Markov Localization of an Indoor Quadcopter using Deep Learning

Farhang Naderi

Submitted to the
Institute of Graduate Studies and Research
in partial fulfillment of the requirements for the degree of

Master of Science
in
Electrical and Electronic Engineering

Eastern Mediterranean University
September 2021
Gazimağusa, North Cyprus

Approval of the Institute of Graduate Studies and Research

Prof. Dr. Ali Hakan Ulusoy
Director

I certify that this thesis satisfies all the requirements as a thesis for the degree of Master of Science in Electrical and Electronic Engineering.

Assoc. Prof. Dr. Rasime Uyguroğlu
Chair, Department of Electrical and Electronic Engineering

We certify that we have read this thesis and that in our opinion it is fully adequate in scope and quality as a thesis for the degree of Master of Science in Electrical and Electronic Engineering.

Prof. Dr. Hasan Demirel
Supervisor

Examining Committee

1. Prof. Dr. Hasan Demirel

2. Prof. Dr. Mustafa Kemal Uyguroğlu

3. Asst. Prof. Dr. Kamil Yurtkan

ABSTRACT

Localization is among one of the interesting subjects in robotics and can be spread from Unmanned Ground Vehicles (UGVs) to aerial ones. It is a point of interest for instance to localize robots in a warehouse or within an open area to define specific tasks. Unmanned Aerial Vehicles are also being used vastly indoors with GPS-denied environments.

There are many localization methods recently being used in industry and research as such as Ultra-Wide Band (UWB), Bluetooth and (Global Positioning System) GPS. They have their own point of application in industry depending on their specifications. One of the best solutions is UWB with the least number of errors.

In this thesis, we implemented a localization method based on Deep Learning. 16 patterns on the floor are used to make a specific map for localization. The proposed Deep Learning algorithm were able to detect each pattern correctly with 100% accuracy using majority voting for decision making in 3 seconds. The detection is performed real-time with the video feed of 30fps. Training and testing the network is done on Mobilenet which is based on Fast R-CNN deep learning architecture. All the processes are done on the quadcopter itself from navigation, control, and deep pattern detection using a single embedded computer. The quadcopter is equipped with a Raspberry Pi, Google Edge TPU embedded device with a flight controller in addition to a tracking and an RGB camera. The whole decision making of the patterns is performed via the embedded device connected to the Raspberry Pi in 30 fps and no pattern recognition process is employed on the ground computer. The

drone odometry data is acquired via an Intel Realsense camera which provides IMU data to the drone. Only the codes for simple movements over the map have been sent to the drone from the ground station. Heading data is also provided by the tracking camera mounted on the quadcopter.

Markov weights and the final decision weights have 100% confidence after each random path has been travelled over by the quadcopter. The drone was able to localize itself as a kidnapped robot, after flying over an average of two or maximum three patterns.

Keywords: deep learning, markov localization, unmanned aerial vehicle

ÖZ

Lokalizasyon, robotic çalışmaları arasında yer alan ilgi çekici konulardan biridir ve İnsansız Kara Araçlarından (UGV'ler) hava araçlarına kadar kullanılmaktadır. Örneğin, belirli görevleri tanımlamak için bir depoda veya açık bir alanda robotları lokalize etmek önemli bir konudur. İnsansız Hava Araçları da GPS'in olmadığı ortamlarda büyük ölçüde iç mekanlarda kullanılmaktadır.

UWB, Bluetooth ve GPS gibi son zamanlarda endüstride ve araştırmalarda kullanılan birçok lokalizasyon yöntemi bulunmaktadır. Spesifikasyonlarına bağlı olarak endüstride kendi uygulama noktalarına sahiptirler. En iyi çözümlerden biri, en az hataya sahip UWB'dir.

Bu tezde Derin Öğrenmeye dayalı bir lokalizasyon yöntemi uyguladık. Lokalizasyon için belirli bir harita oluşturmak için zeminde 16 örüntü kullanıldı. Önerilen Derin Öğrenme algoritması, 3 saniyenin altında, çoğunluk oylaması yöntemini kullanarak her bir örüntüyü %100 doğrulukla tespit edebildi. Algılama, 30 fps'lik video beslemesi ile gerçek zamanlı olarak gerçekleştirimiştir. Ağın eğitimi ve testi, Fast R-CNN derin öğrenme mimarisine dayalı Mobilenet temel alınarak yapıldı. Navigasyon, kontrol ve derin öğrenme dahil tüm işlemler quadcopter üzerinde gerçekleştirildi. Quadcopter, bir takip ve bir RGB kameraya ek olarak bir uçuş kontrol cihazına sahip bir Raspberry Pi, Google Edge TPU gömülü bir sistem ile donatılmıştır. Modellerin tüm karar verme işlemi Raspberry Pi'ye bağlı gömülü cihaz üzerinden 30 fps'de yapıldı ve yer bilgisayarında herhangi bir örüntü tanıma işlemi yapılmadı. Drone odometri verileri, drone'a IMU verileri sağlayan bir Intel Realsense

kamera aracılığıyla elde edildi. Yer istasyonundan drone'a sadece harita üzerindeki basit hareketler için kodlar gönderildi. Yön verileri ayrıca quadcopter üzerine monte edilmiş izleme kamerası tarafından sağlanmaktadır.

Markov ağırlıkları ve nihai karar ağırlıkları, quadcopter tarafından her rastgele yol kat edildikten sonra %100 güven göstermiştir. Drone, ortalama iki veya en fazla üç örüntü üzerinde uçtuktan sonra, kaçırılmış bir robot olarak kendisini konumlandırmayı başarmıştır.

Anahtar Kelimeler: derin öğrenme, markov lokalizasyonu, insansız hava aracı

DEDICATION

To my family, Prof. Dr. Hasan Demirel, Prof. Dr. Mustafa Kemal Uyguroğlu, Assoc. Prof. Dr. Qasim Zeeshan and everyone who supported me during my whole education.

ACKNOWLEDGEMENTS

The most important is to thank Prof. Dr. Hasan Demirel for not only continuous supervision during my thesis studies by providing valuable ideas, but his dedication to be always supporting me from the beginning of my studies in Eastern Mediterranean University. Where I stand today is highly dependent on having him as my consultant. I am thankful to the head of Electrical Engineering department Assoc. Prof. Dr. Rasime Uyguroğlu for her support as much as possible to make my thesis implementations happen and to let me use the available equipment in the department.

Dronecode Foundation, and the PX4 Autopilot community, with special mention to Hardware Manufacturer Holybro for providing most of the hardware, and to Ms. Jinger Zeng and Mr. Ramòn Roche for their outstanding support, providing the tools and guidance to navigate the intricacies of the open-source community and the drone industry.

Cyprus Robotics company owner Mr. Tezel Celebi had a considerable contribution to this thesis as well which without his help I had to limit my work within a simpler framework.

Finally, I am highly thankful to my family for their support either physically or financially and for providing me the best environment possible to work on my thesis.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

LIST OF ABBREVIATIONS

AOA	Angle of Arrival
AP	Access Point
CNN	Convolutional Neural Network
DR	Dead Reckoning
FCC	Federal Communications Commission
FPE	Finger Print Estimation
GPS	Global Positioning System
IMU	Internal Measurement Unit
INS	Inertial Navigation System
kNN	k-Nearest Neighbour
LLSE	Linearized Least Square Estimation
LOS	Line of Sight
LSTM	Long Short-Term Memory
LTE	Long Term Evolution
NLOS	Non-Line of Sight
PDA	Personal Digital Assistant
PFL	Path Loss-based Fast Localization
RFID	Radio Frequency Identification
RMSE	Root Mean Square Error
RSS	Radio Signal Strength
RSSI	Received Signal Strength Indicator
SHF	Super High Frequency
SSD	Single Shot Detector

SVM	Support Vector Machines
TDOA	Time Difference of Arriva
TOA	Time of Arrival
UWB	Ultra-Wide Band
WPAN	Wireless Personal Area Network
WSN	Wireless Sensor Network

Chapter 1

INTRODUCTION

1.1 Motivation

There had been a huge interest during last years in industry for autonomous vehicles. In academia, this interest has been raised as well especially in Robotics field. Among these vehicles, drones have attracted many researchers from industry and academia for research purposes and to use them. The huge amount of interest has led the industries to use the drones for many different purposes these days. Localization for drones is also one of the areas to pay attention to since there are many methods introduced and still indoor navigation for drones is an attractive research topic.

Deep learning on the other hand has made a huge impact in the industry and indeed in academia being used in a vast number of applications and research papers. This topic has been among highly demanded ones recently and has helped industry to develop traditional methods to more efficient ones.

Adding up these two topics mentioned in the previous paragraphs made this research possible which is about localizing a quadcopter using deep learning with Markov algorithm.

1.2 Thesis Objective

As the drones have taken over in the industry recently and as they are a point of interest, any research about the same topic may open current obstacles. In [1], there

has been special patterns on the floor (here 16 of them) to do localization. The setup proposed in the paper consists of all operation in the simulation environment. The objective in this thesis is to bring this into action by using deep learning.

The proposed method on the original paper has used image analysis with feature extraction from each photo that is considered as a localization pattern. The general idea is, a quadcopter starts flying over a pattern at the beginning without orientation and its place being considered. A map of all patterns and their placement with their relative positions has been saved on the computer. The quadcopter has a camera facing downwards that streams the videos to the main computer. As it is hovering on top of a pattern, the streamed photo is detected via the above-mentioned method an initial guess of the position is produced. Then the drone goes over the next pattern and the guesses are updated accordingly. After the confidence level of the predicted position reaches to a certain threshold the drone is localized. The operation can continue afterwards on the whole map since the system is aware of the position. Each time the guesses are stored as Markov weights and they are updated in each displacement. Markov algorithm is a statistical solution in robotics for localization [2].

In this thesis we have replaced the method of feature detection with Deep Learning method to recognize patterns. An approximate of 30 photos from each pattern are fed as a dataset to a R-CNN object detection network and trained accordingly. The training checkpoints are based on COCO dataset pre-trained network. Only the six last layers of the network are unfrozen to train and then they are frozen again after the training. Mobilenet SSD v1 is chosen as a training network. The processor is an Edge TPU processor from Google. These devices are embedded and small scale

(small USB stick) which can provide a high frequency (30 fps) object detection with more than 80% of confidence as the results will mention about them later.

1.3 Thesis Contributions

The implemented method has successfully been tested on the patterns. The results have been satisfactory because the data was limited for each input. This hesis has replaced deep learning with the proposed simple image processing in [1] to have the ability to extend the training patterns and data addition for future implementations. This thesis has also brought the proposed method in [1] into action and practically analysed its possibility although it had been only simulated in the original paper.

1.4 Thesis Organization

This thesis is about localization of a quadcopter using specific patterns by help of deep learning. As the subject states, the first chapter will give information about localization definition and how it is being used right now in industry. Then the most important and related kinds of localization technologies will be discussed in the same chapter in addition to the algorithms. Chapter 2 will discuss about deep learning definition. How deep learning networks are used and trained in addition to the method our object detection has followed will be discussed in that chapter. The following chapter after that will be discussing about the implementation method and how the whole setup has been planned. The setup and the method are based on [1] basically with some slight changes. In the paper, the setup is chosen ideally since the authors had not considered how possible the method can be when implemented practically. Here we have done some slight changes such that the outputs are possible but also not to diverge from the main source implementation. The results from the implementations also will be mentioned in the same chapter. Chapter will be about

the discussions, conclusions, and future possible works to extend this research further in the future.

Chapter 2

LOCALIZATION

2.1 Localization Systems Technologies

Indoor areas such as airports, supermarkets, train stations, and hospitals are becoming increasingly interesting as the Internet of Things (IoT) grows in popularity [3], [4]. In supermarkets, customers can grab a cart with tags on a radio-frequency identification (RFID) and personal digital assistant (PDA) screen , and because location of each cart is known via a system of combination of hybrid Wi-Fi and RFID, the PDA screen can be used to search for a specific product and their location, and then to receive directions to that target by the customer [5]. Instead of handing out pamphlets to visitors, a gadget with Wi-Fi and Bluetooth can aid these roaming tourists. The gadget may provide detailed information on the work of art as well as directions to it in a specific area of the gallery. Tourists can also be alerted if there is a traffic jam in a certain location, allowing them to save time and visit alternative locations [6].

Bluetooth beacons can be used in libraries to lead students to book locations. When the location of the book is needed, the student's coordinates will be compared to the network and recommended to him using a downloaded mobile application. Because the localization precision is within meters, the student is guaranteed to be near to the appropriate shelf [7]. In most situations, RFID technology can be used to track the movements of people suffering from mental illnesses such as Alzheimer's [8].

Patients who require home healthcare may have an RFID tag implanted in their body to report if they are sitting, walking, standing, or collapsing, requiring immediate treatment [9]. A dual-shoe combined inertial sensor and range sensor may be installed to detect a firefighter's location and distance to other team members within a building in the event that one of them becomes trapped or incapacitated while on mission. [10], [11]. A fire detection system that uses ZigBee-based sensor networks can improve the localization of the fire source [12]. Figure 1 has demonstrated a method of localization for an indoor drone to transfer a better idea of localization.

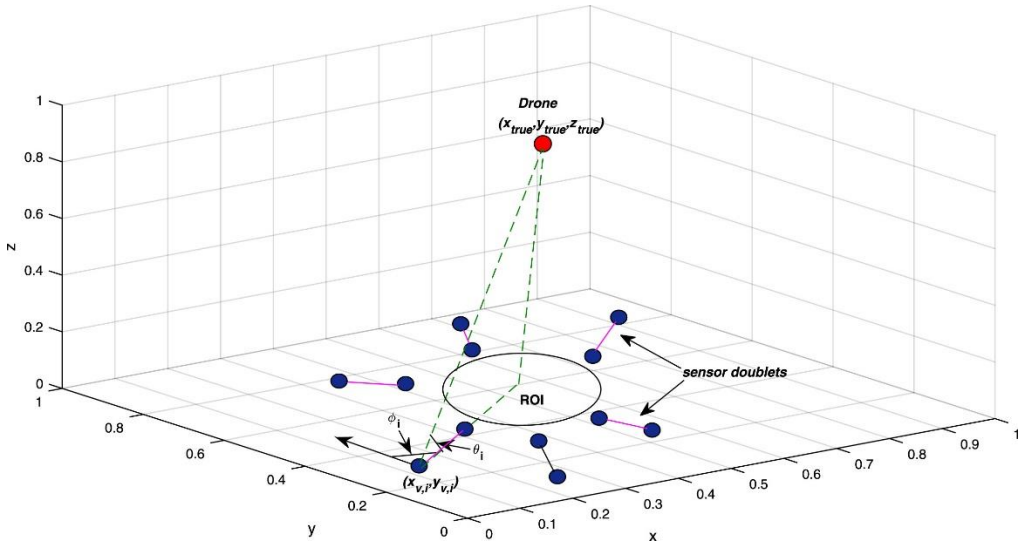


Figure 1. A Drone Localization Using Wireless Sensors [13]

2.1.1 Navigation Based on Satellites

The most recognized system for outdoor positioning is the Global Positioning System (GPS). It should be emphasized that because of line of sight (LOS) needed within the satellites and the device, it will be rendered entirely useless for interior location-based applications due to walls and buildings. [14]. in front of a GPS receiver, a high gain, steerable directional antenna is installed to access the GPS. [14]. Pseudosatellites are useful in situations where the GPS signal cannot be utilized

as a stand-alone method of position. Pseudosatellites, transmitter and receiver antennas, target receivers, and reference receivers are all part of these systems [15]. The aim is to use inside transmitters to replicate the GPS signal that has been received [16].

2.1.2 Navigation System with Inertial Sensors

Inertial Navigation System (INS) can define the directional movement of objects in addition to their location by occupying inertial measuring units (IMU) like gyroscope and accelerometer compared to an initial location, angle, and velocity [17]. In spite of the fact that the sensor has to be connected to surface of the object, INS excels in terms of energy economy and precision [18]. However, because INS may be vulnerable to reading mistakes, a sophisticated filtering mechanism such as the Kalman filter is necessary [19]. Another drawback is the expense and work required to create the location sensor's infrastructure [20]. A unique initial location estimate technique has been implemented in [21] with a combination of concurrent WiFi access points and iBeacons. [22] proposes the iBILL system, which integrates inertial sensors and iBeacons by having two states: localization with iBeacon and particle filter localization (PFL). To help PFL cope with magnetic field variations, ensuring that no mistakes accumulate while walking and lowering PFL's processing cost, iBeacons are used. When iBILL, Magicol (geomagnetism and inertial sensors fusing system), and dead reckoning (DR) were compared, it was concluded that iBILL can reduce mistakes and has higher performance when walking distance is increased.

A hybrid localization approach incorporating acoustic localization and inertial sensor-based dead reckoning, followed by Kalman filter fusion, outperformed the

independent system and compensated for its shortcomings [23]. In addition, a blend of Wi-Fi fingerprinting and inertial sensors outperforms separate methods [24].

2.1.3 Radio Frequency (RF) Based Navigation

The most widely used localization method is RF-based, which is more in a point of interest since it covers a larger region with low-cost hardware [25]. Because RF waves can flow through things such as walls and human bodies, this is easy to understand. Their superior outcomes over other localization methods such as infrared and ultrasonic can also be considered. These systems, on the other hand, should be avoided in planes and hospitals due to the significant risk of interfering with current systems utilizing RF.

Since in the radio spectrum the frequency is lower than 300 GHz, wireless technologies which are a part of localization indoors may be divided based on the different radio frequencies they operate on [26]. Wireless technology's capabilities, such as coverage, wall penetration, and obstacle resistance, are also affected by its frequency. For location-based applications, long, medium and short distance wireless technologies are the three kinds of wireless technologies to be considered [25]. The complexity, accuracy, and environmental factors all influence the kind of distance measurement system that should be utilized for a certain application [27]–[29]. In a Wireless Sensor Network (WSN), node position information is used for routing, clustering, and context-based applications. A network of nodes that detect and wirelessly transmit environmental fields (such as temperature, humidity, and brightness) is defined as WSN [30]. This information is transmitted to the sink node, which collects data. Indoor fire suppression, smart homes, and resuscitation responsibilities are just a few examples [31]. IEEE 802.15.4 has been used to build

wireless personal area networks, (WPAN) or WSNs. WSN localization is the technique of finding an object utilizing wireless sensors' network [32], [33]. Measurements are useless without Knowing the nodes' location making them essential for a WSN. As an example of WSN localization, the usage of RSSI using the ZigBee standard [34]. WSN may utilize range-based (based on internode measurements) and free-range based methods jointly for localization [35], [36]. There are two kinds of lateration and trilateration: exact that is categorized to lateration and trilateration and approximate that is by scene and proximity analysis.

Five sensors are integrated in firefighters' suits, according to [37], that monitor core temperature, blood pressure, heart rate, oxygen saturation, wind speed and heat flux. This information is provided to the team leader on a regular basis to keep tabs on the members' health. In the case of a fire, [38] proposed a similar concept in which victims might be tracked and safe escape routes established. Ultrasonic waves, which are unaffected by smoke, ashes, or fire flames, may be used to assess the status of firefighters inside structures. If the target position is computed by a single computer, the localization system is centralized; however, having target position evaluated by many nodes means spread localization system [39]. WiFi [40], Bluetooth [41] ,Zigbee [42], Ultra-Wideband (UWB) [43], and Radio Frequency Identification (RFID) [26] are examples of RF-based navigation systems. The following sections will shortly mention about them.

2.1.3.1 Frequency Modulation Technology

It has been shown recently that by using frequency modulation (FM) indoor and outdoor localization can be achieved [44]. As a result of its smaller frequency range (88–108 MHz), FM is used less than Wi-Fi (2.45 GHz) and cellular networks (0.9–

1.8GHz). Moreover, broadcasting FM radio is less susceptible to the weather, and terrain, and can pass through obstacles with greater ease as a consequence [45]. As a result of its larger wavelength (3 m), it has a different interaction with interior objects and furnishings than Wi-Fi does Other 2.4 GHz RF components are not affected by FM operation [46]. In addition, less power is consumed when using FM receivers. Fingerprinting for indoor localisation is what RSS relies on for licalisation. According to the information in [47], Gaussian processes (GP) regression, k-nearest neighbour (kNN) and support vector machine (SVM) classifiers were used to assess fingerprinting performance. There is evidence of yielding the results with kNN technique application. Using stations with stronger broadcasts was also recommended to improve accuracy. While Wi-Fi has superior localisation capabilities in large regions such as floors, FM has superior performance in smaller areas such as rooms.

2.1.3.2 Cellular Based Technology

In addition to the three commonly used cellular frequencies, 0.9 GHz, 1.8 GHz, and 2.8 GHz bands are used by cellular networks. This wireless network has a far larger coverage area than Wi-Fi, while without requiring any extra equipment. At first, the idea was to use proximity to ascertain the mobile location. Unfortunately, this approach produced unsatisfactory results [48]. The best method for doing localization with RSS is using the fingerprinting technique. Other scholars are thinking about the alternative solution, in which trilateration is used as a technique for localization. As of the RSS fingerprinting case is assumed, cell-site APs are considered as APs, and the accuracy was inspected to land between 2.5 and 5.4 meters [49].

RSS fingerprinting and radio signals from the Global System for Mobile Communications were used for localization in [50] (GSM). After collecting 29 GSM channels and 6 cells for fingerprint analysis, researchers discovered that fingerprints could be found on both types of media. There was an inaccuracy of less than 5 meters, as if [51] got those findings. To gather fingerprints, a UMTS cell tower was used with interior coverage. In an office environment, measurements were taken in [43]. The tiny cell localization seen in UMTS is comparable to that found in WLAN. When designing an indoor setting, Long-Term Evolution (LTE) was also utilized; according to [52] localization using TOA was performed, and the inaccuracy was less than 8 m in half of the instances. LTE may be utilized for a root mean square error (RMSE) with the value of 3.5 meters approximately with an inertial measuring unit (IMU) [53]. In [54], the synthetic aperture navigation (SAN) architecture has been utilised to minimize multipath signals' effect. An artificial antenna may be used to gather different frequencies at distinct times. Like obtaining a signal from an array, this process is almost identical. SAN will use The ESPRIT (Estimation of Signal Parameters through Rotational Invariance Technique) method will be used by SAN for identifying DOA; according to their study, the RMSE of localization for LTE-SAN was about 4m, while the RMSE for a single LTE was 7 m approximately.

[55] conducted research on localization using only LTE and fingerprinting with LTE-WLAN, finding that poor results come from LTE-only fingerprinting, which at the same time performance is increased by 3.5x with the use of LTE WLAN fingerprinting. To assist them, other current RF localisation systems, such as RFID [56], Wi-Fi [57]–[59], and cellular systems, can be used.

2.1.3.3 Wifi Technology

The term "Wi-Fi" refers to a widely used wireless networking technology. The IEEE 802.11a spectrum utilizes the 5 GHz frequency range, whereas IEEE 802.11b, g, and n use the 2.5 GHz range. Large indoor settings like universities and business buildings have already used WiFi hotspots as network access points that span the whole building. A wide range of devices uses Wi-Fi technology, including video game consoles, computers , mobile phones, cameras, digital music players and tablet computers [60], [61].Costs associated with installing Wi-Fi networks and related equipment may be extremely low, and range has improved from a previous of about 100 meters to approximately 1 kilometre (km). Furthermore, (Received Signal Strength) RSS fingerprinting is used to determine Wi-Fi localisation [62], [63]. Other RF localisation methods, such as RFID [64], might be used with Wi-Fi. While Bluetooth has a more limited range, Wi-Fi offers a wider range of coverage and a higher throughput, making it easier to use [65]. Companies that provide Wi-Fi-based locating solutions include companies such as HERECAST, PlaceLab, RADAR, HORUS and COMPASS [66], [67].

2.1.3.4 ZigBee

ZigBee is an IEEE 802.15.4 standard-based specification which operates in 915 MHz band in the United States and Australia, the 868 MHz band in Europe, and the 2.4 GHz spectrum in the rest of the world. In a wireless mesh network, it is utilized for communication over long-distances between two devices. Compared to WiFi standards, it is inexpensive, has a modest data transfer rate, and has a short latency time. To measure the distance between two or more ZigBee sensor devices in this technology the RSS technique is used [68], [69]. The scanning of access points (APs) via the WiFi interface consumes a lot of electricity. To mitigate this impact, authors

in [70] proposed ZIL, an energy-efficient indoor localization system based on ZigBee that collects Wi-Fi signals via the ZigBee interface. In [71], a proximity learning-based ZigBee localization algorithm was presented; the proposed approach differs from previous standard triangulation-based strategies in that it decreases computing time while retaining accurate placement.

2.1.3.5 Bluetooth

With the standard of IEEE 802.15.1, Bluetooth is designed to allow devices to communicate wirelessly across short distances. Bluetooth, like Wi-Fi, communicates using radio waves with frequencies ranging from 2.402 GHz to 2.480 GHz. It features low transmission power, cost-effectiveness, secure and efficient communications, long battery life, and easily available choices [72], [73]. Bluetooth Low Energy (BLE) is a new Bluetooth version that can span between 70 to 100 m and offer improved power efficient 24 Mbps of bandwidth [98]. As a result, Bluetooth is unsuitable to be considered for large-area localisation [41]. In [73], neural networks (NN) are taught in the training phase using the corresponding coordinates of received signal strength values; once trained, the determination of user position based on live measurements from RSS by using the NN is possible. During the recent years, BLE-based localization is being used as Eddystone (Google) and iBeacons (Apple) in smartphones, where within airports, train stations, large markets, malls, and restaurants, the smartphone can be used for localization by sending the area map to the smartphone and then BLE-based localization is done [74].

2.1.3.6 Ultra-Wide Band

A carrier frequency of over 2.5 GHz and bandwidth of upper than 500MHz is a UWB signal characteristic, based on the Federal Communications Commission (FCC) of the United States [75]. In UWB the power consumption that is low results

in a high-speed communication, broad bandwidth, high temporal resolution, short-wavelength, and high data rate, that makes UWB more resistant to fading and multipath interference. The other advantageous feature of UWB is to be able to operate at low carrier frequencies, in which signals may readily flow through barriers more; moreover by being resistant to interference due to its considerably different spectrum UWB stands out [76]. All these qualities make UWB an excellent choice for indoor wireless positioning. Greater accuracy of TOA and time difference of arrival (TDOA) than other localization methods are expected from UWB signals due to their high temporal resolution, with a reduced multipath effect. UWB is capable of minimizing error to millimetres [43], [77]. The authors of [78] suggested a hybrid localisation method utilizing UWB and Wi-Fi, which can be done by having UWB beacons added to an existing Wi-Fi network. By deploying their algorithms, a combination of UWB and Wi-Fi infrastructure can reduce the precision and cost of UWB; the localization error was restricted to 20 cm. A limited number of tags can be localized by typical UWB systems; according to [79] an unlimited number of tags can be localized using SnapLoc, which is another type of UWB system. [80] investigated the performance of UWB localization systems in LOS and NLOS conditions. The location was calculated using weighted centroid estimation (WCE), linearized least square estimation (LLSE) and fingerprint estimation (FPE),the position was calculated; The research proved that while FPE performs the best, the worst performance is from LLSE.

2.1.3.7 Radio Frequency Identification (RFID)

RFID systems rely on backscattering of RFID tags communication, and the same time they need the created signals within tags and readers to be processed with RFID readers [81]. Tags with RFID are classified as passive, active or semi-active. Tags is

active status have a built-in battery incorporated as a part of electronics. With a detection range of up to 100 m, active RFIDs operate at ultra-high (UHF) and super-high frequencies (SHF) spectrum. As a result, object tracking and long-distance localisation can be achieved by an active RFID [82]–[84]. Active RFID technology, on the other hand, is unreliable for expecting precisions less than a meter and lacks availability on the most of portable devices. Lack of built-in batteries and instead backscatter the signal received from the base station are also disadvantages of passive tags. Because of its multiple advantages passive RFID is widely used for a variety of applications and can bring and ease of manufacturing compared to active RFID, which requires only a tag chip and an antenna, low cost, and reduced size. Sub-meter detections by utilizing passive RFID can be helpful since they can identify targets within a range maximum 10 meters [84]. Because of their low cost, Radio Frequency Identification (RFID) technology have gone widespread. In such technologies, a huge number of reference tags are readily deployed. The Radio Signal Strength Indicator (RSSI) information from the readers around each tag that transmitted the signal will be measured. The reference tags with the RSSI information closest to the RSSI information of the target tag can approximate the position [85], [86]. In [87], an RFID reader, infrared sensor pair and tags, a light-emitting diode LED and a light resistor were utilized as sensor pairs for localisation. In terms of precision and stability, RFID-based localization outperforms traditional sensors. The characteristics of existing localization methods are shown in Table 1 [74], [88]–[93].

Table 1. Comparison Between Different Localization Methods [97]

Technology	Technique	Method	Accuracy(m)	Cost	Coverage	Pros	Cons
Satellite	Trilateration	TOA TDOA	3-5		Floor level	Cheap	
Inertial	Dead Reckoning	-	2	Low	Floor level	Cheap	Accumulative errors
Magnetics Based	Trilateration Fingerprinting	-	2	Low	Floor level	Good Accuracy	Requires mapping
Ultrasonic Based	Trilateration	TOA TDOA	0.01-1	Medium-High	Room level	No effect of multipath	Interference cost for hardware
Acoustic	Trilateration	TOA	Meters	Low	Room level	Cheap	Poor accuracy
Infrared	Proximity Trilateration	TOA	1-2	Medium	Room level (few meters)	Cheap No effect of multipath Low power consumption	Sunlight interference Short-range Cost for hardware
Visible light	Angulation	AOA	0.1	Medium	Floor level	No interfering	Expensive Construction
Wi-Fi	Proximity Trilateration Angulation Fingerprinting RSS-Propagation model	AP ID RSS TOA TDOA AOA	10 (Proximity) 1-5	Low	Floor level (around 35)	Good accuracy Low cost Wi-Fi signals can penetrate walls/ No need for additional infrastructure	RF interference with devices operating at 2.4 GHz Fingerprinting requires a huge effort
ZigBee	Proximity Trilateration Fingerprinting	AP ID RSS	3-5	Medium	Floor level	Low Cost Low power consumption	Requires Special equipment

	RSS-Propagation model						
Bluetooth	Proximity Trilateration Fingerprinting	AP ID RSS TOA	2-5	Low-Medium	Around 10 meters	Good accuracy No need for additional infrastructure Low power consumption	RF inference Limited coverage and mobility
UWB	Trilateration Angulation	TOA TDOA RSS AOA	0.01-1	High	Few meters	Accurate	Expensive
RFID	Proximity Trilateration Fingerprinting RSS-Propagation model	AP ID RSS	1-5	Low	Room level	Cheap Real-time localization	Low accuracy Response time is high
FM	Fingerprinting	RSS	2-4	Low	100 km	Low sensitivity to objects	Vast
Cellular network	Fingerprinting Proximity Trilateration	RSS TOA	2.5-25	Low	80 km	Networks available all over areas	Low Accuracy

2.2 Localization Detection Techniques

2.3 Technique of Using Proximity

(Also known as relative positioning/connectivity) is a low-cost also simple method of estimating the distance between a mobile and an AP location. As long as being within communication range, it makes no difference whether the AP and the mobile exist on the fading channel same as each other or not [94]. The AP's coordinates are used to approximate the mobile's location. Its accuracy is restricted to AP radio coverage while the proximity method is frequently used and simple [95]. In general, there are three types of approaches for proximity technique. The first is sensing physical contact, which uses sensors such as touch sensors, pressure sensors, and capacitive field detectors for detecting physical contact. Another method involves monitoring the wireless signal of a mobile device inside the access point's range. Lastly, credit card payment terminals that are a type of automatic identification systems can be examined by it [96].

2.3 Scene Analysis

By analysing the scene in this method, virtual images, videos, or electromagnetic properties received from the target are compared with the dataset available, allowing the feature to be mapped to a position on the target [98], [99]. Wearable cameras, for example, can associate collected virtual pictures with the target's position. Commonly referred to as fingerprinting is when wireless signal characteristics of specified places may be acquired for generating a radio map and by mapping the mobile's signal data to it can deduce the position of a mobile device. Localization of this form is well-known for its simplicity; nevertheless, it necessitates the collection of a significant quantity of data; moreover [100], altering the environment may result in changes to the feature characteristics, necessitating to update the dataset [96].

2.3.1 Triangulation

The target position may be established using triangulation by constructing triangles from known locations to that same destination. Lateration and angulation are the two types of lateration. Lateration is a distance-based method used in the Time of Arrival (TOA) and Received Signal Strength (RSS) approaches, when a direction-based technique called angulation is used in the Angle of Arrival (AOA) implementation.

2.3.2 Lateration

What determines the distance from mobile to AP depends on the ratio of power to travelling time. The connection may be expressed with a mathematical equation. will be available 2D measurements with two equations can have two potential solutions. For there to be a unique solution, there are three equations necessary; the combination of these equations will decide how the mobile phone is located as illustrated in Figure 2. Lateration is also an option to estimate position using differential measures (signal intensity receipt/time of arrival). The impacts of environmental changes are reduced through differential measurements. The transmitted power is in this instance unknown (DRSS) or if the (TDOA) is not known [101]–[103].

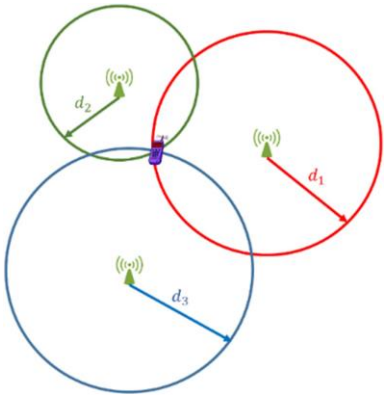


Figure 2. Trilateration Localization [97]

In localization in case of m Aps collaborating, there is going to be $(\frac{m(m-1)}{2})$ differential equations formulated, with $(m-1)$ fundamental equations and the redundant ones as the rest. Each fundamental equation's solution will be on a hyperbola, and the intersection of these hyperbolas provides the mobile's coordinates [76]. There will be two fundamental equations in the 3 APs system, a linear combination of the first two yields the third equation. Different types of localization detection approaches and their accuracies, cost and measurement type compared have been mentioned on Table 2 [26], [91], [92], [109].

Table 2. Comparison Between Localization Techniques

Technique	Accuracy	Measurement type	Cost	Notes
Proximity	Medium	RSS	Low	Adding more AP will increase accuracy and cost Accuracy is specified by AP cell size Not affected by the multipath effect No need for extra hardware
Fingerprinting	High	RSS	Low	Requires massive work to construct a radio map Adding/removing AP will require to update the radio map Not affected by the multipath effect No need for extra hardware
Time	High	TOA	High	Suffer from multipath AP location should be identified
Direction	Medium	AOA	High	Suffer from multipath Requires array antennas with specific angular properties AP location should be identified
Dead Reckoning	Low-Medium	Velocity Acceleration	Low	Suffers from accumulative errors

2.3.3 Angulation

As shown in Figure 3, to localize an item in two dimensions, measurements of two angle and a measurement of single range are needed. The distance between the two arrays may be the range measurement. For 3D measurements, a single azimuth measurement, a single range measurement and two angle measurements are needed [98].

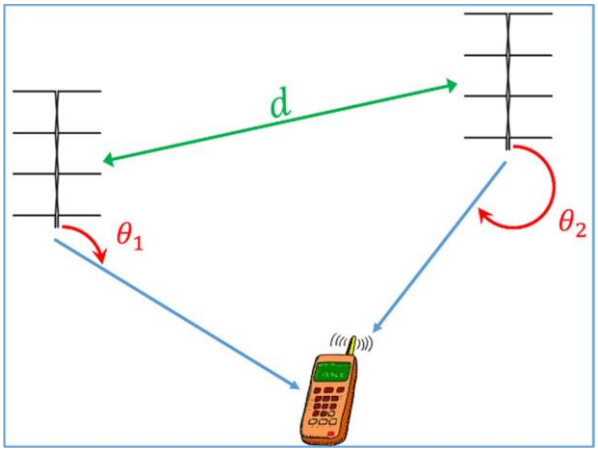


Figure 3. Angulation with Angles And Distances Known from Two Sources [97]

The inertial measurement unit sensors used in the DR method can track target movement using gyroscopes, accelerometers and magnetometers [104]. The position of a target is recalculated by adding the last displacement estimated before [105], being aware of the velocity of the target at a known location. This method is simple and that is why it stands out. However, in order to avoid mistakes, a precise starting position is required, even when errors for correction grow over time since no external reference signals are utilized [106]. To achieve more precise findings, hybrid methods are employed [105]–[107]. In [108], pedestrian DR (PDR) and Wi-Fi fingerprinting were used to conduct localization; led to a conclusion that PDR has the poorest performance, but the greater performance comes from the combination of PDR and Wi-Fi.

2.4 Algorithms And Methods for Localization

2.4.1 Angle of Arrival Measurement

Beamforming and localization are using the direction of arrival (DOA) [109]. To determine angle of arrival arrays of antenna are employed. DOA needs the use of antenna arrays, making it more costly and a power consumption higher than TOA and RSS [110]; nevertheless, less equipment is required because two Aps only are required to calculate the position of the mobile [111].

2.4.2 Time of Arrival Measurements

Utilizing the velocity of wave to estimate distance among two sensors, TOA measurements calculate the time of flight between the AP and mobile [112]. RF and acoustic signals are examples of waves used for localization [112]. Radio waves have a velocity of 3×10^{8} m/s, which at the same time acoustic waves have a speed of 343.59 m/s [113] and accordingly, measurements with RF are more susceptible to mistakes. When utilizing RF waves, a measurement mistake of 1s will result in a 300 m inaccuracy, whereas using acoustic waves will result in a 0.00034359 m error [114]. The receiver resolution will be around 1 10–9 when the receiver bandwidth is 1 GHz. As a result, the highest error will be 0.3 m, but the resolution of the receiver is going to be approximately 1×10^{-7} and the maximum error will approximately be 30 m when utilizing 10 MHz bandwidth [65]. The idea of lateration is used in TOA localization [115]. It must contain three equations to answer these equations (i.e., three APs measurements must be used). To have a unique solution in a 3-Dimensional case (x, y, z), minimum of four APs must be employed. The coordinates of the mobile are deduced by transforming TOA data into circular equations and solving the same equations [116].

The time difference of arrival (TDOA) is a related measurement of time in which difference of time between two TOA measurements is utilized for creating a single equation. Two TDOA values will be obtained from three TOA measurements, but the third equation will be reliant on the other equations and therefore will not offer further information. Four AP measures are utilized to provide a unique solution [116]. The mobile's possible locations will be plotted on a hyperbola [116]. As illustrated in Figure 4, the intersection of two hyperbolae yields the position [116]. To have a unique solution, three fundamental equations are necessary, that is accomplished with another AP included, as illustrated in Figure 4. As a result, 4 APs are required for 2D localization.

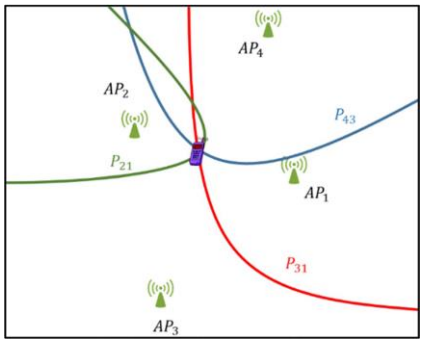


Figure 4. Hyperbolic Localization [97]

In TOA, every sensor, the mobile included, must be synchronized since the mobile phone's time is not the same as the clock in the base station accuracy-wise. As a consequence, there can be errors in estimating flight duration and, as a result, errors in localization; nevertheless, in TDOA, only APs must be synchronized [118]. While TOA uses existing data better, it does not provide the same amount of flexibility for the mobile, which is confined to a circle when using one measurement, and able to be in two places when using two measures; two measurements will allow the mobile to be located on a hyperbola; and three measurements will allow the

mobile to be in three locations. TOA can predict a single solution, but TDOA can predict one or maybe two [119]. Another disadvantage of employing TDOA is the occurrence of sensitivity LOS [120]; because to the hyperbolic nature of the curve, a tiny bit of inaccuracy will cause a huge shift within the curve, making less accurate results [121]. Assume the LOS route has been attenuated and has fallen below the noise rejection threshold. In such scenario, the next path with power over the noise level is treated as the initial arrival path, resulting in inaccurate TOA estimate and, as a result, incorrect localization [110]. During its spread, the wave ran against walls.

2.4.2.1 Techniques Using Correlation

The most thorough approach for estimating TOA is cross-correlation is one of the most [110]. Figure 5 shows TOA estimate; after the signal came, a match filter MF correlated it to a template known $p(t)$. The associated signal's sign is eliminated using a square law device, and a time instant having a maximum value of peak reflects the moment when first the signal is received [122]. Received peaks are going to have similar amplitudes to the proper one in multipath propagation nearby; hence, picking the correct peak becomes confusing, resulting in significant mistakes [122]. Because of its simplicity, this technique is popular; nonetheless, it is susceptible to multipath and noise.

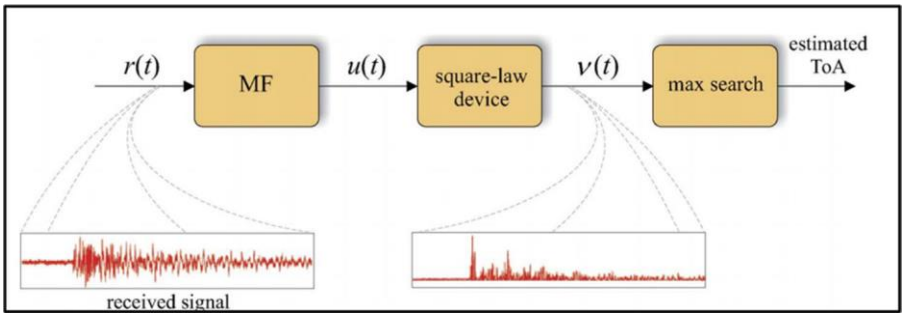


Figure 5. Cross-Correlation Using TOA Estimation [122]

Chapter 3

DEEP LEARNING FOR LOCALIZATION

3.1 Introduction

One of the most frequently used methods for computational intelligence is Artificial neural networks (ANNs) today which began as a software and tailored hardware attempt to imitate adaptive organic nerve systems [123]. For more than 70 years ANNs have been a point of interest in research [124], their popularity has risen and fallen during that period. Following pioneering work by several scholars [125], they have recently made a significant comeback as pattern recognition techniques. Given adequate computer resources and training data, multilayered artificial neural networks have been shown indisputably to be capable of learning complicated, non-linear functional mappings. The intellectual neighborhood has experienced exponential development, both in industry and academia, as a result of these amazing, substantial breakthroughs in strong pattern recognition. Furthermore, multilayer ANNs eliminate much of the human labor previously required for setting up traditional pattern recognizers. In fact, they are unknown systems offering outstanding practical performance that demand unstructured, high dimensional data insights with minimum human intervention [126]–[131].

3.1.1 What Is an Artificial Neural Network?

To tackle classification or regression issues, an artificial neural network consists of vast amount of linked, simple functional neurons, which work together as data

processors in parallel. That example, they can divide the range of all potential input values to discrete classes or estimate the function that can do mapping between inputs and outputs (the black box). When layers of these massively connected neurons are stacked to form a network, the resulting computing infrastructure can:

1. Receive information of one input neurons (a part of the neural network's input layers) and respond to the environment.
2. Invoke design goals and learning rules to transfer information between layers inside the black-box enabling processing (known as a part of the hidden layers of the network).
3. There are atomic units in the neural network's output layers that are known to transmit processed information to the surrounding environment.

Each neuron's output in a hidden layer is linked to a subset (or all) of neurons from the previous layer. The neuron computes the sum of the products of earlier mentioned outputs in addition to weights associated with them. A projection of one vector onto the other or a measure of similarity between the two may be considered of as the dot product of an input vector and a weight vector. Suppose the input vectors and weights are both n-dimensional, and the layer has m neurons. The layer's output is an m-dimensional vector generated by multiplying the training set by a $m \times n$ weights matrix, since weight vector is in each neuron and, given an n-dimensional input vector, the output is an m-dimensional vector. The output of each neuron is essentially a classifier of linear kind, with the input vector lying on one side or the other and the weight vector forming a border between two classes. An m-dimensional hyperplane splits the n levels of the input to two m-dimensional classes in the output classes when all m neurons' outputs are added together. If the weights

are generated using least mean squared (LMS) estimation from matched pairs of input–output data, they create a regression line, i.e. the hyperplane which is the nearest to all the outputs given the inputs in the LMS sense.

The hyperplane translates incoming input values to output which are compatible with the original input data by reducing the error function between calculated outputs and real outputs in the training data. The result of one linear classifier's can be used for the input of another, thus many layers of linear maps are the same as a unique classifier or regression. This is because multiplying the inputs by a $q \times n$ matrix, which is the result of the k matrices multiplied, lowers the output of k distinct levels to a single $q \times n$ matrix one per layer. To classify inputs nonlinearly or simulate a nonlinear function using a regression, each neuron provides a numerical bias number to the output of its input sum of products (the linear classifier) and passes it via a nonlinear activation function. The exact form of the activation function is a design parameter. They all, however, translate the real line via a monotonic rising function with a zero-inflection point. The bias effectively changes the activation function's inflection point to the bias's value in a single neuron. Therefore, the total of products is mapped using a bias centered activation function. Any pair of such specified activation functions may generate a pulse across their turning points if each is scaled and one is subtracted from the other. Each pair of neurons effectively samples the input space and produces a single value for all inputs within the pulse's limits. From training data consisting of input–output pairs - input vectors each with a corresponding output vector - the ANN learns an approximation to the function that generated each of the outputs from its related input. The split of the input space into samples that minimizes the error function between the ANN's output and its training

inputs and outputs given its training inputs is known as this approximation. The universal approximation theorem says that if an ANN has enough neurons in a sufficient number of layers with a given activation function, it can approximate any functional mapping between input and output vectors with arbitrary accuracy [132]–[135]. Figure 6 represents a perception learning model which encapsulates these ideas. By optimizing on the pairs of input and output the weights are generated and the error function is minimized, giving the size of the vectors of input and output, the layers' numbers, an error function, the shape of the activation function and the number of neurons in each layer. Consequently, the resultant network is a close match to the known input–output data.

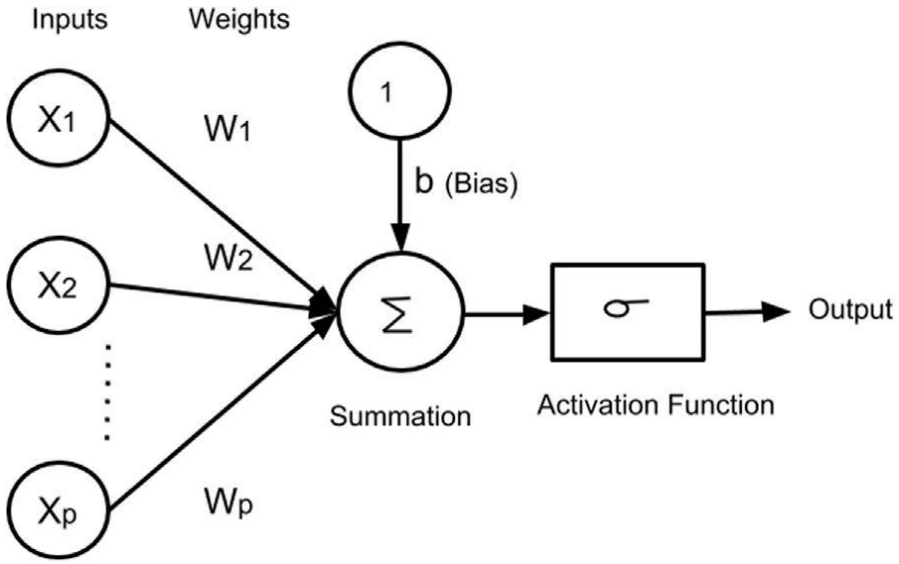


Figure 6. The Perceptron Learning Model [136]

3.1.2 How Do These Networks Learn?

Learning is feasible with neural networks, as it is probable to estimate a function representational the input patterns by altering the weight distribution. The fundamental concept is a black-box re-stimulation the with additional data till it achieves a suitable representation that is well-structured. Weights are given a little

amount the proper direction each time of stimulation, assuming that the algorithm for learning is suitable in application. This ends when the approximation error vs. well-defined measure falls less than a minimum set by the practitioner. The accumulation of neural computations variable length in causal chains [137] attempting for mimicking a specific task of pattern recognition by linearity regulation with activated neurons throughout the architecture is therefore learning. Non-linearity assists the modulation process when a failure of implicit chains of linear activation happens to understand structure related to them. In this application, the term "deep" refers to the spatial complexity of the aggregation chain, which must span several hidden layers in order to acquire suitably comprehensive representations. Although the current limits of the discipline are well known, theorists joint with empiricists have had a contribution to an exponential increase in research employing DNNs [138]–[140]. In contrast to the problem-specific and hard coded, architectures of pattern recognition in the past, deep learning has raised for being the most important components of contemporary artificial intelligence research due to its capability for scaling using data form the input and generalizing through similar underlying feature distributions problems.

3.1.3 Deep Neural Networks Are Getting Attention So Much, Why Is That?

Neural networks with a multi-layer architecture were present since the second half of the twentieth century. Why have deep neural networks attracted a huge amount of focus from academia and industry recently is a reasonable inquiry? Many factors have contributed to the rapid increase in research funding and output. New software platforms such as PyTorch [141], Tensorflow [142], Caffe [143], Chainer [144], Keras [145], BigDL [146], etc. allow architecture integrations seamlessly into a computing environment with GPU. Improved regularization techniques have been

introduced over time to help avoid overfitting as we scale up: techniques such as data augmentation, batch normalization, early stopping, dropout and others are significantly effective to eliminate overfitting and can make an improvement on model performance that scale on their own. Algorithms that are equipped with adaptive learning rates (Ada- Grad, RMSProp, Adaboost, Adam), Particle Swarm Optimization, Stochastic Gradient Descent (with standard or Nesterov momentum), Differential Evolution, and other algorithms produce solutions nearly optimal while being the one with optimization robustness.

3.2 A Short Description of Deep Learning Architectures

A plethora of deep learning architectures exists in the literature, and the number is expanding by the day. It's difficult to provide comparison that is fair for those architectures because different architectures offer different benefits depending on the application and the data characteristics. Convolutional Neural Networks [147] and Recurrent Neural Networks [148] are preferred in computer vision and sequence and time series modeling, respectively. Deep learning is a rapidly expanding discipline, with new architectures and learning algorithms being developed on a regular basis to meet the demand for human-like efficient machines in a variety of applications.

3.2.1 Convolutional Neural Networks

Convolutional Neural Networks (CNNs) are inspired from human’s vision system. Considering the fact that LeCun et al. [147] suggested the notion in 1998, the deep learning community first experienced it in action in 2012, the same time Krizhevsky et al. [149] with AlexNet architecture proposed won the ILSVRC-2012 competition [149]. Artificial intelligence experienced a new era when this astounding victory ushered, with witnessing CNN's and its descendants' extraordinary classification capabilities by the community of computation. Many derivative architectures have

been presented and are still being explored in the years since. The CNN architectures easily could go over human recognition capabilities in many circumstances. Figure 7 depicts the basic architecture of CNN, which includes numerous convolutions pooling layers and convolutions, as well as a consistently linked layer at the conclusion. Pooling layers minimize the feature map dimensionality while keeping the information of features, whereas extractions of essential characteristics convolution layers from the input image while considering how the input pixels are spatially related [99]. When connected fully, each layer can join the network to the output layer (discriminative layer), that produces the outputs required.

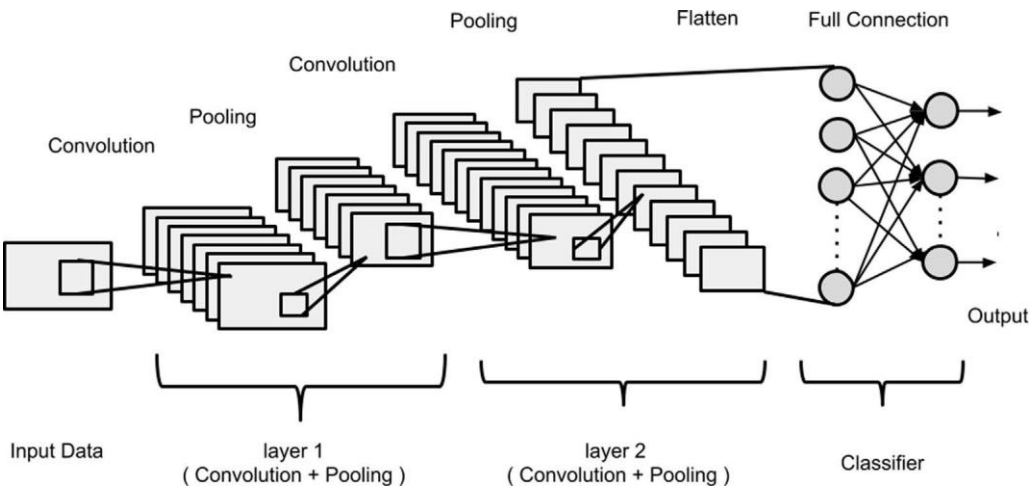


Figure 7. CNN With Pooling Layers And Convolution [136]

What CNNs are especially good at is the ability to extract picture descriptors from spatial data that is latent. Gradients, edges, strokes, contours, textures, color, and orientation are all properties of a picture. A CNN decomposes an image into these types of simple features, which it then learns as representations in different layers [150]. The learning system depicted in Figure 8 is a good illustration of it.

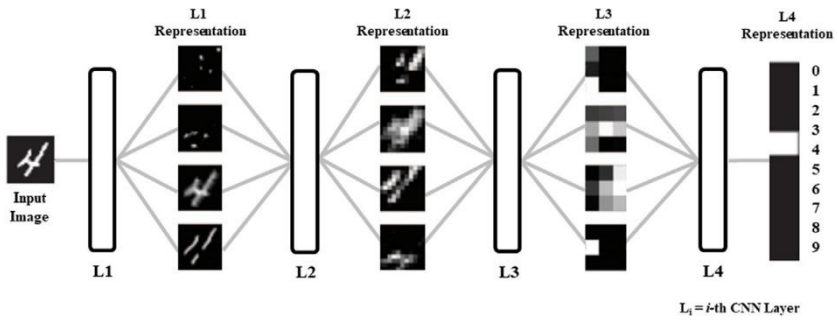


Figure 8. Handwritten Image Digit Learned By CNN [151]

Image detection [152], [153], image segmentation [154], [155], picture classification [156], and image super-resolution reconstruction[157], [158] are only a few of the computer vision tasks that CNNs are used for. Several CNN architectures have been designed to address the needs of real-time applications while also fulfilling high accuracy standards. Recent architectures include YOLO (You Only Look Once) [159] and R-CNN (Region-based CNN) [154]. Because it evaluates a large number of area recommendations to locate an object within an image, the basic approaches of CNN [160] are computationally quite expensive. R-CNN, on the other hand, is a CNN with region-based type which can overcome the limitations of naïve CNN by using a selective search to choose the regions of interest (ROI) and limiting the proposed regions to 2000 [154]. The authors then proposed Fast R-CNN [161] for applying R-CNN to processing in real-time. In contrast to R-CNN, F-RCNN is a faster approach where the convolution operation is performed on each of the 2000 interest regions independently on a single image, the convolution process is only once performed for the entire image. Feature maps extracted are then subjected to a selection search to discover region recommendations. However, the time-consuming part is still selection search method that brings down the speed of object detection process when using Fast R-CNN. Faster R-CNN [162] reduces the time complexity by substituting the selective search strategy with a unique Region Proposal Network

(RPN). The variants of R-CNN outlined utilize a two-stage approach and search within the image's different regions to locate the item inside it [154], [161], [162]. That said, they restrict the capabilities of the network in order to achieve the goal of real-time object detection. Redmon et al. [159] proposed YOLO (You Only Look Once) in 2016, that in comparison to R-CNN records very fast with little performance change. It knows the generalized representation of the image with a single convolutional neural network by looking once at the object. The algorithm, on the other hand, has a spatial constraint when it comes to recognizing smaller things. This issue has been mentioned in Single Shot MultiBox Detector (SSD) [163], that uses multiple anchor box scales [164] as an option to the fixed grid used in YOLO. This might successfully handle objects of various sizes and resolutions, with real-time inference capabilities like YOLO. As a result, various to enhance accuracy while keeping the pipeline intact, YOLO tweaks have been proposed and are speedier overall. Compared to prior versions, YOLOv2 [165] and YOLOv3 [166] have provided considerable gains in accuracy and have also been modified detecting small objects. Aside from these CNN designs, there are various versions of existing traditional ones, such as GoogleNet [167], LeNet [147], VGGNet [168], ResNet [169], ZFNet [170], AlexNet [149] and more. The CNN architectures have had an extraordinary impact on AI-guided vision research, and they appear to be powering it for the promising future.

3.2.2 Recurrent Neural Networks (RNNs)

By the chance of expressing dependencies on time, Hidden Markov Models (HMM) may become impractical computationally especially for ling-term dependencies, that is what RNNs are there for. Using differential equations, [171] provides a comprehensive derivation of the Recurrent Neural Network. RNNs are nets with

feed-forward type that span neighboring time steps, with each node taking hidden node values as well as the current data input collecting information from previous time steps at any one time. A Recurrent Neural Network architecture is shown in Figure 9.

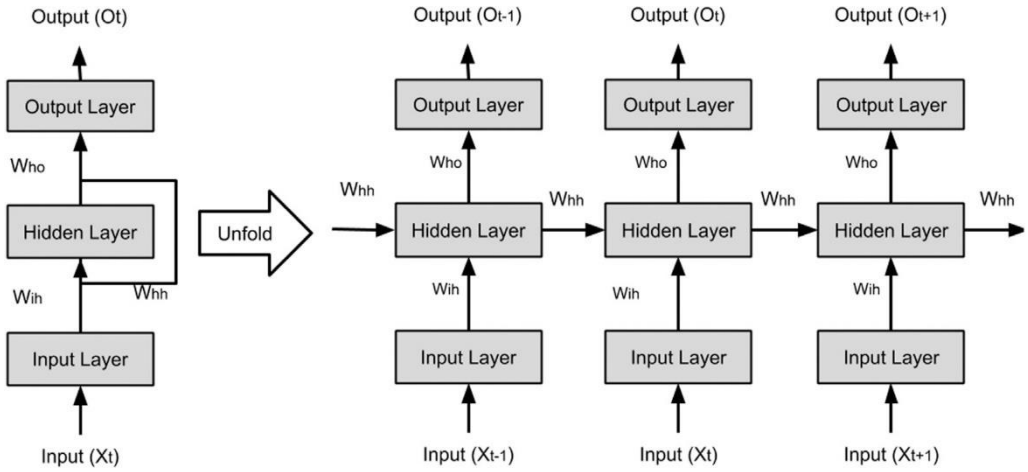


Figure 9. Architecture Of A Recurrent Neural Network [136]

The problem of gradients that are exploding and vanishing occurs during the backpropagation of mistakes across several timesteps, which can be avoided using Hochreiter and Schmidhuber's Long Short Term Memory (LSTM) Networks[172]. "Forget" gate regulates the quantity of information to be kept from earlier time steps, while "input gate" chooses on the new content to be saved in the cell. Finally, the output is controlled by the output gate and the hyperbolic tangent activated candidate value of the state. Figure 10 in LSTM, number 10 depicts a repeating module. LSTM networks with peephole connections [173] use cell state information to update the three gates. In the Gated Recurrent Unit (GRU) [174], a single update gate replaces the forget and input gates, integrating the hidden and cell states. Sak et al. proposed training LSTM RNNs on multicore CPUs in a distributed manner utilizing asynchronous SGD (Stochastic Gradient Descent) optimization for acoustic

modeling in [175]. They demonstrated a two-layer deep LSTM architecture with a linear recurrent projection layer on each layer, allowing for more effective usage of model parameters. Doetch et al. [176] suggested an LSTM-based training framework for handwriting recognition that consists of sequence chunks that comprise mini batches. To minimize runtime by a factor of three, the design employs modified gating units with layer-specific weights for each gate. Palangi et al. [177] used LSTM-RNN to develop a sentence embedding model that systematically collects information out of each phrase and embeds it in a semantic vector until the conclusion of the sentence to produce an overall semantic representation of the entire phrase. In web document retrieval applications, the model's capacity to attenuate insignificant phrases while recognizing important keywords is particularly effective. Pota et al. [178] developed a Bi-LSTM architecture to correlate words' sequence to a sequence of POS tags, which has applications in Natural Language Processing. Gao et al. Using a layered architecture cognition module with long short-term memory and multi-layer perceptron, [179] created a middle point model to segregate targets and visual tracking localization applications.

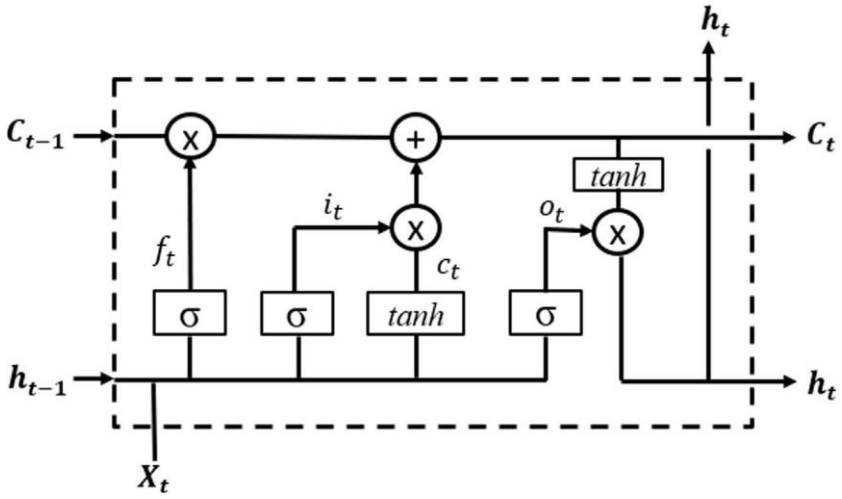
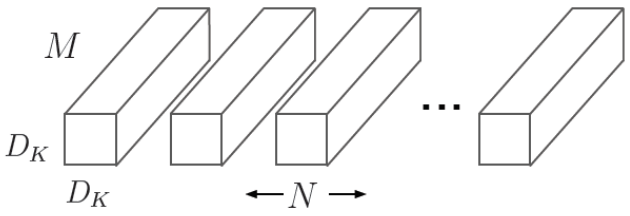


Figure 10. A Repeating Module In LSTM [177]

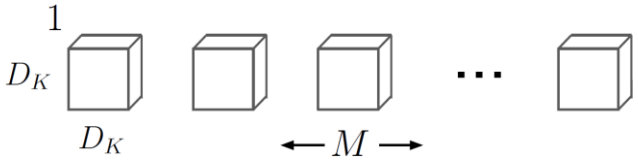
3.2.3 MobileNet Architecture

3.2.3.1 Depth Wise Separable Convolution

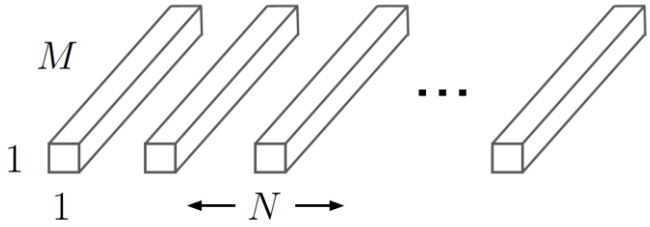
The model of MobileNet is built on separable depthwise convolutions, which are a type of convolution factorized that divides a standard convolution into a pointwise convolution and depthwise. In the depthwise MobileNets single filter is applied to input channels separately. Outputs of the depthwise convolution are then combined using an 1×1 convolution by the pointwise convolution. In one step, a combination of inputs to create new outputs is done by convolution filters. This is separated into two layers by the depthwise separable convolution, one for filtering and the other for combining. This factorization reduces computing time and model size significantly. Figure 11 depicts the factorization of a standard convolution 11(a) into a depthwise convolution 11(b) and a 1×1 pointwise convolution 11(c).



(a) Standard Convolution Filters



(b) Depth wise Convolutional Filters



(c) 1×1 Convolutional Filters called Pointwise Convolution in the context of Depth wise Separable Convolution

Figure 11. The Standard Convolutional Filters

A standard convolutional layer gets in a $D_f \times D_f \times M$ feature map **F** as input while outputting a $D_f \times D_f \times N$ feature map **G**, where D_f is the spatial width and height of a square input feature map, M is the number of input channels (input depth), D_G is the spatial width and height of a square output feature map, and N is the number of output channels (output depth). The conventional convolutional layer is parameterized by a convolution kernel **K** of size $D_k \times D_k \times M \times N$, where D_k is the spatial dimension of the kernel, M is the number of input channels, and N is the number of output channels, as previously specified.

For standard convolution with stride one and padding, the resulting feature map is as follows:

$$\mathbf{G}_{k,l,n} = \sum_{i,j,m} \mathbf{K}_{i,j,m,n} \cdot \mathbf{F}_{k+i-1,l+j-1,m} \quad (1)$$

The computational cost of standard convolutions is:

$$D_K . D_K . M . N . D_F . D_F \quad (2)$$

It can be inferred that computational cost is affected by the number of output channels N, the number of input channels M, the size of the feature map $D_f \times D_f$ and the kernel size $D_k \times D_k$. Each of these terms, as well as their relationships, are addressed in MobileNet models. To begin, depth wise separable convolutions are

used to eliminate the relationship between the kernel size and the number of output channels. The normal operation of convolution has the effect of filtering and merging data based on convolutional kernels to create a new representation. For a significant reduction in computing cost, the filtering and combination phases can be divided into two parts using depthwise separable convolution which is considered as a factorized convolution. There are two layers to depth wise separable convolutions: depthwise convolutions and pointwise convolutions. To apply a single filter to each input channel, we employ depthwise convolutions (input depth). The output of the depthwise layer is then linearly combined using pointwise convolution, a simple 1×1 convolution. Both layers of MobileNets use batchnorm and ReLU nonlinearities.

Depthwise convolution with one filter per input channel (input depth) can be written as:

$$\hat{\mathbf{G}}_{k,l,m} = \sum_{i,j} \hat{\mathbf{K}}_{i,j,m} \cdot \mathbf{F}_{k+i-1,l+j-1,m} \tag{3}$$

$\widehat{\boldsymbol{K}}$ is the $D_k \times D_k \times M$ depthwise convolutional kernel, and the m_{th} filter in $\widehat{\boldsymbol{K}}$ is applied to the m_{th} channel in **F** to form the m_{th} channel of the filtered output feature map $\widehat{\boldsymbol{G}}$.

The cost of computing depthwise convolution is:

$$D_K . D_K . M . D_F . D_F \tag{4}$$

In comparison to ordinary convolution, depthwise convolution is incredibly efficient. This will not, however, combine input channels for producing new features; it just filters them. An additional layer is required that calculates a combination with linearity containing depthwise convolution output via 1×1 convolution for

generating these new features. Depthwise separable convolution, proposed first in [180], is a resultant of 1×1 (pointwise) convolution and depthwise convolution.

Cost of depthwise separable convolutions:

$$D_K . D_K . M . D_F . D_F + M . N . D_F . D_F \tag{5}$$

This is equal to the sum of depthwise and 1×1 pointwise convolutions.

We gain a decrease in computation of: by defining convolution as a two-step filtering and combining procedure:

$$\begin{aligned} & \frac{D_K \cdot D_K \cdot M \cdot D_F \cdot D_F + M \cdot N \cdot D_F \cdot D_F}{D_K \cdot D_K \cdot M \cdot N \cdot D_F \cdot D_F} \\ = \quad & \frac{1}{N} + \frac{1}{D_K^2} \end{aligned} \tag{6}$$

Extra factorization in the spatial dimension, as in [181], [182], does not save much additional work because depthwise convolutions need very little processing.

3.2.4 Mobilenet Structure and Training

Except for the first layer, which is a full convolution, the MobileNet structure is based on depthwise separable convolutions, as discussed in the previous section. We may quickly explore network topologies to identify a nice network by defining the network in such simple terms. Table 3 depicts the MobileNet architecture. With the exception of the final fully connected layer, which has no nonlinearity and feeds into a SoftMax layer for classification, all layers are followed by a batch norm [183] and ReLU nonlinearity.

Table 3. MobileNet Body Architecture [136]

Type / Stride	Filter Shape	Input Size
Conv / s2	$3 \times 3 \times 3 \times 32$	$224 \times 224 \times 3$
Conv dw / s1	$3 \times 3 \times 32$ dw	$112 \times 112 \times 32$
Conv / s1	$1 \times 1 \times 32 \times 64$	$112 \times 112 \times 32$
Conv dw / s2	$3 \times 3 \times 64$ dw	$112 \times 112 \times 64$
Conv / s1	$1 \times 1 \times 64 \times 128$	$56 \times 56 \times 64$
Conv dw / s1	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 128$	$56 \times 56 \times 128$
Conv dw / s2	$3 \times 3 \times 128$ dw	$56 \times 56 \times 128$
Conv / s1	$1 \times 1 \times 128 \times 256$	$28 \times 28 \times 128$
Conv dw / s1	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 256$	$28 \times 28 \times 256$
Conv dw / s2	$3 \times 3 \times 256$ dw	$28 \times 28 \times 256$
Conv / s1	$1 \times 1 \times 256 \times 512$	$14 \times 14 \times 256$
$5\times$ Conv dw / s1	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
$5\times$ Conv / s1	$1 \times 1 \times 512 \times 512$	$14 \times 14 \times 512$
Conv dw / s2	$3 \times 3 \times 512$ dw	$14 \times 14 \times 512$
Conv / s1	$1 \times 1 \times 512 \times 1024$	$7 \times 7 \times 512$
Conv dw / s2	$3 \times 3 \times 1024$ dw	$7 \times 7 \times 1024$
Conv / s1	$1 \times 1 \times 1024 \times 1024$	$7 \times 7 \times 1024$
Avg Pool / s1	Pool 7×7	$7 \times 7 \times 1024$
FC / s1	1024×1000	$1 \times 1 \times 1024$
Softmax / s1	Classifier	$1 \times 1 \times 1000$

Simply defining networks in terms of a small number of Mult-Adds is insufficient. It's also crucial to ensure that these procedures can be carried out efficiently. Unstructured sparse matrix operations, for example, are often slower than dense matrix operations unless the sparsity is quite high. The dense 1×1 convolutions in our model structure handle practically all the computation. This can be done using GEMM functions, which are highly optimized general matrix multiply functions. Convolutions are frequently implemented using GEMMs, but they require an initial memory reordering termed im2col in order to translate it to a GEMM. This method is used, for example, in the popular Caffe package [143]. 1×1 convolutions don't require any memory reordering and can be implemented directly with GEMM, one of

the most optimal numerical linear algebra techniques. As seen in Table 4, MobileNet spends 95 percent of its computing time on 1×1 convolutions, which also have 75 percent of the parameters. The completely connected layer contains nearly all the additional parameters. TensorFlow [142] was used to train MobileNet models using RMSprop [184] and asynchronous gradient descent, comparable to Inception V3 [182]. We utilize fewer regularization and data augmentation approaches when training tiny models than when training large models because small models are less prone to overfitting. We do not employ side heads or label smoothing while training MobileNets, and we also restrict the number of visual distortions by minimizing the size of small crops used in big Inception training [182]. We also discovered that because the depthwise filters have so few parameters, it was critical to apply very little or no weight decay (l2 regularization) to them. All models, regardless of size, were trained using the same training parameters for the ImageNet benchmarks in the next section. Table 4 compares a factorized layer with depthwise convolution, 1×1 pointwise convolution, batch norm, and ReLU nonlinearity to a layer with standard convolutions, batch norm, and ReLU after each convolutional layer. In the depthwise convolutions as well as the first layer, down sampling is handled via striding convolution. Before the completely linked layer, a last average pooling reduces the spatial resolution to 1. MobileNet contains 28 layers when depthwise and pointwise convolutions are counted separately.

Table 4. Resource per Layer Type [136]

Type	Mult-Adds	Parameters
Conv 1×1	94.86%	74.59%
Conv DW 3×3	3.06%	1.06%
Conv 3×3	1.19%	0.02%
Fully Connected	0.18%	24.33%

3.2.5 Mobilenet for Object Detection

In contemporary object detection systems, MobileNet can also be used as a reliable base network. Based on the recent work that won the 2016 COCO challenge [185], we describe the results for MobileNet trained for object detection on COCO data. In Table 5, the Faster-RCNN [162] and SSD [163] frameworks are used to compare MobileNet to VGG and Inception V2 [183]. SSD is tested with 300 input resolution (SSD 300), while Faster-RCNN is tested with both 300 and 600 input resolution (Faster-RCNN 300, Faster-RCNN 600) in our studies. Per image, the Faster-RCNN model assesses 300 RPN proposal boxes. The models are trained and evaluated using COCO train+val except for 8k minival images. MobileNet offers equivalent outcomes to other networks with a fraction of the computational cost and model size in both frameworks.

Table 5. COCO Object Detection Results Comparison Using Different Frameworks and Network Architectures. mAP is Reported with COCO Primary Challenge Metric (AP at IoU=0.50:0.05:0.95)

Framework Resolution	Model	mAP	Bilion Mult-Adds	Milion Parameters
SSD 300	deeplab-VGG Inception V2 MobileNet	21.1 % 22 % 19.13 %	34.9 3.8 1.2	33.1 13.7 6.8
Faster-RCNN 300	VGG Inception V2 MobileNet	22.9% 15.4% 16.4%	64.3 118.2 25.2	138. 13.3 6.1
Faster-RCNN 300	VGG Inception V2 MobileNet	25.7% 21.9% 19.8%	149.6 129.6 30.5	138.5 13.3 6.1

Figure 12 can show the performance of a trained COCO dataset with SSD architecture.

Figure 12. Example Objection Detection Results Using Mobilenet

3.2.6 Single Shot Detector Approach

The Single Shot Detector (SSD) is a feed-forward convolution-based object detector that generates a set of bounding boxes with values and assigns classes to each of them. To be employed in the training process, SSD requires picture input and ground truth boxes, which are a type of square used to designate items to be detected. The architecture of the SSD is shown in Figure 13. Meanwhile, there are various features on SSD that yield high accuracy values, including multi-scale feature maps for detection, convolutional predictors, default boxes, and aspect ratios [161].

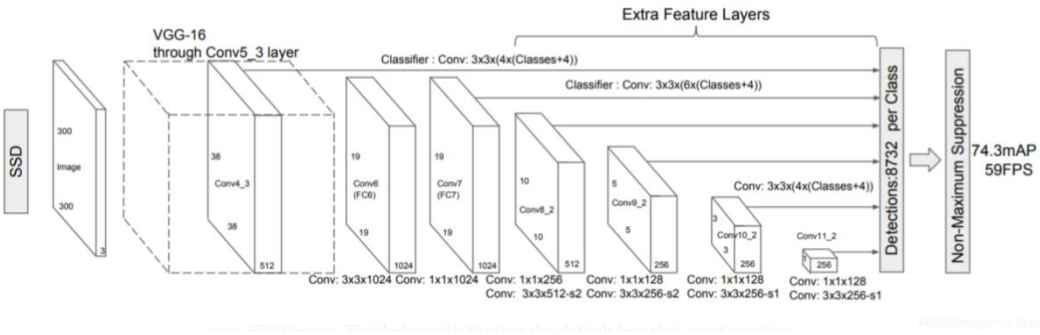


Figure 13. SSD Network Structure

Chapter 4

DEEP LEARNING BASED MARKOV LOCALIZATION

4.1 Introduction

This section will provide information on how the implementation has been done and the steps followed to achieve the results. The overall progress is to have a quadcopter not aware of his initial position within the map hovering on top of the patterns and receives the frames, then decides the pattern detection. Markov weights are also updates in each section and the whole process stops when the final decision comes from with high confidence (more than 99%) from Markov weights.

At the beginning of this chapter, we will go through a comparison with the method mentioned in the reference paper and the proposed method by this thesis. Then the hardware setup will be introduced in addition to how they are used. Afterwards, Deep Learning network training procedure and the outputs will be described. We will continue with the algorithm followed to localize the vehicle. Finally, we will jump into results and their comparison with the literature this work is inspired from.

The proposed method in the original research [1] made use of image analysis with feature extraction from each shot, which was then utilized to create what was deemed to be a pattern of localization. The main concept is that a quadcopter begins flying over a pattern at the beginning of the flight without taking its orientation or location into consideration. On the computer, a map of all the patterns and their placements,

as well as their relative positions, has been saved. The quadcopter is equipped with a camera that is pointed downwards and that sends videos to the main computer. Since the streamed photo is hovering on top of a pattern, it is detected using the above-mentioned method, and an initial prediction as to its location is made. The drone then flies over the next pattern, and the predictions are updated to reflect the new information. Drone localization is completed once the anticipated position's confidence level has reached a predetermined level of confidence.

Because the system is aware of the location, the operation can be carried out on the entire map when it has been completed. When the estimates are made, they are kept as Markov weights, which are updated with each displacement. In robotics, the Markov method is a statistical approach for localization that is used in [1].

In this thesis, we have replaced the feature detection approach with a Deep Learning method to recognize patterns, which is more accurate. A specified number of photographs from each pattern are fed as a dataset to an object detection network, which is then trained in the appropriate manner. The training milestones are based on the pre-trained network from the COCO dataset. Only the last six layers of the network are unfrozen during the training process, and they are frozen again after the training is completed. As a training network, Mobilenet SSD v1 has been selected.

4.2 The Utilized Hardware

The following setup has been used for the system based on the following reasons:

a. Hardware accessibility
b. Deep learning ease of training
c. Available testbed for the actual tests.

Figure 14 shows the architecture of the system which has been used for the setup. In following paragraphs, the system architecture has been described in details.

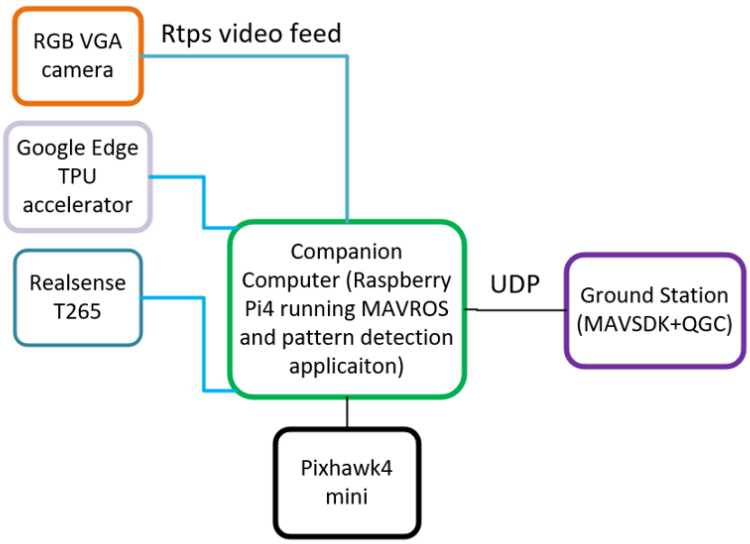


Figure 14. The Architecture of System

As it can be seen from the figure 14, a companion computer (Raspberry Pi4) is mounted on a quadcopter which is getting video feeds from an RGB camera in RTPS format (Real Time Protocol of Streaming) with 30 frames per second. There is an accelerator connected to the companion computer (Google TPU device) to take care of pattern detection. The Intel camera (T265) is also used to acquire Odometry information. Companion computer is also connected to the flight controller (Pixhawk4 mini) for commanding movements. The companion computer is wirelessly communicating with the Ground station using User Datagram Protocol (UDP) to view the position changes and pattern detection decision makings. Figures 15 and 16 show the actual hardware used for this thesis.

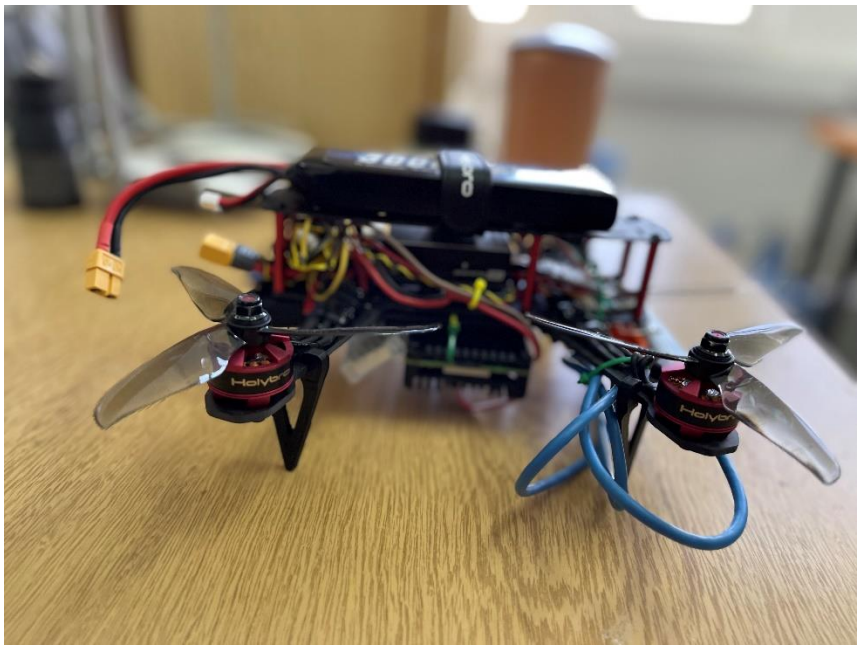


Figure 15. Quadcopter Frame Used for Localization

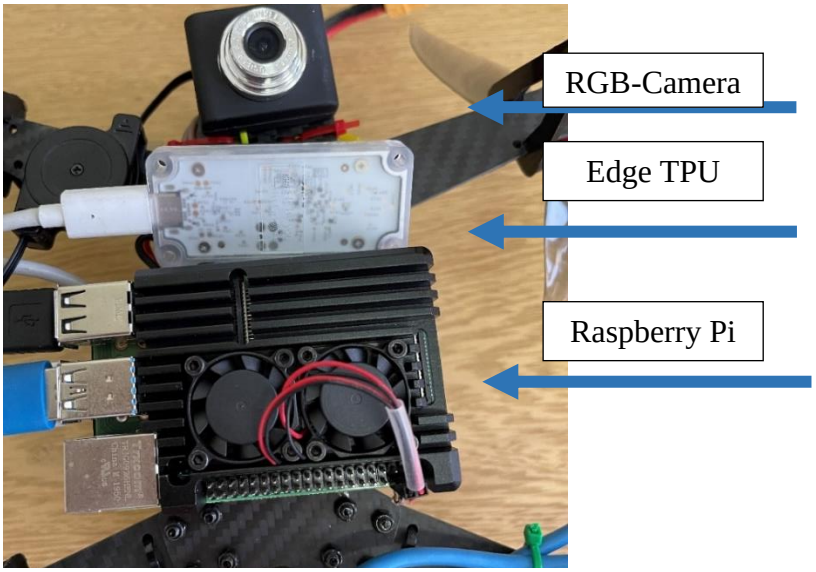


Figure 16. Quadcopter Bottom View Containing the Computing Modules and The Camera

4.2.1 Google Edge TPU

Google IoT includes the Google Edge TPU. They're built to run inferences at the edge with the support of cloud-trained machine learning models. The Google Edge TPU is coupled with the Coral Development Board, which is an ASIC (Application-Specific Integrated Circuit) designed to allow on-device machine learning (Machine

Learning). Coral Development Board is a Single Board Computer (SBC) with Wireless capabilities for high-speed machine learning inferencing. It comes with a removable SoM. (System-on-Module). The operating system on this board is Mendel, a Debian Linux derivative.

The Edge TPU coprocessor can execute 4 trillion Operations Per Second (TOPS) while only consuming 0.5 watts per TOPS. Both C++ and Python programming languages are supported. It makes use of the Mendel Development Utility (MDT), a command-line tool for working with connected Mendel devices. Google TensorFlow Lite and AutoML Vision Edge are both supported. TensorFlow Lite models can only be used using the Python and C++ APIs to make inferences. The EdgeTPU module can be imported to use the Python API, and the edgetpu.h header file can be included to use the C++ API. This development board is mostly used for picture classification and object identification, but it can be utilized for a variety of other tasks. Working with this Dev Board is made easier with good documentation and support. [186] implemented real-time image classification and received quick results, indicating that this development board has a lot of potential for executing real-time ML calculations. [187]. Using a camera module, implemented an object detection demo from video and image categorization. The hardware specifications are listed in Table 6.

Table 6. Hardware Specifications of Coral USB Accelerator.

ML accelerator	Google Edge TPU coprocessor: 4 TOPS (int8); 2 TOPS per watt
Connector	USB 3.0 Type-C* (data/power)
Dimensions	65 m x 30 mm

4.3 Patterns and Their Placement

The map which the drone must occupy to do the localization according to the paper [1] consists of six different patterns shown in Figure 17.

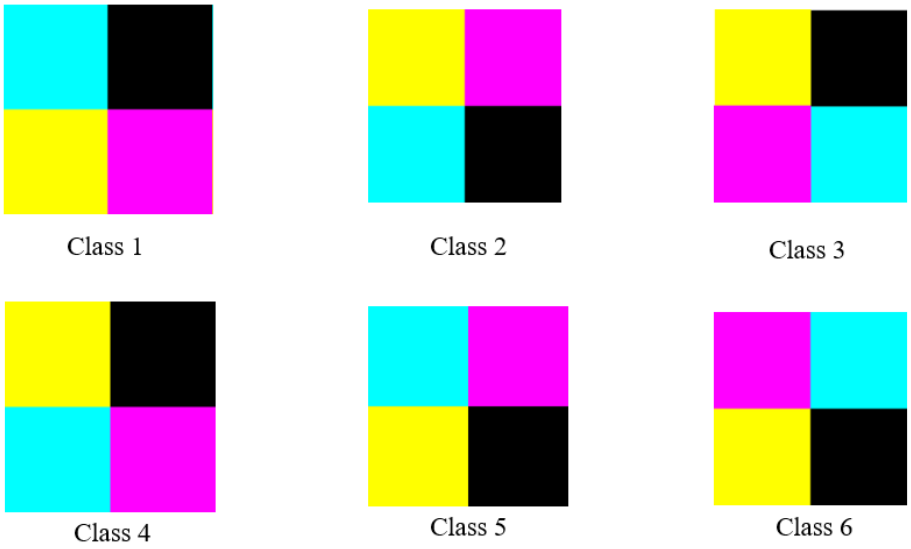


Figure 17. Patterns Classes for Recognition

Each pattern has the dimensions of $50 \times 50\ cm$ which compared to the main reference paper has been reduced to half due to space and expenses' limitations. They are apart from each other for approximately $60cm$. The quadcopter flies over them at a height of $1.0m$ and accordingly each pattern can be seen in the camera separately each time without any interference of other patterns.

These patterns are then put on the ground in a random combination of 16 of them which can be seen in Figure 18. Figure 19 shows the patterns in action and how we put them.

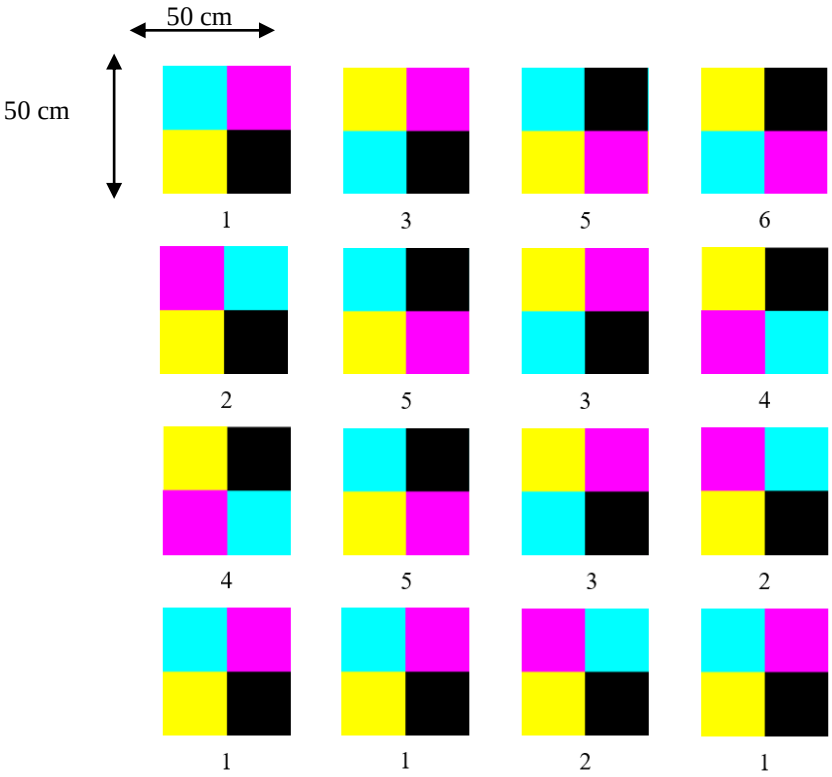


Figure 18. Map of 16 Patterns Retrieved From [1]


Figure 19. Patterns Actual Placement in the Electrical and Electronics Engineering Department

4.4 Markov Localization Theory

Markov localization is a probabilistic localization of a kidnapped robot algorithm. It uses an arbitrary probability density function to represent the robot's position to track its belief state. It does not focus on the location of the car, but rather on the

probability distribution of where it might be. These probabilistic representations allow it to mathematically express many hypotheses [2]. The map has already been preoccupied by the algorithm. When the algorithm receives measurement data from the camera, it refers to the preoccupied map and uses the prediction and measurement update equations to update the probabilities. Prediction update is calculated as below:

$$\overline{Bel}(x_t) = \sum p(x_t|u_t, x_{t-1}) bel(x_{t-1}) dx_{t-1} \quad (8)$$

And the measurement update is:

$$Bel(x_t) = \eta p(z_t|x_t, M) \overline{Bel}(x_t) \quad (9)$$

where x_t denotes the quadrotor's state or location at time instant t, and u_t is the control input. The robot's belief in being at position x_t is expressed by $Bel(x_t)$. Initially, the notion is that all poses have a uniform distribution. Just before including the new observation z_t, which is the sensor readings, $\overline{Bel}(x_t)$ is computed. In addition, η represents the environment map and indicates a normalization factor that ensures that the total of probabilities equals one [188].

4.5 Data Collection

There are totally 6 patterns to be included for training. From each pattern 30 samples have been collected via the same camera which will be on the drone later to take care of the video feed. These photos are then rotated randomly up to 10 degrees maximum and went through a blurred filter to produce more samples. The labeled samples have been fed through the network to be trained. Figure 20 shows how samples are collected.

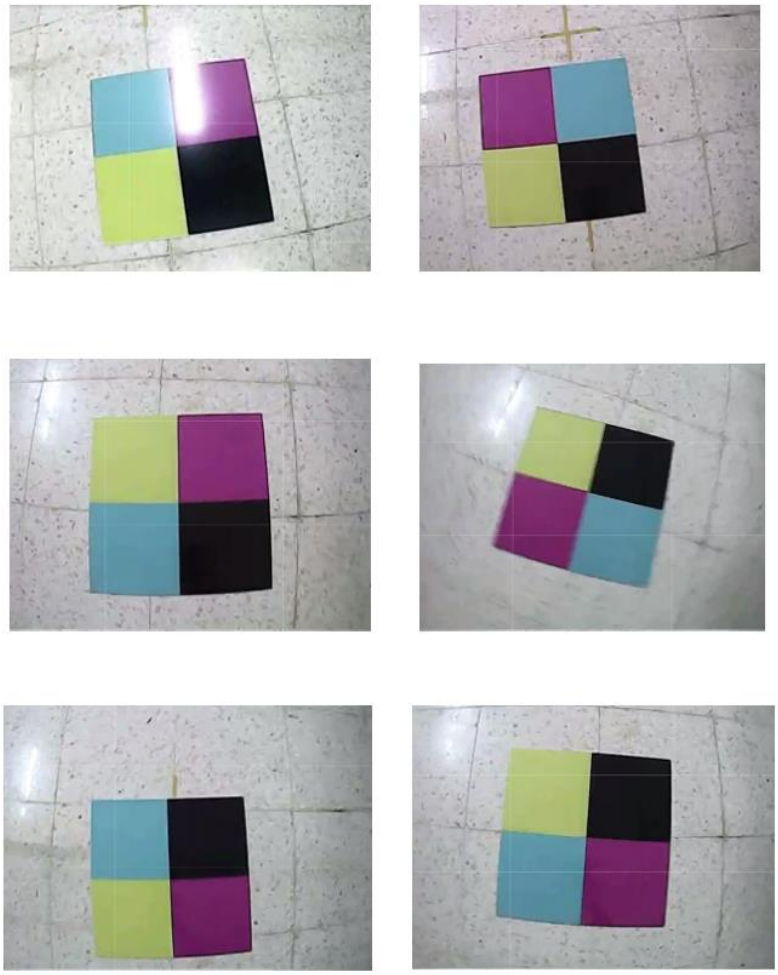


Figure 20. A Sample of Pattern's Data Collected for Training

The collected data then is labeled and added with augmentation and rotation as show some examples of them and how the more data have been prepared by adding noise and augmenting the original one. Figures 21 and 22 show the operation sample applied to the training data.

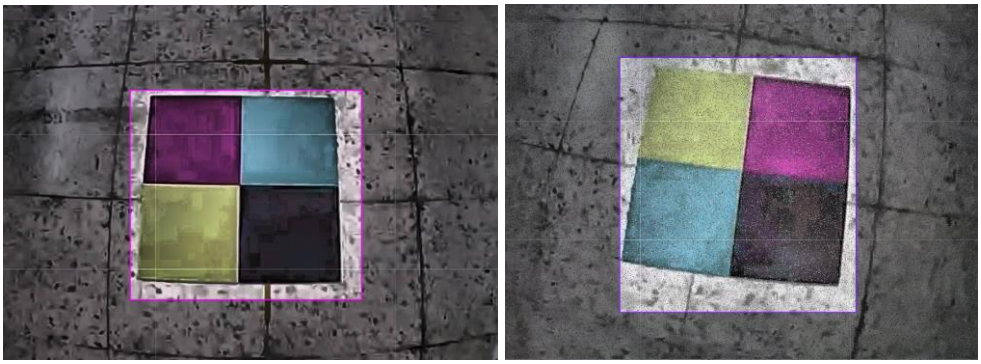


Figure 21. Quality Change (Left) And Introducing Noise (Right) to the Dataset

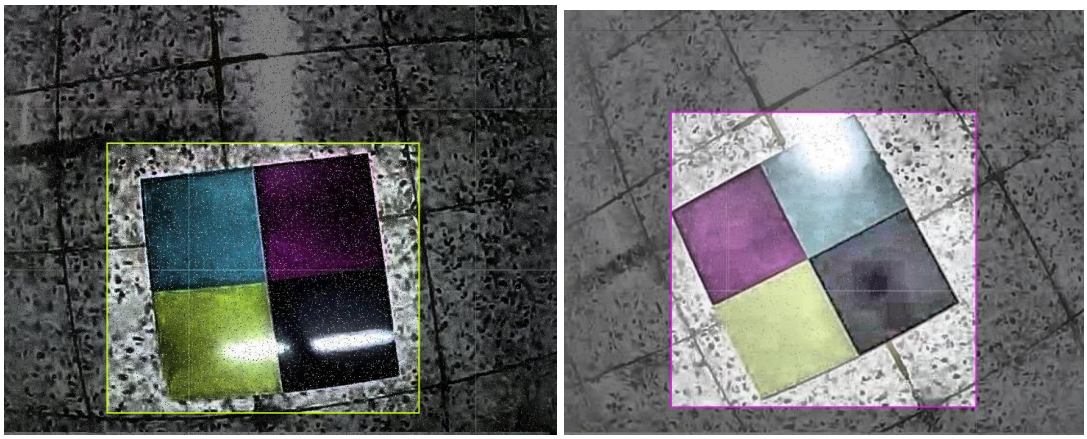


Figure 22. A Sample of Data with Brightness Change (Left) and Rotation Applied (Right)

4.6 Deep Learning Network Training

The network chosen for training is Mobilenet_V1 SSD using TensorFlow. The trained data are then compiled and quantized to be capable of being processed via Edge TPU devices. The network has been trained with transfer learning method using a pre-trained COCO dataset. According to Google Coral Edge TPU device documentation the training steps have been modified to be done up to 5000 epochs, feeding 30 photos from the whole dataset randomly. 80% percent of the collected data have been used for training and 20% for verification of trained network in each training step.

Figure 23 shows the training loss. The horizontal axis shows the number of training epochs. Training loss starts with a random initialized value according to our pipeline configuration that here we used the default one from the hardware provider. As the network learns more about the dataset and tests with evaluation data provided for it, this loss decreases till reaching an optimum value.

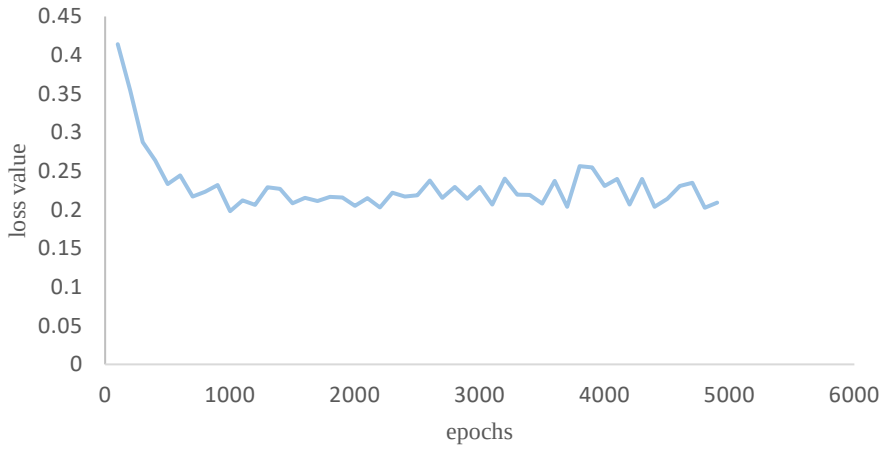


Figure 23. Loss Values During Object Detection Training

Figure 24 represents that our detection boxes can perform an overlapping accuracy of almost 75% between ground truth of data and the represented data up to 80% only after near 1200 steps of training.

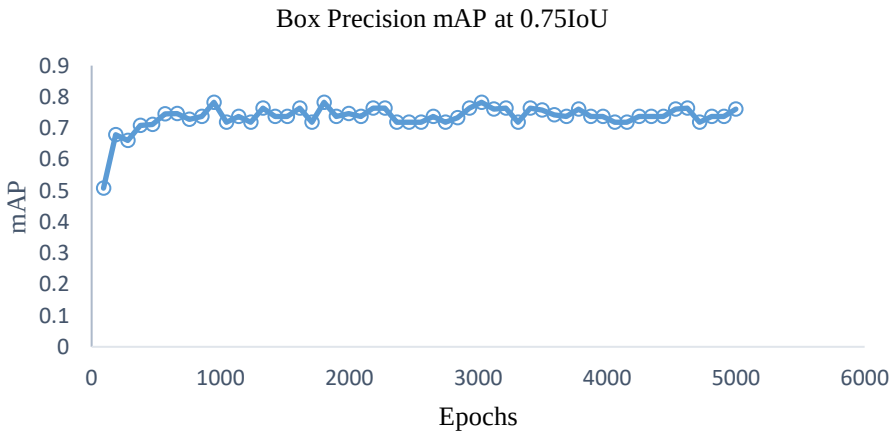


Figure 24. Mean Average Precision for Detection Boxes at 0.75IoU

Precision of detection boxes as can be inferred from Figure 25 can barely reach to 70% on average over the whole training dataset which is compensated by our proposed majority voting decision making method later to achieve near 100% confidence of the detected pattern within a specific timeframe.

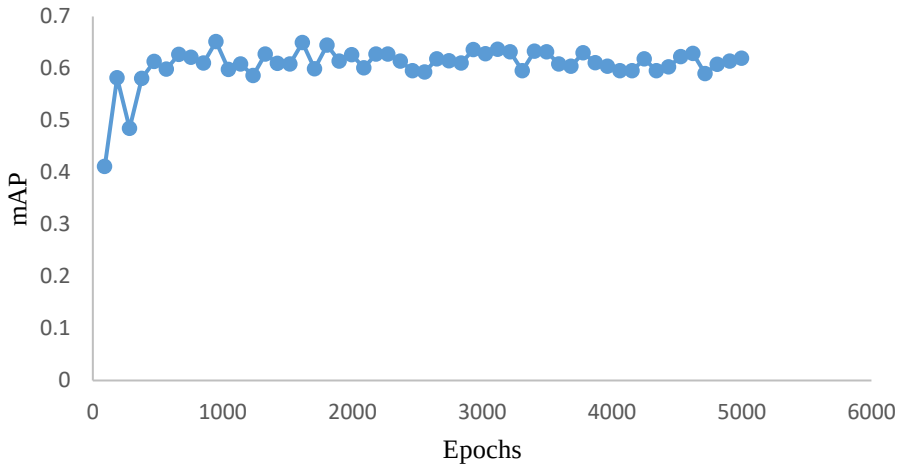


Figure 25. The Precision Detection for Each Epoch of Training

As it can be seen in Fig. 26, after 1000 steps the network is ready for object detection since learning rate has reached to zero. The learning rate starts from a random number and it shows the network's convergence as after each step the training data is compared to the evaluation data.

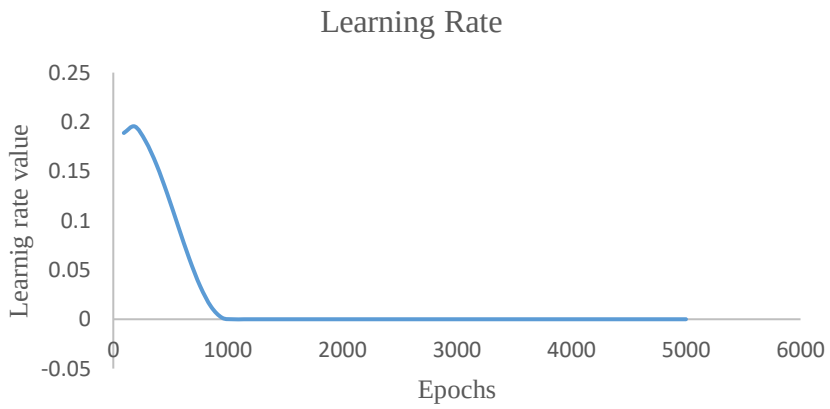


Figure 26. Learning Rate Coefficient Per Epochs of Training

4.7 The Algorithm of Implementation

On the Grid-Map of the world the location of the quadrotor is not known at the beginning and it is considered as a kidnapped robot. It starts by hovering on top of an

initially random unknown pattern. Each detected frame within each second is fed to a list by software and after 90 detection frames, the mode of these decisions is chosen.

The aim here is to find the robot's place on this map. It might be located anywhere on the map, and the likelihood of finding it in all cells is $\frac{1}{16}$. Then when a pattern is observed, the frame is transmitted to the image processing algorithm, which retrieves the information. After all the locations have been discovered, the likelihood is divided based on the number of probable places and the grid-map is updated accordingly. A prediction update is performed whenever the quadrotor traverses from one cell in the grid-map to another cell which can start from anywhere on the map and go in any straight direction. This is referred to as the prediction update. In each cell, the prediction and measurement updates will be repeated until the chance of the quadrotor being localized is 100 percent, at which point the quadrotor will be localized and aware of its position on the map. Figure 27 shows the algorithm followed for localization.

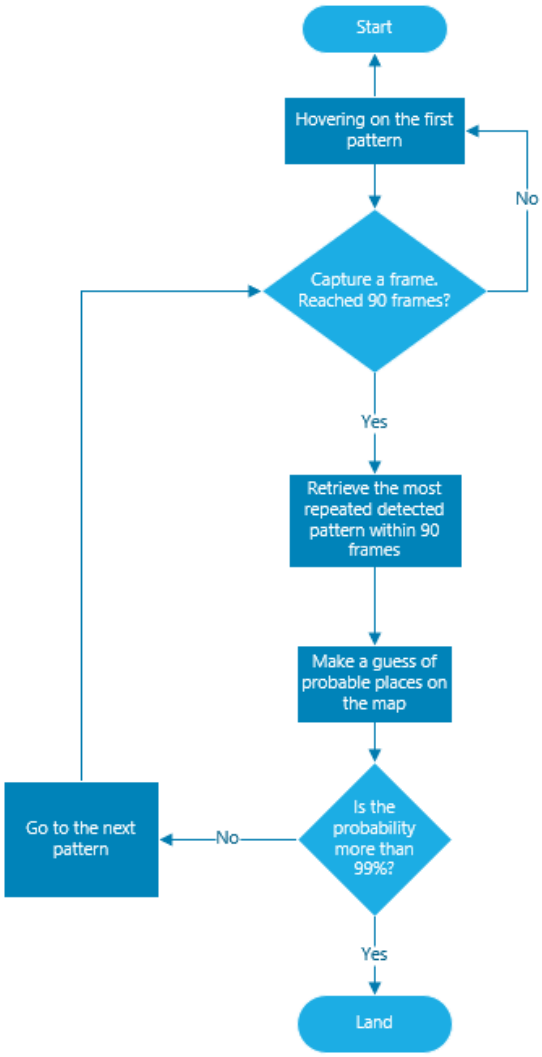


Figure 27. The Algorithm for Localization

4.8 Results and Outputs

First, we have tested our pattern detection and decision-making method with hovering the drone on each of 6 classes separately. Because we do not have more than 80% confidence in pattern detection from each frame and at the same time, we need to decide with a confidence of 100%, we have introduced majority voting method to make the final decision about the correct detection of each pattern. In this method, the drone hovers on top of each pattern for 3 seconds and captures 90 frames. Since there are some wrong detections may happen within these 90 frames, the corresponding numbers of each detected class is added to a list. This means our

list contains 90 detected classes after 3 seconds. By taking the mode of this list we can find out how many times a pattern has been detected the most and choose it as our final decision for the pattern detection.

The following tables can visualize the confusion table. Each table is for hovering on an arbitrary class. The highlighted columns state the pattern we have flown on top of. We have chosen odd numbers of frames with a specific gap of frames to make the tables short. The percentages in each row represent the share of a detected class by our trained neural network among other patterns up to that specific number of frames. It can be inferred that after 90 frames we have only one class which has been detected the most and we choose it as our right detected pattern with 100% confidence. Tables 7 to 12 can show this for each pattern respectively.

Table 7. Detection Accuracy of Pattern 1 within 90 detection frames.

No. of Frames	Pat1	Pat2	Pat3	Pat4	Pat5	Pat6
	Detection Accuracy					
3	33%	33%	0	0	0	33%
11	36%	55%	0	0	0	9%
19	42%	53%	0	0	0	5%
27	52%	44%	0	0	0	4%
35	63%	34%	0	0	0	3%
43	75%	24%	0	0	0	2%
51	78%	20%	0	0	0	2%
59	81%	18%	0	0	0	1%
67	81%	18%	0	0	0	1%
75	79%	20%	0	0	0	1%
90	78%	21%	0	0	0	1%

Table 8. Detection Accuracy of Pattern 2 Within 90 Detection Frames.

No. of frames	Pat1	Pat2	Pat3	Pat4	Pat5	Pat6
	Detection Accuracy					
3	64%	9%	0%	0%	27%	0%
11	68%	16%	0%	0%	16%	0%
19	59%	30%	0%	0%	11%	0%
27	46%	46%	0%	0%	9%	0%
35	31%	63%	0%	0%	6%	0%
43	27%	68%	0%	0%	5%	0%
51	24%	72%	0%	0%	4%	0%
59	22%	74%	0%	0%	4%	0%
67	20%	77%	0%	0%	4%	0%
90	18%	79%	0%	0%	3%	0%

Table 9. Detection Accuracy of Pattern 3 Within 90 Detection Frames.

No. of frames	Detection Accuracy					
	Pat 1	Pat2	Pat 3	Pat4	Pat5	Pat 6
3	0%	0%	0%	0%	100%	0%
11	0%	0%	0%	0%	100%	0%
19	0%	0%	0%	0%	100%	0%
27	0%	0%	11%	0%	89%	0%
35	0%	0%	31%	0%	69%	0%
43	0%	0%	53%	0%	47%	0%
51	0%	0%	59%	0%	41%	0%
59	0%	0%	64%	0%	36%	0%
67	0%	0%	67%	0%	33%	0%
75	0%	1%	69%	0%	30%	0%
90	0%	2%	67%	0%	26%	4%

Table 10. Detection Accuracy of Pattern 4 Within 90 Detection Frames.

No. of Frames	Detection Accuracy					
	Pat1	Pat2	Pat3	Pat4	Pat5	Pat6
3	0%	0%	0%	0%	100%	0%
11	0%	0%	0%	36%	64%	0%
19	0%	0%	0%	37%	58%	5%
27	0%	0%	0%	26%	70%	4%
35	0%	0%	0%	26%	69%	6%
43	0%	0%	0%	39%	51%	10%
51	0%	0%	0%	42%	46%	12%
59	0%	0%	0%	48%	42%	10%
67	0%	0%	0%	52%	38%	10%
90	0%	0%	0%	56%	35%	10%

Table 11. Detection Accuracy of Pattern 5 within 90 Detection Frames.

No. of Frames	Detection accuracy					
	Pat1	Pat2	Pat3	Pat4	Pat5	Pat6
3	0%	0%	0%	0%	100%	0%
11	0%	0%	0%	9%	91%	0%
19	0%	0%	0%	16%	84%	0%
27	0%	0%	0%	11%	89%	0%
35	0%	0%	0%	9%	91%	0%
43	0%	0%	0%	6%	94%	0%
51	0%	0%	0%	5%	95%	0%
59	0%	0%	0%	4%	96%	0%
67	0%	0%	0%	4%	96%	0%
75	0%	0%	0%	4%	96%	0%
90	0%	0%	0%	3%	97%	0%

Table 12. Detection Accuracy of Pattern 6 Within 90 Detection Frames.

No. of frames	Detection Accuracy					
	Pat1	Pat2	Pat3	Pat4	Pat5	Pat6
3	0%	0%	0%	0%	100%	0%
11	0%	0%	0%	0%	64%	36%
19	0%	0%	0%	0%	58%	42%
27	0%	0%	0%	0%	48%	52%
35	0%	0%	0%	0%	37%	63%
43	0%	0%	0%	0%	25%	75%
51	0%	0%	0%	0%	22%	78%
59	0%	0%	0%	0%	19%	81%
67	0%	0%	0%	0%	18%	82%
75	0%	0%	0%	0%	16%	84%
90	0%	0%	0%	0%	14%	86%

As the decision is made via the above tables, we have two outputs on top of each frame on our ground station. Figure 28 shows that on the left side, the number shown is our majority voting decision making algorithm output while at the same time the number on right side is showing the current frame can only be Pattern 6 with 79% of confidence.

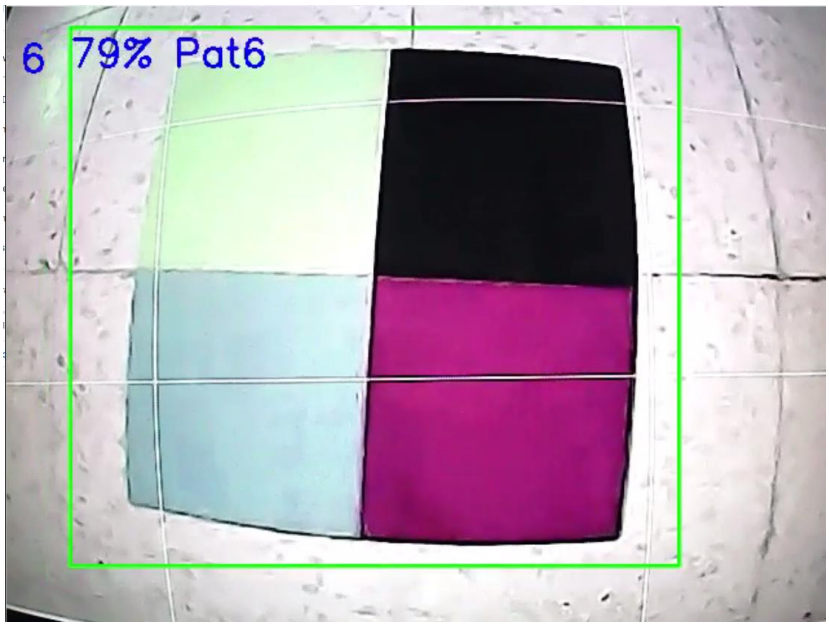


Figure 28. Comparison Between Pattern Recognition Confidence and Final Decision

Secondly, we started implementing Markov algorithm to localize our quadcopter on the map. Markov algorithm works easier when no noise is present in each measurement (here each detected pattern) and we have eliminated that noise by using our majority voting algorithm for decision making.

Figure 29 shows the first route. At the beginning the drone hovers on top of the bottom right pattern (pattern 1) without any information of its location. Pattern 1 is detected and since we have Pattern 1 four times repeated in the whole map, the probability of being on top of any of them is 25% as the map is already predefined by our computer. At this point, for the sake of optimization for our algorithm, we have added border condition checks. To be clearer, if the drone is on top of Pattern 1, we send the drone to the right side. This means the next 3 seconds we are going to have no pattern detected. We have limited this time to one second which means localization is done in total 4 seconds. At this moment the localization process is done and the drone is aware of its location on the map with 100% confidence.

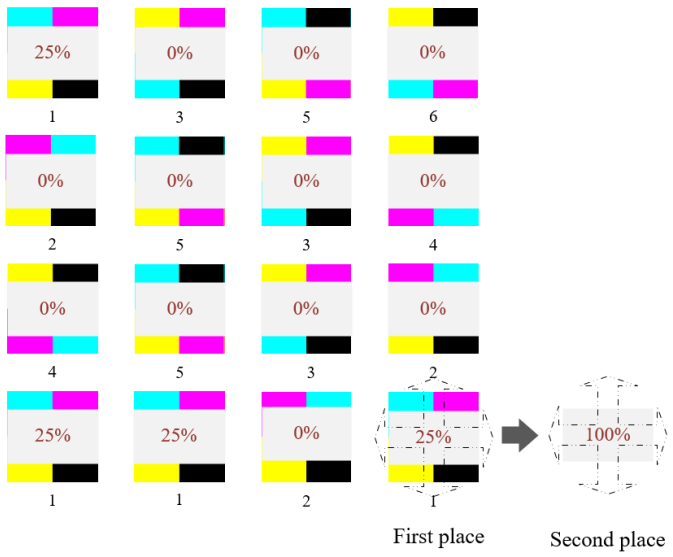


Figure 29. Localization Using Route 1 (Left to Right Is Start to End)

With the proposed method for optimization, we can ensure to have the least amount of time needed for localization. For instance, pattern 6 will take only 3 seconds to achieve localization and it is only one on the whole map leading to a 100% confidence at the first period of detection.

There are indeed middle patterns that the drone may start to fly off from. In this case for having an optimized timing to localize, the predefined map condition will prevent the drone to go to a direction which localization may take longer. For instance, as shown in Figure 30, if we start from Pattern 3 in the middle, going to the left will make us longer to do localization and we choose moving to right side which after two patterns localization is achieved.

Figure 30. Localization Using Route 3

4.9 Comparison of Results with The Concurrent Methods

This implementation had been able to do the detection within 90 frames over each detection period. Below we have the comparison between the original paper and the ones we have proposed here as results. Figure 31 shows the convergence of the

estimated error from the prediction and it can be clearly seen that after about 6 second, localization has been successful.

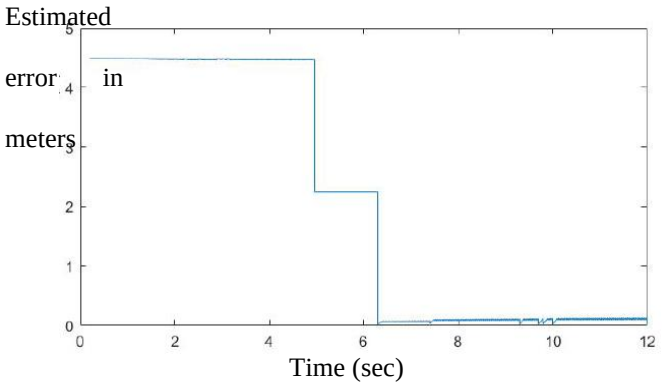


Figure 31. Estimated Time to Localize the Drone from The Concurrent Solution

Table 13 shows how our proposed method can be compared to the concurrent method already implemented by the reference research paper.

Table 13. Comparison Between Thesis Proposed Method the One In [1]

	Overall time consumption for a localization task	Overall number of patterns to achieve near 100% confidence for localization
Concurrent method	Approx. 6 seconds	4
Implemented method in this thesis	Between 4 to 6 seconds	Between 1 to 2

Chapter 5

CONCLUSION AND FUTURE WORKS

5.1 Conclusions

In this work we localized a quadcopter using deep learning and Markov algorithm. Localization is among one the most favorite topics in robotics and can be done in many ways that we have earlier mentioned in this thesis. We have brought an idea of using pattern in 6 different classes to be used for feeding our training network. By using Google Coral Edge TPU which is one of the widely used embedded tensor processors in the industry, we have made the decision making for each pattern detection scenario. Near total of 300 images have been fed to the training network to the retrained in 5000 epochs. The Tensor processor decisions were then given most of the voting within each 91 frames received in real time. The drone hovered over 2 to 3 consecutive patterns and Markov decisions for the location were with 100% confidence in all tested case scenarios relying on a pre-defined map of pattern's placement relative to each other. Each time the drone can come to a final accurate decision of the detected pattern within three seconds and then it traverses to the next one.

On the other hand, this thesis has contributed to the performance of detection. Tables 7 to 12 show that within each inference of 90 frames fed to the detection network how the accuracy of final decision increases based on Sum and Product probability rules.

5.2 Future Work

This thesis was the first practical implementation of a research paper with a different method. In the paper only the simulation results were mentioned and feature extraction had been used to detect patterns. Since the time was limited to have the hardware ready and at the same time do the implementation, we had to limit our work within these 16 patterns. Markov algorithm has worked very fast here since there were no uncertainty in our decision making which if the number of patterns are extended, indeed the certainty will be noisy and we will need more patterns or other routes to be surfed as well for localization that could be a part of any future research.

Additionally, another work indeed can be to compare the accuracy and speed of detection using feature extraction and deep learning. This must also bring the hardware change into account since TPU will not be used anymore in feature extraction.

The other work is to make the patterns smaller and have more of different and more complex ones with more colors. In this case detection with deep learning can be compared with the current method.

Finally, this localization method can be compared to any other ones available in market to give the insight of how this method performs compared to the other ones available in the market now.

REFERENCES

[1] F. T. Ali, O. A. Fahmy, and A. A. El-Badawy, “An indoor vision-based markov localization technique of a quadrotor,” 2019, doi: 10.1109/ICVES.2019.8906468.

[2] Y. Shoukry, W. F. Abdelfatah, and S. A. Hammad, “Real-time Markov localization for autonomous UGV,” 2009, doi: 10.1109/IDT.2009.5404156.

[3] M. Zhang, S. Zhang, and J. Cao, “Fusing received signal strength from multiple access points for WLAN user location estimation,” 2008, doi: 10.1109/ICICSE.2008.24.

[4] G. Giaglis and A. Pateli, “On the potential use of mobile positioning technologies in indoor environments,” *Proc. ...*, 2002.

[5] P. Kourouthanassis, L. Koukara, C. Lazaris, and K. Thiveos, “Last-mile supply chain management: Mygrocer innovative business and technology framework,” 2001.

[6] F. Bellotti, R. Berta, A. De Gloria, and M. Margarone, “User testing a hypermedia tour guide,” *IEEE Pervasive Computing*, vol. 1, no. 2. 2002, doi: 10.1109/MPRV.2002.1012335.

[7] “Chapter 2. Indoor Positioning Services and Location-Based Recommendations,” *Libr. Technol. Rep.*, vol. 53, no. 1, 2017.

[8]L. Calderoni, M. Ferrara, A. Franco, and D. Maio, “Indoor localization in a hospital environment using Random Forest classifiers,” *Expert Syst. Appl.*, vol. 42, no. 1, 2015, doi: 10.1016/j.eswa.2014.07.042.

[9]W. Shuaieb *et al.*, “RFID RSS Fingerprinting System for Wearable Human Activity Recognition,” *Futur. Internet*, vol. 12, p. 33, 2020, doi: 10.3390/fi12020033.

[10] Y. Ji, S. Biaz, S. Wu, and B. Qi, “Impact of building environment on the performance of dynamic indoor localization,” 2006, doi: 10.1109/WAMICON.2006.351900.

[11] J. O. Nilsson, J. Rantakokko, P. Händel, I. Skog, M. Ohlsson, and K. V. S. Hari, “Accurate indoor positioning of firefighters using dual foot-mounted inertial sensors and inter-agent ranging,” 2014, doi: 10.1109/PLANS.2014.6851424.

[12] T. Islam, H. A. Rahman, and M. A. Syrus, “Fire detection system with indoor localization using ZigBee based wireless sensor network,” 2015, doi: 10.1109/ICIEV.2015.7334000.

[13] S. O. Al-Jazzar and Y. Jaradat, “AOA-based drone localization using wireless sensor-doublets,” *Phys. Commun.*, vol. 42, 2020, doi: 10.1016/j.phycom.2020.101160.

[14] S. Nirjon, J. Liu, G. DeJean, B. Priyantha, Y. Jin, and T. Hart, “COIN-GPS: Indoor localization from direct GPS receiving,” 2014, doi:

10.1145/2594368.2594378.

[15] X. Wan and X. Zhan, “The research of indoor navigation system using pseudolites,” in *Procedia Engineering*, 2011, vol. 15, doi: 10.1016/j.proeng.2011.08.268.

[16] R. Xu, W. Chen, Y. Xu, and S. Ji, “A new indoor positioning system architecture using gps signals,” *Sensors (Switzerland)*, vol. 15, no. 5, 2015, doi: 10.3390/s150510074.

[17] G. Fusco and J. M. Coughlan, “Indoor localization using computer vision and visual-inertial odometry,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2018, vol. 10897 LNCS, doi: 10.1007/978-3-319-94274-2_13.

[18] E. Bekir, *Introduction to modern navigation systems*. 2007.

[19] G. Hu, W. Zhang, H. Wan, and X. Li, “Improving the heading accuracy in indoor pedestrian navigation based on a decision tree and kalman filter,” *Sensors (Switzerland)*, vol. 20, no. 6, 2020, doi: 10.3390/s20061578.

[20] S. Y. Jung, S. Hann, and C. S. Park, “TDOA-based optical wireless indoor localization using LED ceiling lamps,” *IEEE Trans. Consum. Electron.*, vol. 57, no. 4, 2011, doi: 10.1109/TCE.2011.6131130.

[21] Z. Chen, Q. Zhu, and Y. C. Soh, “Smartphone Inertial Sensor-Based Indoor

Localization and Tracking with iBeacon Corrections," *IEEE Trans. Ind. Informatics*, vol. 12, no. 4, 2016, doi: 10.1109/TII.2016.2579265.

[22] X. Wu, R. Shen, L. Fu, X. Tian, P. Liu, and X. Wang, "IBILL: Using iBeacon and Inertial Sensors for Accurate Indoor Localization in Large Open Areas," *IEEE Access*, vol. 5, 2017, doi: 10.1109/ACCESS.2017.2726088.

[23] H. Yang *et al.*, "Smartphone-Based Indoor Localization System Using Inertial Sensor and Acoustic Transmitter/Receiver," *IEEE Sens. J.*, vol. 16, no. 22, 2016, doi: 10.1109/JSEN.2016.2604424.

[24] T. D. Vy, T. L. N. Nguyen, and Y. Shin, "A smartphone indoor localization using inertial sensors and single wi-fi access point," 2019, doi: 10.1109/IPIN.2019.8911749.

[25] J. Liu and R. Jain, "Survey of Wireless Based Indoor Localization Technologies," *Washingt. Univ. St. Louis*, 2014.

[26] Z. Farid, R. Nordin, and M. Ismail, "Recent advances in wireless indoor localization techniques and system," *Journal of Computer Networks and Communications*, vol. 2013. 2013, doi: 10.1155/2013/185138.

[27] A. Boukerche, H. A. B. F. Oliveira, E. F. Nakamura, and A. A. F. Loureiro, "Localization systems for wireless sensor networks," *IEEE Wirel. Commun.*, vol. 14, no. 6, 2007, doi: 10.1109/MWC.2007.4407221.

[28] S. Ahson and M. Ilyas, *Location-based services handbook: Applications, technologies, and security*. 2010.

[29] T. N. Lin and P. C. Lin, "Performance comparison of indoor positioning techniques based on location fingerprinting in wireless networks," in *2005 International Conference on Wireless Networks, Communications and Mobile Computing*, 2005, vol. 2, doi: 10.1109/WIRLES.2005.1549647.

[30] F. Reichenbach and D. Timmermann, "Indoor localization with low complexity in wireless sensor networks," 2006, doi: 10.1109/INDIN.2006.275737.

[31] A. Alkhatib, "A Review of Wireless Sensor Networks Applications," 2011.

[32] J. J. Robles, "Indoor localization based on wireless sensor networks," *AEU - Int. J. Electron. Commun.*, vol. 68, no. 7, 2014, doi: 10.1016/j.aeue.2014.04.004.

[33] A. Maddumabandara, H. Leung, and M. Liu, "Experimental Evaluation of Indoor Localization Using Wireless Sensor Networks," *IEEE Sens. J.*, vol. 15, no. 9, 2015, doi: 10.1109/JSEN.2015.2438193.

[34] J. Sangthong, J. Thongkam, and S. Promwong, "Indoor wireless sensor network localization using RSSI based weighting algorithm method," 2020, doi: 10.1109/ICEAST50382.2020.9165300.

[35] Z. Munadhil, S. K. Gharghan, A. H. Mutlag, A. Al-Naji, and J. Chahl,

"Neural Network-Based Alzheimer's Patient Localization for Wireless Sensor Network in an Indoor Environment," *IEEE Access*, vol. 8, 2020, doi: 10.1109/ACCESS.2020.3016832.

[36] T. He, C. Huang, B. M. Blum, J. A. Stankovic, and T. Abdelzaher, "Range-Free Localization Schemes for Large Scale Sensor Networks," 2003, doi: 10.1145/938985.938995.

[37] C. An, A. Timm-Giel, and C. Goerg, "Virtual sensor network lifeline for communications in fire fighting rescue scenarios," 2009, doi: 10.1109/VETECF.2009.5379094.

[38] "Squad positioning system helps fight fires and save lives," 2009. https://newatlas.com/squad-fire-fighter-positioning-system/11929/ (accessed Aug. 09, 2021).

[39] J. J. Robles, M. Deicke, and R. Lehnert, "3D fingerprint-based localization for wireless sensor networks," 2010, doi: 10.1109/WPNC.2010.5653477.

[40] Y. Chapre, P. Mohapatra, S. Jha, and A. Seneviratne, "Received signal strength indicator and its analysis in a typical WLAN system (short paper)," 2013, doi: 10.1109/LCN.2013.6761255.

[41] K. C. Cheung, S. S. Intille, and K. Larson, "An inexpensive Bluetooth-based indoor positioning hack," *Proc. UbiComp'06 Ext. Abstr.*, 2006.

[42] C. N. Huang and C. T. Chan, “ZigBee-based indoor location system by K-nearest neighbor algorithm with weighted RSSI,” in *Procedia Computer Science*, 2011, vol. 5, doi: 10.1016/j.procs.2011.07.010.`

[43] M. Yavari and B. G. Nickerson, “Ultra wideband wireless positioning systems,” *Dept. Fac. Comput. Sci., Univ. New Brunswick, Fredericton, NB, Canada, Tech. Rep. TR14-230*, no. 506, 2014.

[44] A. Popleteev, “Indoor Localization Using Ambient FM Radio RSS Fingerprinting: A 9-Month Study,” 2017, doi: 10.1109/CIT.2017.57.

[45] Y. Chen, D. Lymberopoulos, J. Liu, and B. Priyantha, “Indoor localization using FM signals,” *IEEE Trans. Mob. Comput.*, vol. 12, no. 8, 2013, doi: 10.1109/TMC.2013.58.

[46] J. Xiao, Z. Zhou, Y. Yi, and L. M. Ni, “A survey on wireless indoor localization from the device perspective,” *ACM Computing Surveys*, vol. 49, no. 2. 2016, doi: 10.1145/2933232.

[47] A. Popleteev, V. Osmani, and O. Mayora, “Investigation of indoor localization with ambient FM radio stations,” 2012, doi: 10.1109/PerCom.2012.6199864.

[48] M. Chai, C. Li, and H. Huang, “A New Indoor Positioning Algorithm of Cellular and Wi-Fi Networks,” *J. Navig.*, vol. 73, no. 3, 2020, doi: 10.1017/S0373463319000742.

[49] H. Liu, H. Darabi, P. Banerjee, and J. Liu, “Survey of wireless indoor positioning techniques and systems,” *IEEE Transactions on Systems, Man and Cybernetics Part C: Applications and Reviews*, vol. 37, no. 6. 2007, doi: 10.1109/TSMCC.2007.905750.

[50] V. Otsason, A. Varshavsky, A. LaMarca, and E. De Lara, “Accurate GSM indoor localization,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2005, vol. 3660 LNCS, doi: 10.1007/11551201_9.

[51] A. Varshavsky, E. de Lara, J. Hightower, A. LaMarca, and V. Otsason, “GSM indoor localization,” *Pervasive Mob. Comput.*, vol. 3, no. 6, 2007, doi: 10.1016/j.pmcj.2007.07.004.

[52] M. Driusso, C. Marshall, M. Sabathy, F. Knutti, H. Mathis, and F. Babich, “Indoor positioning using LTE signals,” 2016, doi: 10.1109/IPIN.2016.7743656.

[53] A. A. Abdallah, K. Shamaei, and Z. M. Kassas, “Indoor positioning based on LTE carrier phase measurements and an inertial measurement unit,” 2018, doi: 10.33012/2018.16073.

[54] A. A. Abdallah, K. Shamaei, and Z. M. Kassas, “Indoor localization with LTE carrier phase measurements and synthetic aperture antenna array,” 2019, doi: 10.33012/2019.17030.

[55] J. Turkka, T. Hiltunen, R. U. Mondal, and T. Ristaniemi, “Performance

evaluation of LTE radio fingerprinting using field measurements," in *Proceedings of the International Symposium on Wireless Communication Systems*, 2015, vol. 2016-April, doi: 10.1109/ISWCS.2015.7454387.

[56] A. Aguilar-Garcia, S. Fortes, E. Colin, and R. Barco, "Enhancing RFID indoor localization with cellular technologies," *Eurasip J. Wirel. Commun. Netw.*, vol. 2015, no. 1, 2015, doi: 10.1186/s13638-015-0444-9.

[57] E. Martin, O. Vinyals, G. Friedland, and R. Bajcsy, "Precise indoor localization using smart phones," 2010, doi: 10.1145/1873951.1874078.

[58] R. Alexandrou, H. Papadopoulos, and A. Konstantinidis, "Smartphone Indoor Localization Using Bio-inspired Modeling," 2020.

[59] J. Machaj and P. Brida, "Impact of optimization algorithms on hybrid indoor positioning based on GSM and Wi-Fi signals," in *Concurrency Computation* , 2017, vol. 29, no. 23, doi: 10.1002/cpe.3911.

[60] Z. Chen, H. Zou, H. Jiang, Q. Zhu, Y. C. Soh, and L. Xie, "Fusion of WiFi, smartphone sensors and landmarks using the kalman filter for indoor localization," *Sensors (Switzerland)*, vol. 15, no. 1, 2015, doi: 10.3390/s150100715.

[61] E. Navarro, B. Peuker, M. Quan, A. C. Clark, and J. Jipson, "Wi-Fi Localization Using RSSI Fingerprinting," *Test*, 2010.

[62] N. Le Dortz, F. Gain, and P. Zetterberg, “WiFi fingerprint indoor positioning system using probability distribution comparison,” 2012, doi: 10.1109/ICASSP.2012.6288374.

[63] A. H. Lashkari, B. Parhizkar, and M. N. A. Ngan, “WIFI-based indoor positioning system,” *2nd Int. Conf. Comput. Netw. Technol. ICCNT 2010*, no. June 2016, pp. 76–78, 2010, doi: 10.1109/ICCNT.2010.33.

[64] Y. Wang and X. Xu, “Indoor localization service based on the data fusion of Wi-Fi and RFID,” 2016, doi: 10.1109/ICWS.2016.31.

[65] C. Yang and H. R. Shao, “WiFi-based indoor positioning,” *IEEE Commun. Mag.*, vol. 53, no. 3, pp. 150–157, Mar. 2015, doi: 10.1109/MCOM.2015.7060497.

[66] A. Habibi Lashkari, B. Parhizkar, and M. Ngan, “WIFI-based indoor positioning system,” *Comput. Netw. Technol. Int. Conf.*, vol. 0, pp. 76–78, 2010, doi: 10.1109/ICCNT.2010.33.

[67] A. H. Salamah, M. Tamazin, M. A. Sharkas, and M. Khedr, “An enhanced WiFi indoor localization System based on machine learning,” 2016, doi: 10.1109/IPIN.2016.7743586.

[68] V. Bianchi, P. Ciampolini, and I. De Munari, “RSSI-Based Indoor Localization and Identification for ZigBee Wireless Sensor Networks in Smart Homes,” *IEEE Trans. Instrum. Meas.*, vol. 68, no. 2, 2019, doi:

10.1109/TIM.2018.2851675.

[69] M. Sugano, T. Kawazoe, Y. Ohta, and M. Murata, “Indoor localization system using RSSI measurement of wireless sensor network based on ZigBee standard,” 2006.

[70] J. Niu, B. Wang, L. Shu, T. Q. Duong, and Y. Chen, “ZIL: An Energy-Efficient Indoor Localization System Using ZigBee Radio to Detect WiFi Fingerprints,” *IEEE J. Sel. Areas Commun.*, vol. 33, no. 7, 2015, doi: 10.1109/JSAC.2015.2430171.

[71] C. W. Ou *et al.*, “A ZigBee position technique for indoor localization based on proximity learning,” 2017, doi: 10.1109/ICMA.2017.8015931.

[72] U. M. Qureshi, Z. Umair, and G. P. Hancke, “Evaluating the implications of varying bluetooth low energy (BLE) transmission power levels on wireless indoor localization accuracy and precision,” *Sensors (Switzerland)*, vol. 19, no. 15, 2019, doi: 10.3390/s19153282.

[73] M. Altini, D. Brunelli, E. Farella, and L. Benini, “Bluetooth indoor localization with multiple neural networks,” 2010, doi: 10.1109/ISWPC.2010.5483748.

[74] F. Zafari, A. Gkelias, and K. K. Leung, “A Survey of Indoor Localization Systems and Technologies,” *IEEE Commun. Surv. Tutorials*, vol. 21, no. 3, 2019, doi: 10.1109/COMST.2019.2911558.

[75] M. Ridolfi, S. van de Velde, H. Steendam, and E. De Poorter, “Analysis of the scalability of UWB indoor localization solutions for high user densities,” *Sensors (Switzerland)*, vol. 18, no. 6, 2018, doi: 10.3390/s18061875.

[76] F. Liu, J. Wang, J. Zhang, and H. Han, “An indoor localization method for pedestrians base on combined UWB/PDR/floor map,” *Sensors (Switzerland)*, vol. 19, no. 11, 2019, doi: 10.3390/s19112578.

[77] A. Alarifi *et al.*, “Ultra wideband indoor positioning technologies: Analysis and recent advances,” *Sensors (Switzerland)*, vol. 16, no. 5. 2016, doi: 10.3390/s16050707.

[78] S. Monica and F. Bergenti, “Hybrid indoor localization using WiFi and UWB technologies,” *Electron.*, vol. 8, no. 3, 2019, doi: 10.3390/electronics8030334.

[79] B. Großwindhager, M. Stocker, M. Rath, C. A. Boano, and K. Römer, “Snaploc: An ultra-fast UWB-based indoor localization system for an unlimited number of tags,” 2019, doi: 10.1145/3302506.3310389.

[80] A. Poulose, O. S. Eyobu, M. Kim, and D. S. Han, “Localization Error Analysis of Indoor Positioning System Based on UWB Measurements,” in *International Conference on Ubiquitous and Future Networks, ICUFN*, 2019, vol. 2019-July, doi: 10.1109/ICUFN.2019.8806041.

[81] R. Tesoriero, R. Tebar, J. A. Gallud, M. D. Lozano, and V. M. R. Penichet, “Improving location awareness in indoor spaces using RFID technology,” *Expert*

Syst. Appl., vol. 37, no. 1, 2010, doi: 10.1016/j.eswa.2009.05.062.

[82] R. Tesoriero, J. A. Gallud, M. Lozano, and V. M. R. Penichet, "Using active and passive RFID technology to support indoor location-aware systems," *IEEE Trans. Consum. Electron.*, vol. 54, no. 2, 2008, doi: 10.1109/TCE.2008.4560133.

[83] "Session details: RFID: tagging the world," *Commun. ACM*, vol. 48, no. 9, 2005, doi: 10.1145/3263688.

[84] G. Deak, K. Curran, and J. Condell, "A survey of active and passive indoor localisation systems," *Computer Communications*, vol. 35, no. 16. 2012, doi: 10.1016/j.comcom.2012.06.004.

[85] Z. Dian, L. Kezhong, and M. Rui, "A precise RFID indoor localization system with sensor network assistance," *China Commun.*, vol. 12, no. 4, 2015, doi: 10.1109/CC.2015.7114062.

[86] A. Bekkali, H. Sanson, and M. Matsumoto, "RFID indoor positioning based on probabilistic RFID map and Kalman Filtering," 2007, doi: 10.1109/WIMOB.2007.4390815.

[87] I. Subri and A. T., "Performance Comparison of Three Types of Sensor Matrices for Indoor Multi-Robot Localization," *Int. J. Comput. Appl.*, vol. 181, no. 26, 2018, doi: 10.5120/ijca2018918103.

[88] J. Luo, L. Fan, and H. Li, "Indoor Positioning Systems Based on Visible

Light Communication: State of the Art," *IEEE Communications Surveys and Tutorials*, vol. 19, no. 4. 2017, doi: 10.1109/COMST.2017.2743228.

[89] R. F. Brena, J. P. García-Vázquez, C. E. Galván-Tejada, D. Muñoz-Rodriguez, C. Vargas-Rosales, and J. Fangmeyer, "Evolution of Indoor Positioning Technologies: A Survey," *Journal of Sensors*, vol. 2017. 2017, doi: 10.1155/2017/2630413.

[90] D. H. Stojanović and N. M. Stojanović, "Indoor Localization and Tracking: Methods, Technologies and Research Challenges," *Facta Univ. Ser. Autom. Control Robot.*, vol. 13, no. Iii 43007, 2014.

[91] G. Oguntala, R. Abd-Alhameed, S. Jones, J. Noras, M. Patwary, and J. Rodriguez, "Indoor location identification technologies for real-time IoT-based applications: An inclusive survey," *Computer Science Review*, vol. 30. 2018, doi: 10.1016/j.cosrev.2018.09.001.

[92] G. Mao and B. Fidan, *Localization algorithms and strategies for wireless sensor networks*. 2009.

[93] M. B. Kjærgaard, H. Blunck, T. Godsk, T. Toftkjær, D. L. Christensen, and K. Grønbæk, "Indoor positioning using GPS revisited," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2010, vol. 6030 LNCS, doi: 10.1007/978-3-642-12654-3_3.

[94] N. Patwari and A. O. Hero, “Using Proximity and Quantized RSS for Sensor Localization in Wireless Networks,” 2003, doi: 10.1145/941350.941354.

[95] A. Küpper, *Location-Based Services: Fundamentals and Operation*. 2005.

[96] J. Hightower and G. Borriello, “Location Sensing Techniques,” 2001.

[97] H. Obeidat, W. Shuaieb, O. Obeidat, and R. Abd-Alhameed, *A Review of Indoor Localization Techniques and Wireless Technologies*, vol. 119, no. 1. Springer US, 2021.

[98] X. Li, K. Pahlavan, and J. Beneat, “Performance of toa estimation techniques in indoor multipath channels,” in *IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, PIMRC*, 2002, vol. 2, doi: 10.1109/PIMRC.2002.1047354.

[99] Y. Liu and Z. Yang, *Location, localization, and localizability: Location-awareness technology for wireless networks*. 2011.

[100] B. Viel and M. Asplund, “Why is fingerprint-based indoor localization still so hard?,” 2014, doi: 10.1109/PerComW.2014.6815247.

[101] L. Lin, H. C. So, and Y. T. Chan, “Accurate and simple source localization using differential received signal strength,” *Digit. Signal Process. A Rev. J.*, vol. 23, no. 3, 2013, doi: 10.1016/j.dsp.2012.12.020.

[102] N. Chang, R. Rashidzadeh, and M. Ahmadi, “Robust indoor positioning using differential Wi-Fi access points,” *IEEE Trans. Consum. Electron.*, vol. 56, no. 3, 2010, doi: 10.1109/TCE.2010.5606338.

[103] S. M. M. Dehghan, H. Moradi, and S. A. A. Shahidian, “Optimal path planning for DRSSI based localization of an RF source by multiple UAVs,” 2014, doi: 10.1109/ICRoM.2014.6990961.

[104] H. Bao and W. C. Wong, “A novel map-based dead-reckoning algorithm for indoor localization,” *J. Sens. Actuator Networks*, vol. 3, no. 1, 2014, doi: 10.3390/jsan3010044.

[105] N. Kothari, B. Kannan, E. D. Glasgwow, and M. B. Dias, “Robust indoor localization on a commercial smart phone,” in *Procedia Computer Science*, 2012, vol. 10, doi: 10.1016/j.procs.2012.06.158.

[106] S. Park and S. Hashimoto, “Indoor localization for autonomous mobile robot based on passive RFID,” 2009, doi: 10.1109/ROBIO.2009.4913284.

[107] H. Bao and W. C. Wong, “An indoor dead-reckoning algorithm with map matching,” 2013, doi: 10.1109/IWCMC.2013.6583784.

[108] H. Zou, Z. Chen, H. Jiang, L. Xie, and C. Spanos, “Accurate indoor localization and tracking using mobile phone inertial sensors, WiFi and iBeacon,” 2017, doi: 10.1109/ISISS.2017.7935650.

[109] S. A. R. Zekavat and M. Buehrer, *Handbook of Position Location: Theory, Practice, and Advances*. 2011.

[110] N. Patwari, J. N. Ash, S. Kyperountas, A. O. Hero, R. L. Moses, and N. S. Correal, "Locating the nodes: Cooperative localization in wireless sensor networks," *IEEE Signal Process. Mag.*, vol. 22, no. 4, 2005, doi: 10.1109/MSP.2005.1458287.

[111] H. C. Chen, T. H. Lin, H. T. Kung, C. K. Lin, and Y. Gwon, "Determining RF angle of arrival using COTS antenna arrays: A field evaluation," 2012, doi: 10.1109/MILCOM.2012.6415851.

[112] A. Savvides, C. C. Han, and M. B. Strivastava, "Dynamic fine-grained localization in ad-hoc networks of sensors," 2001, doi: 10.1145/381677.381693.

[113] H. Chen, J. Hu, H. Tian, S. Li, J. Liu, and M. Suzuki, "A Low-Complexity GA-WSF Algorithm for Narrow-Band DOA Estimation," *Int. J. Antennas Propag.*, vol. 2018, 2018, doi: 10.1155/2018/7175653.

[114] J. N. Ash and R. L. Moses, "Acoustic time delay estimation and sensor network self-localization: Experimental results," *J. Acoust. Soc. Am.*, vol. 118, no. 2, 2005, doi: 10.1121/1.1953307.

[115] C. Sertatil, M. A. Altinkaya, and K. Raoof, "A novel acoustic indoor localization system employing CDMA," *Digit. Signal Process. A Rev. J.*, vol. 22, no. 3, 2012, doi: 10.1016/j.dsp.2011.12.001.

[116] S. Wang, J. Min, and B. Yi, “Location Based Services for Mobiles :Technologies and Standards,” *Proceedings of IEEE ICC 2008, (Beijing, China)*. 2008.

[117] S. Go, S. Kim, and J. W. Chong, “An efficient non-line-of-sight error mitigation method for TOA measurement in indoor environments,” 2014, doi: 10.1145/2557977.2558011.

[118] D. H. Shin and T. K. Sung, “Comparisons of error characteristics between TOA and TDOA positioning,” *IEEE Transactions on Aerospace and Electronic Systems*, vol. 38, no. 1. 2002, doi: 10.1109/7.993253.

[119] I. Güvenç and C. C. Chong, “A survey on TOA based wireless localization and NLOS mitigation techniques,” *IEEE Commun. Surv. Tutorials*, vol. 11, no. 3, 2009, doi: 10.1109/SURV.2009.090308.

[120] L. Cong and W. Zhuang, “Non-line-of-sight error mitigation in TDOA mobile location,” in *Conference Record / IEEE Global Telecommunications Conference*, 2001, vol. 1, doi: 10.1109/glocom.2001.965202.

[121] L. Mailaender, “Comparing geo-location bounds for TOA, TDOA and round-trip TOA,” 2007, doi: 10.1109/PIMRC.2007.4393993.

[122] D. Dardari, A. Conti, U. Ferner, A. Giorgetti, and M. Z. Win, “Ranging with ultrawide bandwidth signals in multipath environments,” *Proc. IEEE*, vol. 97, no. 2, 2009, doi: 10.1109/JPROC.2008.2008846.

[123] M. van Gerven and S. Bohte, “Editorial: Artificial neural networks as models of neural information processing,” *Frontiers in Computational Neuroscience*, vol. 11. 2017, doi: 10.3389/fncom.2017.00114.

[124] W. S. McCulloch and W. Pitts, “A logical calculus of the ideas immanent in nervous activity,” *Bull. Math. Biophys.*, vol. 5, no. 4, 1943, doi: 10.1007/BF02478259.

[125] Y. LeCun, Y. Bengio, and G. Hinton, “Deep learning,” *Nature*, vol. 521, no. 7553, pp. 436–444, 2015, doi: 10.1038/nature14539.

[126] S. Lawrence, C. L. Giles, A. C. Tsoi, and A. D. Back, “Face recognition: A convolutional neural-network approach,” *IEEE Trans. Neural Networks*, vol. 8, no. 1, 1997, doi: 10.1109/72.554195.

[127] J. Long, E. Shelhamer, and T. Darrell, “Fully convolutional networks for semantic segmentation,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June-2015, doi: 10.1109/CVPR.2015.7298965.

[128] J. Donahue *et al.*, “Long-Term Recurrent Convolutional Networks for Visual Recognition and Description,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 4, 2017, doi: 10.1109/TPAMI.2016.2599174.

[129] X. Wu, R. He, and Z. Sun, “A Lightened CNN for Deep Face Representation,” *arXiv*, 2015.

[130] A. Diba, V. Sharma, A. Pazandeh, H. Pirsiavash, and L. Van Gool, “Weakly supervised cascaded convolutional networks,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January, doi: 10.1109/CVPR.2017.545.

[131] W. Ouyang *et al.*, “DeepID-Net: Object Detection with Deformable Part Based Convolutional Neural Networks,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 39, no. 7, 2017, doi: 10.1109/TPAMI.2016.2587642.

[132] G. Cybenko and G. Cybenkot, “Approximation by superpositions of a sigmoidal function. Math Cont Sig Syst (MCSS) 2:303-314 Mathematics of Control, Signals, and Systems Approximation by Superpositions of a Sigmoidal Function*,” *Math. Control Signals Syst.*, vol. 2, 1989.

[133] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural Networks*, vol. 4, no. 2, 1991, doi: 10.1016/0893-6080(91)90009-T.

[134] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang, “The expressive power of neural networks: A view from the width,” in *Advances in Neural Information Processing Systems*, 2017, vol. 2017-December.

[135] B. Hanin, “Universal function approximation by deep neural nets with bounded width and ReLU activations,” *Mathematics*, vol. 7, no. 10, 2019, doi: 10.3390/MATH7100992.

[136] S. Sengupta *et al.*, “A review of deep learning with special emphasis on

architectures, applications and recent trends," *Knowledge-Based Syst.*, vol. 194, 2020, doi: 10.1016/j.knosys.2020.105596.

[137] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Networks*, vol. 61. 2015, doi: 10.1016/j.neunet.2014.09.003.

[138] G. Marcus, "Deep Learning: A Critical Appraisal," pp. 1–27, 2018, [Online]. Available: http://arxiv.org/abs/1801.00631.

[139] N. Papernot, P. Mcdaniel, S. Jha, M. Fredrikson, Z. B. Celik, and A. Swami, "The limitations of deep learning in adversarial settings," 2016, doi: 10.1109/EuroSP.2016.36.

[140] E. Abbe and C. Sandon, *Provable limitations of deep learning*. 2018.

[141] D. Mazza and M. Pagani, "Automatic differentiation in PyTorch," *Proc. ACM Program. Lang.*, vol. 5, no. POPL, pp. 1–4, 2021, doi: 10.1145/3434309.

[142] M. Abadi *et al.*, "TensorFlow: A system for large-scale machine learning," 2016.

[143] Y. Jia *et al.*, "Caffe: Convolutional architecture for fast feature embedding," 2014, doi: 10.1145/2647868.2654889.

[144] S. Tokui, K. Oono, S. Hido, and J. Clayton, "Chainer: a Next-Generation Open Source Framework for Deep Learning," 2015, [Online]. Available:

http://learningsys.org/papers/LearningSys_2015_paper_33.pdf.

[145] F. Chollet, “Keras.” 2015, [Online]. Available: https://github.com/keras-team/keras.

[146] J. J. Dai *et al.*, “BigDL: A Distributed Deep Learning Framework for Big Data,” 2019, doi: 10.1145/3357223.3362707.

[147] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proc. IEEE*, vol. 86, no. 11, 1998, doi: 10.1109/5.726791.

[148] S. Kombrink, T. Mikolov, M. Karafiát, and L. Burget, “Recurrent neural network based language modeling in meeting recognition,” 2011.

[149] A. Krizhevsky, I. Sutskever, and G. E. Hinton, “ImageNet classification with deep convolutional neural networks,” in *Advances in Neural Information Processing Systems*, 2012, vol. 2.

[150] G. LeCun, Y., Bengio, Y., Hinton, “Deep learning. nature 521 (7553): 436,” *Nature*, vol. 521, 2015.

[151] C. Francois, “Deep learning with Python,” *Scotts Val. Creat. Indep. Publ. Platf.*, 2017.

[152] D. Tomè, F. Monti, L. Baroffio, L. Bondi, M. Tagliasacchi, and S. Tubaro,

"Deep Convolutional Neural Networks for pedestrian detection," *Signal Process. Image Commun.*, vol. 47, 2016, doi: 10.1016/j.image.2016.05.007.

[153] Z. Q. Zhao, P. Zheng, S. T. Xu, and X. Wu, "Object Detection with Deep Learning: A Review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11. 2019, doi: 10.1109/TNNLS.2018.2876865.

[154] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," 2014, doi: 10.1109/CVPR.2014.81.

[155] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, vol. 2017-October, doi: 10.1109/ICCV.2017.322.

[156] Y. Wang and Z. Wang, "A survey of recent work on fine-grained image classification techniques," *J. Vis. Commun. Image Represent.*, vol. 59, 2019, doi: 10.1016/j.jvcir.2018.12.049.

[157] M. D. Zeiler, D. Krishnan, G. W. Taylor, and R. Fergus, "Deconvolutional networks," in *2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2010, pp. 2528–2535, doi: 10.1109/CVPR.2010.5539957.

[158] H. Noh, S. Hong, and B. Han, "Learning Deconvolution Network for Semantic Segmentation," in *2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1520–1528, doi: 10.1109/ICCV.2015.178.

[159] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, “You only look once: Unified, real-time object detection,” in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.91.

[160] S. Fidler, R. Mottaghi, A. Yuille, and R. Urtasun, “Bottom-up segmentation for top-down detection,” 2013, doi: 10.1109/CVPR.2013.423.

[161] R. Girshick, “Fast R-CNN,” in *Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV)*, 2015, pp. 1440–1448, doi: 10.1109/ICCV.2015.169.

[162] S. Ren, K. He, R. Girshick, and J. Sun, “Faster R-CNN: Towards real-time object detection with region proposal networks,” in *Advances in Neural Information Processing Systems*, 2015, vol. 2015-January.

[163] W. Liu *et al.*, “SSD: Single shot multibox detector,” in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2016, vol. 9905 LNCS, doi: 10.1007/978-3-319-46448-0_2.

[164] D. Erhan, C. Szegedy, A. Toshev, and D. Anguelov, “Scalable object detection using deep neural networks,” 2014, doi: 10.1109/CVPR.2014.276.

[165] J. Redmon and A. Farhadi, “YOLO9000: Better, faster, stronger,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern*

Recognition, CVPR 2017, 2017, vol. 2017-January, doi: 10.1109/CVPR.2017.690.

[166] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *CoRR*, vol. abs/1804.0, 2018, [Online]. Available: http://arxiv.org/abs/1804.02767.

[167] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2015, vol. 07-12-June-2015, doi: 10.1109/CVPR.2015.7298594.

[168] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2015.

[169] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.90.

[170] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, 2014, vol. 8689 LNCS, no. PART 1, doi: 10.1007/978-3-319-10590-1_53.

[171] A. Sherstinsky, "Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network," *Phys. D Nonlinear Phenom.*, vol. 404, 2020, doi: 10.1016/j.physd.2019.132306.

[172] S. Hochreiter and J. Schmidhuber, “Long Short-Term Memory,” *Neural Comput.*, vol. 9, no. 8, pp. 1735–1780, Nov. 1997, doi: 10.1162/neco.1997.9.8.1735.

[173] F. A. Gers and J. Schmidhuber, “Recurrent nets that time and count,” in *Proceedings of the International Joint Conference on Neural Networks*, 2000, vol. 3, doi: 10.1109/ijcnn.2000.861302.

[174] J. Chung, C. Gulcehre, K. Cho, and Y. Bengio, “Empirical evaluation of gated recurrent neural networks on sequence modeling,” *arXiv Prepr. arXiv1412.3555*, 2014.

[175] H. Sak, A. Senior, and F. Beaufays, “Long short-term memory recurrent neural network architectures for large scale acoustic modeling,” 2014.

[176] P. Doetsch, M. Kozielski, and H. Ney, “Fast and Robust Training of Recurrent Neural Networks for Offline Handwriting Recognition,” in *Proceedings of International Conference on Frontiers in Handwriting Recognition, ICFHR*, 2014, vol. 2014-December, doi: 10.1109/ICFHR.2014.54.

[177] H. Palangi *et al.*, “Deep Sentence embedding using long short-term memory networks: Analysis and application to information retrieval,” *IEEE/ACM Trans. Audio Speech Lang. Process.*, vol. 24, no. 4, 2016, doi: 10.1109/TASLP.2016.2520371.

[178] M. Pota, F. Marulli, M. Esposito, G. De Pietro, and H. Fujita, “Multilingual

POS tagging by a composite deep architecture based on character-level features and on-the-fly enriched Word Embeddings," *Knowledge-Based Syst.*, vol. 164, 2019, doi: 10.1016/j.knosys.2018.11.003.

[179] P. Gao, Q. Zhang, F. Wang, L. Xiao, H. Fujita, and Y. Zhang, "Learning reinforced attentional representation for end-to-end visual tracking," *Inf. Sci. (Ny).*, vol. 517, 2020, doi: 10.1016/j.ins.2019.12.084.

[180] L. Sifre, *PhD thesis Rigid-Motion Scattering For Image Classification*. 2014.

[181] J. Jin, A. Dundar, and E. Culurciello, "Flattened convolutional neural networks for feedforward acceleration," 2015.

[182] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna, "Rethinking the Inception Architecture for Computer Vision," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, vol. 2016-December, doi: 10.1109/CVPR.2016.308.

[183] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *32nd International Conference on Machine Learning, ICML 2015*, 2015, vol. 1.

[184] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," *COURSERA Neural networks Mach. Learn.*, vol. 4, no. 2, 2012.

[185] J. Huang *et al.*, “Speed/accuracy trade-offs for modern convolutional object detectors,” in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, 2017, vol. 2017-January, doi: 10.1109/CVPR.2017.351.

[186] S. Cass, “Taking AI to the edge: Google’s TPU now comes in a maker-friendly package,” *IEEE Spectrum*, vol. 56, no. 5. 2019, doi: 10.1109/MSPEC.2019.8701189.

[187] A. Allan, “Hands on with the Coral Dev Board,” 2019. https://aallan.medium.com/hands-on-with-the-coral-dev-board-adbcc317b6af.

[188] “Introduction to autonomous mobile robots,” *Choice Rev. Online*, vol. 49, no. 03, 2011, doi: 10.5860/choice.49-1492.