INTERDEPARTMENTAL PROGRAM OF POSTGRADUATE STUDIES IN INFORMATION SYSTEMS

Master Thesis

LOCATION-BASED SERVICES IN UBIQUITOUS COMMUNICATION SYSTEMS

ANGELIDIS K. NIKOLAOS

2017
SUPERVISORS:

Roumeliotis Manos
Psannis Konstantinos
Acknowledgements

Special Thanks

While attending a lecture of Computer Networks, during my masters in Management Information Systems, I met the professor Manos Roumeliotis. He introduced us to professor Konstantinos Psannis. The latter was the one that urged me to make some research on **big data** and **signal filtering**. It seemed quite interesting and I wanted to make research in order to contribute in this topic. Professor Roumeliotis agreed and it was then, that I started my research. I want to thank all my professors and tutors for all help provided to me, as well as my friends and family for the support. Moreover, special thanks to professor Roumeliotis and Psannis for their support and help during my research and to professors Ksinogalos and Protogeros for their patience and time during their lectures about programming.
ABSTRACT

There are many approaches and models regarding the localization of a mobile terminal. However, each one of them has both advantages and disadvantages. Nowadays, the challenging issue is not only localization, but tracking as well. In this project, related work will be studied, concerning approaches of modifications and optimizations of current models as well as novel proposals, in order to provide a quality review of them, according to challenging criteria, such as accuracy, latency and efficiency. Furthermore, there is a theoretical proposal of a modification or even a potential merging of the studied related work. Specifically, a theoretical Hidden Markov Model-based model is proposed, modified with a novel map-aware model, to improve accuracy and latency. The development of the proposed scheme is considered to be future work, however, an experimental analysis is conducted to test the viability and efficiency of the map-aware modification.

KEYWORDS: LOCATION-BASED SERVICES, COMMUNICATION SYSTEMS, LOCALIZATION, TRACKING, EVALUATION, SIGNAL FILTERING, HIDDEN MARKOV MODEL, HMM
INDEX OF PICTURES

Picture 1: Experimental Testing Results ................................................................. - 29 -
Picture 2: Actual route ............................................................................................. - 46 -
Picture 3 HMM-based tracking route .................................................................... - 47 -
Picture 4: Proposed model’s estimate optimization ............................................... - 47 -
Picture 5: Experimental case (6→1) ....................................................................... - 51 -
INDEX OF SCREENSHOTS

Screenshot 1: GUI of Markov sim in eclipse ................................................................. - 30 -
Screenshot 2: Markov sim results GUI ......................................................................... - 30 -
INDEX OF FIGURES

Figure 1: Proposed Modified model ................................................................. - 45 -
Figure 2: Proposed model on [11] ................................................................. - 49 -
Figure 3: Initial State Estimate Modification Flow Chart ................................ - 49 -
Figure 4: Transition Probability Table Computing Modification Flow Chart .... - 49 -
Figure 5: Pattern Recognition Scheme Flow Chart ......................................... - 49 -
## INDEX OF TABLES

Table 1: Table of Evaluation ........................................................................................................ - 34 -  
Table 2: Noise Transition Probabilities ......................................................................................... - 51 -  
Table 3: Measurements .................................................................................................................. - 51 -  
Table 4: TPT\textsubscript{1} ............................................................................................................... - 52 -  
Table 5: TPT\textsubscript{2} .................................................................................................................. - 52 -  
Table 6: Results ............................................................................................................................. - 52 -
Contents

ACKNOWLEDGEMENTS ........................................................................................................ - 3 -
SPECIAL THANKS .............................................................................................................. - 3 -

ABSTRACT .......................................................................................................................... - 4 -
INDEX OF PICTURES ......................................................................................................... - 5 -
INDEX OF SCREENSHOTS ............................................................................................... - 6 -
INDEX OF FIGURES .......................................................................................................... - 7 -
INDEX OF TABLES ............................................................................................................ - 8 -

1. INTRODUCTION ............................................................................................................ - 11 -
   1.1. HISTORY OF LOCATION-BASED SERVICES ..................................................... - 11 -
   1.2. SIGNIFICANCE OF LOCATION-BASED SERVICES ......................................... - 12 -
   1.3. PROBLEM DEFINITION ..................................................................................... - 12 -
   1.4. RESEARCH OBJECTIVE .................................................................................... - 13 -
   1.5. PROJECT OUTLINE ............................................................................................ - 14 -

2. RELATED WORK ........................................................................................................... - 15 -
   1.6. INTRODUCTION .................................................................................................. - 15 -
   1.7. RELATED WORK ................................................................................................. - 15 -
      1) An Indoor Mobile Location Estimator in Mixed Line of Sight/Non-Line of Sight
         Environments Using Replacement Modified Hidden Markov Models and an Interacting
         Multiple Model ..................................................................................................... - 15 -
      2) Cooperative Positioning and Tracking in Disruption Tolerant Networks .......... - 16 -
      3) Distributed Localization and Tracking of Mobile Networks Including Non-cooperative
         Objects .................................................................................................................. - 18 -
      4) Refining Wi-Fi Based Indoor Localization with Li-Fi Assisted Model Calibration in
         Smart Buildings ................................................................................................... - 18 -
      5) Mobile Location with NLOS Identification and Mitigation Based on Modified Kalman
         Filtering ( NLOS id + Mitigation based on MKF ) .............................................. - 18 -
      6) Particle Filters (PFs) for State Estimate of Jump Markov Linear Systems .......... - 19 -
      7) Position and Velocity Tracking in Mobile Networks Using Particle and Kalman
         Filtering with Comparison .................................................................................. - 19 -
      8) A 2.4 GHz ISM RF and UWB hybrid RFID real time locating system for industrial
         enterprise Internet of Things ............................................................................... - 20 -
      9) Hybrid Unified Kalman Tracking Algorithms for Heterogeneous Wireless Location
         Systems ................................................................................................................... - 20 -
     10) Adaptive tracking of people and vehicles using mobile platforms (DTSMD) ....... - 21 -
     11) High-accuracy vehicle localization for autonomous warehousing..................... - 21 -
     12) Message of Interest: A Framework of Location-Aware Messaging for an Indoor
         Environment .......................................................................................................... - 22 -
     13) An Indoor Continuous Positioning Algorithm on the Move by Fusing Sensors and
         Wi-Fi on Smartphones .......................................................................................... - 23 -
     14) Vehicle Monitoring and Tracking System using GPS and GSM Technologies ... - 23 -
     15) Real-Time Sensor Data Integration in Vertical Transport Systems (AdInspect) - 24 -
     16) Emergency Service using GPS Tracking ......................................................... - 24 -
3. METHODOLOGY ................................................................................................ - 25 -
   1.8. INTRODUCTION ......................................................................................... - 25 -
   1.9. RESOURCE SELECTION ........................................................................ - 25 -
   1.10. EVALUATION AND PROPOSAL .............................................................. - 25 -
   1.11. EXPERIMENTAL ANALYSIS ................................................................. - 27 -
   1.11.1. INTRODUCTION ................................................................................. - 27 -
         1.11.2. Project Specification .................................................................. - 27 -
         1.11.2.1. Requirements Specification ................................................... - 27 -
         1.11.2.2. Software Design .................................................................... - 28 -
         1.11.2.3. Implementation ..................................................................... - 28 -
         1.11.2.4. Testing ..................................................................................... - 28 -
         1.11.2.5. Documentation ....................................................................... - 29 -
4. TABLE OF EVALUATION .................................................................................. - 32 -
   1.12. INTRODUCTION ...................................................................................... - 32 -
   1.13. CRITERIA ................................................................................................. - 32 -
   1.14. TABLE OF EVALUATION ...................................................................... - 34 -
   1.15. TABLE ANALYSIS .................................................................................. - 35 -
5. PROPOSED MODIFICATIONS AND EXPERIMENTAL ANALYSIS .............. - 41 -
   1.16. INTRODUCTION ...................................................................................... - 41 -
   1.17. HIDDEN MARKOV MODEL .................................................................. - 41 -
   1.18. MODIFICATIONS NEEDED .................................................................. - 42 -
   1.19. PROPOSAL ............................................................................................. - 42 -
   1.20. MODEL DEFINITION ........................................................................... - 43 -
         1.20.1. The Location variable .................................................................. - 44 -
   1.21. FLOW CHART ....................................................................................... - 44 -
   1.22. COMPARATIVE VISUALIZATION ........................................................ - 46 -
   1.23. MODEL SUMMARY ............................................................................. - 47 -
   1.24. SCALABILITY ....................................................................................... - 50 -
   1.25. EXPERIMENTAL RESULTS ............................................................... - 51 -
       1.25.1. RESULTS .................................................................................. - 52 -
       1.25.2. SUMMARY .............................................................................. - 53 -
6. FUTURE WORK – CONCLUSION ................................................................. - 54 -
   1.26. FUTURE WORK .................................................................................... - 54 -
   1.27. CONCLUSION ...................................................................................... - 54 -
7. REFERENCES ................................................................................................. - 56 -
APPENDIX A ...................................................................................................... - 59 -
   JAVA CODE FOR BASIC-HMM FILTER: .................................................. - 59 -
   JAVA CODE FOR HMM FILTER WITH MAP-AWARE TRANSITION PROBABILITY TABLE: .... - 85 -
1. Introduction

1.1. History of location-based services

A location-based services (LBS) are software-level services that “gather and deliver” location data [1, p. 1] to control features. By the “location” term we refer to the geographical term that is used to identify a place in the real world [1, p. 17]. Place can be interpreted as a spot or an area over the surface of earth. However, location term can be scaled to many subcategories regarding to the application of the services. As such LBS is an information service and has a number of uses in social networking today as information, in entertainment or security, which is accessible with mobile devices through the mobile network and which uses information on the geographical position of the mobile device [2, p. 176]. LBS can be used in a variety of contexts, such as health, object search, entertainment, work, personal life, tracking, etc. So the LBS term can include many services [3]. For example, services from person positioning and tracking to bank account transactions tracking, parcel/vehicle tracking services and mobile commerce (coupons or location-based advertising), as well as, personalized weather services and even location-based games [4]. LBS has become critical to many businesses and governments as well. Real insight can be provided by processing data related to a specific location where specific activities take place. Those location-related data and services can provide spatial patterns that can be leveraged to better understand patterns and relationships in those activities. [4] [1, pp. 1-7,35-49] Eventually, “Location-based Services” can be considered “a vehicle for telecommunication convergence”. [5, p. 2] [6, p. 12]

Nowadays, LBS are “part of virtually all control and policy systems” [7] which works in our computers and smart devices. They have evolved from simple synchronization based service models to authenticated and complex tools for implementing virtually any location based service model or facility [8, p. 59]. They are the ability to open and close specific data objects, based on the use of location and/or time as controls and triggers or even as part of a cryptographic key or hashing systems, and the data those objects provide access to. [6, p. 12] LBS are part of everything, from control systems to smart
weapons. They consist the most heavily used application-layer decision framework in computing today. [9, pp. 919-921]

1.2. Significance of location-based Services

The knowledge of the location of mobile node can give us some information, however, the knowledge of the location of the node over a time period can give us its moving trace. So another piece of information that LBS help us to obtain is tracking information. By tracking we refer to the computation of an object placement relative to a real world element or location over a time period. The challenge is to predict the tracking path by studying the node’s behaviour. The behaviour term contains too much information. Concerning this field of research, we can incorporate in the term information attributes like location, velocity, acceleration, tracking activity, time, distance and many other variables depending on which specific field we are researching (temperature, air/water pressure, etc) (figured by studying related work). Thus, in order to predict the tracking path, we can focus on accurately predicting the next location(s) of the observed subject. To achieve this, it will be a major help if we can get access to the area geographic and morphological information. Such knowledge can provide us normalization patterns to lessen errors like placing a driving inside the sea near the road since due to signal conflicts.

1.3. Problem definition

There are many approaches to figure out an optimization to location-based services. Some of them, that demonstrate significant results, use algorithms as well as applied combinations of filters such as Particle, Kalman or extended Kalman. Furthermore, another model that is used to solve such problems is the Hidden Markov Model. [10, pp. 354-380] Its use on location-based services can optimise tracking’s accuracy and latency, as well as used for machine learning to improve latency issues or even reveal abnormal activities, such as tracking paths that deviate safe area.

In this project, selected related works on localization and tracking have been studied, in order to provide a quality comparison of them, resulting to figuring a probable optimization of them or even a potential merging, to optimize Location-based
To begin with, however, we need to understand the problem that needs optimization.

Firstly, localization of an object can be as explained as the process of positioning it regarding the surface of the earth. Also, knowing the next locations results in tracking the object’s route. This is how localization and tracking are connected.

In continue, tracking is an issue that needs optimization. However, what do we mean with tracking? Tracking is the ability to know the precise location of something over time. In the term location, there are many attributes that can be included for security or privacy reasons. The major challenge is to collect and use data that can help use predict the following location of the mobile device or user. Velocity is a major attribute that can help us predict if the user is going to stop, turn, or even crash somewhere. This attribute should be studied in relevance to time, so that we know about the acceleration of the device. Additionally, localization is relevant to the terrain, thus terrain data should be utilized somehow to achieve much more efficient results.

To sum up, a modification is essential to cover up the velocity and acceleration parameters as well as to include terrain data in our computations, which cannot be solved my HMM as is. Last but not least we need many data points to have “continuous” information and avoid conflicts and diminish the NLOS error as well. Input and output data should be somehow parted to solve the big data issue and scale the model to cover huge networks as well as areas. Finally, an algorithm such as the Hidden Markov Model (HMM) will be needed to optimize the machine learning part with its pattern recognition applications.

1.4. Research Objective

In this project, a research will be conducted regarding related work of modifications, optimizations as well as novel approaches and models that have been proposed to cure this localization and tracking issue. The main contribution of this research will be a quality evaluation of those approaches and proposals, visualized in a table of evaluation. Moreover, a theoretical approach to a novel and robust model, concerning a merging and modification of related work studied, will be presented, in order to be developed as future work.
1.5. Project Outline

In section 1, there is the Introduction where a brief historical reference to LBS and their significance is made, followed by the problem definition and the objective of this project-research. In the next section related works that are used to conduct this researched are reviewed, in order to figure the main idea of each proposed approach. Section 3 is about the methodology that was used to complete the whole project. There is an analysis of the selection of the resources used, as well as the method used to evaluate them and complete the table of evaluation. In continue, there is a chapter concerning the experimental analysis of a quite simplified version of the map-aware model part of the projects proposal to test its viability for future development. In section 4, the criteria used to evaluate the related work are explained and the Table of Evaluation is being analyzed. Proposed modifications and the novel approach is explained in section 5, where there is a theoretical approach to the models architecture and a flow chart to visualize it. Moreover, the comparative objective of this novel approach is explained. An estimate scalability of the approach is presented, followed by the experimental analysis chapter, where the results are demonstrated and explained. In section 6 there is a chapter about future work and the conclusion. In the end of the project, there is the Appendix A section where the Java code that was used in experimental analysis is available.
2. Related Work

1.6. Introduction

In this section, there is some related work of approaches and proposals that I studied and reached to an outcome, regarding them, contributing in my approach. Apart from them, I used some other sources to study signal filtering and acquire information concerning the history and applications of LBS. In the following chapter there is a brief sum of the main idea from each paper.

1.7. Related Work

1) An Indoor Mobile Location Estimator in Mixed Line of Sight/Non-Line of Sight Environments Using Replacement Modified Hidden Markov Models and an Interacting Multiple Model

As seen in [11], the hidden Markov model (HMM) filter is a grid-based method that uses Bayesian techniques to estimate the location, because there is a location variable in the algorithm. Since we are dealing with both LOS and NLOS situations the MT has a dynamic state matrix, in which there is the transition probability. It is analyzed into Position Transition probability and sight Transition probability.

*Modified Hidden Markov Model (HMM)*

A problem with HMM after the location of the MT is fixed, there is no further modification because efficiency will be reduced. The modifications made stand for improving the localization precision, by placing weights on both Transition and Observation probabilities.
Replacement Modified Hidden Markov Model (RM-HMM)

This is an enhanced version of the previous modification, in which the probabilities for each state per calculation moment are according to the most accurate ones, which can be explained as the replacement of a latest transition probability with the previous one respectively.

Hidden Markov Model (HMM) + Interacting Multiple Model (IMM)

However, the previous is deficient because, in practice, the transition probability matrix should not be unique for variable motion of the MT. This approach to solve the problem is to create several separate transition probability models and derive results depending on the velocity distribution in each. The Interaction Multiple Model (IMM) estimator approach is a dynamic system with multiple switching probabilities used to select the proper transition probability model at the appropriate time, making use of the velocity. There are separate velocity distributions corresponding to a reasonable probability function. The Markov switching model is choosing the right model. The Cramér–Rao Lower Bound is used for the Localization Error in NLOS Environments.

2) Cooperative Positioning and Tracking in Disruption Tolerant Networks

According to [12], Localization in Disruption Tolerant Networks faces two major difficulties: the mobile node can only use sparse reference points to estimate its location, and the tracking server need to determine and predict movement trajectories with partial location information. To overcome these difficulties, PulseCounting and ProbTracking where proposed for positioning and tracking in Disruption Tolerant Networks. The proposed model is Markov-based. It is designed for a region with devices, Access Points (Aps) and an Infostation, which is a server connecting to the APs.
PulseCounting

This is a method for decentralized cooperative positioning in Disruption Tolerant Networks (DTNs). It consists six steps:

- Bootstrapping
- Step Counting
- Direction Mapping
- Trajectory Generation
- Location Estimation
- Refinement, which is divided into
  - Reference Point-based and
  - Mutual Refinement

This method to work needs the Infostation to be updated by the device.

ProbTracking

This is a probabilistic tracking method aiming to create the device’s trajectory from incomplete measurements, since the device is not real-time updating the Infostation.

Combination of cooperative self-localization (CS) and distributed object tracking (DT)

This approach is about a consistent combination of cooperative CS and DT for multiple mobile or static agents and objects as well. The results from ProbTracking are corrected and updated as long as the device communicates with the Infostation.
3) Distributed Localization and Tracking of Mobile Networks Including Non-cooperative Objects

To evaluate [13] we need to understand its contribution. This approach is for distributed localization and tracking of cooperative agents and non-cooperative objects in wireless networks, using measurements between agents and objects and between agents. This method, for the first time, provides a consistent combination of CS and DT in decentralized agent networks where the agents and objects may be mobile. To the best of our knowledge, it is the first method for simultaneous CS and DT in a dynamic setting. For a distributed operation and low complexity, we combine particle-based belief propagation with a consensus or gossip scheme.

4) Refining Wi-Fi Based Indoor Localization with Li-Fi Assisted Model Calibration in Smart Buildings

In continue, this approach (in [14]) takes advantage of smart buildings. Since Wi-Fi services are tending to be upgraded to Li-Fi, a more efficient Li-Fi localization method is proposed, that as well should be integrated to smart buildings’ resident-services. This approach is focusing on improving the QoS of the visitors as well. Tracking of them inside the facilities will help informing the smart building how to adjust the services, temperature, lights, etc. Despite being an indoor approach, it can cover huge buildings and Wi-Fi is quite scalable to going outdoors, communicating with other buildings as well.

5) Mobile Location with NLOS Identification and Mitigation Based on Modified Kalman Filtering

(NLOS id + Mitigation based on MKF)

In [15], a modified Kalman Filtering is proposed. To improve location accuracy, an NLOS identification and mitigation algorithm has been integrated to the Kalman
filtering algorithm. The performance gain has increased computer time, however, it can be recommended for real-time applications.

6) Particle Filters (PFs) for State Estimation of Jump Markov Linear Systems

According to paper [16], there are proposed online simulation-based algorithms to perform optimal filtering and fixed-lag smoothing of Jump Markov Linear Systems (JMLS). Those proposals are quite complex, however, they can be straightforwardly implemented on real-time applications, if we use parallel computer systems.

7) Position and Velocity Tracking in Mobile Networks Using Particle and Kalman Filtering with Comparison

As analysed in [17]:

**MLE APPROACH FOR MS LOCATION ESTIMATION**

A Maximum Likelihood Estimation (MLE) approach is presented to estimate the initial location of the Mobile Station (MS). The MLE approach employs a lognormal propagation channel model. The estimated location is used as the initial state of the EKF approach for the MS final location estimation. Thus, we have a hybrid EKF approach.

**EKF APPROACH FOR MS LOCATION AND VELOCITY ESTIMATION**

Moreover, a Particle filter (PF) approach for MS location and velocity estimation is proposed. The Unscented Particle filter (UPF) is implemented, however, it approximates the optional distribution by a Gaussian distribution using the Scaled Unscented Transformation (SUT) method. This SUT method accurately calculates the posterior covariance to the third order, whereas linearization methods such as the EKF rely on a first-order biased approximation.
As seen in [18], the Internet of Things (IoT) platform consists of a two-layer network hierarchy that employs heterogeneous networks is connected to the Internet through a standard air interface, such as Wi-Fi, GSM/GPRS and 3G. The sensing layer, which is the RFID system where sensor nodes (RFID tags) are coordinated by base stations (readers) through an ultra-low-power short-range wireless link. In the second layer, ad-hoc networks among RFID readers, working as the wireless sensor network, are utilized to perform data interaction.

The proposed system is a hybrid one and it consists of a server with a locating engine, the reader networks and the 2.4-GHz RF and UWB hybrid tags. The 2.4-GHz readers adopting the RSS method are employed to provide metre-level positioning accuracy and the UWB readers are deployed in the critical areas for fine position estimates. The UWB reader is basically a ranging receiver to detect ToA information of UWB pulses from tags. Therefore, the tag’s position is calculated by a TDoA algorithm, that is, different ToAs received by more than three different readers for location estimates. The hybrid tag consists of a 2.4-GHz transceiver and a UWB transmitter. This approach avoids a power-hungry UWB receiver and complex communication protocol, thus maximizing the battery life for extended maintenance cycles.

In this paper [19], hybrid unified Kalman tracking (HUKT) technique is proposed. It is an integrated algorithm for precise location tracking based on both time of arrival (TOA) and time difference of arrival (TDOA) measurements. The major design novelty of the HUKT scheme is that the nonlinear parameters within their respective TOA and TDOA-based location estimators are mathematically combined into a single state variable, which is to be updated within the Kalman filter. A new variable is incorporated as an additional state within the Kalman filtering formulation to consider the nonlinear behavior in the measurement update process. The
relationship between this new variable and the desired location estimate is applied in the state update process of the Kalman filter. For adjusting the weighting value between the TOA and TDOA measurements, three different designs of hybrid factor are proposed. The Hybrid factor is a Geometric Dilution of Precision (GDOP) based factor (GHF). The GDOP describes the geometry influence on location estimation accuracy. Therefore, the GDOP criterion that provides the relative distance information between the MS and BSs can be utilized to determine the hybrid factor that represents the weighting between the TOA and TDOA measurements. Furthermore, the proposed HUKT algorithm can directly be simplified into a unified KT (UKT) scheme for location tracking under the situation with only homogeneous signal sources. An UKT-TOA based scheme and a UKT-DTOA-based one are proposed.

10) Adaptive tracking of people and vehicles using mobile platforms (DTSMD)

In this paper [20], the design of a data-driven tracking system that integrates computational and measurement processes for optimized operation and reliability on mobile devices, is presented. That design is developed by integrating state-of-the-art acoustic-sensor-based tracking algorithms with principles of dynamic, data-driven application systems (DDDAS) and dataflow-based design and implementation of signal processing systems. The proposed tracking system design is called DDDAS-enabled Tracking System for Mobile Devices (DTSMD). DTSMD incorporates measurements throughout application operation to help the system adaptively select tracking algorithm configurations that are most strategic in terms of trade-offs among accuracy, energy efficiency, and real-time performance.

11) High-accuracy vehicle localization for autonomous warehousing

In this paper [21], there are combined several well established algorithms into a high-precision localization pipeline, capable of computing the pose of an autonomous forklift to sub-centimeter precision. The algorithms use only odometry information from wheel encoders and range readings from an on board laser scanner. The effectiveness of the proposed solution is evaluated by an extensive
experiment that lasted for several days, and was performed in a realistic industrial-like environment. During three days and 19 hours of total travel time, the vehicle has logged over eight kilometers, relying only on map information and its sensor readings, without a single failure or operator intervention. The localization experiments have been performed at the Euroimpianti manufacturing and testing facility in Schio (VI), Italy. It consists of an 80 m by 50 m main hall and a 20 m by 50 m storage area. The main hall is used for assembling and testing palletization lines and AGVs, with over 30 people working there on a typical day.

An overview of the localization algorithm proposed:
Adaptive Monte Carlo Localization (AMCL) Algorithm is used to fuse odometry data with laser range measurements to provide a robot’s pose estimate with a known covariance. In continue, this result is used as an estimate for the scan matching Iterative Closest Point (ICP) algorithm. Finally, he obtained result is used as the initial estimate in a discrete Fourier Transformation method, which returns the final result.

12) Message of Interest: A Framework of Location-Aware Messaging for an Indoor Environment

The goal of this paper [22] is to develop and deploy a location-aware messaging framework that is based on indoor location detection. This framework will only deliver messages when users are at work. The system uses beacon technologies to accurately determine the position of a user inside a building. The users, in return, communicate with the system using an application running on their smartphones. The framework delivers customized messages according to the user. This proposal is applicable on smart buildings, however. This approach concerns indoor as well as outdoor users. This framework, also, consists of OWNERS and VISITORS. An OWNER will adjust the message delivery settings and compose it. Each user has a specific signature. If the user is the BEACON-LISTENER, he will receive the message/notification when he will reach the appropriate range from the BEACON. However, if the message is not forwarded to him he will just be a VISITOR to the beacon and not notified for that. All users will be notified through some smartphone application on their mobile device.
13) An Indoor Continuous Positioning Algorithm on the Move by Fusing Sensors and Wi-Fi on Smartphones

In this paper [23], the proposal is referring to an indoor continuous positioning algorithm that is on the move, fusing sensors and Wi-Fi on smartphones. The main innovative points include:

→ an improved Wi-Fi positioning algorithm:

The algorithm has two phases, the OFFLINE training phase and the ONLINE positioning phase. However, to describe the properties of Wi-Fi signal on the move, there were introduced (proposed) two new parameters, “refresh rate” and “loss rate”. The aim of this proposal concerning those two parameters is that they can be deduced in a similar way.

→ the “Trust Chain Positioning Fusion” algorithm (TCPF) which is a positioning fusion algorithm. The TCPF algorithm is proposed to realize the “process-level” fusion of Wi-Fi and Pedestrians Dead Reckoning (PDR) positioning, including three parts: trusted point determination, trust state and positioning fusion algorithm. The PDR positioning module monitors the walking action of the user, then estimates their step length and orientation to estimate the displacement so that positioning estimation can be realized. An experiment is carried out for verification in a typical indoor environment, and the average positioning error on the move is 1.36 m, a decrease of 28.8% compared to an existing algorithm. The results show that the proposed algorithm can effectively reduce the influence caused by the unstable Wi-Fi signals, and improve the accuracy and stability of indoor continuous positioning on the move.

14) Vehicle Monitoring and Tracking System using GPS and GSM Technologies

In this paper [24], the proposal is about vehicle security. The proposed method is about alerting the vehicle’s owner. However, the alerts are not only concerning the position of the vehicle. Alerts about engine status, such as temperature, and speed limit when reaching areas such as schools, playgrounds, hospitals, etc are received as well. GSM circuit’s operation is to find the vehicle whenever it is kept in security mode and it communicates directly with the user’s smartphone via his
number, which is registered to the service, if the vehicle’s position has changed. Then a message can follow which will TURN OFF the vehicle’s engine immediately. The approach seems NOT TESTED for the ACCURACY, however, seems quite a security measure.


In this paper [25], the proposed method (AdInspect) aims to the monitoring of the QoS and the maintenance of Vertical Transport Systems, like an elevator or a lift car. The proposed solution is an Inertial Navigation System (INS), that will track the system’s movement profile. Information, such as velocity and vibration, will be obtained by sensors. However, INS is experiencing ISSUES when velocities aren’t low. It has been tested on a lift car system. It seems like other channels are needed to obtain accurate information for the acceleration and the gravity vectors, such as GPS information.

16) Emergency Service using GPS Tracking

In this paper [26], a proposal is presented regarding ambulance support to people in distress. The system’s architecture consists of two sides: user’s side (a smartphone owned by the user) and ambulance’s side (dedicated device/android smartphone, owned by the driver or fixed in the ambulance. The user will send an ambulance request and a server will process by forwarding it to the nearest vacant ambulance. When an ambulance accepts the request, a response will be sent to the user from the server and the ambulance will be on its way.
3. Methodology

1.8. Introduction

In this section, the steps followed to complete this project are presented. There is an analysis of the criteria that the selection of the resources was conducted. In the next chapter, the method and the steps I followed to evaluate them and a proposal of a probable merging and optimization are presented.

1.9. Resource selection

There are many approaches to optimize Localization and Tracking. A search was made for resources with relevant keywords such as location-based services, signal filtering, ubiquitous communication systems, Kalman filter, hidden Markov model filter, Particle filter, localization and tracking systems, LTE, fusing sensors and mobile tracking. The resources were filtered according to the technological level of the methods and approaches that they used and to the date of publication. The related work that has been selected are mostly up-to-date, due to the rapid advance of technology. Moreover, there are relevant books like [10] and [1] that I studied to complete the research, as well as resources like [5] and [8] that were used for information concerning the Introduction Section.

1.10. Evaluation and Proposal

After doing some research ways to cure localization and tracking issue I ended up with certain challenging criteria. According to them I evaluated the papers in a Low, Medium, High scale relatively to their contribution on optimising each one of the criteria. The evaluation of the related work is document-driven and based mostly on their results over the simulations and experimental analysis that conducted by their authors. However, there are certain criteria about sectors and problems that where not taken into account on some works, hence evaluation was completed by a personal
estimate according to the authors’ referred data. This research’s outcome and be seen and explained in the Table of evaluation Section.

After completing the Evaluation Table, proposal of [11] was considered to be more applicable and scalable to optimizations and modifications than the rest. [11] is an approach that is not based that much on hardware and sensors such those in [22], [23] and [24]. Moreover, the proposed model is working with existing infrastructure that can be found almost everywhere, in contrast to [14], which despite being such a novel and contributing approach Li-Fi and smart buildings are not that widespread yet. Hence, I studied more on the HMM filter and its algorithm and how to optimize it. The reason I chose to propose a HMM-based approach is based on its mechanics, scalability and applications. Also, after some more research on map-aware optimized tracking methods apart from GPS-systems, I reached to a novel and robust optimization, combining the proposed model in [11] with a map-aware filter and a pattern recognition scheme. To prove however that it should be researched I made an experimental analysis to test how the proposed map-aware filter will impact on localization and tracking. This modified merging seems a quite promising one, yet it will be considered as future work to develop and test it. Finally, an estimate scalability analysis is presented.
1.11. Experimental Analysis

1.11.1. Introduction

After completing the model’s scheme and flow chart, a testing was necessary to proceed to the presentation of the theoretical approach. In this chapter, it is explained how the experimental analysis was conducted. Java code (available in Appendix A) was implemented and measurements were used to simulate the results in Eclipse. The Java project is explained in the Project Specification (according to [27]) chapter.

1.11.2. Project Specification

In this chapter there is an analysis of the Java Project (as guided in [27]) used to simulate an experimental analysis. There are 5 stages in software development [27]. Analysing the requirements, designing the software, implementing it, testing it and writing a documentation about it.

1.11.2.1. Requirements Specification

To prove the proposed model’s viability, it was tested in a simplified version. The proposal of map-aware model will be designed to be implemented on a 2nd-grade HMM with and adaptive Transition Probability Table, however, to proceed in this models’ development, an experimental analysis was conducted with a comparative simulation of a basic HMM and a modified one with the map-aware logic. Summing up, a HMM Java code was needed to conduct an experimental analysis and if possible a HMM with the Viterbi optimization. The inputs would be 2 Probability Tables[9x9], a Transition and a Noise one and the experiment’s measurements, both actual and ones distorted with noise. As outputs, essential would be to get the deviation results from the actual path and the calculated one, as well as the time spent in computing.
1.11.2.2. Software Design

Due to my lack of expertise in programming (one year of experience in Java), designing the software from scratch was almost impossible for me. Hence, I asked from professor Psannis a HMM filter Java code to understand it, and adjust it in the experiment. The code I received was a basic HMM filtering code with the ability of using the Viterbi optimization. The code however was quite hardcoded and there was a usage of GUI libraries (like JFrame) which were quite hard to handle considering my lack of experience. After studying the code and understanding its logic, I was able to proceed on refactoring it for my experimental case.

1.11.2.3. Implementation

I hardcoded the Tables as seen in section 5.10 and provided the measurements as a .txt file. The noise Probability Table was designed by a random function and the Transition Probability Tables of the basic HMM and the map-aware filtered where hardcoded according to the HMM logic and map-aware filtering respectively. The map-aware filtering principle followed when constructing the Transition Probability Table is that the transition probability to or from a not viable state (obstacle or wall) will is set to be 0.

1.11.2.4. Testing

Testing was conducted as a simulation in Eclipse. The experimental case is concerning the tracking of a subject that is already moving and continue to a right turn. There are walls to prevent the subject from taking any other locations than the measured ones and this experiment is an example of a case that usually provide lots of outliers in our measurements, often due to noise and NLOS error. The Actual measurements were stable and presenting the actual tracking route. The noise measurements used the the results obtained are displayed in the following scheme:
### 1.11.2.5. Documentation

In this Java project objective is to simulate an experimental tracking case to test the map-aware filtering viability. The system is designed with a user GUI (screenshot 1) after running the project from Eclipse. A window is opening and there is a drop down menu.

→ The system input is a .txt document with the measurements in 2 columns. In the first column there are the actual measurements and in the second there are the ones distorted with noise. More over there is a Noise Transition Probability table hardcoded in the system.

→ the Noise Transition Probability Table can be altered from the user after running loading the measurements.

→ after setting the Noise Transition Probability Table, the user can alter the already hardcoded Transition Probability Table as well.

→ The user has the option to run the filter and obtain results with or without the Viterbi optimization.

<table>
<thead>
<tr>
<th>actual measurements</th>
<th>noise measurements</th>
<th>deviation</th>
<th>time elapsed</th>
<th>initial deviation</th>
<th>V_HMM</th>
<th>ma_HMM</th>
<th>V_ma-HHM</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>2.75</td>
<td>1.5</td>
<td>0.5</td>
<td>1</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>3</td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>3</td>
<td>1.75</td>
<td>1.5</td>
<td>0</td>
<td>2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>2.75</td>
<td>2.5</td>
<td>1.25</td>
<td>2.5</td>
<td>1.25</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>5</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>6</td>
<td>6</td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>2.25</td>
<td>1.5</td>
<td>0.5</td>
<td>1.5</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td>5</td>
<td>8</td>
<td></td>
</tr>
</tbody>
</table>

*Picture 1: Experimental Testing Results*
The functionality of this project is to provide as outputs (screenshot 2):

- measurements deviation
- runtime
The simulation’s objective is to reach into a conclusion concerning the viability of the map-aware filtering on the Transition Probability Table in comparison with the existing HMM. The case was tested with multiple measurements in order to achieve more reliable results. The code can be seen in Appendix A.
4. Table of Evaluation

1.12. Introduction

In this section, there is a personal evaluation of the papers mentioned before. The evaluation is based on some criteria that are being analysed in its first chapter, followed by the chapter of the table of evaluation where there is a comparative preview of the papers, referred in the previous section. The last chapter, is about explaining the evaluation table.

1.13. Criteria

The evaluation was based on some criteria of great importance. These are:

- Communication Channels [CC] (3G, LTE, Wi-Fi) [Net]
- Accuracy [Acc]
- Latency [Lat]
- Real-Time / non Real-Time [RT]
- Power Consumption [PC]
- Network Size [NS]
- Complexity [C]
- Scalability [Sc]
- Cost [Cost]
- Indoor/Outdoor [I/O]

The evaluation scale is L (Low), M (Medium) and H (High):

From L to H is used to describe weather the proposal is fulfilling the criteria. Thus, the scale can be rephrased as “how much the proposal helps in solving each problem”.

In continue, there is a brief explanation of the criteria used, followed by the evaluation table:

#Communication Channels is referring to weather the system refers to a network with 2G/3G/LTE signals or it incorporates Wi-Fi, Bluetooth, etc communications as well.

#Accuracy refers to the precision of the location’s estimation of each proposal.
#Latency refers to the delay until the desired results. Here the scale goes vice-versa, meaning H stands for low latency.

#Real-Time/non Real-Time seems as a Y/N scaled criteria. However, according to the evaluation scale of this paper, L stands for non-Real-Time, M for potentially real-time and H for Real-Time.

#Power Consumption is about the energy-saving prospective that the proposal has. The scale goes vice-versa for this criteria as well.

#Network Size concerns the size of the area where the proposal was implemented/tested. Some proposals however have been tested only at simulation-level. They have been evaluated from L to M according to the simulations’ scenario.

#Complexity refers to the algorithms of the proposal. The scale is also vice-versa. The latter is related to the #Power Consumption one, cause usually complex algorithms are significantly power hungry, hence they need some serious optimization to become efficient.

#Scalability concerns the proposal’s ability to be implemented on different scenarios.

#Cost refers to the funding the will be needed for the implementation of the proposal in real-life.

#I/O stands for weather the proposal is for Indoor (I) application or Outdoor (O) application. However, the evaluation scale will be L for Indoor, M for Outdoor and finally H if the proposal can be used for both.
<table>
<thead>
<tr>
<th>Method</th>
<th>CC</th>
<th>Acc</th>
<th>Lat</th>
<th>RT</th>
<th>PC</th>
<th>NS</th>
<th>C</th>
<th>Sc</th>
<th>Cost</th>
<th>I/O</th>
</tr>
</thead>
<tbody>
<tr>
<td>Li-Fi + Smart Buildings [14]</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td>PF for state estimation of JMLs [16]</td>
<td>L</td>
<td>H</td>
<td>M</td>
<td>H</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>EKF with MLE [17]</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>UPF with SUT [17]</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>L</td>
</tr>
<tr>
<td>Hybrid RFID (IoT) [18]</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>HUKT [19]</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td>AMCL+ICP+FT [21]</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>Message of Interest [22]</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>L</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>fusing sensors &amp; Wi-Fi on smartphones [23]</td>
<td>L</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>L</td>
</tr>
<tr>
<td>Vehicle security alerts [24]</td>
<td>L</td>
<td>L</td>
<td>H</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>L</td>
<td>M</td>
<td>M</td>
<td>M</td>
</tr>
<tr>
<td>Adinspect [25]</td>
<td>L</td>
<td>M</td>
<td>H</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>M</td>
<td>H</td>
<td>H</td>
</tr>
<tr>
<td>Emergency Service Using GPS [26]</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>H</td>
<td>M</td>
<td>M</td>
<td>L</td>
<td>L</td>
<td>L</td>
<td>H</td>
</tr>
</tbody>
</table>

Table 1: Table of Evaluation
1.15. Table Analysis

The L/M/H evaluation seems quite “easy”, however, there is more to point than just this table. Some of those proposals didn’t take under consideration all of the criteria, so the evaluation was a bit harsh regarding those factors.

To begin with, evaluating [11]:

The main goal of that proposal, as mentioned before, is to eliminate NLOS localization errors in unfamiliar indoor environments. Two algorithms to improve the Hidden Markov Model are proposed and a combination of the Hidden Markov Model with an Interacting model as well. Small-scale indoor environments are such, where there are quite many obstacles between the terminal and the access point. The communication channels used seems to be just GSM signal, however, it can be quite scalable to any indoor signal concerning that the Access Points can receive almost anything. There was small latency so it could be considered to be implemented on real-time applications. The accuracy was quite high so the main goal seems fulfilled. There were no such references concerning power consumption, however, the algorithms used are not so power hungry. The testing was conducted with simulation, so the network size cannot be evaluated legitimately. The cost of the implementation cannot be calculated since it was done on sim, but an estimation can place it on the medium standards. Finally, the complexity was quite an issue. In general terms the combination was a quite complex, however the optimizations made on the models where quite simple to understand.

In continue, evaluating [12]:

In this paper, the proposal aimed to positioning and tracking in Disruption Tolerant Networks (DTNs). An experiment was conducted on android devices in a Campus, therefore the network size can be considered quite adequate. The communication channels used where GMS/LTE and GPS. The power consumption of this proposal was considered between online radio listening and video watching, so it can be considered quite low, if we are NOT talking about android devices. In this case, since there are some power consumption optimized devices and others extremely power hungry android smartphones as well, energy consumption cannot be considered that low. The proposal is not real-time because the mobile terminal must update the “infostation” to work, thus the latency is quite high. Regarding accuracy, it deviates 9m to the GPS so it’s not quite accurate. Future work can make this proposal more scalable and increase the
communication channels used. Finally, he proposal was tested outdoors and the cost of implementing was not high.

Wi-Fi and Bluetooth can be integrated to the system, in order to have more channels to update the “infostation” and so the proposal can go real-time.

→ Evaluation of [13]:

In this paper, the proposal is a Bayesian method for distributed sequential localization of mobile networks composed of cooperative agents as well as non-cooperative objects. In this method there is a combination of cooperative self-localization (CS) and distributed tracking (DT). By combining particle-based belief propagation with a consensus or gossip scheme, low complexity is achieved. Moreover, high localization accuracy is achieved through a probabilistic information transfer between the CS and DT parts of the underlying factor graph, yet not that high compared to other proposals studied. Simulation results demonstrate the efficiency of such a combination, and very good scaling properties with respect to the numbers of agents and objects. The proposal can cover many communication channels and the network size of the sim was more than adequate. The algorithm is not complex and its application seems not that power hungry as well. No reference is made concerning the cost of the implementation, however, it seems of average expense. Finally, even though this is a real-time applicable algorithm, for both indoor as well as outdoor applications, the latency issue is quite major and needs to be fixed.

→ Evaluating [14]:

This paper presents a new Wi-Fi based indoor localization technique, that achieves significantly improvement of indoor positioning accuracy with the help of Li-Fi assisted coefficient calibration. Therefore, the feasibility of indoor hybrid Wi-Fi and Li-Fi based positioning technique is investigated. The Communication Channels used are just Wi-Fi and Li-Fi, however Li-Fi is not that “popular” yet. The network size is limited to the inside of the building, however, buildings can be huge. The accuracy of the proposed method is quite high and latency seems to have been diminished. It is a real-time approach and not that complex. It can be scalable, since Wi-Fi infrastructure and Li-Fi as well can be expanded outside of the involved buildings. Furthermore, since smart buildings are involved, infrastructure must exist in them, thus the cost is somehow dropping down. Finally, the power consumption of a mobile terminal using Wi-Fi is quite optimized. The back end, however, is yet to be evaluated.
Evaluation of [15]:
In this paper, the proposal is quite NOT complex and supports real-time applications. The accuracy is quite high, yet the latency numbers need fixing. The communication channels that where supported are not all that are used in everyday life, however, it is quite scalable to covering them. Low complexity of the proposal renders it not that power hungry. The cost of applying the proposal can be estimated to be between average boundaries, however, there is no such estimation or reference from the authors. Last but not least, the network size cannot be evaluated as adequate since the tests were conducted only as simulations and not in a real environment. Finally, the proposal can be implemented on indoor as well as outdoor applications.

Evaluation of [16]:
This proposal is an approach tested with a simulation. Therefore, the communication channels were used in a theoretical level and the network size cannot be evaluated. In spite of that, the small latency and high accuracy measurements, rendered the approach reliable for real-time applications. The complexity of the algorithms used seems quite high, hence, this approach is rather a power hungry one. Future work as well as the approach’s description implies scalability, however, no reference is made concerning the cost of its application. Finally, this approach seems able to be applied for both outdoor and indoor projects as well.

Evaluation of [17]:
In this paper, there is a presentation of two algorithms. The one is an extended Kalman filter based modified by using the maximum likelihood estimate as an initial state, and the other is an unscented Particle filter based modified by approximating the optional distribution by a Gaussian distribution using a scaled unscented transformation method. Both of them provide high accuracy and low latency results. Their implementation is quite power hungry and the algorithms are quite complex. However, it is cost effective and the proposals are scalable as well. The results are taken from a simulation where not many communication channels where used, since it refers only a cellular network, and the network size from the simulation is quite inadequate. Finally, there is no reference concerning its use on indoor or outdoor application.
Evaluating [18]:
This approach integrates lots of hardware. In this paper, there is a real-time locating system that is applied to Internet of Things (IoT). The system is promising concerning scalability. Furthermore, its extremely high accuracy and low latency are making the approach able to be implemented on the real-time applications, both on indoor as well as outdoor environments. Moreover, since there are many kinds of sensors used, many communication channels can be utilized, however this expansion of communication channels is left for future work. The network size was rather small, however, this issue is scalable. Last but not least, implementing this approach is very cost effective and affordable. Finally, the power consumption is very low and complexity of the algorithms used is accessible.

Evaluating of [19]:
In this paper, the approach is about a combination of the least square methods and the Hybrid Unified Kalman Tracking (HUKT) algorithms integrated with an algorithm based on time of arrival (TOA) and time difference of arrival (TDOA) to achieve high precision tracking. However, neither accuracy nor latency results are as expected, yet, according to the results, the approach has high potentials to be implemented as part of real-time applications. Furthermore, many communication channels are utilized, and the network size is big enough. The approach is quite adequate for indoor environments. Last but not least, there is not that much scalability as is, since more optimization is required to achieve better results. The power consumption regarding the complex algorithms that are used is moderate. Finally, implementation cost seems to be of average.

Evaluating [20]:
In this paper, the presented tracking system is a “Dynamic Data-Driven Application System (DDDAS)“-enabled Tracking System for mobile devices (DTSMD). Firstly, the communication channels that are used to collect data cover a great range and bandwidth, concerning the sensors’ scalability. Also, the accuracy is quite high, yet latency not that low. Furthermore, this approach can be considered as part of real-time applications. Moreover, the network size of the tested area was more adequate. In continue, despite the fact that the algorithms are quite complex, the implementation is energy efficient as well as cost effective, hence, considering the scalability potentials,
this approach is quite efficient. Finally, this proposal can be applied in both indoor and outdoor territories.

→ Evaluation of [21]:
This proposal is offering high precision localization pipeline which consist of several algorithms. There are several wireless communication channels used to collect data from sensors. The tested area was not that big, yet adequate to get trustworthy measurements and results. The algorithm stack is quite complex, however, high accuracy with low latency were achieved. Moreover, considering the latency measurements, this proposal is capable of being part of a real-time application. Also, in spite of its complexity, the proposed approach is quite cost effective. Furthermore, it can be applied both indoors and outdoors as well. Finally, scalability is an issue that was not discussed.

→ Evaluation of [22]:
In this approach, the aim is to alert users about activities or interests. Communication channel used in order to position alert users is Wi-Fi. In this proposal, however, we make use of beacon technology. Beacons communicate and distribute data via Wi-Fi, so, since IoT (Internet of Things) is involved in this approach, communication channels and testing are quite scalable. Yet, the proposal is limited for indoor environments. Low latency and high accuracy make it a real-time proposal. The network size is quite large, since more than one building can be involved in this application. In spite of being quite complex, it seems quite cost efficient.

→ Evaluation of [23]:
The main communication channel used in this approach is Wi-Fi. The application is concerning indoor facilities. It is quite real-time with high accuracy and low latency. However, it needs to be tested in bigger areas with more conflicts and noise, since the size of the tested area was quite small. No complexity analysis was presented, however the data collection and the optimized algorithm seem to be quite complex. As far as power consumption, the application is not that power hungry nor energy saving as well. The proposal is certainly scalable to more communication channels and larger facilities. The cost was not discussed.

→ Evaluation of [24]:
This proposal is limited in GPS and GSM signals, so, the communication channels used are not that many and the network size of the tested area was not that huge. However,
despite not having been test for its accuracy, the system is responding promptly, thus 
latency seems not an issue. Furthermore, the main purpose is security so the main goal 
is to alert the user, weather the vehicle is moving and not where it is heading to. This 
proposal is definitely a real-time one. Moreover, power consumption is quite low. No 
complexity analysis is presented to be evaluated nor concerning the cost. As far as 
scalability, this approach is quite scalable to be used in large areas and to be able to 
communicate with the terminal in more communication channels, such as Wi-Fi and Bluetooth.

\[ \text{Evaluating [25]}: \]
The proposed method was tested to track the movement profile of a lift car. There were 
some serious obstacles (noise for example) concerning data gathering in high velocities. 
To make the long story sort, the communication channels used were limited to WSN, 
such as accelerometer and gyroscope, and GPS signals. The accuracy was not that high, 
however the latency was very low. The proposal is definitely real-time applicable and 
has medium power consumption. The cost of implementing is not that much and 
considering that the aim is to achieve in-time maintenance of those means of vertical 
transport, it can be rendered as cost-effective as well. Despite being tested on a lift car, 
it can be used in almost any means of vertical transport, indoors or outdoors.

\[ \text{Evaluation of [26]}: \]
In this paper the model described is about tracking the nearest free ambulance in the 
area using global positioning system and bringing it to the person in distress, hence, this 
is a real-time applicable proposal. The communication channels used are GPS system 
and cellular level. Accuracy is high, since triangulation using satellite is being applied. 
Latency is not that high, however it depends on the reaction of the people in the 
ambulance as well. In spite of the fact that this is a real-time application targeting 
smartphones, no reference is made weather the app is power hungry or any power 
optimization technique is proposed. The targeting network size should be huge (all 
around the world), however this would demand high costs for the servers so that they 
can tackle the data bulk. The cost of implementing this proposal seems to be relative to 
the cost of the hardware used (GPS devices, server(s)). Last but not least, the project is 
planned to improve its user interface adding features, however no reference concerning 
its scalability nor utilizing more communication channels. Finally, this application can 
cover indoor as well as outdoor environments.
5. Proposed Modifications and Experimental Analysis

1.16. Introduction

Approach [11] seems to have almost the most satisfactory results and the Interacting Multiple Model (IMM) modification is something quite helpful. A serious issue is that the approach has not been tested in the field nor expanded for outdoor environments, limiting the results to simulation obtained only. In [11], it is proposed a 2\textsuperscript{nd}– grade HMM with an adaptive Transition Probability Table, according a function that takes into account the corner and the distance of the previous states relatively to the next one. The “replacement modified” optimization is quite contributing combined with the IMM.

1.17. Hidden Markov Model

Let a Hidden Markov Model-based filter [28, pp. 1-8] be used to process the localization data. This is an algorithmic model that uses state estimates in order to provide a transition probability table and figure the final location. The way probabilities as well as state estimates are calculated, is being modified or filtered to achieve better and more efficient results. Hence, the less and more accurate possible states the more optimal results will be. To make the long story short, it seems there are two levels that need optimizations: calculation of the state estimates, both hidden and observations as well, and computing the Transition Probability Table. Moreover, the hidden Markov model (HMM) is a really helpful tool to proceed on concerning geolocation-based services, since, it can be applied for pattern recognition (in [12] and [13]) to continuously update, upgrading as well, the training part of the applications (machine learning) and with some normalization and modifications on the Transition Probabilities Table, much more accurate results with diminished latency can be achieved.
1.18. Modifications Needed

The already proposed IMM should be modified to become scalable for indoor as well as outdoor environments and real-time applications. Latency is a parameter that needs to be lessened and a constant training algorithm should be established to search for any tracking paths that may follow specific patterns. Furthermore, no area information is being used. Such data seems quite essential for the optimization of the localization, since they can alter the way of tracking in a more “logical” way to minimize the number of computational errors based on transition probabilities.

1.19. Proposal

To begin with, having more accurate measurements, is meaning more data resources. Regarding indoor environments, the is quite much conflict between the signals due to the congestion of too many mobile devices inside an area. Bluetooth receivers, signal MIMO antennas and Wi-Fi access points should be considered significantly as DATA access points, since the mobile cellular signal by itself will not be that accurate, considering the NLOS error under such circumstances. Moreover, each device has a signature. This can indicate a “user” of the tracking system. The system should have layers. Each layer should represent a different domain of the area where it is applied. Every “user” changing a domain will have his tracking information connected to him and until “logging out” of the system, his real-time tracking information will be available. The challenge might be to combine mobile device signature with identity recognition, however, this will imply serious privacy issues, such as somehow searching by the device’s signature the owner’s account or google/apple account login to access the regional Wi-Fi, for example, or even gain access to security or traffic cameras.

Finally, to acquire better accuracy results as well as to achieve a tracking normalization, “knowledge of the area” can be considered vital information. Geomorphological information will help to prevent tracking paths that cannot logically exist or, in case such tracking activity may exist, to identify possible accidents. In addition, the knowing how an area is constructed, hence, being able to identify obstacles or shortcuts during a tracking session, will improve the prediction of the tracking route, meaning less accuracy errors. This would be a model that will use some parameters and
and constants which will interact with the way that the Transition Probabilities table will be created and optimize the results. A pattern recognition model will be obtaining the tracking results and this will provide us constantly updated normalization. Summing up, this proposal is presenting a theoretical model in which a combination is implied of the already improved HMM with the IMM of [11] improved additionally with a novel Map-aware model which can be additionally trained with robust pattern recognition scheme in order to achieve normalization.

1.20. Model Definition

In this section, a mathematical definition of the proposed model will be presented in a theoretical level. To begin with, the Bayesian-HMM is parted of the Initial states, and there are the hidden states (x(t)) and the observations (y(t)); assuming N hidden states, V distinct observation symbols for each state and T observations. Also, there is the Transition Probability Table (TPT), which concerns the probabilities of switching from a hidden state to another. However, there are transition probabilities from a hidden state to another (φ_{i=1..N,j=1..V}), as well as that a hidden state observes an observation (θ_{i=1..N,j=1..N}); of course both Φ and Θ must sum to 1. Both states, hidden as well as observations, are relevant to time and their values do follow a distribution. \( X_t \sim D_n(\beta) \) and \( Y_t \sim D_v(\alpha) \) (\( D_n(\alpha) \) is not representing a specific distribution) are relevant to the hyperparameters \( \beta \) and \( \alpha \) respectively. Thus, changing the hyperparameters’ value means interfering with the entropy of the probabilities. Specifically, to alter the observations distribution’s entropy, we need to modify the observation symbols’ distribution (Θ). Moreover, to achieve tracking normalization, the hyperparameter’s \( \beta \) value, that controls the density of the Transition Probability Table, needs to be significantly lowered below 1, having as a result the next states to be highly predictable. However, that value will be changing according the feedback of the Pattern Recognition Scheme and terrain data, processed by the map-aware model, combined with the results from the IMM. Furthermore, the map-aware-model will provide data that will nullify transition probabilities to a not feasible, according to the terrain, state. The idea of this model is to alter the TPT that was researched in [11], using data from the map-aware model and nullifying some probabilities that refer to states that are not viable. In order to make this process efficient, the map-aware model will be held responsible for
filtering the hidden and observation states as well. If a state is not viable, it will not be considered, hence less states will produce more accurate and faster results. Last but not least, the model will try to be simplified, filtering the initial state ($\pi=P(x_0)$) with the map-aware model and applying the Maximum Likelihood Estimate, in case there is noise that cannot track an exact position. Finally, regarding noise filtering, the Gaussian-based model will be followed.

1.20.1. The Location variable

The TPT, as researched in [11], will be divided in two parts; Position Transition Probability (PTP) and Sight Transition Probability (STP). When not in sight, the map-aware model, will be an improvement, limiting the possible position transition states. In addition to this, the IMM, by processing the velocity data, will result to a more normalized tracking path, than the one without map-aware model filtering. The velocity processing is quite simple, since the states are accelerating and decelerating, however, the tracking information that will be provided needs should be filtered, relevantly to the terrain information. For example, if the velocity is quite high and the subject is reaching an obstacle that might crash on, the map aware model will proceed on a normalized tracking according to the terrain, but the IMM will assume the location according to the high speed. In this case there are two different position-state families. This is a conflict that needs to be optimized. The map-aware model will return the dismissed position state as a possible state in the hidden states distribution for the next computing to take into account the possibility that the subject has deviated from the the terrain-aware route.

1.21. Flow Chart

In this section there will be presented a flow chart. This represents the way the HMM will be modified to achieve the desired optimization. The Hidden Markov Model, already optimized and with the IMM, will have another intervention before the output of the Transition Probability Table, elimination abnormal paths, minimizing the transition probabilities to them. After the final state are calculated, data from the existing Transition Probability Table as well as the final states will be used from the
pattern recognition scheme to train the system for abnormal routes and achieve an efficient normalization. The Pattern recognition scheme will have as inputs the initial states as well. This will help the system not to dismiss abnormalities in tracking paths but to use them as initial possible states, in order to identify peculiar routes or locations.

![Diagram of proposed modified model]

**Figure 1: Proposed Modified model**

In figure 1, obtaining the initial state can be optimized using the Maximum Likelihood Estimate (MLE) based on the Map aware model to prevent not feasible location. In continue, Transition Probability Table needs to be calculated. Observation probabilities needed, will be sensitive to parameters calculated from the IMM and the proposed model, thus, more accurate measurements will occur. Moreover, considering the initial state as input, the pattern recognition scheme will “train” the software to
identify abnormal paths. This can provide normalization to the whole model, however, it is not wise to exclude such tracking information, since sometimes it may exist. In the latter case, more research needs to be done over normalization algorithms.

1.22. Comparative Visualization

In this section there is a visualization of the desired tracking result after the proposed optimization. The scenario is about an outdoor area where there are obstacles and the mobile node is moving from the black point to the teal. The node is always accelerating as much as possible and decelerating harshly to provide difficult circumstances for the IMM. In picture 2 you can see the real route:

![Picture 2: Actual route](image)

The following pictures represent the desired result of the proposed optimization in comparison to the proposed that will be modified. Picture 3 represents the IMM (Figure 2) modification and picture 4 is an ideal preview of the proposal’s result (Figure 6).
The Map awareness when calculating the Transition Probability Table will provide us much more accurate results and help us lessen even more the error of NLOS environments.

1.23. Model summary

The HMM modified by the IMM flow chart is presented in Figure 2. The presented model has 3 optimizations. The initial state will be calculated, to prevent noise errors, by using the Maximum Likelihood Estimate(MLE) modified be the map-aware model.
This will exclude locations that are not feasible (Figure 3). Achieving such an optimization is of high value to the project, since knowing the initial location of the subject tracked is quite important, even more when the tracking procedures are about to be scaled with real-time identification of the subject. Furthermore, optimization is needed regarding the tracking accuracy of the model. There are common errors in location positioning models that can produce quite abnormal results, such the on in picture 3, trying to create a continuous tracking path. Those models, however, lack of map awareness, which seems quite essential when tracking a subject in a 3D environment. The proposed model’s goal is to achieve a map-aware tracking, so that location estimates can be improved and even normalized to achieve better latency results.

The proposed model Flow Chart is presented in Figure 4. There are two parts on this model. The computing part and the learning part. The first part, is about calculating the Transition Probability Table of the Hidden Markov Model interfering with the entropy of the HMM, modifying the observation symbols according to the map aware-model. Meanwhile, the Transition Probability Table will be modified by the already proposed Interacting Multiple Model (IMM) as well. Hence accuracy will dramatically improve, since location final states (final location estimates) will be feasible, according to the area morphology. Moreover, along with this part, the model will be “learning” from the pattern recognition scheme. In this learning part, the scheme will be using the final state location estimates in order to achieve a normalization concerning tracking paths as well as location estimates, meaning, additionally, transition probabilities. That scheme will extract data that will continuously update the transition probability computing and will be filtered through the map-aware model. It is quite essential to highlight that this scheme will update and train the system only after it will have enough data to extract some results, since it is a data mining procedure. Thus, updating will be happening, however, not in a way that will interfere with system’s performance. To make the long story short, the pattern recognition scheme is the part of the model that will be held responsible for the normalization of locating and tracking, hence it cannot be used a tool until it will become useful. To achieve such data mining will demand time, but after trained, it will provide quite an efficient tool.
Figure 2: Proposed model on [11]

Figure 3: Initial State Estimate Modification Flow Chart

Figure 4: Transition Probability Table Computing Modification Flow Chart

Figure 5: Pattern Recognition Scheme Flow Chart
1.24. Scalability

The model will be quite scalable. To begin with, layer parting of the area, in which locating and tracking is taking place, will help the big data issue of subjects’ information, yet data will be transferred to a server where the pattern recognition scheme will be proceeding with data mining. Concerning communication channels it can utilize as many as possible to obtain the acquired data and it will be designed to be efficient for huge networks. Furthermore, accuracy as well as latency are about to be dramatically improved. Moreover, this is a research concerning a real-time model. Power consumption of the system and complexity of the algorithm will be some of the main pivots around which the model will be researched.
1.25. Experimental results

Let a subject be moving from block 6 to 5 (Picture 2). The assumptions that will be followed are:

→ the subject can move one block at a time moment or with a very slight chance to stand still.

→ noise will be introduced to the system (we can see the noise transition probabilities in Table 2)

Table 2: Noise Transition Probabilities

<table>
<thead>
<tr>
<th>Pi</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.05</td>
<td>0.3</td>
<td>0.01</td>
<td>0.3</td>
<td>0.3</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>1</td>
<td>0.15</td>
<td>0.12</td>
<td>0.15</td>
<td>0.15</td>
<td>0.25</td>
<td>0.15</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>2</td>
<td>0.01</td>
<td>0.25</td>
<td>0.05</td>
<td>0.01</td>
<td>0.25</td>
<td>0.4</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
</tr>
<tr>
<td>3</td>
<td>0.2</td>
<td>0.1</td>
<td>0.07</td>
<td>0.3</td>
<td>0.3</td>
<td>0.01</td>
<td>0.2</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>4</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
<td>0.2</td>
<td>0.1</td>
<td>0.1</td>
<td>0.1</td>
</tr>
<tr>
<td>5</td>
<td>0.01</td>
<td>0.1</td>
<td>0.2</td>
<td>0.01</td>
<td>0.3</td>
<td>0.07</td>
<td>0.01</td>
<td>0.1</td>
<td>0.2</td>
</tr>
<tr>
<td>6</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.4</td>
<td>0.3</td>
<td>0.01</td>
<td>0.05</td>
<td>0.2</td>
<td>0.01</td>
</tr>
<tr>
<td>7</td>
<td>0.01</td>
<td>0.01</td>
<td>0.15</td>
<td>0.25</td>
<td>0.15</td>
<td>0.15</td>
<td>0.12</td>
<td>0.15</td>
<td>0.15</td>
</tr>
<tr>
<td>8</td>
<td>0.01</td>
<td>0.01</td>
<td>0.01</td>
<td>0.25</td>
<td>0.4</td>
<td>0.01</td>
<td>0.25</td>
<td>0.25</td>
<td>0.05</td>
</tr>
</tbody>
</table>

Java code was used to implement the HMM and conduct the experiment. The actual and the noise measurements can be seen in Table 3.

Table 3: Measurements

<table>
<thead>
<tr>
<th>Actual</th>
<th>Noise</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>6</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>1</td>
</tr>
<tr>
<td>5</td>
<td>8</td>
</tr>
</tbody>
</table>

According to the HMM, which is not terrain aware, the Transition Probability Table can be formed as seen in Table 4. However, the proposed scheme will provide the transition Probability Table of Table 5.
Table 4: TPT₁

<table>
<thead>
<tr>
<th>Pij</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td>0</td>
<td>0.4</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0.3</td>
<td>0.02</td>
<td>0.3</td>
<td>0.04</td>
<td>0.3</td>
<td>0.04</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0.4</td>
<td>0.1</td>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0.3</td>
<td>0.04</td>
<td>0</td>
<td>0.02</td>
<td>0.3</td>
<td>0</td>
<td>0.3</td>
<td>0.04</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0.04</td>
<td>0.2</td>
<td>0.04</td>
<td>0.2</td>
<td>0.04</td>
<td>0.2</td>
<td>0.04</td>
<td>0.2</td>
<td>0.4</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0.04</td>
<td>0.3</td>
<td>0</td>
<td>0.3</td>
<td>0.02</td>
<td>0</td>
<td>0.04</td>
<td>0.3</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>0</td>
<td>0.1</td>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.04</td>
<td>0.3</td>
<td>0.04</td>
<td>0.3</td>
<td>0.02</td>
<td>0.3</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.1</td>
<td>0.4</td>
<td>0</td>
<td>0.4</td>
<td>0.1</td>
</tr>
</tbody>
</table>

Table 5: TPT₂

<table>
<thead>
<tr>
<th>Pij</th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
<th>8</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>2</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>0.4</td>
<td>0</td>
<td>0.4</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.4</td>
<td>0</td>
<td>0.4</td>
<td>0.1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.3</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.7</td>
<td>0.1</td>
<td>0</td>
<td>0.2</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>7</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>8</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

1.25.1. Results

After running the experiment, the results are shown in the following table (table 6: results)

Table 6: Results

<table>
<thead>
<tr>
<th>RESULTS</th>
<th>measurements deviation</th>
<th>runtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>initial</td>
<td>2.25</td>
<td>not applied</td>
</tr>
<tr>
<td>HMM</td>
<td>1.5</td>
<td>2.418</td>
</tr>
<tr>
<td>ma_HMM</td>
<td>0.5</td>
<td>1.887</td>
</tr>
<tr>
<td>V_HMM</td>
<td>1.5</td>
<td>0.056</td>
</tr>
<tr>
<td>V_ma-HMM</td>
<td>0.5</td>
<td>0.054</td>
</tr>
</tbody>
</table>

Table 6 legend:

HMM: hidden Markov Model
ma-: map-aware optimization
V_: using the Viterbi optimization

As can be seen, there is a 300% less deviation using the map-aware model. Moreover, latency is dramatically decreased. The results are acquired 80% faster than the basic ones from the basic-HMM and using the Viterbi optimization the map-aware model still provides them 9.7% faster.
1.25.2. Summary

To sum up, it is obvious the results are quite more accurate and the latency is less as well. However, this experiment is about a static Transition Probability Table, which will be calculated according to the map-aware model. The IMM modification will imply a 2nd-grade HMM and a function, as referred in [11], for an adaptive TPT, however, designing this, will be future work. Yet, the results are far better than the existing model.
6. Future Work – Conclusion

1.26. Future Work

This proposal is based on my personal evaluation of the approaches I studied above. More time is needed to create and apply the proposed models on a new scheme, creating a robust algorithmic model. Meanwhile optimizations will be studied over the [11] and even some combinations with other approaches such as [14] and [22]. Concerning outdoor environments, more research need to be done so that the signature is not lost due to conflicts and noise. Applying the model may require an application which will be part of routers’ firmware or firewall, or even a special modification in mobile network or Bluetooth antennas to obtain such data. Finally, a normalization of the pattern recognition should be studied to identify abnormal tracking information and either investigate them or be considered as algorithm’s deviation errors, and meanwhile, experimental analysis will be conducted to test the model. Scaling the model to a user-friendly app for on-the-go access will be considered as well.

1.27. Conclusion

In conclusion, after a brief introduction to Location Based services, the localization and tracking problem is presented. Obviously, nowadays, being able not only to track, but to predict the following route as well, is a matter of great concern. The objective of this project is to conduct a quality evaluation of proposed approaches and models to cure this challenging issue. The project’s methodology is document-driven since I studied related works and evaluated them based on their publications and contributions. There is a section where an analysis is presented of all the methodology used to complete the project. The evaluation is based on specific challenging criteria and is visualized in a comparative table and analysed as well. In continue, a novel approach of a robust and promising model is presented. That theoretical model is a HMM-based one and it is a combination of a novel map-aware model with the proposal described in [11], enhanced with a pattern recognition scheme to achieve normalization, combining the localization and tracking model with the data mining. To be tested for its
viability, a simplified map-aware filter based on its logic was implemented in a simulation using Java code in Eclipse and the results were quite promising for its development. That model is supposed to be developed in order to meet the criteria used for the evaluation and with the potential to being implemented as an on-the-go application for system superusers.

Summing up, the map-aware model will affect the Transition Probability Table. However, the goal is to combine it with the already proposed Interacting Multiple Model [11]. The latter optimises the results of the Hidden Markov Model by affecting the observation symbols with the nodes’ velocity parameter. Despite, the results still can deviate from the feasible pathways of the area, hence results may be abnormal locations. By optimizing the observation probabilities with an “area/map aware” filter, faster and more accurate results can be obtained. The term faster sounds quite weird, since one more filter is to be applied while computing is not over yet, however, with this model we can achieve continuous normalization on tracking routes and train the model. This will not only end up in achieving lower latency, but will indicate peculiar location that deviate from the normalized ones.
7. References

In this section there are the document resources that were used on this Thesis, however, some term definitions where acquired from Wikipedia. However, since Wikipedia is quite unreliable, they were double checked and even corrected by notes from lectures and courses I attended, in order to expand my knowledge over this scientific sector. The referencing system used is the IEEE one.


Appendix A

Java Code for basic-HMM filter:

```java
package Markov;

//package sensor_fusion;
import java.awt.Color;
import java.awt.Graphics;
import java.awt.Image;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.LineNumberReader;
import java.util.Arrays;
import java.util.StringTokenizer;

import javax.swing.JFileChooser;
import javax.swing.JMenuBar;
import javax.swing.JOptionPane;

@SuppressWarnings("serial")
public class HMMFilter extends javax.swing.JFrame {

    int num_states = 9; //The number of possible states
    final int menubar_height = 50;
    int num_measurements = 0; // how many measurements I have stored so far
    int[] real_measurements; // the actual measurements
    int[] noise_measurements; // the 'noise' measurements
    int[] HMM_measurements; // the calculated filter values

    int[][] best_path; //shows the previous neighbour in the best HMM path
    double[][] transition_probabilities; //HMM model transition probability
    double[][] state_probabilities; //HMM model state probability
    double[][] noise_probabilities; // Input noise data structur
```
double max_value;
int file_opened=0;
int width, height;
int toggled=0;
Image backbuffer;
Graphics screen;

/** Creates new form HMMFilter */
public HMMFilter() {
    initComponents();

    real_measurements = new int[4];  
    noise_measurements = new int[4];
    HMM_measurements = new int[4];
    best_path=new int[num_states][4];
    transition_probabilities=new double[num_states][num_states];
    state_probabilities=new double[num_states][4];
    noise_probabilities=new double[num_states][num_states];

    setSize(500,400);
    width= getSize().width;
    height=getSize().height;

    backbuffer = createImage(width, height);
    screen = backbuffer.getGraphics();
    screen.setColor(Color.white);
    screen.fillRect(0, 0, width, height);
    screen.setColor(Color.black);
}

/** This method is called from within the constructor to
 * initialize the form.
 * WARNING: Do NOT modify this code. The content of this met
 * hod is
 * always regenerated by the Form Editor.
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-
BEGIN:initComponents

    private void initComponents() {
        jDialog1 = new javax.swing.JDialog();
        jButton1 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();
        jScrollPane1 = new javax.swing.JScrollPane();
        jTable1 = new javax.swing.JTable();
        jButton2 = new javax.swing.JButton();
        jCheckBox1 = new javax.swing.JCheckBox();
        jFileChooser1 = new javax.swing.JFileChooser();
        jPanel1 = new javax.swing.JPanel();
        jInternalFrame1 = new javax.swing.JInternalFrame();
        fileChooser = new javax.swing.JFileChooser();
        jDialog2 = new javax.swing.JDialog();
        jButton5 = new javax.swing.JButton();
        jDialog3 = new javax.swing.JDialog();
        jLabel5 = new javax.swing.JLabel();
        jButton3 = new javax.swing.JButton();
        jLabel1 = new javax.swing.JLabel();
        jLabel2 = new javax.swing.JLabel();
        jLabel3 = new javax.swing.JLabel();
        jLabel4 = new javax.swing.JLabel();
        jMenuBar1 = new javax.swing.JMenuBar();
        jMenu1 = new javax.swing.JMenu();
        miOpen = new javax.swing.JMenuItem();
        miDeviation = new javax.swing.JMenuItem();
        miNoise = new javax.swing.JMenuItem();
        HMMParameters = new javax.swing.JMenuItem();
        miClose = new javax.swing.JMenuItem();
        jButton1.setText("OK&Close");
        jButton1.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(java.awt.event.ActionEvent evt) {
                jButton1ActionPerformed(evt);
            }
        });
    }

- 61 -
jButton4.setText("Close");
JButton4.addActionListener(new java.awt.event.ActionEvent() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        JButton4ActionPerformed(evt);
    }
});

jButton4.setModel(new javax.swing.table.DefaultTableModel(
    //transition probability table
    new Object [[]] {
        "0", new Float(0.1), new Float(0.4), new Float(0.0), new Float(0.04), new Float(0.01), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0),
        "1", new Float(0.3), new Float(0.02), new Float(0.3), new Float(0.04), new Float(0.3), new Float(0.0), new Float(0.0), new Float(0.0),
        "2", new Float(0.0), new Float(0.4), new Float(0.1), new Float(0.0), new Float(0.1), new Float(0.4), new Float(0.0), new Float(0.0),
        "3", new Float(0.3), new Float(0.04), new Float(0.0), new Float(0.02), new Float(0.3), new Float(0.0), new Float(0.04), new Float(0.0),
        "4", new Float(0.04), new Float(0.2), new Float(0.04), new Float(0.2), new Float(0.04), new Float(0.2), new Float(0.04),
        "5", new Float(0.0), new Float(0.04), new Float(0.0), new Float(0.0), new Float(0.1), new Float(0.0), new Float(0.02), new Float(0.02),
        "6", new Float(0.0), new Float(0.0), new Float(0.4), new Float(0.1), new Float(0.0), new Float(0.1), new Float(0.4), new Float(0.0),
        "7", new Float(0.0), new Float(0.0), new Float(0.04), new Float(0.3), new Float(0.04), new Float(0.3), new Float(0.02), new Float(0.3),
        "8", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.1), new Float(0.4), new Float(0.0), new Float(0.01)
    },
    new String [] {
"Pij", "0", "1", "2", "3", "4", "5", "6", "7", "8"

} }

Class[] types = new Class[] {
    java.lang.String.class, java.lang.Float.class,
    java.lang.Float.class, java.lang.Float.class,
    java.lang.Float.class, java.lang.Float.class,
    java.lang.Float.class, java.lang.Float.class,
    java.lang.Float.class
};

public Class getColumnClass(int columnIndex) {
    return types[columnIndex];
}

jScrollPane1.setViewportView(jTable1);
jTable1.getColumnModel().getColumn(0).setResizable(true);

jButton2.setText("OK&Filter");
jButton2.addActionListener(new java.awt.event.ActionEvent()
    {
        public void actionPerformed(java.awt.event.ActionEvent
        evt) {
            jButton2ActionPerformed(evt);
        }
    });

jCheckBox1.setText("Use Viterbi decoding");

javax.swing.GroupLayout jDialog1Layout = new javax.swing.GroupLayout
    (jDialog1.getContentPane());
jDialog1.getContentPane().setLayout(jDialog1Layout);
jDialog1Layout.setHorizontalGroup(
    jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jDialog1Layout.createSequentialGroup()
    .addContainerGap()
    .addGroup(jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
    .addComponent(jScrollPane1,
    javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jDialog1Layout.createSequentialGro
up()
  .addComponent(jButton1)
  .addGap(106, 106, 106)
  .addComponent(jButton2)
  .addPreferredGap(javax.swing.LayoutStyle.
    ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
  .addComponent(jButton4))
  .addContainerGap())
  .addGroup(jDialog1Layout.createSequentialGroup()
    .addGap(173, 173, 173)
    .addComponent(jCheckBox1)
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
  jDialog1Layout.setVerticalGroup(
    jDialog1Layout.createParallelGroup(javax.swing.GroupLayout
      Alignment.LEADING)
    .addGroup(jDialog1Layout.createSequentialGroup()
      .addGap(173, 173, 173)
      .addComponent(jCheckBox1)
      .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
  );
  jDialog1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
      .addGap(0, 100, Short.MAX_VALUE)
  );
  javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
  jPanel1.setLayout(jPanel1Layout);
  jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
      .addGap(0, 100, Short.MAX_VALUE)
  );
  javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
  jPanel1.setLayout(jPanel1Layout);
  jPanel1Layout.setHorizontalGroup(
    jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
      .addGap(0, 100, Short.MAX_VALUE)
  );
jPanel1Layout.setVerticalGroup(jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGap(0, 100, Short.MAX_VALUE)
);

jInternalFrame1.setVisible(true);

fileChooser.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        fileChooserActionPerformed(evt);
    }
});

javax.swing.GroupLayout jInternalFrame1Layout = new javax.swing.GroupLayout(jInternalFrame1.getContentPane());
jInternalFrame1.getContentPane().setLayout(jInternalFrame1Layout);

jInternalFrame1Layout.setHorizontalGroup(
jInternalFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addContainerGap()
);
jButton5.setText("OK&Close");
jButton5.addActionListener(new java.awt.event.ActionEvent()
    {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton5ActionPerformed(evt);
        }
    });

jButton8.setText("Close");
jButton8.addActionListener(new java.awt.event.ActionEvent()
    {
        public void actionPerformed(java.awt.event.ActionEvent evt) {
            jButton8ActionPerformed(evt);
        }
    });

jTable2.setModel(new javax.swing.table.DefaultTableModel
    //noise probability table
    new Object[][] {
        {"0", new Float(0.05), new Float(0.3), new Float(0.01), new Float(0.3), new Float(0.01), new Float(0.01), new Float(0.01)},
        {"1", new Float(0.15), new Float(0.12), new Float(0.15), new Float(0.15), new Float(0.15), new Float(0.01), new Float(0.01), new Float(0.01)},
        {"2", new Float(0.01), new Float(0.25), new Float(0.05), new Float(0.01), new Float(0.01), new Float(0.4), new Float(0.01), new Float(0.01), new Float(0.01)},
        {"3", new Float(0.2), new Float(0.1), new Float(0.01), new Float(0.01), new Float(0.07), new Float(0.07), new Float(0.01), new Float(0.2), new Float(0.01), new Float(0.01), new Float(0.1), new Float(0.01)},
        {"4", new Float(0.1), new Float(0.1), new Float(0.1), new Float(0.1), new Float(0.1), new Float(0.01), new Float(0.1), new Float(0.1), new Float(0.1)},
        {"5", new Float(0.01), new Float(0.1), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01)},
        {"6", new Float(0.01), new Float(0.01)}
    });
new Float(0.01), new Float(0.4), new Float(0.4), new Float(0.01), new Float(0.05), new Float(0.2), new Float(0.01)),
("7", new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.15), new Float(0.15), new Float(0.15), new Float(0.12), new Float(0.15))},
new String [] {
"Pij", "0", "1", "2", "3",
"4", "5", "6", "7", "8"
}

Class[] types = new Class [] {
java.lang.String.class, java.lang.Float.class,
java.lang.Float.class, java.lang.Float.class, java.lang.Float.class,
java.lang.Float.class, java.lang.Float.class, java.lang.Float.class,
java.lang.Float.class, java.lang.Float.class
};

public Class getColumnClass(int columnIndex) {
return types [columnIndex];
}
);
jScrollPane2.setViewportView(jTable2);
jTable2.getColumnModel().getColumn(0).setResizable(true);

javax.swing.GroupLayout jDialog2Layout = new javax.swing.GroupLayout(jDialog2.getContentPane());
jDialog2.getContentPane().setLayout(jDialog2Layout);
jDialog2Layout.setHorizontalGroup(
jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
.addGroup(jDialog2Layout.createSequentialGroup())
.addContainerGap()
.addGroup(jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING))
.addComponent(jScrollPane2, javax.swing.GroupLayout.Alignment.TRAILING)
.addGroup(jDialog2Layout.createSequentialGroup())

.addGroup(jDialog2Layout.createSequentialGroup());
.addContainerGap()
.addGroup(jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING))
.addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_ALIGNMENT.LEADING)
.addGroup(jDialog2Layout.createSequentialGroup());
up()  
.addComponent(jButton5)  
.addPreferredGap(javax.swing.LayoutStyle.
ComponentPlacement.RELATED, javax.swing.GroupLayout.DEFAULT_SIZE,
.addComponent(jButton8))  
.addContainerGap();  
jDialog2Layout.setVerticalGroup(
  jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
    .addGroup(jDialog2Layout.createSequentialGroup())  
    .addContainerGap()  
    .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 109, javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGap(48, 48, 48)  
    .addGroup(jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)  
      .addComponent(jButton8)  
      .addComponent(jButton5))  
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE))  
});

jLabel5.setText("jLabel5");

jButton3.setText("OK");

jButton3.addActionListener(new java.awt.event.ActionListener() {  
  public void actionPerformed(java.awt.event.ActionEvent evt) {  
    jButton3ActionPerformed(evt);  
  }  
});

javax.swing.GroupLayout jDialog3Layout = new javax.swing.GroupLayout(jDialog3.getContentPane());  
  jDialog3.getContentPane().setLayout(jDialog3Layout);  
  jDialog3Layout.setHorizontalGroup(  
    jDialog3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)  
      .addGroup(javax.swing.GroupLayout.Alignment.TRAILING)
  )
  .addGroup(javax.swing.GroupLayout.Alignment LENG}
, jDialog3Layout.createSequentialGroup()
    .addContainerGap(26, Short.MAX_VALUE)
    .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
    .addComponent(jButton3)
jMenu1.setText("File");
jMenu1.addMenuListener(new javax.swing.event.MenuListener()
{
    public void menuDeselected(javax.swing.event.MenuEvent evt)
    {
        jMenu1MenuDeselected(evt);
    }
    public void menuCanceled(javax.swing.event.MenuEvent evt)
    {
    }
    public void menuSelected(javax.swing.event.MenuEvent evt)
    {
        jMenu1MenuSelected(evt);
    }
});
miOpen.setText("Open");
miOpen.addActionListener(new java.awt.event.ActionEvent
{
    public void actionPerformed(java.awt.event.ActionEvent evt)
    {
        miOpenActionPerformed(evt);
    }
});
jMenu1.add(miOpen);
miDeviation.setText("Calculate deviation");
miDeviation.addActionListener(new java.awt.event.ActionEvent
{
    public void actionPerformed(java.awt.event.ActionEvent evt)
    {
        miDeviationActionPerformed(evt);
    }
});
jMenu1.add(miDeviation);
miNoise.setText("Set Noise Parameters");
miNoise.addActionListener(new java.awt.event.ActionEvent
{
    public void actionPerformed(java.awt.event.ActionEvent evt)
    {
        miNoiseActionPerformed(evt);
    }
});
HMMParameters.setText("Set Transition Probabilities");
HMMParameters.addActionListener(new java.awt.event.ActionListener() { 
    public void actionPerformed(java.awt.event.ActionEvent evt) { 
        HMMParametersActionPerformed(evt); 
    } 
});
jMenu1.add(HMMParameters);
miClose.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK_F4, java.awt.event.InputEvent.ALT_MASK));
miClose.setText("Close");
miClose.addActionListener(new java.awt.event.ActionListener() { 
    public void actionPerformed(java.awt.event.ActionEvent evt) { 
        miCloseActionPerformed(evt); 
    } 
});
jMenu1.add(miClose);
jMenuBar1.add(jMenu1);
setJMenuBar(jMenuBar1);
pack();
private void miCloseActionPerformed(java.awt.event.ActionEvent evt) { //GEN-FIRST:event_miCloseActionPerformed
        System.exit(0);
} //GEN-LAST:event_miCloseActionPerformed

public static double calcDist(double p1, double p2) { 
    return Math.sqrt(Math.pow(p1-p2, 2));
}
private void miDeviationActionPerformed(java.awt.event.ActionEvent evt) {
    // προκειμένου την απόκλιση του θορύβου, HMM filter από την πραγματική τιμή
    double deviationFromNoise = 0.0, deviationFromHMM = 0.0;

    int num = num_measurements;

    for (int i = 0; i < num; i++) {
        deviationFromNoise += calcDist(real_measurements[i], noise_measurements[i]);
        deviationFromHMM += calcDist(real_measurements[i], HMM_measurements[i]);
    }

    String message = "Deviation between noise and real measurements: " + deviationFromNoise / num + "\n";
    message += "Deviation between HMM filter results and real measurements: " + deviationFromHMM / num + "\n";
    JOptionPane.showMessageDialog(null, message, "Calculated deviation values", JOptionPane.INFORMATION_MESSAGE);
}

private void miOpenActionPerformed(java.awt.event.ActionEvent evt) {
    // Dialog box to open measurements file
    int returnVal = fileChooser.showOpenDialog(this);
    int i=0;
    String display_real_measurements, display_noise_measurements, temp;
    display_real_measurements="measurements:  "
    );
    display_noise_measurements="measurements:  "
    );
temp = new String();

if (returnVal == JFileChooser.APPROVE_OPTION) {

    // Clear screen
    screen.setColor(Color.white);
    screen.fillRect(0, 0, width, height);
    screen.setColor(Color.black);
    repaint();

    File file = fileChooser.getSelectedFile();
    try {
        FileReader fr = new FileReader(file.getAbsoluteP
        ath());
        LineNumberReader inreader = new LineNumberReader
        (fr);
        String line = new String();
        String[] result;

        double x, noiseX;
        while ((line = inreader.readLine()) != null) {
            result = line.split("\s");
            real_measurements[i] = Integer.parseInt(result
            t[0]);
            temp = String.valueOf(real_measurements[i]);
            display_real_measurements = display_real_measurements + temp + " ";

            noise_measurements[i] = Integer.parseInt(res
            ult[1]);
            temp = String.valueOf(noise_measurements[i]);
            display_noise_measurements = display_noise_measurements + temp + " ";

            i++;
        }
        num_measurements = i;
    }

    jLabel1.setText("Real " + display_real_measurements);
    }}
jLabel2.setText("Noise ".concat(display_noise_measurements));

toggleHMMs();

} catch (IOException ex) {
    System.out.println("problem accessing file" + file.getAbsolutePath());
}

} else {
    System.out.println("File access cancelled by user.");
}

miNoise.setEnabled(true);

private void HMMParametersActionPerformed(java.awt.event.ActionEvent evt) {
    jDialog1.setVisible(rootPaneCheckingEnabled); // TODO add your handling code here:
    jDialog1.setSize(487, 250);
    jDialog1.setTitle("Transition probabilities");
}

private void fileChooserActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
}

private void jMenu1MenuDeselected(javax.swing.event.MenuEvent ev) {
    // TODO add your handling code here:
    repaint();
}

private void formMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
}

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
    jDialog1.dispose();
private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // Read transition probabilities
    int x, y;
    String temp;
    for (x = 0; x < num_states; x++)
        for (y = 1; y <= num_states; y++)
            temp = (jTable1.getValueAt(x, y)).toString();
    transition_probabilities[x][y - 1] = Double.valueOf(temp).doubleValue();
    jDialog1.dispose();
}

private void jButton5ActionPerformed(java.awt.event.ActionEvent evt) {
    // Read noise probabilities
    int x, y;
    String temp;
    for (x = 0; x < num_states; x++)
        for (y = 1; y <= num_states; y++)
            temp = (jTable2.getValueAt(x, y)).toString();
    noise_probabilities[x][y - 1] = Double.valueOf(temp).doubleValue();
    calculate_state_probabilities();
    jDialog2.dispose();
}

private void jButton8ActionPerformed(java.awt.event.ActionEvent evt) {
543    // TODO add your handling code here:
544    jDialog2.dispose();
545    } //GEN-LAST:event_jButton8ActionPerformed
546
548    // TODO add your handling code here:
549    jDialog2.setVisible(rootPaneCheckingEnabled);
550    jDialog2.setSize(487, 250);
551    jDialog2.setTitle("Set Noise Parameters");
552    } //GEN-LAST:event_miNoiseActionPerformed
553
555    // Check box to enable exhaustive search decoding or Viterbi decoding
556
557    jButton1ActionPerformed(evt);
558
559    if(jCheckBox1.isSelected()==true) {
560        calculateHMM_Viterbi();
561    } else {
562        calculateHMM();
563    }
564
565    jButton2ActionPerformed(evt);  //GEN-LAST:event_jButton2ActionPerformed
566
567    private void jMenuItem1MenuSelected(javax.swing.event.MenuEvent evt) {  //GEN-FIRST:event_jMenuItem1MenuSelected
568    // TODO add your handling code here:
569    if(file_opened==0) {
570        miNoise.setEnable(false);
571        HMMParameters.setEnable(false);
572    } file_opened=1;
573    } //GEN-LAST:event_jMenuItem1MenuSelected
574
575    private void jMenuItem1MenuSelected(java.awt.event.ActionEvent evt) {  //GEN-FIRST:event_jMenuItem1MenuSelected
576    // TODO add your handling code here:
577    if(file_opened==0) {
578        miNoise.setEnable(false);
579        HMMParameters.setEnable(false);
580    } file_opened=1;
581    } //GEN-LAST:event_jMenuItem1MenuSelected
582
583    private void jMenuItem1MenuSelected(java.awt.event.ActionEvent evt) {  //GEN-FIRST:event_jMenuItem1MenuSelected
584
void calculate_state_probabilities()
{
    // Set HMM state probabilities
    int x, y;
    double temp=0;

    for(y=0;y<num_measurements;y++)
    {
        for(x=0;x<num_states;x++)
        {
            state_probabilities[x][y]=noise_probabilities[noise_measurements[y]][x];
            // System.out.println(noise_measurements[y]);
            // System.out.println(Arrays.deepToString(state_probabilities));
        }
    }
    HMMParameters.setEnabled(true);
}

void toggleHMMs() {
    // Can be used to hide or show the HMM filter series. Currently unused
    if(toggled==0)
    {
        jLabel3.setText("Filtered measurements: ");
        jLabel4.setText("Filtered measurements (Viterbi): ");
    }
    else
    {
        toggled=0;
        jLabel3.setText(null);
    }
void calculateHMM_Viterbi()
{
    // Viterbi decoding
    int counter1, counter2, counter3;
    double tmp[][], temp=0.0, max_value=0.0, temp1, temp2, temp3;
    int output_path[], current_path[];

    output_path=new int[num_measurements]; //The
    current_path=new int[num_measurements];
    tmp=new double[num_states][num_measurements];

    String path=new String();
    long start = System.nanoTime();

    for(counter1=0;counter1<num_measurements;counter1++) //I
        initializations
    {
        output_path[counter1]=-1;
        current_path[counter1]=-1;
        for(counter2=0;counter2<num_states;counter2++)
        {
            best_path[counter2][counter1]=-1;
            tmp[counter2][counter1]=state_probabilities[counter2][counter1];
        }
    }

    for(counter1=1;counter1<num_measurements;counter1++) //I
        identify the best path to each state for each measurement
    {
        for(counter2=0;counter2<num_states;counter2++)
        {
            max_value=0.0;
            for(counter3=0;counter3<num_states;counter3++)
            {
temp=tmp[counter3][counter1-1]*transition_probabilities[counter3][counter2];

    if(temp>max_value)
    {
        max_value=temp;
        best_path[counter2][counter1]=counter3;
        //Memorize the previous node for the best path to the current one
    }
    }
    temp[counter2][counter1]=max_value;
    }
}
max_value=0;
tmp=-1;
for(counter2=0;counter2<num_states;counter2++) //Find the best path for the st of paths to each state of the final measurement
{
    counter1=num_measurements-1;
    current_path[counter1]=counter2;
    temp3=state_probabilities[counter2][counter1];

    while((best_path[current_path[counter1]][counter1]>-1)&&(counter1>0))
    {
        temp1=state_probabilities[best_path[current_path[ counter1]][counter1]][counter1-1];
        temp2=transition_probabilities[best_path[current_ path[counter1]][counter1]][current_path[counter1]];
        temp=temp1*temp2*temp3;
        current_path[counter1-
counter1--;

}  

if((temp>max_value)&&(counter1==0))
{
    max_value=temp;
    for(counter3=0;counter3<num_measurements;counter3++)
    {
        output_path[counter3]=current_path[counter3];
        HMM_measurements[counter3]=output_path[counter3];
    }
}

for(counter1=0;counter1<num_measurements;counter1++)
{
    path=path+String.valueOf(output_path[counter1]);
    path=path+" ";
}

jLabel4.setText("Filtered measurements (Viterbi): "+path);

double elapsedTime = System.nanoTime()-start;
elapsedTime/=Math.pow(10,6);

jDialog3.setVisible(rootPaneCheckingEnabled);
jDialog3.setSize(200, 120);
jDialog3.setTitle("Viterbi decoder");
jLabel5.setText("Running time: "+String.valueOf(elapsedTime)+" msec");
System.out.println(elapsedTime);
```java
void calculateHMM() {
    // Exhaustive search decoding
    long counter, num_paths;

    double max_value=0, temp=1.0, temp1, temp2;
    int output_path[], current_path[], counter1;
    boolean updated;
    String path=new String();
    output_path= new int[num_measurements];
    current_path= new int[num_measurements];

    long start = System.nanoTime();

    num_paths=(long) Math.pow(num_states, num_measurements);
    // The number of all paths to exhaustively search

    for(counter1=0;counter1<num_measurements;counter1++)
    {
        current_path[counter1]=0;
    }

    for(counter=0;counter<num_paths;counter++) // Search each
        path and find the best one
    {
        for(counter1=0;counter1<num_measurements;counter1++)
        {
            if(counter1<num_measurements-1)
            {
                templ=state_probabilities[current_path[counter1]][counter1];
                temp2=transition_probabilities[current_path[counter1]][current_path[counter1+1]];
                temp*=templ*temp2;
            }
            else
            {
```
temp1 = state_probabilities[current_path[counter1]][counter1];

temp* = temp1;
}

if (temp >= max_value) // If true, this path is currently the best path
{
    max_value = temp;
    for (counter1 = 0; counter1 < num_measurements; counter1++)
    {
        output_path[counter1] = current_path[counter1];
        HMM_measurements[counter1] = current_path[counter1];
    }
}

temp = 1.0;
counter1 = num_measurements - 1;

while (current_path[counter1] + 1 >= num_states) // calculate the next path to check
{
    current_path[counter1] = 0;
    counter1--;
    if (counter1 < 0)
        break;
}

if (counter1 >= 0)
    current_path[counter1]++;

for (counter1 = 0; counter1 < num_measurements; counter1++)
{
    path = path + String.valueOf(output_path[counter1]);
    path = path + " ";
double elapsedTime = System.nanoTime() - start;
elapsedTime /= Math.pow(10, 6);

jDialog3.setVisible(rootPaneCheckingEnabled);
jDialog3.setSize(230, 120);
jDialog3.setTitle("Exhaustive search decoder");
jLabel5.setText("Running time: "+String.valueOf(elapsedTime)+" msec");
System.out.println(elapsedTime);

@override
public void update(Graphics g) {
    //g.drawImage(backbuffer, 0, menubar_height, this);
}

@override
public void paint(Graphics g) {
    super.paint(g);
    //update(g);
}

/**
 * @param args the command line arguments
 */
public static void main(String args[]) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new HMMFilter().setVisible(true);
        }
    });
}

// Variables declaration - do not modify//GEN-
BEGIN:variables
  private javax.swing.JMenuItem HMMParameters;
  private javax.swing.JFileChooser fileChooser;
  private javax.swing.JButton jButton1;
  private javax.swing.JButton jButton2;
  private javax.swing.JButton jButton3;
  private javax.swing.JButton jButton4;
  private javax.swing.JButton jButton5;
  private javax.swing.JButton jButton8;
  private javax.swing.JCheckBox jCheckBox1;
  private javax.swing.JDialog jDialog1;
  private javax.swing.JDialog jDialog2;
  private javax.swing.JDialog jDialog3;
  private javax.swing.JFileChooser jFileChooser1;
  private javax.swing.JInternalFrame jInternalFrame1;
  private javax.swing.JLabel jLabel1;
  private javax.swing.JLabel jLabel2;
  private javax.swing.JLabel jLabel3;
  private javax.swing.JLabel jLabel4;
  private javax.swing.JLabel jLabel5;
  private javax.swing.JMenu jMenu1;
  private javax.swing.JMenuBar jMenuBar1;
  private javax.swing.JPanel jPanel1;
  private javax.swing.JScrollPane jScrollPane1;
  private javax.swing.JScrollPane jScrollPane2;
  private javax.swing.JTable jTable1;
  private javax.swing.JTable jTable2;
  private javax.swing.JMenuItem miClose;
  private javax.swing.JMenuItem miDeviation;
  private javax.swing.JMenuItem miNoise;
  private javax.swing.JMenuItem miOpen;
// End of variables declaration//GEN-END:variables
Java Code for HMM filter with map-aware Transition Probability Table:

```java
package ModMarkov;

import java.awt.Color;
import java.awt.Graphics;
import java.awt.Image;
import java.io.File;
import java.io.FileReader;
import java.io.IOException;
import java.io.LineNumberReader;
import java.util.StringTokenizer;
import javax.swing.JFileChooser;
import javax.swing.JMenuBar;
import javax.swing.JOptionPane;

@SuppressWarnings("serial")
public class HMMFilter extends javax.swing.JFrame {

    int num_states = 9; // The number of possible states
    final int menubar_height = 50;
    int num_measurements = 0; // how many measurements I have stored so far
    int[] real_measurements; // the actual measurements
    int[] noise_measurements; // the 'noise' measurements
    int[] HMM_measurements; // the calculated filter values
    int[][] best_path; // shows the previous neighbour in the best HMM path
    double[][] transition_probabilities; // HMM model transition probability
    double[][] state_probabilities; // HMM model state probabilities
    double[][] noise_probabilities; // Input noise data structure
    double max_value;
    int file_opened=0;
    int width, height;
    int toggled=0;
    Image backbuffer;
    Graphics screen;
}
```
/** Creates new form HMMFilter */
public HMMFilter() {
    initComponents();

    real_measurements = new int[4]; // αρχικοποιήσεις
    noise_measurements = new int[4];
    HMM_measurements = new int[4];
    best_path = new int[num_states][4];
    transition_probabilities = new double[num_states][num_states][4];
    state_probabilities = new double[num_states][4];
    noise_probabilities = new double[num_states][num_states];

    setSize(500,400);
    width = getSize().width;
    height = getSize().height;

    backbuffer = createImage(width, height);
    screen = backbuffer.getGraphics();
    screen.setColor(Color.white);
    screen.fillRect(0, 0, width, height);
    screen.setColor(Color.black);

}

/** This method is called from within the constructor to 
 * initialize the form. 
 * WARNING: Do NOT modify this code. The content of this method 
 * is always regenerated by the Form Editor. 
 */
@SuppressWarnings("unchecked")
// <editor-fold defaultstate="collapsed" desc="Generated Code">//GEN-BEGIN:initComponents

    private void initComponents() {
        jDialog1 = new javax.swing.JDialog();
        jButton1 = new javax.swing.JButton();
        jButton4 = new javax.swing.JButton();


jScrollPane1 = new javax.swing.JScrollPane();
jTable1 = new javax.swing.JTable();
jButton2 = new javax.swing.JButton();
jCheckBox1 = new javax.swing.JCheckBox();
jFileChooser1 = new javax.swing.JFileChooser();
jPanel1 = new javax.swing.JPanel();
jInternalFrame1 = new javax.swing.JInternalFrame();
fileChooser = new javax.swing.JFileChooser();
jDialog2 = new javax.swing.JDialog();
jButton5 = new javax.swing.JButton();
jButton8 = new javax.swing.JButton();
jScrollPane2 = new javax.swing.JScrollPane();
jTable2 = new javax.swing.JTable();
jDialog3 = new javax.swing.JDialog();
jLabel5 = new javax.swing.JLabel();
jButton3 = new javax.swing.JButton();
jLabel2 = new javax.swing.JLabel();
jLabel1 = new javax.swing.JLabel();
jLabel3 = new javax.swing.JLabel();
jLabel4 = new javax.swing.JLabel();
jMenuBar1 = new javax.swing.JMenuBar();
jMenu1 = new javax.swing.JMenu();
miOpen = new javax.swing.JMenuItem();
miDeviation = new javax.swing.JMenuItem();
miNoise = new javax.swing.JMenuItem();
HMMParameters = new javax.swing.JMenuItem();
miClose = new javax.swing.JMenuItem();

jButton1.setText("OK&Close");
jButton1.addActionListener(new java.awt.event.ActionEvent
class() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton1ActionPerformed(evt);
    }
});

jButton4.setText("Close");
jButton4.addActionListener(new java.awt.event.ActionEvent
class() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton4ActionPerformed(evt);
    }
});
jTable1.setModel(new javax.swing.table.DefaultTableModel

    // transition probability table
    new Object [][] { 
        [
            ["1", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["2", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["3", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["4", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["5", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["6", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["7", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["8", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)],
            ["9", new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0), new Float(0.0)]
        ],

        new String [] { 
            "Pij", " 0", " 1", " 2", " 3", " 4", " 5", " 6", " 7", " 8"
        }
    };

new Class[] types = new Class [] { 
    java.lang.String.class, java.lang.Float.class
public Class getColumnClass(int columnIndex) {
    return types [columnIndex];
}

jScrollPane1.setViewportView(jTable1);
jTable1.getColumnModel().getColumn(0).setResizable(true);

jButton2.setText("OK&Filter");
jButton2.addActionListener(new java.awt.event.ActionEvent() {
    public void actionPerformed(java.awt.event.ActionEvent ev)
    {
        jButton2ActionPerformed(ev);
    }
});

jCheckBox1.setText("Use Viterbi decoding");

javax.swing.GroupLayout jDialog1Layout = new javax.swing.GroupLayout(jDialog1.getContentPane());
jDialog1.getContentPane().setLayout(jDialog1Layout);
jDialog1Layout.setHorizontalGroup(
    jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jDialog1Layout.createSequentialGroup()
    .addContainerGap()
    .addGroup(jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGroup(jDialog1Layout.createSequentialGroup()))
    .addGap(106, 106, 106)
    .addComponent(jButton1)
    .addGap(18, 18, 18)
    .addComponent(jButton2)
    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
    javax.swing.GroupLayout.DEFAULT_SIZE
    .createSequentialGroup()
    .addGroup(jDialog1Layout.createSequentialGroup()
    .addGroup(jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addComponent(jScrollPane1, javax.swing.GroupLayout.Alignment.TRAILING)
    .addComponent(jButton1)
    .addGap(106, 106, 106)
    .addComponent(jButton2)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.Alignment.TRAILING)
    .addGap(18, 18, 18)
    .addComponent(jButton2, javax.swing.GroupLayout.AlignmentTra
E, Short.MAX_VALUE)

        .addComponent(jButton4))
        .addComponentGap()
        .addGroup(jDialog1Layout.createSequentialGroup()
            .addGap(173, 173, 173)
            .addComponent(jCheckBox1)
            .addComponentGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
        );
    jDialog1Layout.setVerticalGroup(
        jDialog1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGroup(jDialog1Layout.createSequentialGroup()
                .addComponent(jCheckBox1)
                .addComponent(jPanel1)
                .addComponent(jButton2))
            .addComponentGap(18, 18, 18)
            .addComponent(jPanel1Layout.PREFERRED_SIZE, 109, javax.swing.GroupLayout.PREFERRED_SIZE)
            .addComponent(jPanel1Layout.PREFERRED_SIZE, Short.MAX_VALUE))
    )
    javax.swing.GroupLayout jPanel1Layout = new javax.swing.GroupLayout(jPanel1);
    jPanel1.setLayout(jPanel1Layout);
    jPanel1Layout.setHorizontalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 100, Short.MAX_VALUE)
    jPanel1Layout.setVerticalGroup(
        jPanel1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
            .addGap(0, 100, Short.MAX_VALUE)
    );
jInternalFrame1.setVisible(true);

fileChooser.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        fileChooserActionPerformed(evt);
    }
});

javax.swing.GroupLayout jInternalFrame1Layout = new javax.swing.GroupLayout(jInternalFrame1.getContentPane());
jInternalFrame1.getContentPane().setLayout(jInternalFrame1Layout);

jInternalFrame1Layout.setHorizontalGroup(
    jInternalFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jInternalFrame1Layout.createSequentialGroup()
    .addContainerGap())
);  

jInternalFrame1Layout.setVerticalGroup(
    jInternalFrame1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jInternalFrame1Layout.createSequentialGroup()
    .addContainerGap())
);  

jButton5.setText("OK&Close");

jButton5.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
jButton5ActionPerformed(evt);
}

jButton8.setText("Close");
jButton8.addActionListener(new java.awt.event.ActionEvent
{
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        jButton8ActionPerformed(evt);
    }
});

jTable2.setModel(new javax.swing.table.DefaultTableModel
(
    //noise probability table
    new Object [][] {
        {"1", new Float(0.05), new Float(0.3), new Float(0.01), new Float(0.3), new Float(0.3),
            new Float(0.01), new Float(0.01), new Float(0.01)},
        {"2", new Float(0.15), new Float(0.12), new Float(0.15), new Float(0.15), new Float(0.15),
            new Float(0.01), new Float(0.01), new Float(0.01)},
        {"3", new Float(0.01), new Float(0.25), new Float(0.01), new Float(0.01), new Float(0.01),
            new Float(0.01), new Float(0.01), new Float(0.01)},
        {"4", new Float(0.2), new Float(0.1), new Float(0.01), new Float(0.01), new Float(0.07),
            new Float(0.07), new Float(0.01), new Float(0.01)},
        {"5", new Float(0.1), new Float(0.1), new Float(0.1), new Float(0.1), new Float(0.2),
            new Float(0.1), new Float(0.1), new Float(0.1)},
        {"6", new Float(0.01), new Float(0.1), new Float(0.2), new Float(0.01), new Float(0.01),
            new Float(0.01), new Float(0.07), new Float(0.01), new Float(0.1), new Float(0.2)},
        {"7", new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01),
            new Float(0.01), new Float(0.01), new Float(0.01)},
        {"8", new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01),
            new Float(0.01), new Float(0.15), new Float(0.15), new Float(0.01), new Float(0.12),
            new Float(0.01)},
        {"9", new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.01),
            new Float(0.01), new Float(0.01), new Float(0.01), new Float(0.25),
            new Float(0.05)}
    })
new String[] {
    " Pij", " 0", " 1", " 2", " 3 ", " 4", " 5", " 6", " 7", " 8" 
}

Class[] types = new Class[] {
    java.lang.String.class, java.lang.Float.class,
    java.lang.Float.class, java.lang.Float.class
};

public Class getColumnClass(int columnIndex) {
    return types[columnIndex];
}

javax.swing.GroupLayout jDialog2Layout = new javax.swing.GroupLayout(jDialog2.getContentPane());
jDialog2.getContentPane().setLayout(jDialog2Layout);

jDialog2Layout.setHorizontalGroup(
    jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jDialog2Layout.createSequentialGroup()
            .addContainerGap()
            .addGroup(jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addComponent(jScrollPane2, javax.swing.GroupLayout.Alignment.TRAILING)
                .addGroup(jDialog2Layout.createSequentialGroup()
                    .addComponent(jButton5)
                    .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
                            javax.swing.GroupLayout.DEFAULT_SIZE,
                            Short.MAX_VALUE)
                    .addComponent(jButton8)))
            .addContainerGap())
);
```java
jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
    .addGroup(jDialog2Layout.createSequentialGroup()
        .addContainerGap()
        .addComponent(jScrollPane2, javax.swing.GroupLayout.PREFERRED_SIZE, 109, javax.swing.GroupLayout.PREFERRED_SIZE)
    .addGap(48, 48, 48)
    .addGroup(jDialog2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.BASELINE)
        .addComponent(jButton8)
        .addComponent(jButton5))
    .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE));

jLabel5.setText("jLabel5");

jButton3.setText("OK");
jButton3.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt)
    {
        jButton3ActionPerformed(evt);
    }
});

javax.swing.GroupLayout jDialog3Layout = new javax.swing.GroupLayout(jDialog3.getContentPane());
jDialog3.getContentPane().setLayout(jDialog3Layout);
jDialog3Layout.setHorizontalGroup(
    jDialog3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
        .addGroup(javax.swing.GroupLayout.Alignment.LEADING, jDialog3Layout.createSequentialGroup()
            .addContainerGap(26, Short.MAX_VALUE)
            .addComponent(jLabel5, javax.swing.GroupLayout.PREFERRED_SIZE, 279)
        .addContainerGap());

jDialog4.getContentPane().setLayout(jDialog4Layout);
jDialog4Layout.setHorizontalGroup(
    jDialog4Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
        .addGroup(jDialog4Layout.createSequentialGroup()
            .addContainerGap(26, Short.MAX_VALUE)
            .addComponent(jLabel6, javax.swing.GroupLayout.PREFERRED_SIZE, 279)
        .addContainerGap());
```

---

- 94 -
_SIZE, Short.MAX_VALUE))

    jDialog3Layout.setVerticalGroup(
        jDialog3Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
        .addGroup(jDialog3Layout.createSequentialGroup()
            .addContainerGap(16, Short.MAX_VALUE)
            .addComponent(jLabel5)
            .addPreferredGap(18, 18, 18)
            .addComponent(jButton3)
            .addPreferredGap()
        );
    }

    setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
    setTitle("HMM Filter Implementation");
    setCursor(new java.awt.Cursor(java.awt.Cursor.DEFAULT_CURSOR));
    addMouseListener(new java.awt.event.MouseAdapter()
        {  
        public void mouseClicked(java.awt.event.MouseEvent e)  
            formMouseClicked(evt);
        });

// getContentPane().setLayout(new org.netbeans.lib.awtextra.AbsoluteLayout());
// getContentPane().add(jLabel2, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 60, 360, 20));
// getContentPane().add(jLabel1, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 30, 370, 20));
// getContentPane().add(jLabel3, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 90, 370, 20));
// getContentPane().add(jLabel4, new org.netbeans.lib.awtextra.AbsoluteConstraints(20, 120, 380, 20));

    jMenu1.setText("File");
    jMenu1.addMenuListener(new javax.swing.event.MenuListener()
        {  
        public void menuDeselected(javax.swing.event.MenuEvent e)  
            jMenu1MenuDeselected(evt);
        }
    );

    public void menuDeselected(javax.swing.event.MenuEvent event) {
        jMenu1MenuDeselected(evt);
    }

    public void menuCanceled(javax.swing.event.MenuEvent event) {
        jMenu1MenuCanceled(evt);
    }
public void menuSelected(javax.swing.event.MenuEvent evt) {
    jMenu1MenuSelected(evt);
}

miOpen.setText("Open");
miOpen.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        miOpenActionPerformed(evt);
    }
});

jMenu1.add(miOpen);

miDeviation.setText("Calculate deviation");
miDeviation.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        miDeviationActionPerformed(evt);
    }
});

jMenu1.add(miDeviation);

miNoise.setText("Set Noise Parameters");
miNoise.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        miNoiseActionPerformed(evt);
    }
});

jMenu1.add(miNoise);

HMMParameters.setText("Set Transition Probabilities");
HMMParameters.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        HMMParametersActionPerformed(evt);
    }
});
HMMParametersActionPerformed(evt);

jMenu1.add(HMMParameters);

miClose.setAccelerator(javax.swing.KeyStroke.getKeyStroke(java.awt.event.KeyEvent.VK_F4, java.awt.event.InputEvent.ALT_MASK));

miClose.setText("Close");

miClose.addActionListener(new java.awt.event.ActionListener() {
    public void actionPerformed(java.awt.event.ActionEvent evt) {
        miCloseActionPerformed(evt);
    }
});

jMenu1.add(miClose);

jMenuBar1.add(jMenu1);

setJMenuBar(jMenuBar1);

pack();

private void miCloseActionPerformed(java.awt.event.ActionEvent evt) {
    System.exit(0);
}

private static double calcDist(double p1, double p2) {
    return Math.sqrt(Math.pow(p1-p2, 2));
}

private void miDeviationActionPerformed(java.awt.event.ActionEvent evt) {
    // GEN-FIRST:event_miDeviationActionPerformed
    System.exit(0);
    // GEN-LAST:event_miDeviationActionPerformed

    public static double calcDist(double p1, double p2) {
        return Math.sqrt(Math.pow(p1-p2, 2));
    }

    private void miDeviationActionPerformed(java.awt.event.ActionEvent evt) {
        // GEN-FIRST:event_miDeviationActionPerformed
        System.exit(0);
        // GEN-LAST:event_miDeviationActionPerformed

        // υπολογίζεται η απόκλιση του θορύβου, HMM filter από την πραγματική τιμή
        double deviationFromNoise = 0.0, deviationFromHMM = 0.0;
int num = num_measurements;

for (int i = 0; i < num; i++) {
    deviationFromNoise += calcDist(real_measurements[i], noise_measurements[i]);
    deviationFromHMM += calcDist(real_measurements[i], HMM_measurements[i]);
}

String message = "Deviation between noise and real measurements: " + deviationFromNoise / num + "\n";
message += "Deviation between HMM filter results and real measurements: " + deviationFromHMM / num + "\n";
JOptionPane.showMessageDialog(null, message, "Calculated deviation values", JOptionPane.INFORMATION_MESSAGE);
}

private void miOpenActionPerformed(java.awt.event.ActionEvent evt) {
    // Dialog box to open measurements file
    int returnVal = fileChooser.showOpenDialog(this);
    int i=0;
    String display_real_measurements, display_noise_measurements, temp;
    display_real_measurements=new String("measurements: ");
    display_noise_measurements=new String("measurements: ");

    temp=new String();

    if (returnVal == JFileChooser.APPROVE_OPTION) {
        // Clear screen
        screen.setColor(Color.white);
        screen.fillRect(0, 0, width, height);
        screen.setColor(Color.black);
}
repaint();

File file = fileChooser.getSelectedFile();
	ry {
    FileReader fr = new FileReader(file.getAbsolutePath());
    LineNumberReader lnreader = new LineNumberReader(fr);
    String line = new String();
    String[] result = new String[]();
    double x, noiseX;
    while ((line = lnreader.readLine()) != null) {
        result = line.split("\s");
        real_measurements[i] = Integer.parseInt(result[0]);
        temp = String.valueOf(real_measurements[i]);
        display_real_measurements = display_real_measurements + temp + " ";
        noise_measurements[i] = Integer.parseInt(result[1]);
        temp = String.valueOf(noise_measurements[i]);
        display_noise_measurements = display_noise_measurements + temp + " ";
        i++;
    }
    num_measurements = i;
    jLabel1.setText("Real").concat(display_real_measurements);
    jLabel2.setText("Noise").concat(display_noise_measurements);
    toggleHMMs();
} catch (IOException ex) {
    System.out.println("problem accessing file" + file.getAbsolutePath());
}
} else {
System.out.println("File access cancelled by user.");

miNoise.setEnabled(true);

//GEN-LAST:event_miOpenActionPerformed

private void HMMParametersActionPerformed(java.awt.event.ActionEvent evt) {
    jDialog1.setVisible(rootPaneCheckingEnabled); // TODO add your handling code here:
    jDialog1.setSize(487, 250);
    jDialog1.setTitle("Transition probabilities");
} //GEN-LAST:event_HMMParametersActionPerformed

private void fileChooserActionPerformed(java.awt.event.ActionEvent evt) {
    // TODO add your handling code here:
} //GEN-LAST:event_fileChooserActionPerformed

private void jMenu1MenuDeselected(javax.swing.event.MenuEvent evt) {
    // TODO add your handling code here:
    repaint();
} //GEN-LAST:event_jMenu1MenuDeselected

private void formMouseClicked(java.awt.event.MouseEvent evt) {
    // TODO add your handling code here:
} //GEN-LAST:event_formMouseClicked

private void jButton4ActionPerformed(java.awt.event.ActionEvent evt) {
    jDialog1.dispose();
} //GEN-LAST:event_jButton4ActionPerformed

private void jButton1ActionPerformed(java.awt.event.ActionEvent evt) {
    // Read transition probabilities
    int x, y;
    String temp;
for(x=0;x<num_states;x++)
    for(y=1;y<=num_states;y++)
    {
        temp=(jTable1.getValueAt(x,y)).toString();
        transition_probabilities[x][y-1]=Double.valueOf(temp).doubleValue();
    }
    jDialog1.dispose();
}//GEN-LAST:event_jButton1ActionPerformed

    // Read noise probabilities
    int x, y;
    String temp;
    for(x=0;x<num_states;x++)
        for(y=1;y<=num_states;y++)
        {
            temp=(jTable2.getValueAt(x,y)).toString();
            noise_probabilities[x][y-1]=Double.valueOf(temp).doubleValue();
        }
        calculate_state_probabilities();
        jDialog2.dispose();
    }
    jDialog2.dispose();
}//GEN-LAST:event_jButton5ActionPerformed

    // TODO add your handling code here:
    jDialog2.dispose();
}//GEN-LAST:event_jButton8ActionPerformed

    // TODO add your handling code here:
    jDialog2.setVisible(rootPaneCheckingEnabled);
    jDialog2.setSize(487, 250);
}//GEN-LAST:event_miNoiseActionPerformed
jDialog2.setTitle("Set Noise Parameters");

private void jButton2ActionPerformed(java.awt.event.ActionEvent evt) {
    // Check box to enable exhaustive search decoding or Viterbi decoding
    jButton1ActionPerformed(evt);
    if (jCheckBox1.isSelected() == true)
    {
        calculateHMM_Viterbi();
    }
    else
    {
        calculateHMM();
    }
}

private void jMenu1MenuSelected(javax.swing.event.MenuEvent evt) {
    // TODO add your handling code here:
    if (file_opened == 0)
    {
        miNoise.setEnabled(false);
        HMMParameters.setEnabled(false);
        file_opened = 1;
    }
}

private void jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
    jDialog3.dispose();
}

void calculate_state_probabilities()
{
    // Set HMM state probabilities
```java
int x, y;
double temp = 0;

for (y = 0; y < num_measurements; y++) {
    for (x = 0; x < num_states; x++) {
        state_probabilities[x][y] = noise_probabilities[noise_measurements[y]][x];
    }
}
HMMParameters.setEnabled(true);

void toggleHMMs() {
    // Can be used to hide or show the HMM filter series. Currently unused
    if (toggled == 0) {
        jLabel3.setText("Filtered measurements: ");
        jLabel4.setText("Filtered measurements (Viterbi): ");
    }
    toggled = 1;
} else {
    toggled = 0;
    jLabel3.setText(null);
    jLabel4.setText(null);
}

void calculateHMM_Viterbi() {
    // Viterbi decoding
    int counter1, counter2, counter3;
    double tmp[2][2], temp = 0.0, max_value = 0.0, temp1, temp2, temp3;
```
int output_path[], current_path[];

output_path=new int[num_measurements]; //The
current_path=new int[num_measurements];
tmp=new double[num_states][num_measurements];

String path=new String();
long start = System.nanoTime();

for(counter1=0;counter1<num_measurements;counter1++) //I
initializations
{
    output_path[counter1]=-1;
    current_path[counter1]=-1;
    for(counter2=0;counter2<num_states;counter2++)
    {
        best_path[counter2][counter1]=-1;
        tmp[counter2][counter1]=state_probabilities[counter1];
    }
}

for(counter1=1;counter1<num_measurements;counter1++) //I
identify the best path to each state for each measurement
{
    for(counter2=0;counter2<num_states;counter2++)
    {
        max_value=0.0;
        for(counter3=0;counter3<num_states;counter3++)
        {
            temp=tmp[counter3][counter1-1]*transition_probabilities[counter3][counter2];
        
        if(temp>max_value)
        {
            max_value=temp;
            best_path[counter2][counter1]=counter3;
            //Memorize the previous node for the best path to the current on
e

} 

} 

tmp[counter2][counter1] *= max_value;

}

}

max_value = 0;

temp = -1;

for (counter2 = 0; counter2 < num_states; counter2++) // Find the best path for the st of paths to each state of the final measurement
{
    counter1 = num_measurements - 1;
    current_path[counter1] = counter2;
    temp3 = state_probabilities[counter2][counter1];

    while ((best_path[current_path[counter1]][counter1] > -1) && (counter1 > 0))
    {
        temp1 = state_probabilities[best_path[current_path[counter1]][counter1]][counter1-1];
        temp2 = transition_probabilities[best_path[current_path[counter1]][counter1]][current_path[counter1]]; 

        temp = temp1 * temp2 * temp3;
        current_path[counter1-1] = best_path[current_path[counter1]][counter1];
        counter1 --
    }

}

if ((temp > max_value) && (counter1 == 0))
{
    max_value = temp;
    for (counter3 = 0; counter3 < num_measurements; counter3++)
    {

output_path[counter3]=current_path[counter3];

HMM_measurements[counter3]=output_path[counter3];

for(counter1=0;counter1<num_measurements;counter1++)
{
    path=path+String.valueOf(output_path[counter1]);
    path=path+" ";
}

jLabel4.setText("Filtered measurements (Viterbi): "+ path);

double elapsedTime = System.nanoTime()-start;
elapsedTime/=Math.pow(10,6);

jDialog3.setVisible(rootPaneCheckingEnabled);
jDialog3.setSize(200, 120);
jDialog3.setTitle("Viterbi decoder");
jLabel5.setText("Running time: "+String.valueOf(elapsedTime)+" msec");
    System.out.println(elapsedTime);
}

void calculateHMM() {
    //Exhaustive search decoding
    long counter, num_paths;

double max_value=0, temp=1.0, temp1, temp2;
int output_path[], current_path[],counter1;
boolean updated;
String path=new String();
output_path= new int[num_measurements];
current_path= new int[num_measurements];

long start = System.nanoTime();

num_paths=(long) Math.pow(num_states, num_measurements);
// The number of all paths to exhaustively search

for(counter1=0; counter1<num_measurements; counter1++)
{
current_path[counter1]=0;
}

for(counter=0; counter<num_paths; counter++) // Search each path and find the best one
{
  for(counter1=0; counter1<num_measurements; counter1++)
  {
    if(counter1<num_measurements-1)
    {
      temp1=state_probabilities[current_path[counter1]][counter1];
      temp2=transition_probabilities[current_path[counter1]][current_path[counter1+1]];
      temp*=temp1*temp2;
    }
    else
    {
      temp1=state_probabilities[current_path[counter1]][counter1];
      temp*=temp1;
    }
  }

  if(temp>=max_value) // If true, this path is currently the best path
  {
    max_value=temp;
    for(counter1=0; counter1<num_measurements; counter1++)
    {
      output_path[counter1]=current_path[counter1];
      HMM_measurements[counter1]=current_path[counter1];
    }
  }
}
temp = 1.0;
counter1 = num_measurements - 1;

while (current_path[counter1] + 1 >= num_states) // calculate the next path to check
{
    current_path[counter1] = 0;
    counter1--;
    if (counter1 < 0)
        break;
}

if (counter1 >= 0)
current_path[counter1]++;

for (counter1 = 0; counter1 < num_measurements; counter1++)
{
    path = path + String.valueOf(output_path[counter1]);
    path = path + " ";
}

jLabel3.setText("Filtered measurements:" + path);

double elapsedTime = System.nanoTime() - start;
elapsedTime /= Math.pow(10, 6);

jDialog3.setVisible(rootPaneCheckingEnabled);
jDialog3.setSize(230, 120);
jDialog3.setTitle("Exhaustive search decoder");

jLabel5.setText("Running time: " + String.valueOf(elapsedTime) + " msec");

System.out.println(elapsedTime);

@Override
public void update(Graphics g) {
public void paint(Graphics g) {
    super.paint(g);
    //update(g);
}

/**
 * @param args the command line arguments
 */
public static void main(String[] args) {
    java.awt.EventQueue.invokeLater(new Runnable() {
        public void run() {
            new HMMFilter().setVisible(true);
        }
    });
}
// Variables declaration - do not modify//GEN-BEGIN:variables
private javax.swing.JMenuItem HMMParameters;
private javax.swing.JFileChooser fileChooser;
private javax.swing.JButton jButton1;
private javax.swing.JButton jButton2;
private javax.swing.JButton jButton3;
private javax.swing.JButton jButton4;
private javax.swing.JButton jButton5;
private javax.swing.JButton jButton8;
private javax.swing.JCheckBox jCheckBox1;
private javax.swing.JDialog jDialog1;
private javax.swing.JDialog jDialog2;
private javax.swing.JDialog jDialog3;
private javax.swing.JFileChooser jFileChooser1;
private javax.swing.JInternalFrame jInternalFrame1;
private javax.swing.JLabel jLabel1;
private javax.swing.JLabel jLabel2;
private javax.swing.JLabel jLabel3;
private javax.swing.JLabel jLabel4;
private javax.swing.JLabel jLabel5;
private javax.swing.JMenu jMenu1;
private javax.swing.JMenuBar jMenuBar1;
private javax.swing.JPanel jPanel1;
private javax.swing.JScrollPane jScrollPane1;
private javax.swing.JScrollPane jScrollPane2;
private javax.swing.JTable jTable1;
private javax.swing.JTable jTable2;
private javax.swing.JMenuItem miClose;
private javax.swing.JMenuItem miDeviation;
private javax.swing.JMenuItem miNoise;
private javax.swing.JMenuItem miOpen;

// End of variables declaration//GEN-END:variables