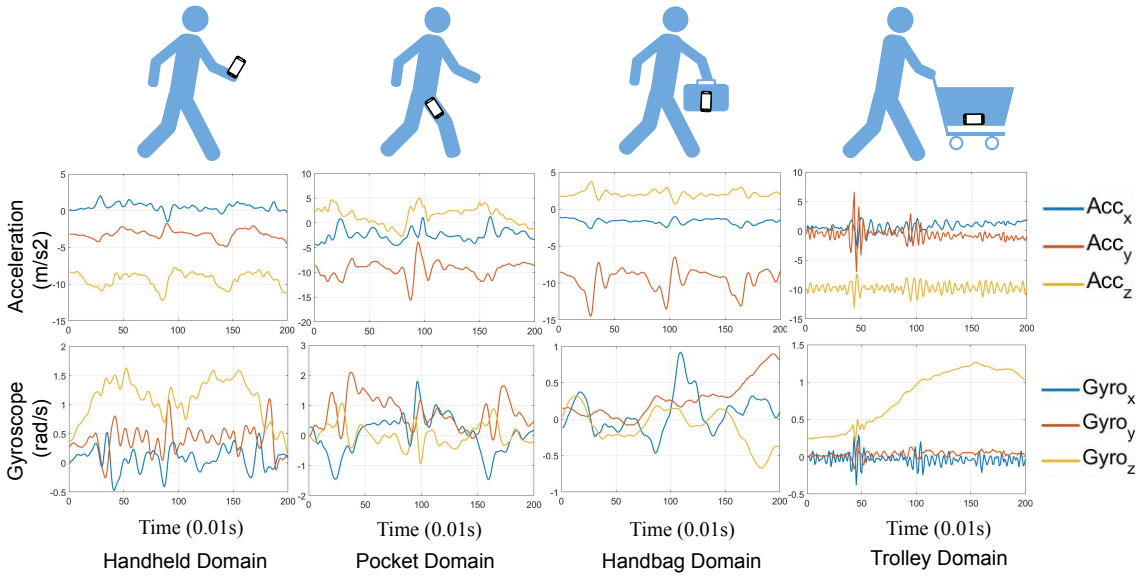


Figure 5.1: An illustration of the phone placements in the (a) hand, (b) pocket, (c) bag and (d) trolley, whilst a user is walking. Compared to firmly holding in the hand, the IMU experiences slight swings in pocket. The angular and linear accelerations in the navigation frame are projected on different axes in each sensor frame, and the gravity is mainly projected onto the y axis rather than z axis. These variations are termed motion domain shifts as the sensor frames are different yet the inertial data in the navigation frame is invariant, which imposes huge challenges on transferring a learned inertial tracking model to a new domain.

deep generative models (generative adversarial networks and autoencoders) for sensory sequence domain transformation. To deal with the situations where one domain (i.e. the source domain) has labelled data and the other (i.e. the target domain) does not, this framework is capable of leveraging the rich set of labels in the source domain and the wealth of data in both domains, as shown in Figure 5.2. Assuming that there exists an intermediate representation that is domain invariant (if the data in two domains are collected from a common agent), its key novelty is to use a shared encoder to transform raw inertial sequences into a domain-invariant hidden representation, without the use of any paired data. In doing so, it exploits the similarity of the motion distributions in two domains, and reduces the influence of the domain specific features, when predicting the labels in target domains. SIDA dramatically reduces the effort in converting raw inertial data to an accurate trajectory, as no labelled or even paired data is required to achieve motion transformation in new domains. Through extensive, real-world experiments, we demonstrate that the framework is able to efficiently and effectively transform an arbitrary domain into an accurate inertial trajectory, benefiting from the knowledge transferred from the labelled source domain.

In addition, we extend this framework to solve domain adaptation for human activity

recognition. Different from estimating inertial odometry, this task trains deep neural network to classify human activities from sensor data instead of regressing real values. The data distributions of sensor measurements are significantly different, if they are collected from two sensor attachments, e.g. in the hand or bag. In this case, SIDA is applied to mitigate the influences from the various sensor collections in new domains. It demonstrates that our proposed SIDA is not restricted to a specific task, but a generic domain adaptation approach that works well on time-series sensor data to solve both classification and regression problems.

In summary, the contributions of the Chapter 5 are as below:

- We propose Sequential Invariant Domain Adaptation (SIDA), a general framework to solve domain adaptation for time-series sensor data.
- By transferring the learned motion knowledge from the labelled source domain, SIDA is able to achieve inertial tracking and reconstruct physically meaningful trajectories in new unlabelled domains.
- Beyond inertial tracking, our work on extending SIDA to human activity recognition proves its capacity to tackle the domain shift problem of classification tasks in an unsupervised manner.
- Through visualizing the sequence encoder, we demonstrate that our method can extract domain invariant features from sequential data that benefit the task in target domains.

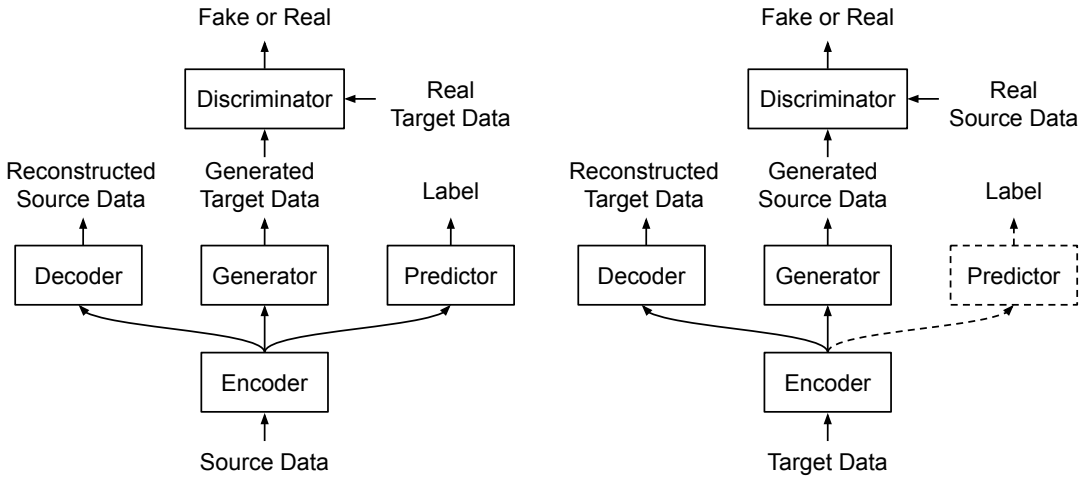
5.2 Related Work

This section discusses the previous work on *domain adaptation* and *inertial navigation systems*, relevant to the research in this chapter.

Domain Adaptation Our work is most related to domain adaptation techniques, which aim to align the learned representation across source and target domains by minimizing maximum mean discrepancy loss [206] or adversarial loss [207, 205]. Recent adversarial approaches have been achieved the state of art results in multiple tasks, for example, sleep stages prediction [208], healthcare data prediction [209] and image-to-image translation [210]. Prior art can be categorized into two main groups: the discriminative adversarial

models seek to align the embedding representation between target and source domain to encourage domain confusion [207, 211, 205]; the generative adversarial models aim to employ generated data for training the prediction networks and meanwhile fooling the discriminator [212, 103, 210]. Here, we utilize the generative adversarial networks (GAN) to generate sensory sequence data from invariant features extracted by an identical encoder. Different from many GAN-based sequence generation models applied in the field of natural language processing [213], where the sequences consist of discrete symbols or words (e.g. dialogue generation, poem generation and unsupervised machine translation) [214], our model focuses on transferring long sequences of continuous time series sensory data. It is worth mentioning that instead of using a conditional autoregressive decoder that takes whole source sequences as input and generates variable-length sequences (generally applied in sequence-to-sequence generation models), our framework is able to take the advantage of time consistency and directly produces the outputs in target domains aligned to the inputs in source domains in every time step.

Inertial Navigation Systems Early inertial navigation systems were developed as the core components in control and navigation systems for missiles, submarines, and spacecraft, relying on expensive, heavy and high-precision inertial measurement units [42]. The traditional strapdown inertial navigation algorithms are hard to realize on low-cost MEMS inertial sensors, because the high measurement noises cause exponential error propagation via open integration, and the inertial output collapses within seconds [5]. To mitigate the unbounded error drift, one solution is to combine cameras with inertial sensors as realized in visual inertial odometry [13]. Another solution is to detect steps and update the trajectory with estimated step length and heading through pedestrian dead reckoning [154]. These model based approaches exploit the context information to reduce the inertial systems drift, for example, via zero-velocity update [47], floor map [50], geomagnetic field [85], but their assumptions are too strong. As a consequence their performance is variable in complex real-world conditions: visual-inertial odometry assumes cameras have feature-rich, well illuminated scenes without occlusion, and PDR assumes the user's personal walking model and the phone placement are prior knowledge [48]. Recent deep learning based inertial tracking, IONet can learn direct location transforms from raw inertial data, and construct continuous accurate trajectories for indoor users, but still suffers from serious domain shifts and generalization problems. Our work aims to unsupervised learn the inertial tracking in unlabelled new domains, effectively increasing its generalization ability and flexibility in real usages.



(a) From the source domain to the target domain (b) From the target domain to the source domain

Figure 5.2: An overview of proposed Sequential Invariant Domain Adaptation (SIDA), consisting of encoders, decoders, generators, discriminators and predictors. (a) The source domain data are transformed into the generated target domain data, the reconstructed source domain data and the corresponding labels. (b) The target domain data are transformed into the generated source domain data, the reconstructed target domain data and the corresponding labels (The predictor is trained when the generated target domain data with the source domain labels or the real target domain data with a few target labels in the semi-supervised learning are available).

5.3 SIDA for Inertial Odometry

This section introduces a novel sequence domain adaptation model, that is able to transfer the learned knowledge from the source domain to the target domains for deep learning based inertial odometry.

5.3.1 Problem Formulation

The physical model, derived from Newtonian Mechanics, integrates the angular rates of the sensor frame $\{\mathbf{w}_i\}_{i=1}^{N}$ ($\mathbf{w}_i \in \mathbb{R}^3$ and N is the length of the whole sequence) measured by the three-axis gyroscope into orientation attitudes. While the linear accelerations of the sensor frame $\{\mathbf{a}_i\}_{i=1}^{N}(\mathbf{a}_i \in \mathbb{R}^3)$ measured by the three-axis accelerometer are transformed to the navigation frame and doubly integrated to give the position displacement, which discards the impact of the constant acceleration of gravity. This physical model is hard to implement directly on low-cost IMUs, because even a small measurement error will be exaggerated exponentially through the integration. Our work on IONet in Chapter 3 breaks the continuous integration by segmenting the sequence of inertial measurements

$\{(\mathbf{a}_i, \mathbf{w}_i)\}_{i=1}^{N}$ into subsequences. We denote a subsequence as $\mathbf{x} = \{(\mathbf{a}_i, \mathbf{w}_i)\}_{i=1}^{n}$, whose length is n. By taking subsequences as inputs, a recurrent neural network (RNN) is leveraged to periodically predict the polar vector $\mathbf{y} = (\Delta l, \Delta\psi)$, which represents the heading and location displacement:

$$(\Delta l, \Delta\psi) = \text{RNN}(\{(\mathbf{a}_i, \mathbf{w}_i)\}_{i=1}^{n}) \tag{5.1}$$

Based on the predicted $(\Delta l, \Delta\psi)$, we are able to easily construct the trajectories. However, it requires a large labelled dataset to build an end-to-end inertial tracking system, and it is infeasible to label data for every possible domain due to the motion dynamics and unpredictability of device placements. Therefore, we introduce the SIDA framework in the next subsection which is able to exploit the unlabelled sensory measurements in new domains and carry out accurate inertial tracking. We assume that the sensor data are collected within the same sample rate (i.e. 100 Hz) and agents perform similar motion across these domains (i.e. walking), although the sequence data from two domains are not paired.

5.3.2 Framework

As Figure. 5.2 illustrates, our proposed SIDA framework consists of encoder, generator, decoder, predictor and discriminator modules. Details of the model structure can be found at Figure 5.3, showing how the source domain sequences are transformed into the reconstructed source domain sequences, the generated target domain sequences and labels. Assume a scenario with two domains: a source domain and a target domain, where the source domain has labelled sequences $(\mathbf{x}^S, \mathbf{y}^S) \in \mathbb{D}^S$ ($\mathbf{y}^S$ is the sequence label - the polar vector according to $\mathbf{x}^S$ in estimating inertial odometry), and the target domain only has unlabelled sequences $\mathbf{x}^T \in \mathbb{D}^T$. Note that the sequences $\mathbf{x}^S$ and $\mathbf{x}^T$ are not aligned. The objectives of SIDA Framework are three-fold: 1) extracting domain-invariant representations $\mathbf{z}$ shared across domains; 2) generating fake sequences $\hat{\mathbf{x}}^T$ in the target domain conditioned on $\mathbf{x}^S$; 3) predicting sequence labels $\mathbf{y}^T$ in the target domain.

Sequence Encoder To extract the domain-invariant hidden representations $\mathbf{z}$ of sensory sequences across different domains, an RNN encoder is employed together with a specific domain vector θ:

$$\mathbf{z} = f_{\text{enc}}(\mathbf{x}, \theta) \tag{5.2}$$

where the hidden representaion $\mathbf{z}_i$ is aligned to the input sequence $\mathbf{x}_i$ at every ith time step, and the domain vector θ remains the same across all the time steps. The domain vectors are randomly generated to represent each domain. For different domains, we apply different

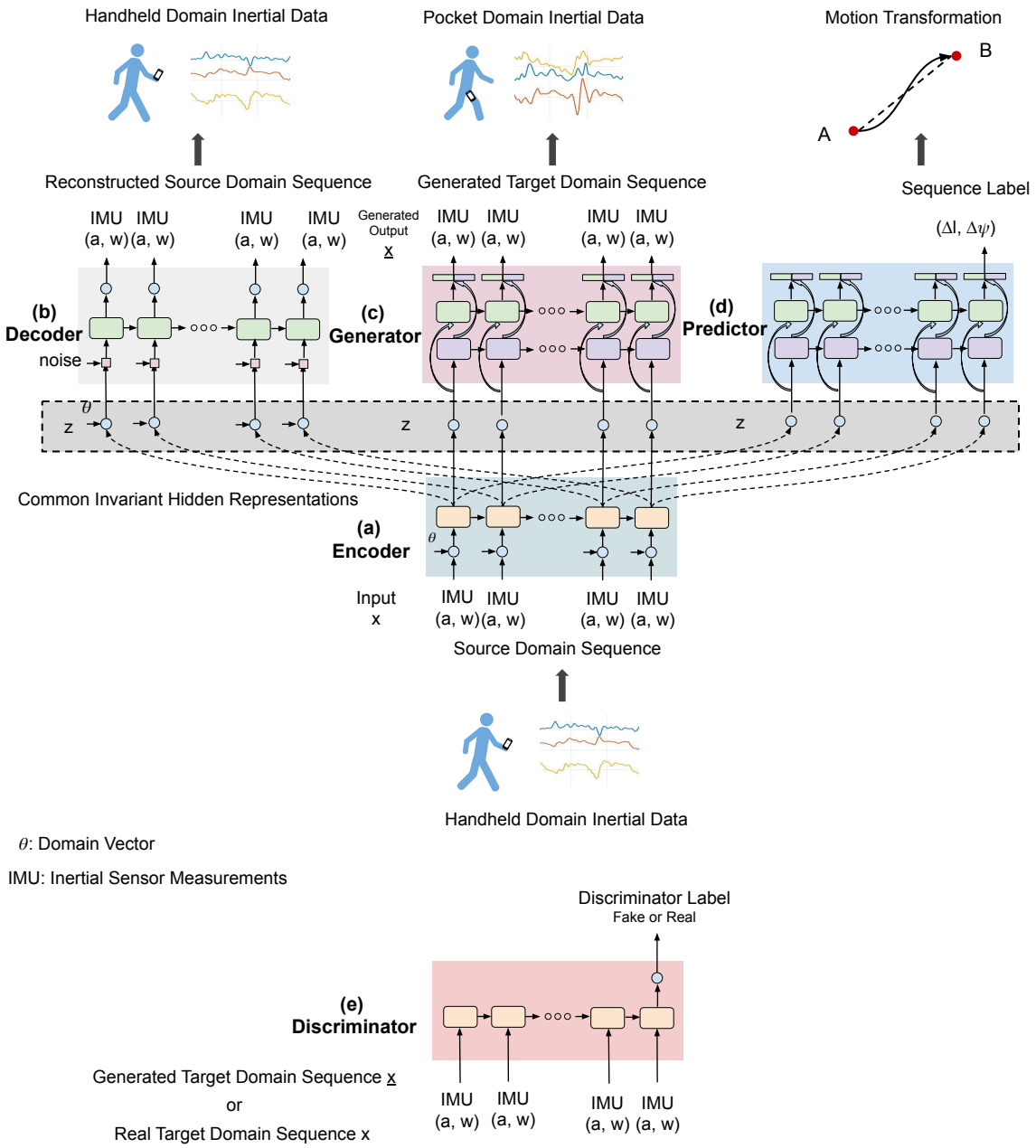


Figure 5.3: The architecture of proposed Sequential Invariant Domain Adaptation (SIDA). It includes (a) the source domain sequence *Encoder* (extracting common features across different domains), (b) the sequence reconstruction *Decoder* (reconstructing the sequence for learning better representations), (c) the target domain sequence *Generator* (generating sensory stream in the target domain), (d) the polar vector *Predictor* (producing a consistent trajectory for inertial navigation) and (e) the target sequence *Discriminator* (indicate whether the input is fake or real). This figure demonstrates the generation process from the source domain to the target domain, but omits the opposite one from the target domain to the source domain.

domain vectors θ that attempt to isolate domain-specific features, while the parameters of f_{enc} are shared across all the domains. An identical RNN encoder produces shared representations for the generator, decoder and predictor.

GAN Generator Having the domain-invariant representations $\mathbf{z}$, an RNN generator $f^T_{\text{gen}}(\mathbf{z})$ can be directly built to generate synthetic sequences $\hat{\mathbf{x}}^T$ in the target domain from the input sequences of source domain $\mathbf{x}^S$. By combining it with the encoder, we derive the sequence transformation model $G_{S\to T} = f^T_{\text{gen}} \circ f_{\text{enc}}$ as:

$$\hat{\mathbf{x}}^T = G_{S\to T}(\mathbf{x}^S, \theta^S) = f^T_{\text{gen}}(f_{\text{enc}}(\mathbf{x}^S, \theta^S)) \tag{5.3}$$

Likewise, we construct an inverse mapping $G_{T\to S} = f^S_{\text{gen}} \circ f_{\text{enc}}$ for generating the synthetic source domain sequences $\hat{\mathbf{x}}^S$ from the input sequences of target domain $\mathbf{x}^T$:

$$\hat{\mathbf{x}}^S = G_{T\to S}(\mathbf{x}^T, \theta^T) = f^S_{\text{gen}}(f_{\text{enc}}(\mathbf{x}^T, \theta^T)) \tag{5.4}$$

$G_{S\to T}$ is trained against a target domain discriminator D^T in the framework of GAN, and vice versa for $G_{T\to S}$. Unlike conventional GAN models, we decompose the generator by the domain-invariant representation $\mathbf{z}$. Intuitively, this architecture encourages the encoder f_{enc} to capture domain-invariant features that is capable of generating sensory sequences in different domains, as the encoding function f_{enc} are shared by both $G_{S\to T}$ and $G_{T\to S}$.

GAN Discriminator The discriminators are trained jointly with the generators, to determine the quality of data generation from the generators. Given an input sequence $\mathbf{x}$, our discriminator D leverages recurrent neural network (RNN) to output a probability value p, that represents the network belief in whether the input is synthetic sequences from the generators:

$$p = D(\mathbf{x}). \tag{5.5}$$

The discriminators are optimized to distinguish whether the input sequence is sampled from real dataset or generated by the generators.

Reconstruction Decoder In addition to the GAN generator, we introduce an RNN decoder f_{dec} to reconstruct the sequences $\breve{\mathbf{x}}$ conditioned on the hidden representations $\mathbf{z}$. This is aimed at reinforcing the learning of domain-invariant features when jointly learned with the GAN generator. Instead of using the conventional Denoising Autoencoders (DAE) [104] and Variational Autoencoders (VAE) [105], we only introduce an additive noise to the hidden representations $\bar{\mathbf{z}} = \mathbf{z} + \epsilon$, where $\epsilon \sim N(0, I^2)$, in order to simplify the learning

of the autoencoder component. Similar to the sequence encoder f_{enc}, the decoder is shared across all the domains and the domain vector θ is concatenated with inputs at every time step:

$$\breve{\mathbf{x}} = f_{\text{dec}}(\bar{\mathbf{z}}, \theta) = f_{\text{dec}}(f_{\text{enc}}(\mathbf{x}, \theta) + \epsilon, \theta) \tag{5.6}$$

By training the decoder along with the encoder as an autoencoder, it forces the hidden representations to keep key information that are necessary for reconstructing the original input.

Predictor Since the source domain has labels (polar vectors) aligned to every sensory sequence, it is straightforward to learn a predictor for carrying out inertial tracking by supervised learning the labels $\mathbf{y}^S$. However, this is not the ultimate objective of this paper, instead we aim at transferring the knowledge learned in the labelled source domain to the unlabelled target domain. Hence, with the help of the sequence encoder f_{enc}, we construct a predictor f_{pred} also shared by both the source domain and the target domain. In this case, though there exists no paired data $(\mathbf{x}^T, \mathbf{y}^T)$ for supervised learning in the target domain, we can still predict the target values $\mathbf{y}^T$ by:

$$\mathbf{y}^T = f_{\text{pred}}(f_{\text{enc}}(\mathbf{x}^T, \theta^T)) \tag{5.7}$$

The predictor f_{pred} takes the hidden domain-invariant features $\mathbf{z} = f_{\text{enc}}(\mathbf{x}^T, \theta^T)$ as input, and hence reduces the impacts from domain specific features in target domains.

5.3.3 Inference

This subsection introduces the learning method for jointly training the modules of our SIDA, including GAN loss $\mathcal{L}_G$, reconstruction loss $\mathcal{L}_{\text{AE}}$, prediction loss $\mathcal{L}_{\text{pred}}$, cycle-consistency $\mathcal{L}_{\text{cycle}}$ and perceptual consistency $\mathcal{L}_{\text{percep}}$:

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{GAN}} + \lambda_1 \mathcal{L}_{\text{AE}} + \lambda_2 \mathcal{L}_{\text{pred}} + \lambda_3 \mathcal{L}_{\text{cycle}} + \lambda_4 \mathcal{L}_{\text{percep}} \tag{5.8}$$

where λ_1, λ_2, λ_3, and λ_4 are the hyper-parameters used as the trade-off for the optimization process.

GAN Loss The GAN generator is one of the most important components in our framework, which is responsible for producing sensory sequences in the unlabelled target domain. Here, following the general GAN framework, we construct a discriminator D^T for

the corresponding target domain and learn to discriminate the generated data $\hat{\mathbf{x}}$ from the real one $\mathbf{x}$. The GAN loss for the target domain generator can be defined as:

$$\begin{aligned}\mathcal{L}_{G^T} =&\mathbb{E}_{\mathbf{x}^T\sim p(\mathbf{x}^T)}[\log D^T(\mathbf{x}^T)]+\\ &\mathbb{E}_{\mathbf{x}^S\sim p(\mathbf{x}^S)}[\log(1-D^T(G_{S\to T}(\mathbf{x}^S,\theta^S))],\end{aligned} \tag{5.9}$$

where $(\mathbf{x}^S, \mathbf{x}^T)$ are the source and target domain sequences, and θ^S is the source domain vector. Similarly, the GAN loss for the source domain generator is:

$$\begin{aligned}\mathcal{L}_{G^S} =&\mathbb{E}_{\mathbf{x}^S\sim p(\mathbf{x}^S)}[\log D^S(\mathbf{x}^S)]+\\ &\mathbb{E}_{\mathbf{x}^T\sim p(\mathbf{x}^T)}[\log(1-D^S(G_{T\to S}(\mathbf{x}^T,\theta^T))]\end{aligned} \tag{5.10}$$

Then, we combine these two losses into the final GAN loss $\mathcal{L}_{\text{GAN}} = \mathcal{L}_{G^S} + \mathcal{L}_{G^T}$.

Reconstruction Loss Considering the inputs are the continuous real-valued sequence data, the Mean Square Error (MSE) loss is chosen to optimize the autoencoder loss for source and target domain data respectively:

$$\mathcal{L}_{\text{AE}} = \mathbb{E}_{\mathbf{x}^S\sim p(\mathbf{x}^S),\mathbf{x}^T\sim p(\mathbf{x}^T)}[\|\breve{\mathbf{x}}^S-\mathbf{x}^S\|_2+\|\breve{\mathbf{x}}^T-\mathbf{x}^T\|_2], \tag{5.11}$$

where $(\mathbf{x}^S, \mathbf{x}^T)$ are the source and target domain sequences, while $(\breve{\mathbf{x}}^S, \breve{\mathbf{x}}^T)$ are the reconstructed source and target domain sequences from the decoders.

Prediction Loss In addition to the original paired data $(\mathbf{x}^S, \mathbf{y}^S)$ in the source domain, we are able to make use of the generated ones $\hat{\mathbf{x}}^T = f^T_{\text{gen}}(f_{\text{enc}}(\mathbf{x}^S, \theta^S))$ produced by the GAN generator in the target domain as well. The domain-invariant representations can be directly applied for the prediction no matter which domain the sequences are from. Both the original source domain data and the synthetic target data (generated from the source domain) share the same groundtruth prediction label $\mathbf{y}^S$. Hence, a joint regression loss can be constructed for learning the predictor:

$$\begin{aligned}\mathcal{L}_{\text{pred}} =&\mathbb{E}_{(\mathbf{x}^S,\mathbf{y}^S)\sim p(\mathbf{x}^S,\mathbf{y}^S)}[\|\mathbf{y}^S-f_{\text{pred}}(f_{\text{enc}}(\mathbf{x}^S,\theta^S))\|_2+\\ &\|\mathbf{y}^S-f_{\text{pred}}(f_{\text{enc}}(\hat{\mathbf{x}}^T,\theta^T))\|_2]\end{aligned} \tag{5.12}$$

Although the adversarial training is unable to produce exactly the same sequences as the ones generated in the target domain, the labels in the source domain encourages the sequence encoder to preserve the prominent features for prediction, so that the domain-invariant representations will be further regularised by the labels in the source domain.

Cycle Consistency Regularisation In order to improve the sensory sequence generation, we apply cycle-consistency regularisation to ensure the sequences generated to the target domain from the source domain can be mapped back without losing too much content information. As demonstrated by [215, 101, 216], this bidirectional architecture encourages the GAN to generate data in meaningful direction by penalizing the optimizer with the L_1 consistency loss defined as:

$$\begin{aligned}\mathcal{L}_{\text{cycle}} =&\mathbb{E}_{\mathbf{x}^S\sim p(\mathbf{x}^S)}\|G_{T\to S}(G_{S\to T}(\mathbf{x}^S,\theta^S),\theta^T)-\mathbf{x}^S\|_1+\\ &\mathbb{E}_{\mathbf{x}^T\sim p(\mathbf{x}^T)}\|G_{S\to T}(G_{T\to S}(\mathbf{x}^T,\theta^T),\theta^S)-\mathbf{x}^T\|_1\end{aligned} \tag{5.13}$$

Perceptual Consistency Regularisation To further regularise the learning of domain-invariant representations, we propose to use the perceptual consistency regularisation. Inspired by the f constancy [217], we employ the encoder f_{enc} as the perceptual function to enforce the semantic representation to remain constant after being transformed into another domain by generators. For example, in source domains, the hidden representation $\mathbf{z}^S = f_{\text{enc}}(\mathbf{x}^S,\theta^S)$ extracted by the encoder conditioned on the source domain vector, will be encouraged to be invariant under $G_{S\to T}$ by minimizing a L_2 distance between the original hidden representation $\mathbf{z}^S$ and the hidden representation $\hat{\mathbf{z}}^S = f_{\text{enc}}(\hat{\mathbf{x}}^T,\theta^T)$ extracted from the generated synthetic target domain data $\hat{\mathbf{x}}^T = G_{S\to T}(\mathbf{x}^S,\theta^S)$. A similar perceptual constraint can be applied for the target domain.

$$\begin{aligned}\mathcal{L}_{\text{percep}} =&\mathbb{E}_{\mathbf{x}^S\sim p(\mathbf{x})^S}\|f_{\text{enc}}(\mathbf{x}^S,\theta^S)-f_{\text{enc}}(G_{S\to T}(\mathbf{x}^S,\theta^S),\theta^T)\|_2\\ &+\mathbb{E}_{\mathbf{x}^T\sim p(\mathbf{x})^T}\|f_{\text{enc}}(\mathbf{x}^T,\theta^T)-f_{\text{enc}}(G_{T\to S}(\mathbf{x}^T,\theta^T),\theta^S)\|_2\end{aligned} \tag{5.14}$$

5.4 SIDA for Human Activity Recognition

Beyond inertial odometry, we also conduct a case study into domain adaptation for inertial sensor based human activity recognition (HAR), to show that our proposed SIDA is not restricted to a specific task. The inertial sensor attached to a moving object can provide not only ego-motion sense, but also additional context and semantic information. Human activity monitoring provides key context information in a wide range of applications [218], such as sports and healthcare monitoring, and human-computer interaction. Moreover, localization systems can be further enhanced with this external activity information: for example, if the IONet framework (in Chapter 3) is trained on the data from each motion mode, an activity recognition module will allow to select a suitable DNN model to predict inertial tracking.

The purpose of this task is to classify a user's activities (i.e. walking, running, and standing still) using smartphone based inertial sensors. Here, 3-dimensional accelerations are used to train a deep neural network to classify activity modes. Given a sequence of accelerations as input $\mathbf{x} = \{(\mathbf{a}_i)\}_{i=1}^{n}$, a recurrent neural network is leveraged to predict the motion label $\mathbf{y}$:

$$\mathbf{y} = \text{RNN}(\{(\mathbf{a}_i)\}_{i=1}^{n}), \tag{5.15}$$

where $\mathbf{y}$ is a category label, representing whether the user is walking, running or standing still.

Similar to inertial odometry, this problem is complicated due to the fact that various sensor attachments (e.g. in the hand, pocket or bag) generate completely different sensor measurements. A deep learning based HAR model trained in one domain is hard to generalize to another domain. Therefore, we consider to employ our proposed SIDA to a HAR model trained in a labelled domain (i.e. the source domain) for its adaptation into a new domain without labelled data (i.e. the target domain). This will enable the framework to exploit massive unlabelled sensor data. As HAR is a classification rather than a regression task, the predictor described in Section 5.3.2 needs to be changed to output a discrete categorical value $\mathbf{y}^T$ instead of a polar vector in the target domain:

$$\mathbf{y}^T = f_{\text{pred}}(f_{\text{enc}}(\mathbf{x}^T, \theta^T)) \tag{5.16}$$

Correspondingly, in the inference process described in Section 5.3.3, the L-2 loss inside the prediction loss $\mathcal{L}_{\text{pred}}$ should be replaced by a cross-entropy loss H:

$$\begin{aligned} \mathcal{L}_{\text{pred}} =& \mathbb{E}_{(\mathbf{x}^S,\mathbf{y}^S)\sim p(\mathbf{x}^S,\mathbf{y}^S)}[H(\mathbf{y}^S, f_{\text{pred}}(f_{\text{enc}}(\mathbf{x}^S, \theta^S)))+ \\ & H(\mathbf{y}^S, f_{\text{pred}}(f_{\text{enc}}(\hat{\mathbf{x}}^T, \theta^T)))]. \end{aligned} \tag{5.17}$$

The remaining modules of the SIDA framework stay unchanged.

5.5 Experiments

In this section, we conducted experiments to evaluate our proposed SIDA on domain adaptation for inertial odometry. This model is first evaluated on unsupervised transfer learning of motion transformation to new domains. Based on that, we show how to solve inertial odometry tracking in unlabelled target domains. An interpretability study is performed to visualize the encoded feature space. In addition, we apply SIDA to tackle domain adaptation for human activity recognition, showing that proposed SIDA model is not restricted to a particular task.

Table 5.1: The number of generated subsequences for training and testing inertial odometry and human activity recognition (HAR)

Inertial Odometry Domain	Training Subsequences	Testing Subsequences
Handheld	45K	4K
Pocket	53K	4K
Handbag	36K	4K
Trolley	29K	4K
HAR Domain	**Training Subsequences**	**Testing Subsequences**
Handheld	10K	2K
Front Pocket	10K	2K
Back Pocket	12K	2K
Bag	14K	2K

5.5.1 Experimental Setups

5.5.1.1 Training Details

OxIOD Dataset: we used Oxford Inertial Odometry Dataset (OxIOD) to train and evaluate our SIDA framework for inertial odometry. These source-domain labels are used for SIDA training while the target-domain labels are used for SIDA evaluation only. The length of each sequence is 200 frames (2 seconds), including three linear accelerations and three angular rates per frame. In summary, the dataset used in this work contains around 45K, 53 K, 36K and 29K sequences for handheld, pocket, bag, trolley domains respectively, as illustrated in Table 5.1. Among them, 4K sequences were selected as validation data in each domain, and the rest was taken as training set. In our training phase, we set the hyper-parameters $\lambda_1 = 0.01$, $\lambda_2 = 100$, $\lambda_3 = 0.1$, and $\lambda_4 = 1$.

HAR Dataset: we used a public IMU HAR Dataset [219] to train and evaluate our SIDA model for human activity recognition. The dataset was collected by inertial sensors from four Smartphones (i.e. Nexus 5), attached to the hand, the front pocket, the back pocket and the bag of each participant. Our experiment uses the Case 1 scenario of the dataset, with data collection of 7 participants (5 males, 2 females) to perform walking, running and standing still. These source-domain labels are used for SIDA training, while the target-domain labels are used for SIDA evaluation only. The length of each sequence is 200 frames (2 seconds), including three linear accelerations per frame. In this experiment, we used 10K, 10K, 12K, 14K sequences from the handheld, front pocket, back pocket and bag domain for training and 2K, 2K, 2K, 2K sequences from the handheld, front pocket, back pocket and bag domain for testing, as illustrated in Table 5.1. In our training phase, we set the hyper-parameters $\lambda_1 = 0.01$, $\lambda_2 = 1.0$, $\lambda_3 = 0.1$, and $\lambda_4 = 1$.

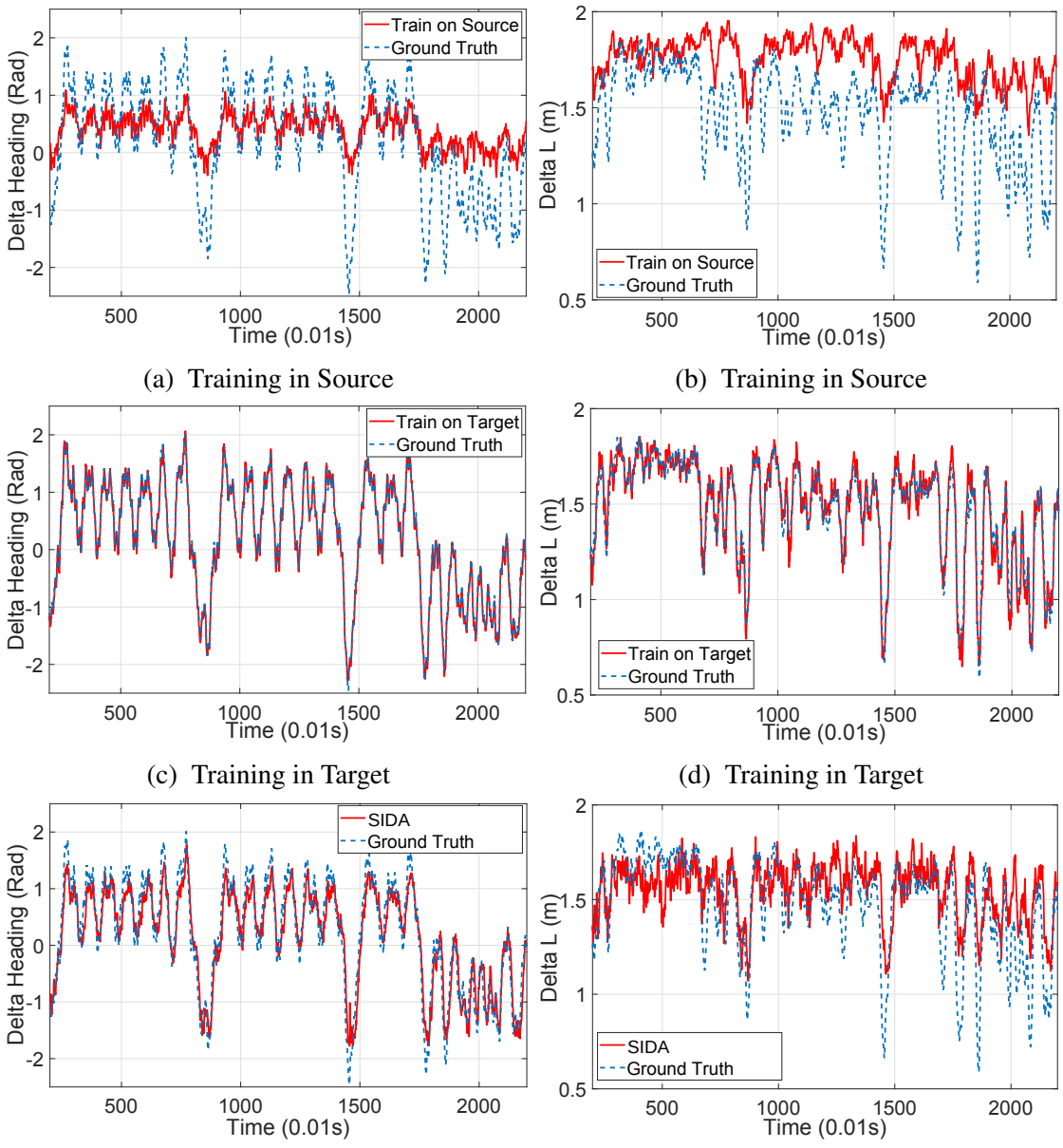
5.5.1.2 Baselines

To compare with our proposed SIDA framework, our chose two representative domain adaptation approaches as baselines, i.e. adversarial discriminative domain adaptation (ADDA) [205] and cycle-consistent adversarial domain adaptation (CyCADA) [103]. ADDA is a discriminative adversarial model that encourages domain confusion by minimizing the discriminative loss between source domain and target domain. CyCADA is based on generative adversarial model to generate fake target domain data, and train the prediction network in target domain with synthetic target data. As both ADDA and CyCADA are used in image processing, we replace their ConvNet encoders with a LSTM based encoders to fit the properties of sequential data in our tasks.

5.5.2 Transferring Across Motion Domains

We evaluate our model on unsupervised motion domain transfer tasks. The source domain is the inertial data collected in the handheld attachment, while the target domains are those collected in the attachments of pocket, handbag and trolley. We test our framework with the real target data. Its generalization performance is evaluated by comparing the label prediction (polar vector) with the ground-truth data captured by Vicon system. We compare with *source-only*, where we use the trained source predictor to predict data directly in the target domain and with *target-only* where we train the target dataset with target labels (40K) to show the performance of fully supervised learning. Figure 5.4 presents the predicted location and heading displacement in pocket domain for the three different techniques. It can be seen that *source-only* is unable to follow either delta heading or delta location accurately, whereas SIDA achieves a level of performance close to the fully supervised target-only, especially for delta heading.

Table. 5.2 presents the *quantitative analysis* with a metric of mean square error of the label prediction against the ground truth. Compared with using a model trained on source data only, our proposed unsupervised sequence domain adaptation technique helps to dramatically decrease the validation loss (almost 6 times, 9 times, and 3 times in pocket, bag and trolley domains). Two popular baselines are compared with SIDA: i) adversarial discriminative domain adaptation (ADDA) [205] and ii) cycle-consistent adversarial domain adaptation (CyCADA) [103]. ADDA is a discriminative model, forcing the feature fusion of two domains by distinguishing the features after the encoder. CyCADA is a generative model, using standard Cycle-GAN framework to generate synthetic target domain data and fine-tune the predictor through synthetic data. Because they both aim to process images



(a) Training in Source

(b) Training in Source

(c) Training in Target

(d) Training in Target

(e) SIDA

(f) SIDA

Figure 5.4: Heading displacement estimation from training in (a) source domain, (c) target domain and (d) SIDA, and location displacement estimation from training in (b) source domain, (d) target domain and (f) SIDA

Table 5.2: Unsupervised Transfer Across Motion Domains

Model	Hand → Pocket	Hand → Bag	Hand → Trolley
Training on Source, Testing on Target	0.718	1.129	0.273
ADDA with LSTM	0.471	0.631	0.216
CyCADA with LSTM	0.237	0.455	0.182
SIDA w/o Reconstruction	0.313	0.335	0.113
SIDA w/o Perceptual Loss	0.315	0.202	0.140
SIDA	**0.119**	**0.123**	**0.098**
Semi-Supervised SIDA (1K)	0.113	0.075	0.047
Train on Target, Test on Target (40K)	0.045	0.010	0.006

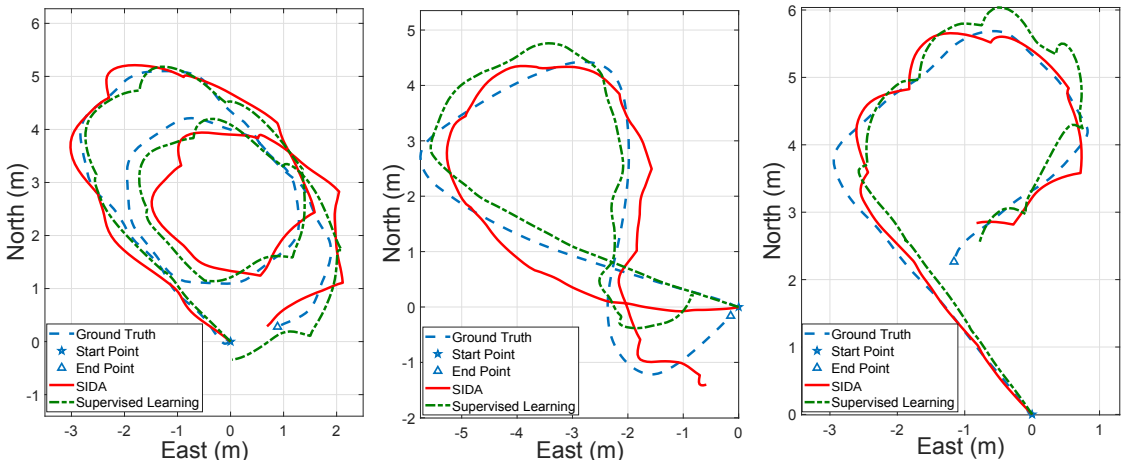
- The results are reported in the averaged RMSE of the predicted polar vectors (the smaller number is better).

rather than continuous sequential data, we replace their convolutional generator and discriminator with the same LSTM layers as described in our frameworks. Moreover, we cut down the reconstruction loss and perceptual loss in our framework respectively to show their impact. As shown in Table. 5.2, our SIDA still achieves competitive performance compared to these baselines.

Lastly, we also evaluate our framework on a semi-supervised domain transfer task, with 1K labelled target domain sequences to train the predictor. As shown in Table. 5.2, the sparse labels in target domains help decrease the prediction error, especially in the bag and trolley domains.

5.5.3 Inertial Tracking in Unlabelled Domains

We argue that the predicted label from our domain transformation framework is capable of solving a downstream task - inertial odometry tracking. In an inertial tracking task, the precision of the predicted label determines the localization accuracy, as the current location (x_n, y_n) is calculated by using an initial location (x_0, y_0) and heading, and chaining the results of previous windows via Equation 5.18. This dead reckoning (DR) technique, also called path integration, can be widely found in animal navigation, which enables animals to use inertial cues (e.g. steps and turns) to track themselves in the absence of vision. The errors in path integration will accumulate and cause unavoidable drifts in trajectory estimation, which imposes a requirement for accurate motion domain transformation. Without domain adaptation, if the model trained on the source domain is directly applied to data from target domains, it will lead to erroneous trajectories. When using only inertial information, traditional model-based navigation algorithms perform poorly in unlabelled



(a) Pocket (b) Trolley (c) Bag

Figure 5.5: Inertial tracking trajectories of (a) Pocket (b) Trolley (c) Handbag, comparing our proposed unsupervised SIDA with Ground Truth and Supervised Learning. SIDA was trained on Handheld data as the source.

scenarios, as SINS will collapse due to the high measurement noises, and PDR will be influenced by incorrect step detection or device orientation.

$$\begin{cases} x_n = x_0 + \Delta l cos(\psi_0 + \Delta\psi) \\ y_n = y_0 + \Delta l sin(\psi_0 + \Delta\psi) \end{cases} \tag{5.18}$$

We show that the inertial tracking trajectory can be recovered from the labels predicted by our domain adaptation framework in *unlabelled* domains. The participant walked with the device placed in the pocket, the handbag and on the trolley. The inertial data during test walking trajectory was not included in training dataset, and collected over different days. Figure 5.5 illustrates that our proposed model succeeds in generating physically meaningful trajectories, close to the ground truth captured by Vicon system. It proves that exploiting the raw sensory stream and transforming to a common latent distribution can extract meaningful semantic features that help solve downstream tasks.

5.5.4 Interpreting the Sequence Encoder

The role of the sequence encoder is evaluated by the use of t-SNE projection [220] to show its ability to map the raw data from two domains to an identical semantic space. We compare it with two other baselines: a domain-specific encoder, which is only employed in the source domain (it is not shared across domains); an ADDA encoder, which is learned jointly with the predictor by adversarial training to force the fusion of representations extracted by the encoder. The t-SNE projection is shown in Figure 5.6, and all of the models

Figure 5.6: Visualization of extracted representations in the source and target domains by t-SNE. It can be seen that the SIDA leads to a more consistent latent representation compared with the disjoint representations of the normal encoder and the ADDA encoder.

apply the same parameters (Perplexity=10, step=5000). As can be seen, the data points of the domain-specific encoder are distinctly separated into two folds. ADDA attempts to fuse the points from two domains but it turns out points are still clearly separated. By contrast, the encoder of our SIDA is able to better scatter the points dispersively in the semantic space, which removes the domain shifts and benefits target label prediction.

5.5.5 Domain Adaptation for Human Activity Recognition

Finally, we come to evaluate the SIDA framework on the human activity recognition problem. We use the data collected from handheld attachment (i.e. hand domain) as the source domain to train an activity recognition model to classify human activity using inertial sensor data. In this dataset, human activity is categorized as three classes: walking, running and standing still. This task is aimed at transferring the HAR model into other domains (i.e. the data from front pocket, back pocket and bag attachments), by exploiting unlabelled data from these target domains.

Table 5.3 shows the classification accuracy of testing models in target domains. *Source-only* denotes that the model is trained on the source domain but tested on the target domain without domain adaptation, while *target-only* denotes that the model is both trained and tested on target domain. It is clear to see that our proposed SIDA approach significantly improves the model without domain adaptation i.e. *source-only*, from 8%-9% to around 70%. Compared with two baselines i.e. ADDA and CyCADA, our SIDA models consistently outperform them in all three domains. Especially, the discriminative adversarial approach (ADDA) has little effect on domain adaption in this case, while the generative adversarial approaches (CyCADA and SIDA) have significant improvement over the basic *source-only* model.

An interesting point is that the performance of *Source Only* and ADDA is even worse than a random guess. This is because the number of testing samples in each category

Table 5.3: Unsupervised Domain Adaptation for Activity Recognition

Model	Hand → Front Pocket	Hand → Back Pocket	Hand → Bag
Source Only	8.75%	8.54%	9.57%
ADDA with LSTM	8.74%	8.54%	9.53%
CyCADA with LSTM	65.01%	57.33%	65.13%
SIDA (Ours)	**70.94%**	**69.47%**	**76.36%**
Target Only	94.92%	97.43%	91.54%

- The results are reported in the classification accuracy of model prediction in target domains (the larger number is better).

is imbalanced: in the front pocket domain, the three classes, i.e. standing still, walking and running occupy 28.88%, 62.42% and 8.71% of the total samples; in the back pocket domain, they occupy 29.25%, 62.25%, and 8.49% respectively; in the bag domain, they occupy 23.60%, 66.92%, and 9.48%. The HAR models without effective domain adaptation will recognize all human activities as running, as the variation of inertial sensor data from the source domain is relatively more smooth (the inertial sensor is fixed in hand firmly), and the trained models experience more dramastic movement in other domains (the inertial sensor swings with the attached pocket or bag), causing the learning models to categorize all data samples as running rather than walking and standing still.

5.6 Summary

Motion transformation between different domains is a challenging task, which typically requires the use of labelled data for training. In the presented framework, by transforming target domains to a consistent invariant representation, a physically meaningful trajectory can be accurately reconstructed. Intuitively, our technique learns how to transform data from an arbitrary sensor domain θ to a common latent representation. Analogously, this is equivalent to learning how to translate any sensor frame to the navigation frame, without any labels in the target domain. Beyond learning based inertial odometry, our framework is a generic framework, that can be applied to solve domain adaptation using time-series data, e.g. human activity recognition in unlabelled domains. Although SIDA has been shown to work on IMU data, the broad framework is likely to be suitable for any continuous, sequential domain transformation task where there is an underlying physical model.

A limitation of this work is that we have not studied the domain shift problem of visual localization models. In visual systems, some environmental issues such as weather, light-

ing conditions and motion dynamics alter visual data significantly in each domain. Compared with raw images, inertial data is relatively low-dimensional (only 6-dimensional per frame), and hence we first considered to solve domain adaptation for inertial positioning. The future direction includes extending this framework to solve more general localization problem. The encoders, generators and decoders of SIDA framework should be replaced with suitable deep convolutional neural networks in order to effectively extract high-level features from a sequence of images i.e. video frames. Another direction is to investigate the effectiveness of SIDA, working in the situations where the data of two domains are sampled from two distinctly different motions.

Chapter 6

Conclusions and Future Work

6.1 Conclusions

This thesis has investigated robust localization approaches using deep learning for real-world deployment on mobile devices, self-driving vehicles and mobile robots.

Due to the unique properties of inertial sensor and its importance in ego-centric localization, the first question is how to effectively exploit noisy inertial data for long-term localization in large-scale real-world environment. The inevitable bias and noise of low-cost inertial sensor makes this problem complicated. To constrain the accumulative inertial system drifts, Chapter 3 formulates inertial tracking as a sequential learning problem. In doing so, we propose Inertial Odometry Neural Network (IONet), a novel deep learning model to estimate polar vectors (i.e. location transformations in polar coordinate) from independent sequences of inertial measurements, with the aid of training above large dataset. IONet can estimate the user's egomotion and reconstruct an accurate trajectory regardless of sensor attachment. Meanwhile, this framework produces uncertainties along with pose predictions, showing to what extent the deep neural network (DNN) models prediction can be trusted. Unlike classical analytical models, IONet demonstrates its ability to solve a more general motion without periodicity, e.g. wheeled configurations, and also work in highly dynamic motion patterns, e.g. running and mixed velocity motion. As a first trial in this direction, we collected a very large dataset with pedestrian, multi-attachment sensor data and high-precision labels, totalling more than 42 km in distance. We released it as the Oxford Inertial Odometry Dataset (OxIOD) to boost research in data-driven pedestrian inertial navigation. Extensive experiments were conducted to show that the proposed IONet model outperforms competing analytical models, i.e. standard inertial navigation algorithm and pedestrian dead reckoning (PDR). Furthermore, Lighweight Inertial Odometry Neural Network (L-IONet) was developed to more efficiently learn and infer inertial

odometry from raw inertial measurements. By replacing the recurrent model of the standard IONet with a casual forward autoregressive model, L-IONet reduces both the training and inference time, allowing real-time inference on low-end devices, such as smartphones and smartwatches. A systematic research is provided on the inference performance of DNN models on several off-the-shelf consumer devices.

Chapter 4 studies how to conduct effective sensor fusion for DNN-based localization. We introduce SelectFusion, a generic framework that models feature selection for robust sensor fusion. Two variants of selective fusion strategies - a deterministic soft fusion and a Gumbel-softmax based hard fusion, are proposed to select the useful features of available sensor modalities, in support of odometry estimation and localization. The proposed selective sensor fusion module can be incorporated into a uniform framework to perform state estimation, which is not restricted by specific modality or task. This framework is evaluated above learning based visual-inertial odometry, depth-vision relocalization and lidar-vision odometry to show the improvement of performance in robustness and accuracy. The fusion mask can be visualised to investigate the correlations between the number of selected features and environment/motion dynamics. In addition, we created challenging dataset on top of current public datasets by considering seven different sources of sensor degradation. Based on that, we conducted a new and complete study on the accuracy and robustness of deep sensor fusion in presence of corrupted data.

Chapter 5 explores the problem of improving the generalization ability of DNN-based localization models in new domains, a key point in locating users robustly in the wild. We propose MotionTransformer, a sequence domain adaptation framework for processing long continuous sensory data with General Adversarial Networks (GANs). The intuition is that the underlying statistical distribution of pose updates must be similar, if derived from a common agent (e.g. human motion). Thus we use a shared encoder to transform sequence data into a domain-invariant hidden representation, whilst discarding domain specific features. This framework reduces the effort in converting the raw sensor data into an accurate trajectory, as no labelled or paired data is required to achieve motion transformation in new domains.

6.2 Future Work

Current work on 'learning to locate robustly' can be naturally extended into 'learning to estimate states robustly' with a broader vision towards enabling mobile agents to perceive and interact with the environment based on their onboard sensors. To achieve reliable,

explainable, and controllable neural network systems, three main factors should be considered: how to build model structures to estimate and update system states, how to infer and optimize neural weights, and how to store and represent world knowledge, each of which will be discussed below as prospective future research directions:

Model Structure - from deterministic to deep probabilistic models: One particularly promising direction is to construct a deep probabilistic model for states estimation instead of learning states deterministically from data. A probabilistic model can capture the inherent stochastic measurement and process noise of a system. It also provides uncertainty estimation, which allows us to know to what extent the predictions from DNNs are trusted, and hence avoids the catastrophic consequences of mispredicting. Moreover, uncertainties can be tightly integrated into a deep probabilistic model to prevent overfitting, achieve more robust predictions, effective fusion of multimodal sensor, and accumulate information for system states update, for example, how to fuse and weight new data in the SLAM backend. With the aid of probabilistic structure, we can formulate prior knowledge or structural constraints as a prior distribution, incorporate it into DNN models. This will reduce the amount of training data and convergence time, while improving the generalization ability in a new domain.System behaviours can be indicated by the explicit variables in probabilistic models. A good example is to achieve probabilistic multimodal localization by integrating probabilistic model into our work in Chapter 4.

Lifelong Learning - from closed-set supervised learning to incremental self-supervised learning: Current DNN models are usually trained and evaluated in a closed-set conditions on top of labelled dataset. For example, our work in Chapter 3 and 5 is achieved via supervised learning. However, when operating in the uncontrolled real-world, robots confront complex dynamics, unknown objects, and ever-changing environments. Hence, neural network based systems should continuously and incrementally adapt into new environments. Through self-supervision signals (e.g. geometry constraints), weak signals (e.g. score function in reinforcement or imitation learning) or inductive biases (e.g. via a generative model), the models can learn to 'fine-tune' weights by themselves with observations only. Above this, high-level tasks, e.g. motion planning, navigation and decision making can be explored. Regarding unknown objects, information retrieval techniques allow for the acquisition of information from other resources, e.g. other sensors, website or human-in-loop to extend the agent's knowledge. A good example is to exploit the self-supervision signal described in [86] to achieve unsupervised visual odometry or visual-inertial odometry based on our work in Chapter 4 and 5.

World Knowledge - from a metric map to general map representation: Localization should be tightly coupled with mapping, as mapping provides reference features to constrain location error, while precise pose information increases the mapping quality. Conventional metric maps (e.g. occupancy grid, landmarks, boundary) are explicit ways to encode scenes into geometric structures. Future map representations should consist of high-level features which capture the scene geometry, appearance and semantics. With recent advances in generative models, i.e. Generative Adversarial Networks (GAN), Variational Autoencoders (VAEs), Generative Query Networks (GQNs), it is possible to learn a semantic, compact, generative scene representation from high-dimensional observations, e.g. images. This general map representation will further help to estimate camera poses, recognize places, compress scenes, and reduce the impact of environmental dynamics. They should be task-driven, which means relevant perceptual information is selected and with complexity varying depending on the task. Moreover, they can be connected in a topological way (e.g. Factor Graph, Graph Neural Networks), which is efficient and effective for the state estimation above it.

References

[1] C. R. Fetsch, A. H. Turner, G. C. DeAngelis, and D. E. Angelaki, "Dynamic Reweighting of Visual and Vestibular Cues during Self-Motion Perception," *Journal of Neuroscience*, vol. 29, no. 49, pp. 15601–15612, 2009.

[2] K. E. Cullen, "The Vestibular System: Multimodal Integration and Encoding of Self-motion for Motor Control," *Trends in Neurosciences*, vol. 35, no. 3, pp. 185–196, 2012.

[3] A. J. Davison, "FutureMapping: The Computational Structure of Spatial AI Systems," *arXiv*, 2018.

[4] N. Sünderhauf, O. Brock, W. Scheirer, R. Hadsell, D. Fox, J. Leitner, B. Upcroft, P. Abbeel, W. Burgard, M. Milford, and P. Corke, "The Limits and Potentials of Deep Learning for Robotics," *International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 405–420, 2018.

[5] R. Harle, "A Survey of Indoor Inertial Positioning Systems for Pedestrians," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 3, pp. 1281–1293, 2013.

[6] M. Gowda, A. Dhekne, S. Shen, R. R. Choudhury, X. Yang, L. Yang, S. Golwalkar, and A. Essanian, "Bringing IoT to Sports Analytics," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2017.

[7] A. Dhekne, A. Chakraborty, K. Sundaresan, and S. Rangarajan, "TrackIO : Tracking First Responders Inside-Out," in *USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2019.

[8] T. D. Barfoot, *State Estimation for Robotics*. Cambridge University Press, 2019.

[9] P. G. Savage, "Strapdown Inertial Navigation Integration Algorithm Design Part 2: Velocity and Position Algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 1, pp. 19–28, 1998.

[10] D. Nister, O. Naroditsky, and J. Bergen, "Visual odometry," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 1, pp. I–652–I–659 Vol.1, 2004.

[11] J. Engel, J. Sturm, and D. Cremers, "Semi-Dense Visual Odometry for a Monocular Camera," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 1449–1456, 2013.

[12] C. Forster, M. Pizzoli, and D. Scaramuzza, "SVO: Fast Semi-Direct Monocular Visual Odometry," in *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 15–22, 2014.

[13] M. Li and A. I. Mourikis, "High-precision, Consistent EKF-based Visual-Inertial Odometry," *The International Journal of Robotics Research*, vol. 32, no. 6, pp. 690–711, 2013.

[14] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-Based Visual–Inertial Odometry Using Nonlinear Optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[15] C. Forster, L. Carlone, F. Dellaert, and D. Scaramuzza, "On-Manifold Preintegration for Real-Time Visual-Inertial Odometry," *IEEE Transactions on Robotics*, vol. 33, no. 1, pp. 1–21, 2017.

[16] T. Qin, P. Li, and S. Shen, "VINS-Mono: A Robust and Versatile Monocular Visual-Inertial State Estimator," *IEEE Transactions on Robotics*, vol. 34, no. 4, pp. 1004–1020, 2018.

[17] J. Zhang and S. Singh, "LOAM: Lidar Odometry and Mapping in Real-time," in *Robotics: Science and Systems*, 2010.

[18] A. J. Davison, I. D. Reid, N. D. Molton, and O. Stasse, "MonoSLAM: Real-Time Single Camera SLAM," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 29, no. 6, pp. 1052–1067, 2007.

[19] R. Mur-Artal, J. Montiel, and J. D. Tardos, "ORB-SLAM : A Versatile and Accurate Monocular SLAM System," *IEEE Transactions on Robotics*, vol. 31, no. 5, pp. 1147–1163, 2015.

[20] Y. Lecun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, 2015.

[21] D. Kumaran, D. Hassabis, T. Graepel, M. Lai, D. Silver, M. Lanctot, L. Sifre, T. Hubert, K. Simonyan, I. Antonoglou, T. Lillicrap, J. Schrittwieser, and A. Guez, "A General Reinforcement Learning Algorithm that Masters Chess, Shogi, and Go through Self-play," *Science*, vol. 362, no. 6419, pp. 1140–1144, 2018.

[22] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems*, pp. 3104–3112, 2014.

[23] K. He, X. Zhang, S. Ren, and J. Sun, "Deep Residual Learning for Image Recognition," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 7, pp. 171–180, 2016.

[24] W. Maddern, G. Pascoe, C. Linegar, and P. Newman, "1 Year, 1000km: The Oxford RobotCar Dataset," *The International Journal of Robotics Research*, vol. 36, no. 1, pp. 3–15, 2016.

[25] L. M. Ni, Y. Liu, Y. C. Lau, and A. P. Patil, "Landmarc: indoor location sensing using active rfid," in *Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003.(PerCom 2003).*, pp. 407–415, IEEE, 2003.

[26] S. A. Ellwood, C. Newman, R. A. Montgomery, V. Nicosia, C. D. Buesching, A. Markham, C. Mascolo, N. Trigoni, B. Pasztor, V. Dyo, *et al.*, "An active-radio-frequency-identification system capable of identifying co-locations and social-structure: Validation with a wild free-ranging animal," *Methods in Ecology and Evolution*, vol. 8, no. 12, pp. 1822–1831, 2017.

[27] V. Dyo, S. A. Ellwood, D. W. Macdonald, A. Markham, N. Trigoni, R. Wohlers, C. Mascolo, B. Pásztor, S. Scellato, and K. Yousef, "Wildsensing: Design and deployment of a sustainable sensor network for wildlife monitoring," *ACM Transactions on Sensor Networks (TOSN)*, vol. 8, no. 4, pp. 1–33, 2012.

[28] P. Bahl and V. N. Padmanabhan, "Radar: An in-building rf-based user location and tracking system," in *Proceedings IEEE INFOCOM 2000. Conference on Computer Communications.*, vol. 2, pp. 775–784, IEEE, 2000.

[29] S. Kumar, R. M. Hegde, and N. Trigoni, "Gaussian process regression for fingerprinting based localization," *Ad Hoc Networks*, vol. 51, pp. 1–10, 2016.

[30] H. Wen, Y. Shen, S. Papaioannou, W. Churchill, N. Trigoni, and P. Newman, "Opportunistic radio assisted navigation for autonomous ground vehicles," in *2015 International Conference on Distributed Computing in Sensor Systems*, pp. 21–30, IEEE, 2015.

[31] Z. Xiao, H. Wen, A. Markham, N. Trigoni, P. Blunsom, and J. Frolik, "Non-line-of-sight identification and mitigation using received signal strength," *IEEE Transactions on Wireless Communications*, vol. 14, no. 3, pp. 1689–1702, 2014.

[32] A. Symington and N. Trigoni, "Encounter based sensor tracking," in *Proceedings of the thirteenth ACM international symposium on Mobile Ad Hoc Networking and Computing*, pp. 15–24, 2012.

[33] S. B. Cruz, T. E. Abrudan, Z. Xiao, N. Trigoni, and J. Barros, "Neighbor-aided localization in vehicular networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 10, pp. 2693–2702, 2017.

[34] X. Zhao, Z. Xiao, A. Markham, N. Trigoni, and Y. Ren, "Does btle measure up against wifi? a comparison of indoor location performance," in *European Wireless 2014; 20th European Wireless Conference*, pp. 1–6, VDE, 2014.

[35] R. Faragher and R. Harle, "Location fingerprinting with bluetooth low energy beacons," *IEEE journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2418–2428, 2015.

[36] S. Gezici, Z. Tian, G. B. Giannakis, H. Kobayashi, A. F. Molisch, H. V. Poor, and Z. Sahinoglu, "Localization via ultra-wideband radios: a look at positioning aspects for future sensor networks," *IEEE signal processing magazine*, vol. 22, no. 4, pp. 70–84, 2005.

[37] F. Zafari, A. Gkelias, and K. K. Leung, "A survey of indoor localization systems and technologies," *IEEE Communications Surveys & Tutorials*, vol. 21, no. 3, pp. 2568–2599, 2019.

[38] K. Vickery, "Acoustic positioning systems. a practical overview of current systems," in *Proceedings of the 1998 Workshop on Autonomous Underwater Vehicles (Cat. No. 98CH36290)*, pp. 5–17, IEEE, 1998.

[39] A. Markham, N. Trigoni, D. W. Macdonald, and S. A. Ellwood, "Underground localization in 3-d using magneto-inductive tracking," *IEEE Sensors Journal*, vol. 12, no. 6, pp. 1809–1816, 2011.

[40] A. Markham and N. Trigoni, "Magneto-inductive networked rescue system (miners) taking sensor networks underground," in *Proceedings of the 11th international conference on Information Processing in Sensor Networks*, pp. 317–328, 2012.

[41] T. E. Abrudan, Z. Xiao, A. Markham, and N. Trigoni, "Distortion rejecting magneto-inductive three-dimensional localization (magloc)," *IEEE Journal on Selected Areas in Communications*, vol. 33, no. 11, pp. 2404–2417, 2015.

[42] P. G. Savage, "Strapdown Inertial Navigation Integration Algorithm Design Part 1: Attitude Algorithms," *Journal of Guidance, Control, and Dynamics*, vol. 21, no. 1, pp. 19–28, 1998.

[43] N. El-Sheimy, H. Hou, and X. Niu, "Analysis and modeling of inertial sensors using Allan variance," *IEEE Trans. Instrum. Meas.*, vol. 57, pp. 140–149, Jan. 2008.

[44] L. Barreda Pupo, "Characterization of errors and noises in mems inertial sensors using allan variance method," Master's thesis, Universitat Politècnica de Catalunya, 2016.

[45] I. Board, "Ieee standard specification format guide and test procedure for single-axis interferometric fiber optic gyros," *IEEE Std*, pp. 952–1997, 1998.

[46] D. W. Allan, "Statistics of atomic frequency standards," *Proceedings of the IEEE*, vol. 54, no. 2, pp. 221–230, 1966.

[47] I. Skog, P. Händel, J.-O. Nilsson, and J. Rantakokko, "Zero-Velocity Detection — an Algorithm Evaluation.," *IEEE transactions on bio-medical engineering*, vol. 57, no. 11, pp. 2657–2666, 2010.

[48] A. Brajdic and R. Harle, "Walk Detection and Step Counting on Unconstrained Smartphones," in *The ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, 2013.

[49] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, "Robust pedestrian dead reckoning (R-PDR) for arbitrary mobile device placement," in *Proc. Int. Conf. Indoor Positioning and Indoor Navigation*, (Busan, South Korea), pp. 187–196, Oct. 2014.

[50] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, "Lightweight Map Matching for Indoor Localization using Conditional Random Fields," in *International Conference on Information Processing in Sensor Networks (IPSN)*, pp. 131–142, 2014.

[51] A. Rai, K. K. Chintalapudi, V. N. Padmanabhan, and R. Sen, "Zee: Zero-Effort Crowdsourcing for Indoor Localization," *The Annual International Conference on Mobile Computing and Networking (MobiCom)*, p. 293, 2012.

[52] S. Wang, H. Wen, R. Clark, and N. Trigoni, "Keyframe based Large-Scale Indoor Localisation using Geomagnetic Field and Motion Pattern," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1910–1917, 2016.

[53] R. A. Newcombe, S. J. Lovegrove, and A. J. Davison, "DTAM : Dense Tracking and Mapping in Real-Time," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 2320–2327, 2011.

[54] S. Wang, R. Clark, H. Wen, and N. Trigoni, "End-to-end, sequence-to-sequence probabilistic visual odometry through deep neural networks," *The International Journal of Robotics Research*, vol. 37, no. 4-5, pp. 513–542, 2018.

[55] G. Klein and D. Murray, "Parallel tracking and mapping for small ar workspaces," in *The 6th IEEE and ACM International Symposium on Mixed and Augmented Reality*, pp. 1–10, IEEE Computer Society, 2007.

[56] S. Thrun, W. Burgard, and D. Fox, *Probabilistic robotics*. MIT press, 2005.

[57] W. Zhang and J. Kosecka, "Image based localization in urban environments.," in *International Symposium on 3D Data Processing Visualization and Transmission (3DPVT)*, vol. 6, pp. 33–40, Citeseer, 2006.

[58] T. Sattler, B. Leibe, and L. Kobbelt, "Fast image-based localization using direct 2d-to-3d matching," in *International Conference on Computer Vision (ICCV)*, pp. 667–674, IEEE, 2011.

[59] M. Cummins and P. Newman, "Fab-map: Probabilistic localization and mapping in the space of appearance," *The International Journal of Robotics Research*, vol. 27, no. 6, pp. 647–665, 2008.

[60] A. I. Mourikis and S. I. Roumeliotis, "A multi-state constraint Kalman filter for vision-aided inertial navigation," in *Proceedings - IEEE International Conference on Robotics and Automation (ICRA)*, pp. 3565–3572, 2007.

[61] M. Bloesch, M. Burri, S. Omari, M. Hutter, and R. Siegwart, "Iterated extended kalman filter visual-inertial odometry using direct photometric feedback," *The International Journal of Robotics Research*, vol. 36, no. 10, pp. 1053–1072, 2017.

[62] A. I. Mourikis and S. I. Roumeliotis, "A dual-layer estimator architecture for long-term localization," in *2008 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops*, pp. 1–8, IEEE, 2008.

[63] T.-C. Dong-Si and A. I. Mourikis, "Motion tracking with fixed-lag smoothing: Algorithm and consistency analysis," in *2011 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 5655–5662, IEEE, 2011.

[64] A. Patron-Perez, S. Lovegrove, and G. Sibley, "A spline-based trajectory representation for sensor fusion and rolling shutter cameras," *International Journal of Computer Vision*, vol. 113, no. 3, pp. 208–219, 2015.

[65] V. Indelman, S. Williams, M. Kaess, and F. Dellaert, "Information fusion in navigation systems via factor graph based incremental smoothing," *Robotics and Autonomous Systems*, vol. 61, no. 8, pp. 721–738, 2013.

[66] M. Kaess, A. Ranganathan, and F. Dellaert, "isam: Incremental smoothing and mapping," *IEEE Transactions on Robotics*, vol. 24, no. 6, pp. 1365–1378, 2008.

[67] P. Tanskanen, T. Naegeli, M. Pollefeys, and O. Hilliges, "Semi-direct ekf-based monocular visual-inertial odometry," in *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6073–6078, IEEE, 2015.

[68] L. Von Stumberg, V. Usenko, and D. Cremers, "Direct sparse visual-inertial odometry using dynamic marginalization," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2510–2517, IEEE, 2018.

[69] J. Zhang and S. Singh, "Loam: Lidar odometry and mapping in real-time.," in *Robotics: Science and Systems*, vol. 2, p. 9, 2014.

[70] J. Zhang and S. Singh, "Visual-lidar odometry and mapping: Low-drift, robust, and fast," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 2174–2181, IEEE, 2015.

[71] J. Zhang and S. Singh, "Laser–visual–inertial odometry and mapping with high robustness and low drift," *Journal of Field Robotics*, vol. 35, no. 8, pp. 1242–1264, 2018.

[72] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet Classification with Deep Convolutional Neural Networks," in *Advances In Neural Information Processing Systems (NeurIPS)*, pp. 1–9, 2012.

[73] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *The International Conference on Learning Representations (ICLR)*, 2015.

[74] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention Is All You Need," in *Advances In Neural Information Processing Systems (NeurIPS)*, 2017.

[75] S. Zhang, L. Yao, A. Sun, and Y. Tay, "Deep learning based recommender system: A survey and new perspectives," *ACM Computing Surveys (CSUR)*, vol. 52, no. 1, pp. 1–38, 2019.

[76] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[77] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *The International Conference on Learning Representations*, 2015.

[78] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[79] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going Deeper with Convolutions," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015.

[80] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," in *The International Conference on Learning Representations (ICLR)*, pp. 1–10, 2015.

[81] A. Kendall, M. Grimes, and R. Cipolla, "Posenet: A convolutional network for real-time 6-dof camera relocalization," in *Proceedings of the IEEE international Conference on Computer Vision (ICCV)*, pp. 2938–2946, 2015.

[82] A. Kendall and R. Cipolla, "Geometric loss functions for camera pose regression with deep learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5974–5983, 2017.

[83] R. Clark, S. Wang, A. Markham, N. Trigoni, and H. Wen, "VidLoc: A Deep Spatio-Temporal Model for 6-DoF Video-Clip Relocalization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[84] S. Brahmbhatt, J. Gu, K. Kim, J. Hays, and J. Kautz, "Geometry-Aware Learning of Maps for Camera Localization," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2616–2625, 2018.

[85] S. Wang, R. Clark, H. Wen, and N. Trigoni, "DeepVO : Towards End-to-End Visual Odometry with Deep Recurrent Convolutional Neural Networks," in *International Conference on Robotics and Automation (ICRA)*, 2017.

[86] T. Zhou, M. Brown, N. Snavely, and D. G. Lowe, "Unsupervised Learning of Depth and Ego-Motion from Video," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[87] Z. Yin and J. Shi, "GeoNet: Unsupervised Learning of Dense Depth, Optical Flow and Camera Pose," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[88] H. Zhan, R. Garg, C. S. Weerasekera, K. Li, H. Agarwal, and I. Reid, "Unsupervised Learning of Monocular Depth Estimation and Visual Odometry with Deep Feature Reconstruction," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 340–349, 2018.

[89] R. Clark, S. Wang, H. Wen, A. Markham, and N. Trigoni, "VINet : Visual-Inertial Odometry as a Sequence-to-Sequence Learning Problem," in *The AAAI Conference on Artificial Intelligence (AAAI)*, pp. 3995–4001, 2017.

[90] K. Greff, R. K. Srivastava, J. Koutník, B. R. Steunebrink, and J. Schmidhuber, "LSTM: A search space odyssey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, pp. 2222–2232, Oct. 2017.

[91] A. Dosovitskiy, J. T. Springenberg, and T. Brox, "An Empirical Exploration of Recurrent Network Architectures," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, vol. 07-12-June, pp. 1538–1546, 2015.

[92] S. Hochreiter and J. J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1–32, 1997.

[93] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder–decoder for statistical machine translation," in *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 1724–1734, 2014.

[94] M. Schuster and K. K. Paliwal, "Bidirectional recurrent neural networks," *IEEE transactions on Signal Processing*, vol. 45, no. 11, pp. 2673–2681, 1997.

[95] A. Graves and N. Jaitly, "Towards End-to-End Speech Recognition with Recurrent Neural Networks," *International Conference on Machine Learning*, vol. 32, no. 1, pp. 1764–1772, 2014.

[96] A. M. Dai and Q. V. Le, "Semi-supervised Sequence Learning," in *Advances In Neural Information Processing Systems (NeurIPS)*, pp. 3079–3087, 2015.

[97] J. Donahue, L. A. Hendricks, S. Guadarrama, M. Rohrbach, S. Venugopalan, T. Darrell, and K. Saenko, "Long-term Recurrent Convolutional Networks for Visual Recognition and Description," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2625–2634, 2015.

[98] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, "Generative adversarial nets," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 2672–2680, 2014.

[99] A. Radford, L. Metz, and S. Chintala, "Unsupervised representation learning with deep convolutional generative adversarial networks," *arXiv preprint arXiv:1511.06434*, 2015.

[100] M. Arjovsky, S. Chintala, and L. Bottou, "Wasserstein generative adversarial networks," *The International Conference on Machine Learning (ICML)*, 2017.

[101] J.-y. Zhu, T. Park, A. A. Efros, B. Ai, and U. C. Berkeley, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2017.

[102] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired image-to-image translation using cycle-consistent adversarial networks," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2223–2232, 2017.

[103] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell, "CyCADA: Cycle-Consistent Adversarial Domain Adaptation," in *The International Conference on Machine Learning (ICML)*, pp. 1–12, 2017.

[104] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked Denoising Autoencoders: Learning Useful Representations in a Deep Network with a

Local Denoising Criterion Pierre-Antoine Manzagol," *Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[105] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in *The International Conference on Learning Representations (ICLR)*, pp. 1–14, 2014.

[106] M. Bloesch, J. Czarnowski, R. Clark, S. Leutenegger, and A. J. Davison, "CodeSLAM — Learning a Compact, Optimisable Representation for Dense Visual SLAM," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[107] K. Xu, J. Ba, R. Kiros, K. Cho, A. Courville, R. Salakhutdinov, R. Zemel, and Y. Bengio, "Show, Attend and Tell: Neural Image Caption Generation with Visual Attention," in *The International Conference on Machine Learning (ICML)*, 2015.

[108] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2019.

[109] T. Naseer and W. Burgard, "Deep regression for monocular camera-based 6-dof global localization in outdoor environments," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 1525–1530, IEEE, 2017.

[110] F. Walch, C. Hazirbas, L. Leal-Taixe, T. Sattler, S. Hilsenbeck, and D. Cremers, "Image-based localization using lstms for structured feature correlation," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 627–637, 2017.

[111] C. Chen, X. Lu, A. Markham, and N. Trigoni, "Ionet: Learning to cure the curse of drift in inertial odometry," in *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[112] R. Li, S. Wang, Z. Long, and D. Gu, "Undeepvo: Monocular visual odometry through unsupervised deep learning," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 7286–7291, IEEE, 2018.

[113] A. Valada, N. Radwan, and W. Burgard, "Deep auxiliary learning for visual localization and odometry," in *2018 IEEE international conference on robotics and automation (ICRA)*, pp. 6939–6946, IEEE, 2018.

[114] E. Brachmann and C. Rother, "Learning less is more-6d camera localization via 3d surface regression," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4654–4662, 2018.

[115] H. Yan, Q. Shan, and Y. Furukawa, "Ridi: Robust imu double integration," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 621–636, 2018.

[116] N. Yang, R. Wang, J. Stuckler, and D. Cremers, "Deep virtual stereo odometry: Leveraging deep depth prediction for monocular direct sparse odometry," in *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 817–833, 2018.

[117] C. Zhao, L. Sun, P. Purkait, T. Duckett, and R. Stolkin, "Learning monocular visual odometry with dense 3d mapping from dense 3d flow," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6864–6871, IEEE, 2018.

[118] C. Chen, Y. Miao, C. X. Lu, L. Xie, P. Blunsom, A. Markham, and N. Trigoni, "Motiontransformer: Transferring neural inertial tracking between domains," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 8009–8016, 2019.

[119] C. Tang and P. Tan, "Ba-net: Dense bundle adjustment network," *International Conference on Learning Representations (ICLR)*, 2019.

[120] E. J. Shamwell, K. Lindgren, S. Leung, and W. D. Nothwang, "Unsupervised deep visual-inertial odometry with online error correction for rgb-d imagery," *IEEE transactions on pattern analysis and machine intelligence*, 2019.

[121] M. R. U. Saputra, P. P. de Gusmao, S. Wang, A. Markham, and N. Trigoni, "Learning monocular visual odometry through geometry-aware curriculum learning," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 3549–3555, IEEE, 2019.

[122] Y. Almalioglu, M. R. U. Saputra, P. P. de Gusmao, A. Markham, and N. Trigoni, "Ganvo: Unsupervised deep monocular visual odometry and depth estimation with generative adversarial networks," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5474–5480, IEEE, 2019.

[123] Y. Li, Y. Ushiku, and T. Harada, "Pose graph optimization for unsupervised monocular visual odometry," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 5439–5445, IEEE, 2019.

[124] N. Piasco, D. Sidibé, V. Gouet-Brunet, and C. Demonceaux, "Learning scene geometry for visual localization in challenging conditions," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 9094–9100, IEEE, 2019.

[125] M. Brossard and S. Bonnabel, "Learning wheel odometry and imu errors for localization," in *2019 International Conference on Robotics and Automation (ICRA)*, pp. 291–297, IEEE, 2019.

[126] C. Chen, S. Rosa, Y. Miao, C. X. Lu, W. Wu, A. Markham, and N. Trigoni, "Selective sensor fusion for neural visual-inertial odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 10542–10551, 2019.

[127] Q. Li, S. Chen, C. Wang, X. Li, C. Wen, M. Cheng, and J. Li, "Lo-net: Deep real-time lidar odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8473–8482, 2019.

[128] W. Lu, Y. Zhou, G. Wan, S. Hou, and S. Song, "L3-net: Towards learning based lidar localization for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6389–6398, 2019.

[129] F. Xue, X. Wang, S. Li, Q. Wang, J. Wang, and H. Zha, "Beyond tracking: Selecting memory and refining poses for deep visual odometry," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8575–8583, 2019.

[130] R. Wang, S. M. Pizer, and J.-M. Frahm, "Recurrent neural network for (un-) supervised learning of monocular video visual odometry and depth," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5555–5564, 2019.

[131] L. Yang, Z. Bai, C. Tang, H. Li, Y. Furukawa, and P. Tan, "Sanet: Scene agnostic network for camera localization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 42–51, 2019.

[132] Z. Huang, Y. Xu, J. Shi, X. Zhou, H. Bao, and G. Zhang, "Prior guided dropout for robust visual localization in dynamic environments," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2791–2800, 2019.

[133] F. Xue, X. Wang, Z. Yan, Q. Wang, J. Wang, and H. Zha, "Local supports global: Deep camera relocalization with sequence enhancement," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2841–2850, 2019.

[134] M. Ding, Z. Wang, J. Sun, J. Shi, and P. Luo, "Camnet: Coarse-to-fine retrieval for camera re-localization," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2871–2880, 2019.

[135] S. Li, F. Xue, X. Wang, Z. Yan, and H. Zha, "Sequential adversarial learning for self-supervised deep visual odometry," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 2851–2860, 2019.

[136] L. Sheng, D. Xu, W. Ouyang, and X. Wang, "Unsupervised collaborative learning of keyframe detection and visual odometry towards monocular deep slam," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 4302–4311, 2019.

[137] M. R. U. Saputra, P. P. de Gusmao, Y. Almalioglu, A. Markham, and N. Trigoni, "Distilling knowledge from a deep pose regressor network," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pp. 263–272, 2019.

[138] L. Han, Y. Lin, G. Du, and S. Lian, "Deepvio: Self-supervised deep learning of monocular visual inertial odometry using 3d geometric constraints," *arXiv preprint arXiv:1906.11435*, 2019.

[139] M. Valente, C. Joly, and A. de La Fortelle, "Deep sensor fusion for real-time odometry estimation," *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2019.

[140] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition (CVPR)*, pp. 7794–7803, 2018.

[141] D. Barnes, W. Maddern, G. Pascoe, and I. Posner, "Driven to distraction: Self-supervised distractor learning for robust monocular visual odometry in urban environments," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1894–1900, IEEE, 2018.

[142] C. Chen, X. Lu, J. Wahlstrom, A. Markham, and N. Trigoni, "Deep neural network based inertial odometry using low-cost inertial measurement units," *IEEE Transactions on Mobile Computing*, 2019.

[143] C. Chen, P. Zhao, C. X. Lu, W. Wang, A. Markham, and N. Trigoni, "Deep-learning-based pedestrian inertial navigation: Methods, data set, and on-device inference," *IEEE Internet of Things Journal*, vol. 7, no. 5, pp. 4431–4441, 2020.

[144] D. Lymberopoulos, J. Liu, X. Yang, R. R. Choudhury, V. Handziski, and S. Sen, "A realistic evaluation and comparison of indoor location technologies: Experiences and lessons learned," in *Proc. Int. Conf. Inf. Process. Sensor Netw. (IPSN)*, (Seattle, Washington), pp. 178–189, 2015.

[145] F. Li, C. Zhao, G. Ding, J. Gong, C. Liu, and F. Zhao, "A reliable and accurate indoor localization method using phone inertial sensors," in *Proc. ACM Int. Conf. Ubiquitous Comput.*, (Pittsburgh, Pennsylvania), pp. 421–430, 2012.

[146] C. X. Lu, Y. Li, P. Zhao, C. Chen, L. Xie, H. Wen, R. Tan, and N. Trigoni, "Simultaneous localization and mapping with power network electromagnetic field," in *Proceedings of the 24th annual international conference on mobile computing and networking (MobiCom)*, pp. 607–622, 2018.

[147] S. Ahuja, W. Jirattigalachote, and A. Tosborvorn, "Improving Accuracy of Inertial Measurement Units using Support Vector Regression," tech. rep., 2011.

[148] X. Xiao and S. Zarar, "Machine Learning for Placement-Insensitive Inertial Motion Capture," *IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6716–6721, 2018.

[149] A. Parate, M. C. Chiu, C. Chadowitz, D. Ganesan, and E. Kalogerakis, "RisQ: Recognizing smoking gestures with inertial sensors on a wristband," in *Annual International Conference on Mobile Systems, Applications, and Services (MobiSys)*, pp. 149–161, 2014.

[150] A. Mannini and A. M. Sabatini, "Walking speed estimation using foot-mounted inertial sensors: Comparing machine learning and strap-down integration methods," *Medical engineering & physics*, vol. 36, no. 10, pp. 1312–1321, 2014.

[151] P. Hippe, *Windup in Control*. 2006.

[152] J. M. Hausdorff, "Gait dynamics, fractals and falls: Finding meaning in the stride-to-stride fluctuations of human walking," *Human Movement Sci.*, vol. 26, pp. 555–589, Aug. 2007.

[153] M. Lindfors, G. Hendeby, F. Gustafsson, and R. Karlsson, "Vehicle speed tracking using chassis vibrations," *IEEE Intelligent Vehicles Symposium*, vol. 2016-Augus, no. Iv, pp. 214–219, 2016.

[154] Z. Xiao, H. Wen, A. Markham, and N. Trigoni, "Robust indoor positioning with lifelong learning," *IEEE J. Sel. Areas Commun.*, vol. 33, pp. 2287–2301, Nov. 2015.

[155] W. Samek, A. Binder, G. Montavon, S. Lapuschkin, and K. Müller, "Evaluating the visualization of what a deep neural network has learned," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 28, pp. 2660–2673, Nov. 2017.

[156] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: Representing model uncertainty in deep learning," *The International Conference on Machine Learning (ICML)*, vol. 48, 2015.

[157] P. Agarwal, *Robust Graph-Based Localization and Mapping*. PhD thesis, University of Freiburg, 2015.

[158] J. L. Crassidis, "Sigma-point kalman filtering for integrated gps and inertial navigation," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 42, pp. 750–756, Apr. 2006.

[159] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *Advances in neural information processing systems*, pp. 6402–6413, 2017.

[160] A. Kendall and Y. Gal, "What uncertainties do we need in Bayesian deep learning for computer vision?," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 5574–5584, 2017.

[161] S. Yao, Y. Zhao, H. Shao, S. Liu, D. Liu, L. Su, and T. Abdelzaher, "FastDeepIoT: Towards Understanding and Optimizing Neural Network Execution Time on Mobile and Embedded Devices," in *The ACM Conference on Embedded Networked Sensor Systems (SenSys)*, 2018.

[162] F. Samie, L. Bauer, and J. Henkel, "From cloud down to things: An overview of machine learning in internet of things," *IEEE Internet of Things Journal*, vol. 6, no. 3, pp. 4921–4934, 2019.

[163] A. van den Oord, S. Dieleman, H. Zen, K. Simonyan, O. Vinyals, A. Graves, N. Kalchbrenner, A. Senior, and K. Kavukcuoglu, "WaveNet: A Generative Model for Raw Audio," in *Arxiv*, pp. 1–15, 2016.

[164] F. Yu and V. Koltun, "Multi-scale context aggregation by dilated convolutions," *The International Conference on Learning Representations (ICLR)*, 2016.

[165] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun, "Vision meets robotics: The KITTI dataset," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1231–1237, 2013.

[166] M. Burri, J. Nikolic, P. Gohl, T. Schneider, J. Rehder, S. Omari, M. W. Achtelik, and R. Siegwart, "The EuRoC micro aerial vehicle datasets," *International Journal of Robotics Research*, vol. 35, no. 10, pp. 1157–1163, 2016.

[167] D. Schubert, T. Goll, N. Demmel, V. Usenko, J. Stückler, and D. Cremers, "The TUM VI Benchmark for Evaluating Visual-Inertial Odometry," in *International Conference on Robotics and Automation (ICRA)*, 2018.

[168] S. Cortés, A. Solin, E. Rahtu, and J. Kannala, "ADVIO: An authentic dataset for visual-inertial odometry," in *The European Conference on Computer Vision (ECCV)*, 2018.

[169] M. Zhang and A. A. Sawchuk, "USC-HAD:A Daily Activity Dataset for Ubiquitous Activity Recognition Using Wearable Sensors," *The ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp)*, p. 1036, 2012.

[170] F. De La Torre, J. Hodgins, A. W. Bargteil, X. Martin, J. C. Macey, A. Collado, and P. Beltran, "Guide to the Carnegie Mellon University Multimodal Activity (CMU-MMAC) Database," Tech. Rep. April, 2008.

[171] R. Chavarriaga, H. Sagha, A. Calatroni, S. T. Digumarti, G. Tröster, J. D. R. Millán, and D. Roggen, "The Opportunity challenge: A benchmark database for on-body sensor-based activity recognition," *Pattern Recognition Letters*, vol. 34, no. 15, pp. 2033–2042, 2013.

[172] K. Siddhartha and W. Nicholas, "Evaluation of the performance of accelerometer-based gait event detection algorithms in different real-world scenarios using the MAREA gait database," *Gait and Posture*, vol. 51, pp. 84–90, 2017.

[173] Vicon, "ViconMotion Capture Systems: Viconn," 2017.

[174] Tango, "Google Tango Tablet," 2014.

[175] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 8024–8035, 2019.

[176] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in *International Conference on Learning Representations (ICLR)*, pp. 1–15, 2015.

[177] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *J. Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[178] J. Graeter, A. Wilczynski, and M. Lauer, "Limo: Lidar-monocular visual odometry," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 7872–7879, IEEE, 2018.

[179] J. Ngiam, A. Khosla, M. Kim, J. Nam, H. Lee, and A. Y. Ng, "Multimodal deep learning," in *International Conference on Machine Learning (ICML)*, pp. 689–696, 2011.

[180] D. Hu, X. Li, *et al.*, "Temporal multimodal learning in audiovisual speech recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3574–3582, 2016.

[181] C. Ding and D. Tao, "Robust face recognition via multimodal deep face representation," *IEEE Transactions on Multimedia*, vol. 17, no. 11, pp. 2049–2058, 2015.

[182] M. A. Lee, Y. Zhu, K. Srinivasan, P. Shah, S. Savarese, L. Fei-Fei, A. Garg, and J. Bohg, "Making sense of vision and touch: Self-supervised learning of multimodal representations for contact-rich tasks," in *International Conference on Robotics and Automation (ICRA)*, pp. 8943–8950, IEEE, 2019.

[183] G.-H. Liu, A. Siravuru, S. Prabhakar, M. Veloso, and G. Kantor, "Learning end-to-end multimodal sensor policies for autonomous navigation," *Conference on Robot Learning (CoRL)*, 2017.

[184] C. Hori, T. Hori, T. Y. Lee, Z. Zhang, B. Harsham, J. R. Hershey, T. K. Marks, and K. Sumi, "Attention-Based Multimodal Fusion for Video Description," *IEEE International Conference on Computer Vision (ICCV)*, vol. 2017-October, pp. 4203–4212, 2017.

[185] D. Pedreschi, F. Giannotti, R. Guidotti, A. Monreale, S. Ruggieri, and F. Turini, "Meaningful explanations of black box ai decision systems," in *The AAAI Conference on Artificial Intelligence (AAAI)*, vol. 33, pp. 9780–9784, 2019.

[186] J. Kim and J. Canny, "Interpretable learning for self-driving cars by visualizing causal attention," in *Proceedings of the IEEE international Conference on Computer Vision (ICCV)*, pp. 2942–2950, 2017.

[187] P. Fischer, E. Ilg, H. Philip, C. Hazırbas, P. V. D. Smagt, D. Cremers, and T. Brox, "FlowNet: Learning Optical Flow with Convolutional Networks," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[188] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778, 2016.

[189] X. Chen, H. Ma, J. Wan, B. Li, and T. Xia, "Multi-view 3d object detection network for autonomous driving," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1907–1915, 2017.

[190] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Machine learning*, vol. 8, no. 3-4, pp. 229–256, 1992.

[191] A. Mnih and K. Gregor, "Neural variational inference and learning in belief networks," in *The International Conference on International Conference on Machine Learning (ICML)*, pp. II–1791, 2014.

[192] E. Jang, S. Gu, and B. Poole, "Categorical reparametrization with gumble-softmax," in *International Conference on Learning Representations (ICLR)*, OpenReview. net, 2017.

[193] C. J. Maddison, A. Mnih, and Y. W. Teh, "The concrete distribution: A continuous relaxation of discrete random variables," *arXiv preprint arXiv:1611.00712*, 2016.

[194] C. J. Maddison, D. Tarlow, and T. Minka, "A* Sampling," in *Advances In Neural Information Processing Systems (NeurIPS)*, pp. 1–9, 2014.

[195] J. S. Hu and M. Y. Chen, "A sliding-window visual-IMU odometer based on tri-focal tensor geometry," in *International Conference on Robotics and Automation (ICRA)*, pp. 3963–3968, IEEE, 2014.

[196] C. Bishop, *Pattern Recognition and Machine Learning*. Springer, 2006.

[197] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon, "Scene coordinate regression forests for camera relocalization in RGB-D images," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2930–2937, 2013.

[198] J. Engel, V. Koltun, and D. Cremers, "Direct sparse odometry," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 3, pp. 611–625, 2017.

[199] A. Geiger, J. Ziegler, and C. Stiller, "Stereoscan: Dense 3d reconstruction in real-time," in *2011 IEEE Intelligent Vehicles Symposium (IV)*, pp. 963–968, Ieee, 2011.

[200] S. Leutenegger, S. Lynen, M. Bosse, R. Siegwart, and P. Furgale, "Keyframe-based visual–inertial odometry using nonlinear optimization," *The International Journal of Robotics Research*, vol. 34, no. 3, pp. 314–334, 2015.

[201] T. C. Wang, A. A. Efros, and R. Ramamoorthi, "Occlusion-aware depth estimation using light-field cameras," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 3487–3495, 2015.

[202] F. Couzinie-Devy, J. Sun, K. Alahari, and J. Ponce, "Learning to estimate and remove non-uniform image blur," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1075–1082, 2013.

[203] N. Naser, El-Sheimy; Haiying, Hou; Xiaojii, "Analysis and Modeling of Inertial Sensors Using Allan Variance," *IEEE Transactions on Instrumentation and Measurement*, vol. 57, no. JANUARY, pp. 684–694, 2008.

[204] Y. Ling, L. Bao, Z. Jie, F. Zhu, Z. Li, S. Tang, Y. Liu, W. Liu, and T. Zhang, "Modeling Varying Camera-IMU Time Offset in Optimization-Based Visual-Inertial Odometry," in *The European Conference on Computer Vision (ECCV)*, 2018.

[205] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell, "Adversarial Discriminative Domain Adaptation," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[206] M. Long, Y. Cao, J. Wang, and M. I. Jordan, "Learning Transferable Features with Deep Adaptation Networks," in *The International Conference on Machine Learning (ICML)*, vol. 37, pp. 1–20, 2015.

[207] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, "Domain-adversarial training of neural networks," *Journal of Machine Learning Research*, vol. 17, pp. 1–35, 2016.

[208] M. Zhao, S. Yue, D. Katabi, T. S. Jaakkola, and M. T. Bianchi, "Learning Sleep Stages from Radio Signals: A Conditional Adversarial Architecture," *The International Conference on Machine Learning (ICML)*, vol. 70, pp. 4100–4109, 2017.

[209] S. Purushotham, W. Carvalho, T. Nilanon, and Y. Liu, "Variational Recurrent Adversarial Deep Domain Adaptation," in *The International Conference on Learning Representations (ICLR)*, pp. 1–11, 2017.

[210] M.-Y. Liu, T. Breuel, and J. Kautz, "Unsupervised Image-to-Image Translation Networks," in *Advances In Neural Information Processing Systems (NeurIPS)*, 2017.

[211] E. Tzeng, J. Hoffman, T. Darrell, K. Saenko, and U. Lowell, "Simultaneous Deep Transfer Across Domains and Tasks," in *IEEE International Conference on Computer Vision (ICCV)*, 2015.

[212] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from Simulated and Unsupervised Images through Adversarial Training," in *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017.

[213] L. Yu, W. Zhang, J. Wang, and Y. Yu, "SeqGAN: Sequence Generative Adversarial Nets with Policy Gradient," in *The AAAI Conference on Artificial Intelligence (AAAI)*, pp. 2852–2858, 2017.

[214] J. Li, W. Monroe, T. Shi, S. Jean, A. Ritter, and D. Jurafsky, "Adversarial Learning for Neural Dialogue Generation," in *Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2017.

[215] T. Kim, M. Cha, H. Kim, J. K. Lee, and J. Kim, "Learning to Discover Cross-Domain Relations with Generative Adversarial Networks," in *The International Conference on Machine Learning (ICML)*, 2017.

[216] Z. Yi, H. Zhang, P. Tan, and M. Gong, "DualGAN: Unsupervised Dual Learning for Image-to-Image Translation," in *IEEE International Conference on Computer Vision (ICCV)*, pp. 2868–2876, 2017.

[217] Y. Taigman, A. Polyak, and L. Wolf, "Unsupervised Cross-Domain Image Generation," in *The International Conference on Learning Representations (ICLR)*, pp. 1–15, 2017.

[218] J. Wang, Y. Chen, S. Hao, X. Peng, and L. Hu, "Deep learning for sensor-based activity recognition: A survey," *Pattern Recognition Letters*, vol. 119, pp. 3–11, 2019.

[219] P. Kasebzadeh, G. Hendeby, C. Fritsche, F. Gunnarsson, and F. Gustafsson, "Imu dataset for motion and device mode classification," in *2017 International Conference on Indoor Positioning and Indoor Navigation (IPIN)*, pp. 1–8, IEEE, 2017.

[220] L. v. d. Maaten and G. Hinton, "Visualizing data using t-sne," *Journal of machine learning research*, vol. 9, no. Nov, pp. 2579–2605, 2008.