TRAJECTORY PLOTTING FOR INDOOR ENVIRONMENTS

A PROJECT REPORT

Submitted by

ARJUN S	**203001018**
RAHUL M	**203001076**
SHYAM R	**203001096**

In partial fulfillment for the award of the degree of

BACHELOR OF ENGINEERING

IN

ELECTRICAL AND ELECTRONICS ENGINEERING

Department of

Electrical and Electronics Engineering

Sri Sivasubramaniya Nadar College of Engineering

(An autonomous institution, affiliated to Anna University)

Rajiv Gandhi Salai (OMR), Kalavakkam-603 110

November 2023

Sri Sivasubramaniya Nadar College of Engineering

(An Autonomous Institution, Affiliated to Anna University)

BONAFIDE CERTIFICATE

Certified that this report titled "**TRAJECTORY PLOTTING FOR INDOOR ENVIRONMENTS**" is the bonafide work of "**S ARJUN** (203001018)**, RAHUL MANIKANDAN** (203001076)**, SHYAM R** (203001096)" who carried out the work under my supervision.

Certified further that to the best of my knowledge, the work reported herein does not form part of any other thesis or dissertation on the basis of which a degree or award was conferred on an earlier occasion on this or any other candidate.

SIGNATURE

Dr. V. RAJINI

HEAD OF THE DEPARTMENT

Department of Electrical and Electronics

SSN College Of Engineering

Kalavakam – 603 110

SIGNATURE

Dr. R. JAYAPARVATHY

SUPERVISOR

Department of Electrical and Electronics

SSN college of engineering

Kalavakam – 603 110

Submitted for project viva-voice examination held on ……

EXTERNAL EXAMINER

INTERNAL EXAMINER

ABSTRACT

Indoor positioning has been a long-awaited technology, enabling mobile location-aware services, augmented reality applications, and location history collection. Since the COVID epidemic, increased focus has been placed on location history, particularly for contact tracking. The location history further enables space utilization studies for indoor space planning or traffic flow analysis for transportation optimization. Indoor positioning systems utilize multi-modal sensor data such as global positioning system (GPS), inertial measurement unit (IMU), and building-dependent infrastructures such as WiFi, Bluetooth, or magnetic fingerprints. WiFi map is the most informative, which holds pairs of access-point IDs (i.e., MAC address) and global positions (i.e., latitude/longitude degrees). The quality of WiFi maps directly influences the accuracy of indoor positioning systems. We propose an integrated hardware device setup that plots the trajectory of the path travelled by a moving body onto the floorplan of the confined space which provides an extensive solution for accurate indoor positioning and path tracking by using WIFI, IMU sensor (gyroscope and accelerometer) fusion with the floorplan data. A state-of-the-art inertial navigation algorithm- RONIN LSTM that uses a Kalman filter and extended Kalman filter is implemented to estimate a relative motion trajectory from the IMU sensor data. We propose a convolutional neural network to refine the location history to be consistent with the floorplan. Our model takes indoor positioning technology to the next level, specifically focusing on location history estimation as an offline task. Our system works with any modern smartphone, while requiring minimal extra energy consumption on the device, where energy-efficient IMU is the only additional data acquisition. We have built a new benchmark with WiFi, IMU, and floorplan data at our department building (training sites), spanning 6+ hours and 3-4 kilometers. Dense ground-truth positions are obtained by manually

geo-localizing relative motion trajectories from a visual-inertial SLAM algorithm. For long sequences through shopping malls in the wild (i.e., as long as 45 minutes), the ground-truth positions are obtained for a sparse set of points by manual specifications. Our qualitative and quantitative evaluations demonstrate that the proposed system is able to produce twice as accurate and a few orders of magnitude denser (i.e., 50Hz instead of 1/60Hz) location history than the currently available system.

ACKNOWLEDGEMENT

We would like to thank our college management for the support provided for the successful completion of our project. We convey our gratitude to **Dr. Kala Vijayakumar**, President, SSN Institutions, and **Dr. V. E. Annamalai**, Principal, Sri Sivasubramaniya Nadar College of Engineering, for providing us the opportunity and facilities to work on our project and complete it.
We sincerely thank Dr. V. Rajini, Professor and Head of the Department of Electrical and Electronics Engineering, for her persistent encouragement and support to work on this project.

We express our deepest gratitude to our guide, **Dr. R. Jayaparvathy**, Professor, Department of Electrical and Electronics Engineering for guiding us with care and attention. She has made a great effort to help us through the course of our project work and has provided us with the necessary guidelines and corrections wherever and whenever required. Without her valuable efforts, this project would not have been completed the way it is now.
We finally thank our families and fellow classmates; without their support and cooperation, this project would not have been a reality.

ARJUN S

RAHUL M

SHYAM R

TABLE OF CONTENTS

List of Tables

List of Figures

List of Abbreviations

API: Application Programming Interface

AR: Augmented Reality

BS: Base Station

CNN: Convolution Neural Network

CPU: Control Processing Unit

CUDA: Compute Unified Device Architecture

DHL: Dense History of Locations

EKF: Extended Kalman Filter

FLP: Fused Location Provider

GB: Gigabytes

GPS: Global Positioning System

GPU: Graphic Processing Unit

Gyro: Gyroscope

Hz: Hertz

ID: Identity Document

IMU: Inertial Measurement Unit

IoT: Internet of Things

IPS: Indoor Positioning System

Khz: Kilohertz

LSTM: Long Short-Term Memory

MCU: Microcontroller Unit

MEMS: Micro-electromechanical system

RAM: Random Access Memory

RNN: Recurrent Neural Network

RSSI: Received Signal Strength Interface

Ronin: Robust Neural Network

SLAM: Simultaneous Localization and Mapping

TL: Transfer Learning

UWB: Ultra Wide Band

UKF: Unscented Kalman Filter

Wifi: Wireless Fidelity

g: Gravity

Acc: Accelerometer

deg/s: Degree/Second

CHAPTER-1

INTRODUCTION

1.1 OVERVIEW

Recently, indoor positioning has attracted a lot of attention from research as well as industry communities. This is partly driven by the increasing market demand for indoor location-based services. Due to the limitation of GPS signal strength indoors, nearby anchors with known positions are generally needed for an indoor positioning system. In general, there are two components in indoor positioning systems. One is nearby anchors with the knowledge of their own location information, while the other is a positioning device with the objective of identifying its location by processing wireless signals through nearby anchors. Although the potential needs of indoor navigation services have been already addressed from industry aspects, the ultimate solution in terms of accuracy, reliability, real-time, and low cost remains open.

There are several factors that make the design of indoor positioning systems challenging:

- Cost: The need to deploy nearby anchors and the development of localization devices to identify objects are costly.
- Numbers of nearby anchors: Unlike the traditional navigation GPS system, which uses 31 active satellites in orbit, the indoor environment and space sometimes limit the number of anchors.
- Complicated indoor environment: The wireless signal used to measure the distance and angle indoors usually suffers significantly from obstacles, such as walls, objects, and/or human beings, which lead to multi-path effects. Due to its wide deployment, we expect WiFi to become a prominent tool for indoor positioning. In this article, we use WiFi access points (APs) with multiple

antennas as nearby anchors, while the positioning device could be any mobile platform with WiFi capability such as smartphones or tablets.

1.2 NEED FOR STUDY

- Indoor location tracking can enhance navigation within large buildings such as airports, shopping malls, and hospitals. In emergency situations, knowing the precise location of individuals inside a building can be crucial for first responders. Indoor location tracking can aid in the rapid and accurate evacuation of people during emergencies**.**
- Indoor location tracking is a key component of smart building technologies. It enables the efficient use of resources such as heating**,** lighting, and cooling based on the occupancy of specific areas contributing to energy conservation and sustainability.
- In healthcare settings, indoor location tracking can be used to monitor the movement of patients and medical staff to improve patient care, streamline workflows, and enhance overall hospital management**.**

1.3 OBJECTIVES OF STUDY

The objectives of the proposed work are:

1. To enhance the performance of the indoor positioning technology in terms of accuracy and dense location history using IMU sensor fusion and novel navigation algorithms.
2. To implement a **convolutional neural network** to refine the location history to be consistent with a floorplan.
3. To enable the system to work with any modern smartphone while requiring minimal extra energy consumption on the device, where energy-efficient IMU is the only additional data acquisition.

CHAPTER 2

LITERATURE SURVEY

Sachini, Herath et al. (2021). Fusion-DHL: WiFi, IMU, and Floorplan Fusion for Dense History of Locations in Indoor Environments.

The proposed algorithm employs three key components, an inertial navigation algorithm that estimates relative motion trajectories based on IMU sensor data, utilization of a WiFi-based localization API to acquire positional constraints and geolocate the trajectory leveraging industry-standard methods, and the application of a convolutional neural network (CNN) to refine and align the location history with the given floorplan. To validate the efficacy of the approach, a custom data acquisition app was developed to construct a comprehensive dataset encompassing WiFi, IMU, and floorplan data, augmented with ground-truth positions. This dataset was gathered across four university buildings and three shopping malls. The evaluation of the proposed system involved both qualitative and quantitative assessments, demonstrating a significant improvement in accuracy—approximately twice as precise—while achieving orders of magnitude denser location history compared to existing standards. Importantly, these advancements were achieved with minimal additional energy consumption.

Zhao, X., et al. (2023). Feature Consistency Constraints-Based CNN for Landsat Land Cover Mapping. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 16, pp. 3545-3554.

The paper explores the effectiveness of Convolutional Neural Networks (CNNs) in image processing, acknowledging their reliance on large-scale, high-quality training samples as a limiting factor for broader application. To address the challenge of obtaining such samples, particularly in remote sensing imagery where manual annotations are expensive and impractical, this study proposes leveraging existing land cover maps as an alternative, despite their inherent noisy labels. To mitigate the impact of this noise, the paper introduces a solution focused on enhancing the consistency feature learning capability of CNNs for practical land cover mapping. The approach involves two key strategies: firstly, incorporating an intraclass feature consistency constraint to maintain consistency within CNN feature maps for the same class, and secondly, utilizing an inter-iteration feature consistency constraint to guide the network in learning features consistent with the broader distribution within a mini-batch.

Daquan, Feng., Wang, Chunqi., Chunlong, He., Yuan, Zhuang., Xiang-Gen, Xia. (2020). Kalman-Filter-Based Integration of IMU and UWB for High-Accuracy Indoor Positioning and Navigation. IEEE Internet of Things Journal, 7(4)

The paper proposes an integrated indoor positioning system (IPS) that combines inertial measurement unit (IMU) and ultrawideband (UWB) technologies using extended Kalman filter (EKF) and unscented Kalman filter (UKF) to improve accuracy and robustness. The simulation and experimental results demonstrate that the proposed IPS significantly improves positioning and navigation accuracy. The algorithm also has high computational efficiency and can be implemented in real-time on embedded devices. The increasing demand for accurate and cost-effective positioning in IoT applications like smart manufacturing and smart homes has led to challenges. Inertial Measurement Units (IMUs) offer fast but increasingly error-prone navigation over time, while Ultrawideband (UWB) systems face uncertainties, especially indoors. This article proposes an Integrated Indoor Positioning System (IPS) that combines IMU and UWB using Kalman filters to enhance accuracy and resilience. It discusses optimizing the placement of base stations (BSs) for better accuracy and presents simulation results indicating IMU's role in reducing UWB

observation errors. The integrated IPS outperforms UWB-based methods, exhibiting improved accuracy and computational efficiency on standard devices. Additionally, it introduces and assesses two motion approximation models, showcasing their robustness and continuous tracking abilities in real environments.

CHAPTER 3

METHODOLOGY

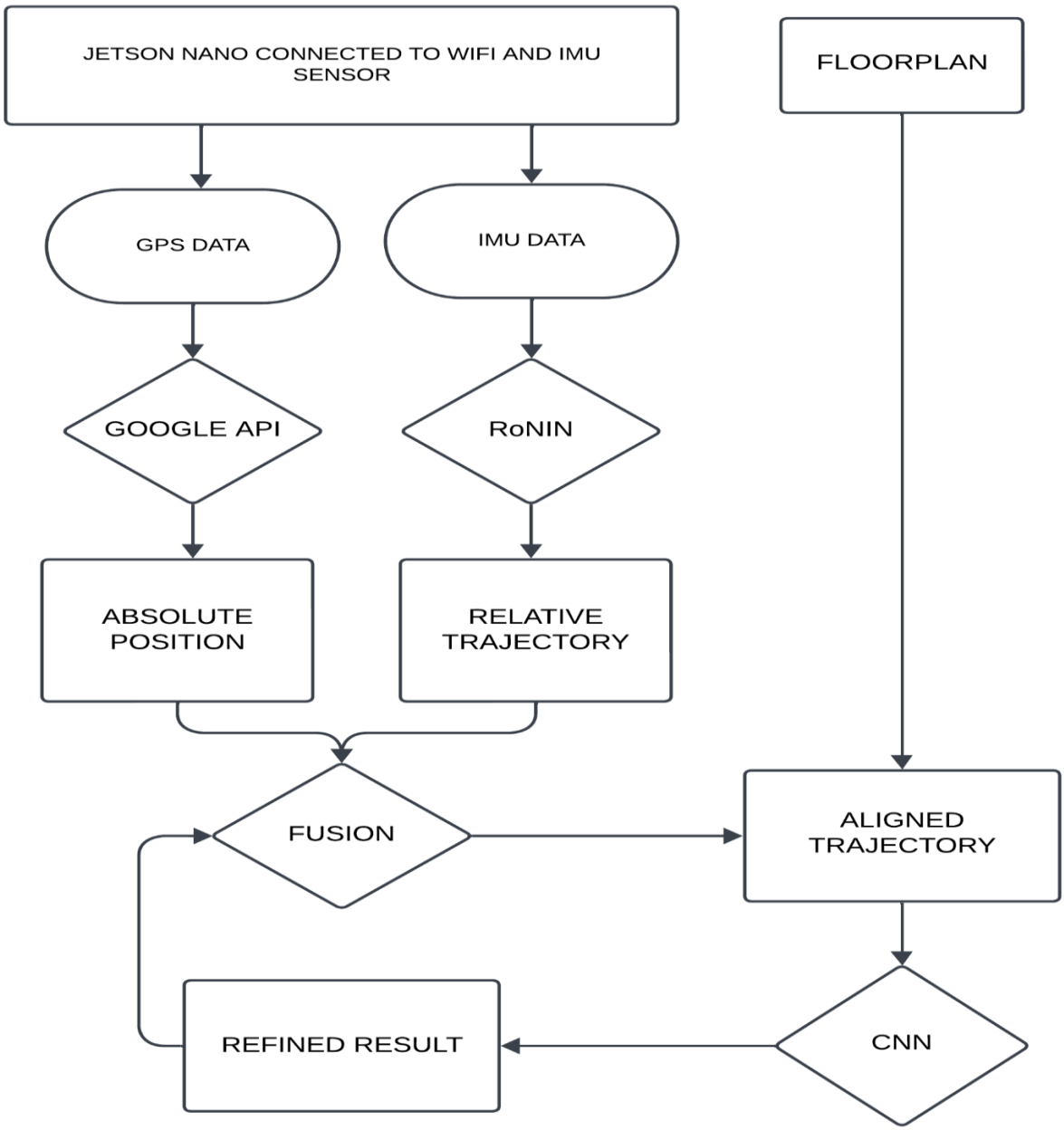


Fig. 3.1 Work Flow Diagram

CHAPTER 4

INERTIAL NAVIGATION

4.1 INERTIAL MEASUREMENT UNIT

An Inertial Measurement Unit (IMU) is a device that can measure and report specific gravity and angular rate of an object to which it is attached. An IMU typically consists of:

- Gyroscopes: providing a measure angular rate
- Accelerometers: providing a measure specific force/acceleration
- Magnetometers (optional): measurement of the magnetic field surrounding the system

IMUs are available in several performance grades. They are divided into one of the four categories based on the specifications of the accelerometer and gyro:

- Consumer/Automotive Grade
- Industrial Grade
- Tactical Grade
- Navigation Grade

These performance categories are typically defined based on the in-run bias stability of the sensor, as the in-run bias stability plays a large role in determining inertial navigation performance.

An accelerometer is the primary sensor responsible for measuring inertial acceleration, or the change in velocity over time, and can be found in a variety of different types, including mechanical accelerometers, quartz accelerometers, and MEMS accelerometers. A MEMS accelerometer is essentially a mass suspended by a spring, as illustrated in Figure 2. The mass is known as the proof mass and the direction that the mass is allowed to move

is known as the sensitivity axis. When an accelerometer is subjected to a linear acceleration along the sensitivity axis, the acceleration causes the proof mass to shift to one side, with the amount of deflection proportional to the acceleration.

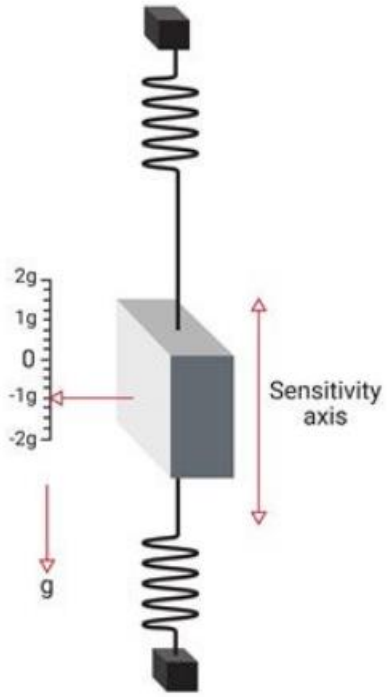


Fig 4.1 MEMS accelerometer model

A gyroscope is an inertial sensor that measure an object's angular rate with respect to an inertial reference frame. There are many different types of gyroscopes available on the market, which range over various levels of performance and include mechanical gyroscopes, fiber-optic gyroscopes (FOGs), ring laser gyroscopes (RLGs), and quartz/MEMS gyroscopes. Quartz and MEMS gyroscopes are typically used in the consumer grade, industrial grade, and tactical grade markets, while fiber-optic gyroscopes span all four of the performance categories. Ring laser gyroscopes typically consist of in-run bias stabilities ranging anywhere from 1 °/hour down to less than 0.001 °/hour, encompassing the tactical and navigation grades. Mechanical gyroscopes make up the

highest performing gyroscopes available on the market and can reach in-run bias stabilities of less than 0.0001 °/hour.

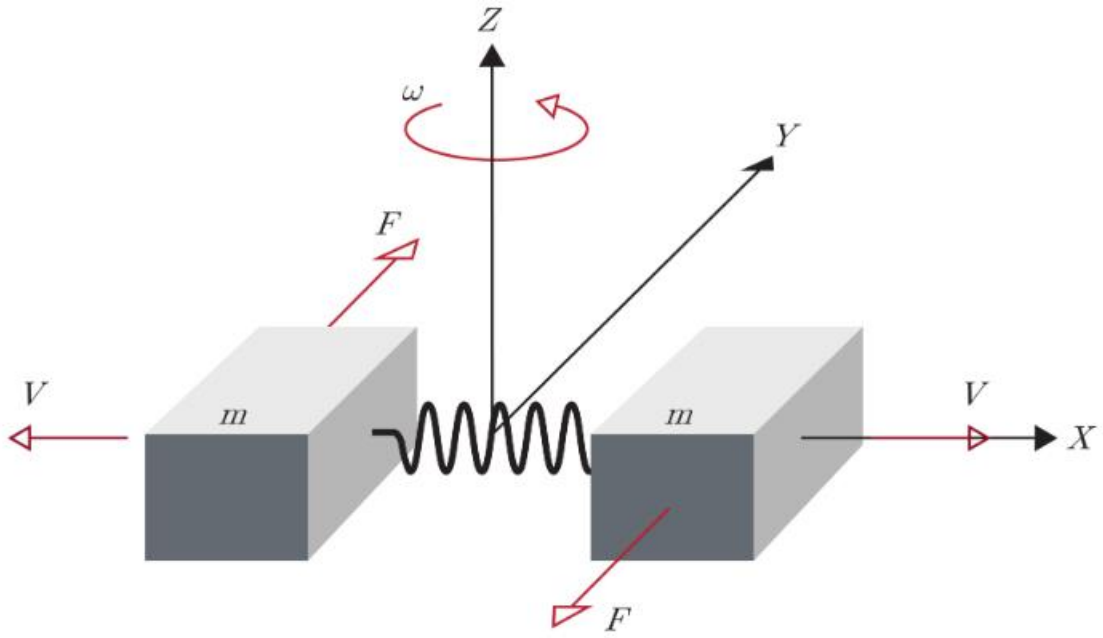


Fig 4.2 MEMS gyroscope model

A magnetometer is a type of sensor that measures the strength and direction of a magnetic field. While there are many different types of magnetometers, most MEMS magnetometers rely on magnetoresistance to measure the surrounding magnetic field. Magnetoresistive magnetometers are made up of permalloys that change resistance due to changes in magnetic fields. Typically, MEMS magnetometers are used to measure a local magnetic field which consists of a combination of Earth's magnetic field as well as any magnetic fields created by nearby objects.

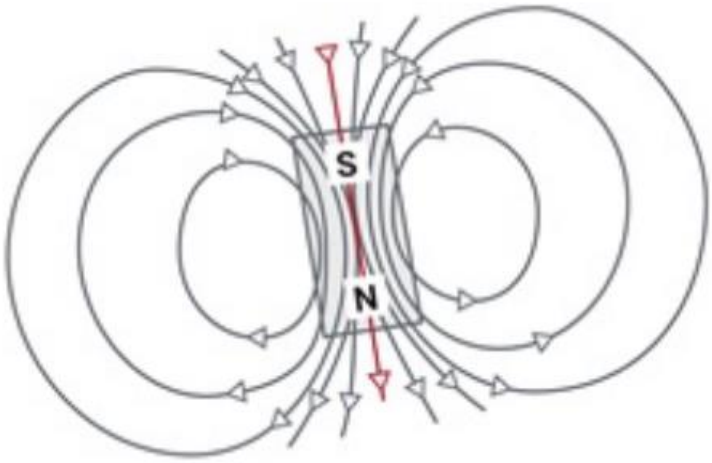


Fig 4.3 MEMS magnetometer model

IMU is energy efficient and works anytime anywhere. Inertial navigation has been a dream technology for many years. Classical algorithms are either physics-based (e.g., IMU double integration), or heuristic-based. With the surge of deep learning, data-driven approaches have made a breakthrough. A state-of-the-art system achieves a positional accuracy of 5 meters per minute under natural activities in the wild. The major error source is the rotational drift due to gyroscope bias. Wireless communication networks (e.g., WiFi [7] or Bluetooth [8]) are the most reliable information for indoor localization. A system requires either 1) an access-point database (i.e., access-point IDs and their geo-coordinates), which allows trilateration or weighted averaging of access-point coordinates; or 2) a fingerprint database (i.e., received signal strength indicators and geo-coordinates at their observations), which allows fingerprint-based geo-coordinate look-ups. Google Fused Location Provider API (FLP) is a widely used system, which we believe is mostly WiFi-based. However, the system is energy-draining for a dense location history and often exhibits positional errors of more than 10 meters. Modern smartphones are equipped with a suite of sensors. WiFi-SLAM minimizes accumulation errors in inertial navigation by

WiFi-based loop-closing constraints. Magnetic field distortions are specific to building infrastructures and provide additional cues for the fingerprint-based system. Map-matching algorithms align motion trajectories with a floorplan via conditional random fields, hidden Markov models, particle filtering, or dynamic programming. The paper proposes a novel fusion of WiFi, IMU, and floorplan data by a combination of optimization and a convolutional neural network (CNN).

Results of ACC and gyro calibration procedures, which primarily use raw outputs of the IMU. Raw outputs need to be converted into raw data that respect the desired units of “g” for acceleration and “deg/s” for angular rates. If this conversion is required, a raw scale factor can be used.

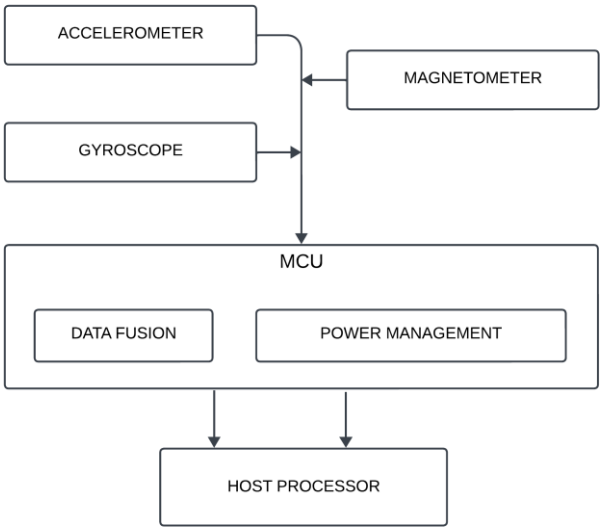


Fig 4.4 IMU Flowchart

4.2 BNO-055 Intelligent 9-axis absolute orientation sensor

- The smart sensor BNO055 is a System-in-package (SiP) solution that integrates a triaxial 14-bit accelerometer, an accurate close-loop triaxial 16-bit gyroscope, and a triaxial geomagnetic sensor.
- This smart sensor is significantly smaller than comparable solutions. By integrating sensors and sensor fusion in a single device, the BNO055 makes integration easy, avoids complex multi vendor solutions, and thus simplifies innovations, e.g. novel applications such as IoT hardware.

The BNO055 is the perfect choice for AR, immersive gaming, personal health and fitness, indoor navigation, and any other application requiring context awareness. It is ideally suited for demanding applications such as augmented reality, navigation, gaming, robotics, or industrial applications.

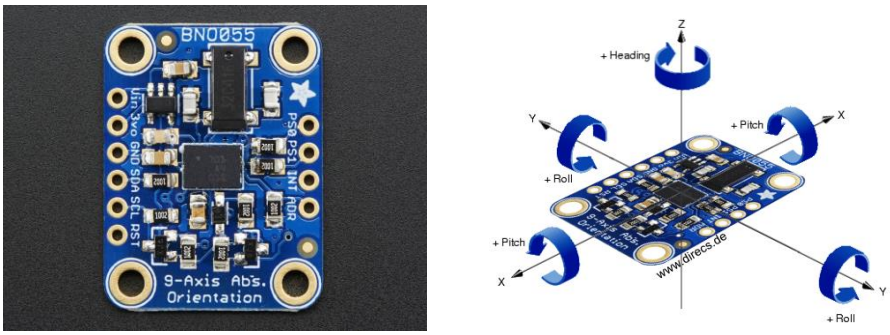


Fig 4.5 BNO 055 Module

Table 4.1 KEY FEATURES OF BNO055

Accelerometer features	
Programmable functionality	Acceleration ranges ±2g/±4g/±8g/±16g Low-pass filter bandwidths 1kHz - <8Hz Operation modes: Normal - Suspend - Low power - Standby - Deep suspend
On-chip interrupt controller	Motion-triggered interrupt-signal generation for - any-motion (slope) detection - slow or no motion recognition - high-g detection

Gyroscope features	
Programmable functionality	Ranges switchable from ±125°/s to ±2000°/s Low-pass filter bandwidths 523Hz - 12Hz Operation modes: - Normal - Fast power up - Deep suspend - Suspend - Advanced power save
On-chip interrupt controller	Motion-triggered interrupt-signal generation for - any-motion (slope) detection - high rate
Magnetometer features	
Programmable functionality	Magnetic field range typical ±1300µT (x-, y-axis); ±2500µT (z-axis) Magnetic field resolution of ~0.3µT Operating modes: - Low power - Regular - Enhanced regular - High Accuracy Power modes: - Normal - Sleep - Suspend - Force

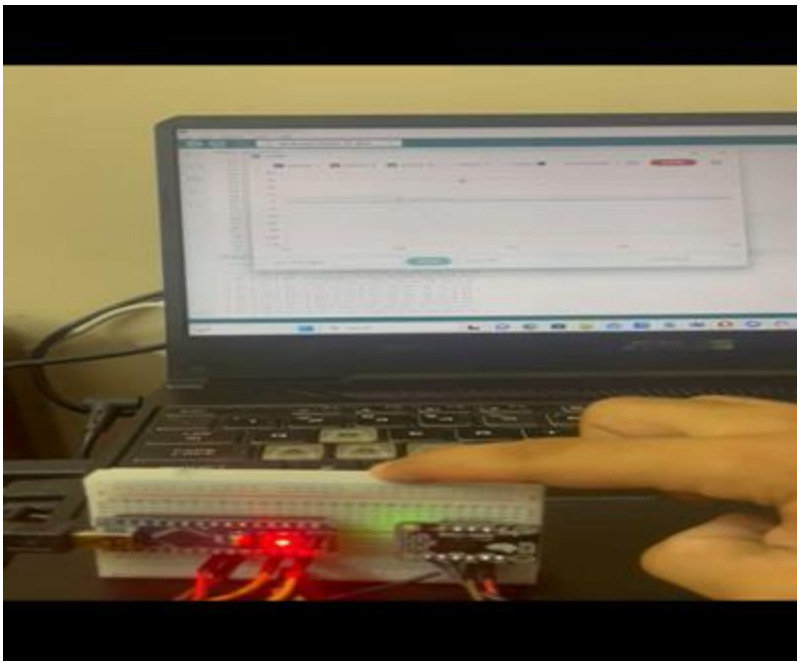


Fig 4.6 IMU actively fetching data

We present a thorough calibration guide accompanied by implementation recommendations. Applied approaches exploit the gravity measurements under static conditions when the ACC triad ought to be calibrated and the referential angle of performed rotation in the case of the gyro triad calibration. The calibration algorithms estimate the ACC and gyro error models, which include scale factor errors, axes misalignment, and offsets. The whole calibration process is supported by publicly available sample codes including raw data and evaluation, which serve the reader as a help in learning about basic principles of low-cost inertial measurement unit calibration.

CHAPTER-5

NVIDIA JETSON NANO

5.1 INTRODUCTION TO JETSON NANO

The NVIDIA Jetson Nano is a compact and powerful single-board computer designed for edge computing and artificial intelligence (AI) applications. Developed by NVIDIA, a leading technology company known for its graphics processing units (GPUs), the Jetson Nano is specifically tailored for developers, hobbyists, and professionals looking to deploy AI at the edge.

Equipped with a quad-core ARM Cortex-A57 CPU and an integrated NVIDIA Maxwell GPU with 128 CUDA cores, the Jetson Nano offers impressive computing capabilities in a small form factor. It also features 4 GB of LPDDR4 RAM, multiple USB ports, GPIO (General Purpose Input/Output) pins, and support for various peripherals, making it versatile for a range of projects.

One of the standout features of the Jetson Nano is its compatibility with popular deep learning frameworks, such as TensorFlow and PyTorch, enabling developers to easily deploy and run machine learning models on the device. This makes it a valuable tool for applications like image and speech recognition, object detection, and other AI-driven tasks in real-time.

Whether used for educational purposes, prototyping, or developing production-ready AI applications, the NVIDIA Jetson Nano provides an accessible platform that combines performance and efficiency, making it a popular choice in the rapidly evolving field of edge computing and artificial intelligence.

5.2 SPECIFICATIONS

1. **GPU**: NVIDIA Maxwell architecture with 128 CUDA cores
2. **CPU**: Quad-core ARM Cortex-A57 @ 1.43 GHz

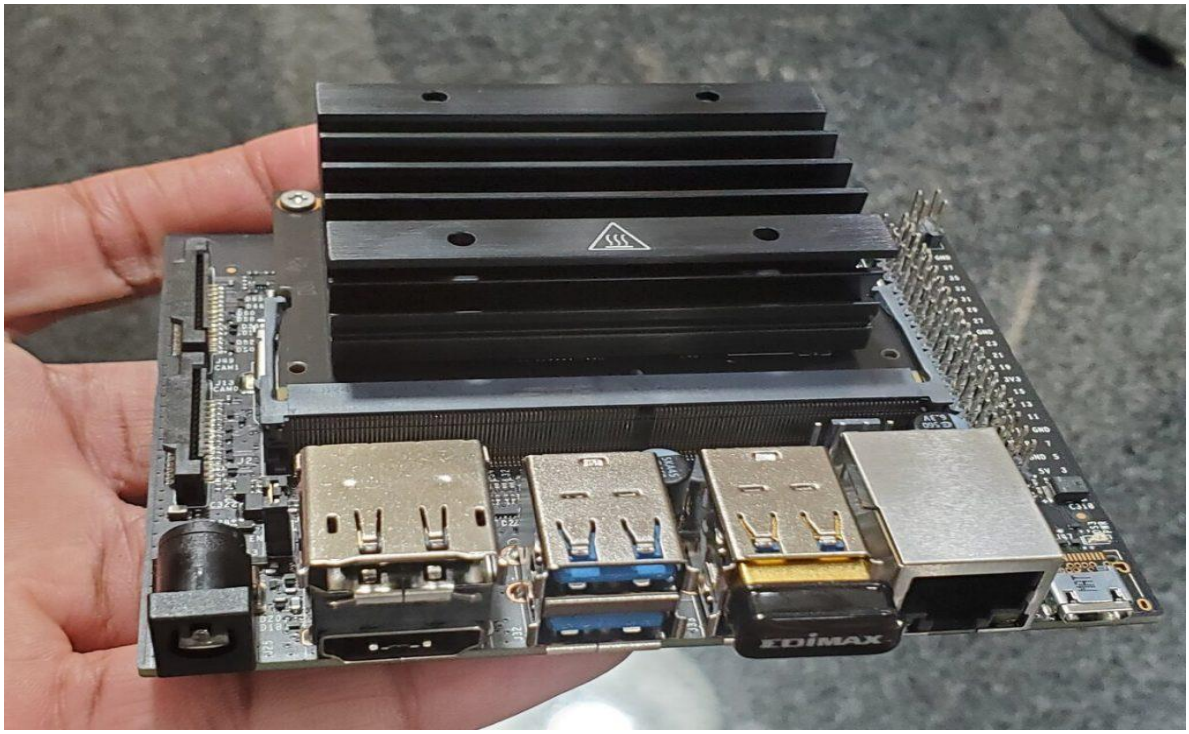


Fig 5.1 Jetson Nano

3. **Storage**: microSD card slot (not included)
4. **Video Encoding**: 4K @ 30 | 4x 1080p @ 30 | 9x 720p @ 30 (H.264/H.265)
5. **Video Decoding**: 4K @ 60 | 2x 4K @ 30 | 8x 1080p @ 30 | 18x 720p @ 30 (H.264/H.265)
6. **Memory**: 4 GB 64-bit LPDDR4 (25.6 GB/s)
7. **Connectivity**: Gigabit Ethernet, 802.11ac wireless (Wi-Fi), Bluetooth
8. **USB Ports**: 4 x USB 3.0, USB 2.0 Micro-B (for power)
9. **GPIO**: 40-pin GPIO header
10. **Display**: HDMI and DisplayPort
11. **CSI Camera Connector**: DSI Display Connector

5.3 BOOT UP AND PROGRAMMING

- **Write the Operating System Image to the microSD Card**

  The Jetson Nano uses a microSD card for storing the operating system. Download, install, and launch the SD Memory Card Formatter for Windows.

  Select the drive where your SD card reader is. Select “Quick format”. Leave “Volume label” blank. Click “Format” to start formatting, and “Yes” on the warning dialog.

  Use a SD card flashing software to flash the OS into the SD card and insert the SD card into your module.

- **FirstBoot**

  Make sure the jumper is pushed into the J48 Power Select Header pins. Connect the Jetson Nano into your monitor, keyboard and your mouse. Connect the power supply to the 5V/4A Power Jack. The developer kit will power on automatically. Allow 1 minute for the developer kit to boot. A green LED next to the Micro-USB connector will light as soon as the developer kit powers on. When you boot the first time, the developer kit will take you through some initial setup.

- **Set Up the Hardware**

  Make the following connections between the BNO055 and the USB to Serial Converter:

  Connect BNO055 Vin to 3.3V (VCC) power.

  Connect BNO055 GND to ground (GND).

  Connect BNO055 SDA (UART transmitter TX) to receiver RXD.

  Connect BNO055 SCL (UART receiver RX) to transmitter TXD.

  Connect BNO055 PS1 to BNO055 Vin (3.3V).

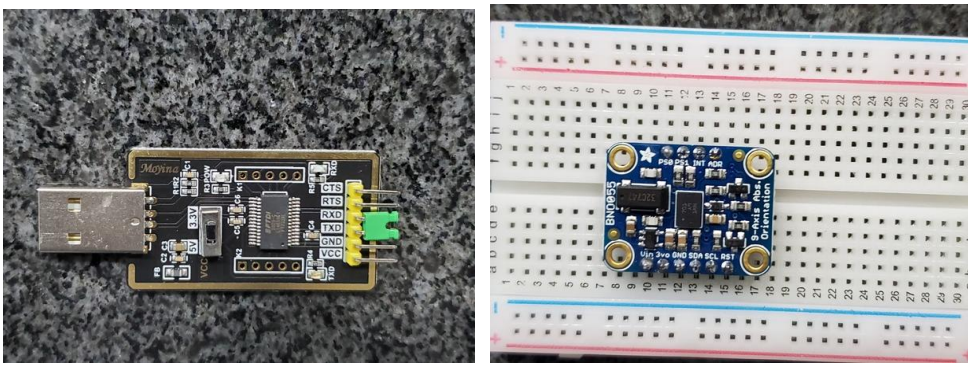


Fig 5.2 USB TTL Converter and BNO055

- **Install and build the BNO055 Robot Operating System (ROS) package**
  After installing the package, build it and add it to the source environment. install the ROS IMU plugin and its necessary items.
- **View the IMU Data**
  Open the ROS terminal and run the "catkin package command", and get the list of devices connected via USB connection.
  Open a new terminal and launch your ROS Rviz to visualize the IMU data.

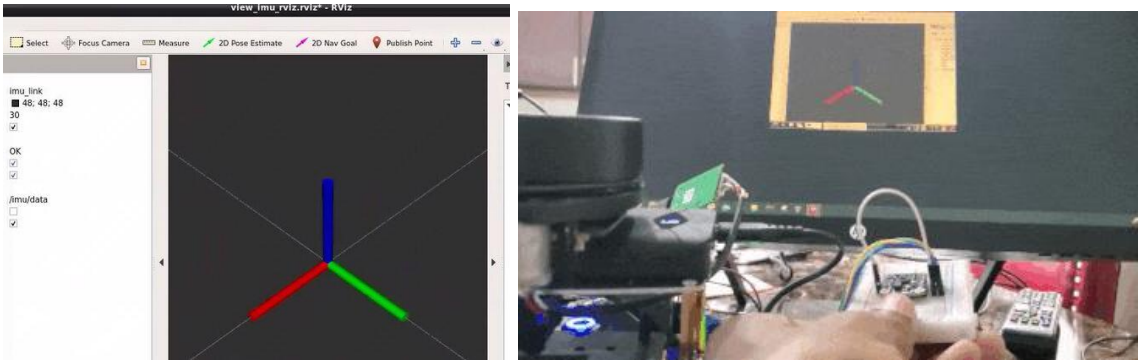


Fig 5.3 visualising the IMU data output from jetson nano

CHAPTER 6

RoNIN ALGORITHM

6.1 INTRODUCTION

The introduction of data-driven inertial navigation provides new opportunities that the pedestrian dead reckoning could not well provide for constraining inertial system error drift on smartphones and has been considered as another promising approach to meet the requirement of location-based services. However, indoor localization systems based on a single technology still have their limitations, such as the drift of inertial navigation and the received signal strength fluctuation of APs, making them unable to provide reliable positioning. To exploit the complementary strengths of each technology, we proposed a feasible fusion framework by utilizing a particle filter to integrate data-driven inertial navigation with localization based on WIFI. For data-driven inertial navigation, under the premise of using the deep neural network with great potential in model-free generalization to regress pedestrian motion characteristics, we effectively combined the method of using gravity to stabilize inertial measurement unit data to make the network more robust. Our neural architecture for inertial navigation, dubbed Robust Neural Inertial Navigation (RoNIN), takes ResNet [8], Long Short Term Memory Network (LSTM) [9], or Temporal Convolutional Network (TCN) [2] as its backbone. Ronin seeks to regress a velocity vector given an IMU sensor history with two key design principles:

1) Coordinate frame normalization defining the input and output feature space and

2) Robust velocity losses improve the signal-to-noise ratio even with noisy regression targets. We now explain the coordinate frame normalization strategy, three backbone neural architectures, and the robust velocity losses. Lastly, the section presents our neural architecture for the body heading regression.

Coordinate frame normalization Feature representations, in our case the choice of coordinate frames, have significant impacts on the training. IMU sensor measurements

come from moving device coordinate frames, while ground-truth motion trajectories come from a global coordinate frame. Ronin uses a heading-agnostic coordinate frame to represent both the input IMU and the output velocity data. Suppose we pick the local device coordinate frame to encode our data. The device coordinate changes every frame, resulting in inconsistent motion representation. For example, target velocities would change depending on how one holds a phone even for exactly the same motions. RIDI proposed the stabilized IMU coordinate frame, which is obtained from the device coordinate frame by aligning its Y-axis with the negated gravity direction. However, this alignment process has a singularity (ambiguity) when the Y-axis points towards gravity (e.g., a phone is inside a leg pocket upside-down), making the regression task harder, usually completely failing due to the randomness. Ronin uses a heading-agnostic coordinate frame (HACF), that is, any coordinate frame whose Z axis is aligned with gravity. In other words, we can pick any such coordinate frame as long as we keep it consistent throughout the sequence. The coordinate transformation into HACF does not suffer from singularities or discontinuities with proper rotation representation, e.g. with quaternion. During training, we use a random HACF at each step, which is defined by randomly rotating ground-truth trajectories on the horizontal plane. IMU data is transformed into the same HACF by the device orientation and the same horizontal rotation. The use of device orientations effectively incorporates sensor fusion 3 into our data-driven system. At test time, we use the coordinate frame defined by system device orientations from Android or iOS, whose Z axis is aligned with gravity.

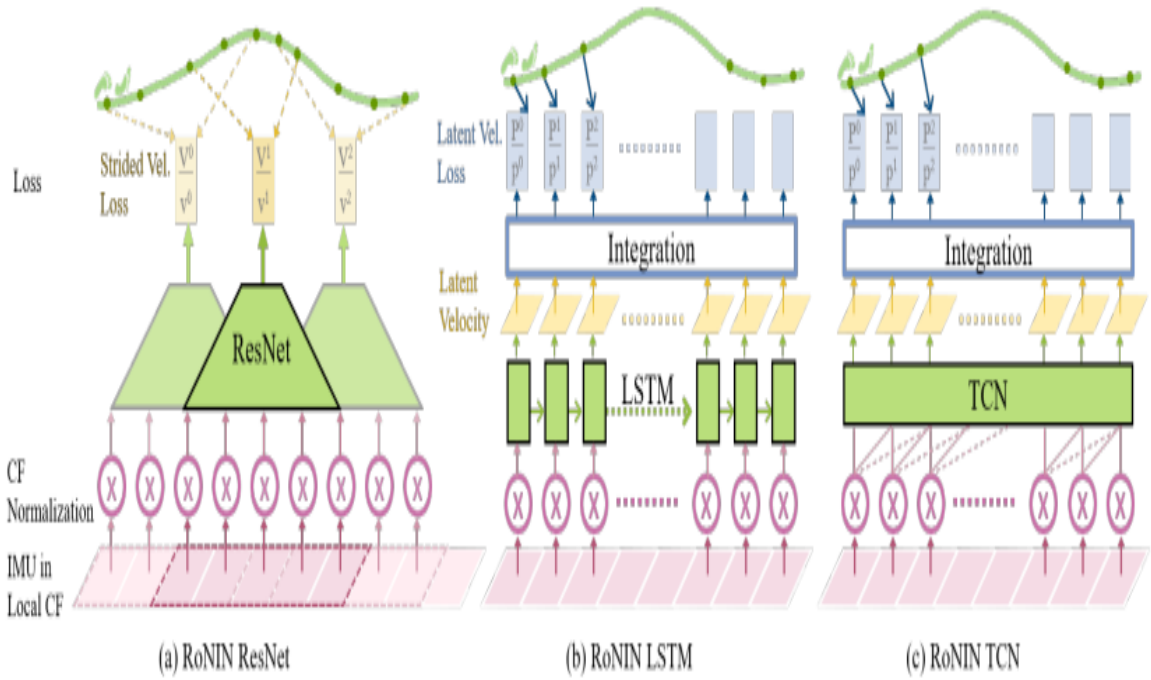


Fig 6.1 RoNIN Architecture

Neural network modules are in green and transformation layers are in white. Different from the position regression, the task of heading regression becomes inherently ambiguous when a subject is stationary. Suppose one is sitting still in a chair for 30 seconds. We need to access the IMU sensor data 30 seconds back in time to estimate the body heading, as IMU data have almost zero information after the sitting event. Therefore, we borrow the RoNIN LSTM architecture for the task, which is capable of keeping a long memory. More precisely, we take the RoNIN LSTM architecture without the integration layer, and let the network predict a 2D vector (x, y), which are sin and cos values of the body heading angle at each frame. During training, we unrolled the network over 1,000 steps for back-propagation. Note that the initial state is ambiguous if the subject is stationary, therefore we only update network parameters when the first frame of the unrolled sequence has velocity magnitude greater than 0.1[m/s]. 4 We use MSE loss against sin and cos values of groundtruth body heading angles. We also add a normalization loss

as k1 − x 2 − y 2k to guide the network to predict valid trigonometric values. We implement the proposed architectures using PyTorch and run our experiments using NVIDIA 1080Ti with 12GB GPU memory. For RoNIN ResNet, we extract one training/validation sample (consisting of 200 frames) every 10 frames. Training samples are randomly shuffled for each epoch. For RoNIN LSTM, we unroll the sequence to 400 steps once per k frame, where k is a random number between 50 and 150. Unrolled sequences are randomly batched to update network parameters. For RoNIN TCN, we construct one training/validation sample with 400 frames per k frame, where k is again a random number between 50 and 150. For RoNIN ResNet (resp. RoNIN LSTM/TCN), we use a batch size of 128 , an initial learning rate of 0.0001 (resp. 0.0003), and an ADAM optimizer while reducing the learning rate by a factor of 0.1 (resp. 0.75) if the validation loss does not decrease in 10 epochs, where the training typically converges after 100 (resp. 300/200) totaling 10 hours (resp. 40/30 hours). For linear layers, we apply dropout with the keep probability of 0.5 for RoNIN ResNet and 0.8 for RoNIN LSTM/TCN.

- As a baseline without the coordinate frame normalization, we supply the raw IMU sensor measurements as input and the ground-truth velocity in the local device coordinate frame as output. Note that all three dimensions are needed and we rotate predicted velocities to the heading agnostic coordinate frame for position integration. The vertical axis is discarded during evaluations.
- As a baseline without the stridden velocity loss for RoNIN ResNet, we apply Gaussian smoothing with $\sigma = 30$ [22] to reduce the noise of ground-truth velocities, as suggested by RIDL.

6.2 DENSE HISTORY OF LOCATION INFERENCE

Our system, coined DHL, works with any modern smartphone, utilizing an IMU and a WiFi receiver. The input to the system is IMU sensor data, global positions by a geo-localization API such as Google FLP at 1/60Hz, and a geo-localized floorplan. This section provides the system overview, which consists of the following steps. A data-driven inertial navigation algorithm (RoNIN) estimates a relative motion trajectory at 50Hz from IMU.

A non-linear least squares optimization is used to geo-localize the trajectory subject to positional constraints by FLP and regularization terms penalizing the deviations from the original trajectory with a sensor-centric error model. Besides the geo-localization, this alignment removes scale and/or rotational drifts inherent to the inertial navigation. A convolutional neural network (CNN) is trained to estimate a correction to the aligned trajectory to be consistent with a floor plan. Lastly, the system runs the optimization and the CNN steps once more, while changing the positional constraints from FLP to the current location history estimation, subsampled once every 200 frames**.**

6.3 DENSE HISTORY LOCATION DATASET

IMU data consists of accelerations (200Hz), angular velocities (200Hz), and device orientations (100Hz). A subject walks on a single floor of a building while carrying a smartphone naturally by hand or in a pocket. We have performed gyro-calibration (e.g., placing on a table for 10 seconds) in advance to simulate the real-world scenarios, which cause severe “bending” due to rotational accumulation errors. WiFi-based geo-positions are collected by the Google Fused Location Provider (FLP). We call the FLP API at 1Hz to obtain geo-positions as much as possible for evaluation. This rate is impractical for real-world applications as the API drains the battery. We subsample FLP data at a realistic rate (1/60Hz) as an input to a system. The Floorplan of the testing area was formed and global coordinates were manually specified between the floorplan. Ground-truth (GT) positions for shopping malls are obtained as either a sparse set of positions from real-time user clicks or dense trajectories from visual-inertial SLAM. First, a set of ground-truth positions is obtained through an Android app where a subject clicks a floorplan on a smartphone in real time during the data acquisition. In total, 200 GT positions are collected for 5.0 hours of data from the three buildings, where a single trajectory spans 10 minutes on average. Second, in one of the floorplans, we use the visual-inertial SLAM of a tango phone to collect relative trajectories, which are geo-localized by a rigid transformation with manual offline specifications to generate dense GT data. Due to the memory limitation of the visual SLAM system, the trajectories with dense GT positions are much shorter and hence less

challenging, where a single trajectory spans 4.2 minutes on average. These data are used only for training a CNN module, where dense GT positions are obtained by geo-localizing inertial navigation trajectories with real-time user clicks by an optimization algorithm in our system.

6.4 IMU AND Wi-Fi FUSION BY OPTIMIZATION

A relative motion trajectory by RoNIN is a sequence of speed sf and motion direction angle θf at every frame f. The position of the trajectory at frame f is calculated as ∑ f i=1 si [cosθi,sinθi] T, assuming that motions are 2D on a single floor and start from the origin. The units of the positional and angular quantities are meters and radians in our formulation, respectively. Inertial navigation exhibits two types of errors: 1) accumulation errors in the device orientation due to gyroscope bias; and 2) scale inference errors (i.e., the walking speed varies per person). We solve for the scale correction Δsf, the angle correction Δθf, and the positional displacement at the first frame Δ −→P0, where the corrected position is calculated as

→Pf = Δ −→P0 + f ∑ i=1 si ·Δsi [cos(θi +Δθi), sin(θi +Δθi)]T. Note that the scale (resp. angle) correction terms should be smooth, and we define a correction term every 100 (resp. 20) seconds while using linear interpolation to derive the amount of correction at an arbitrary frame [5]. In reality, angle corrections should be on the device axes, but we make a simplified assumption and directly model corrections in the heading angle around the gravitational axis. Concretely, we formulate a non-linear least squares optimization problem subject to the positional constraints from FLP and the regularization constraints

(w1 = 10,w2 = 200): ∑ f∈FFLP max(k −→Pf − −−→ P FLP f k 2 −r FLP f,0)

+w1E REG 1).

6.5 RESULTS

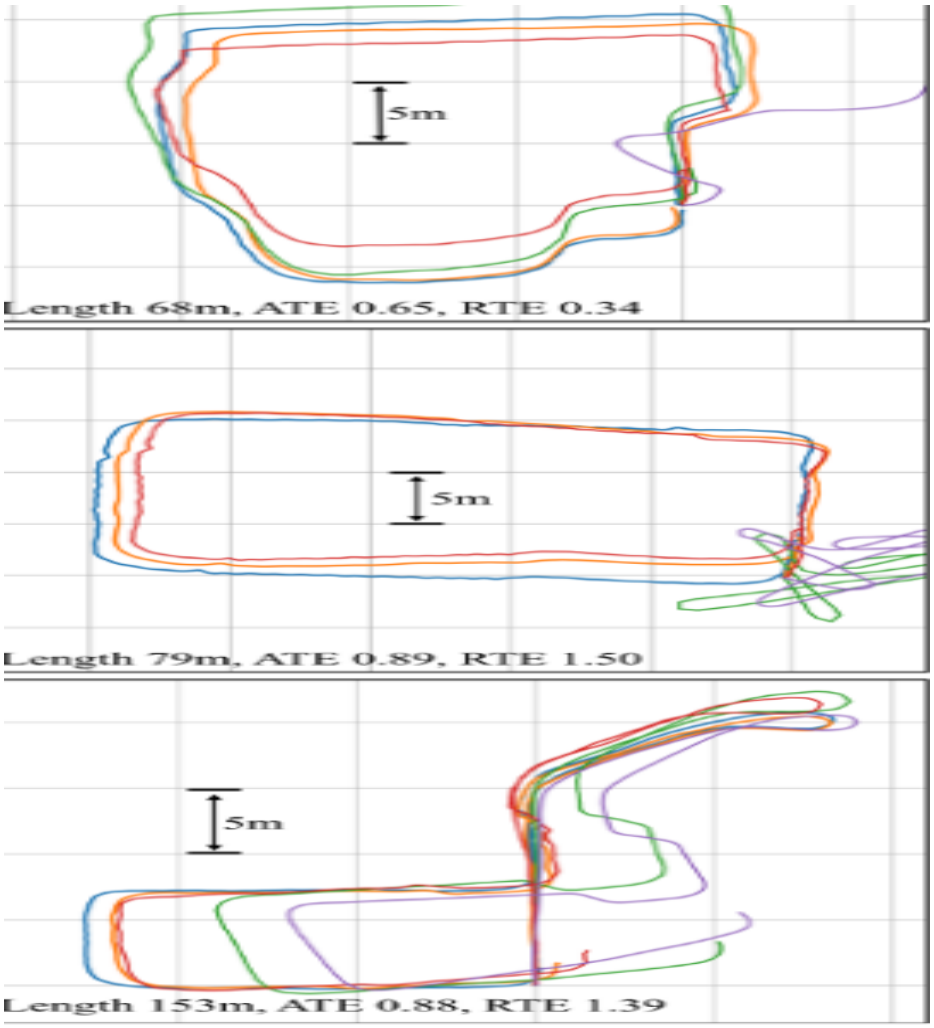


Fig 6.2 RoNIN Results

We select 3 examples from each dataset and visualize reconstructed trajectories from all competing methods. We chose RoNIN LSTM as our method since its performance is more consistent across datasets. For each sequence, we mark the trajectory length and report both ATE and RTE of our method. We also mark physical dimensions in all sequences to demonstrate that our method estimates trajectories with accurate scales. Three examples from our RoNIN dataset (left column) contain complex natural motions, where all other methods fail. Sequences from the RIDI dataset (middle column) contain hard motions. In

particular, the middle example of the middle column contains extensive backward motion, which our method handles elegantly.

1) the new benchmark with the large and diverse quantity of IMU-motion data as in real day-to-day activities;

2) new neural inertial navigation architecture making significant improvements over challenging motion cases; and

3) qualitative and quantitative evaluations of the current competing methods over the three inertial navigation datasets. The major limitation of our approach comes from the reliance on device orientation estimations. The performance degrades significantly given data with poor device orientations, which is the main focus of our future work.

CHAPTER 7

KALMAN FILTER AND EXTENDED KALMAN FILTER

7.1 INTRODUCTION

Kalman filters are a set of algorithms based on the idea of a filter described by Rudolf Emil Kalman in 1960. Kalman filters are used in various application domains, including localization, object tracking, and navigation. The text provides an overview and discussion of the possibilities of using Kalman filters in indoor localization. The problems of static localization and localization of dynamically moving objects are investigated, and corresponding stochastic models are created. Three algorithms for static localization and one algorithm for dynamic localization are described and demonstrated. All algorithms are implemented in the MATLAB software, and then their performance is tested on WIFI from a real indoor environment. The results show that by using Kalman filters, the mean localization error of two meters can be achieved, which is one meter less than in the case of using the standard fingerprinting technique. In general, the presented principles of Kalman filters are applicable in connection with various technologies and data of various natures.

The Kalman filter represents a theoretical basis for various recursive methods in the examination of stochastic (linear) dynamic systems. The algorithm is based on the idea that the unknown state of the system can be estimated using certain measured data. The filter is named after Rudolf Emil Kalman, a Hungarian mathematician living in the United States, who published it in an article in 1960. In the next period, various algorithms based on the principles of the Kalman filter were derived by various authors. These algorithms are all referred to as Kalman filters and can be suitably used in certain specific situations, for example, resulting from failure to meet some theoretical assumptions of the classical Kalman filter in solving practical problems. The Kalman filter, or generally, Kalman filters,

can be used in various application domains, including localization, object tracking, and navigation.

The development in the area of mobile technologies and smart solutions stimulates the need to deal with the issue of locating mobile devices or objects in space. An intelligent location-aware system can adapt and provide the user with information relevant to their geographical context. Services based on the awareness of the user's location are known as location-based services. For the use of these services, it is important to be able to determine the position of a device (and a person using this device) not only in outdoor space but also very often inside buildings (indoor localization). In addition to static positioning, it is possible to track a dynamically moving object and also predict its motion. Indoor localization is used in a number of areas, for example, navigation in public buildings, along with the intelligent provision of necessary or interesting context information to the user, depending on where they are currently located. Among other application areas of indoor localization rescue and crisis management, security and military applications, smart homes, targeted marketing, entertainment and tourism, and many more can be mentioned. Indoor localization also gains increasing importance in connection with today's popular concepts such as augmented reality.

7.1 MOTIVATION FOR USING KALMAN FILTER

However, the disadvantages of using the radio signal in indoor localization follow from the physical patterns of its propagation and the nature of the indoor environment. The signal inside a building collides with a number of different obstacles from different materials (walls, furniture, people, etc.) and thus absorbs, refracts, or reflects. Signal properties (signal strength) vary in such a complex environment, depending on location and time, in a way that cannot be simply described and formalized. Measured signal strength at a particular location also depends on the mobile device (receiver), its current orientation during the measurement, etc. As a result of all the above, it is difficult to achieve the ideal accuracy of localization that would be required inside a building. Therefore, there is constantly an effort to find extensions or improvements to existing techniques or

techniques that are completely new, which will allow for higher accuracy of localization. Very often, these are different mathematical and statistical methods that help to reduce the impact of noise in radio data generated by the aforementioned physical principles of signal propagation. Kalman filters are an example of such a method. Kotanen et al are one of the first who represent the possibilities of using the extended Kalman filter (EKF) in indoor localization. They deal with solving a problem of static localization within a single room, the filter is applied to data on the distances of the localized object to the reference transmitting points calculated from the measured strength of the Bluetooth signal. The extended Kalman filter is also used by Yim et al, their localization technique is based on existing wireless local area networks in buildings.

Two modifications of the standard Kalman filter called the position Kalman filter (PKF)and the fingerprint Kalman filter (FKF) have been developed for use in indoor localization. Both algorithms were originally used by authors for static localization in a building with a Wi-Fi network. The PKF has as its input already calculated position estimates (obtained, for example, by the fingerprinting technique), while the FKF works directly with the measured signal strength, and therefore has a better ability to filter the noise contained therein, subsequent position estimation is based on the principles of the fingerprinting technique and the WKNN algorithm. The Kalman filter can also be used as a tool to combine data of a different nature from different sources. This is used, for example, by when dealing with a dynamic localization problem, they combine radio data and motion data from the respective mobile device sensors. In addition to Kalman filters, particle filters are also used in indoor localization Particle filters have more general application possibilities, but the disadvantage is higher computational costs limiting their real-time use on devices with limited performance. Additionally, the accuracy achieved is not significantly better. Localization techniques using neural networks, genetic algorithms, etc. also exist. The aim of this text is to provide an overview of the possibilities of using Kalman filters in indoor localization. Different localization algorithms based on the Kalman filtering idea will be described and implemented. The algorithms will be based on the above-mentioned papers and some of them will be modified and extended in various

ways. Emphasis will also be placed on their practical demonstration and subsequent testing under realistic conditions, as some of these algorithms are tested by their authors only in precisely prepared laboratory conditions (such as localization in one empty room with four radio transmitters at corners, etc.) or on simulated data. In addition to the original use for static localization, some algorithms will also be employed when solving a problem of localization of a dynamically moving object. Corresponding stochastic models for representing these problems will be created. The obtained results will be discussed so that the text can help and serve as a starting point inspiration for practitioners and researchers dealing with indoor localization and related tasks.

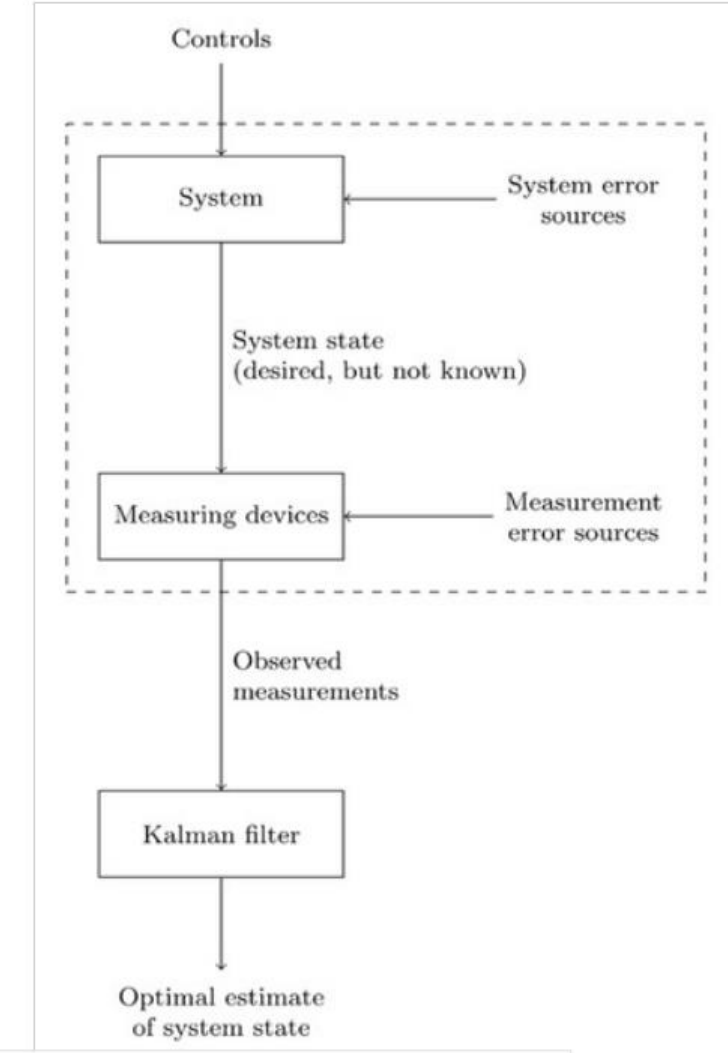


Fig 7.1 Kalman filter flow chart

The Kalman filter works with all available information, i.e., it uses all available measured data, the system model together with a statistical description of its inaccuracies, noise, and measurement errors, as well as information about the initial conditions and the initial state of the system.

7.2 **KALMAN FILTER ALGORITHM**

$$x_k = \phi_{k-1} x_{k-1} + w_{k-1} \quad (1)$$

$$z_k = H_k x_k + v_k \quad (2)$$

Equation (1), referred to as the state equation, describes system dynamics, the vector $\boldsymbol{x}k \in \mathbb{R}n \in \mathbb{R}$ is the (unknown) system state vector at time tk, the matrix $\boldsymbol{\Phi}k-1 \in \mathbb{R}n \times n\Phi-1 \in \mathbb{R}\times$ represents dynamic evolution of the system state between times $tk-1-1$ and tk. Equation (2) is called the measurement equation, the vector $\boldsymbol{z}k \in \mathbb{R}m \in \mathbb{R}$ is referred to as the system output vector, the measurement vector or the observation vector, the matrix $\mathbf{H}k \in \mathbb{R}m \times n \in \mathbb{R}\times$ then describes the relationship between the system state and the measurement. Since this is a stochastic system, the vectors $\boldsymbol{x}k$ and $\boldsymbol{z}k$, k=0, 1, 2, …=0, 1, 2, …, can be considered random variables, their sequences $\{\boldsymbol{x}k\}$ and $\{\boldsymbol{z}k\}$ are then random (stochastic) processes.

$\{\boldsymbol{w}k\}$ and $\{\boldsymbol{v}k\}$ are random noise processes, let us assume that they are uncorrelated Gaussian processes with zero mean and covariance matrix $\boldsymbol{Q}k$ and $\boldsymbol{R}k$, respectively, at time tk (the processes have properties of Gaussian white noise).

Furthermore, let $\boldsymbol{x}0$ be a random variable with a Gaussian (normal) distribution with known mean $\boldsymbol{x}0$ and known covariance matrix $\mathbf{P}0$. In addition, let $\boldsymbol{x}0->0$, and both noises always be uncorrelated. So, we can summarize that for all tk.

The aim of the Kalman filter is to find an estimate of the state vector $\boldsymbol{x}k$ at time tk, denoted as $\hat{\boldsymbol{x}}k$, so that this estimate is optimal (for example, in terms of minimizing the

mean square error). The algorithm of the Kalman filter is recursive, the calculation at general time tk consists of two main steps-first, the a priority estimate $\hat{\boldsymbol{x}}k$ at time tk is calculated by substituting the a posteriori estimate from time $tk-1$ into the deterministic part of the state equation, and then this estimate is updated (corrected) using the measurement taken at time tk, which results in obtaining the a posteriori estimate $\hat{\boldsymbol{x}}k$ at time tk.

The following relation can be written for the a priori estimate of the state vector $\hat{\boldsymbol{x}}k$ at time tk, the uncertainty of this estimate being expressed by the a priori error covariance matrix $\mathbf{P}k$.

$$\hat{x}_k = \phi_{K-1}\hat{x}_{k-1} \tag{3}$$

$$P_{k-1} = \phi_{k-1}P_{k-1}\hat{x}_{k-1} \tag{4}$$

7.3 **EXTENDED KALMAN FILTER**

The use of the Kalman filter, as mentioned above, assumes the system is linear. In reality, however, the system dynamics or the relationship between its state and measurements (or both) may often be nonlinear. The system model then has this form (noise is still assumed to be additive).

$$\hat{x}_{k(-)} = \phi_{k-1}\left(\hat{x}_{k-1^{(+)}}\right) \tag{5}$$

$$p_{k(-)} = \Phi_{-1}P_{k-1^{(+)}}\phi_{k-1}^{T} + Q_{k-1}\hat{x}_{k(+)} \tag{6}$$

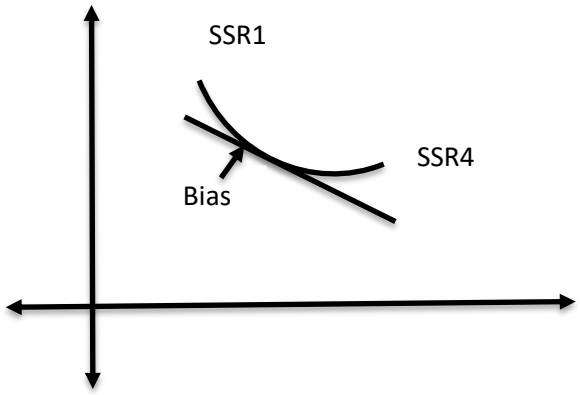
$$\hat{x}_{k(-)} + k_k\left[z_k - h_k\left[\hat{x}_{k(-)}\right]\right] \tag{7}$$

One of the modifications of the standard Kalman filter, which allows operating with non-linear systems, is the extended Kalman filter (EKF). The EKF is based on the linearization of the non-linear function around the current estimate and is particularly suitable for weakly non-linear systems. From the point of view of indoor localization, it is

obvious that the sought position is represented in the system model by the unknown vector $\boldsymbol{x}$. As a measurement vector $\boldsymbol{z}$, different data can be understood. The following part of the text will describe the implementation of three algorithms for static localization and one algorithm for dynamic localization, with measurements $\boldsymbol{z}$ always being data of a different nature. The problem of localization within one floor of a building, i.e., the determination of the coordinates in a two-dimensional space (in the plane), is addressed. We derived a way to determine the bias error. Our earlier collections of datasets were completely based on an assumption of a bias error value using trial error value. Our trial and error method came up with a value of 0.6 to 0.7 but we still were unsure of the exact value. But later, we came up with a derivate formula that not only simplifies and makes the whole prediction process easier but also gives us an easy way to determine the value of bias error which is further used in solving complex matrix equations. We obtained a value around 0.67.

SSR – Sum Of Squared Residuals

SSR = $\sum(observed\ i - predicted\ i\ ) \wedge 2$



$$d(SSR)/d(predicted) = d\left(\sum(observedi - predictedi)^2\right)/d(predicted)$$

$$= \sum -2\,(observedi - predictedi)$$

d(predicted) / d(bias) = 1 [under bias]

d(predicted) / d(bias) = 0 [without bias]

Therefore we get ,

d(predicted)/dB = $\sum(\ observedi\ -\ predictedi)$^2 $\times$ 1 which determines the slope value thus helps us to determine the bias value.

7.4 CONCLUSION

The text studied the possibilities of using Kalman filters in indoor localization. Three algorithms for static localization and one algorithm for locating dynamically moving objects were described. All algorithms were implemented in MATLAB, and subsequently tested on data from a real indoor environment. The results showed that by using Kalman filters, the mean localization error of two meters can be achieved in both types of problems, which is one meter less than in the case of using the standard fingerprinting technique. The best results were obtained by applying the Kalman filter directly to the raw data on the measured radio signal strength. The advantage of Kalman filters can also be seen in the low computational costs, which do not prevent their real-time use. Despite the improvements brought by the use of Kalman filters, it is still difficult to reach the ideal accuracy of localization that would be required in a complex indoor environment with the use of noisy radio data. Some additional improvement could be achieved by combining Kalman filters with other techniques, such as the PDR technique mentioned above. In general, the presented ideas and principles of Kalman filters are applicable in connection with various technologies and data of various natures.

CHAPTER 8

WIFI FUSION

8.1 INTRODUCTION

Recently, several indoor localization solutions based on WiFi, Bluetooth, and UWB have been proposed. Due to the limitations and complexity of the indoor environment, the solution to achieve a low-cost and accurate positioning system remains open. We present a WiFi-based positioning technique that can improve the localization performance from the bottleneck in ToA/AoA. Unlike the traditional approaches, our proposed mechanism relaxes the need for wide signal bandwidth and large numbers of antennas by utilizing the transmission of multiple predefined messages while maintaining high-accuracy performance. The overall system structure is demonstrated by showing localization performance with respect to different numbers of messages used in 20/40 MHz bandwidth WiFi APs. Simulation results show that our WiFi-based positioning approach can achieve 1 m accuracy without any hardware change in commercial WiFi products, which is much better than the conventional solutions from both academia and industry concerning the trade-off of cost and system complexity. Recently, indoor positioning has attracted a lot of attention from research as well as industry communities. This is partly driven by the increasing market demand for indoor location-based services. Due to the limitation of GPS signal strength indoors, nearby anchors with known positions are generally needed for an indoor positioning system. In general, there are two components in indoor positioning systems. One is nearby anchors with the knowledge of their own location information, while the other is a positioning device with the objective of identifying its location by processing wireless signal through nearby anchors. Although the potential needs of indoor navigation services have been already addressed from industry aspects, the ultimate solution in terms of accuracy, reliability, real time, and low cost remains open. There are several factors that make the design of indoor positioning systems challenging:

1) Cost: The need to deploy nearby anchors and the development of localization devices to identify objects are costly. Numbers of nearby anchors: Unlike the traditional navigation GPS system, which uses 31 active satellites in orbit, the indoor environment and space sometimes limit the number of anchors.

2) Complicated indoor environment: The wireless signal used to measure the distance and angle indoors usually suffers significantly from obstacles, such as walls, objects, and/or human beings, which lead to multipath effects. Due to its wide deployment, we expect WiFi to become a prominent tool for indoor positioning. In this article, we use WiFi access points (APs) with multiple antennas as nearby anchors, while the positioning device could be any mobile platform with WiFi capability such as smartphones or tablets. Since the number of WiFi APs is limited and the system bandwidth is narrow (up to 40 MHz for 802.11n and 160 MHz for 802.11a c), we propose a novel method of sending multiple messages to improve the time of arrival (ToA) measurement and angle of arrival (AoA) estimation.

8.2 TIME OF ARRIVAL

Time of arrival is the travel time between a transmitter and a receiver. The distance can be calculated using travel time multiplied by the speed of light. To measure the travel time in the air, this approach usually requires synchronization between transmitters and receivers. In addition, it requires at least three anchors to have the plane-domain (2D) localization as depicted and four anchors for 3D localization. The positioning performance is decided by a signal's bandwidth as well as sampling rate. When the arrival wireless signal is sampled in receiver as depicted , the first sample that captures the arrival signal is not exactly the ToA. In other words, when a signal's bandwidth is not wide enough, the ToA measurement may result in a wide error range of distance. For example, a wireless system with 10 MHz bandwidth as well as a sampling rate can only measure the time duration up to 1 × 10–7 s resolution. Therefore, the maximal error in distance is up to 3 × 108 × 10–7 = 30 m. When the wireless system has 1 GHz bandwidth, the receiver can measure up to 1 × 10–9 s resolution such that the maximal error in distance is lower than

30 CM. Current popular solutions applied to ToA are ultra-wideband (UWB) systems. The accuracy a UWB system can achieve is up to 1 cm. However, it requires a very wide bandwidth as well as a special hardware design to support localization, which results in very high hardware costs.

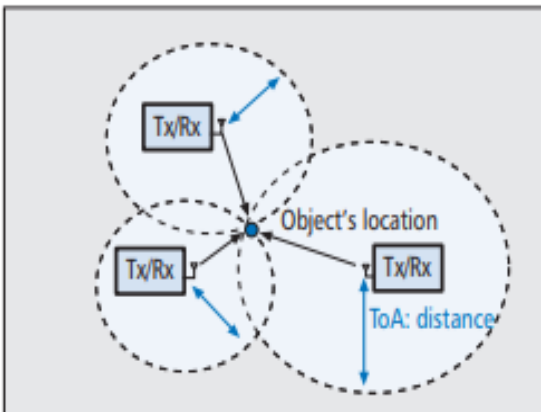


Fig 8.1 Time of arrival

8.3 ANGLE OF ARRIVAL

The angle of arrival measurement is the method that can determine the incoming signal direction from a transmitter on the antenna array. By exploiting and detecting phase differences among antennas, the direction of an incoming signal can be calculated. In order to locate the position, this approach requires two anchors with antenna arrays at different places to obtain the object position as depicted. However, due to multi-path affection, the AoA in terms of line of sight is hard to obtain. An example of a commercial solution applied to AoA is Quuppa's HAIP system where the positioning accuracy can be achieved from 0.5 to 1 m. However, it requires a specific hardware device including 16 array antennas with a transmitter as nearby anchors and a special tag as a positioning device through Bluetooth enhancement of a wireless signal.

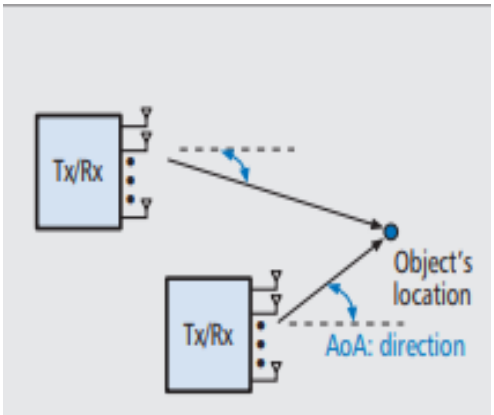


Fig 8.2 Angle of arrival

8.4 HYBRID TOA/AOA APPROACHES

Due to the complicated indoor environment and limited number of nearby anchors, a hybrid ToA/AoA approach has been introduced. By utilizing the information measured from AoA and ToA, the number of nearby anchors can be reduced. As illustrated, it is possible to localize an object using a single nearby anchor. The hybrid approach suffers the same challenge of signal bandwidth as well as the number of antennas. Nevertheless, hybrid systems leverage the benefits of both mechanisms.

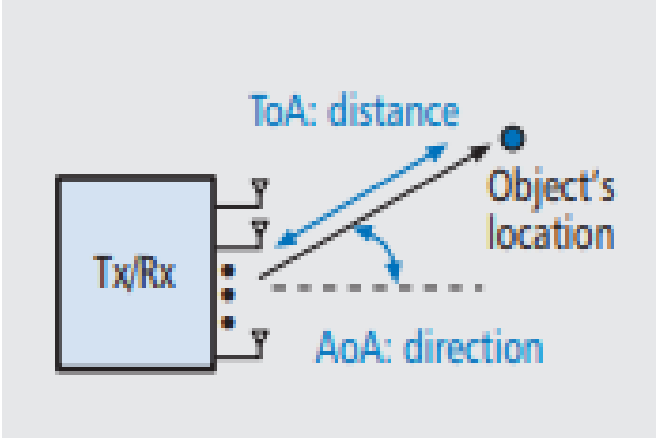


Fig 8.3 TOA/AOA model

8.5 RECEIVED SIGNAL STRENGTH AND FINGERPRINT

The received signal strength (RSS) and fingerprint is a site-survey approach for positioning. For RSS-alone approaches, the received signal strength ratio reflects the distance information. Through calibration of transmitter power with a corresponding free-space channel model established by measuring distance and power at each point in advance, the coarse distance can be estimated for each anchor node. To further improve location precision, RSS with fingerprint combination is proposed. In general, due to the multi-path effect, each location receives a unique signal through the combination of various rays from different paths. Thus, the signal property, such as frequency response, and signal strength regarding the I/Q channel has its own fingerprint. By associating the signal fingerprint with the target, the anchor can deduce possible locations from a pre-measured fingerprint database. This mechanism only requires one anchor node instead of multiple anchors for positioning. The signal fingerprint in each grid is measured in advance. When the object sends the signal toward a nearby anchor (Tx/Rx), the anchor can choose the most similar fingerprint from the database regarding the received signal to do localization. The localization accuracy will depend on not only the size of the grid area but also the signal difference among grids. When the difference between signal fingerprints is small, the uncertainty increases. A commercial solution applying RSS indication (RSSI) alone is Apple's iBeacon technology, and one applying RSS fingerprint-based is Apple's WiFi SLAM. For iBeacon, the location is calculated via a Bluetooth Low Energy system. The coarse distance can be defined as immediate, near, and far, three different statuses that refer to within centimeters, a couple of meters, and more than 10 meters, respectively. The WiFi SLAM solution applies a WiFi RSS fingerprint-based mechanism with gyroscope sensor assistance [8] to record each location's fingerprint. WiFi SLAM can achieve up from 1.75 to 2.5 m accuracy. However, it requires a site survey fingerprint in advance. When the environment is dynamic, such as a shopping mall with a moving crowd, the performance can degrade dramatically. Moreover, the calculation loading increases exponentially with the number of fingerprint points in the database.

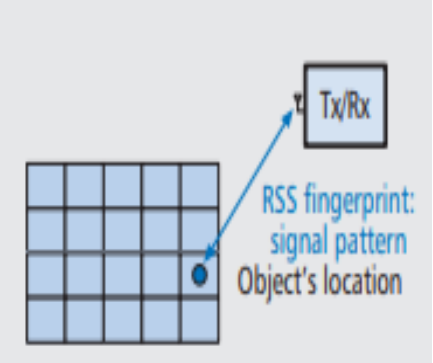


Fig 8.4 RSS model

8.6 THE WIFI-BASED INDOOR POSITIONING SYSTEM

Due to the fact that WiFi technology is widely deployed in indoor environments, we apply our proposed mechanisms to a WiFi system with an access point (AP) with multiple antennas as nearby anchors. The positioning devices could be mobile platforms with WiFi capability such as smartphones and tablets. When there is only one WiFi AP, we apply a hybrid AoA/ToA system to locate the target's positioning. When the number of nearby WiFi APs are two or more, we applied AoA to obtain higher accuracy location. This design can provide accurate positioning service even when the number of nearby anchors is limited. Since WiFi bandwidth is not as wide as that of UWB, we apply the multiple message approach to assist ToA measurement as well as AoA estimation. In our proposed system, each WiFi AP is equipped with NA antennas, while the user's mobile phone is equipped with a single antenna only.

8.7 DISTANCE MEASUREMENT

In our system, we apply the round-trip time (RTT) approach to obtain distance without requiring time synchronization between transmitters and receivers. In the traditional RTT measuring approach, the transmitter sends the message toward the receiver and records the transmit timestamp. Then, when the receiver responds the message back to the transmitter, the RTT can be calculated from the interval between sending and arrival times. However,

this RTT time also includes the receiver processing delay time, and a measurement error occurs. In our scheme, the receiver does not require responding to the message immediately. The transmitter first sends out a burst of messages and records the transmit timestamp of the first message as tS. Then the receiver uses our proposed ToA approach to measure the arrival time stamp tS¢ regarding the first message and allocates the response time started in tS¢ + i × TU, where the TU is the agreement between transmitter and receiver, and i is the arbitrary number decided by the receiver for allocation convenience. Once the transmitter estimates the first message from receipt with arrival times stamp tR, the RTT is RTT = (tR – tS) modulo TU One can notice that the ToAs in tS¢ and tR are calculated at each side individually. Hence, the LTI property does not need to hold for the whole RTT estimation process. It only needs to hold for a short time duration while calculating tS¢ or tR. Here, we choose TU = N × TS, since each OFDM length is NTS. Therefore, the maximal measuring distance in a ToA is NTS× C/2 where C is the speed of light. The maximal measuring distance under this scenario is up to 480 m. Since the WiFi indoor communication distance is usually much less than 480 m, the positioning range from the ToA is quite reliable. By preallocating packet sending in i × N × TS, we can reduce the process delay to nearly zero. ANGLE MEASUREMENT In our system, the mobile phone sends multiple messages toward the WiFi AP, where the AoA is measured by using the channel estimation technique. The AoA can be obtained by any pair of antennas. When the number of antennas is more than two, the AoA can be jointly estimated for better performance. LOCALIZATION PROCEDURE In previous sections we have illustrated how to find out the distance and angle in our system. In the following, we describe the localization procedure:

• User device: Requests positioning service of the WiFi AP

• WiFi AP: Starts sending requests for RTT measuring

1. After granting the positioning request, the WiFi AP starts to send burst M messages and records the sending timestamp of the first message for later RTT time calculation.

2. Each message’s content is the same and contains multiple subfrequency pilots of an OFDM symbol.

• User device: Measures the arrival time and responds with a burst of messages for RTT measurement

1. The user device reconstructs the received signal by re-ordering the M messages with relative time difference.

2. The ToA of the first received message can be obtained from the reconstructed signal.

3. The user device chooses arbitrary i for the sending time i × N × TS to send the burst M messages where each message contains multiple subfrequency pilots.

• WiFi AP: Measures the RTT and AoA in terms of user device

1. The ToA of the first received message can be obtained from the reconstructed signal.

2. The distance is the RTT multiplied by the light speed divided by 2 where the RTT is the time difference between sending time and received time modulo by i × N × TS.

3. The AoA is measured by using the message nearest the ToA to estimate channel state information.

4. The WiFi AP returns its own reference location as well as the distance and direction back to the user device.

• User device: Uses AoA/ToA to locate its own position and sends requests to other WiFi APs 1. If the user device only obtains a position from a single WiFi AP, the device calculates its own position by using the WiFi AP reference location with distance and direction angle. If the user device has positions from more than one WiFi APs, the device only uses AoAs to deduce its own position by combining directions with each WiFi AP.

8.8 CONCLUSION

We illustrate different mechanisms and their applications used in indoor localization in terms of cost and complexity. The conventional ToA/AoA approaches to overcome bandwidth limits and numbers of multipaths are introduced and discussed. We show that our proposed approach can improve ToA resolution compared to the super-resolution method under limited signal bandwidth without heavy calculation loading. Moreover, our proposed AoA approach through multiple message assistance can reduce the need for multiple antennas. We demonstrate the performance by applying the mechanism to a WiFi-

based indoor positioning system where that can support localization with just a single WiFi AP. When the number of nearby WiFi APs is two or more, the position accuracy can improve further by AoA joint positioning. Finally, the simulation showed that our proposed approach provides superior performance without changing hardware settings in a practical WiFi AP setting.

CHAPTER 9

CONVOLUTION NEURAL NETWORK (CNN)

9.1 INTRODUCTION

The agenda for this field is to enable machines to view the world as humans do, perceive it in a similar manner, and even use the knowledge for a multitude of tasks such as Image and video recognition, Image Analysis and classification, Media Recreation, Recommendation Systems, Natural Language Processing, etc. The advancements in Computer Vision with Deep Learning have been constructed and perfected with time, primarily over one particular algorithm — a **Convolutional Neural Network**.

A **Convolutional Neural Network (ConvNet/CNN)** is a Deep Learning algorithm that can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image, and be able to differentiate one from the other. The pre-processing required in a ConvNet is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, ConvNets have the ability to learn these filters/characteristics.

9.2 FLOOR PLAN FUSION BY CNN

A CNN in its simplest form when used on images consists of a convolutional layer where several kernels are applied to every pixel, or convoluted with the image.

9.3 ARTIFICIAL NEURAL NETWORK

ANNs are a powerful machine learning model loosely inspired by the human brain. They are a collection of interconnected neurons organized into layers. The first layer is the input layer. It consumes the raw data, processes it, and passes information on to the next layer as output. A common type of layer in a network is a dense layer or fully-connected layer. This layer will take as input, the outputs of all the neurons in the previous layer.

Information flows through the network until it reaches the output layer which makes a prediction. When information is only passed as the output of one layer to the input of another it is considered a feed-forward neural network layers in between the input and output layers are called hidden layers. Typically, when a network contains multiple hidden layers they are referred to as deep neural networks.

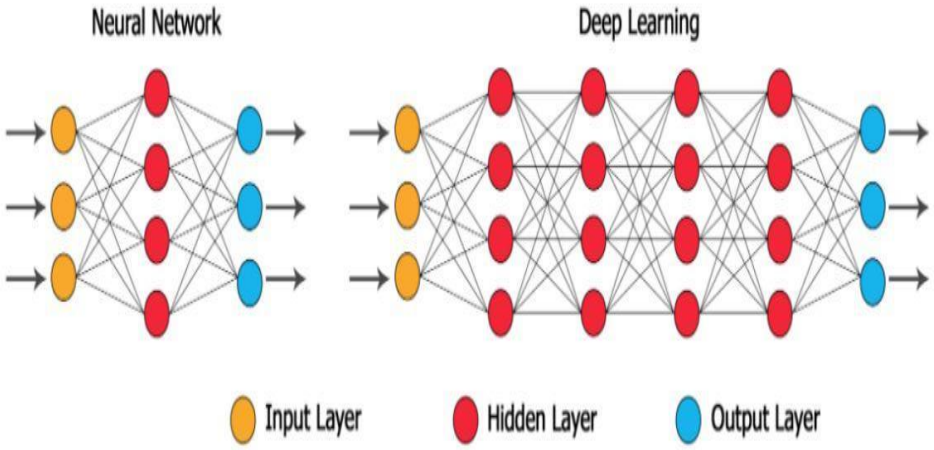


Fig 9.1 Neural network Structure with one and Multiple hidden layers

Notice that this network has multiple hidden layers in between the input and output layers.

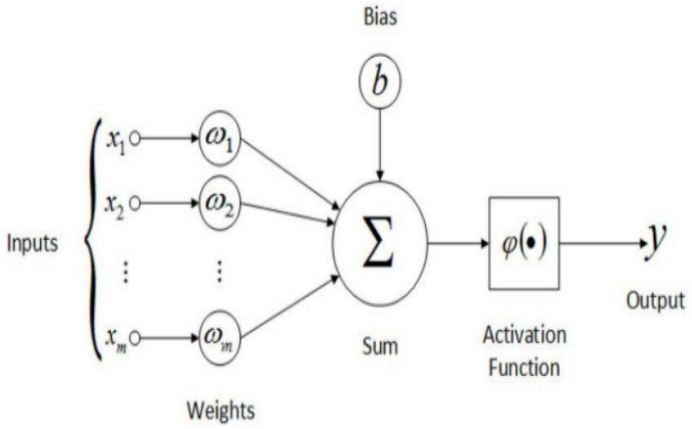


Fig 9.2 CNN Bias Function

9.3 CNN ARCHITECTURE

CNNs, a subset of deep ANNs, specialize in processing grid-like data, particularly images. These networks, functioning across multiple channels, excel in learning spatial patterns within data, forming spatial hierarchies. Primarily used for intricate image classification and object detection tasks, CNNs encompass three fundamental layers: convolution, pooling, and fully connected layers. Convolutions and pooling layers extract features, generating diverse feature maps that signify distinct characteristics in the data. The output of these feature maps typically proceeds to a fully connected layer for classification purposes.

The convolutional layer, pivotal in feature extraction, operates through the convolution process. It involves sliding a convolution kernel across the input data, performing matrix dot products at each position. The ensuing output passed through an activation function, constitutes individual pixels on the feature map, a procedure repeated **for every position to construct a comprehensive feature map.**

The objective function is what needs to be minimized or maximized during the training of a neural network. When it needs to be minimized it is referred to as the loss function. The loss function computes the error of the network by comparing the predicted output with the expected output.

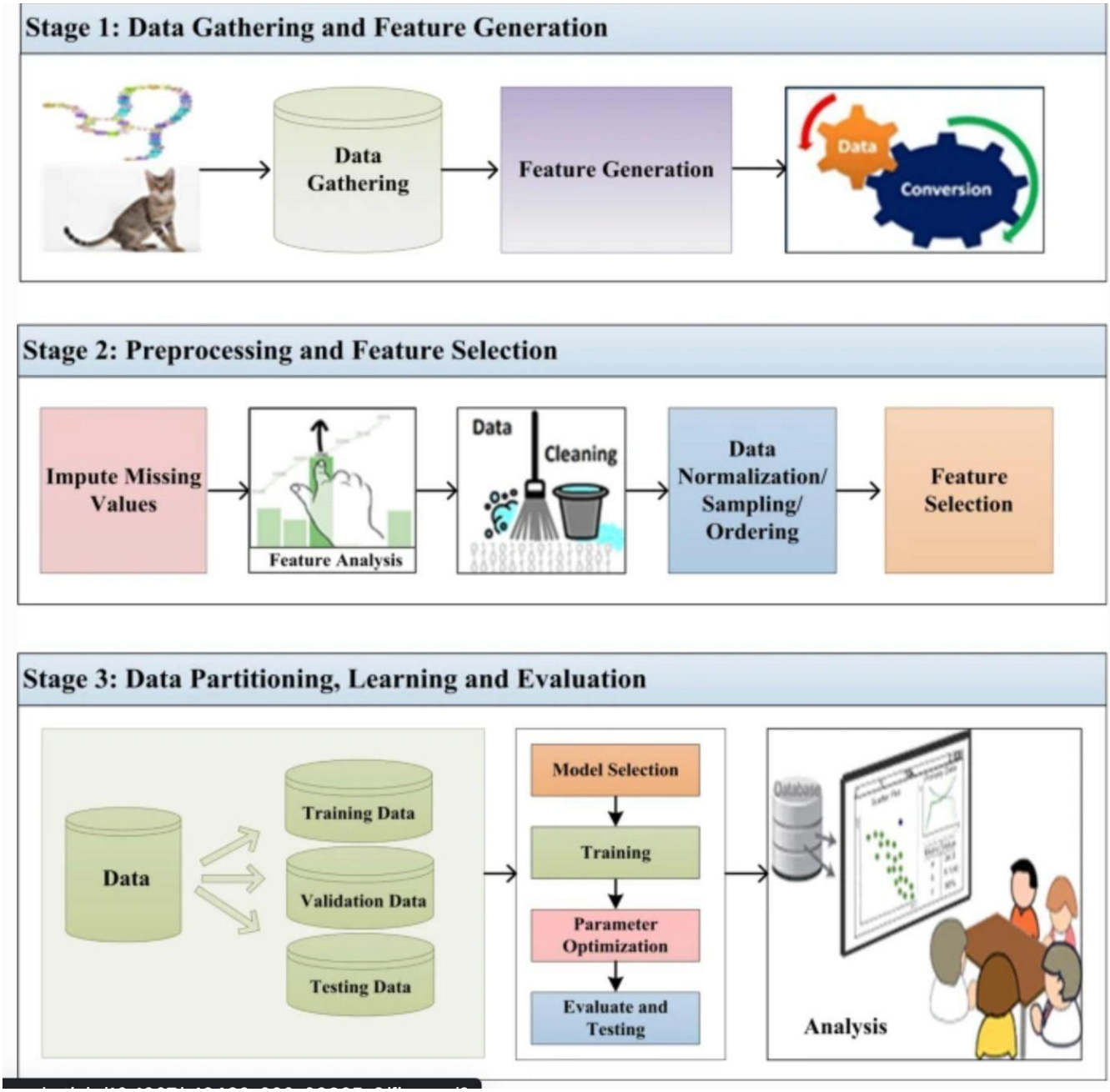


Fig 9.3 CNN Evaluation Process

The error is used by the optimizer to update the weights and biases of the network in order to improve the error. There are a variety of loss functions that can be used and should be chosen to match the problem type. For a regression problem, the common

choice is Mean Squared Error(MSE) defined as

$$MSE = \frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)^2$$

MSE computes the square difference between the network prediction ˆy and the expected output y, squares it, and averages it out of all the training samples N. MSE gets smaller the closer the predicted output is to the expected output. For the regression model, the loss function was set to Mean Squared Error (MSE) and the metric used was a root-mean-square error (RMSE). CNN trained for the purposes of regression. The task for this model was to predict the XY position for an image relative to the given floor plan. This model was trained for 100 epochs and achieved a validation root-mean-square error (RMSE) of 13.57. On training data, this model achieved an RMSE of 13.39. The training loss was reduced to 280.43, whereas the validation loss was reduced to 277.82.

9.4 RESULTS

Table 9.1 Simulation Results

	REGRESSION CNN
Optimizer	**RMSprop**
Loss Function	**Mean Squared Error (MSE)**
Learning Rate	**1e^-4**
Training Loss	**280.43**
Validation Loss	**277.82**
Training CCE	**NA**
Validation CCE	**NA**
Training RMSE	**13.39**
Validation RMSE	**13.57**

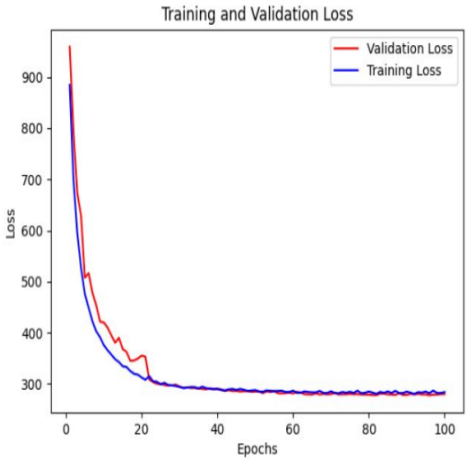


Fig 9.4 Inferred and validated loss plot for regression CNN for 100 epochs

For CNN training, we have used Adam Optimizer. The learning rate has been initialized to 0.001 and reduced by a factor of 10 after every 10 epochs when the loss does not decrease. The total training time is roughly 4 hours. For Ronin software and downloaded the trained model.

FLP positional errors are often 10 meters or more. We refine the aligned trajectory based on the floorplan via a convolutional neural network (CNN). An inertial trajectory can be as long as 45 minutes. We divide into overlapping shorter segments by 1) either taking the first 4 minutes or a segment whose axis-aligned bounding box does not fit inside 250× 250 pixel square of the floorplan, whichever comes first; and 2) repeating the process after removing the three-quarters of the segment to create overlaps. For each segment, we crop a 250×250 pixel region of a floorplan so that the bounding box of the trajectory is centered. Pixels in the overlapping regions have multiple predictions from the CNN model, where we take their weighted average. Suppose a trajectory segment spans a time interval of [0, T]. The weight of a pixel is given by a normal distribution N(T/2, T/4) based on its time.

CHAPTER 10

LONG-SHORT TERM MEMORY

10.1 INTRODUCTION

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) architecture designed to address the vanishing gradient problem associated with traditional RNNs. LSTMs were introduced by Hochreiter and Schmidhuber in 1997 and have since become a fundamental building block in the field of deep learning, particularly for tasks involving sequential data.

Here are key points about LSTMs:

1. Architecture:
   - LSTMs consist of memory cells and various gates (input, forget, output gates) that control the flow of information.
   - The memory cells allow LSTMs to capture and remember information over long sequences, making them well-suited for tasks with dependencies over time.
2. Memory Cells:
   - LSTMs use memory cells to store and access information. These cells can hold information for long durations without it diminishing during the training process.
3. Gates:
   - Input Gate: Regulates the flow of new information into the memory cell.
   - Forget Gate: Manages the removal or "forgetting" of irrelevant information from the memory cell.
   - Output Gate: Controls the information to be output based on the current Input and the memory content.
4. Vanishing Gradient Problem:
   - LSTMs are designed to mitigate the vanishing gradient problem, which can hinder the training of deep neural networks, especially those dealing with sequences.

- The gating mechanism allows LSTMs to selectively update and propagate gradients through time, enabling the training of deep networks on long sequences.

5. Applications:

- LSTMs are widely used in natural language processing tasks, such as language modelling, machine translation, and sentiment analysis.
- They are also employed in time series analysis, speech recognition, and other applications where capturing long-term dependencies is crucial.

6. Flexibility:

- LSTMs offer flexibility in handling different types of sequential data and have been extended and modified to suit specific task requirements.

In summary, LSTM networks are powerful tools for modelling sequential data, providing a solution to the challenges associated with training deep neural networks on tasks involving long-range dependencies. Their ability to capture and remember information over extended sequences makes them well-suited for a variety of real-world applications.

10.2 LSTM ALGORITHM

Modified data preparation:

* Added a new column for the floor number (z).
* Filtered the dataset for the desired single floor.
* Extracted the floor number as a separate feature.
* Dropped the 'z' column from the feature set.
* Considered four access points in the dataset.

Model architecture enhancements:

*Introduced an input layer for the floor number.
*Utilized embedding layers for categorical features.
* Concatenated the floor feature with the LSTM output for combined prediction.
* Incorporated an input layer for selecting the access point.
* Enabled the model to predict x and y coordinates based on the selected access point.

Adjusted training data:

* Split the data into training and testing sets specific to the floor of interest.
* Ensured that the training data matches the desired floor for accurate results.
* Allowed for the selection of the access point during training.

Modified model compilation:

* Customized the loss function to consider both location (x, y) and floor predictions.
* Combined mean squared error for location prediction with categorical cross-entropy for floor prediction.
* Assigned loss weights to balance the objectives

10.3 RESULTS

- Evaluated the model for both location (x, y) and floor predictions.
- Assessed accuracy for floor prediction and RMSE for location prediction.
- Input data included both location (x, y) and floor information, as well as the selected access point.
- Considered the floor number and the chosen access point when making predictions.

CHAPTER 11

CONCLUSION

11.1 CONCLUSION

This project proposes to develop a hardware device that plots the trajectory of the path onto the map/floor plan of that confined space which provides a comprehensive solution for accurate indoor positioning and path tracking by fusing WiFi, IMU, and floorplan data. The current system is designed as an offline process generating location history as opposed to a real-time Indoor GPS system. Our future work is the development of a real-time Indoor GPS system via multi-modal sensor fusion. We examined the applicability of a low-cost IMU in determining the absolute position of an oscillating object. We have successfully developed a low-cost IMU module and a filtering procedure, which can be used for various applications. We demonstrated that the low-cost IMU sensor is suitable as a standalone sensor in monitoring applications where absolute positioning of an oscillating object is required. For this purpose, we evaluated the procedure of ZPF filtering the low-cost IMU measurements. The ZPF proved to be a powerful tool for eliminating low-frequency errors (which accumulate enormous signal drifts) with zero phase shift. By using the ZPF, we successfully eliminated the drift caused by the signal noise of the accelerometer, as well as that caused by the third-order error accumulated by the bias error of the gyroscope. In the case of an oscillating object, the ZPF improved the accuracy of the object trajectory obtained by IMU measurements from a few hundred meters to a few centimeters. This method can provide a good and reliable low-cost monitoring system for oscillating objects in near real-time applications, where delayed results are sufficient.

11.2 FUTURE SCOPE

Global Pandemic – Our work would be cutting-edge technology, especially for times like COVID-19 and in case of future pandemics. Our model helps in tracking every person or patient in a hospital to know how exactly the virus spread among the crowd.

In another 20 years, autonomous robots will be everywhere around us. Ranging from healthcare to industries, we believe autonomous robots will take up most of the work providing leisure time for employees. As the reliability on robots increase, it becomes much more important to keep track of them in-case of a damage or malfunction. This project helps in tracking the robot down and solve the issue in no time. We have provided a tracking method for a single target and we aim to achieve tracking of multiple targets simultaneously, which solves the problem to track multiple robots performing operations . The future scope of indoor navigation using Inertial Measurement Units (IMU) and Wi-Fi is promising, with several potential advancements and applications. Some aspects to consider:

1. Improved Accuracy and Precision:
    - Advances in sensor technology and algorithms may lead to more accurate and precise indoor positioning. This could involve the integration of various sensors beyond IMUs and Wi-Fi, such as magnetic sensors, barometers, or visual sensors.
2. Integration with Augmented Reality (AR):
    - Combining indoor navigation with AR can enhance user experience. Users might receive real-time navigation instructions, information about surroundings, or virtual overlays of navigation paths.
3. Machine Learning and AI Enhancements:
    - Integration with machine learning algorithms could improve the system's ability to adapt and learn from user behavior, leading to personalized navigation solutions.

4. Multi-Sensor Fusion:

    - Integrating data from multiple sensors, such as IMUs, Wi-Fi, Bluetooth, and possibly even visual or LiDAR sensors, can provide a more robust and reliable indoor positioning system.

5. IoT Integration:

    - Indoor navigation systems could be integrated with the Internet of Things (IoT) devices to provide contextual information, such as room temperature, occupancy status, or equipment availability.

6. Smart Building Applications:

    - Indoor navigation can be extended to support smart building applications, including energy management, asset tracking, and facility maintenance.

7. Healthcare and Emergency Services:

    - In healthcare settings, indoor navigation can assist in tracking medical equipment, patients, and personnel. In emergency situations, such systems could help guide first responders to specific locations quickly.

8. Retail and Marketing Applications:

    - Indoor navigation systems can be utilized in retail environments for personalized shopping experiences, location-based promotions, and analytics on customer behavior.

9. Security and Access Control:

    - Integrating indoor navigation with access control systems can enhance security within buildings. For example, ensuring that individuals navigate authorized routes.

10. Open Standardization:

    - Establishing open standards for indoor navigation systems can encourage interoperability and widespread adoption across different platforms and devices.

11. Energy-Efficient Solutions:

    - Developing energy-efficient algorithms for indoor navigation systems, especially for battery-powered devices, can extend the operational life of devices.

12. Privacy Concerns:

- Addressing privacy concerns associated with indoor navigation technologies, especially when using Wi-Fi and other location-based data, will be crucial for widespread acceptance.

The future of indoor navigation using IMUs and Wi-Fi is likely to involve a combination of advancements in hardware, software, and integration with other technologies to create more seamless and sophisticated solutions for various industries and applications.

REFERENCES

1. Bai, Nan, Yuan Tian, Ye Liu, Zhengxi Yuan, Zhuoling Xiao, and Jun Zhou. (2020). "A high-precision and low-cost IMU-based indoor pedestrian positioning technique." IEEE Sensors Journal, 20(12), pp. 6821-6831.
2. Chhikara, Prateek, Rajkumar Tekchandani, Neeraj Kumar, Vinay Chamola, and Mohsen Guizani. (2020). "DCNN-GA: A deep neural net architecture for navigation of UAV in indoor environment." IEEE Internet of Things Journal, 8(6), pp. 4415-4423.
3. Feng, Daquan, Chunqi Wang, Chunlong He, Yuan Zhuang, and Xiang-Gen Xia. (2020). "Kalman-filter-based integration of IMU and UWB for high-accuracy indoor positioning and navigation." IEEE Internet of Things Journal, 7(4), pp. 2984-2993.
4. Foroughi, Farzin, Zonghai Chen, and Jikai Wang. (2021). "A CNN-based system for mobile robot navigation in indoor environments via visual localization with a small dataset." World Electric Vehicle Journal, 12(3), pp. 1-10.
5. Guo, Guangyi, Ruizhi Chen, Feng Ye, Zuoya Liu, Shihao Xu, Lixiong Huang, Zheng Li, and Long Qian. (2022). "A robust integration platform of Wi-Fi RTT, RSS signal, and MEMS-IMU for locating commercial smartphone indoors." IEEE Internet of Things Journal, 9(17), pp. 14824-14834.
6. Shao, Wenhua, Haiyong Luo, Fang Zhao, Yan Ma, Zhongliang Zhao, and Antonino Crivello. (2018). "Indoor positioning based on fingerprint-image and deep learning." IEEE Access, 6, pp. 16671-16680.
7. Wang, Peixiao, Hongen Wang, Hengcai Zhang, Feng Lu, and Sheng Wu. (2019). "A hybrid Markov and LSTM model for indoor location prediction." IEEE Access, 7, pp. 47322-47331.
8. Wu, C., Yang, Z., Liu, Y., & Xi, W. (2012). Will: Wireless indoor localization without site survey. IEEE Transactions on Parallel and Distributed Systems, 24(4), pp. 839-848.