eman ta zabal zazu

Universidad del País Vasco
Euskal Herriko Unibertsitatea

INFORMATIKA FAKULTATEA
FACULTAD DE INFORMÁTICA

Contributions to LiDAR based SLAM and Computational Ethology

Marina Aguilar Moreno

Donostia - San Sebastián, June 2023

Supervised by:

Prof. Manuel Graña Romay - Computational Intelligent Group

Department of Computer Science and Artificial Intelligent

University of the Basque Country (UPV/EHU)

Abstract

This Thesis deals with two different topics centered about applications of Computational Intelligence techniques. The first topic is the implementation of simultaneous localization and mapping (SLAM) algorithms that are appropriate for low cost LiDAR sensors, specifically the Quanergy M8. Conventional and Deep Learning algorithms have shown shortcomings dealing with these data, hence this Thesis proposes a novel hybrid SLAM algorithm that achieves good results over in-house datasets captured with the low-cost LiDAR sensor. The second topic tackled in this Thesis is the discrimination of animal models on the basis of pressure signals. For this task, we work on real experimental data provided by a collaborating neurosciences team. The Thesis deals with the selection of signal features and the experimentation with a diversity of state of the art machine learning algorithms. The application of transfer deep learning upon signal spectrogram images improves significantly over conventional machine learning algorithms, concluding that it is feasible to discriminate animal models on the basis of pressure signal captured during locomotion periods.

Besides the global objectives of the Thesis a number of operational objectives have been achieved and reported, such as the set-up of Quanergy M8 LiDAR, the collection and publication of benchmark in-house datasets, the implementation and validation of the SLAM algorithms after tuning them, the segmentation of the pressure signals of the animal model, and the extensive experimentation carried out regarding diverse feature extraction and classification models.

Keywords: Computational Ethology, Animal Model Discrimination, LiDAR based SLAM, LiDAR based Navigation.

Acknowledgements

First of all I would like to express my sincere gratitude to my advisor Manuel Graña not only for having guided my path during these years of thesis but also for his support and advice. Despite the fact that my doctoral period has been marked by a period of uncertainty, we have been able to continue working to push the thesis forward. I would also like to thank him for sharing his knowledge and experience with me, which have made me grow as a person and researcher.

Secondly, I would like to thank to Professor Xavier Leinekugel for allowing me to spend my international stay in his research group at the Mediterranean Institute of Neurobiology in Marseille. This gratitude is extended specially to the members of the research group, Léna and Arnaldo, for making my stay so pleasant and inspiring.

I would also like to thank the members of the Computational Intelligence Group for all their support in both common and individual projects, especially Moisés, Igone, Guillermo, Goizalde and Anna, without your support it would have been much more difficult. Without forgetting my colleagues in the department for the good times, in particular Imanol and Elena.

I am grateful to the Spanish Ministry of Science, Innovation and Universities for support this Thesis through the PRE2018-085294 grant from FEDER funds through MINECO project TIN2017-85827-P, PID2020-116346GB-I00, and grant

IT1689-22 as university research group of excellence from the Basque Government.

Thanks also to my family for their infinitive support and affection always. To my parents and my brother, for their unconditional support. To my partner, for having come with me during this journey always with a smile. Thank you all for believing in me even in the worst moments.

Finally, I can not forget my friends, from Malaga and from San Sebastian. Thank you all for your support in this stage, you are also part of this.

Marina Aguilar Moreno

"El conocimiento no es una vasija que se llena,
sino un fuego que se enciende"

-Plutarco-

Contents

List of Figures

List of Tables

Nomenclature

AGV	Automated Guided Vehicle
AI	Artificial Intelligence
ALS	Airborne Laser Scanner
ALTM	Airborne Laser Terrain Mapper
AUC	Area Under the Curve
CE	Computational Ethology
CIG	Computational Intelligent Group
CNN	Convolutional Neural Network
CPD	Coherent Point Drift
DBH	Diameter at Breast Height
DRL	Deep Reinforcement Learning
ECG	Electrocardiogram
EEG	Electroencephalogram
EM	Expectation-Maximization

Fmr1-KO	Fmr1-knockout
GAN	Generative Adversarial Network
GMM	Gaussian mixture model
HRA	Hybrid Registration Algorithm
ICP	Iterative Closest Point
IMU	Inertial Measurement Units
INS	Inertial Navigation Systems
k-NN	k-Nearest Neighbors
LAI	Leaf Area Index
LFP	Local Field Potential
LiDAR	Light detection and ranging
LMS	Laser Measurement Systems
MARS	Mouse Action Recognition System
MAV	Micro Aerial Vehicles
MLP	Multi-layer Perceptron
MPA	Modified Procrustes Analysis
NDT	Normal Distribution Transform
ODE	Open Dynamics Engine
RANSAC	Random Sample Consensus
RFID	Radio Frequency Identification

RMSE	Root Mean Squared Error
ROS	Robot Operating System
SDK	Software Development Kit
SLAM	Simultaneous Localization and Mapping
SLOAM	Semantic LiDAR Odometry and Mapping
SPL	Single Photon LiDAR
SVM	Support Vector Machine
TIN	Triangulated Irregular Network
TLS	Terrestrial Laser Scanner
TOF	Time of Flight
UAS	Unmanned Aerial System
UAV	Unmanned Aerial Vehicles
UGV	Unmanned Ground Vehicle
URDF	Unified Robot Description Format
VHR	Very High Resolution
WT	Wild-Type

Chapter 1

Introduction

This Chapter provides the motivation and overall description of this Thesis. Section 1.1 provides some personal context that may allow to understand the evolution of the Thesis works and reported contributions. Section 1.2 provides some background information on the two main topics of the Thesis, and the motivation for the works. Section 1.3 refers the publications achieved while working in this Thesis. Section 1.4 details the objectives and contributions of the Thesis. Finally, Section 1.5 details the structure of the Thesis.

1.1 Research Personal Context

As many other researchers in the world, the pandemic of COVID-19 declared by the WHO has had a strong influence on the research track followed in the past years. The PhD grant started in 2019, few months before the pandemic. At this time, the Computational Intelligence Group had recently purchased a new LiDAR sensor that was advertised as innovative low cost medium resolution system. I started to work on the system, updating the operating system of the accompanying computer, and tuning it to achieve some recordings. We carried out several sessions of data recording in the upper floor of the Facultad de Informatica, and the two best recordings become our in-house experimental datasets. Next, I was working on the application of conventional and deep learning based approaches for SLAM over these datasets. An early surprise was that applying public domain deep learning approaches did not work at all over these datasets. So, we focused on traditional approaches. This work was interrupted by the pandemic declaration, that forced me to fly from San Sebastian, and continue work remotely during the year 2020 and a great part of 2021. With the background of the panic induced by the pandemic declaration and under the draconian regulations imposed by the political powers, we achieved to propose a hybrid algorithm that performed nicely on the most difficult in-house dataset.

During the remote working period, we were doing review research on Computational Ethology and related issues, in preparation for the intended core of the Thesis work. Early 2022, I was able to make a secondment with Dr. Xavier Leinekugel in Marseille, for three months. The goal of the stage was to be acquainted with the actual experimental tools and data of the host research team, in order to advance on the computational tools developed for the analysis of the observed data and to collaborate with the host research team in the generation of new ways to characterize animal models that can be of use in the evaluation of the effect of therapies and new drugs. This stage happened two years later than originally scheduled thanks to the political effects of the pandemic. The second half of the year 2022 and early months of 2023 have been devoted to the formulation and realization of the research work related to Computational Ethology in this Thesis.

1.2 Background and Motivation

In this Section we summarize the background and motivation for the two main topics covered in this Thesis.

1.2.1 SLAM Applications with LiDAR Data

Light detection and ranging (LiDAR) sensors have been used for scanning and reconstruction of indoor and outdoor environments [23], even in underground mining vehicles [209]. The fusion of LiDAR with GPS allows for large scale navigation [49] of autonomous systems. Simultaneous localization and mapping (SLAM) is a highly relevant process for autonomous systems. Accurate sensing provided by range sensors such as the LiDARs improve the speed and accuracy of SLAM, which can become an integral part of the control of innovative autonomous vehicles.

LiDAR sensors have been applied to various SLAM problems in conjunction with the tools commonly used in computer vision, such as loop closure [9], the removal of moving objects in dynamic environments [43], and Federated filtering of SIFT visual features for improve indoor mapping [118]. Low-cost LiDARs have been also installed in unmanned aerial vehicles for remote sensing, where they require non-rigid registration methods to register images and improve the measurement quality [120].

Recently, LiDAR based SLAM is becoming affordable by new sensors such as the M8 Quanergy LiDAR. However, these sensors offer less quality data and lower resolution (much less beams and less samples per second) that hinders the performance of registration methods. The Deep Learning based approaches seem to be sensitive to these data flaws. Specifically, in our experience a state-of-the-art Deep Learning based approach failed to produce meaningful results after several attempts to carry out transfer learning over a dataset collected indoors with one such affordable sensors. Consequently, traditional methods appear to be more likely to output better results than artificial intelligent based methods for these low-cost sensors. In this Thesis a comparison of three traditional registration methods applied to the path estimation followed by the LiDAR sensor is provided; namely, the Iterative Closest Points (ICP), Coherent Point Drift (CPD), and Normal Distributions Transform (NDT) registration methods. Moreover, a hybrid point cloud registration method is proposed to take advantage of the high accuracy provided by the classic ICP algorithm, and the robustness of the NDT registration method.

1.2.2 Computational Ethology

Ethology is defined as the discipline that studies the animal behavior in terms of its phenomenological, causal, ontogenetic and evolutionary aspects, in order to provide answers to the causes and development that animal behavior undergoes, as well as to understand how it is performed [8], bearing in mind that behavior is understood as the set of muscular responses of a living being as a consequence of an external stimulus and internal motivation [73]. In this way it is possible to extract behavioral characteristics and study their alterations due to diseases or disorders. In addition, it is now possible to generate animal models with genetic modifications that provide study subjects with anatomy, physiology or response to a pathogen that are sufficiently similar to humans to be able to extrapolate the results obtained to them, very useful in Pharmacology to test new medicines. The most commonly used models in research are rodents, especially mice and rats, zebra fish, amphibians and reptiles, birds and other small animals. Early ethology studies were conducted visually, describing what researchers saw in each experiment qualitatively. Later on, they began to evaluate certain behaviors on the basis of some predefined criteria, thus beginning a quantitative approach.

Computational ethology is the study of animal behavior using Computer Vision and Artificial Intelligence (AI) techniques to help quantify and analyse behavioral patterns [44]. This is usually done by studying animals in free-ranging environments and analysing specific behaviors, or calculating ethograms, which describe how often each action is performed and the probability of the next action being performed, in order to try to find out how decision-making is carried out [8]. Therefore, Computational Ethology allows the incorporation of advances in Computer Vision and AI in the study of animal behavior.

1.3 Publications Produced During the PhD Thesis

During this PhD Thesis we have produced the following publications:

1. Aguilar-Moreno, M., Graña, M. (2021). A Comparison of Registration Methods for SLAM with the M8 Quanergy LiDAR. In: Herrero, Á., Cambra, C., Urda, D., Sedano, J., Quintián, H., Corchado, E. (eds) 15th International Conference on Soft Computing Models in Industrial and Environmental Applications (SOCO 2020). Advances in Intelligent Systems and Computing, vol 1268. Springer, Cham. https://doi.org/10.1007/978-3-030-57802-2_79

2. Aguilar-Moreno, M., Graña, M. (2020). An Hybrid Registration Method for SLAM with the M8 Quanergy LiDAR. In: de la Cal, E.A., Villar Flecha, J.R., Quintián, H., Corchado, E. (eds) Hybrid Artificial Intelligent Systems. HAIS 2020. Lecture Notes in Computer Science(), vol 12344. Springer, Cham. https://doi.org/10.1007/978-3-030-61705-9_3

3. Graña M, Aguilar-Moreno M, De Lope Asiain J, Araquistain IB, Garmendia X. (2020). Improved Activity Recognition Combining Inertial Motion Sensors and Electroencephalogram Signals. Int J Neural Syst. 2020;30(10): 2050053. https://doi.org/10.1142/S0129065720500537

4. Aguilar-Moreno, M., Graña, M. (2022), On registration methods for SLAM with low resolution LiDAR sensor, Logic Journal of the IGPL; jzac037, https://doi.org/10.1093/jigpal/jzac037

5. Aguilar-Moreno, M., Graña, M. (2023). Computational Ethology: Short Review of Current Sensors and Artificial Intelligence Based Methods. In: Iliadis, L., Maglogiannis, I., Alonso, S., Jayne, C., Pimenidis, E. (eds) Engineering Applications of Neural Networks. EANN 2023. Communications in Computer and Information Science, vol 1826. Springer, Cham. https://doi.org/10.1007/978-3-031-34204-2_2

6. Aguilar-Moreno, M., Graña, M. (2023), Phenotype Discrimination based on pressure signals by transfer learning approaches, International Work-Conference on Artificial Neural Networks (IWANN 2023), accepted.

1.4 Objectives and Contributions of the PhD Thesis

1.4.1 LiDAR based SLAM

The work carried out on LiDAR based SLAM algorithms was intended originally to test the feasibility of using a low-cost LiDAR sensor for indoor SLAM. An operational objective was to set up the system and achieve data recordings of some quality. Another operational objective was to test public domain LiDAR SLAM algorithm implementations over the datasets recorded in-house in order to validate their usefulness for the task. A final objective was to propose a robust algorithm that could cope with the difficulties found by conventional algorithms on this data.

The contributions of the Thesis in this topic are as follows:

- We have carried out a review of the literature regarding LiDAR based SLAM as well as other applications of SLAM. This review is partially reproduced in Chapter 2, but in its extensive form will be submitted for publication in journal after the Thesis dissertation.

- We have setup the low-cost LiDAR system, dealing with non trivial software obstacles that needed the upgrade of the operating system in a very closed setting, with no help from the selling company. This system is currently operational and available for the use of members in the research group.

- We have carried out the data capture of two in-house datasets that support the computational experiments reported in this Thesis. These datasets are available open access data in two different Zenodo entries [1][2]. One of the datasets has been downloaded more than 300 times, and the other more than 100 times.

- We have evaluated deep learning architectures over these in-house datasets. The results achieved were of low quality, some of them are reported here in order to communicate them.

- We have proposed a novel hybrid algorithm that improves over conventional state of the art LiDAR based SLAM algorithms on these in-house datasets.

[1]https://doi.org/10.5281/zenodo.4302360
[2]http://doi.org/10.5281/zenodo.4302366

1.4.2 Computational Ethology

In this Thesis we have analysed data recorded with a multisensor system composed of a top video camera and the Phenotypix platform, which is a piezoelectric pressure sensor that records the movement of animals, developed by Dr. Leinekugel and his research team. Specifically, this Thesis work has been focused on answering the following research question: Is it possible to discriminate animal models by using Artificial Intelligence algorithms to process the piezoelectric pressure signal?

Such a general research question is rather difficult to deal with. We have further constrained it to the following question: Is it possible to discriminate animal models by using Artificial Intelligence algorithms to process the piezoelectric pressure signal extracted during the locomotion periods?

In order to attack the problem, a first operational objective is carry out the locomotion periods of the signal. Another operational objectives are to find a signal representation that is amenable to the analysis, and to test the diverse machine learning algorithms that are available in the state of the art.

The contributions of the Thesis in this topic are as follows:

- We have extracted the locomotion periods on the basis of the analysis of the animal trajectories extracted from the video recordings in an automatic way. These periods are applied to the piezoelectric signal to extract the corresponding signal chunks.

- We have proposed and tested several ways to compute the piezoelectric signal spectrograms of the locomotion periods.

- We have tested several feature extraction algorithms (detailed in Chapter 7) that can be used as input by the machine learning algorithms.

- We have tested and evaluated an exhaustive collection of machine learning algorithms, including the application of deep transfer learning approaches in the discrimination of the animal models based on the pressure signal corresponding to locomotion periods. The conclusions are positive: it is possible to discriminate animal models by using their locomotion pressure signals.

1.5 Structure of the PhD Thesis

The PhD Thesis has the following structure:

- Chapter 2 provides a review of the state of the art of LiDAR based algorithms, including 3D reconstruction as well as SLAM for navigation applications.

- Chapter 3 describes the collection of data in an in-house effort to build resources for SLAM experimentation. It also contains some preliminary results on the failed application of deep learning approaches to this data that motivates the used of classical approaches.

- Chapter 4 Chapter gives the detailed description of SLAM algorithms applied to the data described in the previous Chapter 3.

- Chapter 5 reports the results of computational experiments performing SLAM over the data collected and described in Chapter 3.

- Chapter 6 provides a quick review of the state of the art of Computational Ethology in order to set the stage for the reported experiment.

- Chapter 7 contains the description of the actual data and methods applied in the Computational Ethology experiment. The Chapter describes the animal models, the generation of the signal, the segmentation process, the feature extraction processes, and the classification methods applied to the signal.

- Chapter 8 reports the results of the Computational Ethology experiments. It contains a summary with the results of all computational experiments in terms of classification performance.
- Chapter 9 provides some overall conclusions and proposals of future work for the two lines of research encompassed by this Thesis.

Chapter 2

Introduction to SLAM Applications with LiDAR

This Chapter describes some of the fundamentals of LiDAR sensors, reviewing the state of the art related to the processing approaches for LiDAR data, their applications, and the best known public point cloud databases used for the validation of computational approaches. The Chapter has four main parts. Section 2.1 will explain the LiDAR fundamentals in order to know how LiDARs work and the factors that affect their sensibility. This Section also includes a short summary of the LiDAR brands found in the market, and the platforms and sensors that can be used together with the LiDARs. Section 2.2 provides a review of the applications of LiDAR sensors. Even though this Thesis will be focused on Simultaneous Localization and Mapping (SLAM) applications and algorithms, we include other applications and approaches in this review. Section 2.3 reviews computational approaches developed for LiDAR data processing. Finally, Section 2.4 will go through the most relevant LiDAR databases so far, by using the indexed citation number for each contribution.

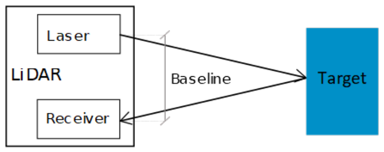


Figure 2.1.1: Triangulation ranging principle.

2.1 LiDAR Fundamentals

Light detection and ranging (LiDAR) is a remote measurement technique based on a light beam that computes distances from the sensor to surrounding obstacles. Once the surrounding obstacles are known, a point cloud is generated that locates all objects found in the LiDAR field of view.

LiDARs can be classified into 3 groups depending on its working principle: optical triangulation, phase difference, and time of flight. The first approach is illustrated in the figure 2.1.1 and it is based on geometrical principles. If three measurements are known in a triangle, it is possible to know any point in the triangle. In this method, the distance is calculated with the separation between the transmitter and the receiver, i.e. baseline, and the both angles with the baseline. This method is focused for short-range measurements as its accuracy decreases with the distance to the target ($\sim R^2$).

Phase difference technology obtains the target distance by means of the phase difference between the transmitter and the receiver taking into account the number of complete wave cycles. The range is computed as half the spatial wavelength of the carrier frequency and the range resolution depends on the modulation frequency and the phase difference resolution [50]. In the figure 2.1.2 is exemplified this physical principle.

Finally, the principle of Time-of-Flight (TOF) technology is based on the light velocity equation and it is shown in the figure 2.1.3. This kind of LiDARs emit a laser beam with a predetermined azimuth and inclination angle towards

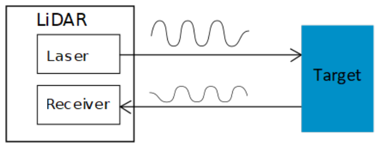


Figure 2.1.2: Phase difference principle.

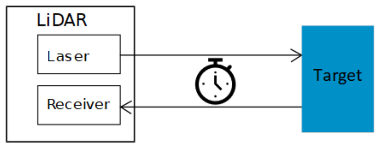


Figure 2.1.3: Time-of-Flight principle.

a surface. The distance between the sensor and this surface is computed by timing the time of flight and measuring the intensity of the light that returns. For this, the laser emits thousands of light pulses per second and adjusts the horizontal and vertical direction to sweep the entire field of vision offered by the device (usually 360º in the horizontal plane and 90º in the vertical plane). The range is only limited by the energy dispersion, hence this kind of LiDARs can be used in short and long range applications, being the most used in research.

The TOF LiDAR accuracy depends on the sensibility of the detector but there are some parameters that can affect LiDAR efficiency [100]:

- The reflectivity of the target material such as color or roughness.
- The inclination of the angle with which the laser hits the surface.
- The distance to the target, that is, the further the pulse reaches, the wider and less intense the light will be.
- Atmospheric conditions: rain, fog, humidity or dust.

Figure 2.1.4: Velodyne HDL-32E LiDAR.

2.1.1 Main LiDAR Brands

There are currently many different brands of LiDAR on the market with a wide range of specifications depending on the application. Within each manufacturer, there exist distinct models depending on the number of beams, precision or weight. For instance, Velodyne brand has the VLP-16, HDL 32-E or HDL 64-E models with 16, 32 and 64 beams respectively, and all of them are used in SLAM, modeling, localization or reconstruction tasks. Furthermore, Velodyne has lighter models as Puck VLP-16 or Puck-Lite VLP-16 for aerial applications. The figure 2.1.4 shows one of the most used commercial LiDAR, the Velodyne HDL-32E[1].

SICK LiDARs also have several models for SLAM or place recognition such as LMS1XX for indoor use and LMS5XX for outdoor use. Moreover, SICK NAV2XX models are designed to obtain the position in navigation tasks. Hokuyo LiDARs are used for SLAM, localization, mapping and reconstruction in Robotics, with ROS[2] (Robot Operating System) support. UTM-30LX-XX sensors are suitable for intelligent robots and applications with high velocity due to its range and fast response and URG-04LX-XXX models are suitable for autonomous navigation and mapping. Leica LiDARs are designed for airborne applications and its ALS model (airborne laser scanner) or the SPL model (single photon LiDAR) are suitable for surveying and forestry applications. Optech

[1]https://velodynelidar.com/products/hdl-32e/
[2]https://www.ros.org/

is another LiDAR brand and with its ALTM (airborne laser terrain mapper) model is able to detect and reconstruct scenarios or estimate tree top height from UAVs or airborne. Riegl LMS (laser measurement systems) LiDAR has several models depending on the application, such as QXXX, for long range airborne laser scanner, which is used for tree segmentation, reconstruction or terrain comparison between meteorological phenomena, or VZ-XXX, for terrestrial tasks, for instance, tree reconstruction or sidewalk inventory. RPLidar is a brand from the ROS component that can be used in autonomous navigation or localization, installed in mobile robots or in Micro aerial vehicles, due to its lightweight.

2.1.2 LiDAR Configurations

LiDAR sensors can been installed in a variety of platforms such as ground vehicles, autonomous robots, flying aircrafts and Unmanned Aerial Vehicles (UAV) , satellites, and some of them can also be operated manually. System configuration depends on the application. In ground vehicles, systems include several kinds of sensors, each fulfilling a role for autonomous driving or driving assistance, including several CCD cameras, sonar, and radar as well as the LiDAR system. Most popular devices for autonomous driving applications are Velodyne LiDARs, which have up to 64 sensing planes. Airborne LiDAR are deployed for the creation of digital elevation maps, forestry management and urban planning. UAVs can be used for agriculture and military missions. Manually operated LiDARs are often used for topological measurements and for human navigation.

There exist certain commercial platforms that have been used to carry out experiments with LiDARs such as the Pioneer 3AT mobile platform, the DiddyBorg six wheeled robot, the ROS Turtlebot mobile platform, or the Husky-A200 series ground vehicle. Moreover, LiDARs are often accompanied by other sensors to improve their accuracy in experiments. These sensors can be cameras (stereo, reflex, RGB / Gray, Kinect, ...), Inertial Measurement Units (IMU), satellite navigation systems (GNSS, GPS), Inertial Navigation Systems (INS) or encoders.

2.2 LiDAR Applications

LiDAR is used in a variety of applications due to its precision, large range, and its ability to process data in real time. The main applications are related to SLAM, robotics, remote sensing, autonomous driving, and 3D mapping, among others.

2.2.1 LiDAR SLAM and Robotics Applications

SLAM algorithms are able to create a map and locate themselves by registering LiDAR point clouds in consecutive time instants. The main field of application of SLAM is Robotics, achieved by mounting the LiDAR sensor in aerial or terrestrial vehicles such as UAV, Automated Guided Vehicle (AGV), and Unmanned Ground Vehicle (UGV).

Examples of applications combining UAV with SLAM are the monitoring of sugarcane crop by using photogrammetry equipment, inertial measurement systems or navigators [178], and, in forest spaces, with a quadcopter and a backpack to have different perspectives [161]. Similarly, AGVs use LiDAR information to carry out indoor SLAM in semiconductor factories where following marks are not available [36], in GNSS-denied environments [78] and for rescue tasks in mining environments [122]. In forestry applications, UGV can be used to map the environment with algorithms such as Graph-SLAM [160] or SLOAM (Semantic LiDAR Odometry and Mapping), which creates also cells for each detected tree [33].

2.2.2 Remote Sensing LiDAR Applications

LiDAR sensors are widely used in remote sensing due to their large range (50m - 300m). Within this category, applications such as forestry measurements and classification, urban landscapes creation or industrial measurement and inventory are the most common.

2.2.2.1 Forestry Measurement and Classification

Airborne systems like ALS or terrestrial systems such as Terrestrial Laser Scanner (TLS) are often used to carry out the measurements necessary to analyse forestry data. Current applications of LiDAR in forest lands are tree species classification, based on hyper-spectral and airborne LiDAR data together with tree species measurements [142] and tree crown segmentation [218], where in a first step false tree crowns are eliminated.

Inventorying a crop field helps to predict the crop yield and to plan the harvesting of fruits. For this purpose, the number of trees and their geometrical parameters, such as tree height or canopy base height, are calculated [77]. In the case of tropical forests, there are other parameters such as Leaf Area Index (LAI) or Diameter at Breast Height (DBH) [91, 46] that are computed to monitor the health of the forest. Finally, there are methods to estimate forest biomass for cartographic studies using airborne LiDAR by applying regression techniques for estimation and prediction [190, 188].

2.2.2.2 Urban Landscapes Creation

LiDAR sensing has acquired relevance in urban reconstruction too. Many practical applications make use of airborne LiDAR data to create urban landscapes by means of intelligent algorithms.

For instance, there are applications for the reconstruction of building roofs, which use a Triangulated Irregular Network (TIN) model to detect outliers and extract the roof structure in the form of a grid and then project it onto a label map in order to refine it and obtain the final surface [123]. Another approach to reconstruct urban scenarios is clustering with aerial LiDAR data to separate the roofs in different parts or known models to refine the surface [165].

Classification algorithms are also used in urban landscapes generation to ease the reconstruction. As an example, it is possible to use a decision tree classifier to assign a label to every building collected with the aerial LiDAR [227] or classify these point clouds into trees, buildings and ground to reconstruct only objects of interest [228]. To deal with the large amount of LiDAR data, reduction method has been developed using feature extractors and optical image [212].

2.2.2.3 Industrial Measurement and Inventory

Laser based metrology is used in many industrial processes. For instance, in steel roller mills, sensors for surface reconstruction are capable of detecting millimeter-scale surface deformations [7]. Another LiDAR application is the inventory of luminaires in buildings, where the ceiling is segmented and the point clouds are converted into binary images to detect the light points and classify them into different types [51].

2.2.3 Autonomous Driving

Autonomous driving requires real-time processing to manage dynamic and static obstacles during movement and a wide field of view to scan the environment and create a 3D map. For this reason LiDAR sensors are widely used in this field to carry out real-time navigation and road inventory.

2.2.3.1 Real Time Navigation

Apart from the advantages of LiDAR, there are other reasons to use it, for example in locations where GNSS signal is not accessible. In fact, there exists a system composed of a LiDAR and a camera, capable of detecting road lines and measuring the distance to them to improve navigation with a known map [172].

In mining environments, where satellite signal is unavailable, Micro Aerial Vehicles (MAV) can be equipped with a LiDAR in order to improve autonomous navigation while classifying images with a Convolutional Neural Network (CNN) [137]. In aerospace tasks, LiDAR is an interesting option for navigation, where simulations are really welcome due to the experimental high costs [150].

LiDAR sensing is used also for guiding disabled people, who can use a walking stick with the pre-loaded map that recalculates the trajectory in real time taking into account the obstacles found during navigation [136]. Others recent applications for navigation with LiDAR can take advantage of intelligent algorithms based on Deep Reinforcement Learning (DLR) [200].

2.2.3.2 Road Inventories

Road inventory is key to improve the management and development of the cities. One example of this application uses a TLS placed on the hood of a car to collect data while driving. This way, LiDAR point clouds can be processed offline in order to obtain road maps with the surface, the central line and the boundary line [90]. Other than roads, a sidewalk inventory can be done using LiDARs, being useful to improve the maintenance plans and as a map for wheelchairs users [84].

2.2.4 3D Mapping

Environment mapping is an important step for autonomous navigation. In this sense LiDARs are also crucial to obtain precise maps with high quality even in GNSS-denied environments [47], in both static and dynamic scenarios [115]. Besides map creation, LiDAR point clouds segmentation and classification are also possible giving labels such as buildings, trees, roads, sidewalks or traffic signals outdoors [226], or ceiling, walls, floor, columns and cars in underground car parks [75]. In agriculture, LiDAR mapping is also an interesting tool to map surfaces from the air and obtain the location of the trees in an apple crop by using depth measurements and computing treetops centroids [77].

Due to its low sensibility with lighting variations, LiDAR sensors are an option to carry out measurements in caves and mines, where GPS signal is denied. One possibility would be a mobile platform equipped with 2 LiDARs to generate maps with high precision odometry [152] or an helicopter with a LiDAR to map the cavity profile [143].

Surveying is another application for LiDARs thanks to their high resolution and lightness, and these sensors can also be combined with Texel cameras into a Unmanned Aerial System (UAS) to obtain better results in reconstruction without the need to use high quality navigation systems [21].

In this field of application, the analysis of coastal barriers before and after a storm can be studied with an aerial LiDAR, in terms of the recovery and the impact of meteorological phenomena [96]. Finally, Archaeology requires airborne LiDAR also to visualize footprints and walking paths from depth data collected from air [196].

2.3 Computational Approaches

LiDAR point clouds allow to obtain useful information about the environment to carry out tasks such as localization, mapping, SLAM, classification, alignment, object extraction, semantic map generation, detection, segmentation and reconstruction. In this section different approach to process point could sets are described, distinguishing two kinds of methods: traditional and artificial intelligent based methods.

2.3.1 Traditional Methods

Traditional methods are the beginning methods to process point clouds, being the Iterative Closest Point (ICP) [15] the earliest and one of the most known methods to align point clouds. Furthermore, there exist variations of the ICP algorithm that are also used in LiDAR data processing, such as Generalized ICP (G-ICP) [175] or Ground Plane ICP (PG-ICP) [104]. The former is used in environment reconstruction [35] and the latter is used in SLAM applications along with ranging beacons [64] that are installed in the environment allowing to select the best candidates for loop closure.

Other methods are Procrustes Analysis (PA) and its variations such as Modified Procrustes Analysis (MPA) [180] that are used in curved building reconstruction, because it uses an affine transformation matrix instead of a rigid one.

Normal Distributions Transform (NDT) [135] is another example of point cloud set alignment method that is very robust and gives excellent results in SLAM and autonomous driving. A variant of NDT method is known as weighted-NDT [115] and it associates a weight to each object detected by the LiDAR, depending on the static probability to be mapped in a dynamic environment, modifying the original algorithm.

Apart from these algorithms, other interesting algorithms have also emerged to process point clouds such as the Hector SLAM algorithm [108] for bridge inspection [157] and to validate new SLAM implementations [32]. Moreover, there exist algorithms based on the Directed Geometric Point [127] or integrated in the mobile mapping system [124] for complex urban environments.

Related to localization, the Monte-Carlo localization [61] is an algorithm widely used not only as the method itself but also as a base for some variants such as the Adaptive Monte-Carlo localization [157] or the Self-adaptive Monte-Carlo localization [214] methods. In the first variation, the map is created using Hector SLAM and then, a mobile robot navigates by means of Adaptive Monte-Carlo Localization to carry out bridge bearing inspections. Moreover, other localization algorithms exist, for instance, to work in severe meteorological conditions or structural changes in the environment [24]. In a second example, the LiDAR is placed on a tripod and remains static all the time to obtain landmark information. With the 5 best matches are obtained, an ICP stage is implemented to refine the matching and choose the best match [110].

Likewise, many methods have been developed for environment mapping such as LOAM [220], LeGO-LOAM [177], A-LOAM[3] or R-LOAM [149], designed specifically for LiDAR data processing and as ground truth for new algorithms.

Relative to the first method, LOAM can create large-scale maps using collaborative robots and then, merging the map pieces generated by each robot [222]. Similarly to other traditional methods, LOAM has also been used as a base for new implementations, such as LOCUS (LiDAR Odometry for Consistent operation in Uncertain Settings), to improve the odometry and the mapping in Real-Time [152].

Furthermore, the LOAM algorithm can be used in environments where the GNSS signal is partially denied [78]. However, another approach is recently implemented for these challenging situations, which develops a virtual GNSS

[3]https://github.com/HKUST-Aerial-Robotics/A-LOAM

based on a particle filter to generate the global position and loop closure [47].

Another example of mapping application with LeGO-LOAM is the implementation of an interactive software for map correction. Initially this algorithm generates the maps and then, the aforementioned G-ICP method refines the results [109].

Finally, a traditional method that deserves mention is Random Sample Consensus (RANSAC) [60]. Despite this algorithm it is not made for aligning point clouds, it is very useful in preprocessing tasks to filter outliers [106] and detect the ground in forest mapping [160], and in postprocessing to remove false matching [21].

2.3.2 Artificial Intelligent Based Methods

Artificial Intelligence plays an important role in point clouds processing due to the increase of the computation capacity and velocity. Firstly, Machine Learning based methods are composed by algorithms such as Support Vector Machines, Random Forest and K-Nearest Neighbor, among others. Secondly, Neural Networks, which includes Convolutional Neural Networks and Deep Neural Networks, are also really used in LiDAR data processing. Finally, Reinforcement Learning also deserves mention since it is used for route planning in autonomous navigation.

2.3.2.1 Machine Learning Based Methods

There are several algorithms applicable to point clouds for classification, reconstruction, characterization, object detection or parameter estimation. These algorithms are Support Vector Machine (SVM) [151], Random Forest, k-Nearest Neighbor (k-NN) [41] , Decision Trees or k-Nearest Neighbor Trees.

SVM uses LiDAR data in several autonomous driving applications for object recognition or classification tasks. For example, point clouds can be classified as pedestrian or not pedestrian (among other objects such as vegetation, light poles, traffic signs or parts of buildings) with SVM [199]. To classify the localization status as success or failure during autonomous driving is also possible by means of two parameters and the pose of the robot [105]. Another example of point clouds classification is the traffic sign detection, where LiDAR data are projected in 2D images and then a hierarchical classification is carried out to obtain semantic information of the signal [179].

Reconstruction tasks from LiDAR data can also be implemented with an SVM. In the case of power pylons, point clouds are processed to obtain the contour maps of the pylon and then the histograms of oriented gradients (HOG) features by projecting them in an image. The SVM stage is fed with these HOG features to classify and assign a category to the pylon head [198].Related to roof reconstruction, point clouds can be classified in building, ground or vegetation with a SVM [48].

Apart from SVM, decision trees can also classify buildings depending on physical and morphological parameters from the LiDAR data [227]. Random Forest [20], which is a combination of prediction trees where each tree votes for the most popular class at input, allows to extract and generate a network road using LiDAR data and very-high-resolution (VHR) aerial images. After pre-processing the LiDAR point clouds into a normalized digital surface model (nDSM), the data are merged with VHR images to create image objects. Then, a random forest classifies them into road, shadow, tree, car, building or bare groups to generate the road network. Once the false segments have been filtered out, the road center-line can be generated. [223]. Another example of a random forest classifier uses airborne LiDAR data and satellite image fusion. Having extracted the features by a Convolutional Neural Network, a random forest classifies the roof shape with the help of a dataset of roof images [28].

In addition to feature classification, random forests also allow the characterization of objects with LiDAR data. An example is the estimation of measured canopy structure: canopy cover and height for tree forest classes [3]. Moreover, k-Nearest Neighbor Trees can also estimate tree top height using LiDAR data and VHR images. This method obtains the height of the trees as an average of the heights of neighboring trees with similar properties [154].

As for k-Nearest Neighbor, one application is the classification of barrier island habitats using two parameters: the elevation of the terrain and the distance from the ocean-facing shore measured with an aerial LiDAR. This classifier distinguishes between beach, dune, woody vegetation or marsh and it can also be implemented using SVM and random forest [55].

2.3.2.2 Neural Networks Based Methods

Pre-processing Unlike machine learning-based methods, neural networks need to modify point clouds to make them more manageable objects because of its irregularity, lack of structure, and disorder. Point clouds also tend to have variable densities which greatly hinders their processing. To deal with these challenges, there are two types of approaches: converting point clouds to structured data types or working with raw data.

In the first approach, point clouds can be pre-processed by means of the voxelization, where each point becomes a voxel (i.e., a 3D pixel), generating a volume. The disadvantages of this method are its high memory consumption and the introduction of artifacts due to the voxelization process. Another option would be the image generation from the point cloud by projecting them from different angles. This method offers better performance than voxelization in terms of performance and artifacts and also benefits from all Computer Vision methods. As an example of this approach, there is a segmentation algorithm for vehicle detection that uses two views of the same point cloud, the front view and the bird's eye view, which are merged and sent to a clustering algorithm for bounding box extraction and vehicle tracking [192].

In the second approach, neural networks are fed with raw point clouds. The PointNet neural network [81] was the first application that used unstructured point clouds directly in Convolutional Neural Networks and is built with 2 symmetric functions, a Multi-layer Perceptron (MLP) and a maxpooling function. Symmetric functions are functions whose output is not dependent on the input order, opening a new way to develop applications with point clouds without the pre-processing stage. PointNet has also been used as feature extractor in various applications [103] and has served as a basis for other neural networks, such as PointNet++ [167]. The novelty of this neural network is its hierarchical layer, which makes it better at extracting small parts, for instance in road environments for pavement inventory, where the PointNet++ neural network segments the point clouds and the next neural network extracts and merges all the pieces of the pavement [84].

Consequently, there are more developments that work directly with the raw point cloud. The following examples feed neural networks directly with the LiDAR data.

Applications LiDAR data can be used in classification tasks, using a feature extractor in the early stages of implementation using a stand-alone neural network [25] or using the first layers of a Convolutional Neural Network [142]. In the DANCE-NET neural network, a convolution operation is used for feature extraction together with a kernel density estimation to deal with the problem of variations in the density of the point clouds. Finally, in the classification stage, the following categories can be differentiated: roof, facade, car shrub, tree, powerline, low vegetation or fence [125].

Point clouds can also be segmented to carry out an inventory of forests, where a Deep Neural Network implements feature extraction and semantic segmentation, creating the Semantic LiDAR Odometry and Mapping for Forest Inventory (SLOAM) network for tree diameter estimation [33]. Moreover, a convolutional neural network can detect potential moving objects and obtain the pose in a SLAM application [203] or in the Liseg network where point clouds are condensed into 2D matrices [226].

Detection tasks can also be implemented by applying neural networks to airborne LiDAR data for applications such as detection and reconstruction of hollow roads for archaeological domains using a ResNet-34 convolutional neural network [194]. Another application can detect and extract residential buildings with Deep Neural Network fed with gray-scale images obtained from point clouds. In this application, the color value is the relative elevation of the point and a Convolutional Neural Network distinguishes between buildings and other objects such as vegetation [229].

In addition to static objects, the detection tasks can be applied to moving objects such as people and, through a Convolutional Neural Network based recognition algorithm, distinguishing between certain previously defined activities [13]. For localization single-line, LiDAR data can be used to feed an artificial neural network search tree [204]. Besides, LiDAR data can also be compared with satellite image to obtain vehicle localization, where a Convolutional Neural Network implements the feature extraction and the comparison with the satellite image [62].

Semantic maps offer a lot of information about the environment, in addition to improving navigation efficiency and positioning. To obtain 3D semantic maps, LiDAR point clouds and images can be merged to feed a neural network such as PSPNet50 [225] or a Convolutional Neural Network with a 3D full connection conditional random fields (CFRs) [121]. Another method for semantic mapping is Recurrent Neural Networks because they take into account temporal variability to make predictions. One example is Recurrent-Octomap, a refinement method for semantic maps [183].

Deep learning processing of 3D point clouds had a breakthrough with the proposal of PointNet [166], solving the unordered permutation problem that arises from the consideration of 3D point clouds. PointNet computes global features that can be used for classification. Recent advances in this area are the proposal of a Generative Adversarial Networks (GAN) for impairing the point cloud [217], and the use of patterns of k-Nearest Neighbors (k-NN) for the application of omnidirectional Graph Convolutional Neural Networks [221]. These advancements have allowed the application of deep learning to approach the problem of 3D point cloud registration [224]. Recent works formulate the registration as regression problem solved with residual based deep architecture [216]. In the Deep Global Registration approach [38], a deep auto-encoder architecture is used for outlier removal in order to improve the efficiency of the ICP algorithm. Siamese networks are also used in order to improve global localization [215].

2.3.2.3 Reinforcement Learning

In a reinforcement learning problem, the system aims to learn the best possible behavior through action and reaction interactions based on Markov decision theory [98]. In each environment, there exist an input i, a state indicator s, and an action a for every interaction. Every time the agent choose an action, a reinforcement signal r is received to obtain the action set that output the best reinforcement.

Autonomous navigation is one of the most common applications of reinforcement learning. During the exploration step, the system needs to know the actions and the possible situations, i.e. the states, and based on a reward system, it will choose the best action for each state. It allows the implementation of robot path planning algorithms without coding the actions expressly [219].

On the other hand, to obtain the best pair state-action, the Q-Learning algorithm is widely implemented, for example in robot navigation, where the LiDAR data can be used in conjunction with RGB camera images [200].

Other more modern reinforcement learning architectures use neural networks to optimize the search for algorithm optimization parameters [34, 67].

2.4 Data Resources

In order to check the validity of new modeling methods, it is necessary to have data with the corresponding ground truth. One option is to design field experiments and perform the data collection, but this process requires financial resources to acquire the LiDAR, the platform and the necessary additional sensors. In addition, it is often time-consuming to collect enough data for the results to be of interest. Other research teams often opt to use public data repositories that come with the corresponding ground truth previously calculated by reliable methods, such as Iterative Closest Point (ICP), Normal Distributions Transform (NDT) or one of its variants for SLAM applications, or manually labeled information for segmentation and classification tasks. Another option is to use simulators where both the measurement equipment and the desired scenarios for the experiments can be modeled in a very realistic way. In the following, the public data repositories and simulation environments most commonly used in the literature are described in more detail.

2.4.1 Public Data Repositories

There are several well known data collections that have been extensively used by researchers to demonstrate their computational approaches. In Table 2.1, some features of the most popular public datasets are summarized. The Kitti Dataset [70] has been extensively used for validation of autonomous navigation and SLAM applications. In fact, we found over 4660 citations to this dataset in a search over Scopus[4].

2.4.2 Simulation Environments

Simulated environments are an important tool for testing new measurement systems or algorithms thanks to their low cost. The most widely used platforms are Gazebo[5] and Stage[6], both belonging to ROS, and they allow simulating mobile platforms, UAV's and commercial platforms such as The Pioneer 3D-X, TurtleBot 2 and PR2 robot not only in an aesthetic way with Unified Robot Description Format (URDF), but also adding physical properties with the Open Dynamics Engine (ODE) or noise, which makes them very realistic. A very important utility is the possibility to develop robotic algorithms since it is possible to design the models to behave like the real robot, being a very interesting method to have an intermediate step between design and implementation, thus reducing the damage that could be caused to the users or to the platforms themselves.

[4]Last checked on 25th April 2023.
[5]http://gazebosim.org/
[6]http://wiki.ros.org/stage_ros

Table 2.1: Public datasets for navigation and object detection.

Dataset	Environment	LiDAR	Platform	Applications	Citations
Ford Campus vision and LiDAR, 2011 [153]	Urban street scenes	Velodyne HDL-64E, Riegl LMS-Q120	Ford F-250 pickup truck	Loop closures for computer vision and SLAM	231
Kitti, 2013 [70]	Real-world traffic scenarios in rural and inner city spaces	Velodyne HDL-64E	Vehicle (Volkswagen Passat B6)	Visual odometry, SLAM, 3D object detection and tracking	4660
Malaga urban, 2013 [17]	Urban scenarios, real-life traffic	3 Hokuyo UTM-30LX, 2 SICK LMS-200	Vehicle (Citroen C4)	SLAM , visual odometry and object detection	178
MIT Stata center, 2013 [56]	Indoor with moving people, furniture relocation and lighting changes	Tilting Hokuyo UTM-30LX	PR2	SLAM and autonomous driving	41
Oxford RobotCar, 2016[133]	Urban scenes in all weather conditions	2 SICK LMS-151 2D, SICK LD-MRS 3D	Autonomous Nisan LEAF	Long-term autonomous driving	739
Google corporation, 2016[82]	Indoor museum scenes	LiDAR	Backpack	SLAM	1121
NCLT, 2016 [26]	Indoor and outdoors spaces in all weather, seasons or lighting conditions	Velodyne HDL-32E	Segway robot	SLAM, navigation, place recognition, object detection and tracking	200
TorontoCity, 2017 [202]	Aerial and terrestrial data from urban scenarios	2 LiDARs	Airborne and vehicle	Segmentation	71
Apolloscape, 2018 [88]	Urban scenarios	Riegl VMX-1HA, 2 VUX-1HA Laser scanners	Vehicle	Self-Localization and semantic scene parsing	202
Urban@CRAS, 2018 [68]	Urban scenarios	Velodyne VLP-16	Vehicle	SLAM	16
Complex Urban, 2019 [92]	Urban scenarios	Velodyne VLP-16, SICK LMS-511	Vehicle	SLAM	98
Newer College, 2020 [170]	Oxford at walking speed	Ouster OS-1 64, (ground truth Leica BLK), LiDAR IMU ICM-20948	Handheld device	SLAM, 3D reconstruction and visual odometry	50
Urban Nav, 2020 [206]	Deep urban canyon of Hong Kong	Velodyne 32	Vehicle	SLAM	6
DUT-AS, 2021 [24]	Exploring the campus across seasons	SICK LMS-511	Vehicle	SLAM	8

Chapter 3

In-house LiDAR datasets

In this short Chapter we describe the LiDAR data generated in-house for the realization of computational experiments that are the main contribution of this Thesis regarding the realization of SLAM processes on the basis of LiDAR data. The organization of the Chapter is as follows: Section 3.1 summarizes the motivation of this contribution and the decision to implement SLAM algorithms using only traditional methods, without resorting to Deep Learning approaches. Section 3.2 gives a short relation of the M8 Quanergy LiDAR characteristics including a table with the more relevant specifications. Section 3.3 provides the information about the location and the experimental settings where the data capture was carried out. Finally, Section 3.4 illustrates the registration failure when applying Deep Learning algorithms to the data captured with M8 Quanergy LiDAR.

3.1 Motivation

In 2019, the Computational Intelligent Group (CIG) acquired a M8 Quanergy LiDAR, which is a low-cost sensor comparing market prices. A series of experiments were proposed to validate the quality of the results by implementing SLAM applications. There is a wide variety of both traditional and artificial intelligence-based SLAM methods based on LiDAR data. Nevertheless, not all of them are appropriate to make calculations with point clouds when their density is very low, providing results that are not acceptable. For instance, Deep Global Registration [38] was used to implement SLAM on the in-house M8 Quanergy LiDAR data described below achieving very poor results. We concluded that Deep Learning approaches require sensors with more quantity of beams to obtain point clouds that provide more information about the environment.

New affordable LiDAR sensors, such as the M8 from Quanergy that we are testing in this Thesis, allow for further popularization of LiDAR based SLAM applications. Due to its specific innovative characteristics, the M8 sensor still needs extensive testing by the community in order to assume its integration in the newly developed systems [144]. The work reported in here is intended partly to provide such empirical confirmation of the M8 sensor quality continuing experimentation over this sensor data reported elsewhere [2]. We have not carried out any precise calibration process of the sensor [116, 117]. Instead, we are assessing the sensor through the comparison of three standard point cloud registration methods over experimental data gathered in-house.

3.2 LiDAR M8 Quanergy

The Quanergy M8 LiDAR sensor is a multi-laser system with 8 2D-line scanners located on a spinning head. The Figure 3.2.1 shows the M8 Quanergy LiDAR physical aspect and some of its specifications. This system is based on TOF technology whose spin rate is between 5 Hz and 20 Hz and its maximum range is 100 m. The Table 3.1 shows the M8 LiDAR main parameters. Besides, M8

Table 3.1: Quanergy M8 sensor specifications.

Parameter	M8 sensor specifications
Detection layers	8
Returns	3
Minimum range	0.5m (80% reflectivity)
Maximum range	>100m (80% reflectivity)
Spin rate	5Hz - 20Hz
Intensity	8 bits
Field of view	Horizontal 360° - Vertical 20° (+3°/-17°)
Data outputs	Angle, Distance, Intensity, Synchronized Time Stamps

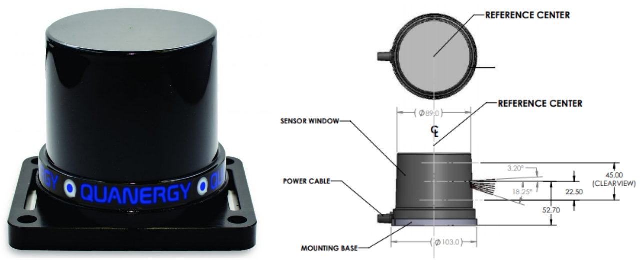


Figure 3.2.1: The M8 Quanergy LiDAR and diagrammatic specs.

LiDAR comes with 2 desktop applications to manage and visualize point clouds, a SDK (Software Development Kit) to record and show data in real time, and another SDK for implementation in the ROS framework.

3.3 Location and Experimental Settings

The experiments were carried out in the third floor of the Computer Science School of the UPV/EHU in San Sebastian. The M8 LiDAR was set on a manually-driven mobile platform in order to record the point clouds. The actual paths followed have small perturbations around the nominal path. We do not have a precise actual path measurement allowing to quantify the error in the trajectory. Figure 3.3.1 and figure 3.3.2 show the nominal path followed to record dataset #1 and dataset #2, respectively.

Both the time sequence of M8 captured point clouds and the MATLAB

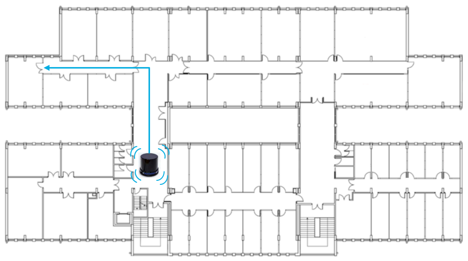


Figure 3.3.1: Nominal path followed during the LiDAR recording for dataset #1.

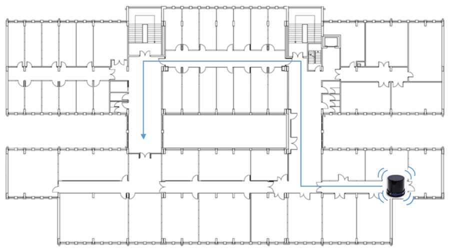


Figure 3.3.2: Nominal path followed during the LiDAR recording for dataset #2.

scripts used to carry out the computational experiments are published as open data and open source code in the Zenodo repository for reproducibility in two different links[1][2].

3.4 The Failure of Deep Learning

The motivation of this Thesis work regarding SLAM algorithms lies in the failure to achieve adequate processing of the collected data with novel state-of-the-art approaches, such as Deep Global Registration [38]. We tried to apply the transfer learning approach to adapt the published system to our data, so we re-trained the system with our data to no avail. The kind of results achieved are like the one shown in Figure 3.4.1. The deep registration system loses track when it reaches the first rotation of the sensor. Our interpretation is that the resolution of the dataset is too low for the Deep Learning approach to be adequately re-trained. However, some classical approaches can deal with the low resolution data appropriately. Consequently, we decided to focus on traditional methods to validate the quality of the results from M8 Quanergy LiDAR measurements.

[1]https://doi.org/10.5281/zenodo.4302360
[2]http://doi.org/10.5281/zenodo.4302366

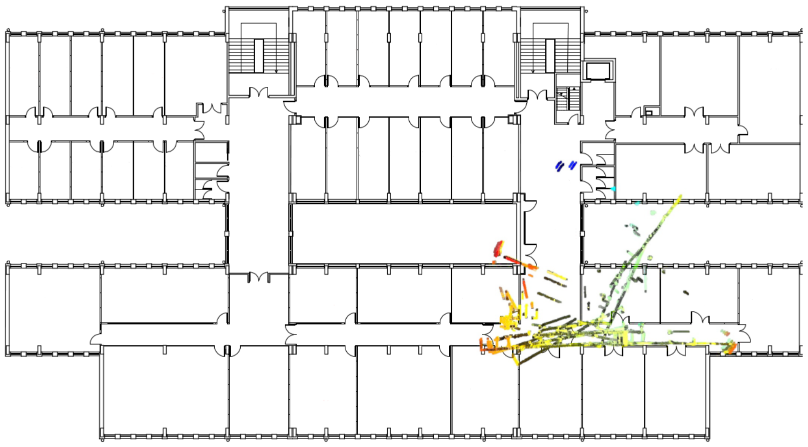


Figure 3.4.1: An instance of the results achieved over in-house dataset #2 after transfer learning applied to the Deep Global Registration published model.

Chapter 4

SLAM Algorithms for LiDAR Data

In this Chapter we provide a formal description of three classical 3D registration methods used in the SLAM computational experiments, and the proposed hybrid algorithm. The structure of the Chapter is as follows: Section 4.1 provides some motivation and introduction to the Chapter. Section 4.2 gives the generic definition of the point cloud registration algorithms. Section 4.3 provides a generic template of SLAM algorithms. Section 4.4 provides the formal definition of the Iterative Closest Point algorithm. Section 4.5 provides the formal definition of the Coherent Point Drift algorithm. Section 4.6 provides the formal definition of the Normal Distribution Transform algorithm. Section 4.7 describes our proposed hybrid approach. Finally, Section 4.8 gives some conclusions and the discussion.

4.1 Introduction and Motivation

Simultaneous Localization and Mapping (SLAM) [54, 11, 22] aims to estimate a reconstruction of the environment along with the path traversed by the sensor, becoming an integral part of ROS [210, 211]. One of the most widely used kinds of sensors in SLAM are laser based depth measurement sensors, or LiDAR sensors, which have been used for scanning and reconstruction of indoor and outdoor environments [23], even in underground mining vehicles [209]. Fusion of LiDAR with GPS allows for large scale navigation of autonomous systems [49].

Often, SLAM processes carry out a sequence of registrations of 3D point clouds, each trying to estimate the rigid transformation of the perceived data that results from the sensor motion in the world. LiDAR sensors are becoming the most used for SLAM in autonomous navigation of robots of quite diverse morphology [169], and autonomous driving [59]. In ground mobile robotics and autonomous driving, the kind of transformations is limited to translations and rotations around the axis perpendicular to the ground plane, though some small rotations in the other axes can happen in autonomous driving applications due to conditions of the road or sudden accelerations. Conventional LiDAR sensors for autonomous driving applications are very expensive, providing dense reading (i.e. 32 lasers in the Velodyne sensor each with thousands of readings) and very large collections of 3D point clouds. New sensors are appearing (i.e. the Quanergy sensor used for the experiments in this Thesis) which are more affordable but provide less resolution data captures, going down several orders of magnitude in the number of 3D point readings that require novel processing methods [213].

Our aim in this Thesis is to propose a hybridization of two well known point cloud registration methods. Hybridization is akin to the composition of subsystems in circuit like systems. It can be done in series, parallel or interleaving the systems in time. In this Thesis, we propose a serial hybridization where one algorithm serves to provide a robust initial condition to the other. This Thesis intends to show that it is possible to overcome the performance degradation of registration methods often used with high density datasets when applied to low density datasets.

The selected point cloud registration methods for the computational experiments are Iterative Closest Point (ICP) [15], Coherent Point Drift (CPD), Normal Distribution Transform (NDT) [16], and a novel Hybrid Registration Algorithm (HRA) that combines ICP and NDT into a more robust algorithm.

4.2 Point Cloud Registration Method

Point cloud registration is the algorithm used to align two or more 3D point cloud related to the same scene but referring to different coordinate systems. It is composed of two steps:

(a) To find the correspondence between the points in one point cloud (the moving) to the points in the other point cloud (the reference).

(b) To estimate the motion parameters that achieve optimal match of the moving points to the reference points after applying the correction.

If the motion is modeled by a rigid body or an affine transformation, then a matrix transformation common to all points is estimated. If the motion is some non linear deformation, then we have to estimate a flow field. For this Thesis we are restricted to rigid body transformations, which are compositions of a translation and a rotation. The transformation estimation process takes the form of a minimization problem where the energy function is related to the quality of the correspondence achieved.

Algorithm 4.1 General template of point cloud registration algorithm.

```
Input: sequence of point clouds $\{N(t)\}_{t=0}^{T}$ captured by the LiDAR
Output: overall point cloud $M(T)$, sequence of registered transformations $\{\mathcal{T}_t\}_{t=1}^{T}$
For $t = 0, \ldots, T$
    1. $N^{(1)}(t) \leftarrow$ remove ground plane from $N(t)$
    2. $N^{(2)}(t) \leftarrow$ remove ego-vehicle from $N^{(1)}(t)$
    3. $N^{(3)}(t) \leftarrow$ down-sample $N^{(2)}(t)$
    4. If $t = 0$ then $M(0) = N^{(3)}(t)$; goto For
    5. $(\mathcal{T}_t, e_t) \leftarrow$ register $\mathcal{T}_{t-1}\left(N^{(3)}(t)\right)$ to $M(t-1)$
    6. $N^{(4)}(t) \leftarrow \mathcal{T}_t\left(N^{(2)}(t)\right)$
    7. $M(t) \leftarrow merge\left(M(t-1), N^{(4)}(t)\right)$
```

4.3 Generic SLAM Framework

Given a sequence of point cloud captures, the process of SLAM based on point cloud registration methods is composed of the following steps:

1. Initialize the process with the first point cloud captured as the reference point cloud.
2. For each of the ensuing captured point cloud, called moving point cloud:
    (a) Find corresponding points of the following moving point cloud in the reference point cloud.
    (b) Estimate the optimal transformation matching the moving points to the reference points. The motion is modeled by a rigid body, so that a matrix transformation common to all moving point cloud is estimated.
    (c) Apply the transformation to the moving point cloud in order to put them in the coordinate system of the reference point cloud.

(d) Merge the corrected point cloud and the reference point cloud to obtain a new set of reference points.

3. Compose the transformations to obtain the trajectory of the sensor.

Thus, the accuracy of the SLAM computations depends critically of the reliability and efficiency of the method used to obtain the transformation for the moving point cloud.

Algorithm 4.1 specifies a general template of the registration process. The input to the algorithm is the sequence of point clouds recorded by the LiDAR $N(t)$; $t = \{0, \ldots, T\}$ while the sensor is being displaced through the scene. The final result of the process is a global point cloud $M(T)$ that contains all the recorded 3D points registered relative to the first acquired point cloud $N(0)$ coordinate system, through the estimation of the transformation matrix for each time instant $\{\mathcal{T}_t\}_{t=1}^{T}$.

The first three steps of the algorithm are related to pre-processing, i.e. the ground plane and vehicle removal, and sub-sampling the data to decrease the computation time, improve accuracy registration and delete the outliers. For each point cloud $N(t)$ acquired at time t, firstly the ground plane is removed by applying a segmentation, denoted $N^{(1)}(t)$. Secondly, the ego-vehicle points are deleted, denoted $N^{(2)}(t)$. Thirdly, we down-sample the point cloud to improve the processing efficiency, denoted $N^{(3)}(t)$. For the initial point cloud, $N^{(3)}(0)$ becomes the global merged reference cloud $M(0)$. For subsequent time instants $t > 0$, the fourth step is to estimate the transformation $\mathcal{T}_t$ of the acquired data $N^{(3)}(t)$ previously pre-processed related to the previous global point cloud $M(t-1)$. The transformation estimation process takes the form of a minimization problem where the energy function is related to the quality of the correspondence achieved. The estimated transformation is applied to the point cloud previous to the down-sampling step

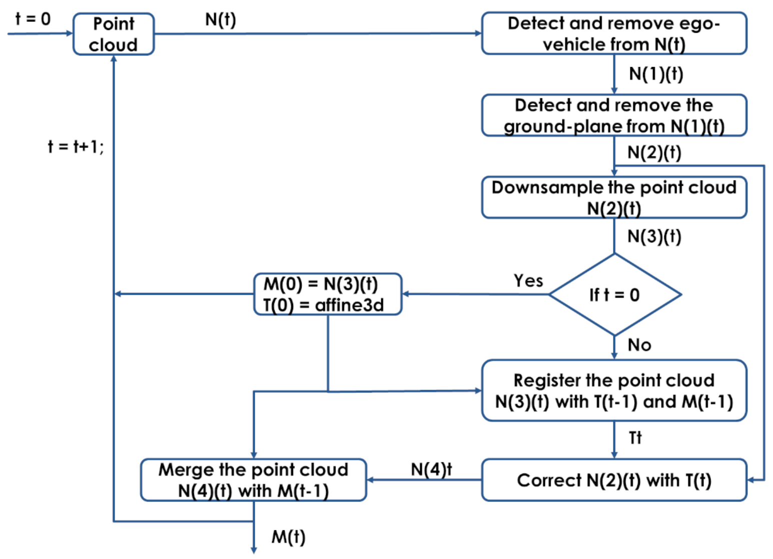

t = 0
Point cloud
N(t)
Detect and remove ego-vehicle from N(t)
N(1)(t)
Detect and remove the ground-plane from N(1)(t)
t = t+1;
N(2)(t)
Downsample the point cloud N(2)(t)
N(3)(t)
Yes
If t = 0
M(0) = N(3)(t)
T(0) = affine3d
No
Register the point cloud N(3)(t) with T(t-1) and M(t-1)
Tt
Merge the point cloud N(4)(t) with M(t-1)
N(4)t
Correct N(2)(t) with T(t)
M(t)

Figure 4.3.1: Flow diagram of the registration algorithm. N(i)(t) is the point cloud at time t after the i-th step of processing. M(t) is the overall point cloud up after merging all the registered point clouds processed up to time t.

$N^{(4)}(t) = \mathcal{T}_t\left(N^{(2)}(t)\right)$, which is used to obtain the new global registered point cloud by merging $M(t) \leftarrow merge\left(M(t-1), N^{(4)}(t)\right)$.

Figure 4.3.1 presents a flow diagram of the general algorithm applied to obtain the registration of the LiDAR point clouds for each time point $t = \{0, \ldots, T\}$.

4.4 Iterative Closest Point Method

The most popular and earliest point cloud registration method is the Iterative Closest Point (ICP) proposed by Besl in 1992 [15]. This technique has been exploited in many domains, giving rise to a host of variations whose relative merits are not so easy to assess [162].

Given a point cloud[1] $P = \{\mathbf{p}_i\}_{i=1}^{Np}$ and a shape described by another point

[1](The original paper includes the possibility to specify other primitives such as lines or triangles with well defined distances to a point, but we will not consider them in this Thesis.)

cloud $X = \{\mathbf{x}_i\}_{i=1}^{Nx}$ the least squares registration of P is given by $(\mathbf{q}, d) = \mathcal{Q}(P, Y)$, where $Y = \{\mathbf{y}_i\}_{i=1}^{Np}$ is the set of nearest points from X to the points in P, i.e.

$$\mathbf{p}_i \in P; \mathbf{y}_i = \arg\min_{\mathbf{x} \in X} \|\mathbf{x} - \mathbf{p}_i\|^2 ,$$

denoted $Y = \mathcal{C}(P, X)$, and operator $\mathcal{Q}$ is the least squares estimation of the rotation and translation mapping P to Y using quaternion notation, thus $\mathbf{q} = [\mathbf{q}_R \mid \mathbf{q}_T]^t$ is the optimal transformation specified by a rotation quaternion $\mathbf{q}_R$ and a translation $\mathbf{q}_T$, and d is the registration error. The energy function minimized to obtain the optimal registration is

$$f(\mathbf{q}) = \frac{1}{N_p} \sum_{i=1}^{Np} \|\mathbf{y}_i - \mathbf{R}(\mathbf{q}_R)\,\mathbf{p}_i - \mathbf{q}_T\|^2 ,$$

where $\mathbf{R}(\mathbf{q}_R)$ is the rotation matrix constructed from quaternion $\mathbf{q}_R$. The iteration is initialized by setting $P_0 = P$, $\mathbf{q}_0 = [1,0,0,0,0,0,0]^t$, and $k = 0$.

The algorithm iteration is as follows:

(1) compute the closest points $Y_k = \mathcal{C}(P_k, X)$,

(2) compute the registration $(\mathbf{q}_k, d_k) = \mathcal{Q}(P_0, Y_k)$,

(3) apply the registration $P_{k+1} = \mathbf{q}_k(P_0)$, and

(4) terminate the iteration if the results are within a tolerance: $d_k - d_{k+1} < \tau$.

A drawback of this registration method is that it can fall in a local minimum, that is the iteration is finished because of the tolerance condition but the results are not as expected. Figure 4.4.1 shows the structure of the algorithm as flow diagram of its basic iterations.

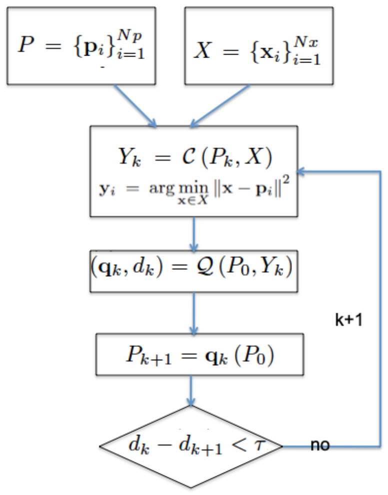

$P = \{\mathbf{p}_i\}_{i=1}^{Np}$
$X = \{\mathbf{x}_i\}_{i=1}^{Nx}$
$Y_k = \mathcal{C}(P_k, X)$
$\mathbf{y}_i = \arg\min_{\mathbf{x}\in X} \|\mathbf{x} - \mathbf{p}_i\|^2$
$(\mathbf{q}_k, d_k) = \mathcal{Q}(P_0, Y_k)$
$P_{k+1} = \mathbf{q}_k(P_0)$
$d_k - d_{k+1} < \tau$
no
k+1

Figure 4.4.1: Flow diagram of the ICP point cloud registration algorithm.

4.5 Coherent Point Drift Method

The Coherent Point Drift (CPD) [145, 131] registration method considers the alignment of two point sets as a probability density estimation problem. The first point set $X = \{\mathbf{x}_i\}_{i=1}^{N}$ is considered the data samples generated from the Gaussian mixture model (GMM) whose centroids are given by the second point set $Y = \{\mathbf{y}_i\}_{i=1}^{M}$. Therefore, the CPD registration tries to maximize the likelihood X as a sample of the probability distribution modeled by Y after the application of the transformation $T(Y, \theta)$, where θ are the transformation parameters.

The GMM model is formulated as

$$p(\mathbf{x}) = \omega \frac{1}{N} + (1 - \omega) \sum_{m=1}^{M} \frac{1}{M} p(\mathbf{x} | m)$$

assuming a uniform distribution for the *a priori* probabilities $P(m) = \frac{1}{M}$, and adding an additional uniform distribution $p(\mathbf{x} | M + 1) = \frac{1}{N}$ to account for noise and outliers. All Gaussian conditional distributions are isotropic with the same variance σ^2, i.e.

$$p\left(\mathbf{x}\left|m\right.\right) = \left(2\pi\sigma^2\right)^{-D/2} \exp\left(\frac{\left\|\mathbf{x}-\mathbf{y}_m\right\|^2}{2\sigma^2}\right).$$

The point correspondence problem is equivalent to selecting the centroid $\mathbf{y}_m$ with maximum *a posteriori* probability $P\left(m\left|\mathbf{x}_n\right.\right)$ for a given sample point $\mathbf{x}_n$. The CPD tries to minimize the negative log-likelihood

$$E\left(\theta,\sigma^2\right) = -\sum_{n=1}^{N}\log\sum_{m=1}^{M}P\left(m\right)p\left(\mathbf{x}\left|m\right.\right)$$

by an Expectation-Maximization (EM) algorithm. The E step corresponds to solving the point correspondence problem using the old parameters, by computing the *a posteriori* probabilities with the old parameters $P^{old}\left(m\left|\mathbf{x}_n\right.\right)$. Let it be

$$p_{n,m}^{old} = \exp\left(-\frac{1}{2}\left\|\frac{\mathbf{x}_n - T\left(\mathbf{y}_n,\theta^{old}\right)}{\sigma^{old}}\right\|\right),$$

then

$$P^{old}\left(m\left|\mathbf{x}_n\right.\right) = p_{n,m}^{old}\left(\sum_{k=1}^{M}p_{k,m}^{old}+c\right).^{-1}$$

The M step is the estimation of the new parameters minimizing the complete negative log-likelihood

$$Q = -\sum_{n=1}^{N}\sum_{m=1}^{M}P^{old}\left(m\left|\mathbf{x}_n\right.\right)\log\left(P^{new}\left(m\right)p^{new}\left(\mathbf{x}\left|m\right.\right)\right).$$

For rigid transformations, the objective function takes the shape:

$$Q\left(\mathbf{R},\mathbf{t},s,\sigma^2\right) = \frac{1}{2\sigma^2}\sum_{n,m=1}^{N,M}P^{old}\left(m\left|\mathbf{x}_n\right.\right)\left\|\mathbf{x}_n - s\mathbf{R}\mathbf{y}_m - \mathbf{t}\right\|^2 + \frac{N_pD}{2}\log\sigma^2,$$

such that

$$\mathbf{R}^T\mathbf{R} = \mathbf{I}, \det\left(\mathbf{R}\right) = 1.$$

Closed forms for the transformation parameters are given in [145].

4.6 Normal Distributions Transform Method

The key of the NDT [16] method is the data is representation. The space around the sensor is discretized into regular overlapped cells. The content of each cell having more than 3 points is modeled by a Gaussian probability distribution of mean:

$$\mathbf{q} = \frac{\mathbf{1}}{\mathbf{n}}\sum_i \mathbf{x}_i,$$

and covariance matrix

$$\Sigma = \frac{1}{n}\sum_i \left(\mathbf{x}_i - \mathbf{q}\right)\left(\mathbf{x}_i - \mathbf{q}\right)^t,$$

so that the probability of a LiDAR sample falling in the cell is of the form:

$$p\left(\mathbf{x}\right) \sim \exp\left(\frac{-\left(\mathbf{x}-\mathbf{q}\right)^t \Sigma^{-1}\left(\mathbf{x}-\mathbf{q}\right)}{2}\right).$$

Given an initial rigid body transformation $T\left(\mathbf{x};\mathbf{p}_0\right)$, where $\mathbf{p}$ is the vector of translation and rotation parameters, a reference point cloud $\{\mathbf{x}_i\}$ modeled by the mixture of the cells Gaussian distributions, and the moving point cloud $\{\mathbf{y}_i\}$, the iterative registration process is as follows: the new laser sample points $\mathbf{y}_i$ are transformed into the reference frame of the first cloud

$$\mathbf{y}'_i = T\left(\mathbf{y}_i;\mathbf{p}_{t-1}\right),$$

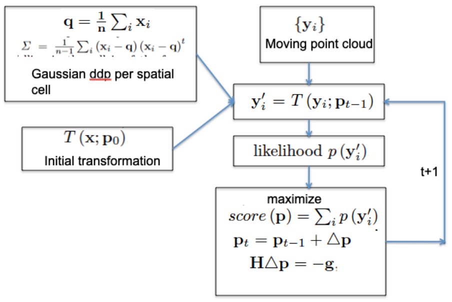

$\mathbf{q} = \frac{1}{n}\sum_i \mathbf{x}_i$
$\Sigma = \frac{1}{n-1}\sum_i (\mathbf{x}_i - \mathbf{q})(\mathbf{x}_i - \mathbf{q})^t$
Gaussian ddp per spatial cell
$\{\mathbf{y}_i\}$
Moving point cloud
$\mathbf{y}'_i = T(\mathbf{y}_i; \mathbf{p}_{t-1})$
$T(\mathbf{x}; \mathbf{p}_0)$
Initial transformation
likelihood $p(\mathbf{y}'_i)$
t+1
maximize
$score(\mathbf{p}) = \sum_i p(\mathbf{y}'_i)$
$\mathbf{p}_t = \mathbf{p}_{t-1} + \triangle\mathbf{p}$
$\mathbf{H}\triangle\mathbf{p} = -\mathbf{g}$

Figure 4.6.1: Flow diagram of the NDT point cloud registration algorithm.

where we find the cell where it falls and use its parameters $(\mathbf{q}, \Sigma)$ to estimate its likelihood $p(\mathbf{y}'_i)$. The score of the transformation is given by

$$score(\mathbf{p}) = \sum_i p(\mathbf{y}'_i).$$

The maximization of the score is carried out by gradient ascent using Newton's method, i.e. $\mathbf{p}_t = \mathbf{p}_{t-1} + \triangle\mathbf{p}$. The parameter update is computed solving the equation $\mathbf{H}\triangle\mathbf{p} = -\mathbf{g}$, where $\mathbf{H}$ and $\mathbf{g}$ are the Hessian and the gradient of the $-score(\mathbf{p}_{t-1})$ function, respectively. Closed forms of $\mathbf{H}$ and $\mathbf{g}$ are derived in [16] for the 2D case. An extension to 3D is described in [135]. Figure 4.6.1 shows the flow diagram of the NDT algorithm.

4.7 Hybrid Point Cloud Registration Algorithm

In this section, Hybrid Registration Algorithm (HRA) is presented, which combines ICP and NDT methods because the former method generates a better surface, i.e. with greater point density than NDT, but it is not able to properly register point sets at turning sections as the latter method. Our hybrid algorithm uses initially the ICP method, changing to the NDT method when the registration error becomes higher than a threshold.

Algorithm 4.2 The Hybrid Registration Algorithm (HRA) combining both ICP and NDT registration methods.

```
**Input**: sequence of point clouds $\{N(t)\}_{t=0}^{T}$ captured by the LiDAR
**Output**: overall point cloud $M(T)$, sequence of registered transformations $\{\mathcal{T}_t\}_{t=1}^{T}$
Method = “ICP”
For $t = 0, \ldots, T$
    1. $N^{(1)}(t) \leftarrow$remove ground plane from $N(t)$
    2. $N^{(2)}(t) \leftarrow$ remove ego-vehicle from $N^{(1)}(t)$
    3. $N^{(3)}(t) \leftarrow$ downsample $N^{(2)}(t)$
    4. If $t = 0$ then $M(0) = N^{(3)}(t)$; GOTO step 1
    5. $(\mathcal{T}_t, e_t) \leftarrow$ register $\mathcal{T}_{t-1}\left(N^{(3)}(t)\right)$ to $M(t-1)$ using Method
    6. If $e_t > \theta_e$ then Method = “NDT”
    7. $N^{(4)}(t) \leftarrow \mathcal{T}_t\left(N^{(2)}(t)\right)$
    8. $M(t) \leftarrow merge\left(M(t-1), N^{(4)}(t)\right)$
```

Algorithm 4.2 presents a description of the proposed hybrid registration method. The input of the algorithm is the sequence of point clouds recorded by the LiDAR $N(t)$; $t = \{0, \ldots, T\}$. The point sets are obtained while the LiDAR sensor is being displaced manually in the environment according to the approximate path in Figure 3.3.2. The final result of the process is a global point cloud $M(T)$ that contains all the recorded 3D points registered relative to the first acquired point cloud $N(0)$, and the estimation of the LiDAR recording positions relative to the initial position given by the composition of the point cloud registration transformations estimated up to this time instant $\{\mathcal{T}_t\}_{t=1}^{T}$.

The process of each point cloud is as follows:

- For each point cloud $N(t)$ acquired at time t, firstly we remove the ground plane applying a segmentation, denoted $N^{(1)}(t)$.
- Secondly we remove the ego-vehicle points, denoted $N^{(2)}(t)$.

- Thirdly, we down-sample the point cloud to decrease the computation time and improve accuracy registration, denoted $N^{(3)}(t)$.
- For the initial point cloud, $N^{(3)}(0)$ becomes the global merged reference cloud $M(0)$.
- For subsequent time instants $t > 0$, the fourth step is to estimate the transformation $\mathcal{T}_t$ of the acquired data $N^{(3)}(t)$ optimally registered to the previous global point cloud $M(t-1)$. For this estimation, we may use ICP or NDT methods.
- We then apply this transformation to the acquired point cloud previous to downsampling $N^{(4)}(t) = \mathcal{T}_t\left(N^{(2)}(t)\right)$, which is used to obtain the new global registered point cloud by merging $M(t) \leftarrow merge\left(M(t-1), N^{(4)}(t)\right)$.

Our hybrid strategy consists in using the ICP method in the initial steps of the algorithm, up a time instant when the registration error meets a given threshold, after this time point the system shifts to use the NDT method to continue registration of all remaining point clouds. The rationale is that the ICP acts as a good initial estimation for the ensuing NDT steps, as will be demonstrated in the results section below.

4.8 Conclusion

In this Chapter we have given the formal description of the benchmark SLAM algorithms and a new hybrid algorithm proposal that combines the best features of the ICP and NDT algorithms. These algorithms are validated over the in-house datasets described previously. Next Chapter provides results and discussions.

Chapter 5

Results of SLAM experiment

This Chapter reports the results of the computational SLAM experiments carried out over the data collected as reported in Chapter 3. Section 5.1 reports the results achieved by the three classical SLAM approaches over the in-house dataset #1. Section 5.2 reports the results of the classical algorithms and our proposed hybrid model over the in-house dataset #2. Section 5.3 gives some conclusions on the experimental results.

5.1 Results over the In-house Dataset #1

Dataset #1 has been used as preliminary benchmark for SLAM algorithms. As discussed in Chapter 3, we found that Deep Learning approaches did not perform correctly over this kind of sparse LiDAR data. In this Section we report the results achieved with the classical algorithms over this dataset.

The measurement of the quality of the registration is the Root Mean Squared

Error (RMSE) between two points clouds after carrying out the registration process applying the estimated transformation. Figure 5.1.1 presents the evolution of the registration error achieved with the generic point cloud registration Algorithm 4.1 for the point clouds recorded along the path shown in Figure 3.3.1 using alternatively the three classical SLAM methods described in Chapter 4, namely ICP, CPD, and NDT. The plot scale is logarithmic in order to be able to represent the three error plots in the same scale. The NDT algorithm gives the minimal error all along the path. The error of both NDT and CPD registration methods remains bounded, however the error of the ICP method explodes after a point in the trajectory, specifically the turning point at the end of the main hallway in Figure 3.3.1.

Figure 5.1.2 (right) shows the overall registered cloud point obtained at the end of the SLAM process, and the estimated trajectory (visualized as white dots). After some point in the trajectory, the ICP registration loses track and gives almost random trajectory tracking results. Figure 5.1.2 (left) shows the results of the ICP registration up to the turning point, which are comparable with the results of the other algorithms.

Figure 5.1.3 (right) shows the results of the CPD algorithm in terms of the registered and merged global cloud of points and the trajectory estimation (white dots). It can also be appreciated in the figure 5.1.3 (left) that the SLAM process gets lost after the path turning point, however the registration of point clouds does not become unwieldy.

Finally, Figure 5.1.4 shows the results of the NDT algorithm. The trajectory (white dots) is quite accurate to the actual path followed by the sensor. The trajectory turning point was in fact as smooth as shown in the figure. The overall registered and merged point cloud has a nice fit of the actual hallway walls, as can be appreciated in Figure 5.1.5, including a communication switch

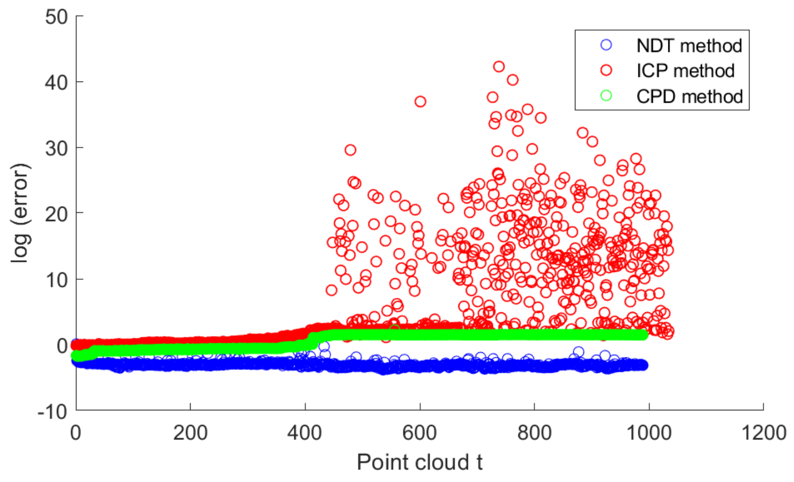


Figure 5.1.1: Evolution of the registration error (log plot) for NDT (blue dots), CPD (green dots), and ICP (red dots) over in-house dataset #1.

closet signaled in the figure with an arrow, which is not present in the original floor plan.

5.2 Results over the In-house Dataset #2

Figure 5.2.1 plots the time evolution of the registration error for ICP, NDT, and HRA setting the threshold to $\theta_e = 0.25$. Each point corresponds to the registration error of a new point cloud. The proposed HRA shows the best performance after the initial phase, where the results of all three methods are mixed up. The peaks in the plot at iterations 500, 1500 and 2500 correspond to the turning points in the path, where all methods suffer due to the high rotational motions of the sensor. Table 5.1 gives a summary of the numerical registration errors, namely the maximum instantaneous RMSE, its median along the path, and the cumulative error.

Figures 5.2.2, 5.2.3, and 5.2.4 show the trajectory estimation and the projection of the reconstructed surface on the ground plane for the ICP, NDT, and

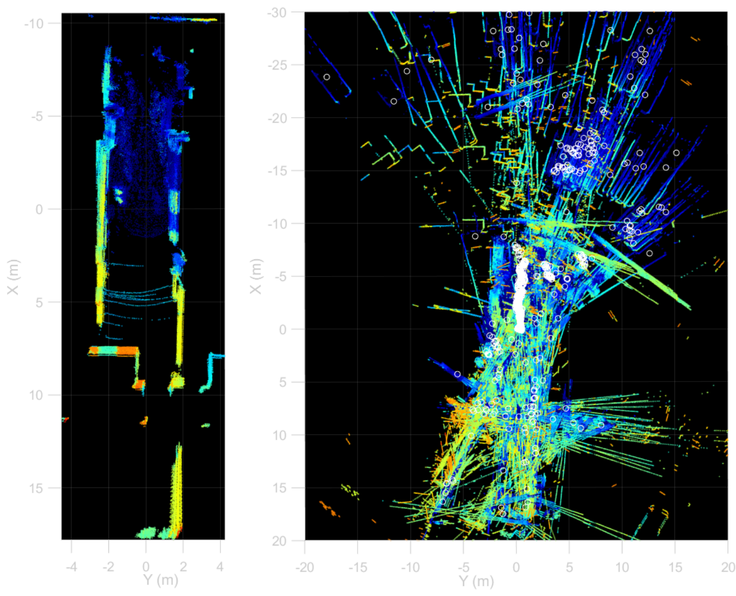


Figure 5.1.2: Registration of the cloud points before reaching the turning point (left) over in-house dataset #1. Estimated trajectory (white dots) and registered cloud of points using ICP (right).

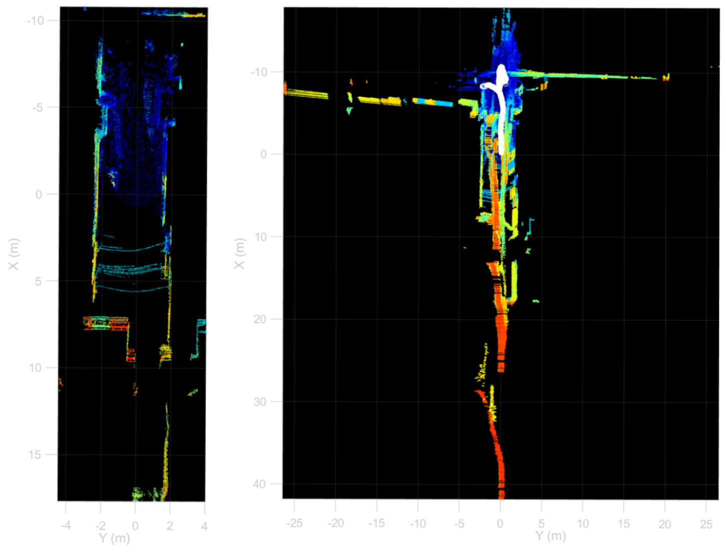


Figure 5.1.3: Registration of the cloud points before reaching the turning point (left) over in-house dataset #. Estimated trajectory (white dots) and registered cloud of points using CPD (right).

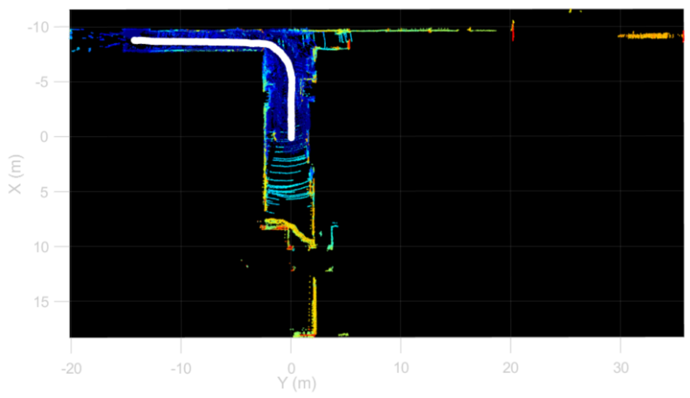


Figure 5.1.4: Estimated trajectory (white dots) and registered cloud of points using NDT over the in-house dataset #1.

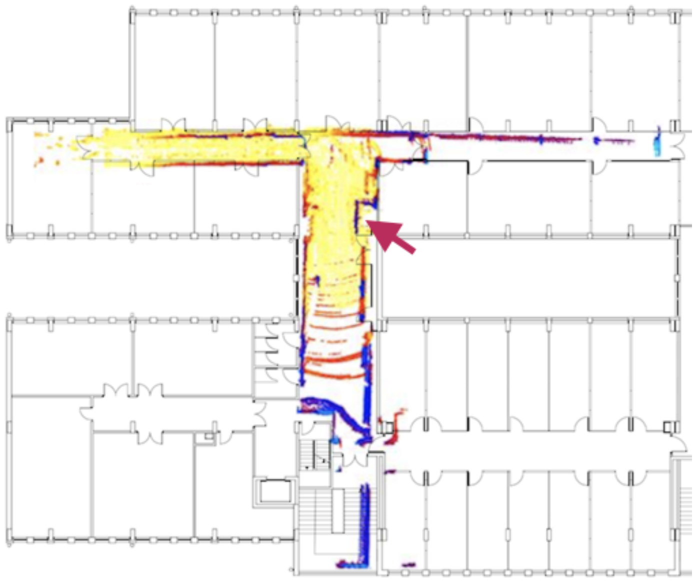


Figure 5.1.5: Projection of the NDT registered point cloud of in-house dataset #1 on the plan of stage 3 of the building.

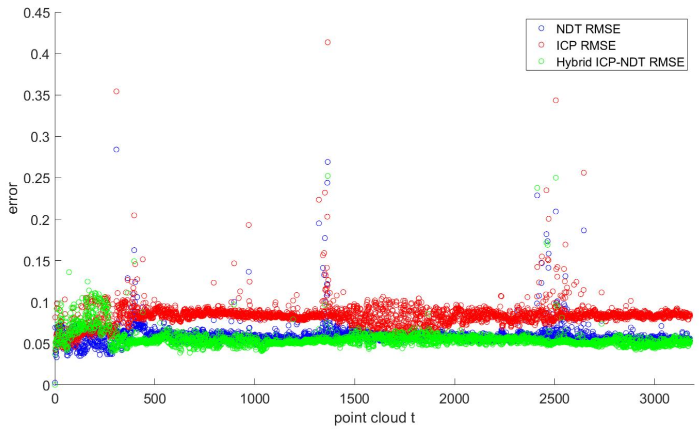


Figure 5.2.1: Time evolution of the registration RMSE for NDT (blue dots), ICP (red dots), and HRA methods (green dots).

Table 5.1: Performance of ICP, NDT, and HRA methods along the experimental path, given by maximum and median instantaneous error, and the cumulative error.

	ICP method	NDT method	HRA method
Maximum RMSE	0.4136	0.2841	0.2522
Median RMSE	0.0835	0.0589	0.0554
Cumulative RMSE	265.29	187.21	176.20

HRA registration methods, respectively. In Figure 5.2.2 the surfaces defined by the registered are aligned with the walls of the plane until the sensor reaches second bend. Afterwards, the trajectory diverges from the nominal path and the surfaces lose alignment. For the NDT method, though its error performance is better than that of the ICP method, in Figure 5.2.3 it can be appreciated that, after the first turning point, the reconstructed surfaces are misaligned with the actual walls of the plan, showing path divergence earlier than ICP. Finally, the HRA trajectory and reconstruction results provided in Figure 5.2.4 show a much better fit of the reconstructed surfaces to the walls of the plane along the whole trajectory, which matches better the nominal path.

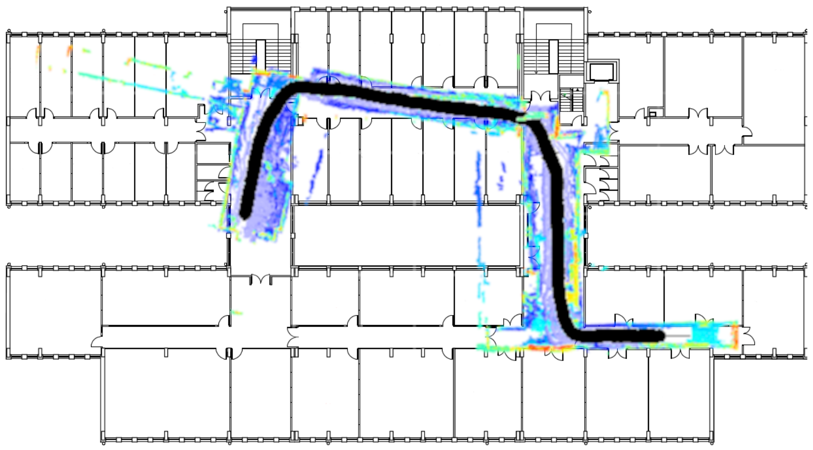


Figure 5.2.2: Projection of the ICP registered point cloud on the plan of stage 3 of the building with the estimated trajectory over in-house dataset #2.

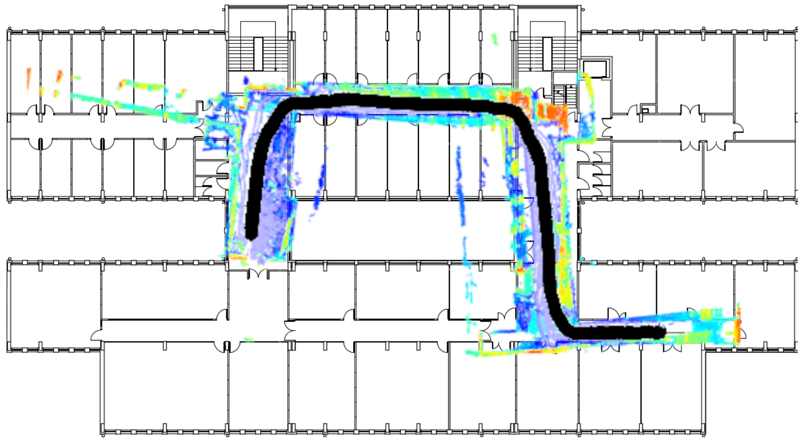


Figure 5.2.3: Projection of the NDT registered point cloud on the plan of stage 3 of the building with the estimated trajectory over in-house dataset #2.

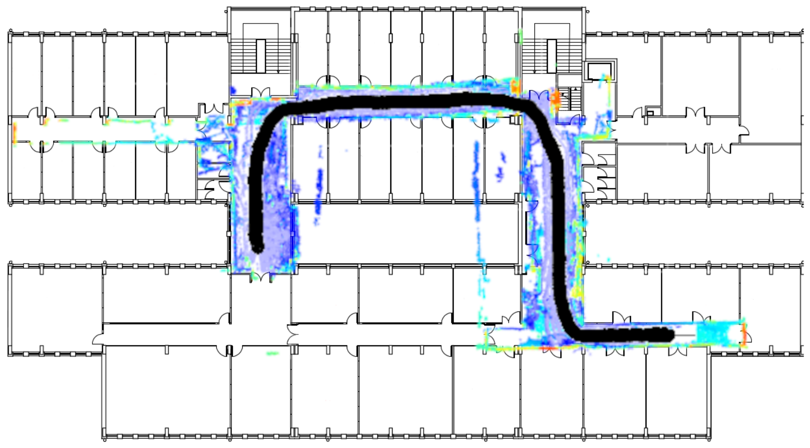


Figure 5.2.4: Projection of the HRA registered point cloud on the plan of stage 3 of the building with the estimated trajectory over in-house dataset #2.

5.3 Conclusion

In this Chapter we report a comparison between three registration methods for 3D point clouds, namely the Iterative Closest Point, Coherent Point Drift and Normal Distributions Transform. To collect point sets, we have located the M8 Quanergy LiDAR sensor on a manually driven mobile platform through the third floor of the Computer Science School of the UPV/EHU in San Sebastian. The registration algorithm followed includes preprocessing (detect and remove ego-vehicle and floor, and downsample), registration, transformation and merger point cloud. For each method described in Chapter 4, we have obtained the registration error, the estimation of the path traversed by the sensor, and the reconstructed point cloud. For the ICP and CPD methods, the error is larger than for the NDT method over the in-house dataset #1. Besides, after the turning point in the nominal path, ICP and CPD obtained path and resulting point cloud are incorrect. NDT registration obtains coherent experimental results and an accurate trajectory compared with the nominal path followed. Therefore, we apply the ICP and NDT for the in-house dataset #2 with suboptimal results, hence we propose a novel hybrid algorithm HRA that improves over both ICP and NDT over this dataset.

Chapter 6

State of the Art of Computational Ethology

This Chapter provides a short introduction to Computational Ethology in order to set the stage for the computational experiments referred in the second part of this Thesis. Section 6.1 provides an introduction to the Chapter. Section 6.2 comments on the diversity of sensors that are used in Computational Ethology. Section 6.3 reviews the most popular computational methods found in the literature. Finally, Section 6.4 gives some conclusions and future perspective.

6.1 Introduction

Ethology is defined as the discipline that studies the animal behavior in terms of its phenomenological, causal, ontogenetic and evolutionary aspects, in order to provide answers to the causes and development that animal behavior undergoes, as well as to understand how it is performed [8], bearing in mind that the behavior is understood as the set of muscular responses of a living being as a consequence of an external stimulus [73] and internal motivation.

In this way it is possible to extract behavioral characteristics and study their alterations due to diseases or disorders. In addition, it is now possible to generate animal models with genetic modifications that provide study subjects with anatomy, physiology or response to a pathogen that are sufficiently similar to humans to be able to extrapolate the results obtained to them. The most commonly used models in research are rodents, especially mice and rats, zebra fish, amphibians and reptiles, birds and other small animals.

Early Ethology studies were conducted visually, describing what researchers saw in each experiment qualitatively. Later on, they began to evaluate certain behaviors on the basis of some predefined criteria, thus beginning a quantitative approach. However, this new methodology had many drawbacks:

- Time-consuming: The time needed to pre-process an experiment can require up to three times its actual duration.

- Humdrum: Behavior observation and annotation is a very repetitive task that must be performed for hours during several weeks. At a certain point, the observer is tired and performs the task mechanically losing ability to notice new patterns.

- Difficult to transfer knowledge: When it is not possible to express it in plain words, different team members may make different annotations for the same experiment. This makes the process strongly subject to the judgment of the scientist and difficult to standardise and reproduce in other laboratories with different equipment.

- Limited to the visual acuity of the observer: If there are several animals, it is difficult to pay attention to the behavior of all of them at the same time.

- Low-dimensional. Through human observation it is not possible to annotate a large number of variables for each behavior.

To address all these issues, a new discipline called Computational Ethology emerged, which is defined as the study of animal behavior using Computer Vision and Artificial Intelligence (AI) techniques to help quantify and analyse behavioral patterns [44]. This is usually done by studying animals moving in a free-ranging environment and analysing specific behaviors or calculating ethograms, which describe how often each action is performed and the probability of the next action being performed, in order to try to find out how decision-making is carried out [8]. Thus, Computational Ethology allows the incorporation of advances in Computer Vision and AI in the study of animal behavior, providing the following advantages:

- Decrease the processing time of experiments, because of the algorithms implementation that automatically extract relevant information from experimental records.

- Eliminate the limitations of the observer, allowing the processing of several animals at the same time with increased accuracy.

- Increase the dimensionality of behavior measurements, extracting more characteristics from the same behavior it is possible to increase the information and, therefore improve its analysis.

- Standardise the characteristics of behaviors, since it is possible to describe behaviors quantitatively thanks to the increase in information, and the standardization of the capture instruments.

6.2 Sensors

This section will describe the most commonly used physical devices in Computational Ethology, i.e. sensors, which are of very different types such as RGB, infrared and depth video cameras, microphones, RFID (Radio Frequency Identification) antennas, pressure sensors, accelerometers, magnetometers and gyroscopes. Currently, RGB video cameras are the most used sensors to record experiments in open field arenas [207, 139], operant conditioning chambers [39], and in animal natural environment [58]. The systems can be composed of a single RGB camera [30] or a multi-camera system[182, 69, 12]. Low cost configurations with video cameras use a Raspberry Pi to carry out the recordings and store the data for further processing [126]. Recent systems featured depth cameras providing 3D measurements of the kinematics of the animals [140]. This type of camera can be used both to track animals and for behavioral detection and classification [71, 163]. Infrared cameras allow for monitoring the behavior of animals during the day and during the darkness of the night [29], and as a non-invasive system in group-housed animals [138]. Specific examples of their application are the study of monkeys hunting fish [186], and the study of Japanese eels to understand the environmental conditions that must be met for them to climb a low-height vertical weir [112]. In addition, infrared cameras have been used to obtain high-resolution images allowing precision tracking of certain parts of a rat's body [156]. Infrared sensors are also used to track an animal's movements and see when it is approaching a certain object [19].

Other sensors also widely used in Computational Ethology are inertial sensors and accelerometers, which can be placed on the head of animals to study sensorimotor responses in pigeons [6], mice [193] and fox squirrels [66].

Though they have reduced precision, RFID antennas are also used to track animals because of their robustness against visual occlusions and the possibility to make experiments in greater space. These antennas have been applied on rodents to track each animal while moving freely in the study of the "Individuality Paradigm” [102], and on broilers to identify, describe and quantify a wide range of behaviors in combination of video recording [52].

Pressure sensors are emerging for behavioral studies because they are non-invasive and allow animals to move without restriction. One example is pressure sensor based on piezoelectric materials, which can detect animal movements for further processing and analysis. These platform sensors also offer a high sensitivity for detecting pressure changes, making it possible to detect freezing episodes, breathing and heartbeats in mice [27]. Another example is the use of wearable pressure sensors to study pressure changes in the jaw movements of cattle [31].

In addition to movements, audio signals captured by microphones allow for the study of the vocalizations that certain animals emit to communicate with each other [146, 195]. This can be used to measure the response to stress [42], or analyse their communication during mating [83], or while performing a task [174]. Specific software applications exist to detect and classify ultrasonic vocalizations such as DeepSqueak [40] or BootSnap [1].

6.3 Methods

In Computational Ethology, techniques developed in the field of Computer Vision and AI are used to process the data collected during experimentation. To study the trajectory and locomotion of an animal, tracking algorithms segment the animal with respect to the background obtaining the center of mass and orientation, among others properties, automatically. To study other behaviors such as grooming, resting or rearing, it is necessary to label these behaviors firstly. This task can be performed using a supervised approach, where the behaviors to be studied are indicated in the algorithm that generates the AI model, and an unsupervised approach, where the data is fed into an algorithm to extract patterns, which is very interesting for highlighting behaviors that were not previously foreseen by the scientist or that escape the human eye. These algorithms study temporal dynamics in the time or frequency domain, where the former studies how data vary as a function of time and the latter how cyclic movements vary as a function of their frequency [44]. In the following we discuss current approaches and tools for the two fundamental tasks of tracking and behavior classification.

6.3.1 Tracking

DeepLabCut [141] is a markerless motion tracking system based on transfer learning that can be easily tailored to the specific experimental setting. Recently, it has been used for measuring monocular ability of mice to assess distances [155, 10], the behavioral risk assessment of mice [201], study of novelty induced behavioral dynamics in threat prediction [5], the management of fish passage in rivers [134], kinematics of time-varying lumbar flexion-extension [74], behavioral profiling of rodent stroke recovery [205], X-ray video analysis of rodent locomotion [107], cardiac physiology assessment in zebra fish [184], multi-animal pose estimation and tracking [114], pose estimation of speech articulators [208], gait analysis in stroke survivors [129]. DeepLabCut can be also used as the basis for the development of dedicated systems, such as the Anipose system for estimation of 3D pose [101]. In addition to estimating the 3D pose, it enables the camera calibration, the filtering of the trajectories and allows the visualisation of the tracked data. Some independent comparative evaluations found that markerless tracking systems are still requiring improvements in order to achieve the same tracking accuracy as current marker based systems [147], while others found good agreement between markerless system and marker based gold standard motion tracking in specific tasks [53, 197, 181].

Bonsai is a programming framework for neuroscience experimentation that allows data acquisition, experiment control, data processing, and pose tracking [130]. Bonsai also allows the integration of one or more sensors such as cameras and Local Field Potential (LFP) recording thanks to its Open Ephys acquisition system [187]. This open-source software has given rise to several products in the field of neuroscience, such as the GoFish platform, which uses Bonsai to experiment with fish [4], rodents [37] and flies [80]. It has been used to study the impact of feeding on the organism neural systems [164], how chronic stress affects the body [171], the frailty associated with Alzheimer's disease [113], and mice torpor [85].

Apart from DeepLabCut and Bonsai, with the advent of AI and computing capacity, a large number of tracking and classification algorithms have been developed for animal experimentation based on machine learning and neural networks. The best known and most widely used is the JAABA application that uses a semi-supervised machine learning algorithm for automatic annotation of animal behavior [97]. Apart from this, there are different tools to classify behaviors using machine learning and Computer Vision [71] or also to detect behaviors based on heuristic algorithms such as BehaviorDEPOT, a tool that uses statistics based on animal dynamics and posture [65]. There are also neural networks that identify the genetics of a mouse by analyzing grooming behavior [72]. And with more computational capacity, algorithms based on Deep Learning can be used, for instance DeepEthogram classifies behaviors in a supervised manner from raw pixels [18] and another Deep Learning-based algorithm that estimates the pose of broiler chickens [57]. Convolutional networks are also very useful for classification and behavior detection using frames and currently also

3D convolutional networks for behavior automatic classification [128] and for grooming detection of mouse [173]. However, transfer learning is still used to train previously configured networks, thus decreasing the amount of data and computation time [95].

Another popular tool is LEAP [159], which also provides a pose estimation and tracking system based on deep neural networks. It follows 3 steps:

1. Registration and alignment of centroid to improve the efficiency and accuracy,
2. Labeling and training of images to create the ground truth to train the neural network and helps the system to find the body parts,
3. Pose estimation itself .

However, LEAP has been superseded by SLEAP (Social LEAP) that is able to track groups of animals in order to study social interactions between individuals. This social tool provides animal poses in a multi-animal system experimentation [158] using a type of convolutional neural network called DenseNet, where all layers are connected directly to each other [87]. DeepPoseKit [76] is another recent toolkit for animal pose estimation based on Stacked DenseNet, which is a variant of DenseNet.

6.3.2 Behavioral Classification

There are many open-source applications to annotate animal behaviors automatically based on AI methods, which can be either supervised or unsupervised. In the former type, the user has to tag the data to train the models, whereas in the latter type there is no need for prior labeling, reducing user bias.

A well-known software resource for behavior classification is JAABA [97], which is an open-source application based on a semi-supervised machine learning algorithm, where the user tags the behaviors of a part of the dataset and then the algorithm is able to label the rest of the dataset. Moreover, this system is used as support for human annotators in the manual process to obtain the ground truth to train other behavior classifiers. Events that occur during experiments with more than one animal can also be labeled [148]. Allowing free interaction among animals is a desirable feature. The Mouse Action Recognition System (MARS) [176] is an automated method for pose estimation and behavior quantification in couples of mice that can interact freely. MotionMapper [14] is another classical system for mapping animal actions from raw images. Once the images are segmented and aligned, a Morlet wavelet is applied to obtain the spectrogram for each postural mode. After a normalization, a watershed transform isolates the peaks to obtain behavioral regions where can be differentiate several movements such as fast leg movements, slow movements, wing movements, posterior movements, locomotion gits and anterior movements. This way, the behavior between males and females can be compared.

Deep Learning techniques are also extensively used for behavior recognition. DeepEthogram[18] classifies behaviors from raw pixels by applying a deep supervised learning algorithm composed of two convolutional neural networks extracting spatial and dynamic flow features. This classifier is widely used, for instance to extract walking or grooming events [63] or small-scale movements such as rat liking after having eaten tasty food [89]. In addition, transfer learning is very useful for building models from previously trained networks as it requires less computational time and less data [95].

There are unsupervised methods such as B-SOiD [86] that identifies behaviors without user interaction. Firstly, it extracts pose relationships to identify patterns and find the optimal number of cluster groups. Then, a random forest model is trained to predict categories of behaviors. This algorithm has been referenced in several works to classify repetitive behaviors using data from tracking beads, where the system is fed with data on distances, angles and velocities [119].

Unsupervised deep learning approaches have also found application in behavior categorization. The Selfee [94] method for self-supervised feature extraction can also be used as input to the B-SOiD algorithm. Selfee uses a convolutional neural network to extract features from raw video recordings to feed other classification algorithms. VAME [132] is another unsupervised deep learning framework that identifies behavioral motifs from bottom-up images. This algorithm uses DeepLabCut to estimate the pose of the animal and once trajectories are obtained, a recurrent neural network uses these trajectories to obtain the motifs. Finally, VAME results can be clustered with the k-means algorithm [171] in order to extract the relevant behavioral motifs. PyRAT [45] is an open-source python library to analyse animal behavior by estimating traveled distance, speed and area occupancy. The unsupervised algorithms used in this library are hierarchical agglomerative clustering and t-distributed stochastic neighbor embedding (t-SNE) for classification and clustering.

Apart from this, there are tools to detect behaviors based on heuristic algorithms such as BehaviorDEPOT, which uses statistics based on animal dynamics and posture [65]. There are also tools, where classification is carried out by using video recordings and a SVM is fed with a 32-dimensional feature vector that indicates the relative frequency of each of the 8 behaviors of interest to be analysed [93]. Finally, a k-NN is used to classify 7 behaviors, compute the total distance traveled and the percentage of time spent in the center from RGB-D images [71] .

6.4 Conclusion and Future Perspective

This Chapter includes an overview of the sensors currently used in experimentation, where the best known are RGB video and depth cameras, pressure sensors, RFID antennas, accelerometers and microphones to record mice vocalizations. This review also includes a compilation of the most widely used and known methods and algorithms in Computational Ethology for both tracking and classification of behaviors. Based on machine learning and Deep Learning, these methods can follow two basic learning approaches: supervised or unsupervised. In the first one, the data is previously labeled by the scientist and in the second approach the data is processed to look for common patterns of behavior and even discover unforeseen ones. While supervised methods may achieve greater precision in tracking, unsupervised methods have the advantage of not requiring labeled inputs in order to obtain relevant behavior categories.

Although the levels of precise measurement of behaviors achieved with current methods is very high, there are still limitations to overcome, such as the extension of the methods to large experimental arenas, which resemble natural spaces more closely.

Chapter 7

Materials for a Computational Ethology Experiment

This Chapter describes the source of data for the computational experiments testing several machine learning approaches to answer the Computational Ethology questions posed over the specific capabilities of a pressure sensor. These data were provided by Prof. Xavier Leinekugel, who maintains the property of the data, therefore all requests for data access should be addressed to him.

Section 7.1 provides an introduction to the Chapter and the definition of the problem addressed by Computational Ethology. Section 7.2 provides a summary description of the animal models. Section 7.3 provides a summary description of the behavior recording environment. Section 7.4 provides a description of the data processing leading to the feature extraction that is the input for the machine learning algorithms applied subsequently. Finally, Section 7.5 provides details on the classification approaches that have been explored in this study.

7.1 Introduction and Problem Statement

Ethology, which is the study of behavior, is key to understand the living things from a biological point of view, by characterising the behavior in natural and artificial environments [8]. Many of these studies are currently carried out in open-field cages where animals can move freely while their behavior is observed and analysed for characterization and quantification. In this way, differences in the behavior of subjects with distinct phenotypes can be found.

To record the animal behavior during these experiments there are a wide variety of sensors that can be either invasive, such as electroencephalogram (EEG), electrocardiogram (ECG) and inertial sensors surgically implanted in the animals or non-invasive, such as RGB, depth and infrared video cameras, and pressure sensors.

In the study reported here, data came from a recording system consisting of a top camera and a platform endowed with piezoelectric pressure sensors. The general aim of the experiments is to differentiate mice phenotypes on the basis of the recorded data. This study uses a Computer Vision-based algorithm to label mice behavior periods (such as locomotion, standing, or grooming) in an straightforward way. These periods are further used to segment the signal from the pressure sensors for strain or behavioral recognition of the mice.

There are nowadays many works that seek to differentiate phenotypes by analysing differences in specific behaviors such as grooming [72] or locomotion [99] or using combinations of them (drinking, rearing, grooming, eating, ...) [93][168]. However, to extract the periods of time in which the animal performs these behaviors is long and time-consuming [44] and although there are many applications in the literature that obtain these periods automatically, using modern Artificial Intelligence techniques and algorithms, it is still necessary to analyse a large part of the behaviors manually in order to train these models.

In this work we have chosen to analyse the animal models only during the locomotion periods, which are easily obtained without prior training by means of the images obtained by video camera. Once the locomotion periods are segmented, the next step is to obtain the features that characterise the behavior and compare them for different strains. This can be done either in the time or frequency domain. In this work we have chosen the frequency domain by calculating the spectrogram of the piezoelectric signal for the periods in which the animal is moving.

In summary, the **problem statement** is as follows: Is it possible to discriminate animal models using the pressure sensor signal corresponding to locomotion periods in an open field cage experiment.

This problem statement can be further refined into the following questions:

- Which are the appropriate features to be used for the classification task?
- How will we build up the datasets for the validation of the feature extraction and classification models?
- What are the duration of the recordings that can be useful for this task?
- Which classifiers achieve better results? We will be carrying out an exhaustive exploration of the performance

7.2 Animal Models

The computational experiments were carried out over data (camera and pressured signal recordings) recorded with 12 mice belonging to two different strains:

- 7 wild-type (WT)[1] mice with a total duration of recordings of 4 hours and 20 minutes
- 5 transgenic Fmr1-knockout (Fmr1-KO)[2] mice with a total duration of recordings of 4 hours and 20 minutes

All the recording took place during the light period and it was the first time these animal models were in contact with the recording system. All animals were bred in the laboratory animal facility in collective cages, and transferred to individual cages for the duration of the experiments. Animals were kept on a 12 h/12 h light/dark cycle, provided with nesting material and food and water *ad libitum*. All experiments were performed during the light period under constant mild luminosity (60-70 Lux). All experimental procedures were performed in accordance with EU directives relating to the protection of animals used for experimental and scientific purposes, in accordance with the reference to work [27].

7.3 Behavioral Data Acquisition

The Phenotypix platform [27] was used to record the experiments by Prof. Leinekugel research team. It is composed of an opaque-walled cage and a base resting on several piezoelectric pressure sensors with a sampling rate of 20 kHz. This platform is mounted on a table supported over a pressured air mechanism to filter out motion noise due to environment vibrations, making the sensor more sensitive and accurate. A Logitech HD Webcam C270 video camera is placed on top of the cage to record the experiment at 25 fps as shown in figure 7.3.1.

[1]Wild-type gene is a term used to describe a gene when it is found in its natural, non-mutated (unchanged) form.

[2]Fmr1-knockout (Fmr1-KO) mice may be useful for studying behavioral and synaptic abnormalities associated with Fragile X Syndrome.

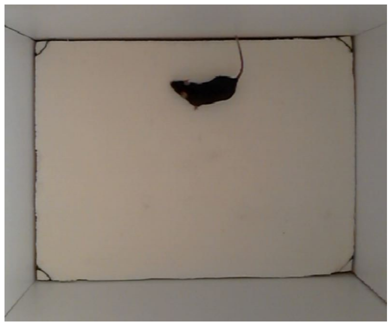


Figure 7.3.1: The Phenotypix platform during a recording session.

Animals were introduced individually into the platform and the walls and cage were cleaned with 70% ethanol before each recording. To visualise the piezoelectric signal obtained during the recordings, Spike2 software (CED, Cambridge, UK) was connected to a computer where the data were stored for later analysis. At the same time the videos were recorded with the top camera and stored separately from the signals. The data were processed with Bonsai [130], MATLAB (Mathworks, Natik, MA, USA) and Python [191]. The length of the recordings range from 20 minutes to 1 hour.

7.4 Data Processing

Once the data are collected after the acquisition phase, they are processed to extract the features for the classification algorithms. The pipeline of this process is shown in figure 7.4.1 and is described in detail below:

1. Check whether the video has the correct format and duration. If not the multimedia framework ffmpeg [189] is used to change the video format and correct its duration by changing the frame rate. This framework can benefit from GPU acceleration and its functions can be automated with Python or MATLAB scripts.

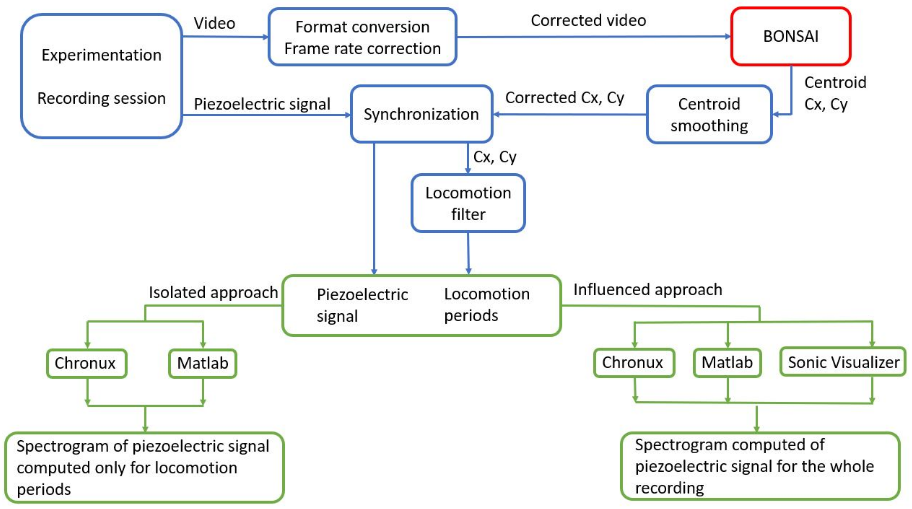


Figure 7.4.1: Data processing pipeline.

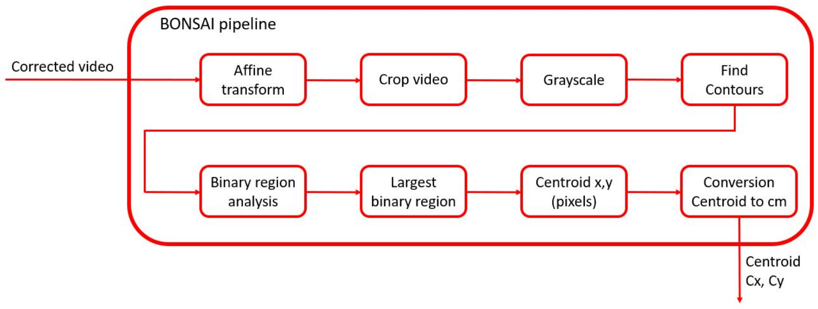


Figure 7.4.2: Bonsai pipeline for video processing.

2. Compute the x and y position of the animal centroid through videos using Bonsai software carrying out the steps shown in figure 7.4.2:

   (a) Apply an affine transformation to align the video with the window frame.

   (b) Crop the video to center the region of interest to the platform on which the animal will move. This way we can delete all the space out of the recording system.

   (c) Apply an binarization to change colors to gray scale. This step facilitates the animal segmentation. The function checks if the color is darker than a predefined threshold to paint the pixel white and otherwise black. The result is an image in black and white where the animal is white and the background is in black. In case there are objects apart from the animal, these object will be seen in white too.

   (d) Find contours based on the color changes.

   (e) Take binary region and select the largest one to avoid other objects out of interest.

   (f) Compute the centroid of the largest binary region in pixels.

   (g) Convert centroid from pixels to cm by using a proportional relationship taking into account the size of the recording platform.

3. Smooth the x and y centroid computing the average of 3 consecutive points. This eliminates possible positional errors and filter the trajectory of the animal.

4. Synchronize the time scale from of the centroid and the piezoelectric signal, taking into account the time offset between both time scales and skipping the first seconds of the video before the animal appears on the recording platform.

5. Apply a locomotion filter to detect periods when the animal is walking using a MATLAB script with a minimum velocity of 2.5 cm/s and a minimum distance of 2.5 cm for each period.

After obtaining the locomotion periods, the spectrograms of the piezoelectric signal can be calculated in two different ways:

- the spectrogram is calculated independently for each of the pressure signal chunks corresponding to each locomotion period detected on the video, or

- the spectrogram is calculated for the whole pressure signal, and chunks of the spectrogram are extracted corresponding to each locomotion period detected on the video.

In the first approach, the spectrogram will be calculated in isolation without taking into account the surrounding behaviors, while in the second approach, the spectrogram will be influenced by the previous and subsequent behaviors.

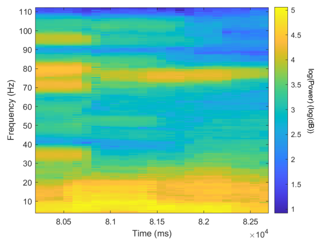
For both approaches, the spectrogram can be computed using the Chronux library[3], the Signal Processing Toolbox of MATLAB, and the graphic software Sonic Visualizer. Specifically, the spectrogram is computed with Chronux and MATLAB libraries when dealing with the segmented pressure signal chunks corresponding to the locomotion periods. For the second approach, the spectrogram computed by Sonic Visualizer is used in addition to Chronux and MATLAB spectrogram routines. Besides, spectrogram images are also obtained to compare the three methods using transfer learning.

In the Chronux library, tuning parameters are window size, window step, number of tapers, and frequency of interest. The window size is defined as how long the number of samples is in every computation. The window step is the size of the displacement to take the next window sample. Modifying the number

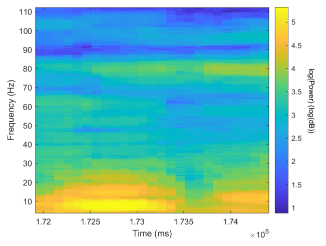
[3] http://chronux.org/

Table 7.1: Parameters for spectrogram computation with Chronux library.

Parameters	Value 1	Value 2	Default values
Window size (s)	1	2	-
Windows step (s)	0.1	0.2	-
Tapers	[4, 2]	[3, 5]	[3, 5]: A numeric vector [TW K] where TW is the time-bandwidth product and K is the number of tapers, less than or equal to 2TW-1
Frequency of interest (Hz)	[1.5 - 40]	[4 - 112]	[0 - Fs/2] (Fs: sampling frequency)



(a)



(b)

Figure 7.4.3: Spectrogram computed with Chronux for a segmented pressure signal chunk in locomotion time for a) wild type and b) transgenic Fmr1-KO mouse.

of tapers is a way to control the degree of the smoothing in the spectrogram and the frequency of interest allows to limit the frequency range. Table 7.1 shows the parameters used in the processing with Chronux library. Figure 7.4.3 shows two spectrogram chunks obtained from the locomotion periods for a WT (a) and transgenic Fmr1-KO (b) mouse models with Chronux. This spectrogram has been calculated for a frequency band of interest of 1.5-40 Hz, a window size of 2 s, a window step 0.5 s and number of tapers [3,5].

Table 7.2: Parameters for computation with the MATLAB *spectrogram* function.

Parameters	Value 1	Value 2	Value 3	Range of values
Number of sections	8	4	-	Integer
Overlap	0.5	0.1	-	[0 - less than window]
Window	Hamming	Chebyshev	Tukey	[Bartlett-Hann, Bartlett, Gaussian ,..., triangular]

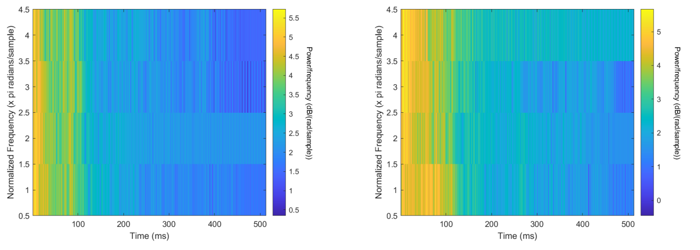


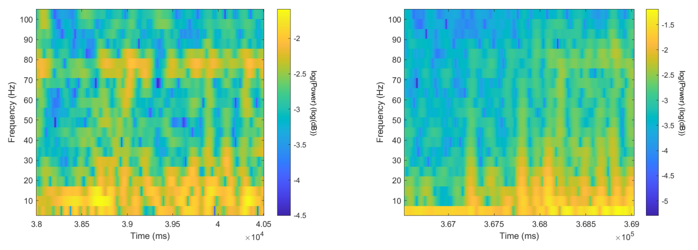
Figure 7.4.4: Spectrogram computed with MATLAB for a segmented signal chunk in locomotion time for a) wild type and b) transgenic Fmr1-KO mouse.

Table 7.3: Parameter for spectrogram computation with Sonic Visualizer.

Parameter	Value	Range of values
Colour	Green	[Green, Sunset, ... , Wasp, Ice, ...]
Scale	dB	[Linear, Meter, dB^2, dB, Phase]
Window size	256	[32, 64, 128, 256, 512, ... , 16384, 32768]
Overlap	93.75%	[none, 25%, 50%, 75%, 87.5%, 93.75%]
Show	All bins	[All Bins, Peak Bins, Frequencies]
Scale	Linear	[Linear, Log]

The MATLAB *spectrogram* function tuning parameters are the type of window (Hamming, Chebyshev, Tukey, Gaussian, ...), the number of overlapped samples, number of DFT points (number of frequency points used to calculate the discrete Fourier transforms), the sample rate, and the number of sections. In this work the number of sections, the overlap percent and the type of window will take different values to compare the results, these values are in the table 7.2. Figure 7.4.4 shows two chunks of spectrograms obtained from the locomotion periods for a WT (a) and transgenic Fmr1-KO (b) mouse model with MATLAB. This spectrogram has been calculated using a Hamming window, 4 sections and an overlap of 0.1.

Finally, in the Sonic Visualizer spectrogram function the parameter tuning was carried out in a manual interaction searching for a configuration that highlights the visual differences in the spectrogram for different strains. This



(a) (b)

Figure 7.4.5: Spectrogram computed with Sonic Visualizer for a segmented signal chunk in locomotion time for a) wild type and b) transgenic Fmr1-KO mouse.

configuration is shown in the table 7.3 and the figure 7.4.5 shows two spectrogram chunks obtained from the locomotion periods for a WT (a) and transgenic Fmr1-KO (b) mouse models with Sonic Visualizer.

7.5 Classification Methods

For feature classification we have exploited the different machine learning-based methods implemented in MATLAB namely decision trees, linear discriminant analysis, logistic regression, Gaussian Naive Bayes, SVM, k-NN, boosted trees, bagged trees, subspace discriminant, subspace k-NN and RUSBoosted trees.

Additionally, a Multi-layer Perceptron (MLP) was implemented with Python-Keras libraries. This MLP has four hidden layers and its configuration is shown in the table 7.4. Depending on the feature extractor, the input layer dimension varies. The output layer is computed with only one neuron and the binary classification result will be Wild Type model if the output is less or equal to 0.5, and Fmr1-KO model if the output is more than 0.5. Another alternatives can be chosen to compute the output such as selecting another threshold instead of 0.5, or computing two outputs and adding a softmax layer. All the parameters were chosen by means of a grid search and selecting the best result: 255 batch

Table 7.4: MLP configuration.

Layer	Layer type	Neurons	Activation function	Dropout normalization
Input layer	-	variable	-	-
First hidden layer	Dense	400	relu	0.2
Second hidden layer	Dense	200	relu	-
Third hidden layer	Dense	60	relu	-
Fourth hidden layer	Dense	35	relu	-
Output layer	Dense	1	sigmoid	-

size, 700 epochs, dropout normalization of 0.2 in the input layer and Adam optimizer.

7.5.1 Transfer learning

For the calculation of classification models over images of spectrograms, we have used pre-trained convolutional networks. This process, known as transfer learning, makes it possible to use a network that has already been trained with a large amount of data and high hardware capacity, so that only the last layer needs to be trained to adapt it to a new data set. This takes advantage of the resources of other research teams and it reduces training time as only part of the network needs to be trained instead of training from scratch. Thus, the performance of three convolutional networks has been explored in this work:

- AlexNet [111] is a well-known convolutional neural network which won the ImageNet Large-Scale Visual Recognition Challenge (ILSVRC) in 2012. ImageNet is a dataset of over 15 million labeled high-resolution images belonging to 22,000 categories. However, a subset of ImageNet with 1000 images for 1000 categories was used for the ILSVRC. The architecture of AlexNet is composed of five convolutional layers and three fully connected layers with a final 1000-way softmax.

- GoogLeNet [185] is a deep convolutional neural network with 22 layers

Table 7.5: Training parameters for transfer learning approach.

Parameter	Value
Solver	Adam
Learning rate	0.0001
Mini batch size	52
L2 Regularization	0.0001
Folds for Cross-validation	5

Table 7.6: Number of layers and learnables parameters for AlexNet, ResNet50 and GoogLeNet models.

Algorithm	Layers	Total learnables
AlexNet	25 (depth 8)	56 876 418
ResNet50	177 (depth 50)	23 538 690
GoogLeNet	144 (depth 22)	5 975 602

winner of ILSVRC in image classification and detection in 2014. This network is based on the Inception architecture that consists of making the network wider than deeper in order to take into account different scales.

- ResNet50 [79] is a deep convolutional neural network which uses learning residual functions to train the model. It won the ILSVRC15 in image detection and localization with the ImageNet dataset doing the task of classifying the image into one of the 1000 categories in the ImageNet hierarchy.

The spectrogram images previously generated during the locomotion periods will be used for training the neural networks with the training parameters shown in the table 7.5. The number of layers and the total of learnables are shown in the table 7.6.

7.5.2 Model Training and Evaluation

The question presented in this contribution is formulated as a binary classification problem that aims to discriminate between wild type (0 class) and Fmr1-KO

Table 7.7: Execution time for training and evaluation (cross validation).

Algorithm	Execution time
Machine Learning algorithms	~ 1s
MLP	~ 2min
AlexNet	~ 35min
ResNet50	~ 4h
GoogLeNet	~ 2h

(1 class) models.

The dataset is divided into two parts, one for training and one for testing with a proportion of 80% and 20%, respectively.

To train the models, 5-fold cross-validation has been applied over the training dataset, dividing the dataset into 5 folds. For each of the 5 training sessions, a model is trained with 4 folds and the 5th is used as development set, repeating this process alternating the development set and obtaining 5 different models.

For validation, the best of the 5 previous models is chosen according to the accuracy and it is applied to the test set. In addition to accuracy, other metrics have been evaluated in the test set to compare the algorithms: Area Under the Curve (AUC), precision and recall.

The input dimension is variable depending on the method and the parameters used to calculate the spectrogram. For image training, the dimensions must be 227 x 227 x 3 for AlexNet and 224 x 224 x 3 for ResNet50 and GoogLeNet.

The algorithms were implemented in MATLAB and Python with Keras library using an Intel(R) Core(TM) i9-9880H computer with 64 GB of RAM and a NVIDIA Quadro RTX 3000 GPU. The execution times for training and validation are illustrated in the table 7.7.

Chapter 8

Results of Computational Ethology Experiments

After obtaining the features and images for each configuration showed in the section 7.4, we have trained the models explained in the section 7.5. In this chapter we report the classification performance results of the wide computational experiments in terms of accuracy and area under curve (AUC) of the models for the dataset validation. The results are divided into two main sections. Section 8.1 provides exploratory results for different minimum locomotion times that allow to determine the optimal duration of locomotion times regarding classification performance. Section 8.2 provides results for different configurations and parameters of feature extraction from the spectrograms. Finally, Section 8.3 provides a discussion of the results in the context of Computational Ethology.

Table 8.1: Accuracy for different minimum locomotion duration.

Minimum locomotion time	1000 ms	1500 ms	2000 ms	2500 ms
window size - step (ms) # samples x features	1000 - 100 15012 x 2056	1500 - 100 6393 x 2056	2000 - 100 2465 x 3591	2500 - 100 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	70.60%	73.90%	76.90%	77.50%
Medium Tree	70.30%	71.40%	73.20%	79.10%
Coarse Tree	66.90%	67.90%	67.70%	72.70%
Linear Discriminant	82.40%	82.70%	84.40%	99.50%
Logistic Regression	82.20%	82.80%	57.40%	53.50%
Gaussian Naive Bayes	68.20%	70.30%	72.40%	79.70%
Linear SVM	84.90%	88.30%	90.90%	93.60%
Quadratic SVM	87.70%	91.50%	94.30%	95.20%
Cubic SMV	88.40%	92.40%	94.50%	98.40%
Fine Gaussian SVM	69.60%	65.70%	64.90%	54.00%
Medium Gaussian SVM	86.70%	89.80%	90.50%	91.40%
Coarse Gaussian SVM	79.90%	80.40%	79.70%	82.40%
Fine kNN	74.20%	87.40%	92.90%	98.40%
Medium kNN	74.40%	79.00%	81.70%	80.70%
Coarse kNN	72.30%	71.70%	66.70%	61.00%
Cosine kNN	75.10%	81.20%	87.80%	87.20%
Cubic kNN	75.80%	78.70%	81.90%	81.80%
Weighted kNN	76.20%	82.60%	86.20%	85.60%
Boosted Trees	78.20%	81.40%	85.60%	90.90%
Bagged Trees	80.70%	84.90%	86.60%	89.30%
Subspace Discriminant	83.80%	87.90%	88.20%	93.00%
Subspace kNN	69.20%	86.60%	96.10%	97.90%
RUSBoosted Trees	76.20%	76.50%	77.30%	88.20%
MLP	91.58%	93.90%	93.71%	93.62%

Table 8.2: AUC for different minimum locomotion duration.

Minimum locomotion time	1000 ms	1500 ms	2000 ms	2500 ms
window size - step (ms) # samples x features	1000 - 100 15012 x 2056	1500 - 100 6393 x 2056	2000 - 100 2465 x 3591	2500 - 100 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.77	0.81	0.77	0.80
Medium Tree	0.74	0.79	0.78	0.82
Coarse Tree	0.70	0.69	0.71	0.76
Linear Discriminant	0.90	0.91	0.84	1.00
Logistic Regression	0.90	0.86	0.57	0.54
Gaussian Naive Bayes	0.69	0.71	0.74	0.79
Linear SVM	0.93	0.96	0.97	0.98
Quadratic SVM	0.95	0.98	0.99	0.99
Cubic SMV	0.95	0.98	0.99	1.00
Fine Gaussian SVM	0.81	0.84	0.90	0.99
Medium Gaussian SVM	0.94	0.96	0.97	0.98
Coarse Gaussian SVM	0.90	0.90	0.90	0.91
Fine kNN	0.75	0.88	0.93	0.98
Medium kNN	0.86	0.91	0.94	0.95
Coarse kNN	0.86	0.86	0.87	0.88
Cosine kNN	0.84	0.90	0.94	0.95
Cubic kNN	0.87	0.91	0.94	0.94
Weighted kNN	0.87	0.93	0.96	0.97
Boosted Trees	0.87	0.91	0.94	0.96
Bagged Trees	0.90	0.92	0.94	0.96
Subspace Discriminant	0.92	0.96	0.96	0.99
Subspace kNN	0.77	0.93	0.99	0.99
RUSBoosted Trees	0.85	0.85	0.85	0.96
MLP	0.91	0.94	0.94	0.94

8.1 Results for Different Minimum Locomotion Duration

The process of animal model classification is based on the data from locomotion periods. These periods are used for signal segmentation and compute the ensuing spectrogram. However, some locomotion periods may be too short to be taken into account, as the stationary part of the signal is negligible compared to the transient parts of the edges. The aim of this section is to identify the optimal minimum size of the segmented locomotion period under consideration.

For comparison, the signal slices were segmented according to the minimum locomotion time and the spectrogram was calculated only for that part of the signal using the MATLAB *spectrogram* function with the following parameter settings:

- Hamming window
- 8 number of sections
- Overlap 50%

Tables 8.1 and 8.2 show the accuracy and AUC results, respectively, for test set validation after filtering locomotion periods longer than 1000, 1500, 2000 and 2500 ms. No experiments have been performed for duration longer than 2500 ms because the resulting number of samples is very small compared to the number of features (366 x 3592).

The best prediction results are obtained for a minimum locomotion duration of 2500 ms, where linear discriminant returns 99.50% accuracy, cubic SVM 98.40% and subspace k-NN 97.90%, among others. Regarding the AUC, the best results are achieved also for a minimum locomotion duration of 2500 ms, and the most remarkable results are 1.00 AUC for linear discriminant and cubic

SVM, or 0.99 for quadratic SVM, fine Gaussian SVM, subspace discriminant and subspace k-NN. Consequently, in the ensuing experiments, the minimum locomotion duration was set to 2500 ms for signal or spectrogram segmentation.

8.2 Results for Different Spectrogram Features and Images

To obtain a dataset containing the spectrogram features or images for the discrimination among animal models, we have followed two radically different approaches. In the first approach, localized spectrograms are obtained from chunks of the piezoelectric signal corresponding to the selected locomotion periods that must be longer than the minimum time threshold. In the second approach, the spectrogram of the whole signal is calculated first, and, then the time periods corresponding to the valid locomotion periods are segmented from this large spectrogram.

This section is divided into three subsections, reporting results on the diverse approaches of dataset construction:

1. Feature classification of the spectrogram calculated only in segmented chunks using MATLAB and Chronux libraries.

2. Feature classification of the spectrogram calculated for the whole signal using MATLAB and Chronux libraries and Sonic Visualizer.

3. Image classification of the spectrogram calculated for the whole signal using MATLAB and Chronux libraries and Sonic Visualizer.

The figure 8.2.1 shows a diagram with the structure of the experiments reported in this Chapter:

The following parameters have been used for the experiments:

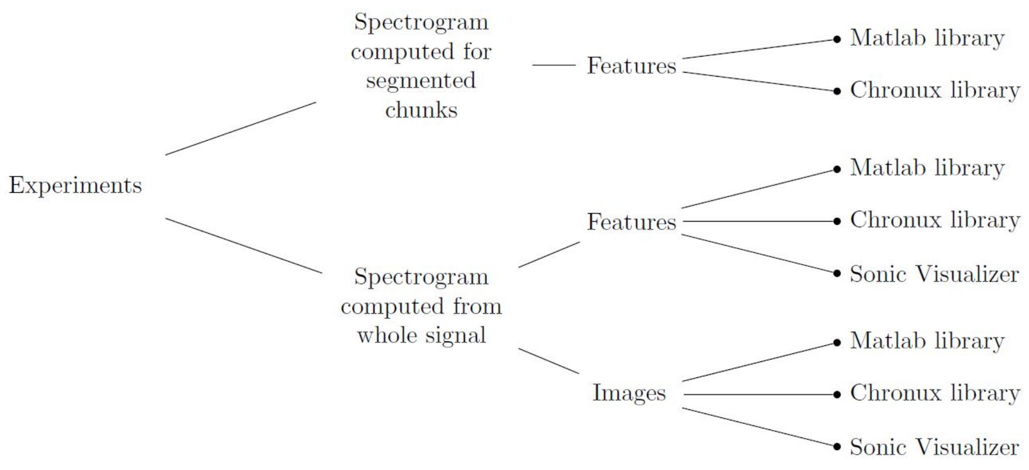


Figure 8.2.1: Structure of the experiments.

- Minimum locomotion duration of 2500 ms
- Sampling window size of 2500 ms
- Window step of 100 ms

8.2.1 Results of Segmented Chunks Spectrogram Feature Classification. Spectrogram from MATLAB, and Chronux.

In this section we show the classification results for the test set using the features obtained with MATLAB and Chronux spectrograms only for those periods previously segmented as locomotion periods.

MATLAB Spectrogram Features For the computation of the spectrograms, different parameters have been studied: the type of window, the number of sections and the overlap, as indicated in the table 7.2.

Table 8.3: Accuracy for spectrogram computed only for segmented chunks with Matlab library and Hamming window.

Window type	Hamming			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	78.10%	73.30%	75.40%	77.50%
Medium Tree	78.10%	77.50%	74.90%	79.10%
Coarse Tree	73.30%	77.00%	77.00%	72.70%
Linear Discriminant	95.70%	80.70%	98.40%	99.50%
Logistic Regression	60.40%	55.10%	70.10%	53.50%
Gaussian Naive Bayes	70.10%	78.10%	83.40%	79.70%
Kernel Naive Bayes	77.00%	80.20%	90.40%	82.90%
Linear SVM	90.90%	85.60%	92.50%	93.60%
Quadratic SVM	95.70%	91.40%	95.70%	95.20%
Cubic SMV	95.20%	93.00%	96.30%	98.40%
Fine Gaussian SVM	57.20%	54.00%	58.80%	54.00%
Medium Gaussian SVM	92.50%	87.70%	95.20%	91.40%
Coarse Gaussian SVM	74.30%	78.60%	84.00%	82.40%
Fine kNN	97.30%	92.50%	96.80%	98.40%
Medium kNN	79.70%	76.50%	87.20%	80.70%
Coarse kNN	61.50%	59.40%	66.80%	61.00%
Cosine kNN	86.60%	84.00%	88.80%	87.20%
Cubic kNN	83.40%	77.00%	86.10%	81.80%
Weighted kNN	85.60%	82.40%	89.30%	85.60%
Boosted Trees	87.20%	81.30%	92.50%	90.90%
Bagged Trees	89.30%	87.20%	94.10%	89.30%
Subspace Discriminant	96.30%	86.10%	97.90%	93.00%
Subspace kNN	97.30%	94.70%	97.30%	97.90%
RUSBoosted Trees	80.70%	83.40%	89.30%	88.20%
MLP	95.74%	91.49%	96.28%	93.62%

Table 8.4: AUC for spectrogram computed only for segmented chunks with Matlab library and Hamming window.

Window type	Hamming			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.80	0.75	0.77	0.80
Medium Tree	0.82	0.78	0.75	0.82
Coarse Tree	0.74	0.80	0.80	0.76
Linear Discriminant	0.96	0.80	0.99	1.00
Logistic Regression	0.64	0.53	0.72	0.54
Gaussian Naive Bayes	0.70	0.79	0.84	0.79
Kernel Naive Bayes	0.79	0.82	0.91	0.86
Linear SVM	0.97	0.93	0.98	0.98
Quadratic SVM	0.99	0.97	0.99	0.99
Cubic SMV	0.99	0.97	1.00	1.00
Fine Gaussian SVM	0.97	0.94	0.99	0.99
Medium Gaussian SVM	0.96	0.96	0.98	0.98
Coarse Gaussian SVM	0.85	0.87	0.91	0.91
Fine kNN	0.97	0.92	0.97	0.98
Medium kNN	0.93	0.92	0.94	0.95
Coarse kNN	0.85	0.88	0.90	0.88
Cosine kNN	0.93	0.94	0.96	0.95
Cubic kNN	0.94	0.93	0.96	0.94
Weighted kNN	0.96	0.94	0.97	0.97
Boosted Trees	0.96	0.92	0.98	0.96
Bagged Trees	0.95	0.92	0.98	0.96
Subspace Discriminant	0.99	0.94	0.99	0.99
Subspace kNN	0.99	0.97	0.98	0.99
RUSBoosted Trees	0.90	0.92	0.97	0.96
MLP	0.96	0.91	0.96	0.94

Table 8.5: Accuracy for spectrogram computed only for segmented chunks with Matlab library and Tukey window.

Window type	Tukey			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	82.40%	71.10%	84.50%	86.60%
Medium Tree	81.80%	77.00%	85.00%	86.10%
Coarse Tree	77.00%	74.90%	74.30%	79.70%
Linear Discriminant	97.30%	93.60%	100.00%	99.50%
Logistic Regression	57.20%	56.10%	67.90%	52.90%
Gaussian Naive Bayes	86.60%	74.30%	86.60%	81.30%
Kernel Naive Bayes	88.80%	77.00%	89.80%	82.40%
Linear SVM	95.20%	90.40%	97.90%	96.80%
Quadratic SVM	98.90%	96.80%	99.50%	98.90%
Cubic SMV	98.90%	97.90%	100.00%	99.50%
Fine Gaussian SVM	70.10%	54.00%	97.30%	59.90%
Medium Gaussian SVM	98.40%	92.00%	98.40%	97.30%
Coarse Gaussian SVM	88.20%	76.50%	89.30%	83.40%
Fine kNN	96.80%	96.80%	100.00%	99.50%
Medium kNN	86.10%	81.30%	90.40%	89.80%
Coarse kNN	67.40%	64.70%	73.80%	63.10%
Cosine kNN	93.00%	86.10%	92.00%	92.50%
Cubic kNN	87.20%	81.30%	90.90%	89.30%
Weighted kNN	95.70%	87.70%	98.90%	97.90%
Boosted Trees	96.30%	85.60%	96.30%	93.00%
Bagged Trees	94.70%	87.20%	93.00%	93.60%
Subspace Discriminant	98.40%	93.00%	100.00%	98.40%
Subspace kNN	100.00%	92.50%	99.50%	98.40%
RUSBoosted Trees	86.10%	84.00%	90.40%	88.20%
MLP	98.94%	97.87%	100.00%	97.34%

Table 8.6: AUC for spectrogram computed only for segmented chunks with Matlab library and Tukey window.

Window type	Tukey			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.86	0.73	0.86	0.87
Medium Tree	0.82	0.74	0.87	0.84
Coarse Tree	0.81	0.81	0.81	0.81
Linear Discriminant	0.97	0.93	1.00	0.99
Logistic Regression	0.59	0.57	0.70	0.54
Gaussian Naive Bayes	0.87	0.74	0.87	0.81
Kernel Naive Bayes	0.91	0.80	0.91	0.85
Linear SVM	0.99	0.97	1.00	0.99
Quadratic SVM	1.00	1.00	1.00	1.00
Cubic SMV	1.00	1.00	1.00	1.00
Fine Gaussian SVM	1.00	0.93	1.00	0.99
Medium Gaussian SVM	1.00	0.97	1.00	1.00
Coarse Gaussian SVM	0.95	0.86	0.95	0.93
Fine kNN	0.97	0.97	1.00	0.99
Medium kNN	0.97	0.94	0.98	0.99
Coarse kNN	0.89	0.86	0.91	0.91
Cosine kNN	0.98	0.95	0.98	0.99
Cubic kNN	0.97	0.91	0.98	0.99
Weighted kNN	1.00	0.96	1.00	1.00
Boosted Trees	1.00	0.93	0.99	0.99
Bagged Trees	0.99	0.93	0.99	0.99
Subspace Discriminant	1.00	0.98	1.00	0.99
Subspace kNN	1.00	0.98	1.00	1.00
RUSBoosted Trees	0.96	0.92	0.97	0.96
MLP	0.99	0.98	1.00	0.97

Table 8.7: Accuracy for spectrogram computed only for segmented chunks with Matlab library and Chebyshev window.

Window type	Chebyshev			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	80.70%	73.30%	85.60%	79.10%
Medium Tree	80.20%	72.70%	85.00%	78.10%
Coarse Tree	81.30%	74.90%	80.70%	77.00%
Linear Discriminant	92.50%	75.90%	97.90%	94.10%
Logistic Regression	51.90%	52.40%	65.80%	50.80%
Gaussian Naive Bayes	79.70%	76.50%	85.00%	75.40%
Kernel Naive Bayes	81.30%	75.90%	85.60%	74.90%
Linear SVM	92.00%	86.60%	95.70%	90.40%
Quadratic SVM	94.70%	91.40%	97.90%	96.30%
Cubic SMV	94.70%	90.40%	99.50%	96.80%
Fine Gaussian SVM	54.50%	54.00%	64.70%	54.00%
Medium Gaussian SVM	90.40%	87.70%	97.90%	90.40%
Coarse Gaussian SVM	82.40%	75.90%	86.10%	75.40%
Fine kNN	96.80%	84.00%	97.30%	97.30%
Medium kNN	86.60%	76.50%	88.20%	81.80%
Coarse kNN	69.00%	61.50%	69.00%	65.20%
Cosine kNN	90.90%	87.20%	93.00%	89.30%
Cubic kNN	85.60%	78.10%	86.60%	81.30%
Weighted kNN	90.90%	80.70%	96.80%	88.80%
Boosted Trees	92.50%	86.60%	94.70%	86.10%
Bagged Trees	88.80%	84.50%	92.00%	84.50%
Subspace Discriminant	94.10%	81.80%	94.70%	94.70%
Subspace kNN	97.30%	84.00%	98.90%	95.70%
RUSBoosted Trees	84.00%	84.50%	86.60%	84.00%
MLP	94.68%	87.77%	98.94%	92.55%

Table 8.8: AUC for spectrogram computed only for segmented chunks with Matlab library and Chebyshev window.

Window type	Chebyshev			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.81	0.74	0.86	0.79
Medium Tree	0.84	0.77	0.82	0.79
Coarse Tree	0.86	0.77	0.79	0.81
Linear Discriminant	0.92	0.76	0.98	0.94
Logistic Regression	0.55	0.52	0.66	0.52
Gaussian Naive Bayes	0.80	0.76	0.85	0.75
Kernel Naive Bayes	0.84	0.78	0.86	0.78
Linear SVM	0.97	0.93	0.99	0.97
Quadratic SVM	0.99	0.96	1.00	1.00
Cubic SMV	0.99	0.94	1.00	1.00
Fine Gaussian SVM	0.96	0.88	1.00	0.95
Medium Gaussian SVM	0.97	0.95	1.00	0.97
Coarse Gaussian SVM	0.90	0.87	0.94	0.86
Fine kNN	0.97	0.83	0.98	0.97
Medium kNN	0.96	0.90	0.97	0.94
Coarse kNN	0.86	0.86	0.91	0.87
Cosine kNN	0.97	0.93	0.98	0.95
Cubic kNN	0.96	0.89	0.97	0.93
Weighted kNN	0.98	0.91	0.99	0.95
Boosted Trees	0.96	0.93	0.99	0.92
Bagged Trees	0.95	0.91	0.98	0.91
Subspace Discriminant	0.99	0.91	0.99	0.99
Subspace kNN	1.00	0.93	1.00	0.98
RUSBoosted Trees	0.92	0.90	0.92	0.91
MLP	0.94	0.87	0.99	0.93

Tables 8.3 and 8.4 show the accuracy and AUC results, respectively, after training the models and validating them with the test set, using a Hamming window. Similarly, tables 8.5, 8.6, 8.7 and 8.8 show the results of accuracy and AUC, for a Tukey and Chebyshev window, respectively.

As can be seen in the tables, for the Hamming window, 97.30% accuracy is obtained for the subspace k-NN and fine k-NN methods with 4 sections and 0.1 overlap. For the Tukey window, 100% accuracy is observed for subspace k-NN with 4 sections and 0.1 overlap, in linear discriminant, subspace discriminant, cubic SVM and fine k-NN with 4 sections and 0.5 overlap. For the Chebyshev window, results of 97.30% for subspace k-NN and 96.80% for fine k-NN with 4 sections and 0.1 overlap are obtained.

Chronux Spectrogram Features For the calculation of spectrograms with Chronux, different parameters can be modified: the frequency of interest, the number of tapers and the window to calculate the spectrogram, its size and step. In this work we have used the parameters shown in table 7.1.

The tables 8.9 and 8.10 show the accuracy and AUC for the range 1.5-40 Hz and for different number of tapers and window step and size. For this frequency range, the best results are 98.90% accuracy with fine k-NN for window parameters of [1, 0.1] seconds and tapers [4, 2], 98.90% accuracy for fine k-NN, or 97.90% with subspace k-NN for window parameters of [2, 0.2] seconds and tapers [4, 2] or 98.40% accuracy with the MLP and subspace k-NN, or 97.30% with fine k-NN for window parameters of [2, 0.2] seconds and tapers [3, 5].

Similarly, the tables 8.11 and 8.12 show the accuracy and AUC for the range 4-112 Hz and for different number of tapers and window step and size. Where the best results are 100% accuracy with fine k-NN for window parameters of [1, 0.1] seconds and tapers [4, 2], 97.30% accuracy for subspace k-NN, or 96.8% accuracy for linear discriminant, weighted k-NN and subspace discriminant for

Table 8.9: Accuracy for spectrogram computed only for segmented chunks with Chronux library and frequencies from 1.5 to 40 Hz.

Frequency range	1.5 - 40 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4, 2 938 x 1008	2, 0.2 - 4, 2 938 x 381	1, 0.1 - 3, 5 938 x 1008	2, 0.2 - 3, 5 938 x 381
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	79.70%	75.90%	88.80%	84.00%
Medium Tree	81.30%	74.90%	87.70%	84.00%
Coarse Tree	77.00%	73.80%	82.40%	79.70%
Linear Discriminant	59.40%	86.10%	59.40%	84.50%
Logistic Regression	49.20%	84.50%	51.90%	82.40%
Gaussian Naive Bayes	65.80%	66.30%	70.10%	69.50%
Kernel Naive Bayes	68.40%	73.80%	71.10%	76.50%
Linear SVM	75.40%	78.10%	79.70%	80.70%
Quadratic SVM	85.60%	89.30%	88.80%	88.20%
Cubic SMV	86.60%	92.00%	93.60%	95.20%
Fine Gaussian SVM	79.10%	89.30%	84.50%	93.00%
Medium Gaussian SVM	84.00%	87.70%	84.50%	86.60%
Coarse Gaussian SVM	67.90%	74.30%	72.70%	71.70%
Fine kNN	98.90%	98.90%	97.30%	97.30%
Medium kNN	75.90%	80.70%	79.10%	87.20%
Coarse kNN	67.90%	66.80%	70.10%	73.30%
Cosine kNN	73.80%	84.00%	81.30%	90.40%
Cubic kNN	76.50%	82.90%	78.10%	87.70%
Weighted kNN	86.10%	95.20%	86.60%	96.30%
Boosted Trees	90.40%	92.00%	89.80%	92.50%
Bagged Trees	92.00%	88.20%	88.80%	94.10%
Subspace Discriminant	72.20%	85.60%	73.30%	84.00%
Subspace kNN	97.30%	97.90%	95.70%	98.40%
RUSBoosted Trees	88.20%	84.00%	82.40%	90.90%
MLP	93.62%	96.81%	97.34%	98.40%

Table 8.10: AUC for spectrogram computed only for segmented chunks with Chronux library and frequencies from 1.5 to 40 Hz.

Frequency range	1.5 - 40 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4, 2 938 x 1008	2, 0.2 - 4, 2 938 x 381	1, 0.1 - 3, 5 938 x 1008	2, 0.2 - 3, 5 938 x 381
	AUC	AUC	AUC	AUC
Fine Tree	0.79	0.75	0.87	0.87
Medium Tree	0.85	0.75	0.89	0.86
Coarse Tree	0.78	0.76	0.85	0.81
Linear Discriminant	0.59	0.91	0.59	0.88
Logistic Regression	0.50	0.85	0.52	0.86
Gaussian Naive Bayes	0.65	0.66	0.69	0.71
Kernel Naive Bayes	0.72	0.76	0.75	0.77
Linear SVM	0.78	0.87	0.89	0.86
Quadratic SVM	0.89	0.95	0.95	0.95
Cubic SMV	0.90	0.97	0.97	0.99
Fine Gaussian SVM	0.97	0.97	0.97	0.99
Medium Gaussian SVM	0.88	0.93	0.94	0.93
Coarse Gaussian SVM	0.76	0.79	0.84	0.81
Fine kNN	0.99	0.99	0.97	0.97
Medium kNN	0.89	0.95	0.89	0.95
Coarse kNN	0.73	0.74	0.79	0.81
Cosine kNN	0.88	0.95	0.88	0.96
Cubic kNN	0.89	0.94	0.87	0.95
Weighted kNN	0.97	1.00	0.95	0.98
Boosted Trees	0.96	0.98	0.95	0.98
Bagged Trees	0.97	0.95	0.95	0.98
Subspace Discriminant	0.78	0.91	0.79	0.94
Subspace kNN	0.99	1.00	0.97	0.99
RUSBoosted Trees	0.96	0.93	0.91	0.97
MLP	0.93	0.97	0.97	0.98

Table 8.11: Accuracy for spectrogram computed only for segmented chunks with Chronux library and frequencies from 4 to 112 Hz.

Frequency range	4 - 112 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4, 2 938 x 2832	2, 0.2 - 4, 2 938 x 1062	1 0.1 - 3, 5 938 x 2832	2 0.2 - 3, 5 938 x 1062
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	82.90%	90.40%	86.10%	89.80%
Medium Tree	82.90%	87.20%	84.50%	89.80%
Coarse Tree	84.00%	85.60%	78.60%	87.70%
Linear Discriminant	87.70%	96.80%	93.60%	82.90%
Logistic Regression	63.10%	82.90%	50.30%	59.40%
Gaussian Naive Bayes	78.10%	77.50%	78.10%	75.90%
Kernel Naive Bayes	63.60%	75.90%	77.50%	78.60%
Linear SVM	87.70%	89.80%	86.60%	89.30%
Quadratic SVM	92.00%	92.00%	93.00%	91.40%
Cubic SMV	93.00%	94.10%	93.60%	95.70%
Fine Gaussian SVM	80.70%	92.50%	85.60%	93.60%
Medium Gaussian SVM	90.90%	91.40%	88.80%	90.40%
Coarse Gaussian SVM	79.70%	82.90%	84.00%	83.40%
Fine kNN	100.00%	99.50%	97.90%	97.90%
Medium kNN	83.40%	87.20%	87.70%	90.90%
Coarse kNN	69.00%	75.40%	74.90%	75.40%
Cosine kNN	85.60%	88.20%	89.80%	92.00%
Cubic kNN	97.70%	86.60%	86.10%	88.80%
Weighted kNN	94.70%	96.80%	95.70%	96.30%
Boosted Trees	93.60%	96.30%	95.20%	95.20%
Bagged Trees	89.80%	93.60%	89.80%	94.70%
Subspace Discriminant	78.10%	96.80%	88.20%	91.40%
Subspace kNN	96.30%	97.30%	95.20%	96.30%
RUSBoosted Trees	88.80%	92.50%	88.20%	95.70%
MLP	97.87%	97.34%	96.81%	97.87%

Table 8.12: AUC for spectrogram computed only for segmented chunks with Chronux library and frequencies from 4 to 112 Hz.

Frequency range	4 - 112 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4, 2 938 x 2832	2, 0.2 - 4, 2 938 x 1062	1 0.1 - 3, 5 938 x 2832	2 0.2 - 3, 5 938 x 1062
	AUC	AUC	AUC	AUC
Fine Tree	0.85	0.90	0.87	0.89
Medium Tree	0.84	0.89	0.81	0.89
Coarse Tree	0.85	0.84	0.83	0.87
Linear Discriminant	0.87	0.97	0.94	0.82
Logistic Regression	0.64	0.85	0.50	0.61
Gaussian Naive Bayes	0.78	0.78	0.77	0.75
Kernel Naive Bayes	0.72	0.77	0.79	0.79
Linear SVM	0.95	0.96	0.93	0.96
Quadratic SVM	0.98	0.98	0.96	0.97
Cubic SMV	0.99	0.99	0.96	0.99
Fine Gaussian SVM	1.00	0.99	0.97	0.99
Medium Gaussian SVM	0.97	0.98	0.93	0.96
Coarse Gaussian SVM	0.88	0.90	0.86	0.89
Fine kNN	1.00	1.00	0.98	0.98
Medium kNN	0.97	0.97	0.97	0.97
Coarse kNN	0.81	0.85	0.83	0.85
Cosine kNN	0.94	0.97	0.95	0.98
Cubic kNN	0.94	0.96	0.95	0.96
Weighted kNN	1.00	1.00	0.98	0.99
Boosted Trees	0.98	0.99	0.98	0.99
Bagged Trees	0.97	0.99	0.95	0.98
Subspace Discriminant	0.85	1.00	0.95	0.97
Subspace kNN	0.99	1.00	0.98	0.99
RUSBoosted Trees	0.97	0.98	0.97	0.99
MLP	0.98	0.97	0.97	0.98

window parameters of [2, 0.2] seconds and tapers [4, 2] or 97.87% accuracy with MLP for window parameters of [2, 0.2] seconds and tapers [3, 5].

8.2.2 Results of whole Signal Spectrogram Feature Classification. Spectrograms from MATLAB, Chronux, and Sonic Visualizer

This section shows the results of classifying the test set using the features obtained with the spectrograms of MATLAB, Chronux and Sonic Visualizer. In this second approach, the spectrogram of the complete signal was calculated and then it is segmented taking into account the locomotion periods.

MATLAB Spectrogram Features Analogously to the previous subsection, different parameters have been studied to train the models: the type of window, the number of sections and the overlap, as shown in table 7.2.

Tables 8.13 and 8.14 show the accuracy and AUC results, respectively, after training the models and validating them with the test set, using a Hamming window. Analogously, tables 8.15 , 8.16 , 8.17 y 8.18 show the results of accuracy and AUC, for a Tukey and Chebyshev window, respectively.

As can be seen in the tables, for the Hamming window, 99.50% accuracy is obtained for the linear discriminant, quadratic and cubic SVM methods with 4 sections and 0.1 overlap. For the Tukey window, 98.90% accuracy is observed for fine k-NN and RUSBoosted trees with 4 sections and 0.5 overlap. For the Chebyshev window, results of 100% for cubic SVM and 99.50% for linear discriminant and quadratic SVM with 8 sections and 0.5 overlap are obtained.

Chronux Spectrogram Features Analogous to the previous section, table 7.1. shows the parameters that have been used to calculate the spectrogram with the Chronux library.

Table 8.13: Accuracy for spectrogram computed for the whole piezosignal with Matlab library and Hamming window and then segmented.

Window type	Hamming			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	95.70%	92.50%	98.40%	96.80%
Medium Tree	95.20%	87.20%	98.40%	96.80%
Coarse Tree	81.30%	75.90%	84.00%	81.80%
Linear Discriminant	99.50%	94.70%	97.30%	97.90%
Logistic Regression	94.10%	91.40%	95.70%	93.00%
Gaussian Naive Bayes	46.50%	48.10%	48.10%	48.70%
Kernel Naive Bayes	95.20%	88.20%	90.90%	92.00%
Linear SVM	95.20%	88.80%	94.10%	95.70%
Quadratic SVM	99.50%	93.00%	97.30%	97.90%
Cubic SMV	99.50%	94.10%	97.90%	98.40%
Fine Gaussian SVM	95.70%	89.80%	97.30%	93.00%
Medium Gaussian SVM	97.90%	90.40%	98.40%	94.70%
Coarse Gaussian SVM	80.70%	77.00%	84.00%	84.50%
Fine kNN	97.30%	97.30%	95.20%	96.30%
Medium kNN	80.70%	72.20%	81.80%	85.60%
Coarse kNN	66.30%	58.80%	65.20%	62.00%
Cosine kNN	89.80%	81.80%	88.80%	90.90%
Cubic kNN	81.80%	72.70%	84.00%	86.10%
Weighted kNN	97.90%	95.70%	97.30%	97.90%
Boosted Trees	97.30%	92.00%	98.40%	96.30%
Bagged Trees	97.90%	92.50%	97.90%	96.30%
Subspace Discriminant	93.60%	94.10%	93.00%	94.70%
Subspace kNN	95.20%	92.00%	96.80%	97.30%
RUSBoosted Trees	96.30%	89.80%	54.00%	96.80%
MLP	97.87%	96.28%	96.28%	97.87%

Table 8.14: AUC for spectrogram computed for the whole piezosignal with Matlab library and Hamming window and then segmented.

Window type	Hamming			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.96	0.92	0.98	0.97
Medium Tree	0.96	0.87	0.98	0.97
Coarse Tree	0.83	0.80	0.84	0.83
Linear Discriminant	0.99	0.95	0.97	0.98
Logistic Regression	0.95	0.91	0.96	0.93
Gaussian Naive Bayes	0.50	0.52	0.52	0.52
Kernel Naive Bayes	0.95	0.89	0.91	0.92
Linear SVM	1.00	0.96	0.99	1.00
Quadratic SVM	1.00	0.99	1.00	1.00
Cubic SMV	1.00	0.99	1.00	1.00
Fine Gaussian SVM	1.00	0.99	1.00	0.99
Medium Gaussian SVM	1.00	0.98	1.00	1.00
Coarse Gaussian SVM	0.91	0.84	0.94	0.94
Fine kNN	0.97	0.97	0.95	0.96
Medium kNN	0.94	0.88	0.96	0.96
Coarse kNN	0.85	0.85	0.89	0.90
Cosine kNN	0.97	0.93	0.97	0.98
Cubic kNN	0.94	0.92	0.95	0.94
Weighted kNN	1.00	1.00	0.99	1.00
Boosted Trees	0.98	0.99	0.98	0.99
Bagged Trees	1.00	0.99	1.00	1.00
Subspace Discriminant	1.00	0.99	0.97	0.98
Subspace kNN	0.99	0.97	0.99	1.00
RUSBoosted Trees	0.98	0.97	-	0.99
MLP	0.98	0.96	0.96	0.98

Table 8.15: Accuracy for spectrogram computed for the whole piezosignal with Matlab library and Tukey window and then segmented.

Window type	Tukey			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	93.60%	91.40%	96.80%	93.60%
Medium Tree	93.60%	87.20%	96.80%	93.60%
Coarse Tree	82.40%	73.80%	85.00%	82.90%
Linear Discriminant	97.90%	96.80%	98.40%	97.90%
Logistic Regression	93.60%	92.00%	93.60%	89.80%
Gaussian Naive Bayes	48.10%	48.70%	50.30%	48.10%
Kernel Naive Bayes	92.50%	90.40%	91.40%	93.00%
Linear SVM	96.30%	94.10%	96.30%	94.10%
Quadratic SVM	97.30%	96.30%	96.80%	95.70%
Cubic SMV	96.30%	96.80%	97.90%	97.30%
Fine Gaussian SVM	94.10%	92.00%	94.70%	92.50%
Medium Gaussian SVM	96.80%	94.70%	95.70%	95.20%
Coarse Gaussian SVM	85.00%	79.10%	84.50%	80.70%
Fine kNN	97.30%	94.70%	98.90%	96.80%
Medium kNN	87.70%	79.70%	91.40%	85.00%
Coarse kNN	64.70%	65.80%	75.40%	67.90%
Cosine kNN	89.30%	90.40%	92.50%	92.50%
Cubic kNN	86.10%	78.60%	91.40%	82.90%
Weighted kNN	96.30%	94.10%	97.30%	97.90%
Boosted Trees	94.10%	97.90%	54.00%	97.30%
Bagged Trees	94.70%	96.30%	97.90%	96.80%
Subspace Discriminant	96.30%	97.30%	90.90%	92.50%
Subspace kNN	95.20%	97.30%	97.30%	96.30%
RUSBoosted Trees	94.10%	94.10%	98.90%	95.70%
MLP	96.81%	93.62%	96.81%	98.40%

Table 8.16: AUC for spectrogram computed for the whole piezosignal with Matlab library and Tukey window and then segmented.

Window type	Tukey			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.94	0.93	0.97	0.95
Medium Tree	0.92	0.87	0.97	0.94
Coarse Tree	0.85	0.76	0.89	0.85
Linear Discriminant	0.98	0.97	0.99	0.98
Logistic Regression	0.94	0.92	0.94	0.90
Gaussian Naive Bayes	0.52	0.52	0.54	0.52
Kernel Naive Bayes	0.92	0.92	0.92	0.94
Linear SVM	1.00	0.99	0.98	0.99
Quadratic SVM	1.00	1.00	0.99	1.00
Cubic SMV	1.00	1.00	1.00	1.00
Fine Gaussian SVM	1.00	0.99	1.00	0.99
Medium Gaussian SVM	1.00	0.99	1.00	0.99
Coarse Gaussian SVM	0.94	0.89	0.93	0.91
Fine kNN	0.97	0.95	0.99	0.97
Medium kNN	0.96	0.94	0.98	0.98
Coarse kNN	0.89	0.88	0.88	0.89
Cosine kNN	0.97	0.97	0.99	0.99
Cubic kNN	0.96	0.93	0.97	0.98
Weighted kNN	0.99	0.99	0.99	0.99
Boosted Trees	0.95	1.00	-	1.00
Bagged Trees	0.99	0.99	1.00	1.00
Subspace Discriminant	0.99	0.99	0.95	0.96
Subspace kNN	0.99	1.00	1.00	1.00
RUSBoosted Trees	0.95	0.98	1.00	1.00
MLP	0.97	0.94	0.96	0.98

Table 8.17: Accuracy for spectrogram computed for the whole piezosignal with Matlab library and Chebyshev window and then segmented.

Window type	Chebyshev			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	97.30%	93.00%	97.30%	93.00%
Medium Tree	96.80%	90.90%	97.30%	92.50%
Coarse Tree	86.10%	80.20%	84.50%	78.60%
Linear Discriminant	98.90%	94.10%	98.40%	99.50%
Logistic Regression	92.50%	88.80%	94.70%	86.60%
Gaussian Naive Bayes	48.70%	48.70%	48.10%	48.10%
Kernel Naive Bayes	93.00%	85.00%	94.70%	91.40%
Linear SVM	95.70%	88.20%	95.70%	97.30%
Quadratic SVM	98.90%	94.10%	97.30%	99.50%
Cubic SMV	98.90%	95.70%	98.40%	100.00%
Fine Gaussian SVM	91.40%	90.90%	95.20%	89.30%
Medium Gaussian SVM	98.40%	92.50%	96.30%	96.80%
Coarse Gaussian SVM	85.00%	74.90%	81.80%	81.80%
Fine kNN	98.90%	94.10%	96.30%	96.30%
Medium kNN	82.90%	76.50%	85.60%	85.00%
Coarse kNN	69.00%	66.30%	71.10%	65.20%
Cosine kNN	91.40%	87.70%	90.90%	90.40%
Cubic kNN	81.80%	75.40%	84.50%	84.50%
Weighted kNN	98.40%	95.20%	96.80%	97.30%
Boosted Trees	95.20%	93.00%	97.30%	94.70%
Bagged Trees	98.90%	94.10%	98.40%	97.30%
Subspace Discriminant	97.30%	90.90%	93.60%	95.70%
Subspace kNN	97.90%	90.90%	96.80%	93.60%
RUSBoosted Trees	97.30%	93.00%	95.20%	96.30%
MLP	96.28%	96.81%	96.28%	97.87%

Table 8.18: AUC for spectrogram computed for the whole piezosignal with Matlab library and Chebyshev window and then segmented.

Window type	Chebyshev			
number of sections - overlap samples x features	4 - 0.1 938 x 2052	8 - 0.1 938 x 1799	4 - 0.5 938 x 4100	8 - 0.5 938 x 4104
	AUC	AUC	AUC	AUC
Fine Tree	0.98	0.94	0.97	0.93
Medium Tree	0.95	0.91	0.97	0.93
Coarse Tree	0.88	0.80	0.87	0.84
Linear Discriminant	0.99	0.94	0.98	1.00
Logistic Regression	0.94	0.89	0.95	0.87
Gaussian Naive Bayes	0.55	0.52	0.52	0.52
Kernel Naive Bayes	0.95	0.87	0.94	0.95
Linear SVM	1.00	0.96	0.99	1.00
Quadratic SVM	1.00	0.98	1.00	1.00
Cubic SMV	1.00	0.98	1.00	1.00
Fine Gaussian SVM	1.00	0.99	1.00	0.99
Medium Gaussian SVM	1.00	0.98	1.00	1.00
Coarse Gaussian SVM	0.93	0.89	0.90	0.93
Fine kNN	0.99	0.94	0.96	0.96
Medium kNN	0.96	0.90	0.96	0.96
Coarse kNN	0.84	0.88	0.87	0.89
Cosine kNN	0.98	0.94	0.97	0.97
Cubic kNN	0.95	0.91	0.95	0.96
Weighted kNN	1.00	0.98	1.00	1.00
Boosted Trees	0.98	0.99	0.97	1.00
Bagged Trees	1.00	0.99	1.00	1.00
Subspace Discriminant	1.00	0.99	0.96	0.99
Subspace kNN	1.00	0.98	1.00	0.99
RUSBoosted Trees	1.00	0.98	1.00	1.00
MLP	0.96	0.97	0.96	0.98

Table 8.19: Accuracy for spectrogram computed for the whole piezosignal with Chronux library for frequencies from 1.5 to 40 Hz and then segmented.

Frequency range	1.5 - 40 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4 2 938 x 1575	2, 0.5 - 4 2 937 x 635	1, 0.1 - 3 5 938 x 1575	2, 0.5 - 3 5 937 x 635
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	85.60%	94.10%	85.60%	96.30%
Medium Tree	84.00%	90.90%	82.90%	94.10%
Coarse Tree	77.50%	80.70%	77.50%	83.40%
Linear Discriminant	67.90%	94.70%	64.70%	96.30%
Logistic Regression	54.00%	93.60%	46.50%	94.70%
Gaussian Naive Bayes	65.80%	72.70%	67.90%	69.50%
Kernel Naive Bayes	68.40%	83.40%	73.30%	77.50%
Linear SVM	75.90%	86.10%	84.00%	85.00%
Quadratic SVM	88.20%	94.10%	91.40%	95.20%
Cubic SMV	88.20%	94.70%	95.70%	95.70%
Fine Gaussian SVM	80.20%	94.10%	86.10%	94.70%
Medium Gaussian SVM	84.00%	90.90%	88.20%	90.40%
Coarse Gaussian SVM	73.30%	78.60%	74.30%	72.70%
Fine kNN	97.90%	96.30%	97.30%	98.40%
Medium kNN	80.20%	87.20%	81.30%	91.40%
Coarse kNN	69.50%	69.50%	71.10%	68.40%
Cosine kNN	76.50%	87.20%	81.90%	92.00%
Cubic kNN	81.80%	87.70%	80.70%	87.20%
Weighted kNN	93.60%	96.30%	91.40%	96.80%
Boosted Trees	94.70%	94.10%	88.20%	95.70%
Bagged Trees	92.50%	95.20%	87.70%	96.80%
Subspace Discriminant	60.40%	95.70%	66.80%	95.70%
Subspace kNN	98.40%	98.90%	96.30%	97.30%
RUSBoosted Trees	90.90%	91.40%	86.60%	96.30%
MLP	94.15%	97.87%	94.68%	97.87%

Table 8.20: AUC for spectrogram computed for the whole piezosignal with Chronux library for frequencies from 1.5 to 40 Hz and then segmented.

Frequency range	1.5 - 40 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4 2 938 x 1575	2, 0.5 - 4 2 937 x 635	1, 0.1 - 3 5 938 x 1575	2, 0.5 - 3 5 937 x 635
	AUC	AUC	AUC	AUC
Fine Tree	0.87	0.95	0.86	0.97
Medium Tree	0.85	0.93	0.82	0.95
Coarse Tree	0.81	0.84	0.83	0.85
Linear Discriminant	0.67	0.95	0.64	0.96
Logistic Regression	0.54	0.94	0.46	0.95
Gaussian Naive Bayes	0.65	0.72	0.68	0.69
Kernel Naive Bayes	0.72	0.83	0.76	0.77
Linear SVM	0.79	0.94	0.91	0.93
Quadratic SVM	0.91	0.98	0.97	0.99
Cubic SMV	0.89	0.98	0.98	0.99
Fine Gaussian SVM	0.98	1.00	0.99	1.00
Medium Gaussian SVM	0.90	0.97	0.96	0.97
Coarse Gaussian SVM	0.78	0.85	0.86	0.85
Fine kNN	0.98	0.96	0.97	0.98
Medium kNN	0.91	0.95	0.91	0.96
Coarse kNN	0.77	0.82	0.80	0.78
Cosine kNN	0.89	0.95	0.91	0.95
Cubic kNN	0.92	0.94	0.91	0.94
Weighted kNN	0.98	0.99	0.97	0.99
Boosted Trees	0.97	0.99	0.95	1.00
Bagged Trees	0.98	0.99	0.96	0.99
Subspace Discriminant	0.64	1.00	0.74	1.00
Subspace kNN	1.00	0.99	0.98	0.99
RUSBoosted Trees	0.97	0.96	0.94	0.99
MLP	0.94	0.98	0.95	0.98

Table 8.21: Accuracy for spectrogram computed for the whole piezosignal with Chronux library for frequencies from 4 to 112 Hz and then segmented.

Frequency range	4 - 112 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4 2 938 x 4425	2, 0.5 - 4 2 937 x 1770	1, 0.1 - 3 5 938 x 4425	2, 0.5 - 3 5 937 x 1770
	Accuracy	Accuracy	Accuracy	Accuracy
Fine Tree	78.60%	95.70%	89.30%	96.30%
Medium Tree	78.60%	95.70%	86.10%	96.30%
Coarse Tree	73.30%	82.40%	79.70%	89.30%
Linear Discriminant	93.00%	96.30%	91.40%	96.30%
Logistic Regression	55.10%	96.30%	54.50%	95.70%
Gaussian Naive Bayes	79.10%	83.40%	74.90%	77.50%
Kernel Naive Bayes	67.90%	89.30%	77.50%	87.20%
Linear SVM	90.90%	92.00%	85.00%	91.40%
Quadratic SVM	93.60%	93.00%	92.50%	95.70%
Cubic SMV	94.10%	95.70%	93.00%	96.30%
Fine Gaussian SVM	84.00%	95.20%	86.60%	93.60%
Medium Gaussian SVM	92.00%	92.00%	88.20%	93.00%
Coarse Gaussian SVM	82.90%	86.10%	80.70%	86.60%
Fine kNN	97.30%	99.50%	96.80%	97.90%
Medium kNN	85.00%	89.80%	83.40%	92.00%
Coarse kNN	70.60%	71.70%	67.90%	78.10%
Cosine kNN	88.20%	90.90%	86.60%	89.80%
Cubic kNN	85.00%	89.80%	84.50%	88.80%
Weighted kNN	92.50%	99.50%	95.70%	98.40%
Boosted Trees	89.80%	98.40%	93.00%	54.00%
Bagged Trees	89.80%	98.40%	92.00%	97.30%
Subspace Discriminant	84.50%	92.50%	88.80%	95.70%
Subspace kNN	97.30%	96.30%	97.30%	98.40%
RUSBoosted Trees	87.70%	98.90%	93.00%	96.80%
MLP	100.00%	96.28%	99.47%	98.40%

Table 8.22: AUC for spectrogram computed for the whole piezosignal with Chronux library for frequencies from 4 to 112 Hz and then segmented.

Frequency range	4 - 112 Hz			
windowsize, windowstep (s) - tapers samples x features	1, 0.1 - 4 2 938 x 4425	2, 0.5 - 4 2 937 x 1770	1, 0.1 - 3 5 938 x 4425	2, 0.5 - 3 5 937 x 1770
	AUC	AUC	AUC	AUC
Fine Tree	0.79	0.96	0.90	0.96
Medium Tree	0.75	0.96	0.88	0.96
Coarse Tree	0.80	0.84	0.82	0.91
Linear Discriminant	0.93	0.96	0.91	0.96
Logistic Regression	0.56	0.97	0.55	0.96
Gaussian Naive Bayes	0.79	0.84	0.74	0.79
Kernel Naive Bayes	0.79	0.90	0.80	0.87
Linear SVM	0.97	0.98	0.93	0.99
Quadratic SVM	0.98	0.99	0.98	0.99
Cubic SMV	0.99	0.99	0.98	0.99
Fine Gaussian SVM	0.98	1.00	0.97	1.00
Medium Gaussian SVM	0.97	0.99	0.95	0.99
Coarse Gaussian SVM	0.91	0.92	0.86	0.92
Fine kNN	0.97	0.99	0.97	0.98
Medium kNN	0.95	0.98	0.95	0.99
Coarse kNN	0.82	0.84	0.79	0.87
Cosine kNN	0.95	0.98	0.95	0.98
Cubic kNN	0.94	0.98	0.95	0.98
Weighted kNN	0.98	1.00	0.99	1.00
Boosted Trees	0.95	0.99	0.98	-
Bagged Trees	0.96	1.00	0.97	0.99
Subspace Discriminant	0.93	0.99	0.95	0.98
Subspace kNN	0.98	1.00	0.98	0.99
RUSBoosted Trees	0.95	0.99	0.98	0.98
MLP	1.00	0.96	0.99	0.98

Tables 8.19 and 8.20 show the accuracy and AUC, respectively for the frequency range 1.5-40 Hz and for different number of tapers and window step and size. The best results are 98.90% accuracy with the subspace k-NN method and 97.87% for MLP for window parameters of [2, 0.5] seconds and tapers [4, 2] or 98.40% with the subspace k-NN method and 97.90% with fine k-NN for a window size and step of [1, 0.1] seconds and tapers [4, 2].

Tables 8.21 and 8.22 show the accuracy and AUC, respectively for the frequency range 4-112Hz and for different number of tapers and window step-size. The results indicate 100% accuracy for MLP with window parameters of [1, 0.1] seconds and tapers [4, 2] and 99.50% accuracy with fine k-NN and weighted k-NN for window parameters of [2, 0.5] seconds and tapers [4, 2].

Sonic Visualizer Spectrogram Features To calculate the spectrogram with Sonic Visualizer, the parameters have been calibrated manually, obtaining those shown in table 7.3.

Table 8.23 shows the classification accuracy and AUC results, where the best results are obtained using the cubic SVM, subspace k-NN and MLP methods. Of the three results, the cubic SVM method also returns an AUC value of 1.

8.2.3 Results of whole Signal Spectrogram Image Classification by Transfer Learning. Spectrograms from MATLAB, Chronux, and Sonic Visualizer

To obtain the spectrogram images, the spectrogram is plotted once its numerical values have been obtained and segmented for each locomotion period. These numerical data correspond to those of the section 8.2.2. The images obtained will be used to train the described convolutional networks already pre-trained with transfer learning.

Table 8.23: Accuracy and AUC for spectrogram computed for the whole piezosignal with Sonic Visualizer and then segmented.

windowsize, overlap samples x features	256, 93.75% 938 x 4095	
	Accuracy	AUC
Fine Tree	83.40%	0.85
Medium Tree	84.00%	0.89
Coarse Tree	79.70%	0.85
Linear Discriminant	74.90%	0.75
Logistic Regression	57.20%	0.59
Gaussian Naive Bayes	63.60%	0.63
Kernel Naive Bayes	72.20%	0.71
Linear SVM	74.30%	0.85
Quadratic SVM	95.20%	0.99
Cubic SMV	95.70%	1.00
Fine Gaussian SVM	58.80%	0.90
Medium Gaussian SVM	94.70%	0.99
Coarse Gaussian SVM	71.10%	0.67
Fine kNN	94.10%	0.94
Medium kNN	90.90%	0.97
Coarse kNN	79.10%	0.83
Cosine kNN	87.70%	0.95
Cubic kNN	89.30%	0.95
Weighted kNN	91.40%	0.98
Boosted Trees	88.20%	0.96
Bagged Trees	95.70%	0.99
Subspace Discriminant	71.70%	0.81
Subspace kNN	96.80%	0.99
RUSBoosted Trees	87.70%	0.95
MLP	96.28%	0.96

Table 8.24: Accuracy for spectrogram images computed for the whole piezosignal with Matlab library and then segmented.

Window type	Chebyshev			
Number of sections - overlap	4 - 0.1	4 - 0.5	8 - 0.1	8 - 0.5
	Accuracy	Accuracy	Accuracy	Accuracy
AlexNet	94.65%	93.05%	96.79%	98.93%
ResNet50	96.79%	93.05%	98.40%	97.86%
GoogleNet	96.79%	95.19%	98.40%	99.47%

Window type	Hamming			
Number of sections - overlap	4 - 0.1	4 - 0.5	8 - 0.1	8 - 0.5
	Accuracy	Accuracy	Accuracy	Accuracy
AlexNet	95.19%	97.86%	94.65%	96.79%
ResNet50	97.33%	97.33%	99.47%	99.47%
GoogleNet	96.79%	95.72%	97.33%	97.86%

Window type	Tukey			
Number of sections - overlap	4 - 0.1	4 - 0.5	8 - 0.1	8 - 0.5
	Accuracy	Accuracy	Accuracy	Accuracy
AlexNet	99.47%	99.47%	97.86%	97.86%
ResNet50	99.47%	97.86%	97.86%	98.40%
GoogleNet	97.33%	97.86%	97.33%	97.86%

Table 8.25: Accuracy for spectrogram images computed for the whole piezosignal with Chronux library and then segmented.

Frequency range	1.5 - 40 Hz			
Windowsize, windowstep (s) - tapers	1, 0.1 - 3, 5	1, 0.1 - 4, 2	2, 0.5 - 3, 5	2, 0.5 - 4, 2
	Accuracy	Accuracy	Accuracy	Accuracy
AlexNet	99.47%	97.86%	93.58%	94.65%
ResNet50	98.40%	98.93%	96.79%	97.33%
GoogleNet	98.40%	96.79%	97.33%	94.12%

Frequency range	4 - 112 Hz			
Windowsize, windowstep (s) - tapers	1, 0.1 - 3, 5	1, 0.1 - 4, 2	2, 0.5 - 3, 5	2, 0.5 - 4, 2
	Accuracy	Accuracy	Accuracy	Accuracy
AlexNet	98.40%	100.00%	98.93%	96.79%
ResNet50	99.47%	99.47%	99.47%	97.86%
GoogleNet	99.47%	99.47%	97.86%	97.33%

Table 8.26: Accuracy for spectrogram images computed for the whole piezosignal with Sonic Visualizer library and then segmented.

Window size, overlap	256, 93.75%
	Accuracy
AlexNet	100.00%
ResNet50	96.79%
GoogleNet	96.26%

Tables 8.24, 8.25 and 8.26 show the accuracy results after using the AlexNet, ResNet50 and GoogLeNet networks for the images calculated using MATLAB spectrogram, Chronux library and Sonic Visualizer.

The tables indicate that all results return values above 93% accuracy. In some cases it reaches 100% as in the case of the AlexNet network for the Chronux and Sonic Visualizer images. Values above 99% are also obtained for images calculated with Chronux, in various configurations, and MATLAB, using the Chebyshev window.

8.3 Discussion

In this Chapter we report a classification approach to discriminate 2 different strains. For that, we have chosen to analyse locomotion periods, as they can be easily extracted by applying Computer Vision techniques.

Once the locomotion periods have been obtained, this work establishes two different approaches. In the first approach, the spectrogram calculation is performed on the signal chunks corresponding to the locomotion periods, so that the result is not influenced by the behavior before and after the periods of interest. In the second approach, the spectrogram of the whole signal is calculated and the locomotion periods are extracted from this spectrogram, so that the preceding and following behavior will influence the results of the segmented periods. In addition, three different methods have been used to compute the spectrogram by varying some parameters in order to perform comparative studies: MATLAB, Chronux and Sonic Visualizer.

Comparing the results obtained after classification, the best accuracy values are mostly obtained with the second approach, e.g. using the MATLAB spectrogram function with a Hamming window together with boosted trees and bagged trees. However, Gaussian Naive Bayes performs better with the first approach. Similarly, for a Tukey window, the first approach is better with cubic SVM while the second approach is better for fine trees. As for the Chebyshev window, the accuracy results obtained are better for the second approach with fine trees, logistic regression, weighted k-NN, bagged trees and boosted trees, while the first approach gives better results with Gaussian Naive Bayes. Regarding the Chronux library, for the frequency range of 1.5 - 40 Hz, better results are obtained for the second approach with linear discriminant, kernel naive Bayes, all SVM, medium k-NN, cosine k-NN and weighted k-NN, and for the frequency range of 4 - 112 Hz, the first approach is better with fine k-NN and the second approach is better with kernel naive Bayes, fine Gaussian SVM and bagged trees.

On the other hand, better results are obtained by computing spectrogram of the whole signal and the *spectrogram* function of MATLAB with a Chebyshev window.

Regarding the classification results using images applying transfer learning, the minimum accuracy obtained is 93.05% and the maximum is 100% with AlexNet using the images generated with the spectrograms of Sonic Visualizer or the Chronux library.

This Chapter also presents the training and evaluation times of the algorithms used for classification, the fastest methods are the machine learning algorithms implemented in MATLAB and MLP implemented in Python. The computation times for the pre-trained models range from 35 minutes with AlexNet to 4 hours with ResNet50. These execution times are very small compared to the good results they offer in terms of accuracy and AUC.

To sum up, it is concluded that it is possible to perform strain classification for mice, using a recording system composed of the Phenotypix piezoelectric platform and a top camera. This system allows experiments to be carried out in a non-invasive way without inhibiting the behavior of the animals and also offers high sensitivity, making it possible to establish differences between different strains and classify them with high accuracy and AUC values.

Chapter 9

Conclusion

This Chapter summarizes the conclusions after this Thesis work. Section 9.1 provides the findings and contributions derived from this Thesis. Subsections 9.1.1 and 9.1.2 comment separately the findings in SLAM applications with LiDAR and the results obtained in Computational Ethology, respectively. Section 9.2 provides the future work derived from the contribution of this Thesis.

9.1 Summary of Findings and Contributions

In this Section we summarize the findings and contributions generated in this Thesis for the two fields of knowledge.

9.1.1 SLAM Applications with LiDAR

Light detection and ranging (LiDAR) sensors have been used for scanning, reconstruction of environments. The fusion of LiDAR with GPS allows for large scale navigation of autonomous systems, where Simultaneous localization and mapping (SLAM) is a highly relevant. Accurate sensing provided by range sensors such as the LiDARs improve the speed and accuracy of SLAM, which can become an integral part of the control of innovative autonomous vehicles.

Recently, LiDAR based SLAM is becoming affordable by new sensors such as the M8 Quanergy LiDAR. However, these sensors offer less quality data and lower resolution that hinders the performance of registration methods. The Deep Learning based approaches seem to be sensitive to these data flaws. Specifically, we have experimented with a state-of-the-art Deep Learning based approach that failed to produce meaningful results after several attempts to carry out transfer learning over a dataset collected indoors with one such affordable sensors. Consequently, traditional methods appear to be more likely to output better results than artificial intelligent based methods for these low-cost sensors. In this regard, a comparison of three traditional registration methods applied to the path estimation followed by the LiDAR sensor was provided; namely, the Iterative Closest Points (ICP), Coherent Point Drift (CPD), and Normal Distributions Transform (NDT) registration methods. For this benchmark experiment, a dataset was collected with M8 Quanergy LiDAR in the 3 floor of the Computer Science School of the UPV/EHU in San Sebastian. The results obtained from this comparison showed both ICP and CPD methods had problems in turning parts of the path, losing the track of the sensor while point cloud registering. On the other hand, NDT method was able to register the whole dataset with high robustness comparing to the others methods. Nevertheless, ICP was capable of constructing surfaces with more point cloud density than NDT did. As a result, a hybrid point cloud registration method is proposed to take advantage of the high accuracy provided by the classic ICP algorithm, and the robustness of the NDT registration method. To test this second analysis, a new dataset was recorded in the same location than the first one and the results showed that the Hybrid Registration Algorithm (HRA) returns a better reconstruction compared to the ICP and NDT methods separately.

The contributions of this Thesis in SLAM application with LiDAR are the following:

- Installation and configuration of M8 Quanergy LiDAR sensor and its drivers for experimentation.
- Recording several datasets with different routes to implement SLAM algorithms and upload the recordings in Zenodo.
- Creation the scripts with the algorithms in MATLAB to implement SLAM by using the datasets recorded previously.
- Comparison of the results obtained from the traditional methods and extract conclusions to design a better algorithm.
- Creation of he Hybrid Registration Algorithm (HRA), based on the previous results, with the joint of the ICP and NDT methods.
- Obtaining a better reconstruction of the surface with the HRA proposed in this Thesis.

9.1.2 Computational Ethology

Computational Ethology is a discipline that studies the behavior to understand the causes and development of animal behavior, as well as to understand how it is performed. By analysing the behavior is possible to extract characteristics from them that allow us to describe them quantitatively. From a pharmacological point of view, ethological experimentation give another possibility to test new medicines, as it is capable of quantifying behaviors and comparing them between different subjects.

This second contribution is a research question based on the hypothesis of whether it is possible to differentiate phenotypes using a recording system composed of the Phenotypix piezoelectric platform and a top camera by means of

AI. To answer this research question, we have used data from animal experimentation in a non-invasive recording system composed of a piezoelectric platform and a to video camera. We have recorded two types of animals such as Wild-type and Fmr1-KO subjects to try to discriminate with AI techniques. With the data collected, we have developed a semi-automated algorithm to preprocess the videos in order to obtain the centroid of the animal and compute its velocity to filter locomotion periods. When the piezoelectric signal is synchronized with the animal velocity data, we can apply two different approaches to obtain the features and the images from the spectrogram with MATLAB, Chronux library and Sonic Visualizer. The first approach segments the locomotion periods in the piezoelectric signal to compute the spectrogram only in these chunks. The second approach obtains the spectrogram for the whole piezoelectric signal to filter in it the locomotion periods. In the first approach the chunks are not influenced by the previous and following behaviors. On the other hand, in the second approach, the chunks of signal are influenced by the surrounding behaviors.

The classifiers used in this contribution are the following:

- For feature classification:
  - Machine Learning algorithms: SVM, decision Trees, k-NN, Naive Bayes, Linear Discriminant, Logistic regression and its variants.
  - Multi-Layer Perceptron
- For transfer learning based image classification:
  - AlexNet
  - GoogLeNet
  - ResNet50

The results obtained from the classification algorithms show high accuracy with the spectrogram computed with MATLAB, the Chronux library and Sonic Visualizer. The metrics show good results in strain classification in both approaches but there are light differences depending on the algorithm and the parameters for spectrogram computation. For instance, regarding the Chronux library, for the frequency range of 1.5 - 40 Hz, better results are obtained for the second approach with linear discriminant, kernel naive Bayes, all SVM, medium k-NN, cosine k-NN and weighted k-NN, and for the frequency range of 4 - 112 Hz, the first approach is better with fine k-NN and the second approach is better with kernel naive Bayes, fine Gaussian SVM and bagged trees.

Image classification results obtained a minimum of 93.05% and the maximum is 100% with AlexNet and the spectrograms computed with Sonic Visualizer and the Chronux library.

To conclude, we are capable of answering the research question proposed: it is possible to discriminate animal models with data from a piezoelectric platform and a video camera and AI based classifiers.

The contributions of this Thesis in Computational Ethology are the following:

- Development of a pipeline to process video recordings from animal experimentation.
- Development of an algorithm to extract features from a recording system composed of a piezoelectric platform and a video camera.
- Application of AI techniques for animal model classification based on features.
- Application of transfer learning approaches for animal model classification based on images.
- Comparison of the results obtained from different AI-based approaches.
- Answer the research question proposed in this Thesis.

9.2 Future Work

As future work, we would like to continue with results obtained in Computational Ethology. In fact, members of the Computational Intelligence Group are currently collecting data from a walking platform and an electroencephalogram (EEG) to identify gait anomalies as a part of an experimental study on healthy ageing in the elderly. This way, the idea is to take advantage of the knowledge acquired in this Thesis to process the data and try to predict a syndrome that affects to elderly, namely fragility, in the early stages to mitigate its effects, improving the quality of life of the society.

Bibliography

[1] Reyhaneh Abbasi, Peter Balazs, Maria Adelaide Marconi, Doris Nicolakis, Sarah M Zala, and Dustin J Penn. Capturing the songs of mice with an improved detection and classification method for ultrasonic vocalizations (bootsnap). *PLoS Comput Biol*, 18(5):e1010049, May 2022.

[2] Marina Aguilar-Moreno and Manuel Graña. A comparison of registration methods for slam with the m8 quanergy lidar. In E. Corchado et al., editor, *SOCO 2020*, 2020.

[3] O.S. Ahmed, S.E. Franklin, M.A. Wulder, and J.C. White. Characterizing stand-level forest canopy cover and height using landsat time series, samples of airborne lidar, and the random forest algorithm. *ISPRS Journal of Photogrammetry and Remote Sensing*, 101:89–101, 2015.

[4] V. Ajuwon, B.F. Cruz, P. Carriço, A. Kacelnik, T. Monteiro, and Champalimaud Research Scientific Hardware Platform. Gofish: A low-cost, open-source platform for closed-loop behavioural experiments on fish. *Behavior Research Methods*, 2023.

[5] Korleki Akiti, Iku Tsutsui-Kimura, Yudi Xie, Alexander Mathis, Jeffrey E Markowitz, Rockwell Anyoha, Sandeep Robert Datta, Mackenzie Weygandt Mathis, Naoshige Uchida, and Mitsuko Watabe-Uchida. Striatal

dopamine explains novelty-induced behavioral dynamics and individual variability in threat prediction. *Neuron*, 110(22):3789–3804, Nov 2022.

[6] N. Aldoumani, T. Meydan, C. M. Dillingham, and J. T. Erichsen. Enhanced tracking system based on micro inertial measurements unit to measure sensorimotor responses in pigeons. *IEEE Sensors Journal*, 16(24):8847–8853, Dec 2016.

[7] Marcos Alonso, Alberto Izaguirre, Imanol Andonegui, and Manuel Graña. Optical dual laser based sensor denoising for onlinemetal sheet flatness measurement using hermite interpolation. *Sensors*, 20(18), 2020.

[8] D.J. Anderson and P. Perona. Toward a science of computational ethology. *Neuron*, 84(1):18–31, Oct 2014.

[9] Saba Arshad and Gon-Woo Kim. Role of deep learning in loop closure detection for visual and lidar slam: A survey. *Sensors*, 21(4), 2021.

[10] Simon Arvin, Rune Nguyen Rasmussen, and Keisuke Yonehara. Eyeloop: An open-source system for high-speed, closed-loop eye-tracking. *Front Cell Neurosci*, 15:779628, 2021.

[11] T. Bailey and H. Durrant-Whyte. Simultaneous localization and mapping (slam): part ii. *IEEE Robotics Automation Magazine*, 13(3):108–117, Sep. 2006.

[12] Praneet C Bala, Benjamin R Eisenreich, Seng Bum Michael Yoo, Benjamin Y Hayden, Hyun Soo Park, and Jan Zimmermann. Automated markerless pose estimation in freely moving macaques with openmonkeystudio. *Nat Commun*, 11(1):4560, Sep 2020.

[13] C. Benedek, B. Gálai, B. Nagy, and Z. Jankó. Lidar-based gait analysis and activity recognition in a 4d surveillance system. *IEEE Transactions on Circuits and Systems for Video Technology*, 28(1):101–113, Jan 2018.

[14] G.J. Berman, D.M. Choi, W. Bialek, and J.W. Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of the Royal Society Interface*, 11(99), 2014.

[15] P. J. Besl and N. D. McKay. A method for registration of 3-d shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):239–256, Feb 1992.

[16] Peter Biber and Wolfgang Straßer. The normal distributions transform: A new approach to laser scan matching. volume 3, pages 2743 – 2748 vol.3, 11 2003.

[17] J-L. Blanco-Claraco, F.-A. Moreno-Dueñas, and J. González-Jiménez. The malaga urban dataset: High-rate stereo and lidar in a realistic urban scenario. *International Journal of Robotics Research*, 33(2):207–214, 2014.

[18] J.P. Bohnslav, N.K. Wimalasena, K.J. Clausing, Yu.Y. Dai, D.A. Yarmolinsky, T. Cruz, A.D. Kashlan, M.E. Chiappe, L.L. Orefice, C.J. Woolf, and C.D. Harvey. Deepethogram, a machine learning pipeline for supervised behavior classification from raw pixels. *eLife*, 10, 2021.

[19] Alexandra Bova, Krista Kernodle, Kaitlyn Mulligan, and Daniel Leventhal. Automated rat single-pellet reaching with 3-dimensional reconstruction of paw and digit trajectories. *J Vis Exp*, 2019(149), Jul 2019.

[20] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[21] T. C. Bybee and S. E. Budge. Method for 3-d scene reconstruction using fused lidar and imagery from a texel camera. *IEEE Transactions on Geoscience and Remote Sensing*, 57(11):8879–8889, Nov 2019.

[22] C. Cadena, L. Carlone, H. Carrillo, Y. Latif, D. Scaramuzza, J. Neira, I. Reid, and J. J. Leonard. Past, present, and future of simultaneous localization and mapping: Toward the robust-perception age. *IEEE Transactions on Robotics*, 32(6):1309–1332, Dec 2016.

[23] I. Caminal, J. R. Casas, and S. Royo. Slam-based 3d outdoor reconstructions from lidar data. In *2018 International Conference on 3D Immersion (IC3D)*, pages 1–8, Dec 2018.

[24] F. Cao, F. Yan, S. Wang, Y. Zhuang, and W. Wang. Season-invariant and viewpoint-tolerant lidar place recognition in gps-denied environments. *IEEE Transactions on Industrial Electronics*, 68(1):563–574, Jan 2021.

[25] Jun Cao, Bi Zeng, Jianqi Liu, Zhenting Zhao, and Yongfeng Su. A novel relocation method for simultaneous localization and mapping based on deep learning algorithm. *Computers & Electrical Engineering*, 63:79–90, 2017.

[26] N. Carlevaris-Bianco, A.K. Ushani, and R.M. Eustice. University of michigan north campus long-term vision and lidar dataset. *International Journal of Robotics Research*, 35(9):1023–1035, 2016. cited By 103.

[27] M.I. Carreño-Muñoz, M.C. Medrano, A. Ferreira Gomes Da Silva, C. Gestreau, C. Menuet, T. Leinekugel, M. Bompart, F. Martins, E. Subashi, F. Aby, A. Frick, M. Landry, M. Grana, and X. Leinekugel. Detecting fine and elaborate movements with piezo sensors provides non-invasive access to overlooked behavioral components. *Neuropsychopharmacology*, 47(4):933–943, 2022.

[28] J. Castagno and E. Atkins. Roof shape classification from lidar and satellite image data fusion using supervised learning. *Sensors (Switzerland)*, 18(11), 2018.

[29] Shayna-Lee Chaput, Warren W. Burggren, Peter L. Hurd, and Trevor J. Hamilton. Zebrafish (danio rerio) shoaling in light and dark conditions involves a complex interplay between vision and lateral line. *Behavioural brain research*, 439:114228, Feb 2023.

[30] Chun-Peng J. Chen, Gota Morota, Kiho Lee, Zhiwu Zhang, and Hao Cheng. Vtag: a semi-supervised pipeline for tracking pig activity with a single top-view camera. *Journal of animal science*, 100, Jun 2022.

[31] Guipeng Chen, Cong Li, Yang Guo, Hang Shu, Zhen Cao, and Beibei Xu. Recognition of cattle's feeding behaviors using noseband pressure sensor with machine learning. *Frontiers in veterinary science*, 9:822621, 2022.

[32] L. Chen and C. Peng. A robust 2d-slam technology with environmental variation adaptability. *IEEE Sensors Journal*, 19(23):11475–11491, 2019.

[33] S. W. Chen, G. V. Nardari, E. S. Lee, C. Qu, X. Liu, R. A. F. Romero, and V. Kumar. Sloam: Semantic lidar odometry and mapping for forest inventory. *IEEE Robotics and Automation Letters*, 5(2):612–619, 2020.

[34] H.-T.L. Chiang, A. Faust, M. Fiser, and A. Francis. Learning navigation behaviors end-to-end with autorl. *IEEE Robotics and Automation Letters*, 4(2):2007–2014, April 2019.

[35] K. Chiang, G. Tsai, Y. Li, and N. El-Sheimy. Development of lidar-based uav system for environment reconstruction. *IEEE Geoscience and Remote Sensing Letters*, 14(10):1790–1794, Oct 2017.

[36] Hyunhak Cho, Eun Kyeong Kim, and Sungshin Kim. Indoor slam application using geometric and icp matching methods based on line features. *Robotics and Autonomous Systems*, 100:206–224, 2018.

[37] S. Choi, J. Hachisuka, M.A. Brett, A.R. Magee, Y. Omori, N.-U.-A. Iqbal, D. Zhang, M.M. DeLisle, R.L. Wolfson, L. Bai, C. Santiago, S. Gong, M. Goulding, N. Heintz, H.R. Koerber, S.E. Ross, and D.D. Ginty. Parallel ascending spinal pathways for affective touch and pain. *Nature*, 587(7833):258–263, 2020.

[38] Christopher Choy, Wei Dong, and Vladlen Koltun. Deep global registration, 2020.

[39] Erik K H Clemensson, Morteza Abbaszadeh, Silvia Fanni, Elena Espa, and M Angela Cenci. Tracking rats in operant conditioning chambers using a versatile homemade video camera and deeplabcut. *J Vis Exp*, (160), Jun 2020.

[40] K.R. Coffey, R.G. Marx, and J.F. Neumaier. Deepsqueak: a deep learning-based system for detection and analysis of ultrasonic vocalizations. *Neuropsychopharmacology*, 44(5):859–868, 2019.

[41] T.M. Cover and P.E. Hart. Nearest neighbor pattern classification. *IEEE Transactions on Information Theory*, 13(1):21–27, 1967.

[42] Ana G. Cristancho, Natalia Tulina, Amy G. Brown, Lauren Anton, Guillermo Barila, and Michal A. Elovitz. Intrauterine inflammation leads to select sex- and age-specific behavior and molecular differences in mice. *International journal of molecular sciences*, 24, Dec 2022.

[43] Xiangwei Dang, Zheng Rong, and Xingdong Liang. Sensor fusion-based approach to eliminating moving objects for slam in dynamic environments. *Sensors*, 21(1), 2021.

[44] S.R. Datta, D.J. Anderson, K. Branson, P. Perona, and A. Leifer. Computational neuroethology: A call to action. *Neuron*, 104(1):11–24, Oct 2019.

[45] T.F. De Almeida, B.G. Spinelli, R. Hypolito Lima, M.C. Gonzalez, and A.C. Rodrigues. Pyrat: An open-source python library for animal behavior analysis. *Frontiers in Neuroscience*, 16, 2022.

[46] Daniel de Almeida Papa, Danilo Roberti Alves de Almeida, Carlos Alberto Silva, Evandro Orfanó Figueiredo, Scott C. Stark, Ruben Valbuena, Luiz Carlos Estraviz Rodriguez, and Marcus Vinício Neves d' Oliveira. Evaluating tropical forest classification and field sampling stratification from lidar to reduce effort and enable landscape monitoring. *Forest Ecology and Management*, 457:117634, 2020.

[47] Lucas de Paula Veronese, Claudine Badue, Fernando Auat Cheein, Jose Guivant, and Alberto Ferreira De Souza. A single sensor system for mapping in gnss-denied environments. *Cognitive Systems Research*, 56:246–261, 2019.

[48] Y. Dehbi, A. Henn, G. Gröger, V. Stroh, and L. Plümer. Robust and fast reconstruction of complex roofs with active sampling from 3d point clouds. *Transactions in GIS*, 25(1):112–133, 2021.

[49] Y. Deng, Y. Shan, Z. Gong, and L. Chen. Large-scale navigation method for autonomous mobile robot based on fusion of gps and lidar slam. In *2018 Chinese Automation Congress (CAC)*, pages 3145–3148, Nov 2018.

[50] J.C.F. Diaz, W.E. Carter, R.L. Shrestha, and C.L. Glennie. *Lidar remote sensing*, volume 2. 2013.

[51] L. Díaz-Vilariño, H. González-Jorge, J. Martínez-Sánchez, and H. Lorenzo. Automatic lidar-based lighting inventory in buildings. *Measurement*, 73:544–550, 2015.

[52] J. E. Doornweerd, G. Kootstra, R. F. Veerkamp, B. de Klerk, I. Fodor, M. van der Sluis, A. C. Bouwman, and E. D. Ellen. Passive radio frequency identification and video tracking for the determination of location and movement of broilers. *Poultry science*, 102:102412, Dec 2022.

[53] John F Drazan, William T Phillips, Nidhi Seethapathi, Todd J Hullfish, and Josh R Baxter. Moving outside the lab: Markerless motion capture accurately quantifies sagittal plane kinematics during the vertical jump. *J Biomech*, 125:110547, Aug 2021.

[54] H. Durrant-Whyte and T. Bailey. Simultaneous localization and mapping: part i. *IEEE Robotics Automation Magazine*, 13(2):99–110, June 2006.

[55] N.M. Enwright, L. Wang, H. Wang, M.J. Osland, L.C. Feher, S.M. Borchert, and R.H. Day. Modeling barrier island habitats using landscape position information. *Remote Sensing*, 11(8), 2019.

[56] M. Fallon, H. Johannsson, M. Kaess, and J.J. Leonard. The mit stata center dataset. *International Journal of Robotics Research*, 32(14):1695–1699, 2013. cited By 32.

[57] C. Fang, T. Zhang, H. Zheng, J. Huang, and K. Cuan. Pose estimation and behavior classification of broiler chickens based on deep neural networks. *Computers and Electronics in Agriculture*, 180, 2021.

[58] Jiangfan Feng and Xinxin Xiao. Multiobject tracking of wildlife in videos using few-shot learning. *Animals : an open access journal from MDPI*, 12, May 2022.

[59] Duarte Fernandes, António Silva, Rafael Névoa, Cláudia Simões, Dibet Gonzalez, Miguel Guevara, Paulo Novais, João Monteiro, and Pedro Melo-Pinto. Point-cloud based 3d object detection and classification methods for self-driving applications: A survey and taxonomy. *Information Fusion*, 68:161–191, 2021.

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

[61] Dieter Fox, Wolfram Burgard, Frank Dellaert, and Sebastian Thrun. Monte carlo localization: efficient position estimation for mobile robots. pages 343–349, 1999.

[62] Mengyin Fu, Minzhao Zhu, Yi Yang, Wenjie Song, and Meiling Wang. Lidar-based vehicle localization on the satellite image via a neural network. *Robotics and Autonomous Systems*, 129:103519, 2020.

[63] T. Fujiwara, M. Brotas, and M.E. Chiappe. Walking strides direct rapid and flexible recruitment of visual circuits for course control in drosophila. *Neuron*, 110(13):2124–2138.e8, 2022.

[64] N. Funabiki, B. Morrell, J. Nash, and A. a. Agha-mohammadi. Range-aided pose-graph-based slam: Applications of deployable ranging beacons for unknown environment exploration. *IEEE Robotics and Automation Letters*, 6(1):48–55, Jan 2021.

[65] Christopher J Gabriel, Zachary Zeidler, Benita Jin, Changliang Guo, Caitlin M Goodpaster, Adrienne Q Kashay, Anna Wu, Molly Delaney, Jovian Cheung, Lauren E DiFazio, Melissa J Sharpe, Daniel Aharoni, Scott A Wilke, and Laura A DeNardo. Behaviordepot is a simple, flexible tool for automated behavioral detection based on markerless pose tracking. *Elife*, 11, Aug 2022.

[66] Matt Gaidica and Ben Dantzer. An implantable neurophysiology platform: Broadening research capabilities in free-living and non-traditional animals. *Frontiers in neural circuits*, 16:940989, 2022.

[67] Junli Gao, Weijie Ye, Jing Guo, and Zhongjuan Li. Deep reinforcement learning for indoor mobile robot path planning. *Sensors*, 20(19), 2020.

[68] Ana Rita Gaspar, Alexandra Nunes, Andry Maykol Pinto, and Aníbal Matos. Urban@cras dataset: Benchmarking of visual odometry and slam techniques. *Robotics and Autonomous Systems*, 109:59–67, 2018.

[69] Jinne E Geelen, Mariana P Branco, Nick F Ramsey, Frans C T van der Helm, Winfred Mugge, and Alfred C Schouten. Markerless motion capture: Ml-mocap, a low-cost modular multi-camera setup. *Annu Int Conf IEEE Eng Med Biol Soc*, 2021:4859–4862, Nov 2021.

[70] Andreas Geiger, Philip Lenz, Christoph Stiller, and Raquel Urtasun. Vision meets robotics: The kitti dataset. *International Journal of Robotics Research (IJRR)*, 2013.

[71] A. Gerós, A. Magalhães, and P. Aguiar. Improved 3d tracking and automated classification of rodents' behavioral activity using depth-sensing cameras. *Behavior Research Methods*, 52(5):2156–2167, Oct 2020.

[72] B.Q. Geuther, A. Peer, H. He, G. Sabnis, V.M. Philip, and V. Kumar. Action detection using a neural network elucidates the genetics of mouse grooming behavior. *eLife*, 10, Mar 2021.

[73] Alex Gomez-Marin. A clash of umwelts: Anthropomorphism in behavioral neuroscience. *Behav Brain Sci*, 42:e229, Nov 2019.

[74] Paul N Goncharow and Shawn M Beaudette. Assessing time-varying lumbar flexion-extension kinematics using automated pose estimation. *J Appl Biomech*, 38(5):355–360, Oct 2022.

[75] Z. Gong, J. Li, Z. Luo, C. Wen, C. Wang, and J. Zelek. Mapping and semantic modeling of underground parking lots using a backpack lidar system. *IEEE Transactions on Intelligent Transportation Systems*, 22(2):734–746, Feb 2021.

[76] J.M. Graving, D. Chae, H. Naik, L. Li, B. Koger, B.R. Costelloe, and I.D. Couzin. Deepposekit, a software toolkit for fast and robust animal pose estimation using deep learning. *eLife*, 8, 2019.

[77] Edyta Hadas, Grzegorz Jozkow, Agata Walicka, and Andrzej Borkowski. Apple orchard inventory with a lidar equipped unmanned aerial system. *International Journal of Applied Earth Observation and Geoinformation*, 82:101911, 2019.

[78] G. He, X. Yuan, Y. Zhuang, and H. Hu. An integrated gnss/lidar-slam pose estimation framework for large-scale map building in partially gnss-denied environments. *IEEE Transactions on Instrumentation and Measurement*, 70:1–9, 2021.

[79] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. *CoRR*, abs/1512.03385, 2015.

[80] S.F. Henriques, D.B. Dhakan, L. Serra, A.P. Francisco, Z. Carvalho-Santos, C. Baltazar, A.P. Elias, M. Anjos, T. Zhang, O.D.K. Maddocks, and C. Ribeiro. Metabolic cross-feeding in imbalanced diets allows gut microbes to improve reproduction and alter host behaviour. *Nature Communications*, 11(1), 2020.

[81] M. Hernandez-Pajares, J. M. J. Zomoza, J. S. Subirana, and O. L. Colombo. Feasibility of wide-area subdecimeter navigation with galileo and modernized gps. *IEEE Transactions on Geoscience and Remote Sensing*, 41(9):2128–2131, Sep. 2003.

[82] W. Hess, D. Kohler, H. Rapp, and D. Andor. Real-time loop closure in 2d lidar slam. volume 2016-June, pages 1271–1278, 2016. cited By 592.

[83] Kayleigh E. Hood, Eden Long, Eric Navarro, and Laura M. Hurley. Playback of broadband vocalizations of female mice suppresses male ultrasonic calls. *PloS one*, 18:e0273742, 2023.

[84] Qing Hou and Chengbo Ai. A network-level sidewalk inventory method using mobile lidar and deep learning. *Transportation Research Part C: Emerging Technologies*, 119:102772, 2020.

[85] S. Hrvatin, S. Sun, O.F. Wilcox, H. Yao, A.J. Lavin-Peter, M. Cicconet, E.G. Assad, M.E. Palmer, S. Aronson, A.S. Banks, E.C. Griffith, and M.E. Greenberg. Neurons that regulate mouse torpor. *Nature*, 583(7814):115–121, 2020.

[86] A.I. Hsu and E.A. Yttri. B-soid, an open-source unsupervised algorithm for identification and fast prediction of behaviors. *Nature Communications*, 12(1), 2021.

[87] G. Huang, Z. Liu, L. Van Der Maaten, and K.Q. Weinberger. Densely connected convolutional networks. volume 2017-January, pages 2261–2269, 2017.

[88] X. Huang, X. Cheng, Q. Geng, B. Cao, D. Zhou, P. Wang, Y. Lin, and R. Yang. The apolloscape dataset for autonomous driving. volume 2018-June, pages 1067–1073, 2018. cited By 104.

[89] M.M. Hurley, A.N. Nawari, V.X. Chen, S.C. O'Brien, A.I. Sabir, E.J. Goodman, L.J. Wiles, A. Biswas, S.A. Aston, S.G. Khambadkone, K.L. Tamashiro, and T.H. Moran. Adolescent female rats recovered from the activity-based anorexia display blunted hedonic responding. *International Journal of Eating Disorders*, 55(8):1042–1053, 2022.

[90] Arshad Husain and Rakesh Chandra Vaishya. Road surface and its center line and boundary lines detection using terrestrial lidar data. *The Egyptian Journal of Remote Sensing and Space Science*, 21(3):363–374, 2018.

[91] Indu Indirabai, M.V. Harindranathan Nair, Jaishanker R. Nair, and Rama Rao Nidamanuri. Direct estimation of leaf area index of tropical forests using lidar point cloud. *Remote Sensing Applications: Society and Environment*, 18:100295, 2020.

[92] J. Jeong, Y. Cho, Y.-S. Shin, H. Roh, and A. Kim. Complex urban dataset with multi-level sensors from highly diverse urban environments. *International Journal of Robotics Research*, 38(6):642–657, 2019. cited By 38.

[93] H. Jhuang, E. Garrote, X. Yu, V. Khilnani, T. Poggio, A.D. Steele, and T. Serre. Automated home-cage behavioural phenotyping of mice. *Nature Communications*, 1(6), 2010.

[94] Y. Jia, S. Li, X. Guo, B. Lei, J. Hu, X.-H. Xu, and W. Zhang. Selfee, self-supervised features extraction of animal behaviors. *eLife*, 11, 2022.

[95] T. Jin and F. Duan. Rat behavior observation system based on transfer learning. *IEEE Access*, 7:62152–62162, 2019.

[96] Cody L. Johnson, Qin Chen, and Celalettin E. Ozdemir. Lidar time-series analysis of a rapidly transgressing low-lying mainland barrier (caminada headlands, louisiana, usa). *Geomorphology*, 352:106979, 2020.

[97] M. Kabra, A.A. Robie, M. Rivera-Alba, S. Branson, and K. Branson. Jaaba: Interactive machine learning for automatic annotation of animal behavior. *Nature Methods*, 10(1):64–67, 2013.

[98] L.P. Kaelbling, M.L. Littman, and A.W. Moore. Reinforcement learning: A survey. *Journal of Artificial Intelligence Research*, 4:237–285, 1996. cited By 3976.

[99] Neri Kafkafi, Michal Pagis, Dina Lipkind, Cheryl L. Mayo, Yoav Bemjamini, Ilan Golani, and Gregory I. Elmer. Darting behavior: a quantitative movement pattern designed for discrimination and replicability in mouse locomotor behavior. *Behavioural Brain Research*, 142(1):193–205, 2003.

[100] Ioannis Karamitros, Athanassios Ganas, Alexandros Chatzipetros, and Sotirios Valkaniotis. Non-planarity, scale-dependent roughness and kinematic properties of the pidima active normal fault scarp (messinia, greece) using high-resolution terrestrial lidar data. *Journal of Structural Geology*, 136:104065, 2020.

[101] Pierre Karashchuk, Katie L Rupp, Evyn S Dickinson, Sarah Walling-Bell, Elischa Sanders, Eiman Azim, Bingni W Brunton, and John C Tuthill.

Anipose: A toolkit for robust markerless 3d pose estimation. *Cell Rep*, 36(13):109730, Sep 2021.

[102] Gerd Kempermann, Jadna Bogado Lopes, Sara Zocher, Susan Schilling, Fanny Ehret, Alexander Garthe, Anne Karasinsky, Andreas M. Brandmaier, Ulman Lindenberger, York Winter, and Rupert W. Overall. The individuality paradigm: Automated longitudinal activity tracking of large cohorts of genetically identical mice in an enriched environment. *Neurobiology of disease*, 175:105916, Dec 2022.

[103] Ahmed El Khazari, Yue Que, Thai Leang Sung, and Hyo Jong Lee. Deep global features for point cloud alignment. *Sensors*, 20(14):4032, 07 2020.

[104] Hyungjin Kim, Seungwon Song, and Hyun Myung. Gp-icp: Ground plane icp for mobile robots. *IEEE Access*, 7:76599–76610, 2019.

[105] J. Kim, J. Park, and W. Chung. Self-diagnosis of localization status for autonomous mobile robots. *Sensors (Switzerland)*, 18(9), 2018.

[106] Pileun Kim, Jingdao Chen, and Yong K. Cho. Slam-driven robotic mapping and registration of 3d point clouds. *Automation in Construction*, 89:38–48, 2018.

[107] Nathan J Kirkpatrick, Robert J Butera, and Young-Hui Chang. Deeplabcut increases markerless tracking efficiency in x-ray video analysis of rodent locomotion. *J Exp Biol*, 225(16), Aug 2022.

[108] Stefan Kohlbrecher, Oskar von Stryk, Johannes Meyer, and Uwe Klingauf. A flexible and scalable slam system with full 3d motion estimation. In *2011 IEEE International Symposium on Safety, Security, and Rescue Robotics*, pages 155–160, 2011.

[109] K. Koide, J. Miura, M. Yokozuka, S. Oishi, and A. Banno. Interactive 3d graph slam for map correction. *IEEE Robotics and Automation Letters*, 6(1):40–47, Jan 2021.

[110] M. Kolakowski, V. Djaja-Josko, and J. Kolakowski. Static lidar assisted uwb anchor nodes localization. *IEEE Sensors Journal*, pages 1–1, 2020.

[111] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C.J. Burges, L. Bottou, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 25. Curran Associates, Inc., 2012.

[112] Manabu Kume, Yudai Yoshikawa, Tomoichiro Tanaka, Shun Watanabe, Hiromichi Mitamura, and Yoh Yamashita. Water temperature and precipitation stimulate small-sized japanese eels to climb a low-height vertical weir. *PloS one*, 17:e0279617, 2022.

[113] E. Kuramoto, A. Kitawaki, T. Yagi, H. Kono, S.-E. Matsumoto, H. Hara, Y. Ohyagi, H. Iwai, A. Yamanaka, and T. Goto. Development of a system to analyze oral frailty associated with alzheimer's disease using a mouse model. *Frontiers in Aging Neuroscience*, 14, 2022.

[114] Jessy Lauer, Mu Zhou, Shaokai Ye, William Menegas, Steffen Schneider, Tanmay Nath, Mohammed Mostafizur Rahman, Valentina Di Santo, Daniel Soberanes, Guoping Feng, Venkatesh N Murthy, George Lauder, Catherine Dulac, Mackenzie Weygandt Mathis, and Alexander Mathis. Multi-animal pose estimation, identification and tracking with deeplabcut. *Nat Methods*, 19(4):496–504, Apr 2022.

[115] S. Lee, C. Kim, S. Cho, S. Myoungho, and K. Jo. Robust 3-dimension point cloud mapping in dynamic environment using point-wise static

probability-based ndt scan-matching. *IEEE Access*, 8:175563–175575, 2020.

[116] J. Levinson and S. Thrun. Robust vehicle localization in urban environments using probabilistic maps. In *2010 IEEE International Conference on Robotics and Automation*, pages 4372–4378, May 2010.

[117] Jesse Levinson and Sebastian Thrun. *Unsupervised Calibration for Multi-beam Lasers*, pages 179–193. Springer Berlin Heidelberg, Berlin, Heidelberg, 2014.

[118] HuiXia Li, LongHui Ao, Hang Guo, and XiaoYi Yan. Indoor multi-sensor fusion positioning based on federated filtering. *Measurement*, 154:107506, 2020.

[119] J. Li, P.A. Kells, A.C. Osgood, S.H. Gautam, and W.L. Shew. Collapse of complexity of brain and body activity due to excessive inhibition and mecp2 disruption. *Proceedings of the National Academy of Sciences of the United States of America*, 118(43), 2021.

[120] Jianping Li, Bisheng Yang, Chi Chen, and Ayman Habib. Nrli-uav: Non-rigid registration of sequential raw laser scans and images for low-cost uav lidar point cloud quality improvement. *ISPRS Journal of Photogrammetry and Remote Sensing*, 158:123–145, 2019.

[121] Jing Li, Xin Zhang, Jiehao Li, Yanyu Liu, and Junzheng Wang. Building and optimization of 3d semantic map based on lidar and camera fusion. *Neurocomputing*, 409:394–407, 2020.

[122] M. Li, H. Zhu, S. You, L. Wang, and C. Tang. Efficient laser-based 3d slam for coal mine rescue robots. *IEEE Access*, 7:14124–14138, 2019.

[123] Minglei Li, Franz Rottensteiner, and Christian Heipke. Modelling of buildings from aerial lidar point clouds using tins and label maps. *ISPRS Journal of Photogrammetry and Remote Sensing*, 154:127–138, 2019.

[124] Shuaixin Li, Guangyun Li, Li Wang, and Yuchu Qin. Slam integrated mobile mapping system in complex urban environments. *ISPRS Journal of Photogrammetry and Remote Sensing*, 166:316–332, 2020.

[125] Xiang Li, Lingjing Wang, Mingyang Wang, Congcong Wen, and Yi Fang. Dance-net: Density-aware convolution networks with context encoding for airborne lidar point cloud classification. *ISPRS Journal of Photogrammetry and Remote Sensing*, 166:128–139, 2020.

[126] Yunlin Li, Fengye Wu, Qinyan Wu, Wenya Liu, Guanghui Li, Benxing Yao, Ran Xiao, Yudie Hu, and Junsong Wang. A novel open-source raspberry pi-based behavioral testing in zebrafish. *PloS one*, 17:e0279550, 2022.

[127] S. Liang, Z. Cao, C. Wang, and J. Yu. A novel 3d lidar slam based on directed geometry point and sparse frame. *IEEE Robotics and Automation Letters*, 6(2):374–381, 2021.

[128] L. Long, Z.V. Johnson, J. Li, T.J. Lancaster, V. Aljapur, J.T. Streelman, and P.T. McGrath. Automatic classification of cichlid behaviors using 3d convolutional residual networks. *iScience*, 23(10), 2020.

[129] Luca Lonini, Yaejin Moon, Kyle Embry, R James Cotton, Kelly McKenzie, Sophia Jenz, and Arun Jayaraman. Video-based pose estimation for gait analysis in stroke survivors during clinical assessments: A proof-of-concept study. *Digit Biomark*, 6(1):9–18, 2022.

[130] G. Lopes and P. Monteiro. New open-source tools: Using bonsai for behavioral tracking and closed-loop experiments. *Frontiers in Behavioral Neuroscience*, 15, 2021.

[131] J. Lu, W. Wang, Z. Fan, S. Bi, and C. Guo. Point cloud registration based on cpd algorithm. In *2018 37th Chinese Control Conference (CCC)*, pages 8235–8240, July 2018.

[132] K. Luxem, P. Mocellin, F. Fuhrmann, J. Kürsch, S.R. Miller, J.J. Palop, S. Remy, and P. Bauer. Identifying behavioral structure from deep variational embeddings of animal motion. *Communications Biology*, 5(1), 2022.

[133] Will Maddern, Geoff Pascoe, Chris Linegar, and Paul Newman. 1 Year, 1000km: The Oxford RobotCar Dataset. *The International Journal of Robotics Research (IJRR)*, 36(1):3–15, 2017.

[134] Dipendra Magaju, John Montgomery, Paul Franklin, Cindy Baker, and Heide Friedrich. Machine learning based assessment of small-bodied fish tracking to evaluate spoiler baffle fish passage design. *J Environ Manage*, 325(Pt A):116507, Jan 2023.

[135] Martin Magnusson, Achim Lilienthal, and Tom Duckett. Scan registration for autonomous mining vehicles using 3d-ndt. *Journal of Field Robotics*, 24:803–827, 10 2007.

[136] A. Mancini, E. Frontoni, and P. Zingaretti. Embedded multisensor system for safe point-to-point navigation of impaired users. *IEEE Transactions on Intelligent Transportation Systems*, 16(6):3543–3555, Dec 2015.

[137] Sina Sharif Mansouri, Christoforos Kanellakis, Dariusz Kominiak, and George Nikolakopoulos. Deploying mavs for autonomous navigation in

dark underground mine environments. *Robotics and Autonomous Systems*, 126:103472, 2020.

[138] Alan David Marcus, Satyanarayana Achanta, and Sven-Eric Jordt. Protocol for non-invasive assessment of spontaneous movements of group-housed animals using remote video monitoring. *STAR protocols*, 3:101326, Jun 2022.

[139] Markus Marks, Jin Qiuhan, Oliver Sturman, Lukas von Ziegler, Sepp Kollmorgen, Wolfger von der Behrens, Valerio Mante, Johannes Bohacek, and Mehmet Fatih Yanik. Deep-learning based identification, tracking, pose estimation, and behavior classification of interacting primates and mice in complex environments. *Nature machine intelligence*, 4:331–340, Apr 2022.

[140] Jesse D. Marshall, Tianqing Li, Joshua H. Wu, and Timothy W. Dunn. Leaving flatland: Advances in 3d behavioral measurement. *Current opinion in neurobiology*, 73:102522, Apr 2022.

[141] A. Mathis, P. Mamidanna, K.M. Cury, T. Abe, V.N. Murthy, M.W. Mathis, and M. Bethge. Deeplabcut: markerless pose estimation of user-defined body parts with deep learning. *Nature Neuroscience*, 21(9):1281–1289, Sep 2018.

[142] Janne Mäyrä, Sarita Keski-Saari, Sonja Kivinen, Topi Tanhuanpää, Pekka Hurskainen, Peter Kullberg, Laura Poikolainen, Arto Viinikka, Sakari Tuominen, Timo Kumpula, and Petteri Vihervaara. Tree species classification from airborne hyperspectral and lidar data using 3d convolutional neural networks. *Remote Sensing of Environment*, 256:112322, 2021.

[143] Jordan Mitchell and Joshua A. Marshall. Towards a novel auto-rotating lidar platform for cavity surveying. *Tunnelling and Underground Space Technology*, 97:103260, 2020.

[144] M. A. Mitteta, H. Nouira, X. Roynard, F. Goulette, and J. E. Deschaud. Experimental Assessment of the Quanergy M8 LIDAR Sensor. *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 41B5:527–531, Jun 2016.

[145] A. Myronenko and X. Song. Point set registration: Coherent point drift. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(12):2262–2275, Dec 2010.

[146] Darshana Z. Narayanan, Daniel Y. Takahashi, Lauren M. Kelly, Sabina I. Hlavaty, Junzhou Huang, and Asif A. Ghazanfar. Prenatal development of neonatal vocalizations. *eLife*, 11, Jul 2022.

[147] Laurie Needham, Murray Evans, Darren P Cosker, Logan Wade, Polly M McGuigan, James L Bilzon, and Steffi L Colyer. The accuracy of several pose estimation methods for 3d joint centre localisation. *Sci Rep*, 11(1):20673, Oct 2021.

[148] J.P. Neunuebel, A.L. Taylor, B.J. Arthur, and S.E. Roian Egnor. Female mice ultrasonically interact with males during courtship displays. *eLife*, 4(MAY):1–24, 2015.

[149] M. Oelsch, M. Karimi, and E. Steinbach. R-loam: Improving lidar odometry and mapping with point-to-mesh features of a known 3d reference object. *IEEE Robotics and Automation Letters*, pages 1–1, 2021.

[150] R. Opromolla and A. Nocerino. Uncooperative spacecraft relative navigation with lidar-based unscented kalman filter. *IEEE Access*, 7:180012–180026, 2019.

[151] Edgar Osuna, Robert Freund, and Federico Girosi. Support vector machines: Training and applications. Technical report, USA, 1997.

[152] M. Palieri, B. Morrell, A. Thakur, K. Ebadi, J. Nash, A. Chatterjee, C. Kanellakis, L. Carlone, C. Guaragnella, and A. a. Agha-mohammadi. Locus: A multi-sensor lidar-centric solution for high-precision odometry and 3d mapping in real-time. *IEEE Robotics and Automation Letters*, 6(2):421–428, 2021.

[153] Gaurav Pandey, James R. McBride, and Ryan M. Eustice. Ford campus vision and lidar data set. *International Journal of Robotics Research*, 30(13):1543–1552, NOV 2011. PT: J; UT: WOS:000296206800001.

[154] C. Paris and L. Bruzzone. A three-dimensional model-based approach to the estimation of the tree top height by fusing low-density lidar data and very high resolution optical images. *IEEE Transactions on Geoscience and Remote Sensing*, 53(1):467–480, Jan 2015.

[155] Philip R L Parker, Elliott T T Abe, Natalie T Beatie, Emmalyn S P Leonard, Dylan M Martins, Shelby L Sharp, David G Wyrick, Luca Mazzucato, and Cristopher M Niell. Distance estimation from monocular cues in an ethological visuomotor task. *Elife*, 11, Sep 2022.

[156] Pierantonio Parmiani, Cristina Lucchetti, Claudio Bonifazzi, and Gianfranco Franchi. A kinematic study of skilled reaching movement in rat. *J Neurosci Methods*, 328:108404, Dec 2019.

[157] H. Peel, S. Luo, A.G. Cohn, and R. Fuentes. Localisation of a mobile robot for bridge bearing inspection. *Automation in Construction*, 94:244–256, 2018.

[158] Talmo D. Pereira, Nathaniel Tabris, Arie Matsliah, David M. Turner, Junyu Li, Shruthi Ravindranath, Eleni S. Papadoyannis, Edna Normand, David S. Deutsch, Z. Yan Wang, Grace C. McKenzie-Smith, Catalin C. Mitelut, Marielisa Diez Castro, John D'Uva, Mikhail Kislin, Dan H. Sanes, Sarah D. Kocher, Samuel S.-H. Wang, Annegret L. Falkner, Joshua W. Shaevitz, and Mala Murthy. Sleap: A deep learning system for multi-animal pose tracking. *Nature methods*, 19:486–495, Apr 2022.

[159] T.D. Pereira, D.E. Aldarondo, L. Willmore, M. Kislin, S.S.-H. Wang, M. Murthy, and J.W. Shaevitz. Fast animal pose estimation using deep neural networks. *Nature Methods*, 16(1):117–125, 2019.

[160] Marek Pierzchała, Philippe Giguère, and Rasmus Astrup. Mapping forests using an unmanned ground vehicle with 3d lidar and graph-slam. *Computers and Electronics in Agriculture*, 145:217–225, 2018.

[161] Przemyslaw Polewski, Wei Yao, Lin Cao, and Sha Gao. Marker-free coregistration of uav and backpack lidar point clouds in forested areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 147:307–318, 2019.

[162] François Pomerleau, Francis Colas, Roland Siegwart, and Stéphane Magnenat. Comparing icp variants on real-world data sets. *Autonomous Robots*, 04 2013.

[163] Patricia Pons, Javier Jaen, and Alejandro Catala. Assessing machine learning classifiers for the detection of animals' behavior using depth-based tracking. *Expert Systems with Applications*, 86:235 – 246, 2017.

[164] A. Popov, N. Brazhe, A. Fedotova, A. Tiaglik, M. Bychkov, K. Morozova, A. Brazhe, D. Aronov, E. Lyukmanova, N. Lazareva, L. Li, E. Ponimaskin, A. Verkhratsky, and A. Semyanov. A high-fat diet changes astrocytic metabolism to promote synaptic plasticity and behavior. *Acta Physiologica*, 236(1), 2022.

[165] C. Poullis. Large-scale urban reconstruction with tensor clustering and global boundary refinement. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 42(5):1132–1145, May 2020.

[166] Charles R. Qi, Hao Su, Kaichun Mo, and Leonidas J. Guibas. Pointnet: Deep learning on point sets for 3d classification and segmentation, 2017.

[167] Charles R. Qi, Li Yi, Hao Su, and Leonidas J. Guibas. Pointnet++: Deep hierarchical feature learning on point sets in a metric space. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 5105–5114, Red Hook, NY, USA, 2017. Curran Associates Inc.

[168] L.P. Quinn, T.O. Stean, B. Trail, M.S. Duxon, S.C. Stratton, A. Billinton, and N. Upton. Laboras™: Initial pharmacological validation of a system allowing continuous monitoring of laboratory rodent behaviour. *Journal of Neuroscience Methods*, 130(1):83–92, 2003.

[169] M. Ramezani, G. Tinchev, E. Iuganov, and M. Fallon. Online lidar-slam for legged robots with robust registration and deep-learned loop closure. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4158–4164, 2020.

[170] M. Ramezani, Y. Wang, M. Camurri, D. Wisth, M. Mattamala, and M. Fallon. The newer college dataset: Handheld lidar, inertial and vi-

sion with ground truth. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020.

[171] D. Rodrigues, L. Jacinto, M. Falcão, A.C. Castro, A. Cruz, C. Santa, B. Manadas, F. Marques, N. Sousa, and P. Monteiro. Chronic stress causes striatal disinhibition mediated by som-interneurons in male mice. *Nature Communications*, 13(1), 2022.

[172] C. Rose, J. Britt, J. Allen, and D. Bevly. An integrated vehicle navigation system utilizing lane-detection and lateral position estimation systems in difficult environments for gps. *IEEE Transactions on Intelligent Transportation Systems*, 15(6):2615–2629, Dec 2014.

[173] N. Sakamoto, K. Kobayashi, T. Yamamoto, S. Masuko, M. Yamamoto, and T. Murata. Automated grooming detection of mouse by three-dimensional convolutional neural network. *Frontiers in Behavioral Neuroscience*, 16, 2022.

[174] Nivethini Sangarapillai, Markus Wöhr, and Rainer K. W. Schwarting. Appetitive 50 khz calls in a pavlovian conditioned approach task in cacna1c haploinsufficient rats. *Physiology & behavior*, 250:113795, Jun 2022.

[175] Aleksandr Segal, Dirk Haehnel, and Sebastian Thrun. Generalized-icp. In *Robotics: science and systems*, volume 2, page 435. Seattle, WA, 2009.

[176] C. Segalin, J. Williams, T. Karigo, M. Hui, M. Zelikowsky, J.J. Sun, P. Perona, D.J. Anderson, and A. Kennedy. The mouse action recognition system (mars) software pipeline for automated analysis of social behaviors in mice. *eLife*, 10, 2021.

[177] Tixiao Shan and Brendan Englot. Lego-loam: Lightweight and ground-optimized lidar odometry and mapping on variable terrain. In *2018*

IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), pages 4758–4765, 2018.

[178] Jeremy Sofonia, Yuri Shendryk, Stuart Phinn, Chris Roelfsema, Farid Kendoul, and Danielle Skocaj. Monitoring sugarcane growth response to varying nitrogen application rates: A comparison of uav slam lidar and photogrammetry. *International Journal of Applied Earth Observation and Geoinformation*, 82:101878, 2019.

[179] M. Soilán, B. Riveiro, J. Martínez-Sánchez, and P. Arias. Traffic sign detection in mls acquired point clouds for geometric and image-based semantic inventory. *ISPRS Journal of Photogrammetry and Remote Sensing*, 114:92–101, 2016.

[180] J. Song, S. Xia, J. Wang, and D. Chen. Curved buildings reconstruction from airborne lidar data by matching and deforming geometric primitives. *IEEE Transactions on Geoscience and Remote Sensing*, 59(2):1660–1674, Feb 2021.

[181] Oliver Sturman, Lukas von Ziegler, Christa Schläppi, Furkan Akyol, Mattia Privitera, Daria Slominski, Christina Grimm, Laetitia Thieren, Valerio Zerbi, Benjamin Grewe, and Johannes Bohacek. Deep learning-based behavioral analysis reaches human accuracy and is capable of outperforming commercial solutions. *Neuropsychopharmacology*, 45(11):1942–1952, Oct 2020.

[182] Feng Su, Yangzhen Wang, Mengping Wei, Chong Wang, Shaoli Wang, Lei Yang, Jianmin Li, Peijiang Yuan, Dong-Gen Luo, and Chen Zhang. Noninvasive tracking of every individual in unmarked mouse groups using multi-camera fusion and deep learning. *Neuroscience bulletin*, Dec 2022.

[183] L. Sun, Z. Yan, A. Zaganidis, C. Zhao, and T. Duckett. Recurrent-octomap: Learning state-based map refinement for long-term semantic mapping with 3-d-lidar data. *IEEE Robotics and Automation Letters*, 3(4):3749–3756, 2018.

[184] Michael Edbert Suryanto, Ferry Saputra, Kevin Adi Kurnia, Ross D Vasquez, Marri Jmelou M Roldan, Kelvin H-C Chen, Jong-Chin Huang, and Chung-Der Hsiao. Using deeplabcut as a real-time and markerless tool for cardiac physiology assessment in zebrafish. *Biology (Basel)*, 11(8), Aug 2022.

[185] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. volume 07-12-June-2015, pages 1–9, 2015.

[186] Masaki Takenaka, Kosuke Hayashi, Genki Yamada, Takayuki Ogura, Mone Ito, Alexander M. Milner, and Koji Tojo. Behavior of snow monkeys hunting fish to survive winter. *Scientific reports*, 12:20324, Nov 2022.

[187] G. Tarcsay, B.L. Boublil, and L.A. Ewell. Low-cost platform for multi-animal chronic local field potential video monitoring with graphical user interface (gui) for seizure detection and behavioral scoring. *eNeuro*, 9(5), 2022.

[188] Leyre-Torre Tojal, Aitor Bastarrika, Brian Barrett, Javier Maria Sanchez Espeso, Jose Manuel Lopez-Guede, and Manuel Graña. Prediction of aboveground biomass from low-density lidar data: Validation over p. radiata data from a region north of spain. *Forests*, 10(9), 2019.

[189] Suramya Tomar. Converting video formats with ffmpeg. *Linux Journal*, 2006(146):10, 2006.

[190] Leyre Torre-Tojal, Jose Manuel Lopez-Guede, and Manuel M. Graña Romay. Estimation of forest biomass from light detection and ranging data by using machine learning. *Expert Systems*, 36(4):e12399, 2019. e12399 10.1111/exsy.12399.

[191] Guido Van Rossum and Fred L Drake Jr. *Python reference manual*. Centrum voor Wiskunde en Informatica Amsterdam, 1995.

[192] V. Vaquero, I. d. Pino, F. Moreno-Noguer, J. Solà, A. Sanfeliu, and J. Andrade-Cetto. Dual-branch cnns for vehicle detection and tracking on lidar data. *IEEE Transactions on Intelligent Transportation Systems*, pages 1–12, 2020.

[193] S. Venkatraman, X. Jin, R.M. Costa, and J.M. Carmena. Investigating neural correlates of behavior in freely behaving rodents using inertial sensors. *Journal of Neurophysiology*, 104(1):569–575, 2010.

[194] Wouter B. Verschoof-van der Vaart and Juergen Landauer. Using carcassonnet to automatically detect and trace hollow roads in lidar data from the netherlands. *Journal of Cultural Heritage*, 47:143–154, 2021.

[195] Heike Vester, Kurt Hammerschmidt, Marc Timme, and Sarah Hallerberg. Quantifying group specificity of animal vocalizations without specific sender information. *Phys Rev E*, 93(2):022138, Feb 2016.

[196] Justin M. Vilbig, Vasit Sagan, and Christopher Bodine. Archaeological surveying with airborne lidar and uav photogrammetry: A comparative analysis at cahokia mounds. *Journal of Archaeological Science: Reports*, 33:102509, 2020.

[197] Elise Klæbo Vonstad, Xiaomeng Su, Beatrix Vereijken, Kerstin Bach, and Jan Harald Nilsen. Comparison of a deep learning-based pose estimation

system to marker-based and kinect systems in exergaming for balance training. *Sensors (Basel)*, 20(23), Dec 2020.

[198] H. Wang, T. Hu, Z. Wang, Z. Kang, P. H. Akwensi, and J. Yang. Reconstruction of power pylons from lidar point clouds based on structural segmentation and parameter estimation. *IEEE Geoscience and Remote Sensing Letters*, pages 1–5, 2020.

[199] Heng Wang, Bin Wang, Bingbing Liu, Xiaoli Meng, and Guanghong Yang. Pedestrian recognition and tracking using 3d lidar for autonomous vehicle. *Robotics and Autonomous Systems*, 88:71–78, 2017.

[200] Jiexin Wang, Stefan Elfwing, and Eiji Uchibe. Modular deep reinforcement learning from reward and punishment for robot navigation. *Neural Networks*, 135:115–126, 2021.

[201] Jinxin Wang, Paniz Karbasi, Liqiang Wang, and Julian P Meeks. A layered, hybrid machine learning analytic workflow for mouse risk assessment behavior. *eNeuro*, Dec 2022.

[202] S. Wang, M. Bai, G. Mattyus, H. Chu, W. Luo, B. Yang, J. Liang, J. Cheverie, S. Fidler, and R. Urtasun. Torontocity: Seeing the world with a million eyes. volume 2017-October, pages 3028–3036, 2017. cited By 57.

[203] Z. Wang, W. Li, Y. Shen, and B. Cai. 4-d slam: An efficient dynamic bayes network-based approach for dynamic scene understanding. *IEEE Access*, 8:219996–220014, 2020.

[204] Z. Wang, L. Zhang, S. Zhao, and S. Zhang. Global localization with a single-line lidar by dense 2d signature and 1d registration. *IEEE Sensors Journal*, pages 1–1, 2020.

[205] Rebecca Z Weber, Geertje Mulders, Julia Kaiser, Christian Tackenberg, and Ruslan Rust. Deep learning-based behavioral profiling of rodent stroke recovery. *BMC Biol*, 20(1):232, Oct 2022.

[206] Weisong Wen, Xiwei Bai, Li-Ta Hsu, and Tim Pfeifer. Gnss/lidar integration aided by self-adaptive gaussian mixture models in urban scenarios: An approach robust to non-gaussian noise. In *2020 IEEE/ION Position, Location and Navigation Symposium (PLANS)*, pages 647–654, 2020.

[207] Matthew R Whiteway, Dan Biderman, Yoni Friedman, Mario Dipoppa, E Kelly Buchanan, Anqi Wu, John Zhou, Niccolò Bonacchi, Nathaniel J Miska, Jean-Paul Noel, Erica Rodriguez, Michael Schartner, Karolina Socha, Anne E Urai, C Daniel Salzman, John P Cunningham, and Liam Paninski. Partitioning variability in animal behavioral videos using semi-supervised variational autoencoders. *PLoS Comput Biol*, 17(9):e1009439, Sep 2021.

[208] Alan Wrench and Jonathan Balch-Tomes. Beyond the edge: Markerless pose estimation of speech articulators from ultrasound and camera images using deeplabcut. *Sensors (Basel)*, 22(3), Feb 2022.

[209] D. Wu, Y. Meng, K. Zhan, and F. Ma. A lidar slam based on point-line features for underground mining vehicle. In *2018 Chinese Automation Congress (CAC)*, pages 2879–2883, Nov 2018.

[210] Z. Xuexi, L. Guokun, F. Genping, X. Dongliang, and L. Shiliu. Slam algorithm analysis of mobile robot based on lidar. In *2019 Chinese Control Conference (CCC)*, pages 4739–4745, July 2019.

[211] R. Yagfarov, M. Ivanou, and I. Afanasyev. Map comparison of lidar-based 2d slam algorithms using precise ground truth. In *2018 15th International*

Conference on Control, Automation, Robotics and Vision (ICARCV), pages 1979–1983, Nov 2018.

[212] Lin Yang, Yehua Sheng, and Bo Wang. Lidar data reduction assisted by optical image for 3d building reconstruction. *Optik*, 125(20):6282–6286, 2014.

[213] Qin Ye, Pengcheng Shi, Kunyuan Xu, Popo Gui, and Shaoming Zhang. A novel loop closure detection approach using simplified structure for low-cost lidar. *Sensors*, 20(8), 2020.

[214] Abdurrahman Yilmaz and Hakan Temeltas. Self-adaptive monte carlo method for indoor localization of smart agvs using lidar data. *Robotics and Autonomous Systems*, 122:103285, 2019.

[215] H. Yin, Y. Wang, X. Ding, L. Tang, S. Huang, and R. Xiong. 3d lidar-based global localization using siamese neural network. *IEEE Transactions on Intelligent Transportation Systems*, 21(4):1380–1392, 2020.

[216] Shangshu Yu, Cheng Wang, Zenglei Yu, Xin Li, Ming Cheng, and Yu Zang. Deep regression for lidar-based localization in dense urban areas. *ISPRS Journal of Photogrammetry and Remote Sensing*, 172:240–252, 2021.

[217] Yikuan Yu, Zitian Huang, Fei Li, Haodong Zhang, and Xinyi Le. Point encoder gan: A deep learning model for 3d point cloud inpainting. *Neurocomputing*, 384:192–199, 2020.

[218] Ting Yun, Kang Jiang, Guangchao Li, Markus P. Eichhorn, Jiangchuan Fan, Fangzhou Liu, Bangqian Chen, Feng An, and Lin Cao. Individual tree crown segmentation from airborne lidar data using a novel gaussian filter and energy function minimization-based approach. *Remote Sensing of Environment*, 256:112307, 2021.

[219] F. Zhang, N. Li, R. Yuan, and Y. Fu. Robot path planning algorithm based on reinforcement learning. *Huazhong Keji Daxue Xuebao (Ziran Kexue Ban)/Journal of Huazhong University of Science and Technology (Natural Science Edition)*, 46(12):65–70, 2018.

[220] Ji Zhang and Sanjiv Singh. Loam: Lidar odometry and mapping in real-time. In *Robotics: Science and Systems*, volume 2, 2014.

[221] Wenjing Zhang, Songzhi Su, Beizhan Wang, Qingqi Hong, and Li Sun. Local k-nns pattern in omni-direction graph convolution neural network for 3d point clouds. *Neurocomputing*, 413:487–498, 2020.

[222] Y. Zhang, L. Chen, Z. XuanYuan, and W. Tian. Three-dimensional cooperative mapping for connected and automated vehicles. *IEEE Transactions on Industrial Electronics*, 67(8):6649–6658, 2020.

[223] Z. Zhang, X. Zhang, Y. Sun, and P. Zhang. Road centerline extraction from very-high-resolution aerial image and lidar data based on road connectivity. *Remote Sensing*, 10(8), 2018.

[224] Zhiyuan Zhang, Yuchao Dai, and Jiadai Sun. Deep learning based point cloud registration: an overview. *Virtual Reality & Intelligent Hardware*, 2(3):222–246, 2020. 3D Visual Processing and Reconstruction Special Issue.

[225] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6230–6239, 2017.

[226] Z. Zhao, W. Zhang, J. Gu, J. Yang, and K. Huang. Lidar mapping optimization based on lightweight semantic segmentation. *IEEE Transactions on Intelligent Vehicles*, 4(3):353–362, 2019.

[227] Y. Zheng and Q. Weng. Model-driven reconstruction of 3-d buildings using lidar data. *IEEE Geoscience and Remote Sensing Letters*, 12(7):1541–1545, July 2015.

[228] Qian-Yi Zhou and Ulrich Neumann. Complete residential urban area reconstruction from dense aerial lidar point clouds. *Graphical Models*, 75(3):118–125, 2013. Computational Visual Media Conference 2012.

[229] Zixiang Zhou and Jie Gong. Automated residential building detection from airborne lidar data with deep neural networks. *Advanced Engineering Informatics*, 36:229–241, 2018.