UNIVERSITÀ DEGLI STUDI DI CATANIA
Dipartimento di Matematica e Informatica
Dottorato di Ricerca in Matematica e Informatica XXXIV Ciclo

Giuseppe Puglisi

Spatial understanding and localization in indoor environments

Tesi di Dottorato di Ricerca

Tutor: Prof. Giovanni Maria Farinella

Academic year 2019/2020

"You can be anything you want to be, just turn yourself into anything you think that you could ever be."

F. Bulsara

Abstract

Every day humans use artificial intelligence in their work or for a better lifestyle. An intelligent system, to carry out these purposes, must have behaviors and characteristics typical of human beings, for better cooperation in an indoor environment. A skill that helps this cooperation is given by the understanding of space and intrinsically also that of knowing how to locate oneself. This understanding of space becomes fundamental in the ability to move in a space full of objects and furnishing accessories, thus being an increasingly difficult task. Some intelligent systems may have vitally important functions, such as saving people in dangerous situations. In this thesis we will use deep learning and computer vision, in particular we will exploit the first person vision for intelligent systems to help a user achieve certain goals in their work or daily life experience. We will also focus on the spatial understanding task and the localization task in indoor environments, more specifically the use of first-person vision in understanding space, and we will use radio wave signals to locate a certain object in indoor environment. These results that we will show in this thesis are the fruit of collaboration between the University of Catania and Essex University for the spatial understanding and between the University of Catania and the OrangeDev company for the localization of an object such as a BLE beacon.

Acknowledgements

First of all, I would to thank my tutors Professor Giovanni Maria Farinella and Professor Antonino Furnari. Without their professionalism and guidance, I would have not been able to achieve any results obtained in these three years. They have always pushed me to get better results, making me grow professionally and humanly, giving me advice, making me increase the necessary skills in research. I would also like to thank very much the professor Gallo, professor Battiato and professor Stanco. They founded the Image processing laboratory (IPLab), that gave me an ongoing support during these years. I would like to thank the staff and the team involved in IPLab, which is reflected in the people of Francesco, Daniele, Santi, Marco, Giovanna, Rosario and Irene. All of them made me feel part of a big family from the first day. These three years were not going to be equally special without my colleagues: Giuseppe, Paolo, Fabio, Federico Santoro, Federico Savasta and Diego. I shared with them many professional and life-experiences in this three years. I would like to thank professor Dimitri Ognibene, my tutor in my period abroad, where i spent five fantastic months in Essex University, that gave me support, pushed me to better results and with an awesome enthusiasm, he educated me on many topics related to my research. Thanks to my parents for everything they have given me all these years. Their love helped me in difficult times, and they helped me support them when they couldn't be independent. Their disappearance, in these tragic years, marked me deeply, on the other hand it made me understand that there is nothing better than family. I also thank my sisters Agata and Barbara who supported, helped and pushed me forward in this three year journey.

Contents

Chapter 1

Introduction

1.1 Introduction

The ability to navigate an environment is essential for any autonomous intelligent system. Mobile robots are a research area that deals with the control of autonomous vehicles with or without an internal map. A robot has many characteristics and abilities and one of them is the ability to learn from a space that is partially observable by a sensor and must take actions to observe its environment and react to unexpected things. After receiving a new observation, a robot must update its knowledge of the surrounding world and calculate a new action plan. Other skills, inherent to its purposes, are the incremental acquisition of knowledge, the ability to recognize objects or places and to be able to give signals in real time in contrast to accidental events in a given internal space. In particular, moving today in an unknown environment is tiring and characterizes both people and intelligent systems. Thus the understanding of space and its awareness of the surrounding environment acquire greater importance in the scientific community. Becoming aware of the objects, of the furniture, of the space available where a mobile system can move, are objectives that a mobile robot must have. Beyond the dimension of space, one of the nuclei of a shift is the task of localization, because localization is the first step for the most difficult research fields. In dangerous situations, such as indoors, it becomes an important skill for an autonomous vehicle to locate and be aware of the surrounding space, because it can help guide a user on how to find an exit or put people to safety. Today, the understanding of space is at the heart of research, as it encompasses different fields of the scientific community. Spatial understanding, in particular, is the primary knowledge of research in navigation activities, for example in jobs where it

is as important as finding an exit for a user in an unknown indoor environment.

1.2 Motivation

One of the major research challenges involving robotics and artificial intelligence lies in creating robots that have to cooperate with human-populated environments, such as indoor environments. Robots, in fact, are progressively moving from industrial domains to our homes and offices, to be used, for example, as assistants in everyday life. Thanks to this change, robots will be increasingly used by people with little or no training in robotics, so the automation of intelligent systems helps communication and interaction between robots and humans to become a key issue for these systems. Such robots, therefore, need skills such as awareness and knowledge of space and place, skills that allow them to interact with the world and human beings who live and work in the same environment. By definition, then, spatial understanding is knowing where a robot is in space in relation to objects or other people. To have good spatial awareness and exploit this ability, an autonomous vehicle must also understand and respond to a change in position by these objects, with the help of image o video in input by sensors. This is a complex skill for an intelligent system, while it is very easy for a human being and develops from an early age. Locating or positioning, on the other hand, is the process of obtaining the position of an object or person with respect to a certain place. In this thesis we localize an object starting from radio wave signals. Understanding of space and location complement each other for this purpose, and these activities promote cooperation between intelligent systems and people.

1.3 Use of computer vision for space understanding and localization

As we have just stated, the location task is the basis for a movement of a mobile robot. A robot of this type is usually equipped with a camera, thus having a partial observation of an environment available. In this situation, computer vision becomes of fundamental importance for the spatial understanding and positioning of

an intelligent system. For the spatial understanding task a robot will face challenges such as:

- reconstruct a 3D model of the environment with an incremental knowledge
- localize itself
- reach a certain point in the environment exploiting the available space
- obstacles avoidance if there are obstacles in the path

To reconstruct a 3D model of the environment we use the computer vision, in particular an instrument that make use of the epipolar geometry, such as Structure from Motion.

Structure from Motion (which its acronym is SfM) is a photogrammetric range imaging technique for estimating three-dimensional structures from unsorted set of two-dimensional image sequences taken from a camera or a set of frames from a video. After the reconstruction a robot must be have the ability of localize itself. The localization task make use of geometry-based or deep learning-based solution to predict the pose of six Degree of freedom (DoF). **Six degrees of freedom** refers to the freedom of movement of a rigid body in three-dimensional space. Specifically, the body is free to change position as up/down, backward/forward, left/right translation in three perpendicular axes, combined with changes in orientation through rotation about three perpendicular axes, often termed yaw (normal axis), pitch (transverse axis), and roll (longitudinal axis). And finally the ability to reach a certain point in the environment, thus having the ability to navigate starting from the images of a camera, and to recognize obstacles that must be avoided. These challenges are not straightforward and implies the use of computer vision.

1.4 Problem Definition

This thesis is going to address the problems of spatial understanding and the localization task, where the last one exploits as input the radio wave signals. We can define spatial understanding as the problem that refers to the ability to be aware of objects and space, in such a way as to exploit this knowledge for other tasks. Spatial understanding, therefore, can give awareness of the surrounding space. This

capability for an intelligent spatial understanding system is important for several reasons, such as:

- **Location:** Spatial understanding gives context to the location of an object, such as a fire extinguisher,
- **Movement:** knowing spatial awareness, a mobile robot can move in respect of how people and objects move through the environment. This can help you navigate in an unknown environments,
- **geometry:** geometry of the spatial is very important for applications such as Navigation task.

In addition, positioning, as well as spatial understanding, acquires a very important role in modern society, as tracking objects or people inside is very useful. In this thesis we therefore address this research field by exploiting radio wave signals such as RSSI, which stands for Received Signal Strength Indication, to estimate the position of a BLE Beacon. More specifically we considered the following tasks:

- Navigation: we try to find an exit from an indoor environments, starting from an input image,
- Localization: positioning of a Beacon in an indoor environment starting from a sequence of RSSI radio wave signals.

1.5 Contribution

In this thesis we address the aforementioned problems using Computer Vision and Deep Learning based solutions. In particular for the task of navigation we discuss about find an exit from any building, investigating in methods that exploit videos or images captured in first person vision. The task of locate a beacon, instead, use radio wave signals such as RSSI and use Deep Learning-base solution to better cope the input signals' noise. We propose a pipeline which uses an autonomous robot to collect RSSI-image pairs and Structure from Motion to associate 2D positions to the RSSI values based on the inferred position of each image. This method, as we shown in the paper, allows to acquire large quantities of data in an inexpensive way.

Using the collected data, we experiment with machine learning models based on RNNs and propose an optimized model composed of a set of LSTMs that specialize on the RSSI observations coming from different antennas. This localization work has been published in paper [1] introduced in Sigmap 2022, the 19^{th} International Conference on Signal Processing and Multimedia Applications.

1.6 Outline

The thesis is divided into 5 chapters. Each chapter treats a specific aspect of the investigated topics.

- **Chapter** 1 explains the motivation and the investigated arguments in this thesis,
- **Chapter** 2 expounds the state of art, the literature related to the topics addressed in this thesis,
- **Chapter** 3 illustrates the topic of the spatial understanding and the relative work,
- **Chapter** 4 investigates the localization task in an environment, exploiting the radio waves signal,
- **Chapter** 5: Conclusion of these works and the next challenges.

Chapter 2

Related works

In this chapter we will discuss the literature related to the two main lines of this thesis: spatial understanding and the localization task. Spatial understanding is related to the navigational task, awareness of the surrounding space and datasets suitable for this type of activity, so we will illustrate the works that follow these branches of research. In particular, we will illustrate the relative works on indoor datasets for a scene understanding task. We will also illustrate the work related to the localization of BLE beacons that use radio signals and to the collection of data sets suitable for positioning. Specifically, the works that will deal with these topics in scenarios such as interior environments and the way in which these two main lines interact with each other will be discussed. We will also discuss the literature related to transfer learning, illustrating related works on the creation of networks used in the Large Scale Visual Recognition Challenge (ILSVRC).

2.1 Spatial understanding

Spatial understanding is a field of research with a large interest in this recent years, for instance just think of a simple navigation or object interaction task that have been faced many times in research. Also, it involves several activities like crowd counting, context awareness, scene understanding and so on. In this section we are going to depict the relative works which involves the space awareness and understanding, and image-based datasets found in the literature with their goal for which they were made.

2.1.1 Spatial understanding and relative works

One of the paper which involves a task such as crowd counting is [2], which present a novel architecture called SPatial Awareness Network (SPANet) to incorporate spatial context for crowd counting. Furthermore, the paper propose a loss called Maximum Pixel Excess Loss (MEP) to finding the sub-region at the pixel level from the ground truth. In recent years deep learning has benefited medical research, especially on medical image analysis, in fact, in paper [3] a self supervised method used the space understanding from the images to have the structural space of the organs. Navigation task is an embodied area of space understanding, paper [4] proposes a Habitat simulator-based tool that leverages real-world images of the environment, to produce more realistic navigational episodes. Although navigation is an important topic in research, spatial understanding also has implications in robotics. In the article [5] the authors describe a logic and framework for the development of spatial understanding in a robotic platform, using the representation of knowledge in the form of a hybrid space-ontological model of the physical world. In addition, a 2007 paper [6] tackle the problem of spatial understanding of human-made environments. In particular the authors presented an approach to create conceptual representations that support situated interaction and spatial understanding.

2.1.2 Image-based spatial understanding datasets

Many of the datasets concern the activity of understanding the scene, using both real and synthetic images as inputs. Most of the dataset contains depth images, due to the high signals given from the deepness that is the good expression of the space understanding. These dataset, also the depth images, use synthetic and semantic pictures.

- In paper [7] has been introduced a dataset called DIODE(Dense Indoor/Outdoor DEpth). The dataset is composited by thousands of diverse, high-resolution color images with their dense, long-range depth mensuration. The RGB-D images represent two type of scenarios, the indoor and outdoor environments.
- In [8], the goal of paper is to analyze typical, often messy, interior scenes in walls, support surfaces, and object regions and retrieve supporting relationships, in such a way as to understand the surrounding space The dataset

contains 1449 RGB-D images, capturing 464 different indoor scenarios, with detailed annotations. The data was collected by a Kinect RGBD camera, which provides noisy depth results.

- Paper [9] introduce a RGB-D video database, called SUN3D, with the joining of hand-labelling and Structure from motion for reconstruction the scenario for scene understanding. In paper the authors also present a a tool for improve the structure from motion with an object labelling such as a generalization of bundle adjustment that incorporates object-to-object correspondences. And a tool for transferring the labels to other frames of partial reconstruction.

- In the article [10] the authors present a large-scale RGB-D dataset containing 10800 panoramic views from 194400 of 90 scenes. The labels are provided with camera poses, and 2D and 3D semantic segmentations. This large dataset enable a large variety of computer vision tasks, including scene understanding.

- Paper [11] introduce an indoor dataset title ScanNet, an RGB-D video dataset composited by $2.5M$ views in 1513 scenes annotated with 3D camera poses, surface reconstructions, and semantic segmentations.

- In paper [12] the authors present SceneNN, an RGB-D scene dataset consisting of more 100 scenes which are reconstructed into triangle meshes. the authors also enriched the dataset with fine-grained information such as axis-aligned bounding boxes, oriented bounding boxes and object poses.

- The authors of paper [13] presented a new method for semantic analysis of 3D points of an entire building using a hierarchical approach. They also introduced, for evaluation a new dataset of several buildings with a covered area of over $6000m^2$ and over 215 million points, demonstrating robust results.

- In paper [14] the authors presented a dataset called Replica, a RGB-D database of 18 highly photo-realistic 3D indoor scene reconstructions at room and building scale with HDR textures and renderable reflectors. Replica's purpose is to improve research in fields such as egocentric computer vision, semantic segmentation in 2D and 3D, geometric inference and virtual robots that navigate, follow instructions and answer questions, based on the space and geometrical relationships of the indoor scenario.

- Paper [15] the authors focused on semantic scene completion, a task for producing a complete 3D representation of volumetric occupancy from a single depth map observation. To evaluate their results they construct a manually large scale dataset of synthetic 3D scenes.
- SceneNet [16] is a synthetic dataset composed by 57 scenes and 3699 object instances created for the scene understanding using the synthetic data.
- Paper [17] proposed Gibson, a virtual environment with a database of spaces includes 572 full buildings composed of 1447 floors covering a total area of $211000m^2$. Each space has a set of RGB panoramas with global camera poses and reconstructed 3D meshes

2.2 Localization task and relative works

To tackle the problem of localization task, different technologies have been used so far: GPS, radio-wave signals, laser ranging scanners, and cameras. We can summarize two different localization task,

- Outdoor localization
- Indoor localization

Although outdoor localization task is already faced and solved with a slightest error thanks to GPS (in the outdoor environment, Global Positioning System is the most famous localization technology which has reached the accuracy of 5 meters), the positioning in an indoor environment of an object remains a demanding task and it is challenging in research. We focuses on localizing a BLE beacon in movement, which is a tough task due to Bluetooth and / or WiFi nature, because radio wave signals are affected by fluctuations. This is the case of Received Signal Strength Indicator: RSSI. A lot of works addressed this problem, using RSSI as input for predict target positions, exploiting the Recurrent Neural Networks (RNN) to make the most of the nature variability of the signal, or using the beacon to calculate the probability distance or the geometric equations to arrive to the target values. With the development of deep learning, fingerprinting-based indoor localization has become a popular method of collecting radio signal waves in many research areas

and it can be seen as a combination of functions. In recent papers various advanced fingerprint-based localization methods have been proposed, which mainly lead to two categories of innovation: the first creates new hardware systems that provide some new functionalities of the mobile target and the second creates some efficient algorithms to improve the accuracy and reliability of location positioning. In this section, not only we show the papers that using fingerprinting, but we show the ones that using different radio wave signals or different technologies to tackle the localization task in an indoor scenario. Furthermore, we describe the most interesting articles related to this task in order to illustrate and facilitate the understanding of the methods and experiments performed in the following chapters.

2.2.1 Localization using radio signals and machine learning-based solution

Localization task is tackled in several various works, for its importance and as basis for other tasks (such as navigation).

- Paper [18] proposed a survey on RNNs for a fingerprinting indoor localization using RSSI. The key idea of the authors is to use RNNs to predict a user's trajectory for a certain period of time, in order to calculate the user's final target position.

- In the article [19] the authors propose a new indoor localization technique that uses magnetometer sensor readings as input for artificial neural network models. Instead of using the RSSI as input, in this work it is interesting how a RNN can fit to the localization problem using a magnetic signal, different from the radio one. The idea of this work, in fact, is to create a pattern of the magnetic field coming from the movement of an object or person over time, in such a way as to use a RNN. Therefore, the use of RNN is not random, as it can characterize a particular position based on the current input and the past input sequence.

- In paper [20] has been used the fingerprinting to collecting RSSIs data. With the RSSIs data the authors developed a decentralized LSTM method to improve the indoor localization task with a regression between the fingerprint

and the positions. Thanks to the decentralization of the LSTM with different recurrent units, they decreasing the time of fingerprinting to create an almost real-time method to localize a position.

- Article [21] not only offer a comprehensive analysis of the state of the art on indoor wireless tracking, but it focuses on the pint of view of devices. In this article, the authors review recent advances in both modalities by elaborating the underlying wireless modes, basic localization principles, and data fusion techniques, with particular emphasis on
  - exploiting smartphones to integrate wireless and sensor functions and extend them to the social context for device-based location.
  - The extraction of specific wireless functionalities to activate a new localization without human-centered devices.

  The authors compare these technologies in terms of accuracy, cost, scalability and energy efficiency.

- Paper [22] is very similar to the previous, the authors aim to provide a detailed investigation of different indoor tracking techniques, such as flight return time (RTOF), time of flight (ToF), angle of arrival (AoA) and intensity indication. received signal (RSSI). The document discusses localization and positioning of human users and their devices, highlighting the strengths of existing systems proposed at the state of the art. Rather than comparing technologies or techniques, the authors compare localization systems and summarize their principle of operation. They also discuss the remaining challenges for accurate internal position.

- In paper [23] the purposes of the authors are to evaluate the performances of Recurrent Neural Network (Elman's RNN, where you can find more information in 4.7.4) as a deep learning technique to be implemented in this area of research. Furthermore they implemented a variant of RNN, a LSTM to perform the indoor positioning. The purpose of their implementation is to explore both LSTM and Elman's RNN to be utilized for localization in indoor positioning scheme, especially for Wi-fi Fingerprinting Dataset.Their results show that proposed approach could get 99.7% accuracy for predicting which

floor the sensor belongs. Their method has a distance errors of around 2.5 - 2.7 meters.

- In the article [24] the authors investigated the problem of indoor location using an algorithm called weighted center of gravity location, based on the RSSI of the closest beacons relative to the weight of the beacons to be located. The key idea is to use a triangulation and trilaterion using a weighted average between the closest beacons, using as input the RSSI, both filtered with the Kalman filter and with raw radio waves.

- Paper [25] introduce an early stage to an indoor positioning-solution, based on Bluetooth Low Energy (BLE) beacons and RSSI fingerprinting, using an algorithm such as K-Nearest-Neighbour, getting an euclidean distance error of 5 and 6 meters. They evaluate their method in a room that has a size of about fourteen meters long, seven meters width, and three meters and half height. The measurement points were arranged in a fingerprint in matrix with a distance of one meter between each other.

- Paper [26] introduces an improvement of the localization task, using a combination of beacons, fingerprinting for RSSI and Wi-Fi access point as input for their method, obtaining a best accuracy on localization of $0,77 meter$, showing theat the Wi-Fi information can improve the result on this task.

- Paper [27] improves localization and consequently the navigation activity with two innovations:

  - BoW algorithm using the visual information of a probe image for speed up localization
  - innovative structure from motion for improves a real - time navigation.

  These innovations help a person which navigate in an indoor environment with a real-time navigation app.

2.2.2 RSSI-based localization datasets

Our research is related to locating a beacon by radio waves as RSSI signals, where our work is presented in chapter 4. Therefore, in this paragraph, we focus on previous works where datasets composed by RSSIs were presented. Paper [28] introduced one of the most popular dataset of RSSI for positioning targets, a multi-building and multi-floor localization database based on WLAN fingerprinting. It is composited by 21049 records, with 520 RSSI's levels. This is the largest and the first publicly available database in time whose paper was presented. Article [29] provides WiFi database is composed of 63504 measurements, organized into datasets that resulted from several collection campaigns. The authors solve the two main problems of Wi-Fi fingerprinting, such as the lack of robustness to short and long time signal changes and the difficult reproducibility of the new methods presented in the relevant literature. In paper [30] the authors provides a method to predict a zone position starting from real RSSI datasets collected from a real environment implemented using a Zigbee network. Their proposal for collecting RSSI consists in building the database of n fingerprints, where a fingerprint consists of a vector of RSSI measurements and the region in which they were taken. In paper [31] the authors compare four wireless technologies for indoor localization such as Wi-Fi, Bluetooth low energy, Zigbee and long-range wide-area network. These technologies are compared in terms of localization accuracy and power consumption Furthermore, they collects received signal strength indicator-based dataset for evaluation of their trilateration model for the accuracy of the localization task. Different dataset of RSSi was introduced in paper [32] that consist in a semi-supervised deep reinforcement learning for the localization task. The dataset was created using RSSI readings from an array of 13 iBeacon [1] on the first floor of Waldo Library, Western Michigan University. The data was collected using iPhone 6S. The dataset contains two secondary datasets: a labeled dataset (1420 instances) and an unlabeled dataset (5191 instances).

2.2.3 Image-based localization datasets

Our research, not only is concerned to the RSSI-based dataset, is also related to previous works focusing on the creation of image-based datasets ofr localization. An

[1] https://developer.apple.com/ibeacon/

indoor dataset covering an entire floor of a building with a total area of $5000m^2$ is described in [33]. The dataset was collected using a mobile system equipped with three laser rangefinders and six cameras, and contains 1095 high-resolution images. The authors of [34] collected a dataset for indoor localization in a shopping mall covering an area of $5575m^2$ using DSLR cameras for the train set, while the test set consists of 2000 photos collected with mobile phones by different users. To estimate camera poses, the authors used a 3D-2D fit algorithm based on a 3D model obtained with a high-precision LiDAR scanner. LiDAR is the acronym for Laser Imaging Detection and Ranging, it can work indoors or outdoors, managing to capture what surrounds it up to 5 meters through a reading at the photon level and with an operating speed of a few nanoseconds, therefore instantaneous. In [35] the Cambridge landmark dataset was introduced. The dataset includes 5 different outdoor scenes and contains approximately 12.000 images tagged with 6 degrees of freedom (6DOF) camera poses. Furthermore, in the paper the authors introduced a robust six degrees of freedom relocation system composed by a an efficient deep 23-layer convnet, called PoseNet, to regress the 6-DOF camera pose from a single RGB image. The system proposed can operate indoors and outdoors, taking 5 milliseconds per frame to compute. In [36] was presented two large Internet photo collections, datasets called Rome16K and Dubrovnik6K respectively. The first dataset is composited of 16,179 and second one has 6,844 outdoor images downloaded from Flickr. The pose information in both datasets was obtained using Structure From Motion [23]. In some cases, especially for indoor environments, the use of dedicated hardware is preferred over Structure from Motion. For example, the 7 scenes dataset was presented in [37], which contains images from seven different indoor environments and includes a total of 43000 labeled frames, was captured using a handheld Kinect RGBD sensor. In paper [38] was presented a dataset of 30 million images, of which nearly two million have been tagged into one of the 500 categories. The dataset and categories are automatically formed from Flickr's geotagged photos. With this dataset the authors create a multi-class support vector machine to classifies the landmarks. Similarly to the aforementioned works, we collect a dataset of images and use structure from motion techniques to attach camera poses to the images. However, differently from those, we aim to create a high-quality dataset of RSSI values associated to ground truth positions. Hence, the collected images are used

mainly as a means to automatically obtain a ground truth signal in our work.

2.3 Transfer learning and models of ILSVRC

Even though a machine learning can model can be of high quality, it can be makes mistakes, because it is used in different scenarios with different its different training data. n these cases it is useful to transfer knowledge between domains for a different task of the various scenarios. In this section will describe the technique of transfer learning, starting from its definition and illustrating its potential.

2.3.1 Definition and methods to use transfer learning

For definition the transfer learning can be summarized in: "Given a source domain D_i and a general learning task T_i, a new domain D_n and a new learning task T_n, *transfer leaning* aims to help improve the learning of the new predictive function F for the new target domain using the knowledge in D_i and T_i, where $D_i \neq D_n$ and $T_i \neq T_n$". A transfer learning process is illustrated in figure 2.1, the process of the left corresponds to a traditional machine learning process while the process on the right corresponds on a transfer learning one. As is depicted in figure, transfer learning not only use the data of the new task domain as input for the new learning algorithm, but also the one of the previous domain, including the training data, the model, and the knowledge acquired. To design a transfer leaning algorithm we need to consider three main issues:

- **What to transfer**: determines which part of knowledge we need to be transferred between the domains and the tasks.
- **When to transfer**: we are interested in what situations applying knowledge transfer turns out to be bad for learning. and consequently a transfer of brute force, in these situations, will not be successful. In some situation when the domains and the task are not relater each other, the transfer learning may hurt on the learning process.
- **How to transfer**: we transfer in different categories:
  - model-based

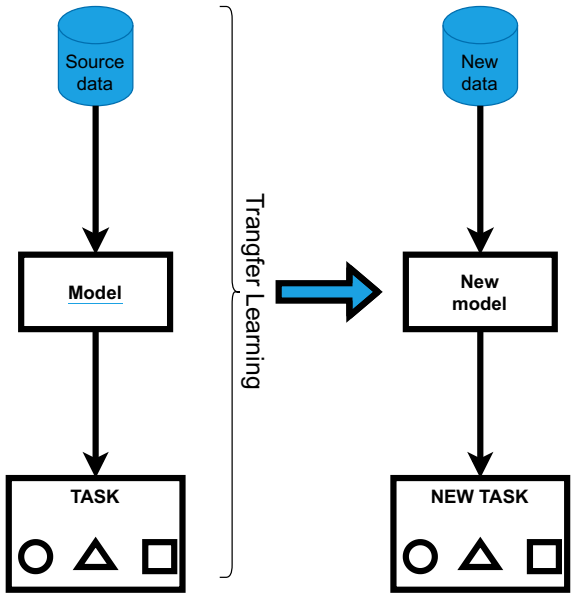


Figure 2.1: Transfer Learning adaptation

- features-based
- relation-based
- instances-based

These categories represent which part of knowledge can be a vehicle to facilitate the new domain/task learning process.

Every year there is a challenge where a dataset called ImageNet is used to evaluate the best convolution neural network, this competition is called ImageNet Large Scale Visual Recognition Challenge (ILSVRC) ImageNet is a images-based dataset composited by more of 14 millions images and more of 21 thousands of classes indexed. In ILSVRC competition a subset of ImageNet in considered for the competition, which 1.2 million of images and 1000 classes to predict. The best CNN is evaluated with two different error metrics:

- Top one: the percentage error on the most probable class given by a neural network convolution trained in the classification task,
- top five: the percentage error on the five most probable classes given by a Convolution Neural Network (CNN) trained in the classification task.

2.3.2 LeNet

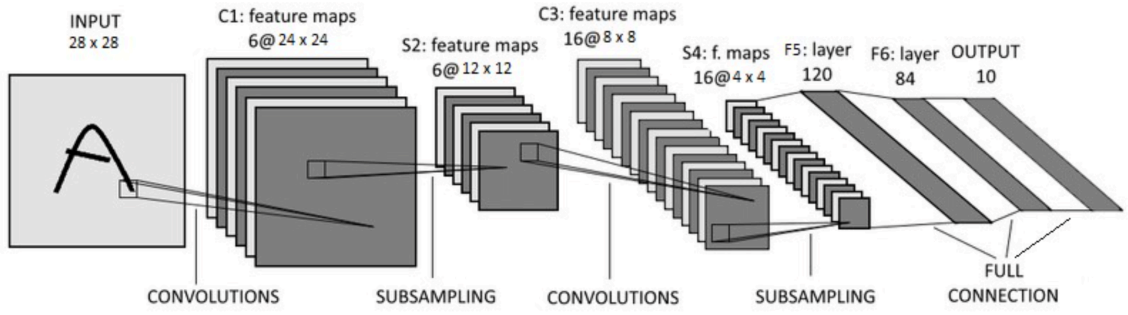


Figure 2.2: First convolutional neural network

LeNet is the first CNN in the literature presented in 1998 by Yann LeCun in the article [39]. The model was proposed to solve the problem of digit classification of the MNIST-DIGITS dataset, this model has seven layers:

- A C1 convolution layer: the layer takes as input an image of size $28x28$ pixels and a single channel. By the application of 6 convolution kernels of $5x5$ size, 6 feature maps are calculated. No type of padding is applied, so each feature map is $24x24$ in size. In practice, the input of layer C1 is a tensor of dimension $1x28x28$, while its output is a tensor of dimension $6x24x24$. Each convolution is done by applying a $5x5$ kernel and adding a bias term to the result. The total number of parameters that can be optimized for level C1 is therefore equal to $6(5x5 + 1) = 156$.

- An S2 sub-sampling layer: the layer takes the 6 $24x24$ maps as input and sub-samples them obtaining in output maps of $12x12$ dimensions. Sub-sampling is performed by dividing the input map into $2x2$ neighborhoods and calculating the average value of each of these neighborhoods (average pooling); in practice, the input of the layer is a $6x24x24$ dimension tensor, while its output is a $6x12x12$ dimension tensor. The layer does not contain any optimizable parameters.

- A C3 convolution layer: the layer takes the $6x12x12$ size tensor as input and computes 16 of $8x8$ size feature maps using 5x5 size kernels. The number of parameters that can be optimized is equal to $6x5x5x16 + 16 = 2416$.

- An S4 sub-sampling layer: in a similar way to layer S2, it takes 16 of $8x8$ maps as input and produces $4x4$ maps. This level also does not contain optimizable parameters;

- A fully connected linear transformation layer F5: the layer takes as input the tensor of dimensions $16x4x4$. The tensor is therefore considered as a one-dimensional vector of 256 units and transformed (by linear transformation) into a tensor of 120 units. The total number of parameters is given by $120x256 + 120 = 30840$.

- A linear transformation layer F6: the layer takes as input the vector of 120 units and transforms it into a vector of 84 units. The number of parameters that can be optimized is equal to $84x120 + 84 = 10164$.

- A linear transformation layer F7: the layer takes as input the vector of 84 units and transforms it into a vector of 10 units (the scores relating to the MNIST-DIGITS classes). The number of level parameters that can be optimized is $84x10 + 10 = 850$.

2.3.3 Alexnet

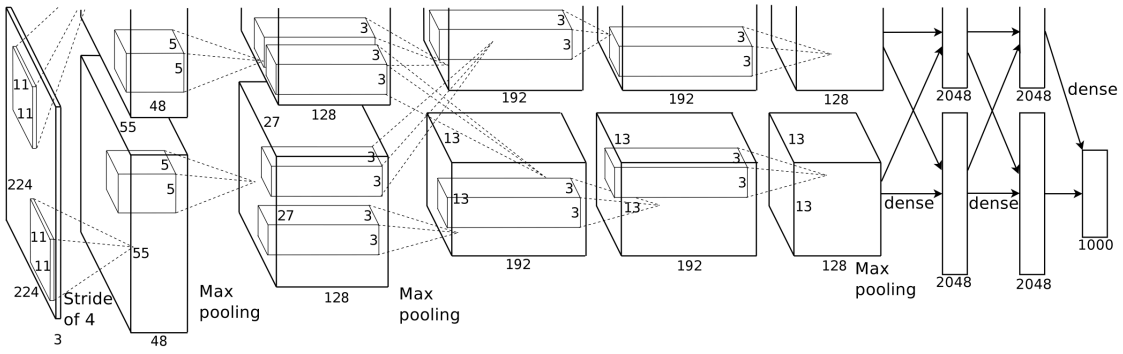


Figure 2.3: AlexNet ILSVRC in 2010

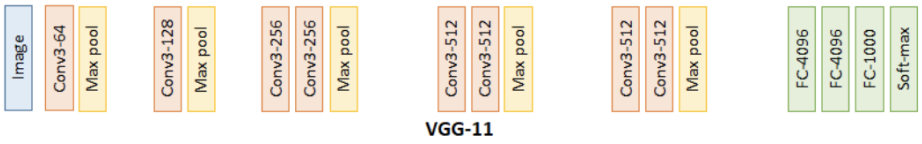
AlexNet entered in the ILSVRC challenge in 2010, achieving the best results by introducing the now famous techniques to reduce overfitting, such as data augmentation and the dropout. As shown in figure 2.3, the net (presented in paper [40]) contains eight layers composited by five convolutional layers and the remaining three

are fully connected. The first convolutional layer filters the input image $3x224x224$ with 96 kernels of size $3x11x11$. The second convolutional layer takes as input the output (normalized and pooled) of the first convolutional layer and filters it with 256 kernels of $48x5x5$ dimensions. The other three convolutional layers are connected together without any intermediate levels such as pooling or normalization. The third convolutional layer has 384 kernels of $256x3x3$ dimensions connected to the outputs of the second convolutional layer. The fourth convolutional layer has 384 kernels of $192x3x3$ size and the fifth convolutional layer has 256 kernels of $192x3x3size$. The fully connected layers have 4096 neurons each. The output of the last fully connected layer is fed to a 1000 way softmax which produces a distribution over 1000 class labels.

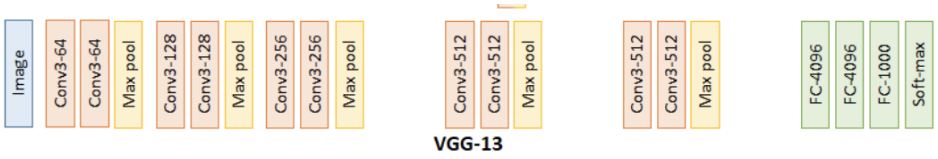
2.3.4 VGG

Convolutional networks (CNNs) enjoyed great success in large-scale image and video recognition until 2014, when VGGs were presented in the ILSVRC competition, illustrated in paper [41]. The VGG networks has shown that a very depth of a CNN can be achieved good performance. In their paper, the authors presented different architectures, differing in depth and results as top 5 errors:

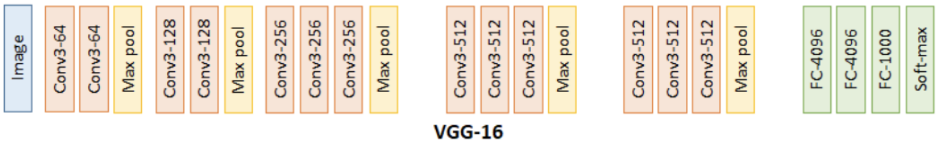
- **VGG-11**: Composited of 11 layers with a top 5 error of 10.4% .



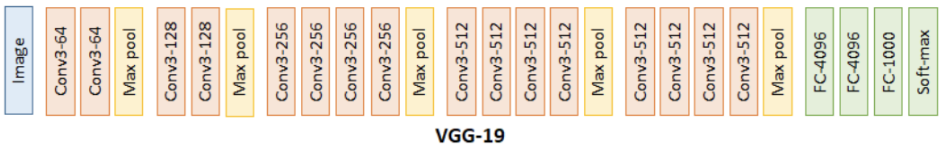
- **VGG-13**: Composited of 13 layers with a top 5 error of 9.9% .



- **VGG-16**: Composited of 16 layers with a top 5 error of 8.8% .



- **VGG-19**: Composited of 19 layers with a top 5 error of 9.0% .



2.3.5 ResNet

In the paper [42] ResNet were presented, engineering and improving convolutional neural networks, and obtaining the best results in the ILSVRC challenge. The article showed that having a deeper and deeper network, it shows no improvement in performance, rather it gets worse, countering the key idea of the VGG paper. One of the key idea is to form the CNN with blocks and if the function of the block remain an identity function, in learning process they skip the block with a **Shortcut Connections**. Furthermore, the ResNet's do not calculate a mapping function between the input and the output as CNN did until then, but they do calculate the residuals (from which they take their name). We consider $\mathcal{H}(x)$ as an underlying mapping to be adapted to a few overlapping levels, with x denoting the inputs at the first of these levels. If we assume that several non-linear levels can asymptotically approximate complicated functions, then it is equivalent to hypothesizing that they can approximate the residual functions asymptotically: $\mathcal{H}(x)-x$; assuming that the input and output are of the same dimensions.So rather than expecting the stacked levels to approximate $\mathcal{H}(x)$, we explicitly let these levels approximate a residual function $\mathcal{F} := \mathcal{H}(x) - x$. The original function thus becomes $\mathcal{F}(x) + x$. The paper also presents different architectures made up of more or less building blocks:

- ResNet-18,
- ResNet-34
- ResNet-50,

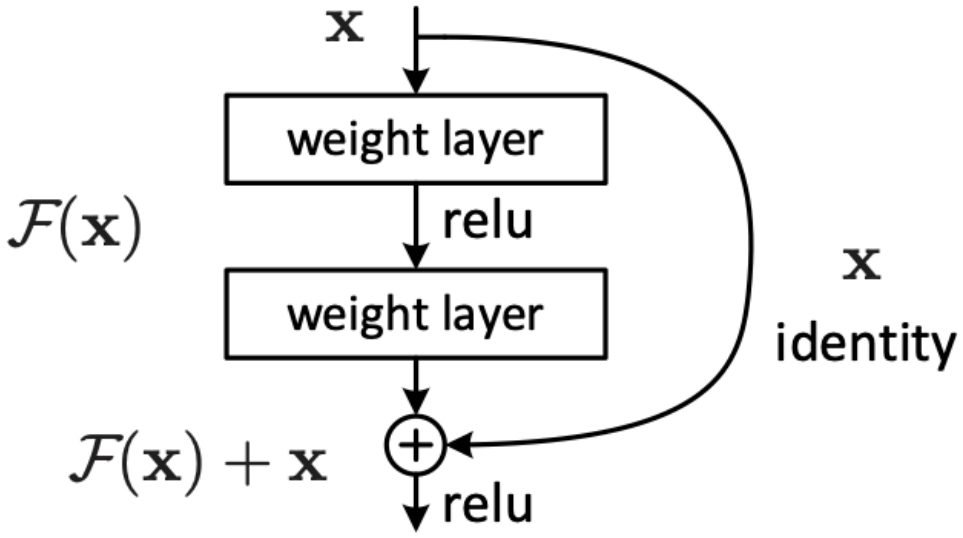


Figure 2.4: Building block that calculate $y = \mathcal{F}(x, \{W_i\}) + x$

- ResNet-101,
- ResNet-152.

The ResNet-152 achieved in the ILSVRC challenge a top-1 error of 21.43% and a top-5 error of 5.71%.

2.3.6 Fully-Convolutional Network model:FCN ResNet50

Fully Convolutional Networks (FCN), introduced in paper [43] are variants of Convolutional Neural Network, and are used typically for tasks such as semantic segmentation. Remember that semantic segmentation is the process that divide a picture in different classes. In following chapter we used a Fully Convolutional Network (FCN) model with a ResNet-50 2.3.5 as backbone,for reconstructing an intermediate representation for a navigation task. In paper [43] the key insight of the authors is to build FCN networks that take input of arbitrary size and produce correspondingly-sized output with efficient inference and learning, typical of semantic segmentation. In the article they define the space of FCN, explaining their application to spatially

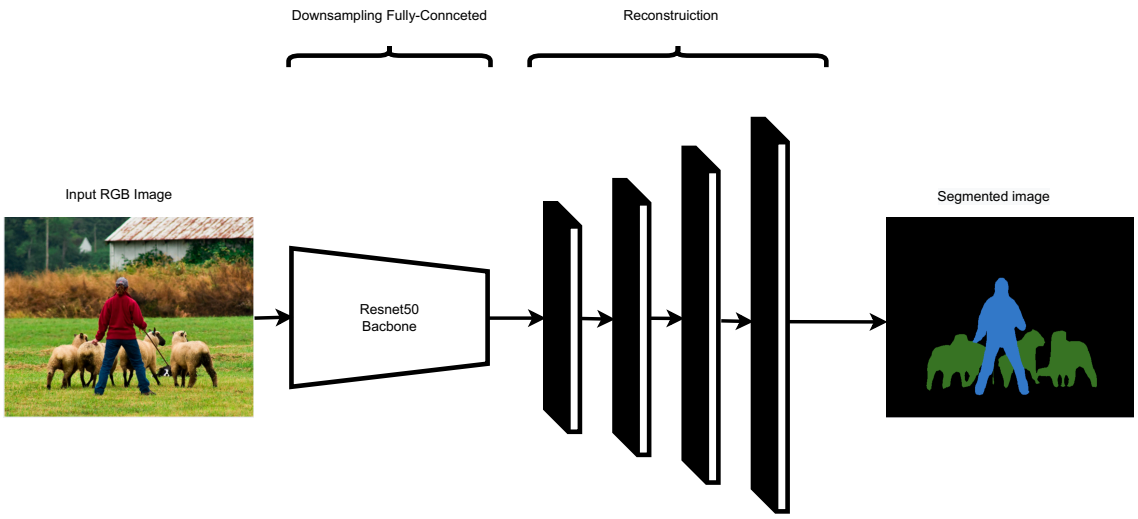


Figure 2.5: FCN with ResNet 50 as backbone

dense prediction tasks, and draw connections to prior models. Paper used as backbones different CNN such as AlexNet 2.3.3, VGG 2.3.4 and GoogleNet. This paper, published in 2014, explains that the CNNs are used in different research ares such as classification of the whole picture, a bounding box for object detection and semantic segmentation with poor results. The next natural step in the progression from coarse to fine inference is to make a prediction at each pixel. The authors that a fully convolutional network (FCN), trained end-to-end, pixels-to-pixels on semantic segmentation overcome the poor results of the state of art. They are the first that trained this type of model in that time.

2.4 Filtering techniques

In the real world, the RSSI values are strongly influenced by the environment and consequently have high noise levels. This noise is, for example, caused by multi-path reflections: signals bounce off objects in the environment such as walls and furniture. Furthermore, the proximity of the environment to an electric radio station generates noise to the signal which can make it more unstable. RSSI fluctuations due to signal noise significantly affect both stability and efficiency. One of the ways to combat noise on the signal is to use a technique such as filtering. These techniques allow to limit the noise of the signal, making it as linear as possible. In this section we

will describe the filters in discrete form, whose mathematical formulation refers to the paper [44]. While we will be using raw RSSI, talking about how noise can be attenuated in signals such as radio waves makes us understand how reducing noise is not an easy practice to achieve. In our work, to limit noise, we will use a solution based on machine learning to tackle the problem of locating an object starting from a "raw" RSSI signal.

2.4.1 Kalman Filter

The most filter technique used to denoising the RSSI signal is the Kalman filter. It is a recursive method that is not suitable in this form for a temporal radio wave signals. To use this technique i transform this filter in a discrete way. The discrete Kalman filter module attempts to estimate the RSSI values by representing the temporal transformation of the RSSI signal as a combination of signal noise (measurement noise) and maximum signal evolution (process noise). A linear stochastic equation models the RSSI evolution, with signal / processing noise assumed to be independent of each other, with white noise and with normal probability distribution. This iterative method try to minimizing the process noise and it is carried out in two phases:

- A predictor performs next RSSI estimation,
- A corrector improves RSSI estimation by exploiting current RSSI measurement.

The formulas for linearity equations, that iterate for each phase are below:

- First phase - Prediction:

$$\hat{x}^{-}(k) = A\hat{x}(k-1) + w(k-1) \tag{2.1}$$

$$P^{-}(k) = AP(k-1)A^{T} + Q \tag{2.2}$$

$$K(k) = P^{-}(k)H^{T}(HP^{-}(k)H^{T} + R)^{-1} \tag{2.3}$$

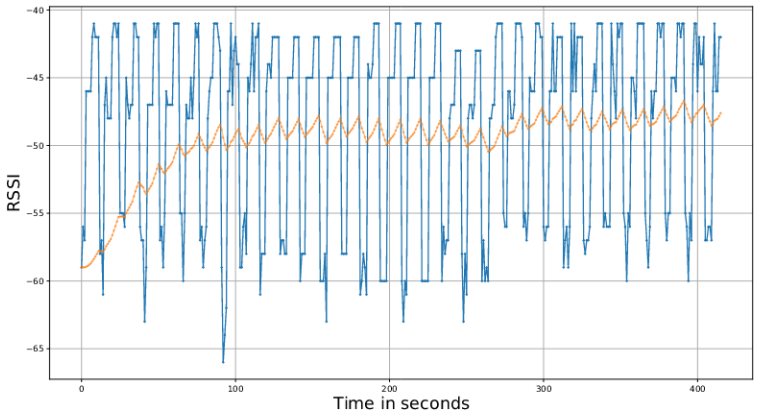


Figure 2.6: Kalman filtering

- Second phase - Update:

$$\hat{x}(k) = \hat{x}^{-}(k) + K(k)(z(k) - H\hat{x}(k)) \tag{2.4}$$

$$P(k) = (1 - K(k)H)P^{-}(k) \tag{2.5}$$

where $P(k)$ is the covariance matrix of the state estimation error at step (k), with initial value Q, and $K(k)$ is usually indicated as the Kalman gain, a matrix that takes values from high error to low one.

Application of Kalman filter

The Kalman filter is not only used to limit noise in the radio signals like RSSI, but its applications in the real world are very different. An example application could provide accurate and continuously updated information on the position and velocity of an object given only a sequence of observations on its position, each of which includes some errors. For a similar and more concrete example, in a radar application, where one is interested in tracking a target, information about the target's position, velocity and acceleration is measured at all times, with significant noise degradation. The Kalman filter exploits the dynamics of the target, which governs its evolution

over time, to remove the effects of noise and obtain a good estimate of the position of the target in the present moment (filtering), in a future time (forecasting), or in a past moment (interpolation or smoothing). Other applications are the weather forecasting, speech enhancement, the economy sphere, autopilot, and so on. As for our experiment, i save seven minutes of information (RSSI) from a given beacon and calculate the filtered signal using the Kalman filter. In the figure 2.6 i plot the raw RSSI signal in blue and the filtered one in orange.

2.4.2 Gray model Filter

The Gray Model filter module calculates filtered RSSI values based on a finite series of relative radio wave values monitored in the recent past. In particular, given a set of actual RSSI values, measured on the client side, $R_0 = \{r_0(i), r_0(i+1)., r_0(n)\}$ where $r_o(i)$ is the value of RSSI at time i and n is the total number of the set of RSSI to filter, it is possible to calculate $R_1 = \{r_1(i), r_1(i+1)., r_1(n)\}$ where:

$$r_1(i) = \sum_{j=1}^{N} r_0(j) \tag{2.6}$$

From the discrete gray model filter differential equation:

$$\frac{dr_1(i)}{di} + ar_1(i) = u \tag{2.7}$$

we can use the parameters a and u to determinate the discrete value of RSSI $pr(i)$ at time i:

$$pr(i) = \left(r_1(1) - \frac{u}{a}\right) e^{-ak} + \frac{u}{a} \tag{2.8}$$

2.4.3 Fourier Filter

Fourier filtering is a type of filtering function based on the manipulation of specific frequency components of a signal. It works by taking the Fourier transform of the signal, then attenuating or amplifying specific frequencies, and finally reversing the result. In many scientific measurements, such as spectroscopy and chromatography, signals are relatively smooth shapes that can be represented by a surprisingly small

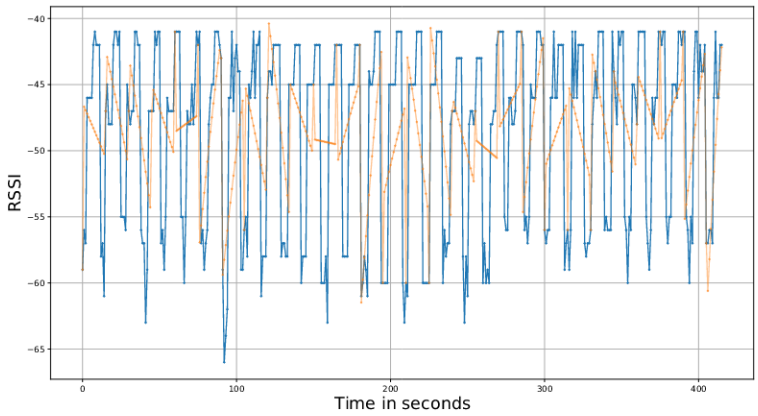


Figure 2.7: Same values used for Kalman, filtered with the Grey model filtering

number of Fourier components. Given a set of RSSI $R_0 = \{r_0(i), r_0(i+1)., r_0(n)\}$, the Discrete Fourier Transform (DFT) calculates the coefficient A_i and B_i representing the RSSI sequence in the frequency domain. The coefficient are calculated with the same Fourier equations:

$$A_0 = \frac{1}{N}\sum_{n=1}^{N} y(t_n) \tag{2.9}$$

$$B_0 = B_{\frac{N}{2}} = 0 \tag{2.10}$$

$$A_{\frac{N}{2}} = \frac{1}{N}\sum_{n=1}^{N} y(t_n)\cos(n\pi) \tag{2.11}$$

$$A_p = \frac{2}{N}\sum_{n=1}^{N} y(t_n)\cos\frac{n\pi pn}{N} \tag{2.12}$$

where $p = 1...\frac{N}{2} - 1$

$$B_p = \frac{2}{N}\sum_{n=1}^{N} y(t_n)\sin\frac{n\pi pn}{N} \tag{2.13}$$

where $p = 1...\frac{N}{2} - 1$

In these formulas $T_n = n\Delta t$ and $\Delta t = \frac{T}{N}$ and $N = R_0 size$

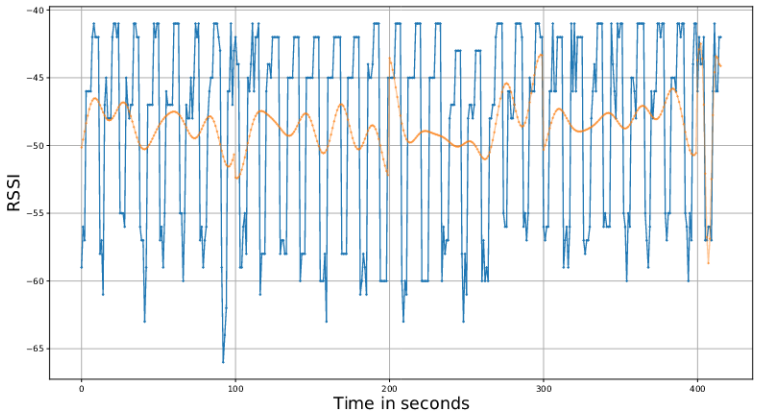


Figure 2.8: Same values used for Kalman, filtered with the Fourier model filtering

The set of coefficients is the basis to regenerating the RSSI with the Inverse Discrete Fourier Transform:

$$f(t_n) = \frac{1}{2}A_0 + \sum_{p=1}^{M}[A_p \cos \omega_p + B_p \sin \omega_p] \tag{2.14}$$

The regenerated RSSI sequence do not show its high frequency components and shows a more regular trend, filtering the data.

2.4.4 Particle Filter

As Discrete Kalman filtering , the Particle one module tries to estimate RSSI by minimizing measurement and process noise, but without imposing a linear equation modeling and without imposing normal distribution for signal noise. The basic idea is to have an algorithm that computes several possible filtered RSSI values for each measured RSSI at each step. The filter associates, then, each candidate value with a weight and chooses the most promising values among them when a new measured RSSI is available; finally, it perturbs candidate values, thus obtaining a new filtered RSSI which is the average value of the most promising candidates. To better understand how this filter works, let us show an example of algorithm iteration with 8 particles, which represents 8 possible filtered RSSI values in figure 2.9. There

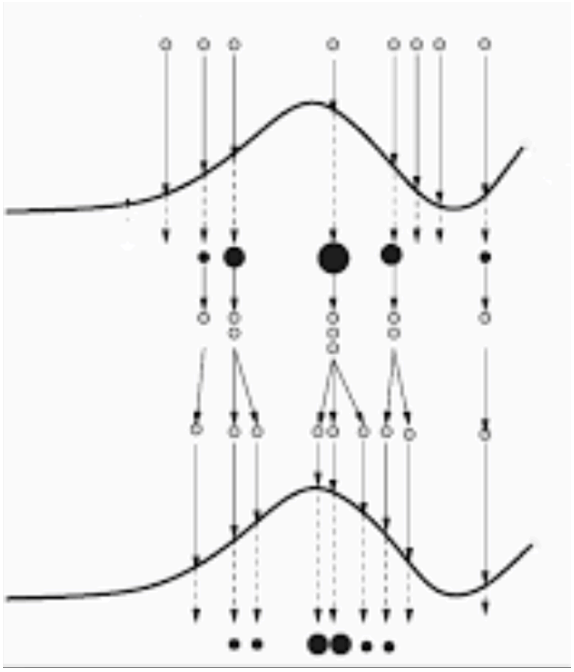


Figure 2.9: Example of particle filter

are 8 possible filtered RSSI values, in starting step, all with the same weight; In the second step, by exploiting state estimate probability (black curve) obtained from RSSI measurement, the algorithm assigns a weight at each filtered RSSI value; in the third step, heavy RSSI are spread in different RSSI values, all with the same weight In fourth step the filtered RSSI are randomly perturbed The number of particles strongly influences the particle filter performance; in general greater is the particle number, better the filtered RSSI follows the actual RSSI sequence.

2.4.5 Moving average filter

In statistics, the moving average of a data set consists of creating a series of averages of various subsets starting from the complete set. Given a series of numbers and a fixed subset size, the first element of the moving average is obtained by taking the average of the initial fixed subset of the number series. Then the subset is modified by "shifting forward"; that is, excluding the first number of the series and including the next value in the subset. A moving average is commonly used

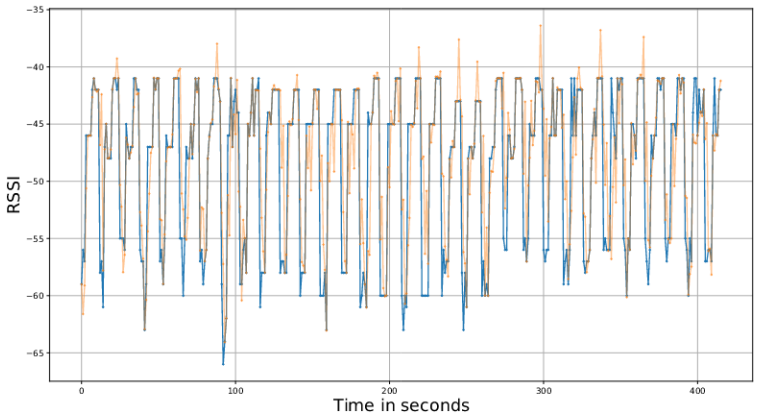


Figure 2.10: Same values used for Kalman, filtered with the Particle model filtering, with a number of 400 particles

with time series data to smooth out short-term fluctuations and highlight longer-term trends or cycles. The threshold between short-term and long-term depends on the application, and the parameters of the moving average will be set accordingly. Mathematically, a moving average is a type of convolution and so it can be viewed as an example of a low-pass filter used in signal processing. When used with non-time series data, a moving average filters higher frequency components without any specific connection to time, although typically some kind of ordering is implied. Viewed simplistically it can be regarded as smoothing the data. In our case this filter is used to smooth the set of RSSI's signals. The filter obviously returns a set of values lower than the starting set, it is due to the size of the window that moves along the entire set of RSSI's. In figure 2.11 it was used a window size of 50 values.

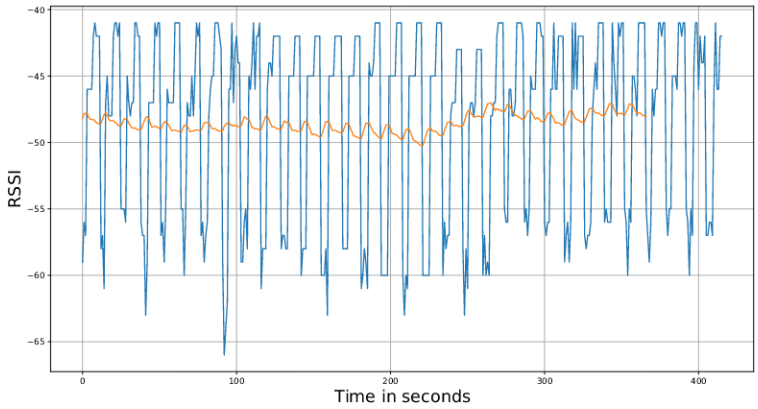


Figure 2.11: Same values used for Kalman, filtered with the moving average model filtering

Chapter 3

Spatial understanding

In recent years, space understanding includes many fields of study, the ability to reconstruct a surrounding space from a single partially observable point of view, today plays an important role in many fields of research. This partial vision of space can derive from two paradigms, which can be considered as sub-fields of computer vision, such as:

- **First person view** (or egocentric vision),
- **Third person view**.

We can define egocentric vision or first person vision (FPV) as a filed of computer vision, which involves images and videos captured continuously by a wearable camera. These cameras are mostly placed on the top of the head or chest. These videos and images captured by wearable devices are perfectly suited as inputs to machine learning-based methods that are trained to solve certain kind of problems. There are many wearable devices on market today, but the most important ones used in research are:

- Microsoft Hololens [1] ™,
- Google Glass [2] ™,
- GoPro [3] ™.

Great progress has been made in computer vision based system, especially the egocentric vision paradigm, has been used in topics such as action anticipation, action

[1] https://www.microsoft.com/en-us/hololens
[2] https://www.google.com/glass/start/
[3] https://gopro.com/it/us/us/

recognition, navigation task and so on; and implies solutions born of deep learning, reinforcement learning or deep reinforcement learning. Learn from video captured from a point of view of an user or a robot is important in many domains (including industrial scenario). Spatial understanding, in fact, according to the first person view, can reconstruct the surrounding environment in 3D taking an ongoing information by the point of view of an user.. Typical of the "Third Person Vision" (TPV) paradigm, instead, are video surveillance cameras, in which the scene is acquired by a static camera that remains neutral with respect to the observed events. We use spatial understanding for a navigation task taking information by a fixed camera in a robot, which it does not remain neutral with respect to the observed events. After gaining awareness of the positions of objects and the space available in an indoor environment, it is possible for an intelligent system to move. This is challenging in research, moving in a difficult environment is the core of the knowledge to enable the navigation activity. An important result of this activity is reflected in finding the exit from any environment, an important task in dangerous situations.

3.1 Egocentric vision paradigm

First-person (egocentric) vision is a relatively new research topic in the field of computer vision, which is attracting more and more interest from the scientific community. In recent years, many papers have used this particular paradigm to make the most of computer vision and solve some important tasks. This paradigm makes large use of wearable devices, ever smaller and with a digital recording with an ever better resolution. This human-centered perspective is naturally suited to gathering visual information about daily observations and interactions, which in turn can reveal the wearer's attention and behavior. Third person vision, instead, taking sight in a fixed point, having a neutral position and maybe can gathering information that is useful only for purposes such as surveillance and tracking, which are not task tackled by this thesis. It necessary explain the main differences between the first person vision and the third person vision. Third person vision uses fixed, non-mobile cameras and the contents are often edited (e.g. collections of YouTube videos or movies, video for surveillance or tracking object or people in movement), in FPV, instead, video is usually ongoing captured and therefore tends to be unstructured

Figure 3.1: Third person video camera fixed and first person person video such as Hololens of Microsoft

and difficult to index, containing a lot of visual information useful for a variety of tasks. The task and research area where first person vision paradigm is used are:

- Attention-based activity task
- Social interaction task
- Multi-agent egocentric vision systems
- Social salience estimation
- Hand pose estimation
- Understanding social dynamics and attention
- Use robotic vision as self-centered sensing
- Activity (actions) anticipation and forecasting

Leveraging the FPV paradigm, the space understanding of egocentric images and videos is advancing rapidly, gaining ground in the scientific community.

3.1.1 Advantages of first person vision

In the FPV paradigm, images are acquired without interruptions by means of a wearable camera that is carried by the user at all times, in contrast to the TPV paradigm the images are acquired by a fixed camera placed in a certain position with respect to the activity considered. This is one of the key differences between FPV and TPV: the mobility of the camera. The contents acquired by the FPV paradigm continuously give us visual information from the point of view of a user (or in the broad sense of the paradigm, from the point of view of a robot), information on the activities he is carrying out, on the workplace or on his interactions with the environment. An example of the advantages of First Person Vision systems is shown in 3.1, which compares the images acquired according to the two paradigms. The advantage of FPV systems lies in the ability to be transported by an user or a robot, observing the world from his perspective. Furthermore, another advantage that should not be underestimated is the continuity of the information acquired which allows the acquisition of large quantities of data.

3.1.2 History of First person vision

The forerunner who came up with the idea of using a wearable camera to collect visual data was Steve Mann; a Canadian engineer and professor working in fields such as augmented reality, computational photography, especially wearable computing and high dynamic range imaging. This professor created "eyeglasses" in the 1970s, a device that, when worn, causes the human eye itself to effectively become both an electronic camera and a television display. Later, in 2006, Microsoft created a neck-worn camera, called the SenseCam, used for health research work. The interest of the computer vision community in the first-person (egocentric) paradigm grew slowly until 2010. Then there has been rapid growth in recent years, driven by both the growing number of potential applications and the massive expansion of wearable technologies. In fact, in 2012, the first egocentric prototype was born, consisting of three components that give the user a complete situation in temporal and spatial information, which can be used as an object of assistance for a human being. Nowadays first person view is used in many fields of research and the community is moving forward in this direction.

3.2 Problem formulation

The goal of this third chapter is to describe the carried out research related to the spatial understanding, which involves the task of how to find an exit from any type of building. This research has a very specific purpose, which is to allow and learn a robotic system to navigate indoors scenarios to find a specific target, i.e the exit. In accord to the FPV, this chapter are going to address the aforementioned problem, using computer vision and deep learning-base methods as possible solutions. First we will describe the pipeline to collect the dataset, then we will tackle the main problem with different methods and possible solutions that are affected by the difficulties in using different types of indoor environments, with a rather common problem in this particular type of activity: the overfitting.

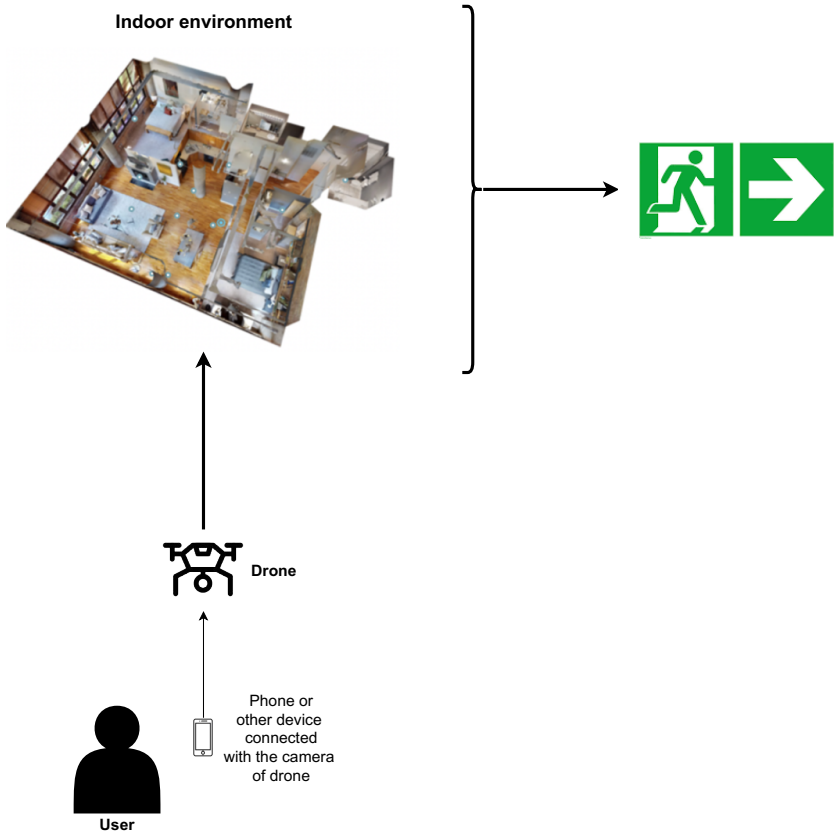


Figure 3.2: Problem formulation

3.3 Dataset

In accord with the goal of research, the dataset contains images representing indoor scenarios, able to provide spatial and geometric information for the type of movement that the robotic system will have to perform.

This dataset is the result of the cooperation between the mathematics and computer science department of the university of Catania and the engineering and computer science department of Essex university (United Kingdom). In particular the IPLab of the mathematics and computer science department of the university of Catania, with my tutor, professor Giovanni M. Farinella (one of the founder of the laboratory), and the professor Dimitri Ognibene (professor of Essex university).

The dataset is created with unity. We load a map of Stanford university and we placed some point of interest all around the map. After we moved a virtual camera on the whole map and for any point of view (Field Of View - FOV) we take a picture of the environment. The dataset describes for each point the action performed by the camera to reach the relative FOV, in terms of Delta X, Delta Y and delta angle (which describe the rotation of a drone after a movement). The camera takes also all the images of the nine different orientations. The movement along the axes is in meter and recorded as absolute coordinates and the rotation is in a range from 0 to 360 degrees, with a step size of 45. The brightness, the furnishings and furniture are all objects created with Unity 3D.

3.3.1 Images-based dataset for indoor environment

We hence describe all the images and information present in the dataset. It is made up of as many folders as there are points located on the unity map. For each folder we have different images and other sub-folders, one for each frontier point. In these

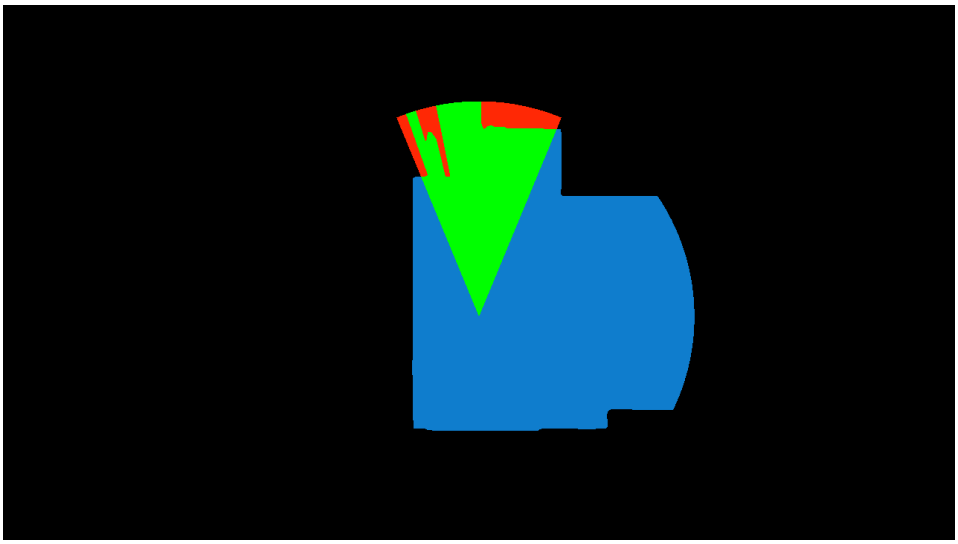
Figure 3.3: Unity logo

sub-folders we have different Json files that describe the spatial information for each rotation. The images present for every point are:

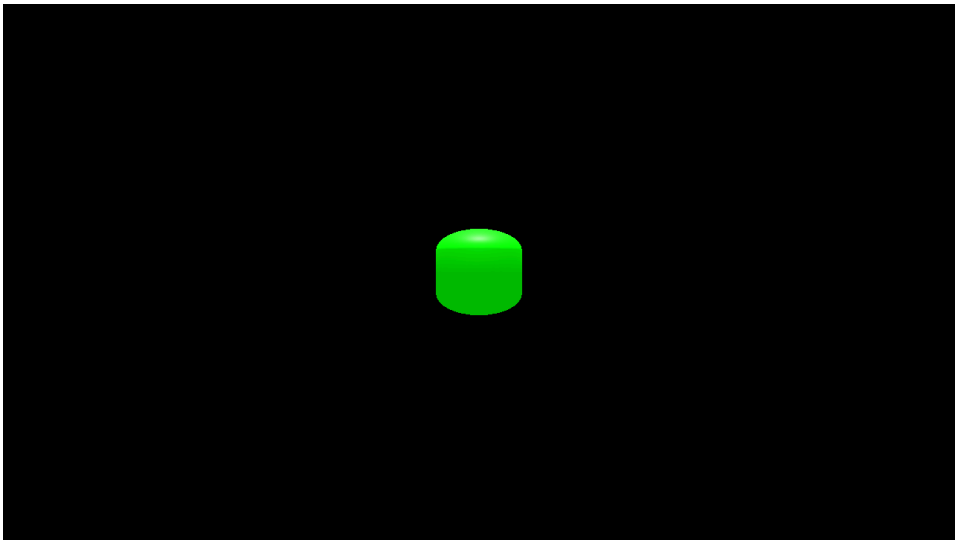
- A **starting image** the FoV where start an action which a drone will perform. This image is the typical input for a baseline, in fact we will use all these images as input for our experiments, to predict the movement and the rotation which the drone will perform to arrive at the destination image.

- One of the **Destination image** reachable from the starting image after a certain movement and rotation.

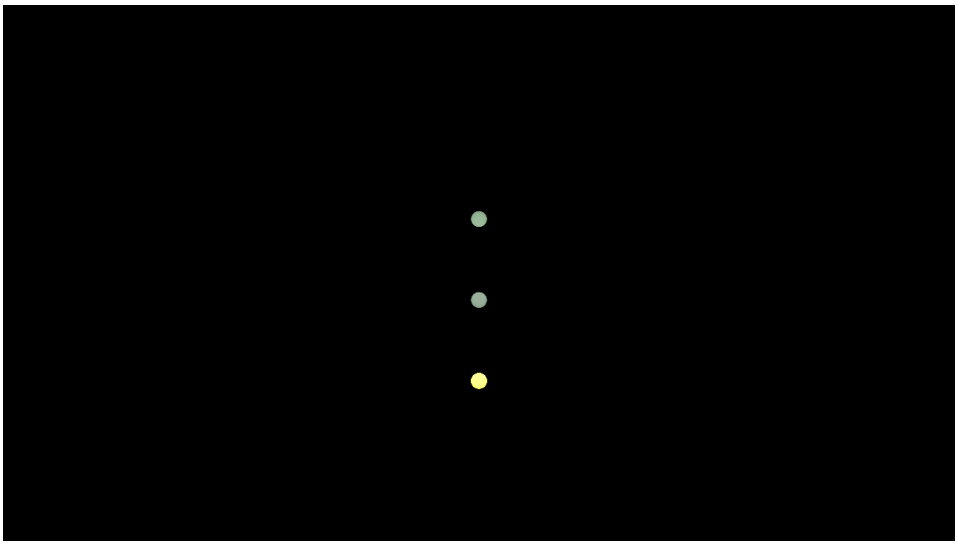
- **Top-view image**, the image represents different type of navigation from top view:
  - the green color represents the space in which the drone can move
  - the blue color represents the available space
  - the red color represents the obstacles



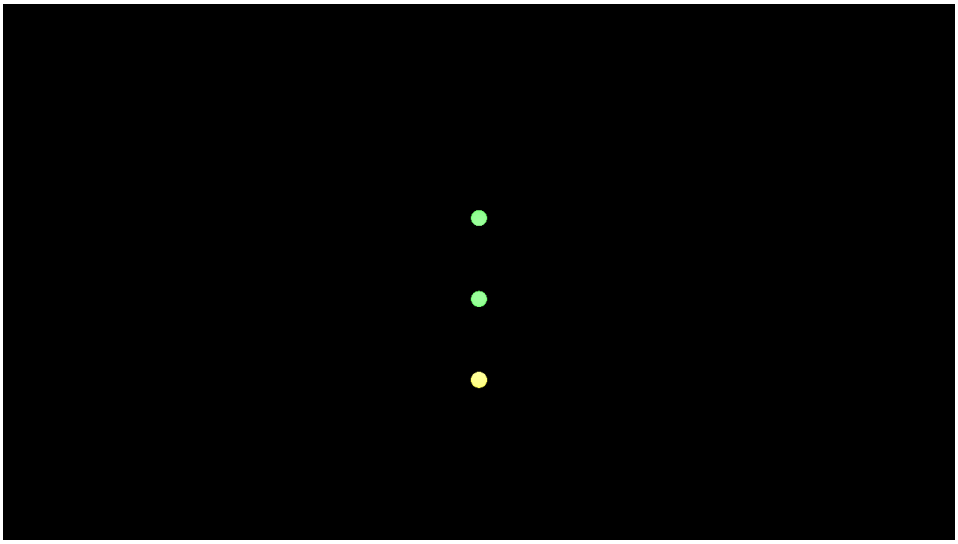
- The image with a **cuboid** that with its shape and position represents the movement of action which the drone has to perform



- Image which with its small dots with their color and **intensity** describe the movement to perform



- Image which all **reachable points** which represents the available space.



3.3.2 Dataset statistics and data preparation

The whole dataset is very large with 2305 points in the different buildings of the Stanford map. From any point we can see 9 points of view (in accord with the orientation), so we created 2 types of comma separated values files (CSV) for extract information, the first one is composited with the best action from a starting point, so we can using the starting image as input of a machine learning-based architecture. The best action is selected with the criteria which the destination point is able to let me see more points, so, the action is composited with the best displacement and rotation. This criterion is due to the fact that i can create a path that allows me to reach the exit. In this CSV we have 6809 pairs of point-action. The second CSV is composed of the whole set of possible actions with an amount of 102736 records. Both CSV contains the three target values for each record:

1. The movement along the two axes X and Z,

2. the Orientation as Delta Angle, included in a range between 180° to 180°, and also transformed in the vectors U and V.

The movement is calculated from the starting coordinates (absolute X and Z) and the final coordinates. Since the coordinates are in the absolute form, we calculated them in a relative form, with the matrix of rotation for delete the angle perspective

of the tow images (the starting one and the final one). After this we have to calculate only the differences. Both CSV have the same columns described in following table

Columns dataset	
Name column	**Description**
Starting image	Path of the starting image, this image is the input of the experiments
Index of FoV	Index of Point of View of starting image
Angle of starting image	Orientation of the starting image
World coordinates X	Absolutes coordinate for X axes of starting image
World coordinates Z	Absolutes coordinate for Z axes of starting image
Destination Image	Image of arrive after a displacement and rotation of a drone
Index of FoV	Index of Point of View of destination image
Angle of destination image	Orientation of the destination image
World coordinates X	Absolute coordinate for X axes of destination image
World coordinates Z	Absolute coordinate for Z axes of destination image
Delta X	Displacement along X axes in relative coordinate
Delta Z	Displacement along Z axes in relative coordinate
Delta Angle	Rotation to perform after a movement

Table 3.1: Information available from dataset

3.3.3 Splitting of data

Before to run the experiments we splitted the csv in three files (train. validation and test) with the best action following the same distribution of the target values.

To do this we used the dissimilarity distance between two probability distributions, called Bhattacharyya distance. Given two probability distributions P and Q on the same domain X the Bhattacharyya distance is defined as:

$$D_B(P,Q) = -ln(BC\,(P,Q)) \tag{3.1}$$

where:

$$BC\,(P,Q) = \sum_{x \in X} \sqrt{P\,(x)\,Q\,(x)} \tag{3.2}$$

is the coefficient of Bhattacharyya. Using this coefficient we split the data so that both training, validation and test set follow the same probability distributions for the three target values. In the figures below we show how the split works.

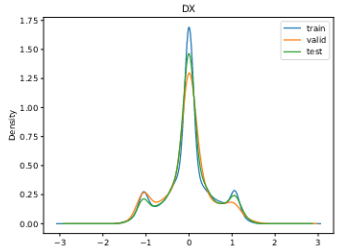


Figure 3.4: Split for delta X

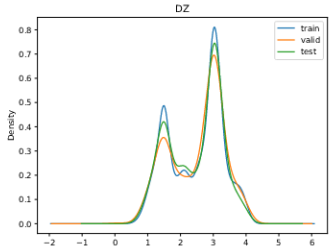


Figure 3.5: Split for delta Z

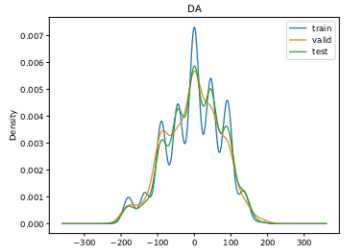


Figure 3.6: Split delta angle

3.4 Experiments

In this section we address the navigation task, the core for find an exit from any building, using the new dataset, introduced in 3.3. Furthermore we will use in our experiments, the transfer learning technique with the ResNet-50 as backbone of the models.

The following sections discuss the different approaches to one-step navigation which have been bench marked on the considered data.

3.4.1 Naive approach

To tackle a problem like browsing, this first approach must be as simple as possible. This aspect has several advantages:

1. creating an upper limit for the error to the following experiments,
2. study the goodness of the data and see how a simple regression task works

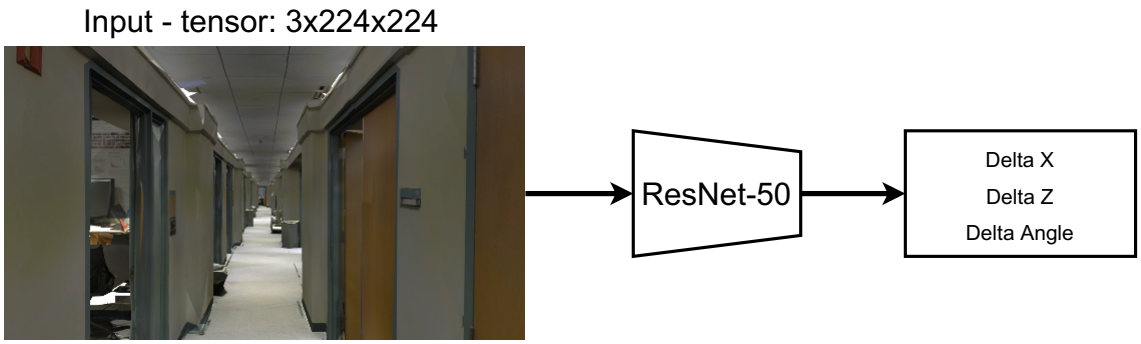


Figure 3.7: Naive approach

This approach, therefore, is based on the use of transfer learning, to exploit the reuse of a pre-trained network such as a ResNet-50, which thanks to shortcut connections, it. takes less time in the training phase, allowing us to have results in a short time. As input data to this network we used the data set with the best actions, as it allows us to see as many points as possible arrived at a certain target image. In particular, we try to predict, with a regression task, the three target columns (Delta X, which represents the movement along the X axis; Delta Z which represents the movement along the Z axis and Delta Angle which represents a rotation after a certain movement) that allow us to arrive at a target image starting from its starting image. The scheme of this approach is depicted in 3.7.

Experimental setup and evaluation

The optimizer used in thsi first approach was Stochastic Gradient Descent [45] with different classical momentum (without Nesterov's one because we use a pre-trained

ResNet [46, 47]). Due to simplicity of this first approach we used a classical regression task loss: Mean Squared Error

$$MSE = \frac{1}{n}\sum_{i=1}^{N}(y_i - y_{pred})^2 \tag{3.3}$$

The metrics for measure the errors are the mean euclidean distance (for the displacement) and the mean angular distance (for the rotation)and as described in table 3.2 the model suffer of overfitting, even adding different levels of dropout.

Table 3.2: Results Naive approach

Model	**Mean error in euclidean distance**	**Mean error in angular distance**
ResNet-50	$76.65cm$	$48.74°$

In figure below 3.8 is depicted how the model wrongs, in particular the first row there are the starting image, the intensity image which describe the movement and the space available in that point. In the second row there are the image of the cube which represents the displacement, a grid with the ground truth and the predictions of our model, at the end the destination image.

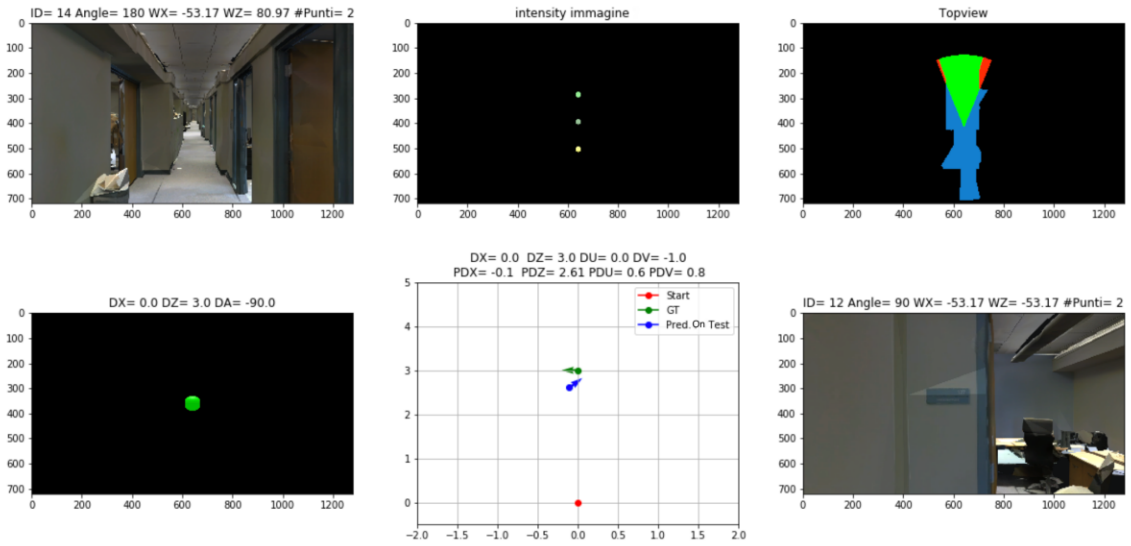


Figure 3.8: How Resnet50 predict an action on test set in the first approach

3.4.2 Approach based on the prediction of probability distribution

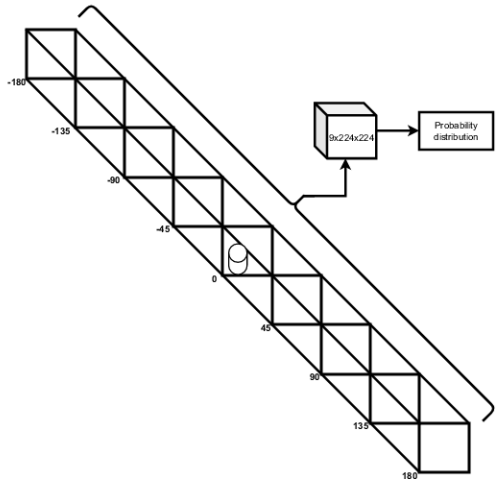


Figure 3.9: Intermediate representation (a)

Although the Naive approach gave us bad results, it made us understand that we needed an intermediate representation, to take steps towards the solution of the problem of navigation without an internal map. We thought of a probability distribution that represented each action, exploiting the image with the cuboid, whose shape and positioning in the image represent the movement that must be made to reach the desired destination 3.9. This intermediate representation is like an

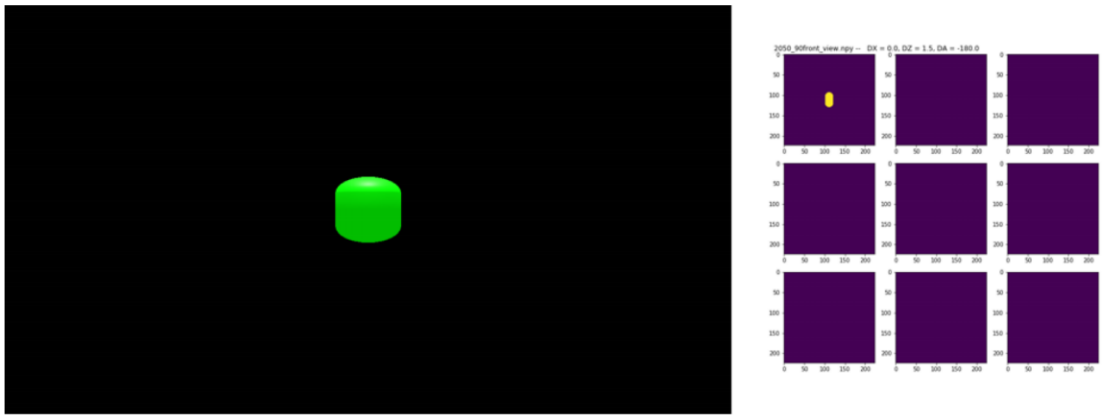


Figure 3.10: Intermediate representation (b)

image with 9 channels, where each channel represents the rotation to perform 3.10.

It can be treated as a probability distribution, because the sum of numbers that depicts the cuboid is equal to 1. First of all, we train a simple CNN starting from this structure to predict the target action. From these ground truth data we arrive to an error of 11.57*cm* for the movement and an error of 4.66° for the rotation. This experiment with the Ground truth of this intermediate representation gave us a good result, now the difficulty is to create a model that predict a good and faithful depiction of these nine-channels images. To tackle this difficulty we used a model pre-trained for a task of semantic segmentation such as a **Fully Connected Network ResNet-50** (FCN-ResNet50 2.3.6) that can be seen such as a Convolution auto-encoder with a Resnet50 as encoder. Finally, the trained model of this approach is illustrated in figure 3.11.

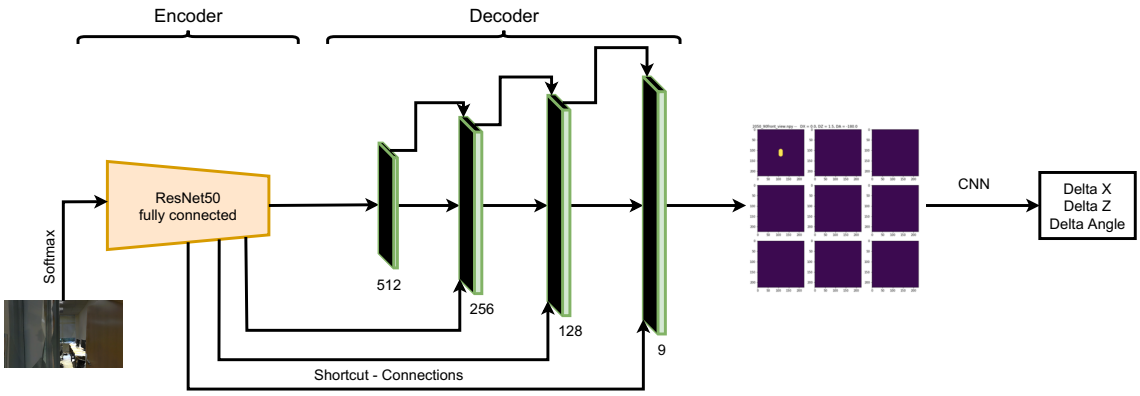


Figure 3.11: Approach based on the prediction of probability distribution.

Experimental setup and evaluation

We have used a different optimizer, Adam [48], from the one in the naive approach, as it gave us better results. And starting from the softmax distribution of the input image, we trained the FCN-ResNet 50 with a Kullback-Leibler Divergence loss, that measure the divergence between two probability distributions (the softmax of the input image and the probability distribution of the intermediate representation). It is defined as:

$$D_{KL} = \sum_{x \in X} P(x) \, log_2 \left(\frac{P(x)}{Q(x)} \right) \tag{3.4}$$

The model is not able to reconstruct a faithful representation of the probability distribution, even by changing loss and fine-tuning the model. Instead of using a simple CNN, after the reconstructing of the intermediate representation, we used a shallow customized ResNet with 4 basic blocks. After the ablation study, the results of the model are shown in table 3.3.

Table 3.3: Results of the approach based on the prediction of probability distribution

Model	**Mean error in euclidean distance**	**Mean error in angular distance**
FCN-ResNet50+CNN	70.08cm	48.09°

3.4.3 Approach based on Actor Critic

With deep learning we have not found a good solution due to the large overfitting that afflicts our experiments. We therefore changed the paradigm of learning using the deep reinforcement learning. In particular we studied a typical algorithm of this paradigm of Learning: **Actor Critic** in a continuous domain. Actor-critic methods are Temporal Difference methods that have a separate memory structure to explicitly represent the policy, independently of the value function. The policy structure is the actor in this scheme, because it is used to select actions; the estimated value function is the critic, because it criticizes the actions performed by the actor. Learning is always in line with policy: the critic must know and criticize whatever policy is currently followed by the actor. The critic takes the form of a temporal difference error. This error, is a scalar signal which is the only output of the critic and drives all learning, as depicted in Figure 3.12.

In this task the policy learn a continuous probability distribution represented by a Multivariate normal (also called Gaussian). The choice of the reward follows a very specific strategy, finding the point that guarantees us, through the execution of an action, to find as many visible points as possible. The actor is performed by a ResNet50 which its input is concatenated by the Critic, performed by a simple Multi Layer Perceptron. The output tensor of the two networks fed an hidden layer that has two output, one for the actor with two probability distribution (the first for the movement and the second for the rotation) and the other output is for the value function: a scalar. Finally the model is illustrated by the figure 3.13.

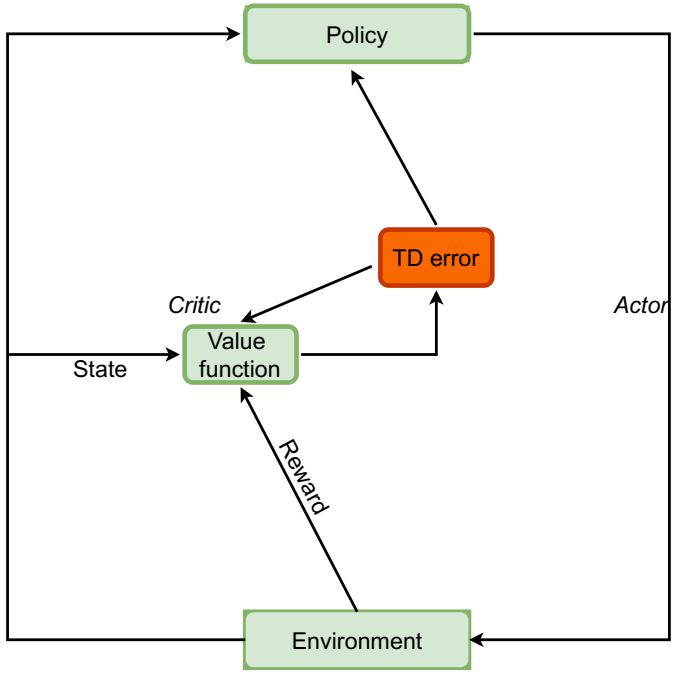


Figure 3.12: Actor critic scheme

Experimental setup e reward achieved

We used the Adam optimizer, and a Smoothed L1 Loss that is a derivative of the Huber Loss, which is less sensitive to the outliers. We achieved a mean reward

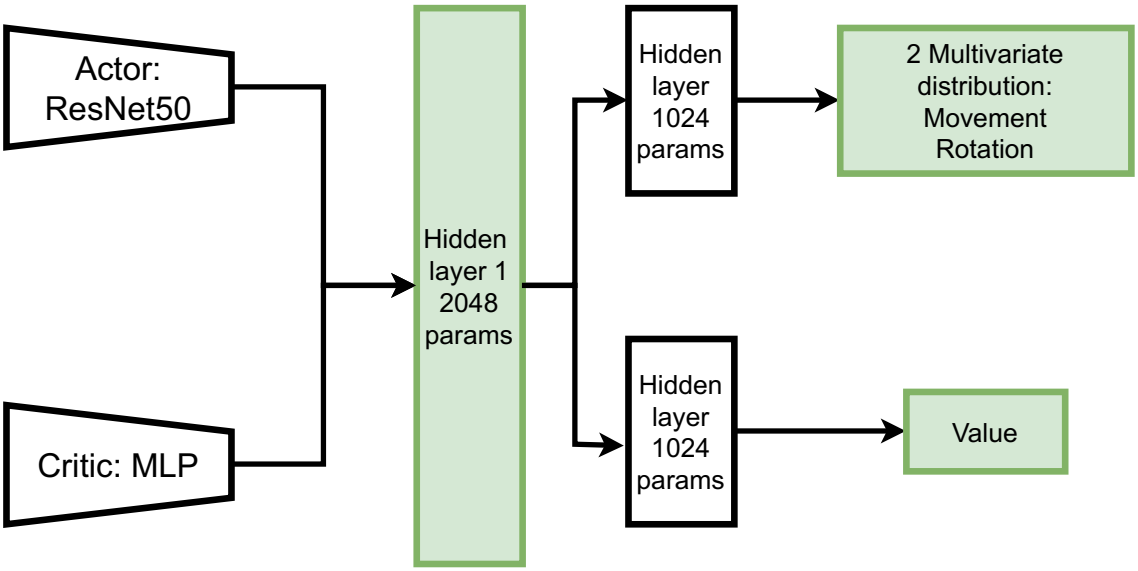


Figure 3.13: Actor critic model used

of 0.27, is a low result due to a lot of point with 0 visible points. If we get poor results with deep learning, we can't hope for a better results with deep reinforcement learning.

3.4.4 Approach based on the prediction of top-down maps

The fourth type of experiments only concern the prediction of displacement. By involving the image with the points that represent the movement to be performed, we can create grids that can be used as an intermediate representation, in fact the points in the image have a position and intensity of color that combined are a good expression of the movement to perform. Furthermore, by using a morphological operation such as dilation and a kernel such as the Gaussian filter, we not only increase the dots in the image but also eliminate possible noise. Let remember that the basic effect of morphological operator (dilatation) on a binary image is to gradually enlarge the boundaries of regions of foreground pixels (i.e. white pixels in most cases). Thus, areas of foreground pixels grow in size while holes within those regions become smaller. Thus, applying a dilatation on a $5x5$ for the dilation

Table 3.4: Kernel matrix of the dilatation operator

0	1	1	1	0
1	1	1	1	1
1	1	1	1	1
1	1	1	1	1
0	1	1	1	0

operation (as the table depicted in 3.4), the image with the dots becomes exactly as shown in the figure 3.14.

The Gaussian filter applies some kind of blur to images, usually it is used to reduce Gaussian noise. The images from those dilated to those with the Gauss filter applied are like the one in the figure 3.15. We exploit an auto - encoder which is a neural network that learns to copy its input to its output or a similar image of input. It has an internal (hidden) layer, that describes a code used to represent the input, and it is constituted by two main parts: an encoder that maps the input into the code (the bottleneck), and a decoder that maps the code to a reconstruction

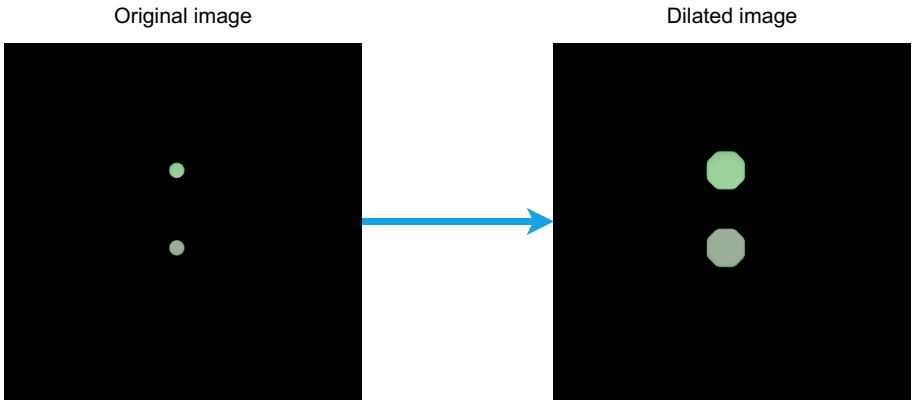


Figure 3.14: Transformation with Dilatation operator

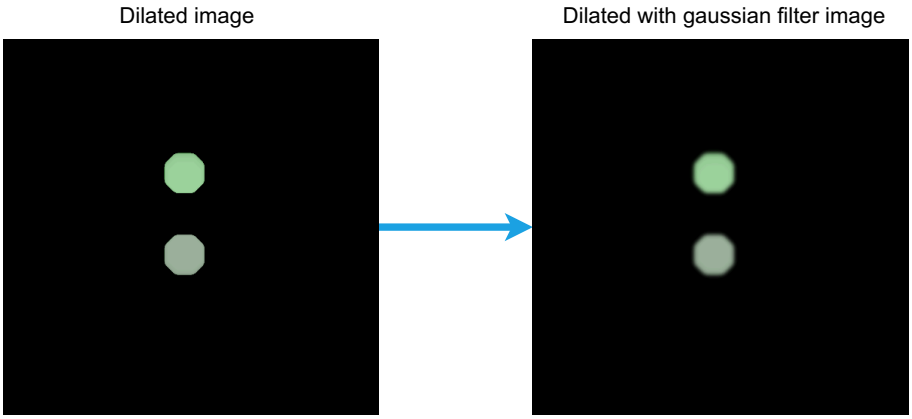


Figure 3.15: Transformation applying Gaussian filter on dilated image

of the original input or a similar image. Their most traditional application was dimensional reduction or feature learning, but more recently the auto - encoder concept has become more widely used for learning generative models of data. I used a convolution auto - encoder composite by a ResNet50 as encoder, and some linear layers as decoder for reconstruct the image dilated with Gaussian filter. The key idea of this approach is to reconstruct the intermediate representation to predict the displacement. Finally the model is depicted in figure 3.16.

Experimental setup and evaluation

The model was be trained with the Adam optimizer, with a Cross entropy loss for the reconstruction of the grid and the Smoothed L1 Loss for the prediction of the

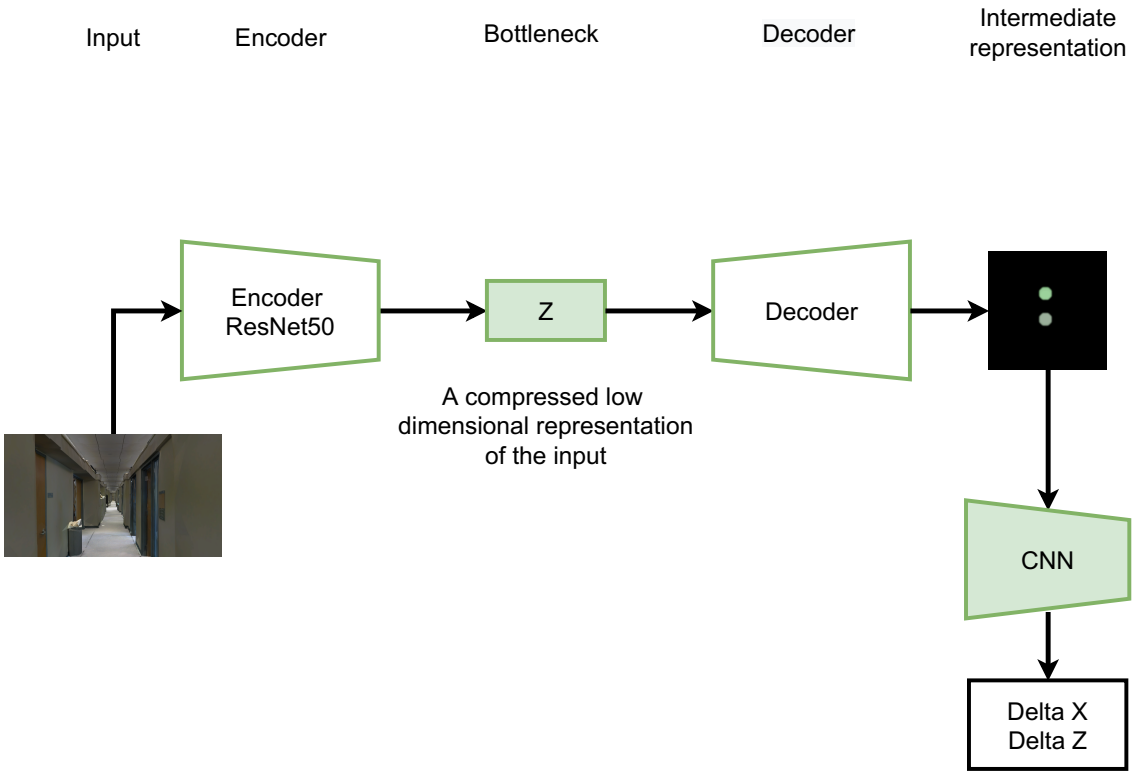


Figure 3.16: Model used in the 4 approach

two target values (only the movement). The errors of this approach are depicted in table 3.5

Table 3.5: Results of approach based on the prediction of top-down maps

Model	Mean error in euclidean distance
Auto - encoder + CNN	78.42cm

3.4.5 Multi-head approach

In this approach we investigate the problem using the multitasking, in particular we use in a different way the architecture of the fourth approach to predict the three target values. Using the intermediate representation and the auto-encoder, we attached, starting from bottleneck, three linear layers, exploiting the compressed information of input and the weight for the reconstruction of the intermediate representation.

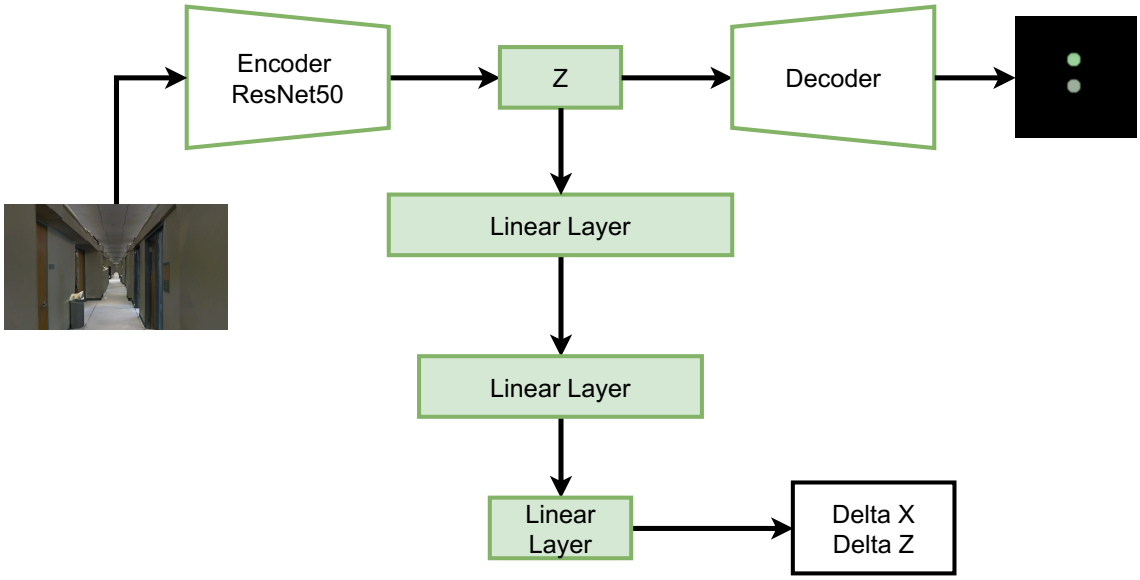


Figure 3.17: Model used in the 5 approach

The architecture is illustrated in figure 3.17 The key idea of this architecture is to extract what the auto - encoder learned from the image of input and the weights to reconstruct the grid. In this way we achieved a slightly better performance, using a multi-head architecture.

Experimental setup and evaluation

The model, which shows slightly better performance, has been trained with the Adam optimizer with the Smoothed L1 Loss for the prediction of the two target values. The errors of this approach are illustrated in table 3.6

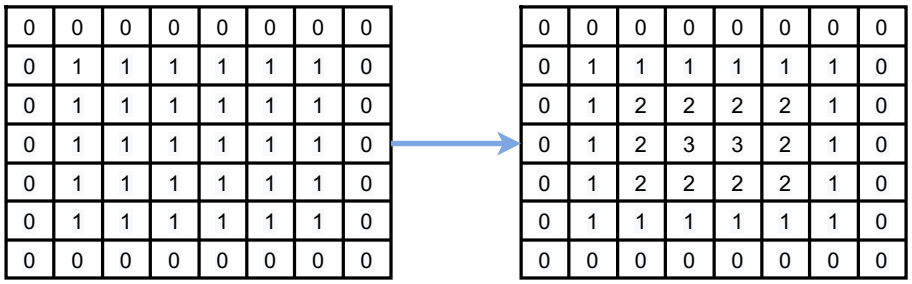
Table 3.6: Results of Multi-head approach

Model	Mean error in euclidean distance
Multi-head	$75.99cm$

3.4.6 Approach based on distance transform

The sixth approach created a substantial modification to the fifth approach model, thus attempting to calculate only the displacement of the target. I transform the

intensity images by applying the distance transformation. The distance transform is a morphology operator that is normally applied only to binary images. The result of the transformation is a gray scale image similar to those intermediate representations used in the fourth and fifth approaches. One way to think about distance transformation is to first imagine that the foreground regions in the binary image are made up of a uniform, slow-burning flammable material. So it is considered simultaneously to start a fire at all points on the border of a foreground region and let the fire make its way inland. If we then label each point inside with the amount of time it took the fire to reach that point, then we have actually calculated the distance transform of that region. The figure 3.18 shows an example of the distance transformation for a simple table as an image. So, I calculated the inverse of the



0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0

0	0	0	0	0	0	0	0
0	1	1	1	1	1	1	0
0	1	2	2	2	2	1	0
0	1	2	3	3	2	1	0
0	1	2	2	2	2	1	0
0	1	1	1	1	1	1	0
0	0	0	0	0	0	0	0

Figure 3.18: Example of distance transform in an image

distance transformation to have an image with a black background and white dots in the center. The transformation of these intermediate representations is represented in 3.19. The key idea is similar to the intermediate representation of the fourth

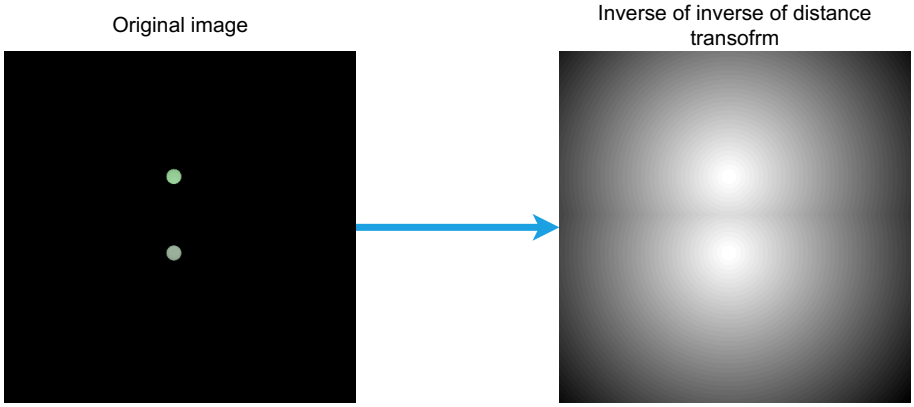


Figure 3.19: Example of distance transform in an image

approach 3.4.4. Having a binarized image with a color change from black to white can be a high signal for the network, furthermore, this representation represents a real grid; but the drawback is that we lost the intensity of the color, an important representative element for movement. In fact, using this intermediate representation as input to a simple CNN, we encountered an error of 48.68cm, a rather large error for ground truth images.

3.4.7 Approach based on probability distribution with a parallel ResNet

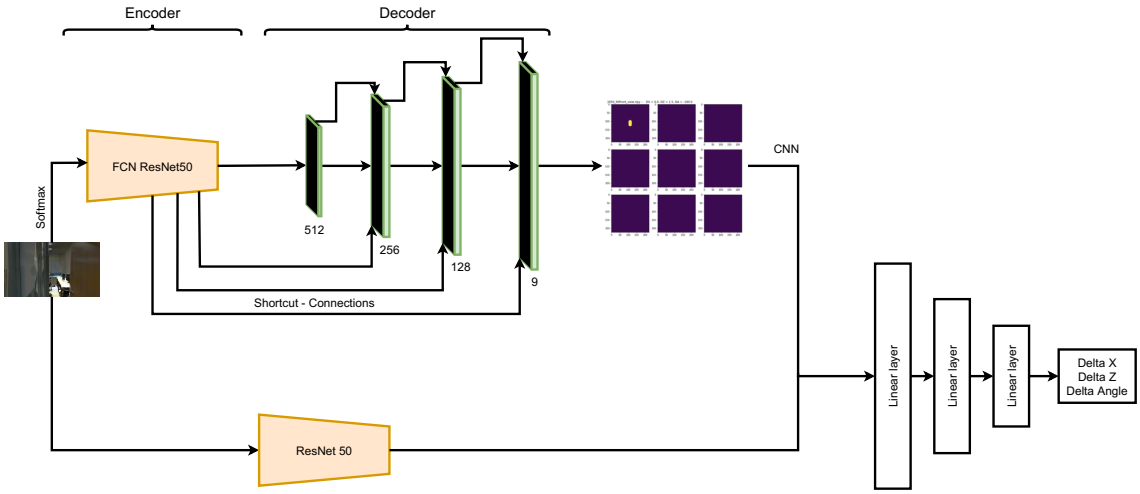


Figure 3.20: Seventh approach model

The best approach in these series of experiments was the second one 3.4.2, giving us the better performances, in term of euclidean and angular distance error. For this reason this approach is a continuous of the aforementioned architecture. The key idea is to add another network architecture in parallel to push learning towards target values. The model described above is illustrated in figure 3.20. We supposed that the parallel ResNet50 learn from image and its output tensor, if added to the one of the FCN - Resnet50, can help the learning of the target values. In this way this architecture is formed by two parts, the first one learn to create an intermediate representation, more specifically a probability distribution, while the parallel ResNet50 learn from the input image to predict the target values. The two output are added into one tensor and passed into three linear layers to predict the

three target values. This architecture is the result of an ablation study and the performances are described in the following paragraph

Experimental setup and evaluation

In training time, we used an Adam optimizer because it can handle sparse gradients on noisy problems. We used a Smoothed L1 Loss, that is a derivative of the Huber loss, to be less sensitive to outliers than Mean squared Error (MSE). The error after the evaluation of the model are shown in table 3.7.

Table 3.7: Results of approach based on probability distribution with a parallel ResNet

Model	**Mean error in euclidean distance**	**Mean error in angular distance**
FCNResnet50 - ResNet50 in parallel	66.67cm	46.87°

3.5 Comparison of the approaches used

Table 3.8: Comparison of the approaches used to solve a Map-Less navogation problem

Model	**Mean error in euclidean distance or accuracy**	**Mean error in angular distance**
Naive approach	76.65cm	48.74°
Approach based on the prediction of probability distribution	70.08cm	48.09°
Approach based on Actor Critic	0.27 of accuracy	/
Approach based on the prediction of top - down maps	78.42cm	/
Multi-head approach	75.99cm	/
Approach based on distance transform	48.68cm whit ground truth data	/
Approach based on probability distribution with a parallel ResNet	66.67cm	46.87°

Table 3.8 shows and highlights how a naive approach can result in a limit for the worst error. By gradually using more refined approaches to deal with the problem, the table shows how the error decreases, The table also depicts that even by changing the learning paradigm, the results do not change, in fact if we are unable to have an efficient baseline with deep learning we cannot hope for better results with deep reinforcement learning. The best baseline is the one that uses a probability distribution as an intermediate representation, using a FCN ResNet 50 (suitable for semantic segmentation purposes) and a ResNet 50 in parallel, to exploit and improve the low level representation of the input, thus feeding an Multi layer perceptron that allows us to have the best results for the problem covered in this chapter.

3.6 Conclusions and future works

The displacement and its relative rotation are assumed as if they were a single action to be performed. To choose the best action, we used a strategy of finding the end point (after performing an action) with multiple points visible, to allow an intelligent system to find a path to the exit. We have had the problem of overfitting in all architectures of the approaches, that is when a model is perfectly calibrated for the training set, without being able to generalize. This problem is due to different images having the same action to be performed, so any model fails to create a mapping function between the input image and the action to be performed on the output. In addition to overfitting, this problem presents multiple difficulties, first of all, without an internal map and with different scenarios, it is difficult to create a knowledge of the surrounding space. Even if you are able to have an awareness of the surrounding space, moving and creating a path to an exit is a very difficult task. Furthermore, the noise present in the images does not help at all, in fact we have not been able to create a faithful intermediate representation. A future work to improve this task is a change of dataset, with real images in color and with their own depth and semantic ones, to have an high signal for the models. The problem faced in this chapter has great resonance in research, just think of paper [4] that describe navigation episodes for a Map-Less autonomous robot in indoor environments.

Chapter 4

Visual RSSI Fingerprinting for Radio-Based Indoor Localization

In this chapter we are going to tackle the problem of localization starting from RSSI as input, a radio wave signal which is affected by noise and ambiguity. Due to nature of this radio signal, localize a beacon through RSSI become a tough task, but with a machine learning-based solution which have the advantage to better cope with noise, even though it is not trivial to use the raw signal without filter as an input. This work [1] was published and introduced in the conference Sigmap 2022 [1].

4.1 Introduction

Being able to infer the position of an object, a person or a robot in an environment is an important task for many applications including tracking goods in a warehouse, helping people to locate [49, 50] and navigate an environment, or the prediction of their intentions [51, 52, 53, 54, 55]. To tackle this problem, different technologies have been used so far: GPS, radio-wave signals, laser ranging scanners, and cameras [21]. Among these approaches, we focus on radio-wave signals, which are convenient thanks to cheap hardware solutions, which work in a good way in indoor settings (e.g., based on WiFi or Bluetooth). Localization through radio-wave signals leverages the processing of RSSI values (Received Signal Strength Indicator - a measure of the power of a radio signal) observed by a beacon (or the beacon itself) attached to the object to be localized while receiving signals from a set of antennas placed at known locations [22]. Since RSSI values can be used to estimate

[1] https://sigmap.scitevents.org

the distance between the observer (the object to be located) and the emitter of the signal (an antenna placed in the known position), it is possible to use geometric methods to directly estimate the position of the target . However, due to the nature of the input, these geometric solutions (trilateration or triangulation) are affected by a previous error. To overcome this problem, in this chapter we describe the pipeline for building the dataset and use it to train a machine learning-based solution to address the localization task problem.

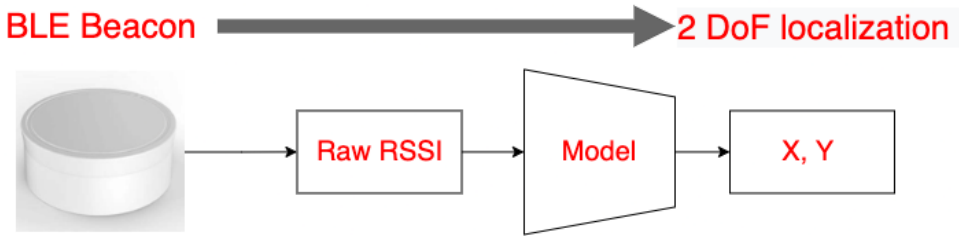


Figure 4.1: Problem formulation

4.1.1 Chapter outline and contributions

The problem treated in this chapter is to localize a beacon in movement in indoor environment, using the RSSI as input signals and the Recurrent Neural Networks to better cope the input's noise and to exploits the natural temporal variability of RSSI. The data collection and labeling process problem, which is not trivial and it typically requires building a grid of dense observations, which can be resource-intensive. To overcome this issue, in this chapter, we propose a pipeline which uses an autonomous robot to collect RSSI-image pairs and Structure from Motion to associate 2D positions to the RSSI values based on the inferred position of each image. This method allows us to acquire large quantities of data in an inexpensive way. This chapter is divided, thus, in two main lines:

- The construction of the dataset and the tools used to obtain it, illustrating how our pipeline overcomes the typical literature methods for collecting these types of data (RSSI), and
- a baseline created to tackle the localization task, using the new dataset.

This work leads to a paper [1] whose contributions are precisely the new dataset and its pipeline to build it, and a baseline that uses the aforementioned dataset to tackle the problem of localize a beacon in movement in an indoor environment.

4.2 Dataset

To date we know that the entire world of artificial intelligence, and its branches, is entirely based on the presence of a large amount of data. The dataset is a set of structured data created to be read and processed by an algorithm. Usually this dataset is very large, one of the main characteristic for using Deep Learning-based algorithms. One the main problem in research today, hence, is to create a labelled dataset, perfectly suitable as input for tackling a particular task. In this section we describe all the tools to collect pairs of RSSI-Image, that is the main parts of our dataset. Although radio waves have several problems, on the other hand they have several advantages, such as cheap hardware. The main advantages of the gathering of RSSI by a beacon are:

- low cost
- low energy
- low acquisition costs
- fast processing

Radio waves signals are very suitable for a dataset to localize and train an intelligent system for the position of an object, a person or a robot in an indoor scenarios. In this chapter we address the problem to localize a beacon, starting from radio waves signals, such as the RSSI, exploiting its time variability for training models composed by Recurrent Neural Networks (RNN).

4.2.1 Radio wave signal: RSSI and its limits

The RSSI, acronym of Received Signal Strength Indicator, is the value that indicates the strength of the received signal, in particular, this radio wave signal allows

to know whether or not a signal is sufficient to establish a wireless connection between a transmitter and a receiver. It has a range in decibel, from -100 to 0 dBm with different power levels. A more highly value indicates a more powerful signal, especially, the closer the RSSI value is to 0 dBm, the stronger the signal. Despite the advantages of this signal described above , RSSIs also present two significant drawbacks:

- *ambiguity*, i.e. two different devices at the same position can measure different RSSI values (and vice-versa),
- *instability*, i.e., objects present in the environment, other radio signals or external factors can create noise, disturb the radio signal and make the measure less reliable.

Due their nature, the RSSIs values can be compromised also from other radio wave signals (such as an electrical station with radio base antennas), the presence of objects that represent obstacles (such as furniture and furnishments), walls, other objects with signal radio and so on.

4.2.2 Relations between Euclidean distance and RSSI

Machine learning and geometric based methods can tackling partially the advantages described above. In geometry there are some mathematical relationship between distance in meters and the RSSI, such as the following formula:

$$Distance = 10^2 \times \frac{MeasuredPower - RSSI}{10 \times N} \tag{4.1}$$

Another way to estimate the distance is given by the ratio between transmitter and receiver, the **Friis equation**. This equation take the name of its inventor Harald Trap Friis born in 22 February 1893 and died 15 June 1976, that was a Danish-American radio engineer whose work at Bell Laboratories included pioneering contributions to radio propagation, radio astronomy, and radar. His equation allows to calculate the ratio between the power received by a device and the transmitted power of another one.

$$\frac{P_t}{P_r} = G_t G_r \left(\frac{\lambda}{4\pi R}\right) \tag{4.2}$$

where:

- P_r is the received signal strength in watts,
- P_t is the transmitted signal strength in watts,
- G_t is the gain of the transmitting antenna,
- G_r is the gain of the receiving antenna,
- λ is the the wavelength of the radio carrier
- R is the distance between the transmitter and receiver.

If we extract the R from this formula, we can achieve the distance. After the calculation of the distance between a receiver and a transmitter, the position is given by a triangulation problem with a certain error due to the instability of a signal. Solution on machine learning-based, instead, have the advantage to better cope with noise and instability, but require many radio signal observations associated to the correct position in the target space.

4.3 Tools used to construct the dataset

The instruments used to transmit and receive a payload with the relative information are the **beacons** and the antennas gateways or other devices.

4.3.1 Beacons

The term Beacon was first coined in 2011 when Apple developed and launched the iBeacon technology. It was an indoor positioning system, to which Google immediately responded with its Eddystone. The Cupertino multinational itself described it as "a new class of low-power, low-cost transmitters that can notify their presence to nearby devices running iOS 7 and iOS 8". Beacons are sensors based on BLE (Bluetooth Low Energy) technology that have the ability to connect and transmit data to mobile devices such as tablets, smartphones and antennas. Able to send messages at a distance between 10 and 70 meters, the operation of a Beacon is based on the communication between two different devices. The first, the sender, can be a PC,

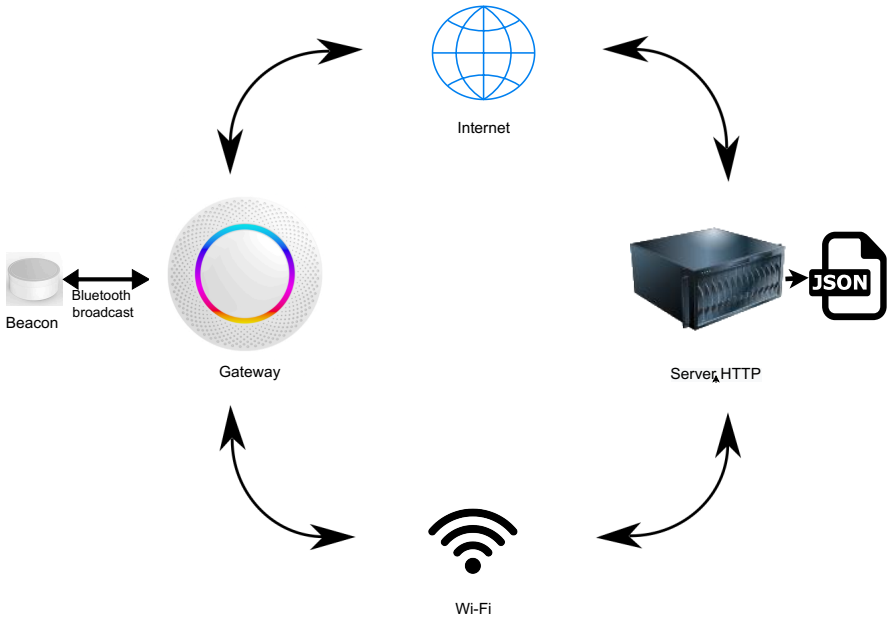


Figure 4.2: Cycle: Beacon-Gateway-Json

a smartphone or any type of dedicated device, has the task of sending a certain message repeatedly. The second, the receiver (the smartphone app or the antennas) alerts the user every time he receives the message transmitted by the Beacon. The radio wave signal sent by the Beacon sensor and "delivered" to all nearby devices with active Bluetooth, contains a unique universal identification number (UUID). Beacons can find application in many sectors, such as museums, marketing, people with disabilities and so on, becomes an accessible technology with high potential and cheap cost.

4.3.2 Antennas

Instead of using the smartphone app to receive the position related RSSI signal, a client-server infrastructure was used, where the beacon transmitted the signal and the antennas sent it back to a cloud server. To receive the payloads from the antennas and create the relative dataset, instead to use a cloud server, we used a http server, that was raised to manage the post message and save the payloads inside a Json file, all inside in a wireless local area network (LAN). In a nutshell, the entire infrastructure is orchestrated by the coordination between beacons and

antennas, the latter of which transmit on the relative http server, which stores the relative Json files as shown in the figure 4.2. The antenna in the indoor environment are five and were bought from Minew Store. [2]The type of antenna used to collect the data is depicted in figure 4.3

Figure 4.3: Antenna

[2]https://www.minewstore.com/product/g1-iot-gateway/

4.3.3 Raspberry Pi 4

Raspberry Pi is a multi core mini PC, a complete ARM hardware architecture device enclosed in a single board paired with a Linux-distro platform, such as Raspberry pi OS. It was launched for the first time in 2012, to carry out robotics projects and research in a professional level, in fact it is considered an alternative to the more famous Arduino. The card is preconfigured, however modules can be added to increase and customize its functionality based on the type of use. We used the raspberry to create a server HTTP raised to manage and storage the RSSI Payload - Image pairs, while the sanbot was in movement around the environment.

Figure 4.4: Raspberry PI 4

4.3.4 SanBot

Sanbot is a robotic system created by a company called QIHAN, with many features. In the image 4.5 the robot is shown, which is able to move dynamically in an indoor environment, since it has several sensors that allow it to avoid obstacles, create an online environment thanks to its vision system (even partial) and more. Thanks to this capability, the robot was used to mount the beacons and the Raspberry on top to create the dataset and move around the whole environment, capturing many images with the RealSense camera, also installed on top of the robot. Subsequently, starting from the collected images, a 3D model of the surrounding environment was created and then the target coordinates for the localization activity were created.

Figure 4.5: Sanbot

4.3.5 RealSense Camera

To collect the image of its payload, we placed a camera above the sanbot as it dynamically moved around the environment. The purpose of the camera is to synchronize an image for each payload received. The camera used for this is the RealSense Camera D435. After having mounted the relative SDK on raspberry to manage the image-payload synchronization and store these RSSI-image pairs.

Figure 4.6: Real Sense Camera

Figure 4.7: Environment

4.3.6 Environment

The collaboration between the University of Catania, represented by the department of mathematics and computer science, in particular the IPLab, and the company OrangeDev SRL, has made it possible to conduct this research, which concerns the positioning of an object (e.g. beacon) within an indoor scenario.

The indoor space used for this localization activity is the OrangeDev corporate environment. This environment is shown in figure 4.7, it covers about $160m^2$, a large space that is a difficult problem in a localization business. To make things even more difficult, the environment is affected by the radiation of a radio-electric station and the installation of an additional 7 beacons (for a total of 10) compromising the

resolution of the localization task and to make the environment more difficult as possible.

4.3.7 Comparison filtering techniques and pre-experiments

Our work on the localization problem focuses on the exploitation of the raw signal, to make a contribution in related research. But talking about a comparison on filtering techniques 2.4 makes us understand which are the best filters to limit the noise of the radio signal such as RSSI. Although the gray model, compared to the current RSSI, has far fewer fluctuations, it tends to overestimate and amplify the increase in RSSI in the case of rapid growth, such as when the beacon is very close to the antenna. Fourier and Kalman, on the other hand, have similar behaviors, tend to mitigate RSSI fluctuations quite well, and accurately follow the actual RSSI sequence. The particle filter, on the other hand, mitigates the fluctuations of the RSSI very well, but sometimes introduces a delay between the raw and filtered values. These behaviors are illustrated in Figure 4.8. The moving average filter has a big disadvantage: it limits noise very well but inserts untrue values, which could lead to signal fitting problems.

Before the building of the dataset, we made experiments with a beacon to better understand how the RSSI behaves in the environment. The types of experiments performed were based on the movement of a beacon in comparison to a fixed antenna. In our experiments we left a beacon stationary near an antenna, either we move it away from the antenna in question. We also calculated the Kalman and Fourier filters to see the filtered responses of the raw signal. Basically, the beacon stood near or moved away from each antenna. In the figures 4.9 and 4.10 we show the raw RSSI of one antenna, sent by one beacon; in the first figure the beacon was near the antenna, in the second the beacon moved away from the gateway. Both figures show the signals filtered in the orange line (Kalman and Fourier filters).

4.4 New pipeline to collect many data

In this section we are going to describe the new pipeline to collect many data, in particular we illustrate the phases for the building of the dataset, which consists of radio wave signals such as RSSI, each one connected to relative image. We have used

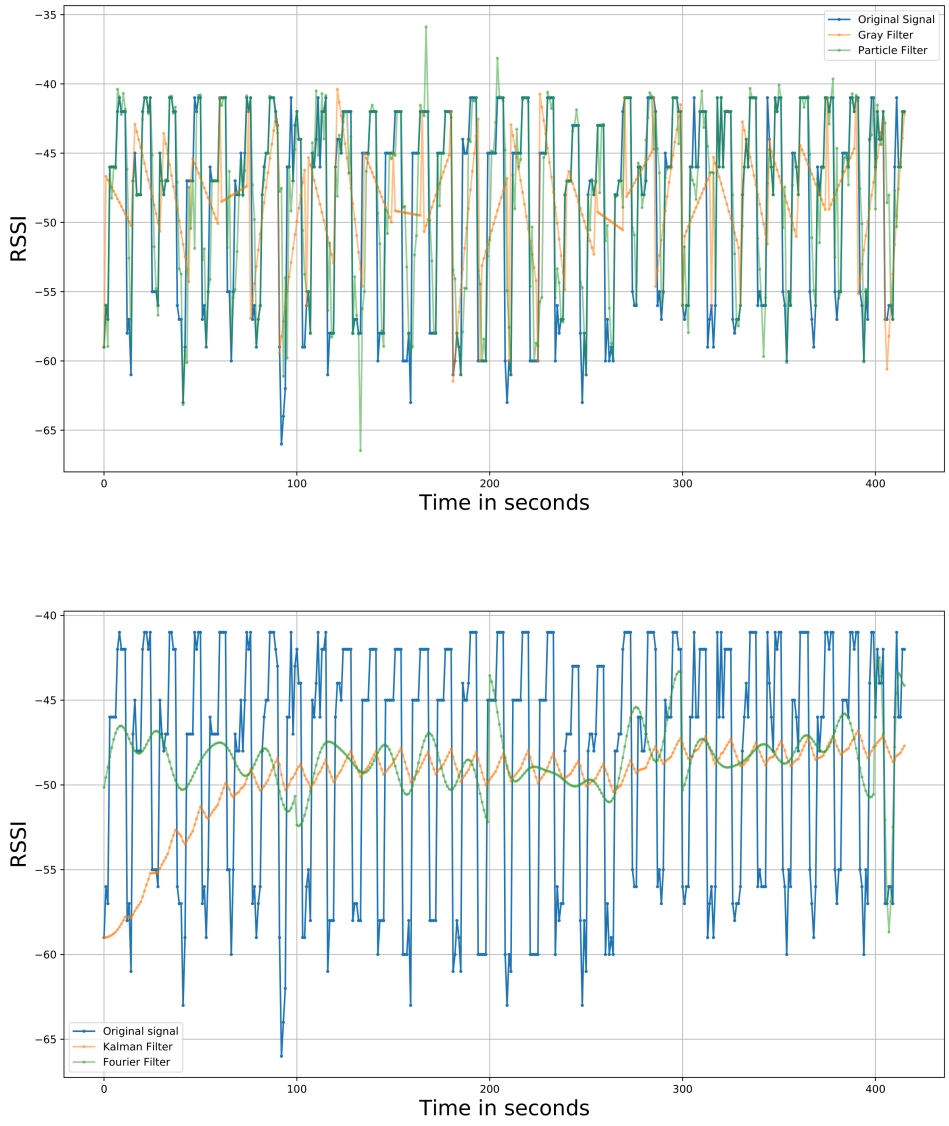


Figure 4.8: Filters comparison

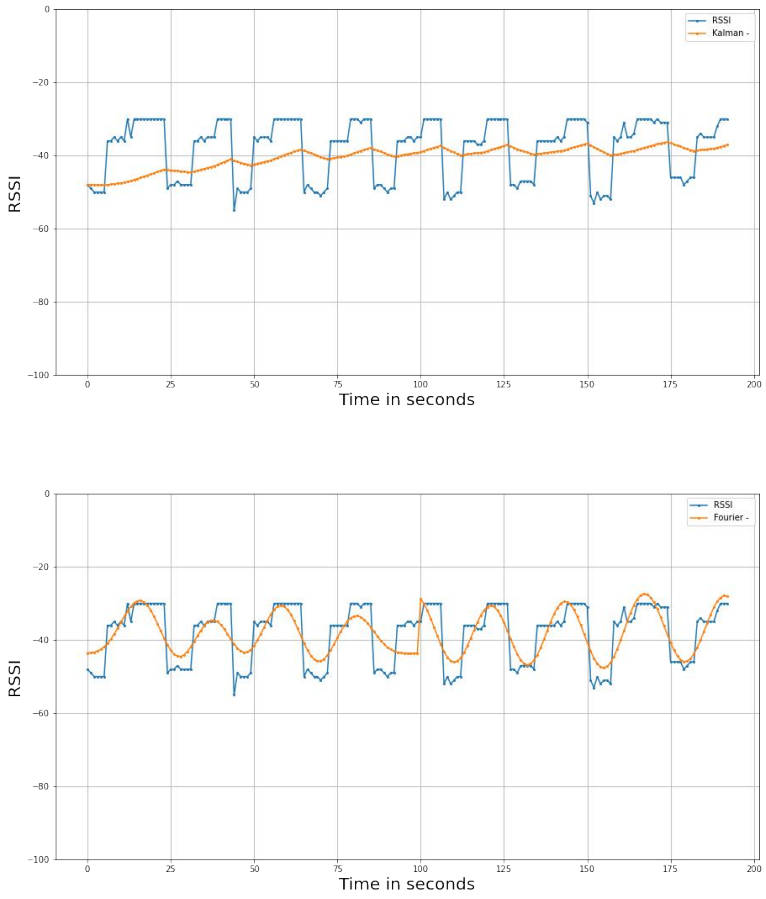


Figure 4.9: In first image the beacon stood near one antenna and the RSSI was filtered with Kalman filter, the second figure, instead, used the Fourier filter

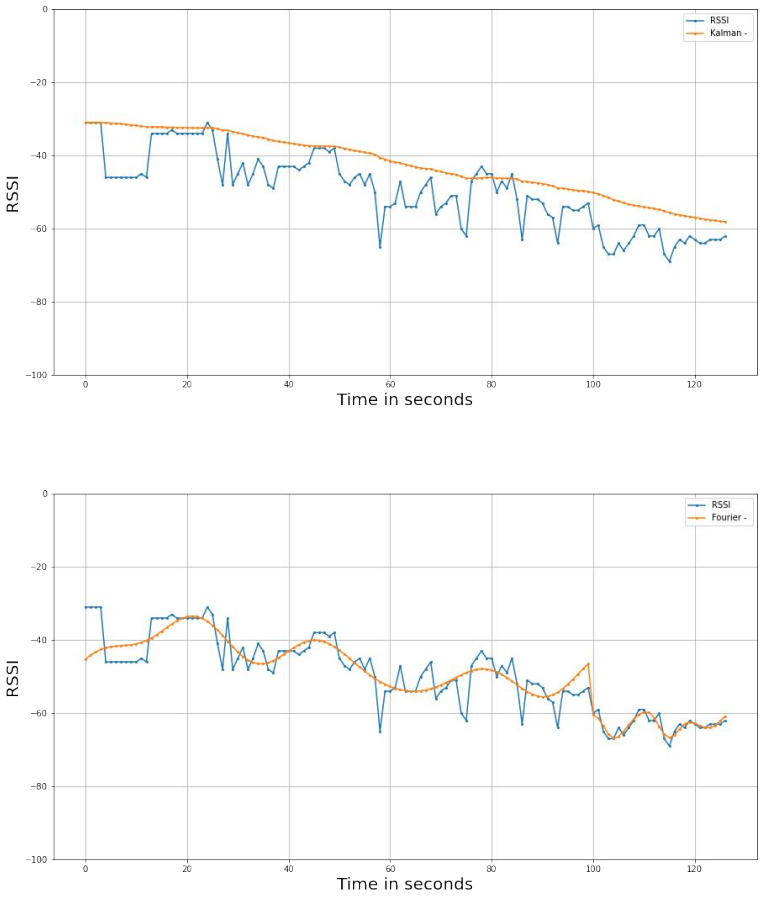


Figure 4.10: In first image the beacon move away from one antenna and the RSSi was filtered with Kalman filter, the second figure, instead, used the Fourier filter

the a technique of computer vision to reconstruct a 3D model of the environment, using Structure from Motion (SfM). From the model of the environment we extract the two degrees of freedom information that will be the target locations to predict.

4.4.1 Building SDK Real Sense on Raspberry

After having carried out many experiments with the elements of the infrastructure, such as beacons and gateways, we installed the SDK of the Real Sense D435 camera on the Raspberry Pi 4, in order to take a photo for each payload received, synchronizing this two elements. The camera, to collect the dataset, has a fundamental role, in fact all the images taken will be used for the reconstruction of the 3D model of the environment, using COLMAP. Intel, the manufacturer of the Realsense Camera, provides a guide to install the SDK on Raspberry Pi 3, perfectly suitable also for Raspberry Pi 4, which uses Raspberry OS [3]. In this way we are able to take pictures from the camera when the Sanbot moves randomly around the whole environment.

4.4.2 Data collection

We installed three beacons on top of the sanbot, with difference sending frequency, to sent the RSSIs to the five gateways. We have deployed others seven beacons to create a virtual tough setup. While the sanbot explores the environment, the raspberry coordinates everything:

- raise an HTTP server to handle all POST messages, such as payloads containing received RSSIs and stored in a JSON file,
- for each payload received from an antenna, it takes a picture from the camera to collect pairs RSSI-image,
- all pairs are stored in the memory of raspberry

Thus we propose a new pipeline that collect a lot of data in several seven sessions of thirty minutes each one, and this pipeline is divided in this following steps:

- we leverage an autonomous robot randomly moving in the target environment to densely collect RSSI observations at various locations,

[3]https://www.raspberrypi.com/software/operating-systems/

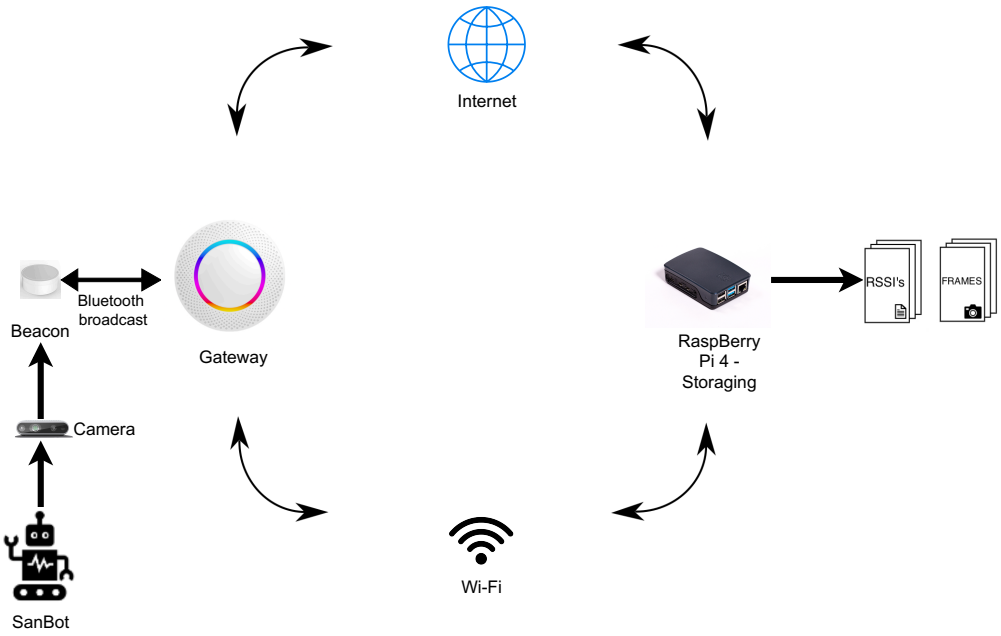


Figure 4.11: Cycle: Beacon-Gateway-Pairs

- each RSSI value is associated to an image captured from the robot's point of view,
- we then use structure from motion (SfM) to create a 3D model of the environment
- we used a small, set of images, to scale the 3d model of SfM with the world coordinates
- from 3D model we extract the target coordinates, connected to the RSSI's.

At the end of this process, each image, and hence each RSSI observation, is associated to a position within the environment, which can be used to form $RSSI - targetposition$ pairs suitable for the training of a supervised machine learning model. It is worth noting that, differently from previous works relying on RSSI observations manually acquired at known locations, the proposed procedure is automatic and naturally allows to obtain a large quantity of examples. Figure 4.12 illustrates the proposed data acquisition pipeline. The main advantage of this approach is that the

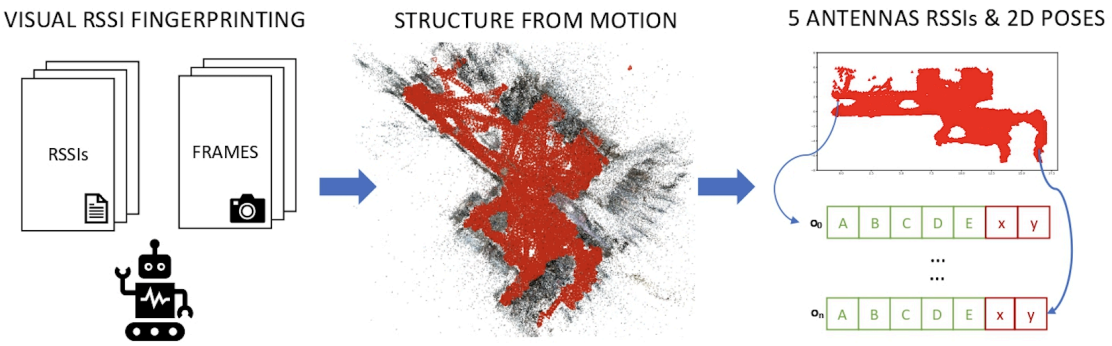


Figure 4.12: Method used to collect the dataset automatically.

trained algorithm (that we show in following sections), can implicitly model sensor noise as well as the positions of the antennas, which hence do not need to be known beforehand. On the downside, collecting RSSI and target position pairs is generally a non-trivial task and often involves performing measurements at known positions sampled through a dense grid over the environment, see Figure 4.13. This manual process, used in previous literature, is often time-consuming, tedious, and prone to error.

4.5 Structure from Motion

The reconstruction of 3D models from images was one of the central topics in computer vision since its beginning, indeed 3D modeling was useful in applications such as autonomous navigation, augmented reality and last but not least **spatial understanding**. This process often involves simultaneously estimating both 3D geometry (structure) and camera pose (motion), it is commonly known as structure from motion [56]. Structure from motion is a photogrammetric gamma imaging technique for estimating three-dimensional structures from ordered or disordered sequences of two-dimensional images.

4.5.1 COLMAP

Despite its widespread use, in the literature it has not been possible to design a generic system for effecting Structure from Motion (SfM). Although existing systems had greatly advanced the relative research, robustness, accuracy and completeness

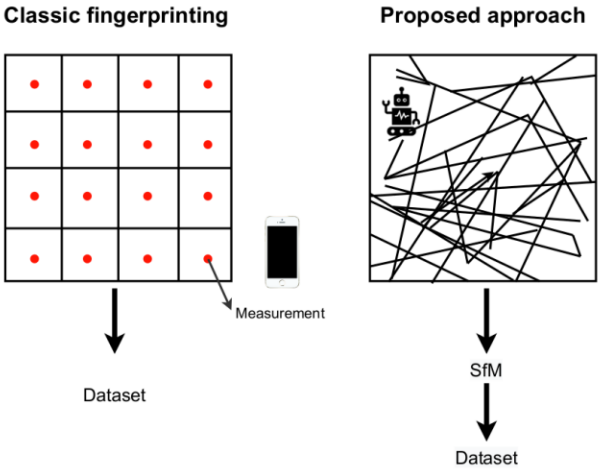


Figure 4.13: Classic manual fingerprinting compared to the proposed approach.

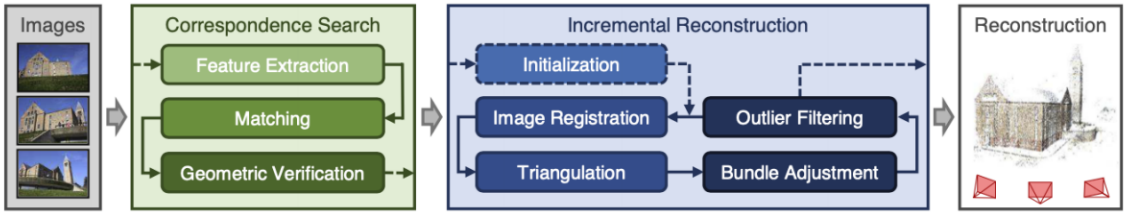


Figure 4.14: Generic pipeline for structure from motion, reproduced from paper [57]

remain the key issues in SfM that prevent its use as a generic method. Many methods were created in the state-of-art until two papers were presented,introducing a generic pipeline for structure from motion with a software called COLMAP [57, 58]. COLMAP is a general-purpose Structure-from-Motion and Multi-View Stereo pipeline with a graphical and command-line interface. It propose a wide range of features for a 3D reconstruction starting from ordered/unordered set of images [4]. Figure 4.14 shows the pipeline presented in the paper [57], a general pipeline for structure from motion procedure. This pipeline is divided in two main stages:

- Correspondence Search
- Incremental Reconstruction

[4]https://github.com/colmap/colmap

Correspondence Search

Given a set of images, the correspondence search is the first stage which try to find scene overlap and identifies projections of the same points in overlapping images. The output of this stage is a set of image pairs geometrically verified and a graph of image projections for each point. This stage has several sections such as:

- **Features extraction**: the features extractions, extracts a set of features for each image, that are invariant to scale, such as the SIFT
- **Matching**: the matching finds the images with scene overlap leveraging the set of features.
- **Geometric verification**: In this section, given a homography **H** describing the transformation of a purely rotating or moving camera capturing a planar scene and the epipolar geometry describing the relationship for a moving camera through the essential matrix **E** (calibrated) or the fundamental matrix **F** (not calibrated), if a valid transformation maps a sufficient number of features between the images, they are considered geometrically verified. After geometric verification, many outliers can exist, so robust estimation technique such as **RANSAC** come into play. Used several methods, this section,in addition to image pairs . return a scene graph with images as nodes and verified pairs of images as edges.

Incremental reconstruction

Given a scene graph as input to the reconstruction stage, the outputs of the incremental reconstruction are pose estimates for registered images and the reconstructed scene structure as a set of points. This stage has several sections such as:

- **Initialization**: Initializing from a dense location in the scene graph with many overlapping cameras gives a more robust and accurate reconstruction due to increased redundancy; but, initializing from a sparser location SfM has a lower runtime process for the reconstruction.

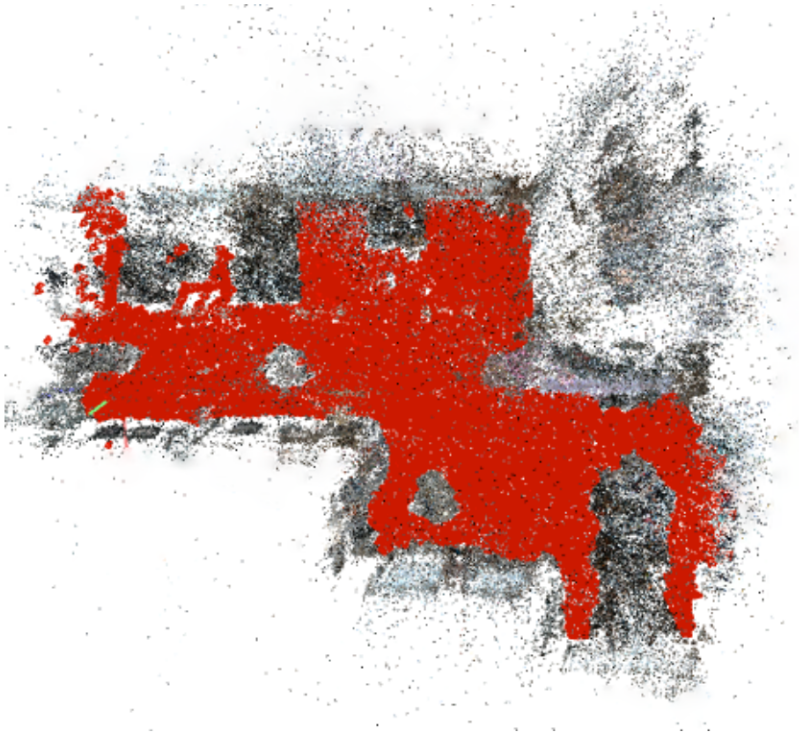


Figure 4.15: Sparse model of our environment with Colmap's SfM pipeline

- **Image Registration**: Starting from a metric reconstruction, new images can be registered to the current model by solving the Perspective-n-Point problem, using feature correspondences to triangulated points in already registered images.

- **Triangulation**: A newly recorded image must observe the existing scene points. Furthermore, it can also increase the coverage of the scene by extending the set of points through triangulation. A new scene point can be triangulated and added to the set of points as soon as at least one other image is registered, which also covers the new part of the scene but from a different point of view.

- **Bundle Adjustment**: Without a further refinement, SfM usually drifts quickly to a non-recoverable state. Bundle adjustment is the joint non-linear refinement of camera parameters and point parameters that minimizes the reprojection error using a function that projects scene points to image space and a loss function to potentially down-weight outliers.

4.6 Cleaning and formatting of data

After the process of collecting data, our dataset contains 7 sessions of pairs. As it

Statistics dataset sessions	
Sessions	**Pairs RSSi-Images**
1 session	9258
2 session	9000
3 session	8980
4 session	9045
5 session	9675
6 session	9335
7 session	9185
Total number of pairs	**64478**

Table 4.1: Number of pairs in each session.

showed in table 4.1 the dataset contains a total number of 64478 image-RSSI pairs. After collecting these data, we used all the images for the creation of 3D model using COLMAP, which used 57158 images registered and the same total number of positions. A 3D model obtained by SfM lacks scale information, and its coordinates are not related to the real-world ones. To realize our localization system, we need to convert the coordinate system to one that can be understood. To do this, we have used an approach which insert as input some images (at least 3) in the model to scale it in real world coordinates. These images are first localized in the 3D model, then the similarity transformation between the obtained locations in the model and the images' real-world coordinates are computed, re-scaling all the positions with RANSAC [59](used in COLMAP).

4.6.1 Coordinates extraction

COLMAP return three files extracted from the model:

- *camera.txt*: this file contains the intrinsic parameters of all reconstructed cameras in the dataset,
- *images.txt*: this file contains the pose and key points of all reconstructed images in the dataset

- *points3D.txt*: this file contains the information of all reconstructed 3D points in the dataset

We used the images.txt file that gives me a quaternion (QW, QX, QY, QZ) and a translation vector (TX, TY, TZ). The coordinates of the projection/camera center are given by $-R^t * T$, where

- R^t is the inverse/transpose of the 3x3 rotation matrix composed from the quaternion
- T is the translation vector

The local camera coordinate system of an image is defined in a way that the X axis points to the right, the Y axis to the bottom, and the Z axis to the front as seen from the image. In a top point of view, we have the 2 DoF (X and Z) and consequently the target position of each point.

4.6.2 Interpolation of missing RSSI and splitting of dataset

While the sanbot moving around the environment, possibly not all the antennas sent the payload. Furthermore the missing target positions are calculates with linear interpolation because not all the images are registered in the 3D model of COLMAP. To solve these possible errors we filled the missing values with the linear interpolation. The linear interpolation calculates the average between the previous and the next value. It is important having a lot of data as we will show in the next section. Furthermore due to nature of RSSI's, that have e temporal variability, we split the all dataset sequentially, in training, validation and test set.

4.6.3 Normalization of Data

We process RSSI data using a sliding window of one second: at every second we collect the RSSIs information of the five antennas dislocated in the environment. In order to do this, we create several csv files of seven columns, when each row represents a second: in the first 5 columns we have the RSSIs from the 5 antennas (one column - one antenna), the last two represent the target position in that second.

To mitigate noise and signal instability, we normalized data to have 0 mean and standard deviation equal to 1, using the formula:

$$\overline{x_i} = \frac{x_i - \mu_i}{\sigma_i} \tag{4.3}$$

where x_i is the i-th feature (i.e., the value of the i-th antenna), μ_i and σ_i are the mean and standard deviation values of the features computed on the training set, and $\overline{x_i}$ is the i-th normalized feature. The normalization in this way has been a goo method to have better perfomrance of our method.

4.7 Experiments

In this section we describe all the experiments used to address the localization activity. Due to the temporal variability of our input, the raw RSSI, we use in most of the experiments performed, the recurrent neural networks as a solution of deep learning models to predict positions.. Our sequential dataset makes it possible to use the RNN. and all the experiments were performed using the PyTorch footnote https://pytorch.org/ framework.

4.7.1 Brief history of Recurrent Neural Networks

A recurrent neural network (RNN) is a type of artificial neural network which uses sequential or time series data. While traditional deep neural networks assume that inputs and outputs are independent of each other, the output of recurrent neural networks depend on the prior elements within the sequence, because the future events can be the output of a given sequence [60, 61, 62]. These algorithms, therefore, not only keep track of current input value but also of historical data. The simplest recurrent neural network introduced in paper [63] in 1990, is the Elman's RNN, its key idea is to unfold the network through the time. Although the training phase of this architecture can arise the vanishing gradient problem, it a naive method to address the problem. To overcome this problem, in 1997 an architecture is presented which takes the name of Long Short Term Memory (LSTM) [64]. The key idea of this architecture is a cell state that preserving the historical sequential data, thus LSTM is capable of learning long-distance dependencies. While a simple RNN uses only

Table 4.2: Comparison of our method against baselines of experiments

Method	**FC**	**Temporal Window**	**Mean error**
KNN	-	/	1.99 m
MLP	3	3s	1.80 ± 1.25 m
RNN	2	20s	1.32 ± 0.91 m
GRU	1	15s	1.35 ± 0.94 m
GRU	2	15s	1.31 ± 0.93 m
GRU	3	15s	1.28 ± 0.95 m
LSTM	1	20s	1.64 ± 1.07 m
LSTM	2	20s	1.29 ± 0.88 m
LSTM	3	20s	1.25 ± 0.90 m
BiGRU	3	20s	1.84 ± 1.02 m
BiLSTM	3	20s	1.60 ± 0.98 m
OURs	1	20s	2.18 ± 1.45 m
OURs	2	20s	1.44 ± 0.99 m
OURs	3	20s	1.22 ± 0.93 m
OURs	4	20s	$\mathbf{1.17 \pm 0.90}$ **m**

one layer, LSTM uses four layers that interact in a specific way. In this way, LSTM has the ability to remove or add information to the cell state which is regulated by structures called gates. Another architecture that aims to solve the vanishing gradient problem is the Gated Recurrent Unit (GRU), presented in 2014 in paper[65]. This architecture can also be considered as a variation on the LSTM because both are designed similarly and, in some cases, produces equally excellent results. To do that, the network use the gate and the reset gate. Basically, these are two vectors which decide what information should be passed to the output. Since it uses fewer gates and does not have a memory cell, in training time it has numerous advantages, and in many cases it has the same performance as an LSTM. A compounded model of LSTM is the Bidirectional LSTM (BiLSTM). It is a sequence processing model that consists of two Long-short-term-memory,the first one taking the input in a forward direction, and the other in a backward one. The same concept can be used for the Gated recurrent unit (BiGRU). In the following section we will describe the details about the RSSI indoor localization task using this type of models. In order to use the RNN we have to present to these architectures temporal data, and the

nature of RSSI is suitable. We investigate the localization task using the Recurrent neural network and neural network to predict the 2 DoF positions.

4.7.2 K-Nearest Neighbors

The simplest method is K-nearest neighbour (KNN), which is a classification algorithm that allows us to classify data by comparing it with a pre-existing set of labeled data (the training set). Given a labeled dataset $X = (x_i, y_i)_{i=1}^{N}$ where x_i represent the samples and the y_i the relative labels, for a new test sample $\overline{x}$ to classify, the algorithm tries to find the first K samples of the training set most similar to $\overline{x}$ according to some measure of similarity δ. The KNN algorithm implicitly subdivides the data space into decision regions, where the decision value in a particular point of the space depends on the class of belonging of the nearest neighboring K. Our aim, however, is to illustrate and emphasize how KNN can be equally effective when the target variable is continuous in nature, as well as in our case. We use, so, K-nearest -neighbour for a regression task, using the information of 5 antennas to regress the two degrees of freedom - target location. We were aware to expect that the performances were not that good, but the algorithm gave us a top boundary for the error. From this section on we will calculate the error in meters, in accord with the Euclidean distance. Using the best K the result of KNN gave us an error of about 2 meters as showed in table of experiments 4.2. This large error is probably due to the inability of the algorithm to model the temporal nature of observations

4.7.3 Multi Layer Perceptron

A Multi Layer Perceptron is a deep network, composited in layers, where a node in a level is connected with all the node in the next layers. This architecture can be composed by three stages:

- **Input layer:** level in which the input data will be presented;
- **Hidden layer/s:** one or more levels which will contain latent representations of the data;
- **Output layer:**level which will contain the output data.

Our MLP baseline with the best ablation of the model, was composited by an input layer, two hidden layers and an output layer, as showed in figure 4.16 This baseline performs slightly better result than KNN, with an average error of 1.80 meters. Note that while MLP can model more complex input-output relationships than KNN, it cannot really exploits the temporality of observations, in fact we just used 3 seconds at each training step, trying to exploit a minimal temporal nature of the data.

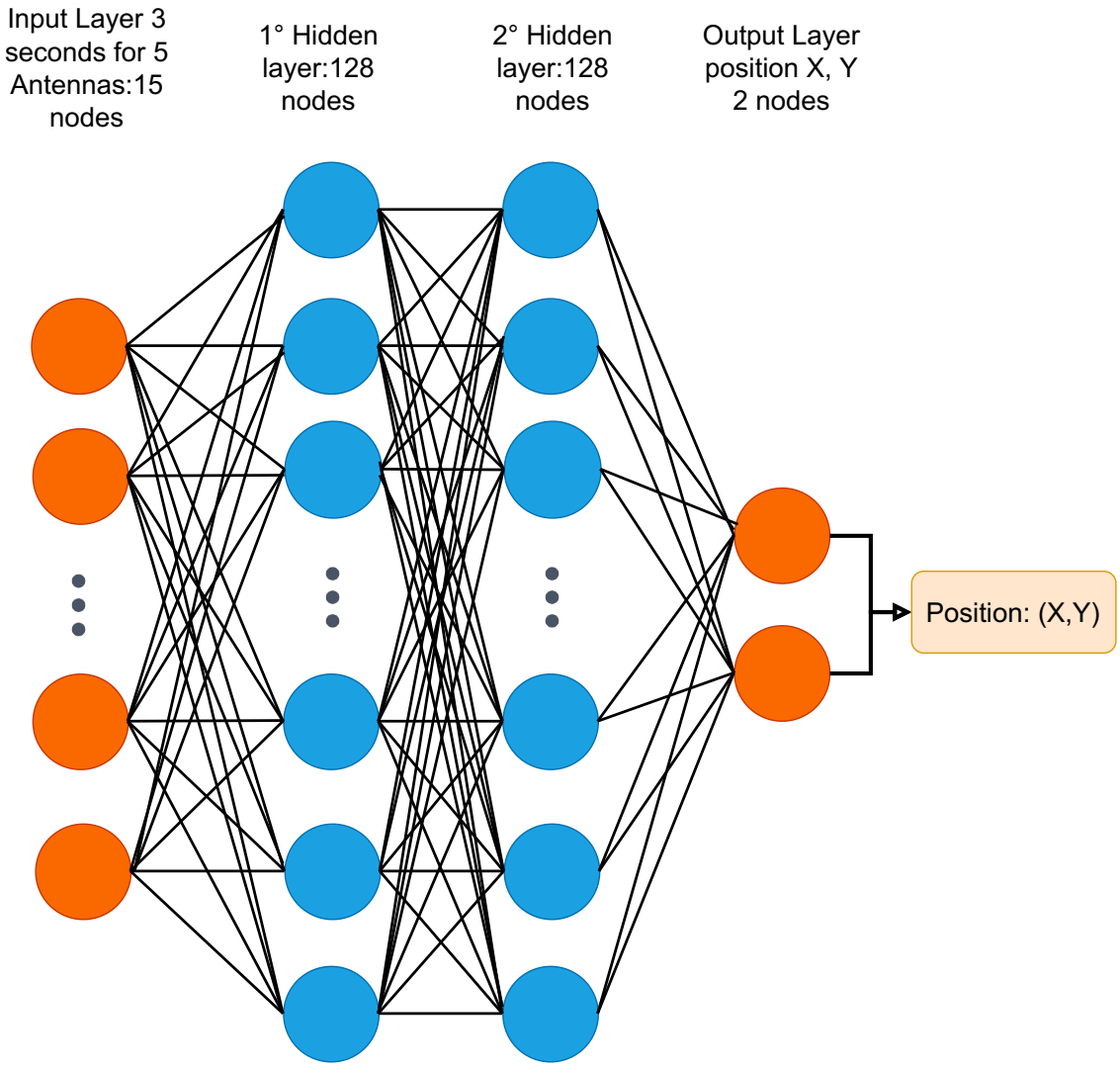


Figure 4.16: Multi Layer Perceptron Architecture

4.7.4 Elman's Recurrent Neural Network Architecture

The main characteristic of all the neural networks such as Multi Layer Perceptron (MLP) or a Convolution Neural Network (CNN), is that they have no memory. Each input is processed independently, which means that these networks do not take advantage of the sequential nature of the data (such in our input case). The first successful example of a recurrent network trained with back-propagation was the Elman's Recurrent Neural network, introduced by Jeffrey Elman Network [63] in 1990. This was a highly influential work for in cognitive science. Elman was focused with the problem of representing sequences or time as input in neural networks. This architecture can be thought of as a graph of RNN cells 4.17, where each cell (neuron) performs the same operation on each element of the sequence. The RNN cells incorporate this dependence thanks to a state, or memory, which maintains what has been seen up to that moment. At time t the cell has an input x_t and an output o_t. A part of the output o_t (the state s_t) is re-entered into the cell to be used at time $t+1$. The state at time t (i.e. s_t) is a function of the input at

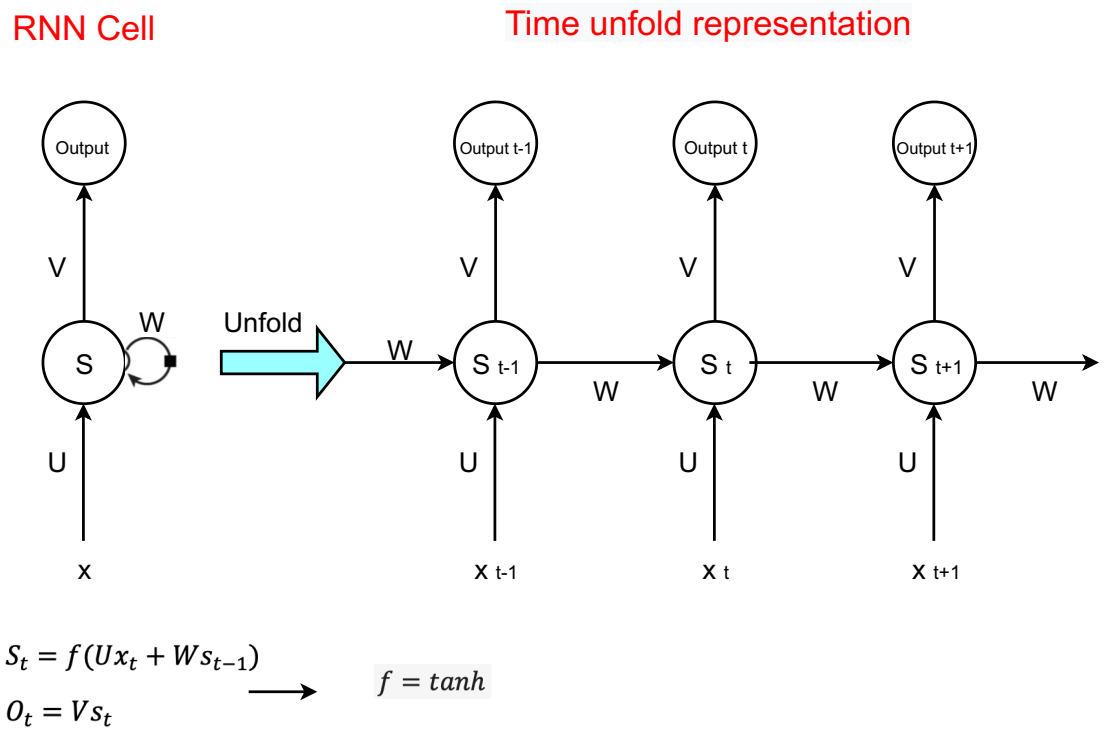


Figure 4.17: Elman's Architecture

current time t and of the state at previous time $t-1$. The parameters of the RNN are represented by three matrices of weights U, V, W, corresponding respectively to the input, output and status. These weights are shared across the steps (this allows to reduce the number of parameters that an RNN has to learn). Each output at time t produces a loss, and the total loss (the one we have to optimize) is the sum of all the losses $L = \sum_t^T L_t$. The gradient descent method is used as a learning method (i.e. calculate gradients with respect to all weights). The algorithm is the **Backpropagation Through Time - BPTT**: the gradient at each output depends not only on the current time t but also on all the previous ones. During the feed-forward step the network produces the outputs which are compared with the real labels to compute the loss at each time t. During the backpropagation the gradients of the loss with respect to the parameters U, V, W are computed at each time t, and the parameters are updated with the sum of the gradients. The updating of the gradients of the matrix's weights is:

$$\begin{cases} \frac{\partial L_t}{\partial U} = \sum_{k=0}^{t} \frac{\partial L_t \partial o_t \partial s_t \partial s_k}{\partial o_t \partial s_t \partial s_k \partial U} & U \\ \frac{\partial L_t}{\partial V} = \sum_{k=0}^{t} \frac{\partial L_t \partial o_t \partial s_t \partial s_k}{\partial o_t \partial s_t \partial s_k \partial V} & V \\ \frac{\partial L_t}{\partial W} = \sum_{k=0}^{t} \frac{\partial L_t \partial o_t \partial s_t \partial s_k}{\partial o_t \partial s_t \partial s_k \partial W} & W \end{cases} \tag{4.4}$$

This RNN has two major drawbacks [66, 67, 68]:

- **Vanishing gradient problem:** this problem occurs when the gradients are so small (in general less than one), such a long dependency will be hard to learn for a RNN where gradients vanish as we move backward in the network neural network,
- **Exploding gradients:** this problem gradient problem will completely derail the learning process. In very deep networks this is often a problem because multiple levels amplify the effect of large gradients, transforming into very large updates to the weights of the network, until the values explode completely.

In our experiment we use the Elman's recurrent neural network followed with a MLP composited of two linear layer, such is showed in figure 4.18

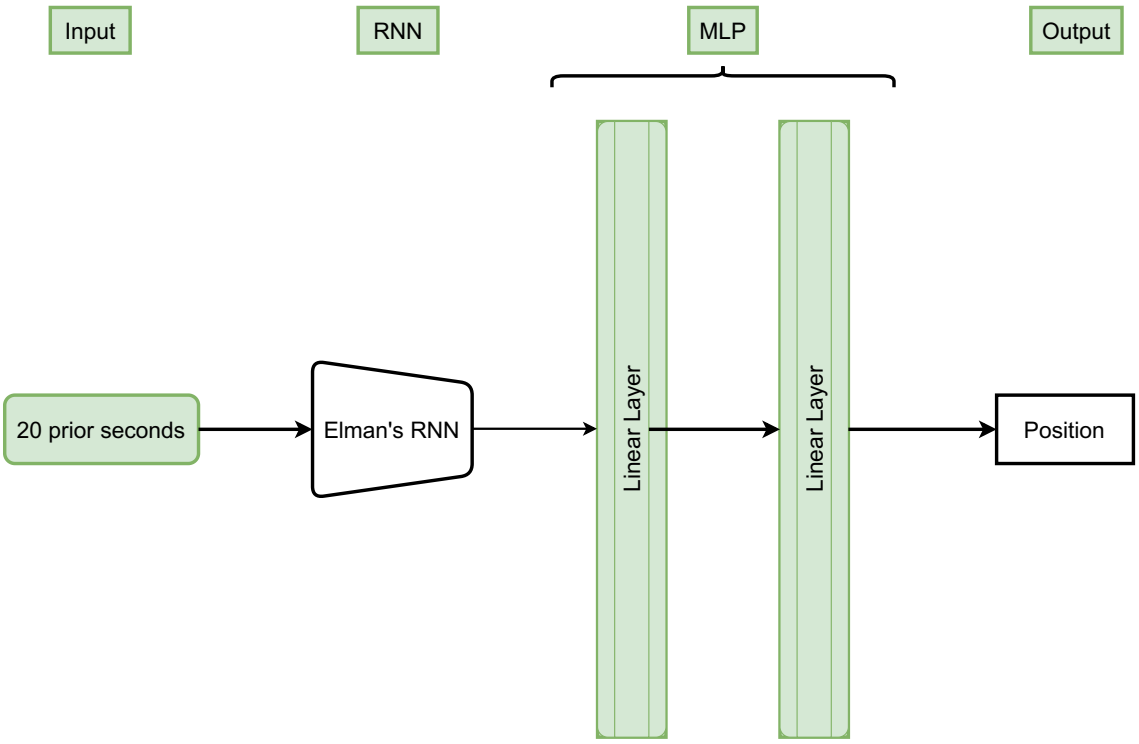


Figure 4.18: RNN experiment

4.7.5 Long Short Term Memory

In addition to vanishing and exploding gradients problems, we have the fact that forward computation is slow, as RNNs cannot compute in parallel: to preserve time dependencies across tiers, each tier must be computed sequentially, which of course requires more time. This architecture has proven effective in solving simple problems, but as sequences have increased in size and complexity, this type of network is slow to train and hard to learn. Different techniques and approaches have tried to solve the aforementioned problems, but the architecture that really advanced the field of the recurrent neural network was the Long Short-Term Memory (LSTM) network. As the name suggests, the defining feature of LSTMs is the addition of a drive that combines short and long memory capacity. While an Elman RNN uses only one *tanh* (Hyperbolic tangent) layer, LSTM uses different layers that interact in a specific way.

- **Cell state:** represents the internal memory of the unit

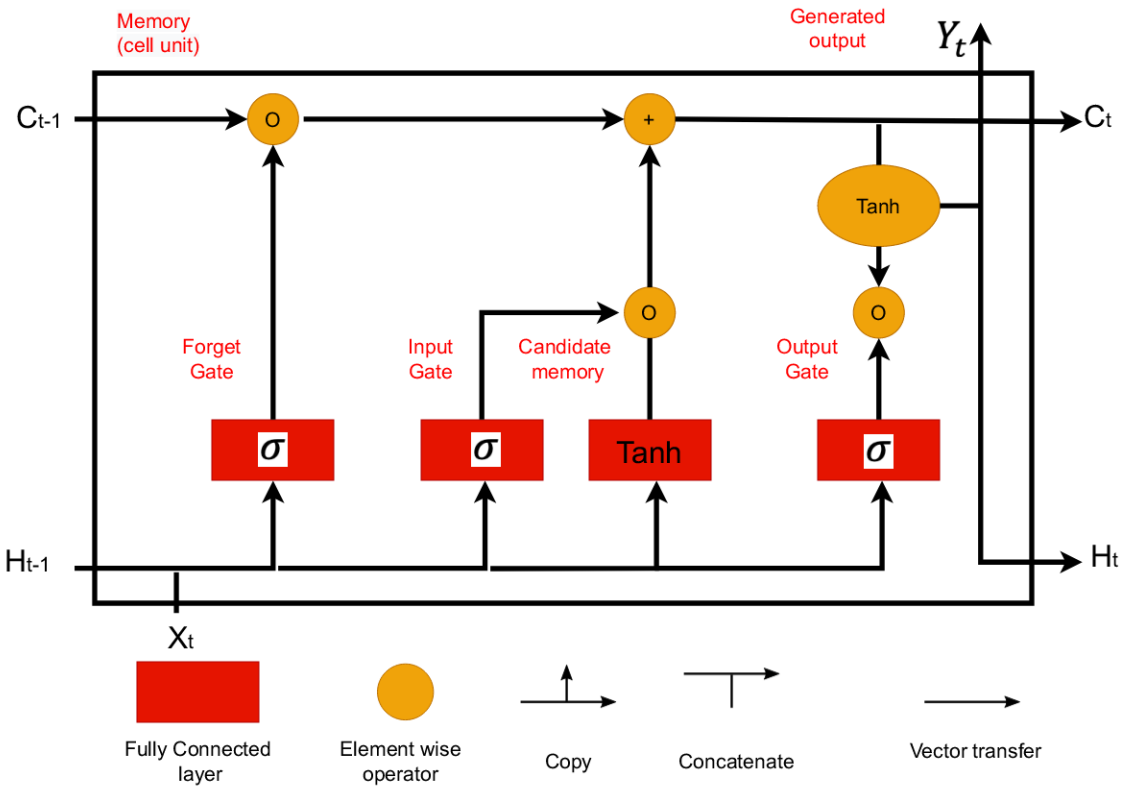
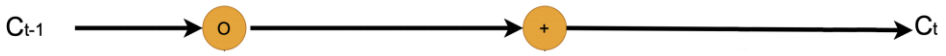


Figure 4.19: Long Short Term Memory Architecture



The first element wise operator determines what to forget or keep from the current state.The second element-wise operator determines how to update the current state given a new input

- **Gate units:** regulate the information that must be kept or removed from the state of the cell. They are characterized by a sigmoid layer and a punctual multiplication operation: **Input gate**, **Forget gate**, **Output gate**

  – **Input gate:** allows to decide what new information should be added to the state, the *sigmoid* and *tanh* calculate the new information candidate to be added to the state, so $i_t = \sigma(W_i * [h_{t-1}, x_t] + b_i)$ where the b_i is the bias,

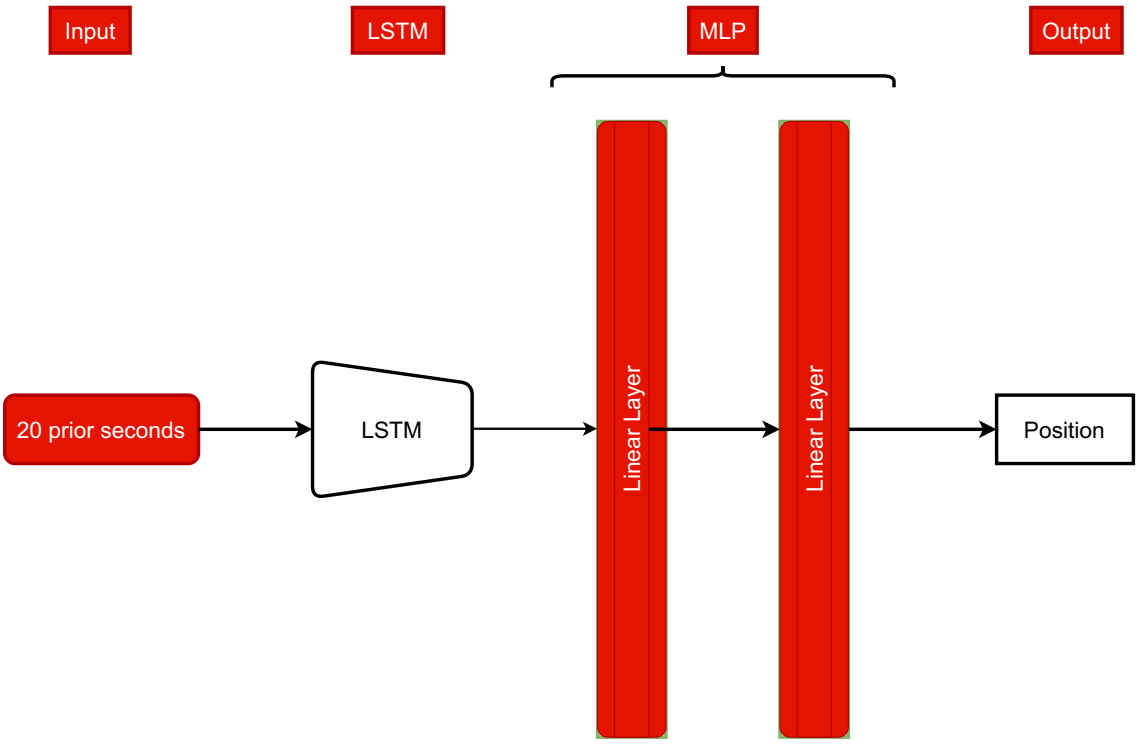


Figure 4.20: Experiment that involves a single LSTM followed by a MLP

- **Forget state:** allows to decide which information will be removed from state of the cell, i.e. how much to keep of the previous state, so $f_t = \sigma(W_f * [h_{t-1}, x_t] + b_f)$ where the b_f is the bias,
- **Output gate:** allows to decide which parts of the cell status to output, so $o_t = \sigma(W_o * [h_{t-1}, x_t] + b_o)$ where the b_o is the bias

Then the cell status update is as follows:

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \tag{4.5}$$

where:

- f_t is the forget layer
- $Ct - 1$ is the content of the cell
- i_t is the input layer

- $\hat{C}_t$ is the new information calculated in this way $\hat{C}_t = tanh(W_c[h_{t-1}, x_t] + b_c$ where the b_c is the bias.

At the end the output of LSTM is calculated in this way:

$$Y_t = o_t * tanh(C_t) \tag{4.6}$$

The experiment which involves a single LSTM, whose output is the input of the MLP formed by two levels, is shown in the figure 4.20 with an error of 1.22 meters

4.7.6 Gated Recurrent Unit

Many architectural variants have been developed to solve the problem of evanescent gradients or explosive gradients, which are often encountered during the operation of a recurrent Elman neural network. One of the most famous variants is the Long Short Term Memory (LSTM) while one of the lesser known but equally effective variants is the Gated Recurrent Unit Network (GRU) [65]. Unlike the LSTM, the GRU is composed of only three doors and does not maintain an internal cell state. The main difference between the two architectures is in the internal functioning of each recurrent unit since Gated Recurrent Unit networks are made up of gates that modulate the current input and the previous hidden state.

The different gates of a GRU are as described below:

- **Update Gate - Z_t :** determines how much of the $t-1$ time knowledge is to be transmitted into the t time. It is similar to the out gate of a LSTM,
- **Current Memory Gate - H_t :** it is often neglected during a typical GRU discussion. It incorporated into the Reset Gate and is used to introduce some non-linearity into the input. it is reduce the effect that previous information has on the current information that is being passed into the t time,
- **Reset Gate - R_t:** determines how much of the past knowledge to forget. It is similar to grouping the Input Gate and Forget Gate of a LSTM.

How a GRU works

how a GRU works? We can summarize its operation in 4 steps

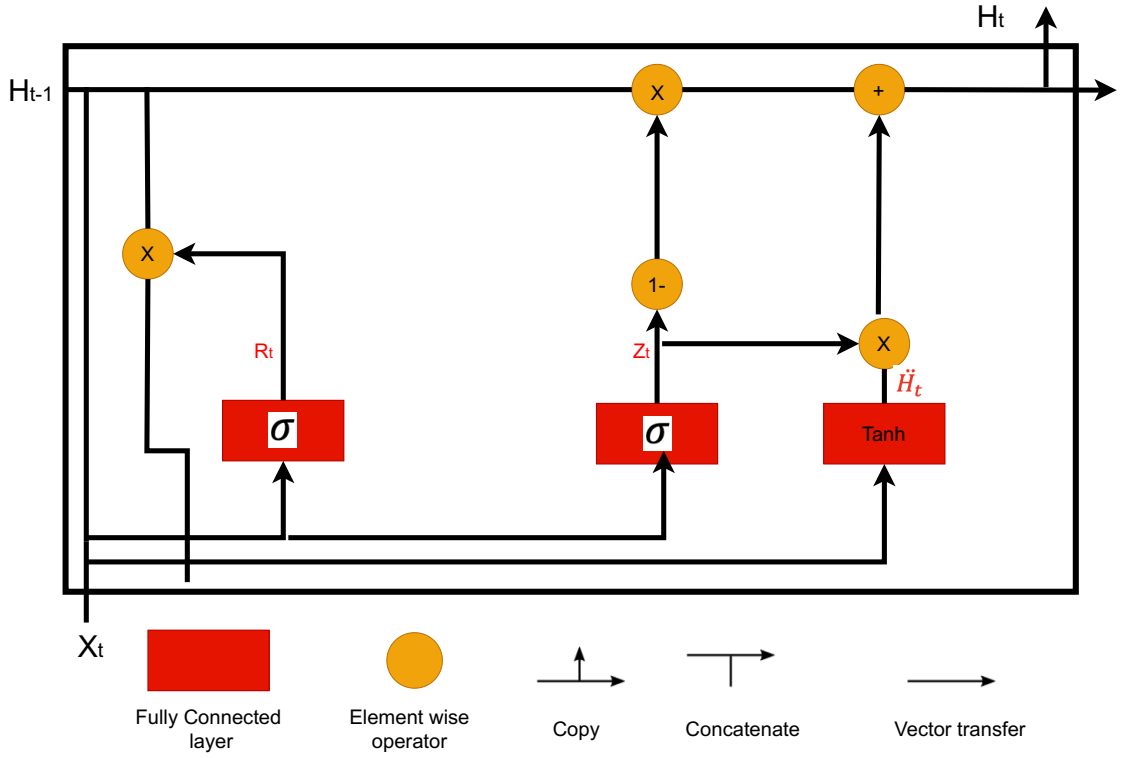


Figure 4.21: Gated Recurrent Unit Architecture

- **Step 1:** take the current input and the previous hidden state as vectors,
- **Step 2:** calculate the values of the three different gates
- **Step 3:** calculate the Current Memory Gate. It is a little different from the other two gates. First of all, the Hadamard product (is a binary operation which, for two matrices of the same size, associates another matrix, of the same size, and where each coefficient is the term-term product of the two matrices) is calculated between the Reset Gate and the previously hidden state vector. Then this vector is parameterized and then added to the parameterized current input vector: $\hat{h_t} = tanh(w * (r_t * ht - 1 + W * x_t)$
- **Step 4:** To calculate the current hidden state, GRU first defines a unit vector of the same size as the input one. Then is calculate the Hadamard Product of the update gate and the previously hidden state vector and generate a new vector by subtracting the update gate from ones, After the GRU calculates the

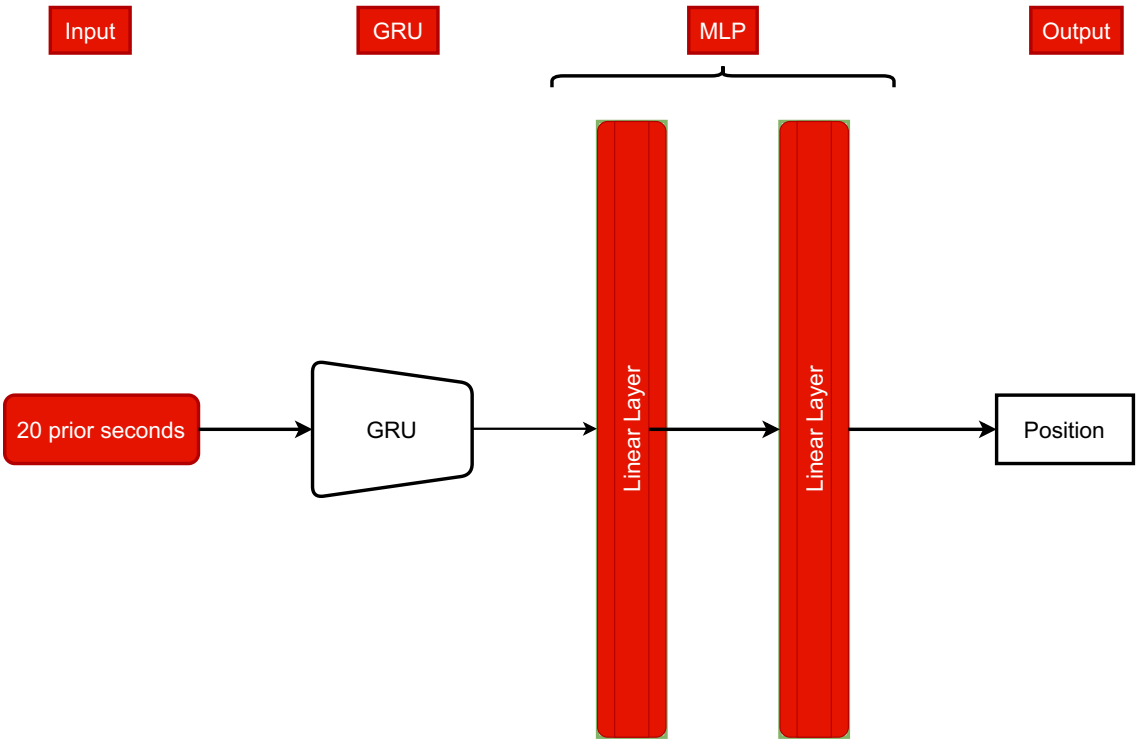


Figure 4.22: Experiment that involves a single GRU followed by a MLP

Hadamard Product of the newly generated vector with the current memory gate. Finally, add the two vectors to get the currently hidden state vector: $h_t = z_t * h_{t-1} + (1 - z_t) * \hat{h}_t$

The experiment which involves a single GRU, whose output is the input of a MLP formed by two levels, is shown in the figure 4.22 with an error of 1.24 meters

4.8 Method

In this section we describe the model used to address the localization task starting from the raw RSSis data. As we said before, raw RSSIs are radio wave signals with a lot of noise, with several difficulties to predict the two degree of freedom position (x and y). Although this radio-wave signal has several disadvantages, raw RSSI is easily to obtain with low costs and low energy with this type of infrastructure 4.2. As said in section 4.7.5, thanks to its cell memory, this type of model has in general

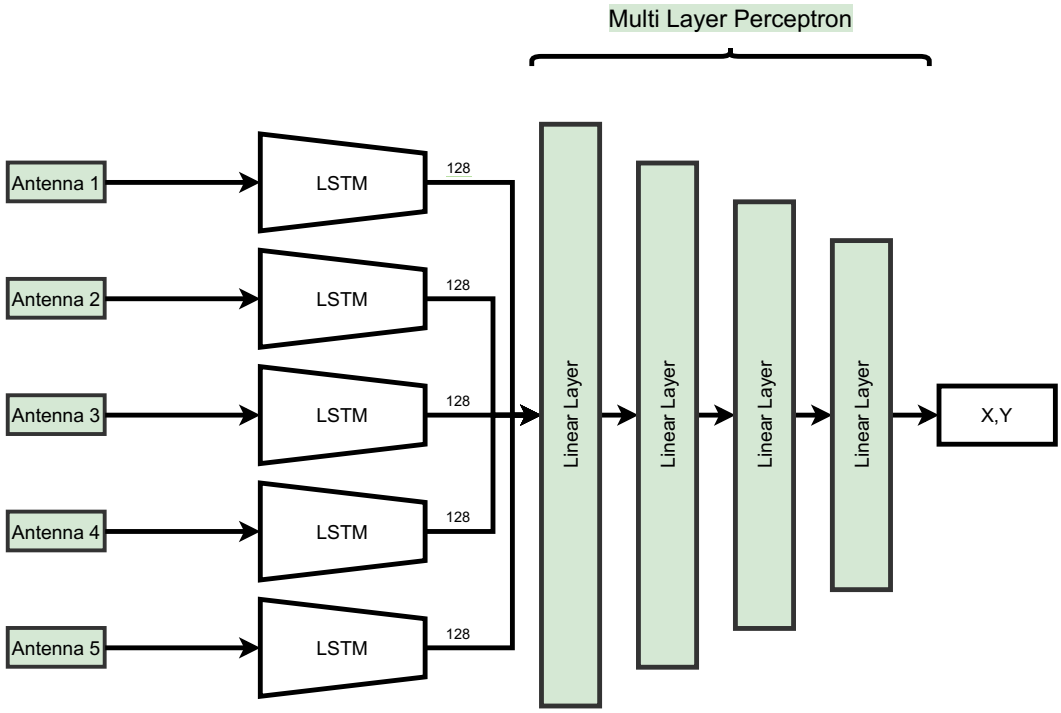


Figure 4.23: Scheme of proposed model

better performance than the GRU 4.7.6; using this advantage, our model used some LSTMs. The remaining part of the model was composited with a MLP.

4.8.1 Proposed Model

Our model, is a neural network architecture which exploits the temporal nature of the data and the different contribution of each antenna, which is illustrated in Figure 4.23. Specifically, we design an architecture composed by five LSTMs, one for each antenna, to process in parallel features related to the different antennas. The key idea is to use the different contributions of each gateway. In training time, at each training step, every LSTM takes as input a sequence containing the RSSI signals of the last 20 seconds measured with respect to each corresponding antenna. The 128-dimensional hidden vectors of the different LSTMs are then concatenated in a single one and fed to a Multi Layer Perceptron (MLP) made of 4 fully connected layers to regress the final 2D pose. To train the model, we used the smoothed L1 loss

Table 4.3: Average error of the proposed method for varying training set size

Portion of training set	**Error**
25%	2.01 ± 1.33 m
50%	1.46 ± 1.03 m
75%	1.30 ± 0.92 m
100%	$\mathbf{1.17 \pm 0.90}$ **m**

function [69], which is closely related to Huber-Loss. We choose this loss because it is less sensitive to outliers compared to Mean Squared Error in a regression task. We used a learning rate of 10^{-4}, which is halved every 100 epochs. We used the Adam optimizer to train the model because it can handle sparse gradients on noisy problem. As showed in table 4.2 with this model we achieved the best result of 1.17 meters of error.

4.9 Evaluation of the proposed method

In this section we are going to discuss of the evaluation of our proposed method, that achieved an error of $1.17m^2$. One of the main contributions of this localization system is the large amount of our dataset. The choice of having several observations is not casual, and it is given by the fact that having more data the system improves its performances. As shown in table 4.3 we ran our model with different portions of our training set. The table highlights that having more data is important in this type of task, a localization system starting from RSSI observations. Furthermore, this table proves what we have said in this chapter,that a machine-learning based solution better cope with a noisy input. Our proposed method approaches this problem as a regression task. The better visible evaluation of a regression task is given by the rec curve, as depicted in figure 4.24 As showed in figure we achieve a small area over curve, with the same value of our error in euclidean distance with a big area under curve. This means that our system performs well in most cases.

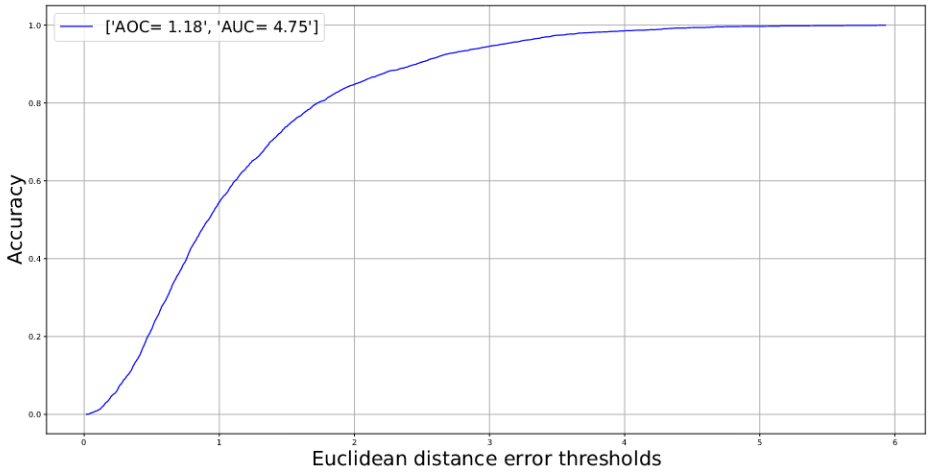


Figure 4.24: Rec curve

4.10 Discussion

In this chapter we described two main contributions:

- A new pipeline to collect a lot of data for a dataset composited of RSSIs
- A model to address the localization task starting from raw RSSI.

Our pipeline described in 4.4 uses the contribution of computer vision for the construction of the 3D model, used to extract the target positions. We collect a dataset of over 64,000 observations for a total of nearly 4 hours of video recording. The motivation for this large dataset is that having more many observations can give us better performance, which reflects the potential of our pipeline. We performed several experiments using mainly the recurrent neural network. Our best method and its ablation is our proposed method 4.8.1 which uses a set of LSTMs, to predict target positions, exploiting the temporal sequence of the input. This type of problem can be faced with other types of more modern architectures, such as transformers, which take a sequence as input and transform it into the desired output, in accordance with the supervised learning paradigm. Typically transformers use a low-level sequence (such as a low-level representation of an image that matches the output of a CNN) and through an attention module, the transformers return the

desired output. Our proposed method that uses a set of LSTM, can be replaced with transformers.

Chapter 5

Conclusion

In this thesis we tackle the problems of spatial understanding 3.2, that is to find an exit from any building; and localization task 4.1.1,the problem to localize a beacon in movement, both in indoor environments. Chapter 2 describe the relative works of the aforementioned problems, introducing the most methods and the new datasets created in literature. Chapter 3, describes a study composed of different approaches to make an intelligent system aware of the surrounding space, in order to create episodes of navigation inside in indoor environments. Our investigation highlighted the following:

1. A dataset consisting of images created from a virtual environment in Unity, representing several buildings with indoor environments

2. Several approaches to tackle the problem of find an exit from any environment using first person vision.

Chapter 4 describes the recognition of the position of an object from a radio signal such as RSSI, identifying a beacon in movement in an indoor environment. Unlike previous works we present several contributions such as creating a pipeline to collect a dataset with many observations, demonstrating that having more data improves the localization system of an object, such as a beacon. Our findings are as follows:

1. A pipeline to collect a lot of data of radio wave signals such as RSSI as mentioned above.

2. A dataset of RSSI to tackle the localization task where we ran our experiments.

3. We have described a survey on recurrent neural networks, carrying out several experiments to address the localization activity, thanks to the exploitation of the temporal variability of the input.

4. We also show that having a large amount of observations improves accuracy in this type of task.

5.0.1 Contributions and future works

In this thesis we studied the problems of spatial understanding and localization task, both in indoor environments The chapter of spatial understanding, introduced a new dataset and several approaches to address the problem. We believe that with contributions the research moves forward to find diverse episodes of navigation strategies using the spatial understanding. In chapter 4 we have seen that the RSSI radio wave signal introduces many advantages in terms of cost and availability. Although this signal has several advantages, it has two drawbacks such as ambiguity and instability which affects the signal with fluctuations and consequent noise. These drawbacks can be limited by adopting solutions based on machine learning that better cope with this limitations. We think the new pipeline and baseline introduced in the aforementioned chapter can help to make progress in this area of research, adopting new methodologies to get even better results, leveraging raw RSSI signals to locate an object such as a beacon. Positioning is also one of the cardinal principles of many other tasks, if the localization through RSSI improves and perfects itself as an established methodology, it could also help research on the mobility front through intelligent systems, which tasks span indoor environments. We believe that the pipeline described and the released dataset can help the research in this topic.Although the chapter 3 shows different approaches to move an autonomous robot towards the exit of any building, this problem still remains open. Using the different approaches shown on a different dataset, it may be possible to have a working baseline, thus showing the progress made by the research.

Bibliography

[1] G. Puglisi, D. D. Mauro, A. Furnari, L. Gulino, and G. M. Farinella. "Visual RSSI fingerprinting for radio-based indoor localization". In: *International Conference on Signal Processing and Multimedia Applications (SIGMAP)*. 2022. URL: https://iplab.dmi.unict.it/VisualRSSI/.

[2] Z.-Q. Cheng, J.-X. Li, Q. Dai, X. Wu, and A. G. Hauptmann. "Learning spatial awareness to improve crowd counting". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 6152–6161.

[3] X.-B. Nguyen, G. S. Lee, S. H. Kim, and H. J. Yang. "Self-supervised learning based on spatial awareness for medical image analysis". In: *IEEE Access* 8 (2020), pp. 162973–162981.

[4] M. Rosano, A. Furnari, L. Gulino, and G. M. Farinella. "On embodied visual navigation in real environments through habitat". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 9740–9747.

[5] M. Lochner, C. Sennersten, A. Morshed, and C. Lindley. "Modelling Spatial Understanding: Using knowledge representation to enable spatial awareness in a robotics platform". In: *The 6th International Conference on Advanced Cognitive Technologies and Applications*. Citeseer. 2014, pp. 689–699.

[6] H. Zender, P. Jensfelt, O. M. Mozos, G.-J. M. Kruijff, and W. Burgard. "An integrated robotic system for spatial understanding and situated interaction in indoor environments". In: *AAAI*. Vol. 7. 2007, pp. 1584–1589.

[7] I. Vasiljevic, N. Kolkin, S. Zhang, R. Luo, H. Wang, F. Z. Dai, A. F. Daniele, M. Mostajabi, S. Basart, M. R. Walter, et al. "Diode: A dense indoor and outdoor depth dataset". In: *arXiv preprint arXiv:1908.00463* (2019).

[8] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus. "Indoor segmentation and support inference from rgbd images". In: *European conference on computer vision*. Springer. 2012, pp. 746–760.

[9] J. Xiao, A. Owens, and A. Torralba. "Sun3d: A database of big spaces reconstructed using sfm and object labels". In: *Proceedings of the IEEE international conference on computer vision*. 2013, pp. 1625–1632.

[10] A. Chang, A. Dai, T. Funkhouser, M. Halber, M. Niessner, M. Savva, S. Song, A. Zeng, and Y. Zhang. "Matterport3d: Learning from rgb-d data in indoor environments". In: *arXiv preprint arXiv:1709.06158* (2017).

[11] A. Dai, A. X. Chang, M. Savva, M. Halber, T. Funkhouser, and M. Nießner. "Scannet: Richly-annotated 3d reconstructions of indoor scenes". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 5828–5839.

[12] B.-S. Hua, Q.-H. Pham, D. T. Nguyen, M.-K. Tran, L.-F. Yu, and S.-K. Yeung. "Scenenn: A scene meshes dataset with annotations". In: *2016 fourth international conference on 3D vision (3DV)*. Ieee. 2016, pp. 92–101.

[13] I. Armeni, O. Sener, A. R. Zamir, H. Jiang, I. Brilakis, M. Fischer, and S. Savarese. "3d semantic parsing of large-scale indoor spaces". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 1534–1543.

[14] J. Straub, T. Whelan, L. Ma, Y. Chen, E. Wijmans, S. Green, J. J. Engel, R. Mur-Artal, C. Ren, S. Verma, et al. "The Replica dataset: A digital replica of indoor spaces". In: *arXiv preprint arXiv:1906.05797* (2019).

[15] S. Song, F. Yu, A. Zeng, A. X. Chang, M. Savva, and T. Funkhouser. "Semantic scene completion from a single depth image". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1746–1754.

[16] A Handa, V Patraucean, V Badrinarayanan, S Stent, and R Cipolla. "Scenenet: understanding real world indoor scenes with synthetic data. arXiv preprint (2015)". In: *arXiv preprint arXiv:1511.07041* (2015).

[17] F. Xia, A. R. Zamir, Z. He, A. Sax, J. Malik, and S. Savarese. "Gibson env: Real-world perception for embodied agents". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2018, pp. 9068–9079.

[18] M. T. Hoang, B. Yuen, X. Dong, T. Lu, R. Westendorp, and K. Reddy. "Recurrent neural networks for accurate RSSI indoor localization". In: *IEEE Internet of Things Journal* 6.6 (2019), pp. 10639–10651.

[19] H. J. Jang, J. M. Shin, and L. Choi. "Geomagnetic field based indoor localization using recurrent neural networks". In: *GLOBECOM 2017-2017 IEEE Global Communications Conference*. IEEE. 2017, pp. 1–6.

[20] B. Xu, X. Zhu, and H. Zhu. "An efficient indoor localization method based on the long short-term memory recurrent neuron network". In: *IEEE Access* 7 (2019), pp. 123912–123921.

[21] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni. "A survey on wireless indoor localization from the device perspective". In: *ACM Computing Surveys (CSUR)* 49.2 (2016), pp. 1–31.

[22] F. Zafari, A. Gkelias, and K. K. Leung. "A survey of indoor localization systems and technologies". In: *IEEE Communications Surveys & Tutorials* 21.3 (2019), pp. 2568–2599.

[23] C. Wu. "Towards linear-time incremental structure from motion". In: *2013 International Conference on 3D Vision-3DV 2013*. IEEE. 2013, pp. 127–134.

[24] S. Subedi, G.-R. Kwon, S. Shin, S.-s. Hwang, and J.-Y. Pyun. "Beacon based indoor positioning system using weighted centroid localization approach". In: *2016 Eighth International Conference on Ubiquitous and Future Networks (ICUFN)*. IEEE. 2016, pp. 1016–1019.

[25] S. G. Obreja, T. Aboul-Hassna, F. D. Mocanu, and A. Vulpe. "Indoor localization using radio beacon technology". In: *2018 International Symposium on Electronics and Telecommunications (ISETC)*. IEEE. 2018, pp. 1–4.

[26] P. Kriz, F. Maly, and T. Kozel. "Improving indoor localization using bluetooth low energy beacons". In: *Mobile information systems* 2016 (2016).

[27] T. Ishihara, J. Vongkulbhisal, K. M. Kitani, and C. Asakawa. "Beacon-guided structure from motion for smartphone-based navigation". In: *2017 IEEE Winter Conference on Applications of Computer Vision (WACV)*. IEEE. 2017, pp. 769–777.

[28] J. Torres-Sospedra, R. Montoliu, A. Martínez-Usó, J. P. Avariento, T. J. Arnau, M. Benedito-Bordonau, and J. Huerta. "UJIIndoorLoc: A new multi-building and multi-floor database for WLAN fingerprint-based indoor localization problems". In: *2014 international conference on indoor positioning and indoor navigation (IPIN)*. IEEE. 2014, pp. 261–270.

[29] G. M. Mendoza-Silva, P. Richter, J. Torres-Sospedra, E. S. Lohan, and J. Huerta. "Long-term WiFi fingerprinting dataset for research on robust indoor positioning". In: *Data* 3.1 (2018), p. 3.

[30] S. Mahfouz, P. Nader, and P. E. Abi-Char. "RSSI-based classification for indoor localization in wireless sensor networks". In: *2020 IEEE International Conference on Informatics, IoT, and Enabling Technologies (ICIoT)*. IEEE. 2020, pp. 323–328.

[31] S. Sadowski and P. Spachos. "Rssi-based indoor localization with the internet of things". In: *IEEE Access* 6 (2018), pp. 30149–30161.

[32] M. Mohammadi, A. Al-Fuqaha, M. Guizani, and J.-S. Oh. "Semisupervised deep reinforcement learning in support of IoT and smart city services". In: *IEEE Internet of Things Journal* 5.2 (2017), pp. 624–635.

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

[34] X. Sun, Y. Xie, P. Luo, and L. Wang. "A dataset for benchmarking image-based localization". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 7436–7444.

[35] A. Kendall, M. Grimes, and R. Cipolla. "Posenet: A convolutional network for real-time 6-dof camera relocalization". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 2938–2946.

[36] Y. Li, N. Snavely, and D. P. Huttenlocher. "Location recognition using prioritized feature matching". In: *European conference on computer vision*. Springer. 2010, pp. 791–804.

[37] J. Shotton, B. Glocker, C. Zach, S. Izadi, A. Criminisi, and A. Fitzgibbon. "Scene coordinate regression forests for camera relocalization in RGB-D images". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 2930–2937.

[38] Y. Li, D. J. Crandall, and D. P. Huttenlocher. "Landmark classification in large-scale image collections". In: *2009 IEEE 12th international conference on computer vision*. IEEE. 2009, pp. 1957–1964.

[39] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner. "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11 (1998), pp. 2278–2324.

[40] A. Krizhevsky, I. Sutskever, and G. E. Hinton. "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems* 25 (2012).

[41] K. Simonyan and A. Zisserman. "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556* (2014).

[42] K. He, X. Zhang, S. Ren, and J. Sun. "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 770–778.

[43] J. Long, E. Shelhamer, and T. Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440.

[44] P. Bellavista, A. Corradi, and C. Giannelli. "Evaluating filtering strategies for decentralized handover prediction in the wireless internet". In: *11th IEEE Symposium on Computers and Communications (ISCC'06)*. IEEE. 2006, pp. 167–174.

[45] J. Kiefer and J. Wolfowitz. "Stochastic estimation of the maximum of a regression function". In: *The Annals of Mathematical Statistics* (1952), pp. 462–466.

[46] Y Nesterov. "A method of solving a convex programming problem with convergence rate mathcal {O}(1/kˆ {2})". In: *Sov. Math. Dokl.* Vol. 27.

[47] I. Sutskever, J. Martens, G. Dahl, and G. Hinton. "On the importance of initialization and momentum in deep learning". In: *International conference on machine learning*. PMLR. 2013, pp. 1139–1147.

[48] D. P. Kingma and J. Ba. "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980* (2014).

[49] A. Furnari, G. M. Farinella, and S. Battiato. "Temporal Segmentation of Egocentric Videos to Highlight Personal Locations of Interest". In: *International Workshop on Egocentric Perception, Interaction and Computing (EPIC) in conjunction with ECCV, The Netherlands, Amsterdam, October 9*. Vol. 9913. Lecture Notes in Computer Science. Springer Lecture Notes in Computer Science, 2016, pp. 474–489. URL: http://iplab.dmi.unict.it/PersonalLocationSegmentation

[50] S. Battiato, G. M. Farinella, G. Gallo, and D. Ravì. "Spatial hierarchy of textons distributions for scene classification". In: *International Conference on Multimedia Modeling*. Springer. 2009, pp. 333–343.

[51] A. Kamali. "Smart warehouse vs. traditional warehouse-review". In: *Automation and Autonomous Systems* 11.1 (2019), pp. 9–16.

[52] C. Häne, L. Heng, G. H. Lee, F. Fraundorfer, P. Furgale, T. Sattler, and M. Pollefeys. "3D visual perception for self-driving cars using a multi-camera system: Calibration, mapping, localization, and obstacle detection". In: *Image and Vision Computing* 68 (2017), pp. 14–27.

[53] S. Gupta, J. Davidson, S. Levine, R. Sukthankar, and J. Malik. "Cognitive mapping and planning for visual navigation". In: *IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 2616–2625.

[54] F. Ragusa, A. Furnari, S. Battiato, G. Signorello, and G. M. Farinella. "EGO-CH: Dataset and fundamental tasks for visitors behavioral understanding using egocentric vision". In: *Pattern Recognition Letters* 131 (2020), pp. 150–157.

[55] A. Furnari, S. Battiato, and G. M. Farinella. "Leveraging Uncertainty to Rethink Loss Functions and Evaluation Measures for Egocentric Action Anticipation". In: *European Conference on Computer Vision Workshops*. Springer. 2018, pp. 389–405.

[56] S. Ullman. "The interpretation of structure from motion". In: *Proceedings of the Royal Society of London. Series B. Biological Sciences* 203.1153 (1979), pp. 405–426.

[57] J. L. Schönberger and J.-M. Frahm. "Structure-from-Motion Revisited". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016.

[58] J. L. Schönberger, E. Zheng, M. Pollefeys, and J.-M. Frahm. "Pixelwise View Selection for Unstructured Multi-View Stereo". In: *European Conference on Computer Vision (ECCV)*. 2016.

[59] M. A. Fischler and R. C. Bolles. "Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography". In: *Communications of the ACM* 24.6 (1981), pp. 381–395.

[60] J. J. Hopfield. "Neural networks and physical systems with emergent collective computational abilities." In: *Proceedings of the national academy of sciences* 79.8 (1982), pp. 2554–2558.

[61] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Learning internal representations by error propagation*. Tech. rep. California Univ San Diego La Jolla Inst for Cognitive Science, 1985.

[62] M. I. Jordan. "Serial order: A parallel distributed processing approach". In: *Advances in psychology*. Vol. 121. Elsevier, 1997, pp. 471–495.

[63] J. L. Elman. "Finding structure in time". In: *Cognitive science* 14.2 (1990), pp. 179–211.

[64] S. Hochreiter and J. Schmidhuber. "Long short-term memory". In: *Neural computation* 9.8 (1997), pp. 1735–1780.

[65] K. Cho, B. Van Merriënboer, D. Bahdanau, and Y. Bengio. "On the properties of neural machine translation: Encoder-decoder approaches". In: *arXiv preprint arXiv:1409.1259* (2014).

[66] Y. Bengio, P. Simard, and P. Frasconi. "Learning long-term dependencies with gradient descent is difficult". In: *IEEE transactions on neural networks* 5.2 (1994), pp. 157–166.

[67] R. Pascanu, T. Mikolov, and Y. Bengio. "On the difficulty of training recurrent neural networks". In: *International conference on machine learning*. PMLR. 2013, pp. 1310–1318.

[68] G. Philipp, D. Song, and J. G. Carbonell. "The exploding gradient problem demystified-definition, prevalence, impact, origin, tradeoffs, and solutions". In: *arXiv preprint arXiv:1712.05577* (2017).

[69] R. Girshick. "Fast r-cnn". In: *Proceedings of the IEEE international conference on computer vision*. 2015, pp. 1440–1448.

La borsa di dottorato è stata cofinanziata con risorse del
Programma Operativo Nazionale Ricerca e Innovazione 2014-2020 (CCI 2014IT16M2OP005),
Fondo Sociale Europeo, Azione I.1 “Dottorati Innovativi con caratterizzazione Industriale”

UNIONE EUROPEA
Fondo Sociale Europeo