UNIVERSITEIT VAN PRETORIA
UNIVERSITY OF PRETORIA
YUNIBESITHI YA PRETORIA

A COMPARATIVE STUDY OF MULTIPLE HYPOTHESIS AND VITERBI BASED TRACK STITCHING

by

Lodewyk Johannes van der Merwe

Submitted in partial fulfilment of the requirements for the degree

Master of Engineering (Electronic Engineering)

in the

Department of Electrical, Electronic and Computer Engineering
Faculty of Engineering, Built Environment and Information Technology
UNIVERSITY OF PRETORIA

April 2016

ACKNOWLEDGEMENTS

The author would like to thank the Armaments corporation of South Africa (ARMSCOR) and the South African national defence force (SANDF) for financial support during this study through the LEDGER initiative via the Council for Scientific and Industrial Research (CSIR).

The author would also like to thank Dr. Pieter de Villiers for continued support and guidance throughout the study, without whom this work would not have been possible.

Finally the author would like to thank the organisers and paper reviewers of the 16th International conference on Information FUSION for the opportunity to present this work at the conference in Istanbul, Turkey.

SUMMARY

A COMPARATIVE STUDY OF MULTIPLE HYPOTHESIS AND VITERBI BASED TRACK STITCHING

by

Lodewyk Johannes van der Merwe

Supervisor(s):	Dr. J.P. de Villiers
Department:	Electrical, Electronic and Computer Engineering
University:	University of Pretoria
Degree:	Master of Engineering (Electronic Engineering)
Keywords:	Track stitching, track fragments, sequential Viterbi, multiple hypothesis.

The track stitching problem can be divided into two tasks, namely target tracking to create track fragments, and track stitching to stitch the resulting track fragments together. In this study a tracking algorithm is developed to track multiple targets and to create track fragments. The track fragments are generated through the use of a Markov model where a target can be assumed to be in an occluded or a visible state. The track fragments are modelled as nodes in a state diagram, which in turn is used to create a time dependent trellis diagram. The columns in the trellis diagram represents the time when a particular track fragment appeared while the nodes represent certain kinematic features of the track fragments themselves. This trellis is then solved using a sequential Viterbi algorithm and using each node exactly once, to obtain the most viable track fragment to track fragment associations. Each solution path through the trellis diagram represents a set of track fragments which where created by a specific target. Results are presented by simulating multiple crossing targets with fragmented tracks. It is shown that the algorithm successfully stitches track fragments together in the presence of false tracks caused by noisy observations. Further evaluation of the performance of the algorithm is presented with a number of scenarios where increasingly ambiguous track fragment to track fragment associations exist. The sequential Viterbi track stitching algorithm is also compared against a multiple hypothesis track fragment stitching algorithm. This algorithm was extended from the classic hypothesis based multiple hypothesis tracking algorithm (MHT) by, performing track fragment to track associations instead of observation to track associations. In these comparisons, the code used to run

the simulations have been optimised and vectorised as far as possible to ensure a fair comparison. It is shown that the Viterbi based track stitching algorithm performs somewhat better that the multiple hypothesis track stitching algorithm for similar execution times. The Viterbi based track stitching algorithm is also shown to produce more consistently acceptable results.

SAMEVATTING

'N VERGELYKENDE STUDIE VAN VEELVULDIGE-HIPOTESE VOLGERSEGMENTASSOSIASIE EN VITERBI-GEBASEERDE VOLGERSEGMENTASSOSIASIE

deur

Lodewyk Johannes van der Merwe

Studieleier(s):	Dr J.P. de Villiers
Departement:	Elektriese, Elektroniese en Rekenaar-Ingenieurswese
Universiteit:	Universiteit van Pretoria
Graad:	Magister in Ingenieurswese (Elektroniese Ingenieurswese)
Sleutelwoorde:	Volgersegmentsamevoeging, volgersegmente, Viterbi-algoritme, veelvuldige-hipotese volgeralgorithme.

Die volgersegmentsamevoegingprobleem kan verdeel word in twee hooftake, naamlik teikenvolging, waar die volgersegmente geskep word en volgersegmentstikking, waar die geskepte segmente met mekaar geassosieer word. In hierdie studie word 'n volgeraloritme ontwikkel om veelvuldige teikens te volg en om volgersegmente te skep. Die volgersegmente word geskep deur gebruik te maak van 'n Markov-model waar 'n teiken in 'n versteekte of 'n sigbare toestand kan wees. Die volgersegmente word gemodelleer as nodes in 'n toestandsdiagram, wat omgeskakel word in 'n tyd-afhanklike latwerkdiagram. Die kolomme in hierdie diagram stel die tyd voor wanneer 'n spesifieke volgersegment sigbaar geword het, terwyl die nodes die volgersegmente voorstel. Hierdie latwerk diagram word dan opgelos deur gebruik te maak van 'n sekwensiële Viterbi-algoritme waar elke node slegs een keer in die oplossing gebruik word om die mees waarskynlike paaie te verkry. Elke pad deur die diagram stel 'n groep volgersegmente voor wat deur 'n spesifieke teiken gegenereer is. Resultate word voorgelê deur veelvuldige kruisende teikens te genereer waarvan die teikenwaarnemings yl is. Daar word aangetoon dat die algoritme die volgersegmente korrek met mekaar assosieer in die teenwoordigheid van vals volgersegmente wat deur valsalarmwaarnemings veroorsaak is. Verdere evaluering, volgens die prestasie van die algoritme, word aangetoon deur 'n aantal gevalle te simuleer waar die assosiasies tussen die volger segmente meer dubbelsinning is. Die opeenvolgende

Viterbi-volgersegmentstikkingalgorithme word laastens vergelyk met ’n veelvuldige-hipotese volgersegmentstikkingalgorithme wat uitgebrei is uit die hipotese-gebaseerde veelvuldige-hipotese volgeralgoritme om segment met segment te assosieer in plaas daarvan om waarnemings met volgers te assosieer. In hierdie vergelykings is die kode gebruik is om die simulasies uit te voer so ver moontlik geoptimeer om sodoende ’n eerlike vergelyking te kan tref. Die Viterbi-gebaseerde algoritme vaar ietwat beter as die veelvuldige-hipotese algoritme in gevalle waar die uitvoertye soortgelyk is. Die Viterbi-gebaseerde algoritme blyk ook meer konsekwent te wees, en lewer dus meer aanvaarbare resultate.

LIST OF ABBREVIATIONS

BP	Belief propagation
FA	False alarm
FAR	False alarm rate
GBT	Graph based tracking
GNN	Global nearest neighbour
HMM	Hidden Markov model
IMM	Interactive multiple model
KF	Kalman filter
MC	Markov chain
MHT	Multiple hypothesis tracking
MHTS	Multiple hypothesis track stitching
MTT	Multiple target tracking
NN	Nearest neighbour
NT	New track
PRT	Pulse repetition time
RMSE	Root mean squared error
TS	Track stitching

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

"The mere formulation of a problem is far more often essential than its solution, which may be merely a matter of mathematical or experimental skill."

Albert Einstein (1879-1955)

1.1 PROBLEM STATEMENT

1.1.1 Background and context

The multiple target tracking (MTT) problem has been encountered in a variety of environments. Producing tacks from targets of interest using raw sensor observations has applications in numerous areas including military target tracking, guidance systems, civilian surveillance and threat assessment applications.

The track stitching problem is inherent to many multiple target tracking (MTT) environments, where occlusions and ambiguous possible track to track associations arise. This problem can be divided into two parts, namely tracking and track stitching [1]. The track of a target can be broken due to a number of reasons, while a target is being tracked. These include a sufficient number of missed updates, due to a low sensor detection probability, target occlusions or in cases where the sensor update period might be long. In these cases the target track will usually be dropped by the tracking algorithm and re-initiated at a later time when a sufficient number of updates are received. This results in poor long term track management and in turn affects the performance of higher level algorithms such as threat assessment, target classification, resource assignment and guidance algorithms. The total number of track fragments and therefore, the total number of possible track fragment to track fragment associations scales exponentially with the number of fragments and the tracking time [2],

thereby increasing the computational complexity of the problem further.

1.1.2 Research motivation

Typically, the total number of track fragments increases as the number of targets increase. The number of track fragments also typically increases in time and with the size of the tracking window. The tracking window is the period of time of interest while observing the targets as well as the spacial domain of the tracking system [3]. As with many data association problems the amount of associations results in a combinatorial explosion [4], [2]. In this study a sequential Viterbi based [5] track stitching algorithm is developed and is compared to a multiple hypothesis [6] track stitching algorithm. The multiple hypothesis track stitching algorithm was extended from the classic hypothesis based multiple hypothesis track stitching algorithm, where instead of associating observations to tracks, track fragments are associated to tracks, in a manner similar to [7].

1.2 RESEARCH OBJECTIVE AND QUESTIONS

In this study the following research questions are answered:

- Can the track stitching problem be structured in such a way that a Viterbi based track stitching algorithm be used?
- Given the fact that the number of associations will invariably increase, can the problem of exponential dimensionality growth be mitigated somewhat?
- Is it possible to stitch track fragments together in real time, autonomously, using a Viterbi based track stitching algorithm, while simultaneously tracking multiple targets?
- How does the Viterbi based approach compare to a multiple hypothesis based approach in terms of complexity and efficiency?

1.3 RESEARCH OBJECTIVES

The objective of this study is to implement a multiple target tracking algorithm that performs track stitching in real time, using a sequential Viterbi track stitching algorithm, while also reducing the number of possible track fragment to track associations. This is achieved through the use of a sliding

window approach where unassociated or false track fragments are purged and those already associated to target tracks are permanently fused. The hypothesis based multiple hypothesis tracking (MHT) algorithm is extended to perform track fragment to track association instead of observation to track associations. The objective is then to perform a comparative investigation between the sequential Viterbi track stitching algorithm and the multiple hypothesis track stitching algorithm both in terms of performance and execution time.

1.4 HYPOTHESIS AND APPROACH

In this study the track stitching problem is divided into two tasks. The true targets are occluded using a Markov model [8] where the target can either be in a visible or an occluded state. The targets are then tracked to create the track fragments. The resulting track fragments are stitched to create longer tracks through the use of either a sequential Viterbi track stitching algorithm or a multiple hypothesis track stitching algorithm. Confirmed associations are permanently fused, while those track fragments which are unlikely to be associated to other tracks are purged to reduce the computational complexity. Both the tracking, track stitching as well as the fusing and purging tasks are performed in real time. A comparison between the novel Viterbi based approach and multiple hypothesis based approach will be performed through Monte Carlo simulations. It is conjectured that the result of performing track stitching will result in longer tracks, which in turn will be useful in applications such as threat assessment, target classification, resource assignment and guidance algorithms.

The tracking and track stitching algorithms were developed to work in tandem in a simulation environment. The tracking algorithm deals with tracking targets by producing tracks from raw sensor data, while storing selective information about the track fragments. The track fragment algorithm stitches the track fragments together in real time using the selective information from the tracking algorithm. The track stitching also employs the use of an N-scan algorithm to purge or fuse track fragments based on previous associations or the lack thereof. In the simulation, various multiple target tracking scenarios in the presence of false detections were created to compare the performance between the sequential Viterbi track stitching algorithm and the multiple hypothesis track stitching algorithm and running Monte Carlo (MC) simulations for the different scenarios.

1.5 RESEARCH CONTRIBUTION

The contribution that this research makes to the field of track stitching are in providing a framework to model the track fragments as nodes in a lattice or trellis structure. A sequential Viterbi algorithm is used perform track-to-track stitching in a multiple target tracking environment. The use of the Viterbi algorithm to perform track stitching has not been found in the literature. The proposed track stitching algorithm with the the joint use of a graphical model and a Viterbi data association algorithm has novel aspects and is of interest.

1.6 OUTLINE OF THE DISSERTATION

The rest of this dissertation is organised as follows: In Chapter 2 a literature study is presented that summarises the previous advancements pertaining to the track stitching problem. Important aspects relating to this study are also presented. In Chapter 3 the methods used to solve the track stitching problem are presented and implementation details are discussed. In Chapter 4 the results of the implemented algorithms discussed in Chapter 3 are presented and discussed. Finally, in Chapter 5 conclusions and areas to expand on this work are presented.

1.7 PUBLICATIONS AND SUBMISSIONS

1.7.1 Conference Proceedings

The following paper was presented at the 16th international conference on Information fusion in Turkey, Istanbul in 2013:

1. Van der Merwe, L.J., De Villiers, J.P., "Track-stitching using graphical models and message passing", *16th International Conference on information Fusion (FUSION), 2013*, vol., no., pp.758-765, 9-12 July 2013.

1.7.2 Journal submissions

The following paper was submitted to the IET Radar, Sonar & Navigation journal and has been accepted for publication:

1. Van der Merwe, L.J., De Villiers, J.P., "A comparative investigation into graph based and multiple hypothesis based track stitching", *Accepted for publication.*

In the next chapter an overview of previously conducted research in the field of target track stitching is presented. Important concepts that pertain to this study are also discussed.

CHAPTER 2

LITERATURE STUDY

"Study the past, if you wish to define the future."

Confucius (551-479 BC)

2.1 CHAPTER OVERVIEW

In this chapter an overview of previous advancements made in an effort to solve the track stitching problem are discussed. Concepts pertaining to this study are discussed thereafter. Previous approaches that are different or similar to this study are also highlighted.

2.2 PREVIOUS EFFORTS

The track stitching or track to track association problem entails matching pieces of tracks or track fragments to each other track to create a larger track. The track fragments composing the larger track are believed to have all originated from the same target. These track fragments are sometimes referred to as "tracklets" in the literature [9]. However, in data fusion literature "tracklets" typically have a different meaning. They refer to multiple track segments that have originated from different sensors that need to be fused centrally. In this study track fragments refer to segments of a track that may have originated from a common target.

Numerous contributions in the literature have addressed the track stitching problem using various methods. In [10], [11] and [12] rank minimisation of a Hankel matrix (formed by combining the known and missing data) is used to estimate the missing values between two segments of a track, that are maximally consistent with the known data. In this study the focus is rather on the association of track fragments, and the missing estimated target positions are linearly interpolated between two associated track fragments after track fragment to track fragment association has been performed.

A Kalman filter (KF) [13] based approach has also been proposed in [14] to estimate the global track states. In this approach each track is tested to determine whether it should be stitched to an existing global track or if a new global track should be initialised. A list of the stitched tracks and the observations from all track fragments are used to update then Kalman filter. In this study a Kalman filter is used in the tracking of the targets and to create the tracks from the sensor observations along with an M/N track maintenance algorithm. A coasting Kalman filter is also used to determine which track fragments can potentially associated to each other through the use of an ellipsoidal gate.

In [2] an N-scan approach is used along with a graphical model to accomplish multiple target tracking. The nodes in this model are modelled according to a node potential dependent on the probability of detection and the probability of a false alarm detection. A belief propagation algorithm is used in [2] to reduce the computational overhead incurred when updating the states of the nodes in a graphical model.

In [15] a Bayesian network is constructed based on the track fragments and then solved to obtain the most likely target tracks. A pruning algorithm [4] is used to limit the number of connections in the Bayesian network, thereby reducing the complexity.

An IMM (Interacting Multiple Model) approach to determine the underlying model is used in [16] along with a prediction and retrodiction approach with gating to obtain the most likely track fragment associations. Here track deletion is used to terminate low quality tracks based on the last update time of the track [17], [18].

A flow network approach is used in [19] to model tracks as nodes. The transition from one node to the next is governed by a likelihood equation. The most likely paths through the flow network corresponds to the most likely sequences of global tracks.

More recently a track fragment partitioning approach based on time information and a multiple hypothesis filter has been used by [20] to stitch track fragments together to aid in guidance systems, while in [9] the track fragment to track fragment association task is modelled using a track stitching graph extended from graph based tracking (GBT) algorithm, thereby reducing the task to a maximum weight bipartite matching problem.

In this study the track fragments are modelled as nodes in a trellis diagram and subsequently solved using a Viterbi algorithm [21], [5], and is compared to a multiple hypothesis based approach. The

solution paths through this trellis provides the most likely sequences of track fragments representing the stitched target tracks.

2.3 CONCEPTS PERTAINING TO THIS STUDY

In this section important concepts pertaining to this study are reviewed, including radar principles, the target tracking problem, Markov and hidden Markov models, Bayesian estimation and the Kalman filter, the data association problem, observation gating and belief propagation algorithms.

2.3.1 Radar principles

In this subsection the radar principles important to the study are reviewed. These concepts include false alarm returns, the probability of detection for a given radar sensor and observation gating.

2.3.1.1 False alarm rate

In this study the the false alarm rate (FAR) is used to model noise in the radar system. The false alarm rate is defined [3] as the ratio of the false detections per radar scan over all the range cells, given by,

$$FAR = \frac{false\ detections\ per\ scan}{unit\ volume}. \tag{2.1}$$

In practice, false alarms (FA) can be caused by thermal noise exceeding a pre-determined detection threshold value, by the presence of spurious, non-persistent signals or by equipment malfunction. The detection threshold value must be set low enough to allow the detection of valid targets, but high enough to reduce the number of false alarms so that these do not mask the valid target returns.

In general the distribution of the false detections can be assumed to be uniform over the scanning area, while the arrival of false detections in each scan follows a Poisson distribution [22] [17]. In this study the FAR is effectively represented by these two distributions. The parameter β_{FA} is used to control the false alarm rate and ultimately the amount of false detections produced by the Poisson distribution.

2.3.1.2 Probability of a detection

The probability of detection, P_D is defined as the probability that the radar system will detect a return from a target, given that the target is present. In this study the probability of detection is simply used to provide a quantitative estimation of the quality of the sensor and of the target being detected. Given a low probability of detection the radar system may miss a return from a present target at any given time step, therefore hampering the tracking performance.

2.3.2 Single target tracking

Single target tracking forms the basis of this study and the concepts are briefly discussed here.

2.3.2.1 True target dynamics

In this study an almost constant velocity model [23] is used to model target movement. The constant velocity model is given by

$$\begin{aligned} \boldsymbol{x}_k &= \Phi \boldsymbol{x}_{k-1} + \boldsymbol{a}_k \boldsymbol{B} \\ &= \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_{k-1} + \boldsymbol{a}_k \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}. \end{aligned} \tag{2.2}$$

In equation 2.12, $\boldsymbol{a}_k$ is a 2×1 acceleration noise vector sampled from the zero mean Gaussian distribution, $\mathcal{N}(0, \sigma_a^2)$ and T is the sampling period. The format of the state variable $\boldsymbol{x}_k$ is in the form

$$\boldsymbol{x}_k = \begin{bmatrix} x_k \\ y_k \\ \dot{x_k} \\ \dot{y_k} \end{bmatrix}, \tag{2.3}$$

where the x_k and y_k denote the Cartesian position coordinate and $\dot{x_k}$ and $\dot{y_k}$ denote the velocity of the target in Cartesian coordinates. The values of x_k and y_k denote the height above the sensor and the

distance along the surface of the earth from the sensor. In this study the sensor is assumed to be at position $(0,0)$ in the $x-y$ Cartesian plane. This model results in a smooth target motion.

2.3.2.2 Radar observation generation

Radar observations are often times received in the polar plane (range and azimuth) [23]. In this study noise is added to the true target positions in the polar plane through the equations

$$\tilde{r}_k = \sqrt{(x_k^2 + y_k^2)} + n_{ra}, \tag{2.4}$$

$$\tilde{\eta}_k = \arctan\left(\frac{y_k}{x_k}\right) + n_{az}, \tag{2.5}$$

where n_{ra} and n_{az} represent the sensor noise in range and azimuth respectively and are drawn from the zero mean Gaussian distributions $\mathcal{N}(0,\sigma_{ra}^2)$ and $\mathcal{N}(0,\sigma_{az}^2)$.

The noisy polar coordinate observations are now converted back to Cartesian coordinates in order to perform target tracking, assuming that the radar sensor is placed at the origin of the Cartesian plane. The conversion is given by

$$\boldsymbol{z}_k = \begin{bmatrix} \tilde{x}_k \\ \tilde{y}_k \end{bmatrix} = \begin{bmatrix} \tilde{r}_k \cos(\tilde{\eta}_k) \\ \tilde{y}_k \sin(\tilde{\eta}_k) \end{bmatrix}. \tag{2.6}$$

In this study, these noisy observations are used as input to the target tracking algorithm.

2.3.3 Markov models and hidden Markov models

In this study extensive use is made of the Markov property. The discussion commences with Markov processes and proceeds on to hidden Markov models.

Markov processes [24],[25] are characterised by the fact that the current state in the process is only dependent on the previous state. A Markov chain model of length $n+1$, is denoted by a directed graph, with arrows indicating state transitions and circles indicating nodes, as shown in Figure 2.1.

The joint distribution of a sequence of N observations under this model is given by

$$p(\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_2,\ldots,\boldsymbol{x}_N) = p(\boldsymbol{x}_1)\prod_{n=2}^{N} p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}). \tag{2.7}$$

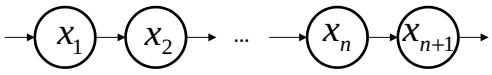

Figure 2.1: A Markov chain of length of length N. Arrows indicate transitions between states, while nodes indicate the states.

Through the d-separation property, the conditional probability distribution of observation $\boldsymbol{x}_n$, given all the observations up to time $n-1$ is given by

$$p(\boldsymbol{x}_n|\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\ldots,\boldsymbol{x}_{n-1}) = p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1}). \tag{2.8}$$

For higher order models, the derivation is straight forward. The dependent states simply increase in a higher order model. The conditional probability distribution for observation $\boldsymbol{x}_n$ for an mth order Markov model is given by

$$p(\boldsymbol{x}_n|\boldsymbol{x}_1,\boldsymbol{x}_2,\boldsymbol{x}_3,\ldots,\boldsymbol{x}_{n-1}) = p(\boldsymbol{x}_n|\boldsymbol{x}_{n-1},\boldsymbol{x}_{n-2},\boldsymbol{x}_{n-3},\ldots,\boldsymbol{x}_{n-m}) \text{ where } n > m. \tag{2.9}$$

Hidden Markov models (HMM) [25], [26] are defined as Markov models where the the states of the model are unobserved (latent) [27]. These models have been used in a wide variety of applications, from speech recognition [28], [29] to gene prediction applications [30]. Lately, HMMs have increasingly been used in the multiple target tracking environment [3].

The hidden (latent) states of the HMM are observed through other variables which may be of a different dimensionality than the the latent variables. The hidden states can be inferred through the known observations as well as estimated previous states [25]. Figure 2.2 shows the general architecture for a HMM. The observations are shown as grey nodes, while the hidden states are shown as white nodes.

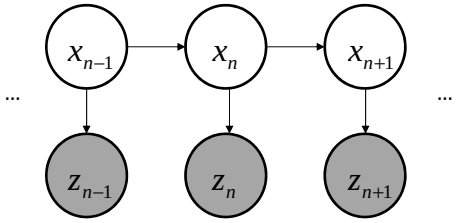

Figure 2.2: The general structure of a hidden Markov model. The hidden states (white nodes) are estimated through the observed states (grey nodes).

As can be seen from Figure 2.2 the conditional probability distribution of the hidden variable $\boldsymbol{x}_n$, given the values of the hidden variable $\boldsymbol{x}$ at all n, depends only on the value of the hidden variable $\boldsymbol{x}_{n-1}$. Similarly, the observed state $\boldsymbol{z}_n$ only depends on the hidden state $\boldsymbol{x}_n$ for all n.

The joint distribution for this model is given by,

$$p(\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_N, \boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N) = p(\boldsymbol{x}_1) \left[\prod_{n=2}^{N} p(\boldsymbol{x}_n | \boldsymbol{x}_{n-1}) \right] \prod_{n=1}^{N} p(\boldsymbol{z}_n | \boldsymbol{x}_n). \tag{2.10}$$

In Equation 2.10 the first multiplication is as a result of the chain structure, as in the first order Markov chain in Figure 2.1, while the second multiplication indicates the association between the latent and the observed variables.

2.3.4 Bayesian estimation

In a Bayesian estimation framework a hidden state $\boldsymbol{x}_n$ is observed through a noisy observation state $\boldsymbol{z}_n$ at step n that may be of a different dimensionality than $\boldsymbol{x}_n$. Estimation is performed in two discrete steps as depicted in Figure 2.3. First, the next state, $\boldsymbol{x}_{n+1}$, is predicted by the Bayesian estimator for time $n+1$. Secondly, The Bayesian estimator uses the next noisy observation, $\boldsymbol{z}_{n+1}$, to update the prediction. These two steps are carried out recursively.

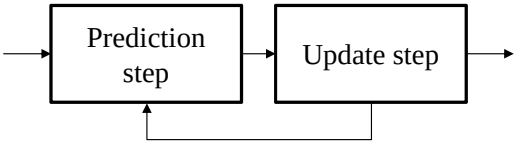

Figure 2.3: Bayesian estimation can be divided into two discrete steps, namely the prediction and the update step. These steps are performed recursively.

In a linear, Gaussian tracking environment the Bayesian estimation algorithm is realised using a Kalman filter. The Kalman filter also has the form of a hidden Markov model as depicted in Figure 2.2, where the states to be estimated, $\boldsymbol{x}_n$, are modelled as the hidden variables for all time, and the observed variables $\boldsymbol{z}_n$ are the noisy observations. In the case of a Kalman filter all state and observation functions are linear and all state and observation distributions are Gaussian.

The association between the latent states $\boldsymbol{x}_n$ and $\boldsymbol{x}_{n+1}$ is dependent on an underlying kinematic model while the association between the latent and the observed variable is dependent on an observation model.

2.4 TARGET TRACKING AND MODELLING

In this section the theory relating to target tracking an modelling is briefly discussed/ In this study a multiple target tracking algorithm is used to create target track fragments that may have originated from a common target.

2.4.1 Target modelling

A number of true targets are modelled first. The true targets have a state vector of the form $\boldsymbol{x}_k = [x_k, y_k, \dot{x}_k, \dot{y}_k]^\top$. In this notation x_k and y_k represents the Cartesian position of the target, while $\dot{x}_k$ and $\dot{y}_k$ represents the velocity of the target in the x and y Cartesian directions respectively. The general state space model given by [31],

$$\boldsymbol{x}_k = f(\boldsymbol{x}_{k-1}) + \boldsymbol{w}_k, \tag{2.11}$$

can be formulated as a constant velocity model [23], [17] to model the kinematic behaviour of the targets. This model is given as

$$\begin{aligned} \boldsymbol{x}_k &= \Phi \boldsymbol{x}_{k-1} + \boldsymbol{a}_k \boldsymbol{B} \\ &= \begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \boldsymbol{x}_{k-1} + \boldsymbol{a}_k \begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}. \end{aligned} \tag{2.12}$$

In equation 2.12, $\boldsymbol{a}_k$ is a 2×1 acceleration noise vector sampled from the zero mean Gaussian distribution, $\mathcal{N}(0, \sigma_a^2)$ and T is the sampling period.

2.4.1.1 Target occlusions and missed updates

To model occlusions, each point in the true target track, created by the model in 2.12 are assumed to be in one of two discrete states.

These states are denoted as visible or occluded. The state diagram in Figure 2.4 shows this behaviour. From Figure 2.4 it can be seen that the target can be either be in an occluded or a visible state. The transition probabilities in Figure 2.4 can be chosen in such a way that the model creates bursts

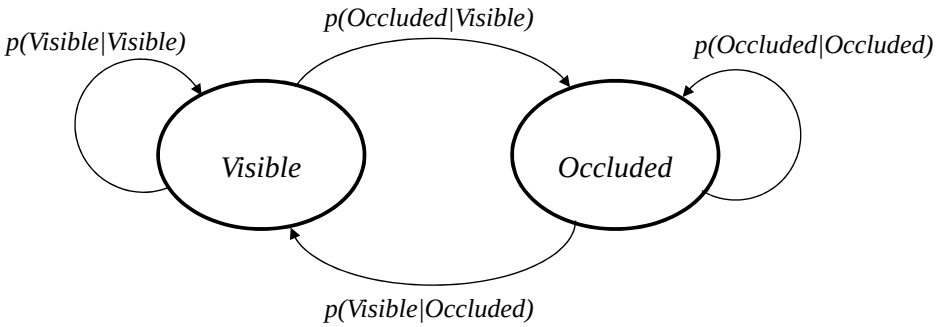

Figure 2.4: The target can be either visible or occluded. By choosing the transition probabilities appropriately the model will create bursts of true track fragments.

true track fragments. This implies that the transition probabilities between the visible and occluded states should be relatively low, when compared to the transition probabilities returning to the same node.

The transition probabilities are denoted as follows:

1. $p(V|V)$: the probability that the target remains in the visible state,
2. $p(V|O)$: the probability that the target moves from a visible state to an occluded state,
3. $p(O|V)$: the probability that the target moves from an occluded state to a visible state,
4. $p(O|O)$: the probability that the target remains in an occluded state.

The reliability of the sensor is further modelled by including a probability of detection, P_D, which determines whether an observation was received by the sensor or not.

2.4.2 Sensor modelling

The true target positions are next modelled as sensor observations in polar coordinates [32]. It is assumed that the sensor is placed at the origin of a Cartesian scanning area. Further it is assumed, that the sensor measurements are performed with one-standard-deviation accuracies of σ_{ra} and σ_{az} in range and azimuth respectively in polar coordinates.

The sensor observations for range and azimuth in equation 2.13 and equation 2.14 respectively, are generated by using a Cartesian to polar coordinate conversion. A tilde accent is used to do denote

these noisy sensor observations.

$$\tilde{r}_k = \sqrt{(x_k^2 + y_k^2)} + n_{ra} \tag{2.13}$$

$$\tilde{\eta}_k = \arctan\left(\frac{y_k}{x_k}\right) + n_{az} \tag{2.14}$$

Where n_{ra} and n_{az} represent the sensor noise in range and azimuth respectively and are drawn from the zero mean Gaussian distributions $\mathcal{N}(0, \sigma_{ra}^2)$ and $\mathcal{N}(0, \sigma_{az}^2)$.

The noisy polar coordinate observations are now converted back to Cartesian coordinates in order to perform target tracking. This results in the noisy Cartesian observation vector $\boldsymbol{z}_k$ given by equation 2.15.

$$\boldsymbol{z}_k = \begin{bmatrix} \tilde{x}_k \\ \tilde{y}_k \end{bmatrix} = \begin{bmatrix} \tilde{r}_k \cos(\tilde{\eta}_k) \\ \tilde{y}_k \sin(\tilde{\eta}_k) \end{bmatrix} \tag{2.15}$$

Owing to the conversion from polar to Cartesian coordinates in 2.15, the elements in $\boldsymbol{z}_k$ are not Gaussian [23], while the variables $\tilde{r}_k$ and $\tilde{\eta}_k$ are. The measurement can however be linearised through a first order expansion so that the elements of $\boldsymbol{z}_k$ are Gaussian, as a linear combination of a Gaussian random variable is also Gaussian. This results in the observation covariance matrix $\boldsymbol{R}$ for the observation $\boldsymbol{z}_k$ as

$$\boldsymbol{R} = \begin{bmatrix} \sigma_{xx}^2 & \sigma_{xy}^2 \\ \sigma_{xy}^2 & \sigma_{yy}^2 \end{bmatrix}. \tag{2.16}$$

Where the elements of the covariance matrix $\boldsymbol{R}$ in equation 2.16 are given by [23]

$$\sigma_{xx}^2 = \sigma_{ra}^2 \cos^2 \tilde{\eta}_k + \sigma_{az}^2 \tilde{r}_k^2 \sin^2 \tilde{\eta}_k, \tag{2.17}$$

$$\sigma_{yy}^2 = \sigma_{ra}^2 \sin^2 \tilde{\eta}_k + \sigma_{az}^2 \tilde{r}_k^2 \cos^2 \tilde{\eta}_k \text{ and} \tag{2.18}$$

$$\sigma_{xy}^2 = \frac{1}{2}\left(\sigma_{ra}^2 - \tilde{r}_k^2 \sigma_{az}^2\right) \sin 2\tilde{\eta}_k. \tag{2.19}$$

The noisy range, $\tilde{r}_k$, and azimuth, $\tilde{r}_k$, values are used since the true range ans azimuth values are typically not known. The observation covariance matrix is of particular importance in the filtering and gating algorithms used in this study.

2.4.3 Modelling of false detections

In this study it is assumed that all stationary persistent false detections has been removed. As a consequence only non-persistent random false detections are considered. It is assumed that the spacial distribution of the false detections over the scanning area follows a uniform distribution. The probability of a false alarm detection, however is modelled by a Poisson distribution, given as

$$P_{FA}(m_k) = \frac{(\beta_{FA}V)^{m_k}\exp(\beta_{FA}V)}{m_k!}. \tag{2.20}$$

Here, m_k denotes the number of false detections in the current scan and V denotes the volume of the scanning area. β_{FA} denotes the false alarm rate per scan per unit volume.

2.4.4 Track initiation, track maintenance and track deletion

Referring to Figure 3.1 in the next chapter, it can be seen that the tracking scenario has now been set up as discussed in the preceding subsections and the sequential tracking algorithm is now covered. In this study a general N_1/N_1 & M_2/N_2-logic track initiator [17], where $N_1 > 0$ and $0 < M_2 < N_2$ are natural numbers representing the M/N logic parameters is used to perform track initiation, maintenance and deletion. The M/N initiator initialises a tentative track if all the N_1 valid measurements in the first N_1 scans are received and confirms the tentative track when at least M_2 valid measurements in the following N_2 scans are received. A tentative track is deleted otherwise. A valid observation refers to the observation that falls within the gate of a track. The M/N algorithm is outlined in Algorithm 1. It is worth noting here, that although a confirmed track may be dropped by the M/N algorithm, certain features of the dropped track are retained to perform track stitching.

The single point initiation technique [33] is used when initialising a tentative track. The initial state of the track is then given by

$$\hat{\boldsymbol{x}}_0 = [\tilde{x}_0, \tilde{y}_0, 0, 0]^\top, \tag{2.21}$$

while the initial covariance of the track is given by

$$\boldsymbol{P}_0 = \begin{bmatrix} \boldsymbol{R}_0 & 0 & 0 \\ 0 & \frac{\dot{x}_{max}}{\kappa} & 0 \\ 0 & 0 & \frac{\dot{y}_{max}}{\kappa} \end{bmatrix}. \tag{2.22}$$

Here, the hat-notation is used to indicate an estimation. The symbols $\tilde{x}_0$ and $\tilde{y}_0$ are the initial x and y elements of the first observation $\boldsymbol{z}_0$. The scaling parameter, κ is chosen to be close to unity and

$\dot{x}_{max}$ and $\dot{y}_{max}$ are the assumed maximum velocity of the target in the x and y directions respectively, thus representing the *a priori* information. The symbol $\boldsymbol{R}_0$ denotes the initial observation position covariance as determined by equation 2.16.

Algorithm 1: The M/N logic algorithm

Initialisation: The first valid measurement, $\boldsymbol{z}_k$, of a tentative track arrives.

Perform the following steps:

- Label the track as tentative.
- Set the total number of measurements collected for this initiator to $\psi_k = 1$.
- Reset the total number of updates missed since the initiator has been created, $\tilde{\psi}_k = 0$.
- Set the age of the initiator, $n_k = 1$.

Update with $\boldsymbol{z}_{k+i}$:

for $i = 1, \ldots, N_1 + N_2$ **do**

- Increment the age of the initiator n_{k+i}:

$$n_{k+i} = n_{k+i-1} + 1.$$

- Update ψ_{k+i} and $\tilde{\psi}_{k+i}$:

if $\boldsymbol{z}_{k+i} = \emptyset$

$$\tilde{\psi}_{k+i} = \tilde{\psi}_{k+i-1} + 1$$

$$\psi_{k+i} = \psi_{k+i-1}$$

else

$$\psi_{k+i} = \psi_{k+i} + 1$$

$$\tilde{\psi}_{k+i} = \psi_{k+i-1}$$

- Decide on the track state as follows:

if $n_{k+i} < N_1$

if $\tilde{\psi}_{k+i} > 0$

Delete the track.

else

The track is still tentative.

else

if $\tilde{\psi}_{k+i} > N_2 - M_2$

Delete the track.

else if $\psi_{k+i} \geq N_1 + M_2$

Confirm the track.

else

The track is still tentative.

end

2.4.5 The Kalman filter

In this study the Kalman filter [13] is used extensively. A full derivation of the Kalman filter equations can be found in Addendum A.1. As mentioned in previously the Kalman filter realises Bayesian estimation in a linear Gaussian tracking environment and therefore has a prediction and update step, as depicted in Figure 2.3. The Kalman filter can be represented by a hidden Markov model (HMM) as shown in Figure 2.5, where the potential functions of the model are all Gaussian. The Kalman filter is used to estimate the hidden states (white nodes in Figure 2.5) in a dynamic linear model, by using the observations (gray nodes in Figure 2.5). The term $\hat{\boldsymbol{x}}_t$ denotes the estimated value of $\boldsymbol{x}_t$ at time step t.

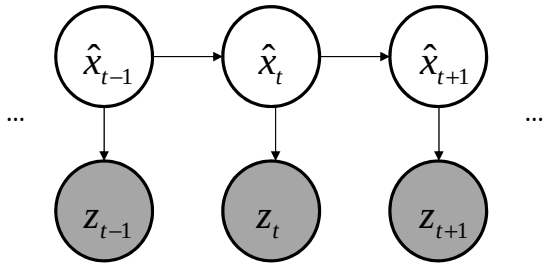

Figure 2.5: The Kalman filter is represented by a hidden Markov model (HMM). The observed nodes are shown in gray, while the hidden states are shown in white.

The Kalman filter takes as input two linear Gaussian models. The first model is the dynamic model and corresponds to a first order Markov model, as given by

$$\boldsymbol{x}_t = \boldsymbol{\Phi}\boldsymbol{x}_{t-1} + \boldsymbol{u}_t. \tag{2.23}$$

An overview of the Kalman filter algorithm is depicted in Figure 2.6.

Where $\boldsymbol{\Phi}$ is the transition matrix, $\boldsymbol{x}_n$ is the state at the current time step n, $\boldsymbol{x}_{n-1}$ is the state at the previous time step, and $\boldsymbol{u}_n$ represents the process noise, sampled from a zero mean Gaussian distribution with covariance $\boldsymbol{Q}$, $\mathcal{N}(0, \boldsymbol{Q})$.

The second model that the Kalman filter takes as input is the observation model. This model is also represented by a linear Gaussian model, given by

$$\boldsymbol{z}_n = \boldsymbol{H}\boldsymbol{x}_n + \boldsymbol{v}_n. \tag{2.24}$$

Where $\boldsymbol{z}_n$ represents the observation at time n, x_n represents the state, $\boldsymbol{H}$ is a transition matrix and

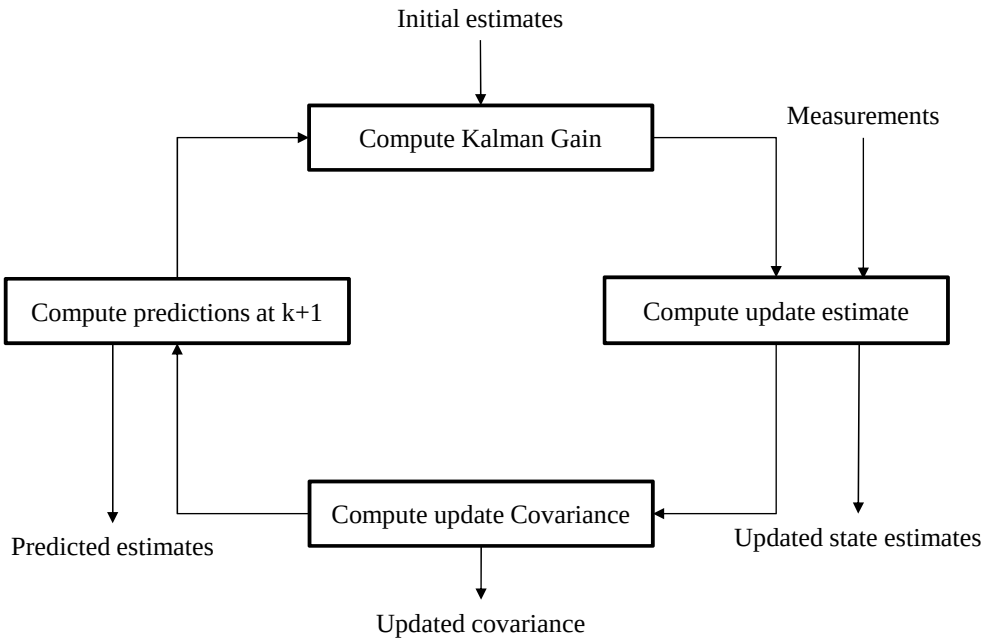

Figure 2.6: An overview of the Kalman filter algorithm.

$\boldsymbol{v}_t$ is the observation noise sampled, from a zero mean Gaussian distribution with covariance $\boldsymbol{R}$, $\mathcal{N}(0,\boldsymbol{R})$.

The algorithms related to the Kalman filter are given below, and has been organised into prediction and update groups as to conform to the Bayesian model.

1. Prediction equations:

   (a) State extrapolation (*a priori*):

$$\hat{\boldsymbol{x}}_{n|n-1} = \boldsymbol{\Phi}_n \hat{\boldsymbol{x}}_{n-1|n-1} \tag{2.25}$$

   (b) Covariance extrapolation (*a priori*):

$$\boldsymbol{P}_{n|n-1} = \boldsymbol{\Phi}_n \boldsymbol{P}_{n-1|n-1} \boldsymbol{\Phi}_n^\top + \boldsymbol{Q}_n \tag{2.26}$$

2. Update equations:

   (a) Kalman gain computation:

$$\boldsymbol{K}_n = \boldsymbol{P}_{n|n-1} \boldsymbol{H}_n^\top \left[\boldsymbol{H}_n \boldsymbol{P}_{n|n-1} \boldsymbol{H}_n^\top + \boldsymbol{R}_n \right]^{-1} \tag{2.27}$$

(b) State update (*a posteriori*):

$$\hat{\boldsymbol{x}}_{n|n} = \hat{\boldsymbol{x}}_{n|n-1} + \boldsymbol{K}_n \left[\boldsymbol{z}_n - \boldsymbol{H}_n \boldsymbol{x}_{n|n-1} \right] \tag{2.28}$$

(c) Covariance update (*a posteriori*):

$$\boldsymbol{P}_{n|n} = (\boldsymbol{I} - \boldsymbol{K}_n \boldsymbol{H}_n) \boldsymbol{P}_{n|n-1} \tag{2.29}$$

Where $\boldsymbol{\Phi}$ is the linear state extrapolation matrix, $\boldsymbol{Q}_n$ is the process noise which is Gaussian zero mean, $\boldsymbol{H}$ is a transition matrix, $\boldsymbol{R}_n$ is the observation noise which is Gaussian zero mean and $\boldsymbol{I}$ is an identity matrix. The resulting updated state is a weighted average based on the certainty of the prediction, as indicated by the filter covariance.

It is worth noting here, that if there is no valid observation to perform an update with, no update is performed and the filter coasts. The result is that the filter predicts the next state, while the filter covariance, $\boldsymbol{P}_{n|n-1}$ grows because of the addition of the noise term $\boldsymbol{Q}$.

2.4.6 The data association problem

In this study data association plays an important role in assigning observations to track as well as assigning track fragments to tracks. Data association almost always suffers from the curse of dimensionality because of the combinatorial nature of the problem [17]. This subsections deals with data association in the observation to track association case. Observation gating is discussed in the next subsection, which aims to reduce the computational complexity inherent in the data association problem.

2.4.6.1 Observation to track data association

In this study two approaches to perform observation to track data association are considered, namely the nearest neighbour (NN) and global nearest neighbour (GNN) approaches [17].

However, nearest neighbour observation to track data association is the simplest form of data association. For each of the measurements that satisfy the gating condition, the best observation is determined as that observation that minimises the Mahalanobis distance determined as

$$\boldsymbol{z}(k)_{best} = \min \left[\tilde{\boldsymbol{y}}(k) \boldsymbol{S}_k^{-1} \tilde{\boldsymbol{y}}(k)^\top \right]. \tag{2.30}$$

Nearest neighbour data association however is only effective in the single target tracking environment where track coalescence is negligible.

Global nearest neighbour (GNN) is useful in a multiple target tracking environment where observation to track association conflicts might arise. These association conflicts can be resolved by use of an assignment matrix, where the elements in the assignment matrix are given by an association cost defined by a generalised statistical distance given by

$$c_{ij} = d_{ij}^2 + \ln(|\boldsymbol{S}_{ij}|), \tag{2.31}$$

where c_{ij}^2 is the the Mahalanobis distance between observation i and the prediction of track j as defined earlier. The addition of the log-covariance term leads to further penalisation of tracks with a larger prediction uncertainty [17].

The association cost matrix is then defined by

$$\boldsymbol{C}_{GNN} = \begin{bmatrix} c_{11} & c_{12} & \dots & c_{1m} \\ c_{21} & c_{22} & \dots & c_{2m} \\ \vdots & \vdots & \vdots & \vdots \\ c_{n1} & c_{n2} & \dots & c_{nm} \end{bmatrix}, \tag{2.32}$$

where m denotes the number of measurements in the scan and n represents the number of tracks (confirmed and tentative).

Numerous techniques exist to solve the data association problem described by the association matrix in equation 2.32 [34]. Several techniques exists for finding the most probable association hypotheses by solving the the association matrices. These include the Hungarian method [35], Jonker-Volgenant [36] relaxation, Munkres's algorithm [37] and 0-1 integer programming [34] to name but a few. In this study the auction algorithm [38] is used, as it is deemed to be the most effective by the authors of [17].

The aim of the auction algorithm is to maximise the gain, where all buyers (tracks) are "almost happy" with their items (observations) as in a real auction. The notion of "almost happy" can be defined mathematically as

$$\max_i (a_{ij} - pr_{ij}) - (a_{i_j j} - pr_{ij}) \leq \varepsilon, \tag{2.33}$$

where $a_{i_j j}$ is the reward from assigning observation j to track i_j, pr_{ij} is the price of assigning observation j to track i_j and ε is the maximum amount of deviation allowed from the optimal solution. The auction algorithm is summarised in Algorithm 2. Here the assignment matrix $\boldsymbol{A}_i$ consists of the rewards $a_{i_j j}$.

Algorithm 2: The Auction algorithm

Data: Given the assignment matrix $\boldsymbol{A}_i$.

Result: List of the best assignments of observations to tracks

Initialise all observations as unassigned. Initialise the track prices as $pr_i = 0$

for *observation* $j \in$ *unassigned observations* **do**

Find "best" track, i_j for the current unassigned observation, j such that

$$a_{i_j j} - P_{ij} = \max_{i=1\ldots n} (a_{ij} - P_i).$$

If any observation is assigned to track i_j, deassign this observation and assign observation j to track i_j.

Set the price of track i_j to the level at which observation j is "almost happy", given by

$$pr_{ij} = pr_{ij} + y_j + \varepsilon,$$

where y_j is the difference between the best and second best assignment values for observation j.

end

2.4.7 Observation gating

In this study gating techniques are used to eliminate unlikely observation to track associations, in order to reduce to the dimensionality of the association problem [17]. Only measurements falling within the gate are considered and processed by the data association logic. Observation gating is performed around the extrapolated measurement, $\hat{\boldsymbol{y}}(k)$ of the track, given by

$$\hat{\boldsymbol{y}}(k) = h(\hat{\boldsymbol{x}}(k|k-1)), \tag{2.34}$$

where, $\hat{\boldsymbol{x}}(k|k-1)$ is the state estimation at the previous time step and $h(\cdot)$ is the function used to determine the predicted measurement at time k.

When determining the size of the gate, the residual vector is of importance, as it is used to provide measure of certainty about the predicted measurement through a Mahalanobis distance. The residual

vector is defined as

$$\tilde{\boldsymbol{y}}(k) = \boldsymbol{z}(k) - \hat{\boldsymbol{y}}(k), \tag{2.35}$$

where $\boldsymbol{z}(k)$ is the noisy received observation of the true state at time k, defined by,

$$\boldsymbol{z}(k) = \boldsymbol{H}_k \boldsymbol{x}(k) + \boldsymbol{v}(k), \tag{2.36}$$

where, $\boldsymbol{x}(k)$ is the true target state, $\boldsymbol{v}(k)$ is zero mean Gaussian white noise and $\boldsymbol{H}_k$ is a transition matrix used to map the true state space into the observed state space. The residual covariance is also of importance when defining the size of the gate and is given by

$$\boldsymbol{S}_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top + \boldsymbol{R}_k, \tag{2.37}$$

where $\boldsymbol{R} = \mathbb{E}[\boldsymbol{v}_k \boldsymbol{v}_k^\top]$ and $\boldsymbol{P}_{k|k-1}$ is the predicted estimate covariance matrix. The predicted estimate covariance matrix is given by

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{F}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{F}_k^\top + \boldsymbol{Q}_k, \tag{2.38}$$

where $\boldsymbol{F}_k$ is the state transition model used to predict $\boldsymbol{P}_{k|k-1}$ from the previous covariance matrix $\boldsymbol{P}_{k-1|k-1}$, and $\boldsymbol{Q}_k$ is the covariance of a zero mean multivariate normal distribution defined by the process noise.

Two types of gates are typically considered in the literature [17], namely rectangular and ellipsoidal gates. Rectangular gating is the simplest form of gating. The gating region is defined as a rectangle dependent on the absolute of the value of each of the elements $\tilde{y}_i \in \tilde{\boldsymbol{y}}(k)$ given by

$$|\tilde{y}_i| \leq K_{GR} \sigma_r. \tag{2.39}$$

Where K_{GR} is referred to as the rectangular gating constant. In general $K_{GR} \geq 3.0$ to account for uncertainty in the dynamics of the target [17] and σ_r is the innovation standard deviation.

In this study ellipsoidal track gates are used to gate observations. An ellipsoidal gating a gate G is defined such that the Mahalanobis distance d^2 satisfies

$$d^2 = \tilde{\boldsymbol{y}}(k) \boldsymbol{S}_k^{-1} \tilde{\boldsymbol{y}}(k)^\top \leq G, \tag{2.40}$$

where $\boldsymbol{S}_k^{-1}$ is the inverse of the residual covariance. In this study no signal or attribute data is assumed, in this case the equation

$$\ln\left(\frac{P_D}{(2\pi)^{M/2}\beta_{FA}\sqrt{|\boldsymbol{S}_k|}}\right) - \frac{d^2}{2} \geq \ln(1-P_D) \tag{2.41}$$

holds [17]. Where M is the order of the observation vector, $|\boldsymbol{S}_k|$ is the determinant of the residual covariance, β_{FA} is the false alarm rate per scan per unit volume and P_D is the probability of detection as defined in subsection 2.3.1.2. By substituting Equation 2.41 into Equation 2.40, the gating condition for the case when no signal or attribute data is assumed is given by

$$d^2 \leq G_0 = 2\ln\left(\frac{P_D}{(2\pi)^{M/2}\beta_{FA}\sqrt{|\boldsymbol{S}_k|}(1-P_D)}\right), \tag{2.42}$$

where $|\boldsymbol{S}|$ is the determinant of the filter residual covariance matrix, and M is the dimension of the observation vector. P_D and β_{FA} are the probability of detection and the false alarm rate respectively, as mentioned earlier. In equation 2.42 no signal or attribute data is assumed.

Any observations that lie outside of the gating region of the track are not considered as a viable update to the track. These observations are used to initialise new tentative tracks using the Kalman filter.

Only the observations that satisfy the gating condition are considered to update the track. However, only a single observation is used to update the track, while any other observations are used to initialise new tentative tracks. The procedure to decide which valid observation to assign to a given track is discussed in the data association subsection.

2.4.8 Track maintenance

Track maintenance includes the creation updating and deletion of target tracks. Several algorithms exists to perform this including the track score algorithm [39] and the M/N algorithm [17]. In this study the M/N algorithm is used to perform track maintenance, because of the simplicity and effectiveness of the algorithm.

The M/N track maintenance algorithm is a simple counting based algorithm, with low computation. In the M/N algorithm all new measurements are tentative tracks. The track is only confirmed, when a predefined M out of N consecutive measurements satisfy the gating condition of the track. All

tentative tracks that do not satisfy this requirement are deleted. Deletion of the confirmed tracks occurs when a predefined number of consecutive misses (no targets in the track gate) occur.

This algorithm is independent of the quality of the track as long as the gating criteria for the track is satisfied.

2.4.9 Belief Propagation

Belief propagation algorithms, also known as message passing algorithms are powerful tools used to perform inference on statistical graphical models. The sum-product or max-product algorithms can be used to find the the most likely assignments in a large number of graphical models [25]. In this study the Viterbi algorithm is of specific interest and is a related algorithm to the general sum-product algorithm [5].

2.4.9.1 The Viterbi algorithm

The Viterbi [5] algorithm is a related algorithm of the sum-product belief propagation (message passing) algorithm and is often known as the min-sum, or max-product algorithm. The Viterbi algorithm solves the problem of maximising the probability of a sequence [21]. The goal is to find the sequence in the set of states, $\boldsymbol{x}$, that maximises a global function, g, i.e. the sequence of most probable states. The most probable sequence in $\boldsymbol{x}$ can be defined by the term

$$\arg\max_{\boldsymbol{x}} g(\boldsymbol{x}). \tag{2.43}$$

The Viterbi algorithm can be summarised as follows. For a discrete state Hidden Markov Model (HMM) with state space S, initial probabilities π_i of being in state i and transition probabilities $a_{i,j}$ of transitioning from state i to state j, outputs $\boldsymbol{y}_1,\ldots,\boldsymbol{y}_N$ are observed. The most likely state sequence, $x_1,\ldots,x_N$, that produces the observations is given by the recurrence relations in the equations [21]

$$p_{1,k} = p(\boldsymbol{y}_1|k)\pi_k, \tag{2.44}$$

$$p_{n,k} = p(\boldsymbol{y}_n|k)\max_{\boldsymbol{x}\in S}\left(a_{\boldsymbol{x},k}p_{t-1,\boldsymbol{x}}\right), \tag{2.45}$$

where $p_{n,k}$ is the probability of the most probable state sequence responsible for the first n observations, with the final state k. The Viterbi path can be retrieved by storing pointers of which state x was

used in Equation 2.45. Let $\mathrm{Pt}(k,n)$ be the pointer function that returns the value of x used to compute $p_{n,k}$. Then,

$$\boldsymbol{x}_N = \arg\max_{\boldsymbol{x}\in S}(p_{N,\boldsymbol{x}}),$$

$$\boldsymbol{x}_{n-1} = \mathrm{Pt}(\boldsymbol{x}_n, n).$$

The algorithm can now backtrack to find the most probable state sequence (Viterbi path) using the pointer function, Pt. The probabilities can be implemented as costs (inverse probabilities), in this case the Viterbi algorithm aims to minimise the Viterbi cost. The complexity of the Viterbi algorithm is a function of both the state space of the HMM as well the number of observed outputs, and is given by $O(N \times |S|^2)$.

2.4.10 Multiple hypothesis tracking

In this study the hypothesis based MHT algorithm [40] is extended to perform track fragment to track associations [7] in order to evaluate the performance of the developed algorithm. The standard MHT algorithm [6] keeps multiple hypotheses about the past and current observation to track association uncertainties at each time step. Unlike the tracking algorithm described earlier in this study, the MHT algorithm does not use a separate track initiation procedure, because the track initiation procedure is integrated into the tracking algorithm. Between any two time steps the MHT algorithm keeps a record of the hypotheses as well as the current tracks. An overview of the MHT algorithm is provided in Figure 2.7.

The MHT algorithm starts at the initialisation block of Figure 2.8. The first set of empty hypotheses are given by $\{\Theta_0^i\}_{i=1}^{N_h}$, where i indicates the hypothesis index and N_h indicates the number of hypotheses. After a new set of measurements is received a new set of association hypotheses is created using the new measurements. The set of association probabilities $\{P(\Theta_k^i)\}_{i=1}^{N_h}$ for each association hypothesis is then calculated. Hypothesis reduction techniques such as clustering and hypothesis pruning are then performed and the most probable hypothesis Θ_k^i is determined. This hypothesis is then presented to the user.

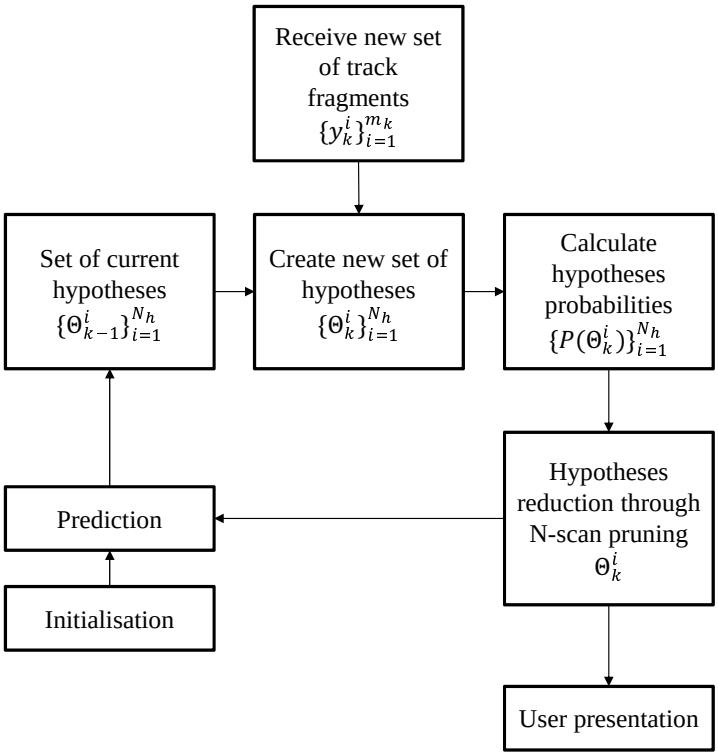

Figure 2.7: An overview of the hypothesis based MHT tracking algorithm.

2.4.10.1 Hypothesis generation

Hypotheses are generated after prediction and gating. Each hypothesis $\{\Theta_{k-1}^i\}_{i=1}^{N_h}$ kept from the previous time steps are characterised by the number of tracks (assumed number of targets) and the corresponding statistics of the hypothesis. To explain the standard MHT algorithm an example scenario is used. Figure 2.8 [17] shows two example hypotheses up to discrete time k.

In Figure 2.8 on the left, hypothesis Θ_{k-1}^1 assumes two tracks, with their gates shown, while the figure on the right shows hypothesis Θ_{k-1}^2 with only one track, and its gate shown.

Observations for the current scan are now received. This is depicted in Figure 2.9. As can be seen in Figure 2.9 observations $\boldsymbol{z}_k^1$, $\boldsymbol{z}_k^2$ and $\boldsymbol{z}_k^3$ are now received at the indicated positions. Each hypothesis, Θ_{k-1}^1 and Θ_{k-1}^2 is now extended to determine all possible association hypotheses for the current time step, while not violating the gating conditions. The valid hypotheses can be illustrated by expanding a tree graph for each original hypothesis Θ_{k-1}^1 and Θ_{k-1}^2. This is shown in Figure 2.10. The new set of hypotheses for each original hypothesis is dependent on the original hypothesis Θ_{k-1}^i, as well as the association hypothesis θ_k about the current measurement $\boldsymbol{z}_k$ as per the gating conditions. This is

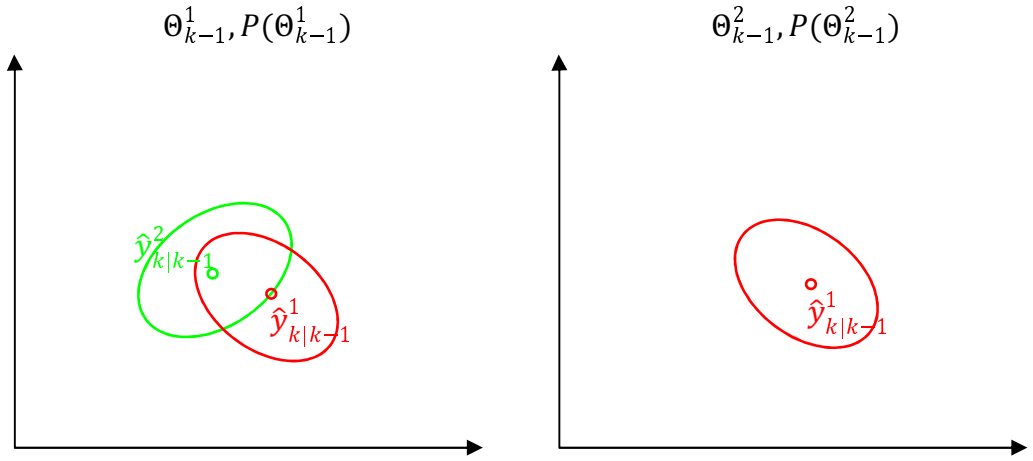

Figure 2.8: The figure on the left shows a hypothesis (Θ^1_{k-1}) with two tracks, while the figure on the right shows another hypothesis (Θ^2_{k-1}) with only one track. Each hypothesis is further characterised by a probability of the hypothesis being true.

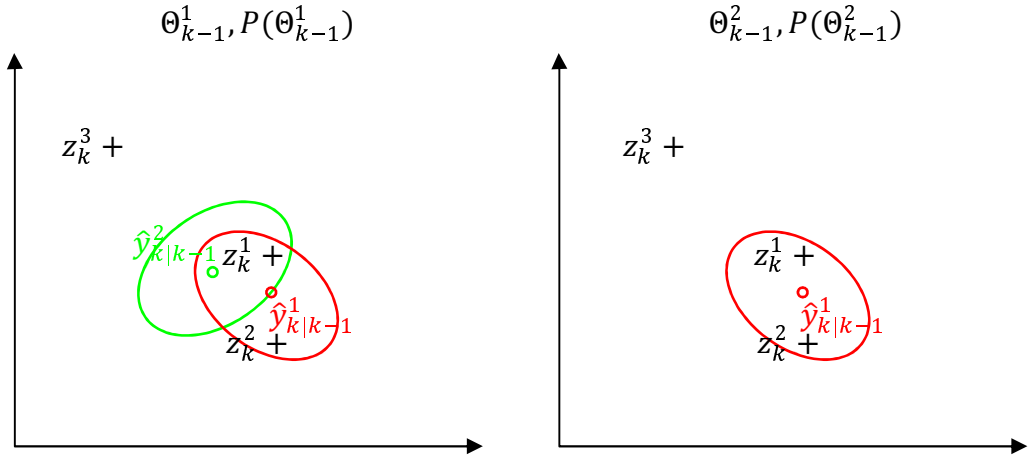

Figure 2.9: Observations z^1_k, z^2_k and z^3_k are now received at the positions indicated by a '+'.

given by

$$\Theta^l_k = \{\theta_k, \Theta^i_{k-1}\}. \tag{2.46}$$

Each new hypothesis is shown as a horizontal limb of the tree diagram from Figure 2.10. The probability of each hypothesis is determined next.

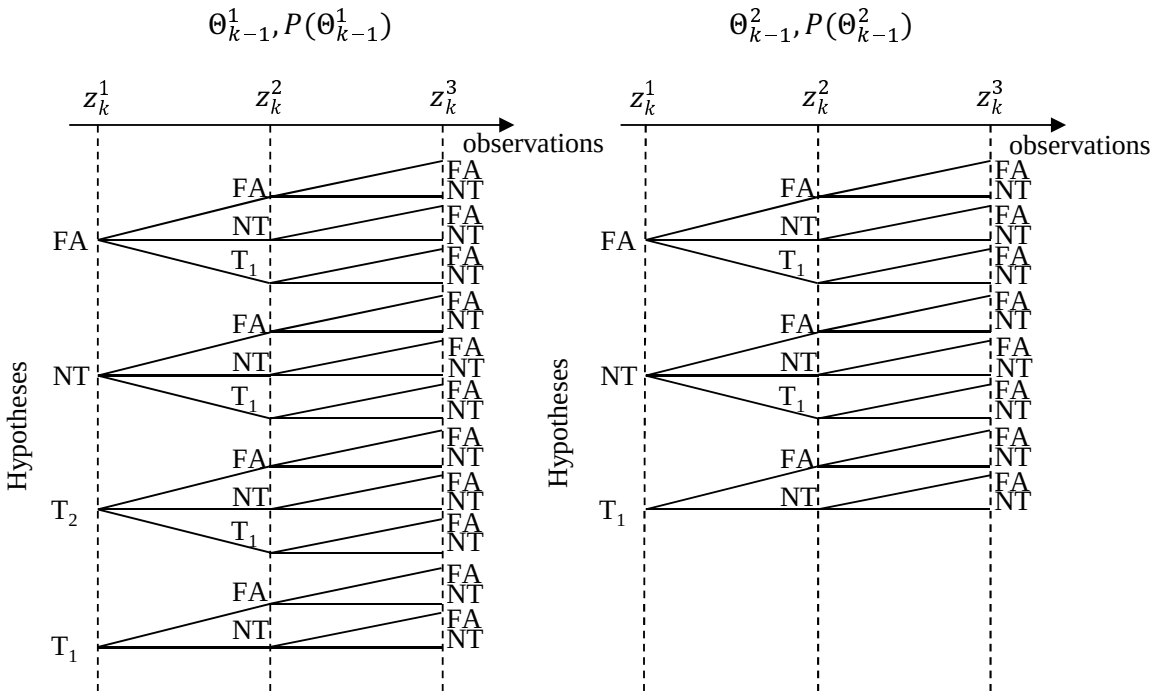

Figure 2.10: The association hypotheses in the original hypotheses Θ^1_{k-1} and Θ^2_{k-1} can be expanded using a tree graph. The graphs flow from left to right. Each horizontal limb represents one association hypothesis in each of the original hypotheses. T_1 and T_2 corresponds to the tracks in each of the hypotheses, while NT denotes new track and FA denotes a false alarm. Each expansion in the tree graph is a new hypothesis for the current time step. Hence there are a total of 38 hypotheses originating from the two hypotheses of the previous time step.

2.4.10.2 Hypothesis probability

Each hypothesis, represented by a horizontal limb in the tree diagram in Figure 2.10 has the *posterior* hypothesis probability

$$P(\Theta^l_k|\boldsymbol{z}_{0:k}) \propto \beta_{FA}^{m_k^{FA}} \beta_{NT}^{m_k^{NT}} \left[\prod_{j \in \mathcal{T}_D} \frac{P_D^j p^j_{k|k-1}(\boldsymbol{z}_k^{\theta_k^{-1}(j)})}{(1 - P_D^j P_G^j)} \right] C_i P(\Theta^i_{k-1}|\boldsymbol{z}_{0:k-1}), \tag{2.47}$$

where β_{FA} denotes the false alarm rate, β_{NT} denotes the new track rate, P_D^j and P_G^j denotes the probability of detection and deletion of the jth target respectively [40], P_G^j denotes the gate probability of the jth target and $p^j_{k|k-1}(\boldsymbol{z})$ denotes the predicted measurement density of the jth target. The set $\mathcal{T}_D$ denotes the indices of targets which were assigned a measurement by Θ^i_k. The term C_i is a constant and is defined below, while the term $P(\Theta^i_{k-1}|\boldsymbol{z}_{0:k-1})$ denotes the probability of the of the most pro-

bable hypothesis in the last time step, given all the observations up to the last time step. The equation for the *posterior* hypothesis probability given in Equation 2.47 is derived in Addendum A.2.

The constant C_j is given by

$$C_j = \left[\prod_{j\in\mathcal{T}_{ND}} 1 - P_D^j P_G^j\right]\left[\prod_{j\in\mathcal{T}_D} P_D^j P_G^j\right] = \prod_{j=1}^{n_T}(1 - P_D^j P_G^j),$$

where $\mathcal{T}_{ND}$ denotes the set of indices of targets which were not assigned a measurement by Θ_k^i. The aim is now to find the set of hypotheses $\{\Theta_k^l\}_{N_h}$ that maximises $P(\Theta_k^l|\mathbf{z}_{0:k})$.

2.4.11 Techniques to reduce the number of hypotheses

A number of hypothesis reduction techniques exist for MHT. These include

1. hypothesis clustering,
2. hypothesis merging and
3. hypothesis pruning techniques.

In this study a simple, but effective N-scan pruning algorithm is used to reduce the number of hypotheses in the track stitching algorithm. The N-scan pruning algorithm assumes that any uncertainty at time $(k-N)$ will be resolved by time k, for all time k.

2.4.11.1 Implementation of MHT

In implementation only the best N_h maximum number of hypotheses are generated [40]. Murty's algorithm is then used to find the N_h-best solutions with the minimum number of unnecessary hypothesis generations [41]. By returning to the example scene in Figure 2.9, the observations may be assigned to the existing tracks, the observations may be as a result of new targets, or the observations may be caused by false detections. Two assignment matrices can therefore be constructed for each of the hypotheses. This is shown Figure 2.11.

In Figure 2.11

$$l_{ij} = \left[\frac{P_D^j p_{k|k-1}^j(\mathbf{z}_k^i)}{(1 - P_D^j P_G^j)}\right],$$

$\Theta^1_{k-1}, P(\Theta^1_{k-1})$

A_1	T_1	T_2	FA_1	FA_2	FA_3	NT_1	NT_2	NT_3
z^1_k	l_{11}	l_{12}	$\log\beta_{FA}$	$\times$	$\times$	$\log\beta_{NT}$	$\times$	$\times$
z^2_k	l_{21}	$\times$	$\times$	$\log\beta_{FA}$	$\times$	$\times$	$\log\beta_{NT}$	$\times$
z^3_k	$\times$	$\times$	$\times$	$\times$	$\log\beta_{FA}$	$\times$	$\times$	$\log\beta_{NT}$

$\Theta^2_{k-1}, P(\Theta^2_{k-1})$

A_2	T_1	FA_1	FA_2	FA_3	NT_1	NT_2	NT_3
z^1_k	l_{11}	$\log\beta_{FA}$	$\times$	$\times$	$\log\beta_{NT}$	$\times$	$\times$
z^2_k	l_{21}	$\times$	$\log\beta_{FA}$	$\times$	$\times$	$\log\beta_{NT}$	$\times$
z^3_k	$\times$	$\times$	$\times$	$\log\beta_{FA}$	$\times$	$\times$	$\log\beta_{NT}$

Figure 2.11: The possible assignment matrices A_1 and A_2 corresponding to the hypotheses Θ^1_{k-1} and Θ^2_{k-1} for the scene in Figure 2.9

denotes the likelihood to be maximised by Murty's algorithm. β_{FA} and β_{NT} denote the false alarm and new track rates respectively. The symbol $\times$ represents impossible associations due to gating constraints, and is set to the log likelihood of minus infinity.

The quantity $p^j_{k|k-1}(\boldsymbol{z}^i_k)$ is the predicted Gaussian measurement density of the jth target. An example of such a density is shown in Figure 2.12.

The assignment problem can now be solved by using Murty's algorithm. The algorithm is summarised in Algorithm 3.

As can be seen from Algorithm 3, Murty's algorithm uses the auction algorithm to determine the best solution to the assignment problem. The auction algorithm was discussed earlier in subsection 2.3.1.4., with the algorithm summarised in Algorithm 2.

After application of Murty's algorithm the

$$\beta_{FA}^{m_k^{FA}} \beta_{NT}^{m_k^{NT}} \left[\prod_{j \in \mathcal{T}_D} \frac{P_D^j p^j_{k|k-1}(\boldsymbol{z}_k^{\theta_k^{-1}(j)})}{(1 - P_D^j P_G^j)} \right]$$

part of Equation 2.47 is maximised. The latter part of Equation 2.47 is only dependent on the previous

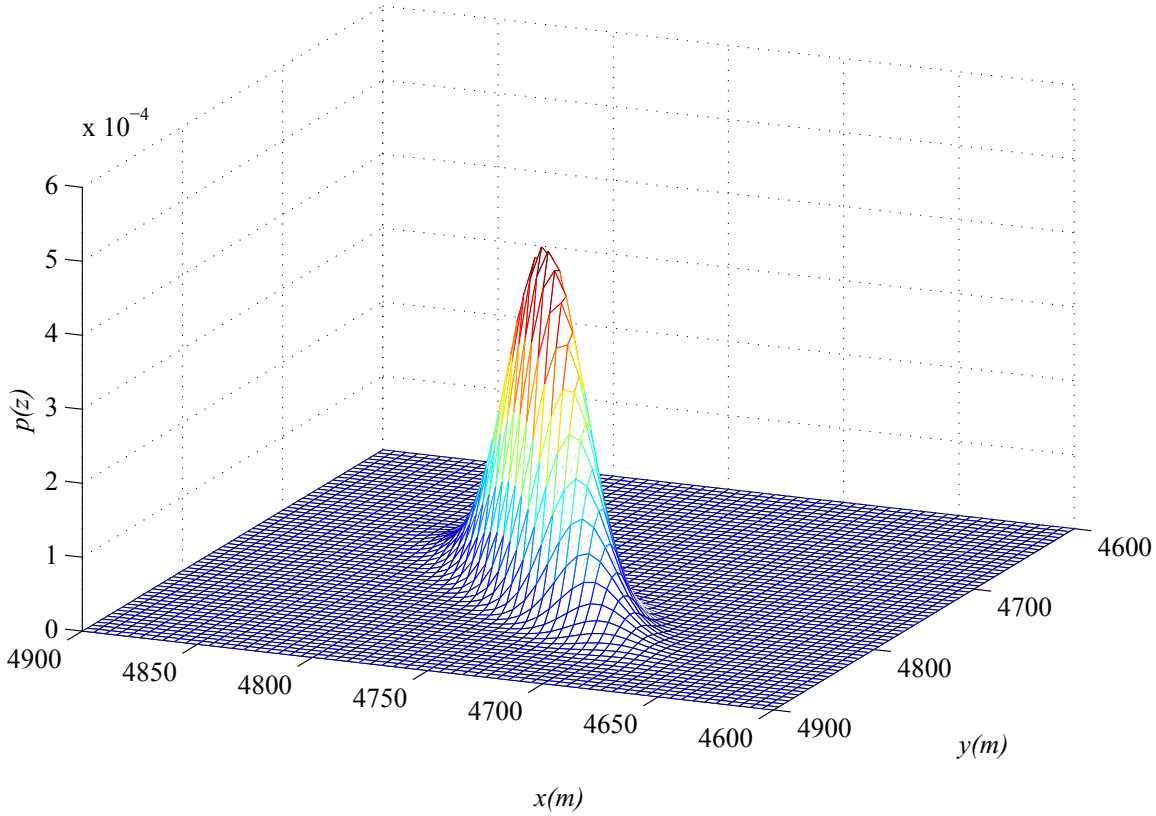

Figure 2.12: The predicted Gaussian measurement density of jth target.

Algorithm 3: Murty's algorithm

Data: Given the assignment matrix A_i and the number of required solutions, N_h.

Result: Ordered list of the best solutions to the assignment problem

Find the best solution to the assignment problem using the auction algorithm.

Find the next next $N_h - 1$ solutions as follows:

for $i \in N_h - 1$ *required solutions* **do**

Express the ith best solution as the solution of a number of best solution assignment problems.

Find the solution to each of these using the auction algorithm.

The solution that results in the maximum reward is the next best solution.

end

hypothesis and the constant term C_i.

The best (highest probability) solution of the N_h solutions produced by Murty's algorithm for each of the hypotheses is now selected and these hypotheses are updated. The best solution in each of the

hypotheses now form the new set of hypotheses Θ_k^i.

For the hypotheses where a new track is the most probable, a new single measurement track is initiated for these observations, and the updated tracks are added to the track list.

Observations that are associated to columns of the assignment matrix corresponding to false alarm measurements are discarded.

The tracks that did not have a valid update in a fixed number of consecutive scans N_P are pruned.

In the following chapter the methods used to solve the track stitching problem are discussed. An extension of the MHT algorithm to perform track-to-track associations is also discussed.

CHAPTER 3

METHODS

"All progress is precarious, and the solution of one problem brings us face to face with another problem."

Martin Luther King, Jr. (1929-1968)

3.1 CHAPTER OVERVIEW

In this chapter the methods used to perform multiple target tracking as well as track stitching are discussed. The chapter commences with a discussion on how target tracking is performed by use of a track maintenance algorithm and a Kalman filter. It is also discussed how track stitching is performed by using selectively stored information from the track fragments, representing nodes in a graphical model. A discussion on how the graphical model is solved using belief propagation (message passing) techniques is then presented. This section concludes with a discussion on the extension of a multiple hypothesis tracking (MHT) algorithm, to perform track to track associations instead of observation to track associations.
It is also worth noting here that the following assumptions are made in this chapter.

1. The sensor measurements arrive in the correct chronological order, i.e. not out of sequence.

2. A target can generate exactly one measurement at any given time, i.e. the sensor measurements are resolved.

3. It is also assumed that the target will not manoeuvre while occluded, and will continue in an almost linear fashion. A gating procedure is used to allow for small deviations from this.

3.2 TARGET TRACKING

In order to perform track stitching from track fragments, a multiple target tracking (MTT) environment must first be implemented to perform target tracking in the presence of false detections. The resulting tracks or fragments of tracks will then be used to perform track stitching. Figure 3.1 shows an overview of the target tracking algorithm, including target and false detections generation.

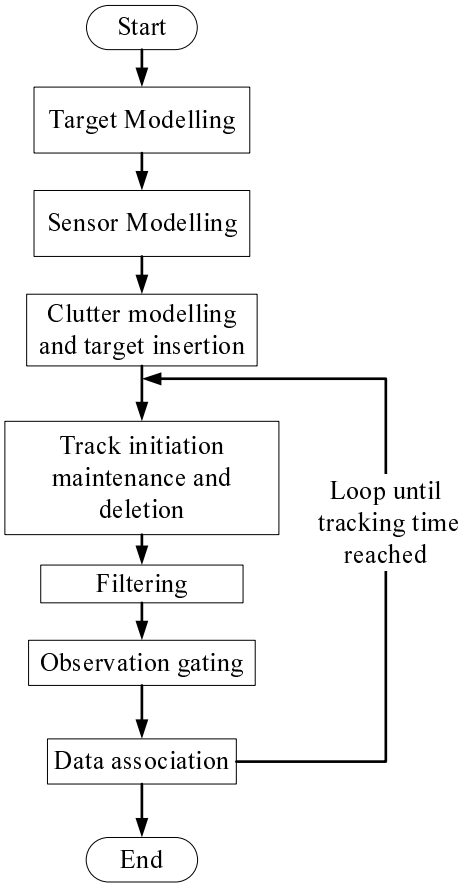

Figure 3.1: An overview of the short term tracking algorithm, including target and false detections generation. Selective track features are retained to perform track stitching.

As can be seen from Figure 3.1, the tracking environment is first set up, by generating true targets, and modelling these targets as radar observations. False detections is then generated and the noisy target observations are inserted into the generated false detections. The false detections may also create false target tracks. The sequential target tracking algorithm then commences. First, tracks are initialised based on the observations through a track management algorithm, thereafter, the observations are filtered and the observations from the following scan are gated. Lastly data association is performed on the observations within the gating area of each of the tracks. These concepts were discussed in the

previous chapter.

The track stitching aspects in Figure 3.1 is the focus of this section. It is worth noting here, that selective information about the tracks are stored, in order to perform track stitching in real time. The developed track stitching algorithm is discussed in the next section.

3.3 TRACK STITCHING

In this section the track stitching algorithm proposed by this research is discussed. The underlying path stitching problem is an NP-hard problem in general, and both the sequential Viterbi track stitching algorithm and the Multiple Hypothesis track stitching algorithm use pruning algorithms to address the combinatorial computational complexity. The proposed track stitching algorithm has novel aspects with the joint use of a graphical model, belief propagation and a Viterbi data association algorithm.

In order to perform track stitching, selective information about the track fragments to be stitched together are retained. To conserve memory not all the information about a particular track fragment is stored.

The retained parameters of interests are:

1. the track start state,
2. the filter covariance at the start of the track,
3. the track end state,
4. the filter covariance at the end of the track,
5. the creation time of the track,
6. the termination time of the track.

Because of the single point initiation technique used in the tracking algorithm, as discussed in subsection 3.2.4, the very first filter covariance cannot be taken as the start covariance of the track as this will be much larger than the consequent covariances after an update has been received.

3.3.1 Track-to-track time independence

Given the assumption listed at the start of this chapter it can be inferred that a confirmed track can only be the result of exactly one target at any given time, t. It can also be inferred that each track can be associated with exactly one target. These constraints then leads to the conclusion that two tracks, existing at overlapping time intervals, must have been generated by two different targets. Hence, two tracks that exists at overlapping time intervals cannot be associated with each other. In subsequent subsections this property is exploited to reduce the number of track-to-track association hypotheses.

3.3.2 Track stitching model

In this subsection the model that was used to perform track fragment to track fragment stitching is discussed. The model is discussed by assuming a two target example, where the two targets are denoted as M_1 and M_2.

Both targets are partially occluded, according to the model mentioned earlier, and produced track fragments $T_1, T_2 ... T_N$ as output from the tracking algorithm. These track fragments are represented by a vector containing the saved track information. It is assumed that the track fragment indices coincide with the chronological order of the track fragments for each target, although the two targets may have overlapped in time. An example scenario with 5 track fragments is shown in Figure 3.2. This scenario will be used to discuss the track stitching algorithm further. It is important to note that although the targets are later denoted as M_1 and M_2, nothing is assumed about the number of targets *a priori*.

To determine which track fragments could possibly be stitched together a gating algorithm is utilised. When a track fragment is terminated, the predicted current position of each existing track fragment is determined using a Kalman propagator (i.e. the Kalman filter coasts up to the current point in time for each track fragment).

The coasting operation of the Kalman filter causes the filter covariance to increase, this in turn increases the size of the ellipsoidal gate. Each track fragment is then evaluated in turn. If the estimated starting position of a later track fragment falls within the gate around the predicted position of an earlier track fragment, the two track fragments are deemed to may have been produced by the same

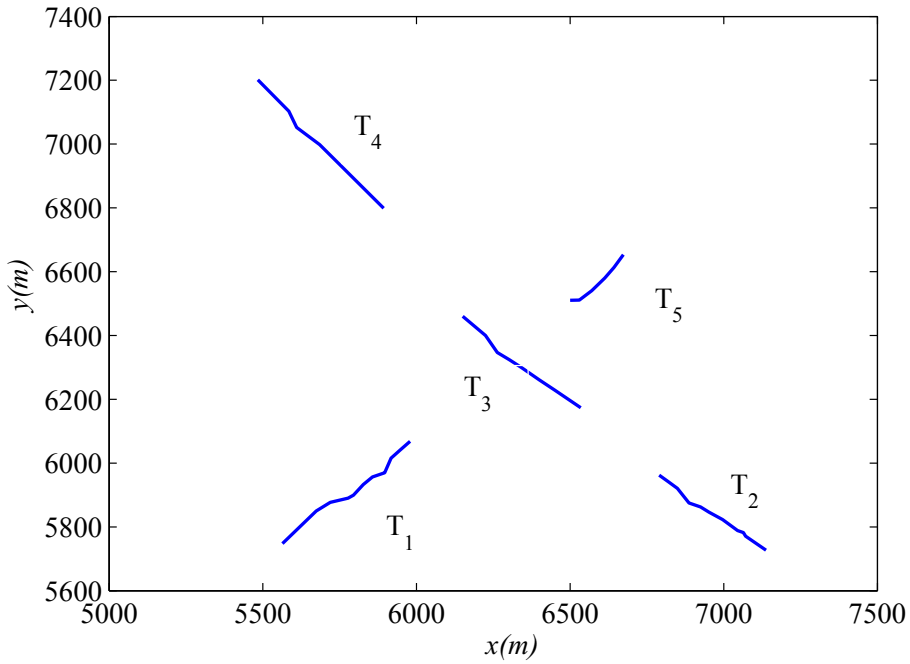

Figure 3.2: An example scenario of 5 track fragments, $T_1 \ldots T5$.

target, and could therefore be stitched together, depending on the association certainty.

The track fragment gating process is illustrated in Figure 3.3, where track fragments T_a and T_b are arbitrary track fragments used to illustrate the process.

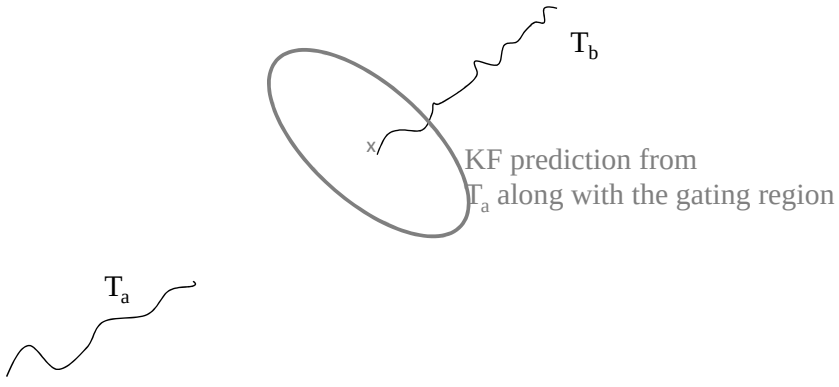

Figure 3.3: The gating region and position prediction from earlier track, T_a is shown in grey. The starting position of the later track, T_b, falls within this gating region, and is therefore considered a possible association.

An ellipsoidal gate [17] G_a is used to perform the track fragment gating process and is given by

$$G_a = 2\ln\left(\frac{P_D}{(1-P_D)(2\pi)^{M_a/2}\beta_{FA}\sqrt{|\boldsymbol{S}_a|}}\right), \tag{3.1}$$

where P_D denotes the probability of detection, M_a denotes the dimension of the estimated state vector of the earlier track and $|\boldsymbol{S}_a|$ denotes determinant of the predicted innovation covariance of the earlier track fragment as determined from the covariance of the Kalman propagator. The predicted innovation covariance is given by

$$\boldsymbol{S}_k = \boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top + \boldsymbol{R}_k. \tag{3.2}$$

The value $\boldsymbol{H}_k$ is the transition matrix, $\boldsymbol{P}_{k|k-1}$ is the predicted covariance matrix and $\boldsymbol{R}$ is the observation noise at time k as predicted from track fragment T_a.

The track fragments in Figure 3.2 and possible associations, as determined by the gating process can now be modelled as a Markov model as shown in Figure 3.4. For the purposes of this example, it is assumed that the possible association pairs from the gating process are T_1 and T_3, T_1 and T_5, T_2 and T_3, T_2 and T_4 and lastly T_3 and T_4.

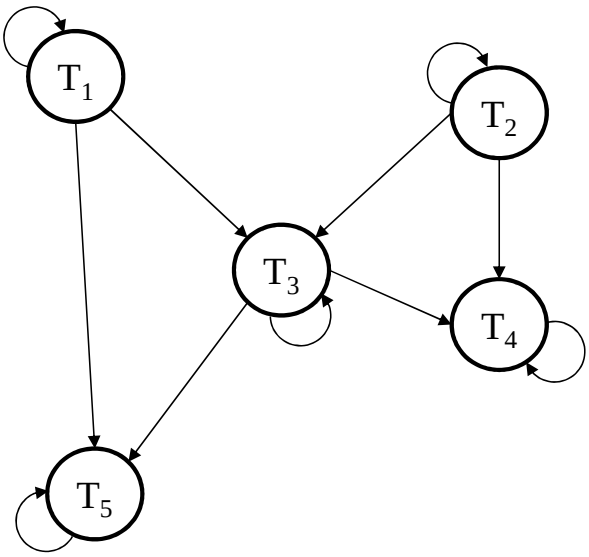

Figure 3.4: A number of track fragments T_1 to T_5, represented by a state transition diagram, where the state transitions show how paths generated by an initially unknown number of targets enter track fragments (states) in sequences $[T_1, T_3, T_5]$, $[T_1, T_3, T_4]$, $[T_2, T_3, T_4]$, $[T_2, T_3, T_5]$, $[T_1, T_5]$, $[T_1]$, $[T_2]$, $[T_3]$, $[T_4]$ and $[T_5]$

In Figure 3.4 the nodes are referred to as states, where each state represents a track fragment. The

number of targets is initially unknown, but can at most be equal to the number of track fragments. A track fragment can either be stitched to a previous track fragment, from the same target or a track fragment can be the result of a new target (false or real). If unassociated, each of the track fragments can be a target on its own. A transition to a different node represents that the target that generated the former track fragment is in the latter track fragment. Transitions to the same node are reserved for the case where the underlying target generating the track fragment does not transition to another track fragment, i.e. no association is made to another track fragment. It is important to note here, that because of the assumption that observations and therefore the tracks are in the correct chronological order, these diagrams do not allow traversing from a later state to an earlier one. The cardinality of the state space also changes with time, growing when a new track fragment becomes available and shrinking when associated track fragments are fused or when unnassociated track fragments are purged.

The state diagram in Figure 3.4 can be restructured into a lattice or trellis like diagram. In this study this configuration is referred to as a track trellis. For the sake of clarity, the invalid associations were removed. In implementation however, the invalid associations are set to have a likelihood of zero (infinite cost). The resulting track trellis can be seen in Figure 3.5. The columns of the graph represent the time when the track fragments were inserted, at irregular intervals. A column is only inserted in the graph when a new track fragment becomes available. This reduces the computational complexity, in that the graph only has to be solved once a new track fragment becomes available or when a track fragment is updated, instead of at every time step during tracking.

The association probability of the the track fragment to itself for the time before the track fragment is inserted is set equal to one, $p(T_n|T_n) = 1$, i.e. no other associations are possible, up until the column where the fragment was inserted into the graph. The grey nodes and grey horisontal arrows are used only for illustration purposes to indicate when a track fragment becomes available.

3.3.3 Track-to-track association probability

The track-to-track association probabilities are defined in this subsection. These probabilities describe the viability of associating one track fragment to another. Equations 3.3 to 3.5 describe the costs and

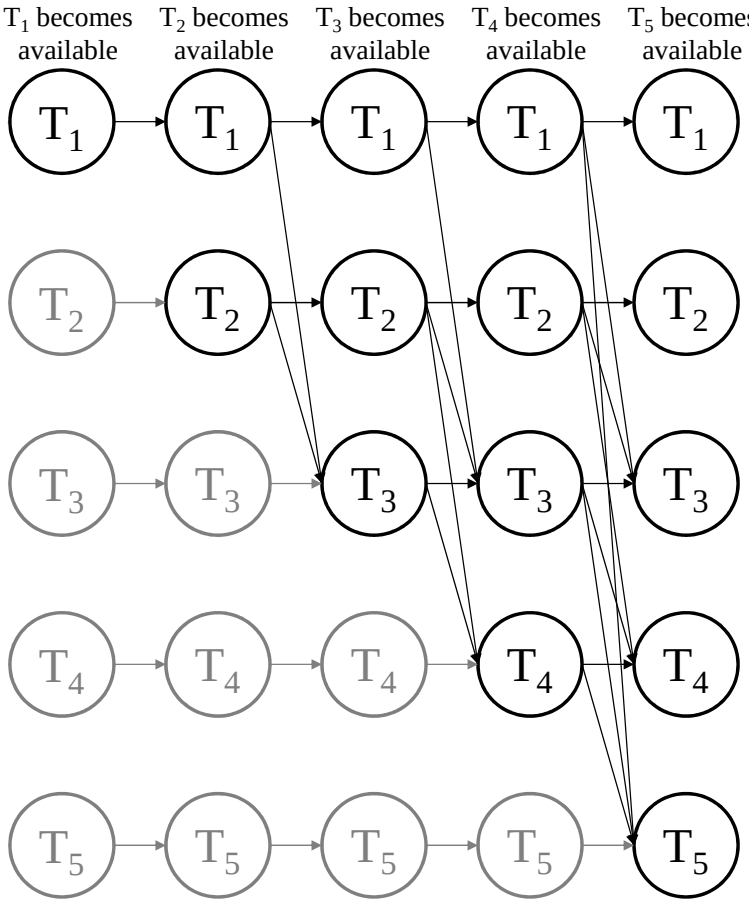

Figure 3.5: The track graph trellis for the state diagram in Figure 3.4. The grey nodes represent states that are inserted to preserve the square structure of the graph.

likelihoods of associating an earlier track, T_i to a later track T_j. These equations are given by

$$\boldsymbol{P}_t^{ij} = \boldsymbol{P}_t^i + \boldsymbol{P}_t^j, \tag{3.3}$$

$$\hat{\boldsymbol{\Delta}}_t^{ij} = \hat{\boldsymbol{x}}_t^i - \hat{\boldsymbol{x}}_t^j, \tag{3.4}$$

$$\delta_t^{ij} = \hat{\boldsymbol{\Delta}}_t^{ij\top} [\boldsymbol{P}_t^{ij}]^{-1} \hat{\boldsymbol{\Delta}}_t^{ij}, \tag{3.5}$$

$$p(T_i|T_j) \propto \frac{1}{\delta_t^{ij}}. \tag{3.6}$$

In equations 3.3 to 3.5, $\boldsymbol{P}_t^{ij}$ is the combined covariance of the predicted state of the earlier track fragment, predicated to the start time of the later track segment and the initial covariance of the later track fragment. The term $\hat{\boldsymbol{\Delta}}_t^{ij}$ is the difference between the state estimations of the predicted state of the earlier track fragment, predicated to the start time of the later track fragment, and the initial state estimate of the later track. δ_t^{ij} represents an association cost between the two tracks, and $p(T_i|T_j)$ is

the probability of assigning an earlier track, T_i, to a later track, T_j. $\boldsymbol{P}_t^i$ is the estimated covariance of the earlier track at the initiation time of the later track fragment, $\boldsymbol{P}_t^j$ is the covariance at the start of the later track and $\boldsymbol{P}_t^{ij}$ is the combined covariance at time t. $\hat{\boldsymbol{x}}_t^i$ is the the predicted state of the earlier track fragment to the state of the later track fragment, while $\hat{\boldsymbol{x}}_t^j$ is the estimated starting position of the later track. Association costs are converted to association probabilities, by inverting the costs and ensuring that all the probabilities $p(T_i|T_j)$. entering a node in the graph in Figure 3.5 sum to 1. These equations hold only when it is assumed that the estimation errors, $\hat{\boldsymbol{x}}^i - \boldsymbol{x}^i$ and $\hat{\boldsymbol{x}}^j - \boldsymbol{x}^j$ are independent [3]. This assumption does not generally hold but is a simplification, and only results in a larger combined covariance in equation 3.3, because of the lack of a cross-covariance term, which leads to a larger ellipsoidal gate for all associations.

3.3.4 Solving the track graph

The track trellis can now be solved sequentially by maximising the association probability for the associations at each node by using the Viterbi algorithm. This is akin to finding the most probable paths (lowest cost) through the track graph, where the paths are all mutually disjoint [42], i.e. the paths do not share any common nodes. Assuming that the track graph has columns, L, and rows, M. The path cost is given by

$$D_m = D_{l-1,m^*} + \min(\delta^G_{m,l-1}), \tag{3.7}$$

$$\tag{3.8}$$

while the path probability is related to the path cost by

$$p(m) \propto \frac{1}{D_m}, \tag{3.9}$$

for a path terminating at column l.

In equation 3.7 D_{l-1,m^*} is the path cost up to the the previous column, and was found to terminate in row m^*, and D_m is the new path cost, terminating in row m. $\min(\delta^G_{m,l-1})$ is the minimum transition cost from column $l-1$ to the node in row m, column l. The symbol $p(m)$ is the probability associated with the path cost D_m. Again the path probabilities can be normalised by ensuring the probabilities of all the paths sum to 1.

Messages of the current path probability from a node in the current column are passed to nodes in the next column of the graph through equation 3.7. These path probabilities are updated using equation

3.9. Once the end of the graph is reached the algorithm backtracks through the most probable nodes to obtain the most likely path. This step is performed sequentially until all the nodes have been included in a path exactly once. These paths then represent the most probable track-to-track associations. Paths that contain only horizontal associations (i.e. only one sequence of nodes on the same vertical level of the trellis) represents a single target track and is not stitched to any existing track. The horizontal associations after the track fragment arrives denotes the *no association* or *single track* transition, this value is fixed and is dependent on the expected number of targets per simulation.

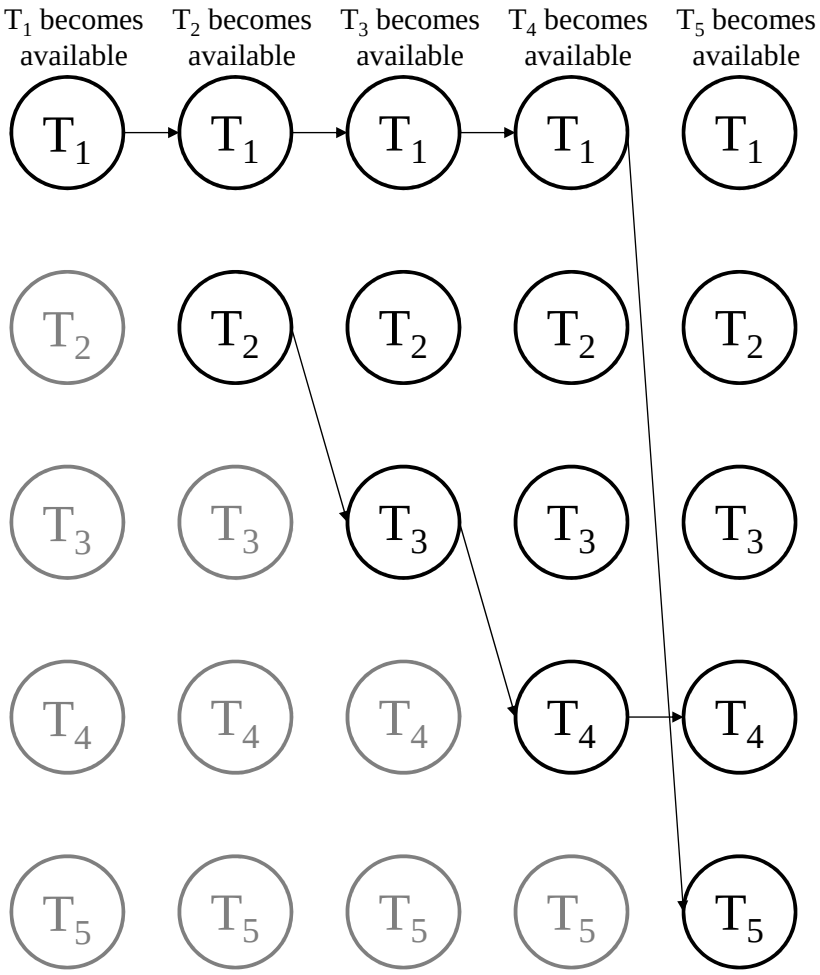

Figure 3.6: The solution of the track fragment graph for the situation shown in Figure 3.2.

The surviving paths are shown in the track fragment graph in Figure 3.6. These two paths correspond to the following sequences of track fragments:

1. $M_1 = \{T_1, T_5\}$,

2. $M_2 = \{T_2, T_3, T_4\}$,

where M_1 and M_2 denote the two target tracks consisting of the track fragments. The results of the track stitching algorithm can be seen in Figure 3.7, where the missing positions have been linearly interpolated.

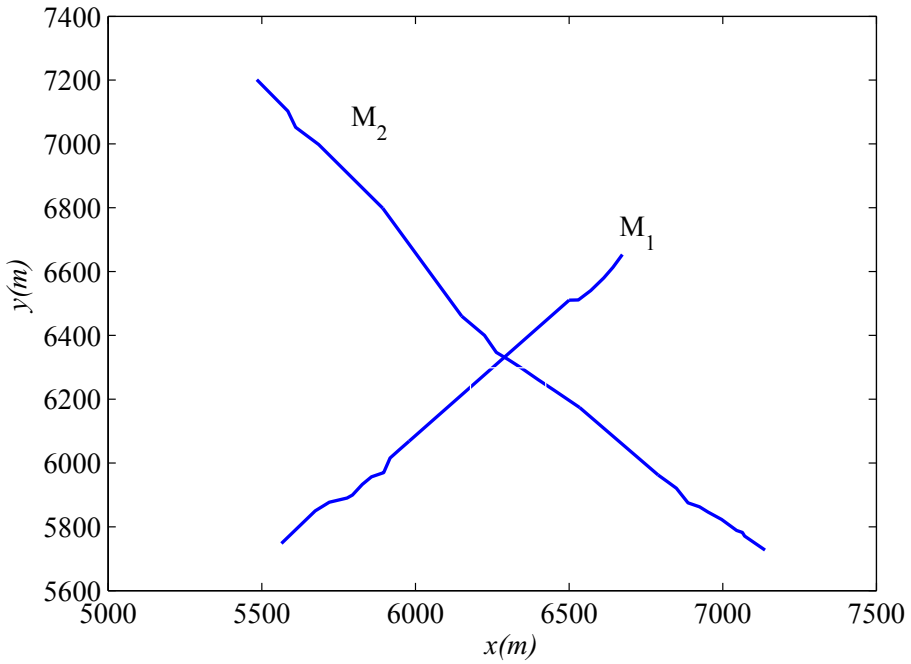

Figure 3.7: The stitched track fragments for the situation shown in Figure 3.2. The missing data is linearly interpolated.

3.3.4.1 Sequential Viterbi data association

In general, the number of targets are not known *a priori* [43]. However, in multiple Viterbi data association (MVDA) implementations [44] the number of required solutions are often required before hand, which in this context translates to knowing how many targets there are by either assuming this or counting the targets [45]. In this study no assumption is made about the number of targets and the sequential Viterbi solution determines the track fragment to track fragment associations by solving the track trellis until all nodes are used exactly once.

3.4 TRACK FUSION AND PRUNING

At some point unused tracks need to be purged and associated tracks need to be permanently fused. This is performed by a track purging and fusion algorithm, using an N-scan approach. Unlike the approach in [4], where an explicit track merging and splitting algorithm is used, the approach followed in this study purges unassociated tracks and handles track splits by instantiating new target tracks.

The algorithm counts the number of consecutive time steps in which a later track fragment has been associated with an earlier track fragment. Once this counter reaches a predetermined number of time steps, the two track fragments are fused together permanently, and a new track fragment is produced according to the following rules.

1. The new track has the starting parameters of the earlier track.
2. The new track has the ending parameters of the later track.
3. The missing positions are linearly interpolated.
4. The two separate tracks are removed from memory, and replaced with the new track.
5. All of the assumptions regarding time constraints, track independence and gating techniques mentioned earlier now applies to the new track.

The deletion of a track fragment occurs in a similar manner. The number of consecutive time steps where a track fragment has had no associations are counted, once this counter reaches a predetermined value, the track fragment is pruned from the track fragment association algorithm. This algorithm is similar to the M/N track managing algorithm.

In the next section Multiple Hypothesis Track Stitching (MHTS) is discussed. It is extended from the standard Multiple Hypothesis Tracking algorithm (observation to track association), discussed in the previous chapter, to a track to track association algorithm.

3.5 MULTIPLE HYPOTHESIS TRACK STITCHING (MHTS)

In this section the Multiple Hypothesis Tracking (MHT) algorithm discussed in the previous section is extended to perform track stitching.

The same track to track gating technique is used as discussed in the previous section to generate a binary validation matrix for each hypothesis Θ_{k-1}^{i},where i denotes the index of the hypothesis, and $k-1$ denotes the previous time step.

From the generated validation matrix for the current hypothesis an assignment matrix is generated as described in the previous chapter. The likelihoods l_{ij} in the validation matrix now denotes the track fragment to track association likelihood and is given by

$$l_{ij} = \left[\frac{P_D^j p_{k|k-1}^j(\hat{\boldsymbol{x}}_{0k}^i)}{(1 - P_D^j P_G^j)} \right],$$

where P_D refers to the probability of true track confirmation from the M/N manager and is slightly lower than the probability of detection, $p_{k|k-1}^j(\hat{\boldsymbol{x}}_{0k}^i)$ is the predicted Gaussian measurement density of the jth target at the first estimated position of track fragment i and P_G^j denotes the gate probability of the jth target. The variable β_{FA} in the assignment matrix is replaced by β_{FT} to denote the false track rate. The mode of the PDF of $p_{k|k-1}^j(\hat{\boldsymbol{x}}_{0k}^i)$ is generally lower than that of $p_{k|k-1}(\boldsymbol{z}_k)$ in the MHT case discussed in the previous chapter, because of the added uncertainty in the track to track association case. This added uncertainty is due to the lower probability of true track confirmation from the M/N manager when compared to the probability of detection of an observation.

The best solution (highest probability) of the N_h solutions produced by Murty's algorithm for each of the hypotheses is now selected and these hypotheses are updated (stitched) using the corresponding track fragments. The best solution in each of the hypotheses now form the new set of hypotheses Θ_k^i.

For the hypotheses where a new track is the most probable, the track fragment is added to the list of tracks and other track fragments can now be stitched to it.

The tracks and track fragments that did not have a valid update in a fixed number of consecutive scans N_P (pruning interval) are pruned and are not considered in subsequent hypotheses.

CHAPTER 4

RESULTS AND DISCUSSIONS

"The true method of knowledge is experiment."

William Blake (1757-1827)

4.1 CHAPTER OVERVIEW

In this chapter the results and findings are presented. The operation of the tracking algorithm is confirmed and evaluated. It is shown that track fragments can be stitched together using the methods and procedures discussed in the previous chapter in real time. The effectiveness and efficiency of the sequential Viterbi algorithm and the multiple hypothesis track stitching algorithm are also compared.

4.2 TARGET TRACKING RESULTS

The results of the target tracking algorithm is presented and discussed first. Before presenting the target tracking simulation results, it is reasonable to summarise the assumptions and simulation parameters made. The simulation parameters for the target tracking algorithm are shown in Table 4.1. The state transition matrix, $\mathbf{\Phi}$, and the the acceleration matrix $\boldsymbol{B}$ is chosen as that of a constant velocity model. The sampling period T,is chosen as 1 second as this is a typical sampling period value. The standard deviation of the acceleration noise vector, σ_a is chosen as 2 m/s^2. The false alarm rate β_{FA} is chosen as 9^{-8} false alarm detections per area per scan. The scanning area is 10 km $\times$ 10 km with the sensor at the Cartesian origin. The probability of detection of an observation, P_D is chosen 0.9 and accounts for missed detections by the sensor. The standard deviation accuracy for the sensor in range σ_{ra} and azimuth σ_{az} is chosen as 10 m and 0.10 radians respectively to account for sensor noise when observations are performed in polar coordinates. The maximum assumed velocity is of the target is

chosen as 40 m/s and is application specific. The values of the M/N track maintenance algorithm is chosen as $N_1 = 2$, $N_2 = 2$ and $M_2 = 3$ and provides a good trade off between initializing new tracks and deleting tracks where no updates have been received.

Table 4.1: Simulation parameters for the target tracking algorithm

Parameter symbol	Description	Parameter value
$\boldsymbol{\Phi}$	State transition matrix.	$\begin{bmatrix} 1 & 0 & T & 0 \\ 0 & 1 & 0 & T \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
$\boldsymbol{B}$	Acceleration matrix for a constant velocity motion model.	$\begin{bmatrix} \frac{T^2}{2} & 0 \\ 0 & \frac{T^2}{2} \\ T & 0 \\ 0 & T \end{bmatrix}$
T	Sampling period.	1 second
σ_a	Standard deviation of the acceleration noise vector.	2 m/s^2
β_{FA}	False alarm rate.	9^{-8} number of FA/area/scan
P_D	Probability of detection.	0.9
σ_{ra}	Standard deviation accuracy of sensor in range.	10 m
σ_{az}	Standard deviation accuracy of sensor in azimuth.	0.10 radians
v_{max}	Maximum target velocity	40 m/s
N_1, N_2, M_2	Parameters of the M/N track maintenance algorithm	$N_1 = 2\ N_2 = 2, M_2 = 3$

4.2.1 True target generation with no fragmentation

Figure 4.1 shows the track of a true target generated as described in the previous chapter. From Figure 4.1 it can be seen that the target follows a fairly linear motion, where the deviation are only caused by the addition of the additive Gaussian white noise in the target motion model. In this case the initial state vector of the target is given by $\boldsymbol{x}_0 = [5000\ 5000\ 25\ 25]^\top$, where $\boldsymbol{x}_0$ takes the form of the state variables defined in equation 2.3

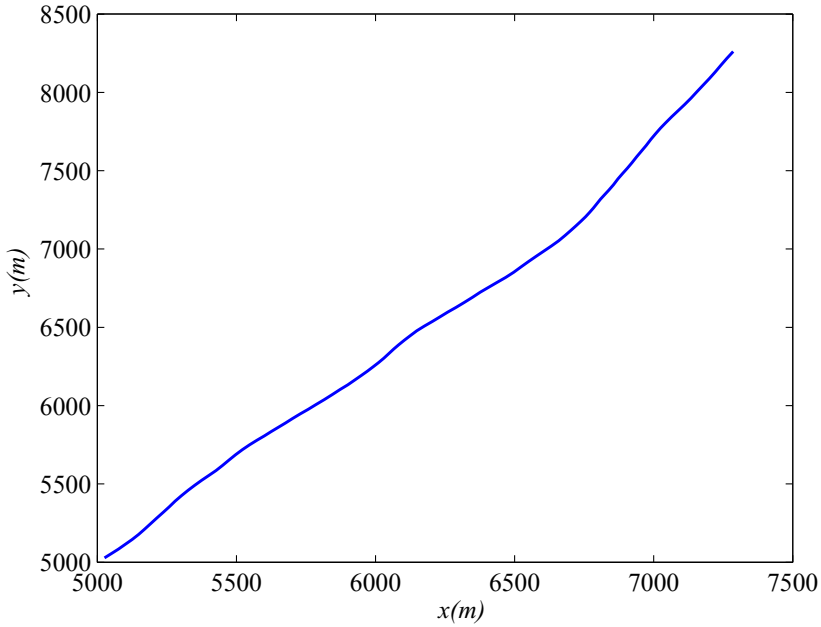

Figure 4.1: The true track of a single target, simulated according to the target motion model as described previously, with no occlusions.

4.2.2 Tracking results with no track fragmentation

The tracking results with no occlusions are presented first to verify the operation of the tracking algorithm. The true target simulated in the previous subsection is tracked though noisy observations in the presence of false detections as described previously. In this simulation the parameters listed in Table 4.1 were used.

Figure 4.2 shows the result of tracking the target through noisy observations in the presence of false detections. The x's indicate the noisy observations used to generate the estimated target track. The

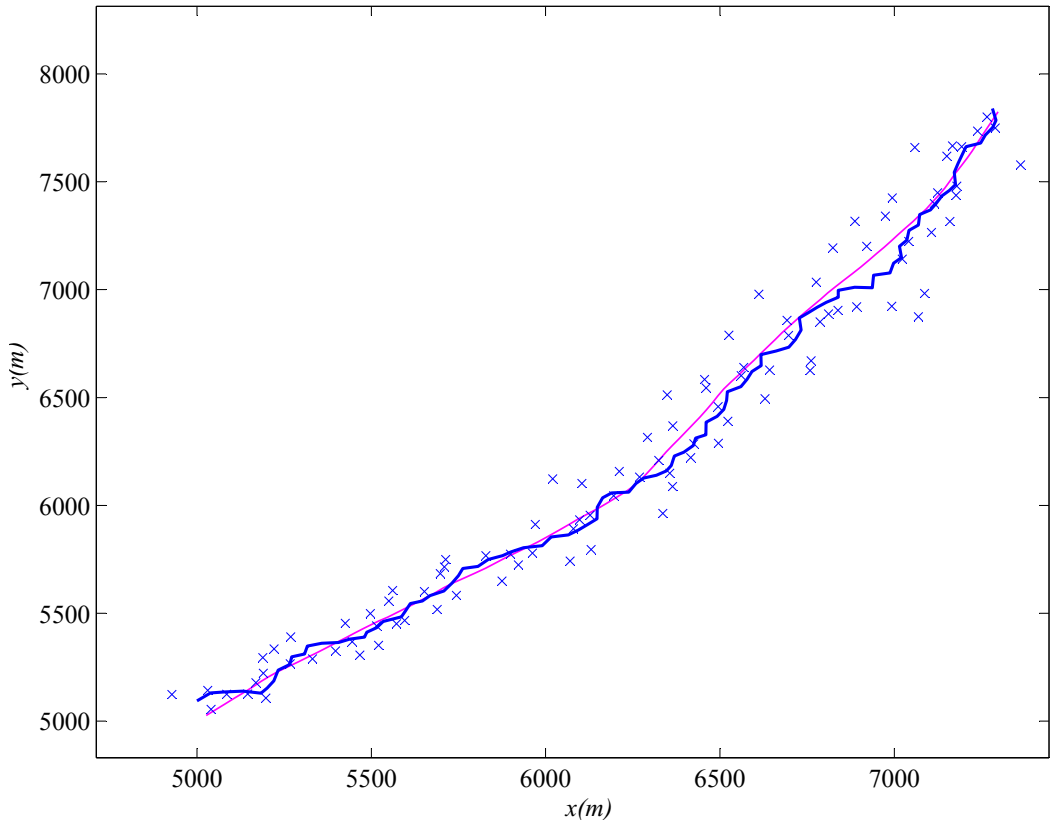

Figure 4.2: The short term tracking algorithm tracks the target successfully when no fragmentation occurs. The estimated target track is shown by the darker line, while the lighter line shows the true target track. The noisy observations are shown as x's.

estimated target track is shown by the darker line, while the lighter line shows the true target track. As can be seen from Figure 4.2, the algorithm successfully tracks the target, without the track being deleted by the M/N track maintenance logic. The gating operation discussed previously can be seen in Figure 4.3. In the figure, the track gate is shown as the ellipse around the prediction from the previous time step, while the updated estimation at the current time step is shown as an '$\times$'. The gate is a summation of the measurement uncertainty ellipse and the observation uncertainty ellipse shown in Figure 4.3. The observation used to update the prediction and determine the estimation is the dark dot within the track gate

A number of Monte Carlo simulations where performed to determine the performance of the tracking algorithm and the track stitching algorithm. The rule of thumb for a simple Monte Carlo simulation is that at least $100/\rho$ number of simulations are needed to reasonably estimate an event of probability ρ [46].

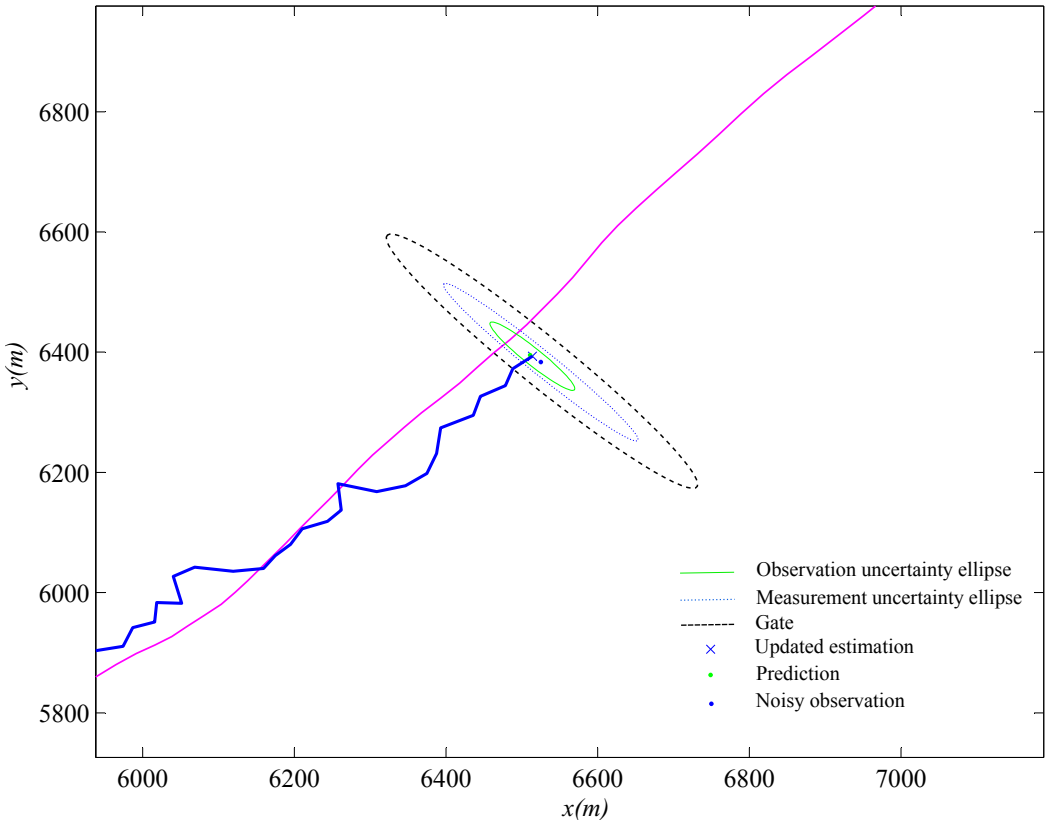

Figure 4.3: The ellipsoidal gates used in the short term tracking algorithm. The gate around the prediction from the previous time step can be seen as the ellipse. The observation used to update the prediction is shown as a dot.

To determine the probability of dropping a target track, 1000 Monte Carlo simulations were performed without the presence of false detections and the number of instances where the M/N track maintenance algorithm deleted the track because of missed updates where counted.

Table 4.2 summarises the results of these Monte Carlo simulations. The probability of detection, P_D and the M/N parameters were selected as shown in Table 4.1.

Table 4.2: Tracking algorithm performance - track maintenance

Number of Monte Carlo Simulations	Number of deleted tracks during simulation	Track deletion probability
1000	32	0.032

To determine the average number of false track initiations caused by false detections, 1000 Monte Carlo simulations were performed. This is summarised in Table 4.3.

Table 4.3: False track initiations in due to false detections.

Number of Monte Carlo Simulations	Number of false tracks	Average number of false tracks per 100 time step simulation
1000	2413	2.4

The performance of the tracking algorithm and in particular the tracking filter was further evaluated by performing Monte Carlo simulations and determining the root mean squared error (RMSE). The RMSE is given by

$$RMSE = \sqrt{\frac{1}{N}\sum_{i=1}^{N}(\boldsymbol{\chi} - \hat{\boldsymbol{\chi}})^2}, \tag{4.1}$$

where N denotes the number of simulations, $\boldsymbol{\chi}$ denotes the true value of the parameter of interest and $\hat{\boldsymbol{\chi}}$ denotes the estimated parameter of interest.

The Monte Carlo simulations were performed over $N = 250$ runs for 100 time step simulations. In this case $\hat{\boldsymbol{\chi}}$ refers to the *posterior* estimate and $\boldsymbol{\chi}$ refers to the true value of the target track at each time step. The RMSE for the x-position and the x-velocity is shown in Figures 4.4 and 4.5 respectively for the parameters above.

Similar simulations were performed for the y-positions and y-velocity RMSE. These results are shown in Figure 4.6 and Figure 4.7 respectively.

As can be seen from Figures 4.4 to 4.7 the tracking algorithm successfully locks on to the target and tracks it.

4.2.2.1 Filter consistency

The filter consistency gives an indication of how good the produced estimations are without considering the true values, which is generally unavailable. To determine the consistency of a Kalman filter it is sufficient to examine the innovations and innovation standard deviations. Figure 4.8 and 4.9 shows

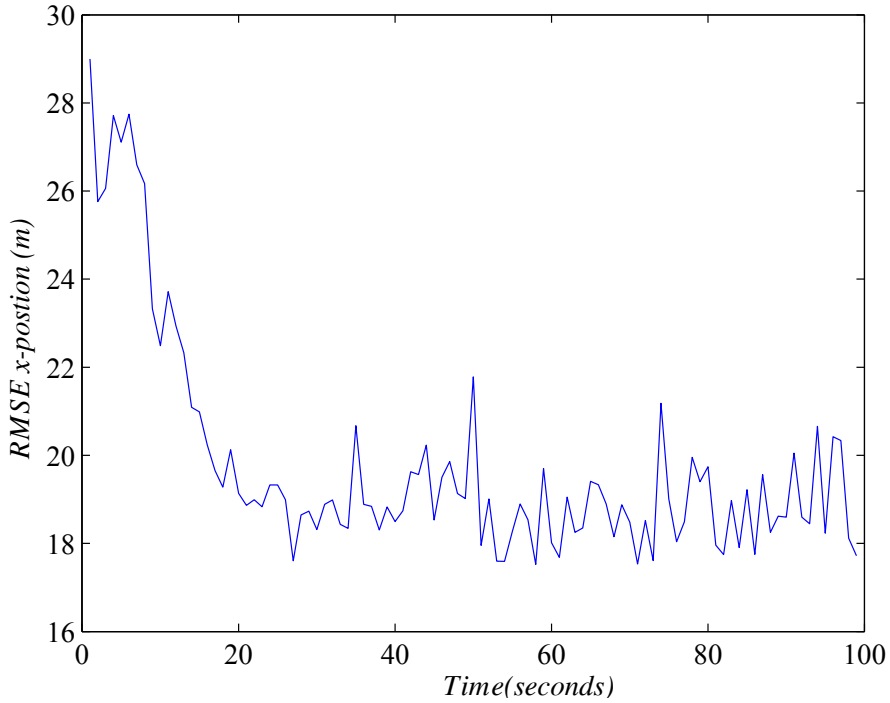

Figure 4.4: The x-position RMSE for the tracking algorithm over 250 Monte Carlo runs for 100 time step simulations

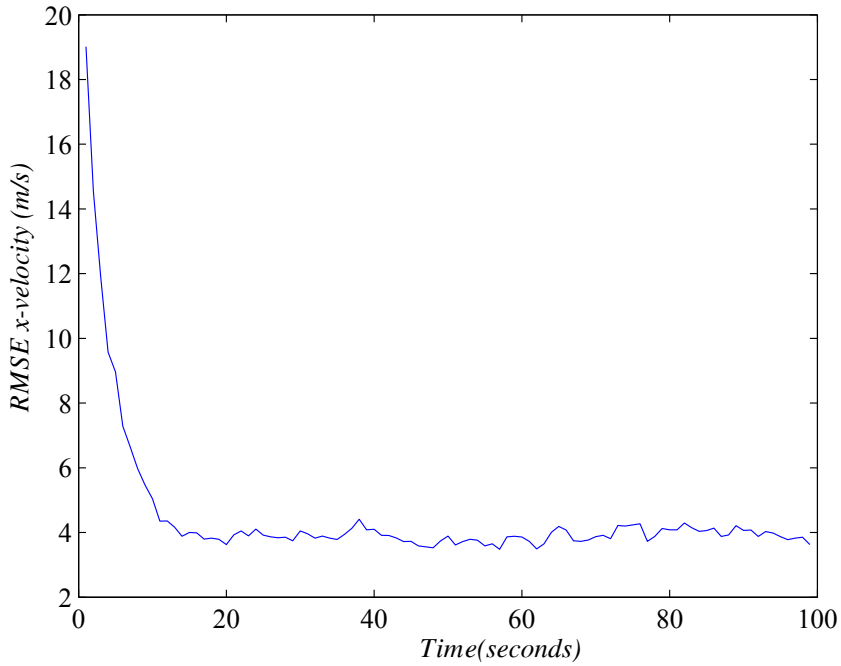

Figure 4.5: The x-velocity RMSE for the tracking algorithm over 250 Monte Carlo runs for 100 time step simulations

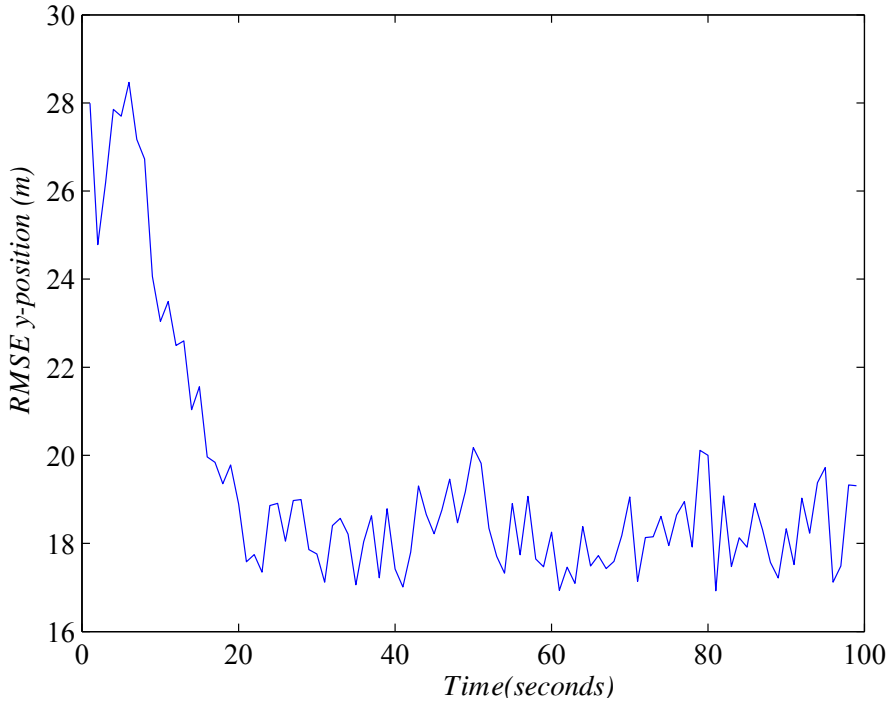

Figure 4.6: The *y*-position RMSE for the tracking algorithm over 250 Monte Carlo runs for 100 time steps.

the innovation sequences in the *x* and *y* directions respectively, together with the corresponding $\pm 2\sigma$ confidence bounds for a single simulation run.

As can be seen from Figure 4.8 and 4.9 the innovation sequences has zero-mean and is white with approximately 95% of the innovations falling within the $\pm 2\sigma$ confidence bounds for any simulation run. This confirms that the Kalman filter implementation is indeed consistent when tracking in the *x* and *y* directions.

4.3 SEQUENTIAL VITERBI TRACK STITCHING RESULTS

In this section the track stitching results, using the Sequential Viterbi track stitching algorithm discussed in the previous chapter, are presented. Again, it is reasonable to summarise the simulation parameters used for the simulation. These parameters are shown in Table 4.4. The simulation parameters from the previous section remain the same except where stated explicitly.

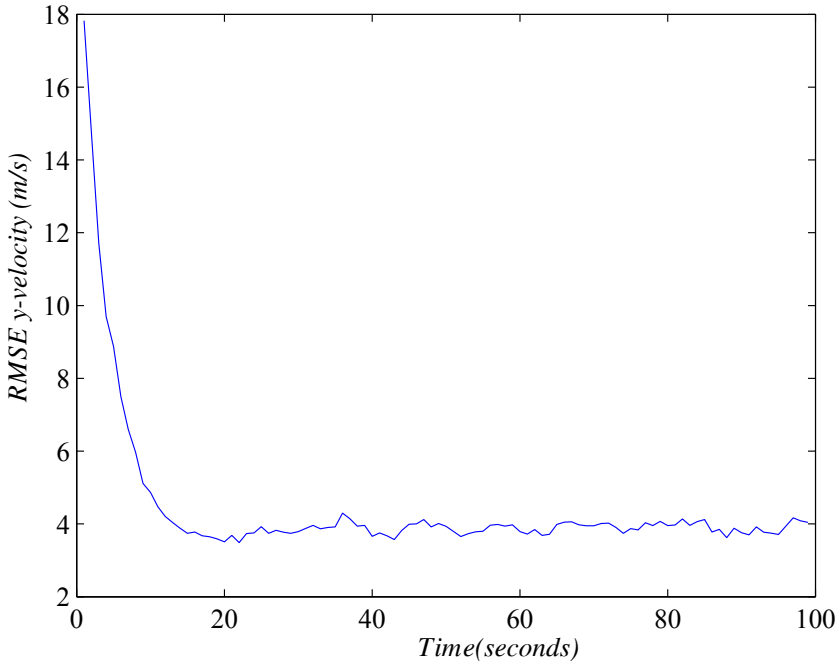

Figure 4.7: The y-velocity RMSE for the tracking algorithm over 250 Monte Carlo runs for 100 time steps.

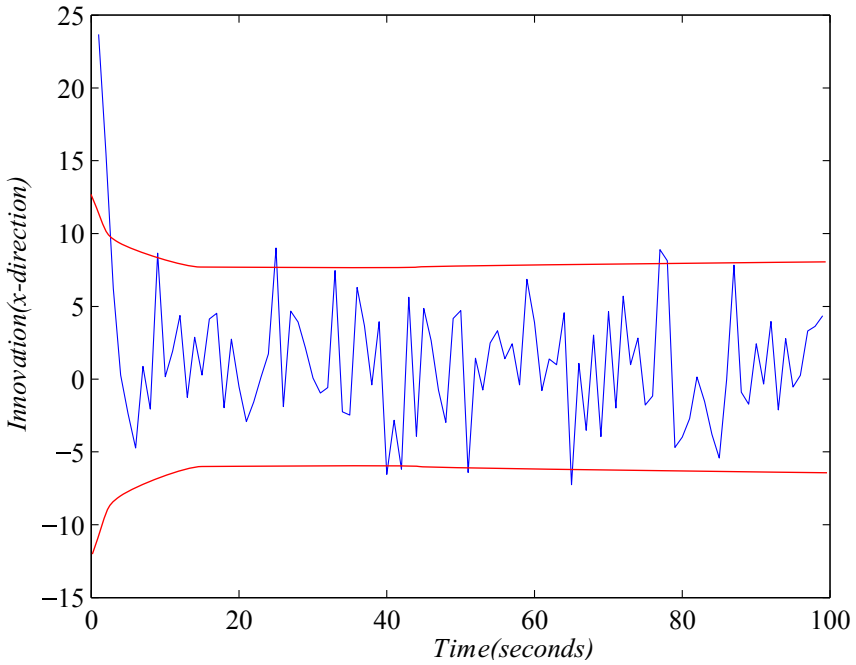

Figure 4.8: The innovation in the x-direction with the $\pm 2\sigma$ confidence bounds shown.

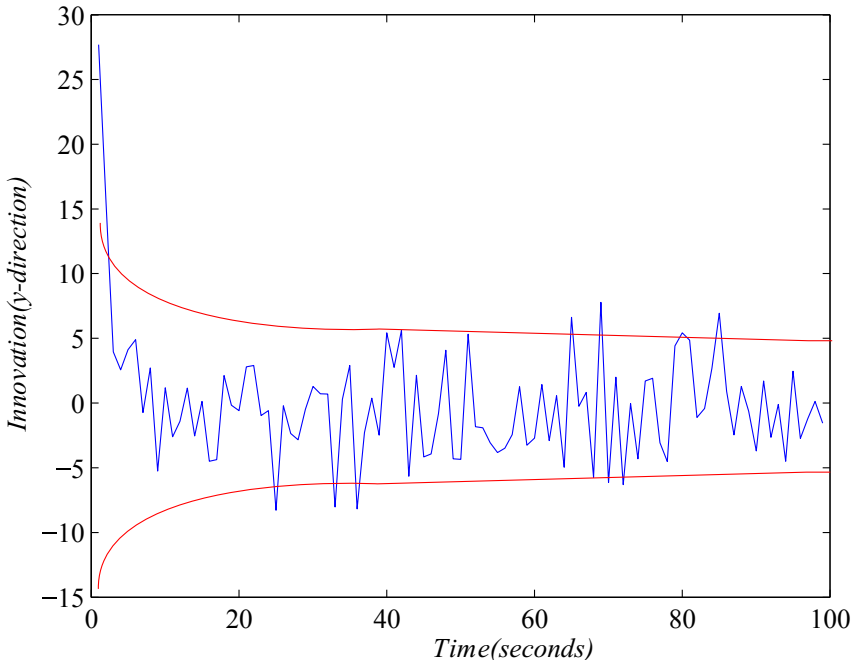

Figure 4.9: The innovation in the y-direction with the $\pm 2\sigma$ confidence bounds shown.

Table 4.4: Simulation parameters for the Viterbi track stitching algorithm

Parameter symbol	Description	Parameter value
$p(V\|V)$	Probability of the true target staying in the visible state, given it is in the visible state.	0.8
$p(V\|O)$	Probability of the true target transitioning to the visible state, given it is in the occluded state.	0.4
$p(O\|V)$	Probability of the true target transitioning to the occluded state, given it is in the visible state.	0.2
$p(O\|O)$	Probability of the true target staying in the occluded state, given it is in the occluded state.	0.6

4.3.1 Track fragment generation

In order to simulate track fragments, the model Markov model described in the previous chapter is used to create true track fragments from the generated true target track. This model with the chosen

transition probabilities is shown in Figure 4.10.

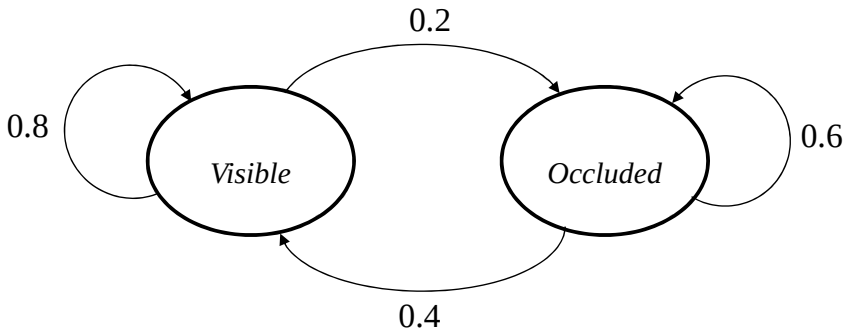

Figure 4.10: The Markov model with transition probabilities shown to create the track fragments.

The values of the transition probabilities where chosen such that the model produced bursts of true target fragments from the original true target tracks.

Figure 4.11 shows the result of creating bursts of true track fragments from a true target track.

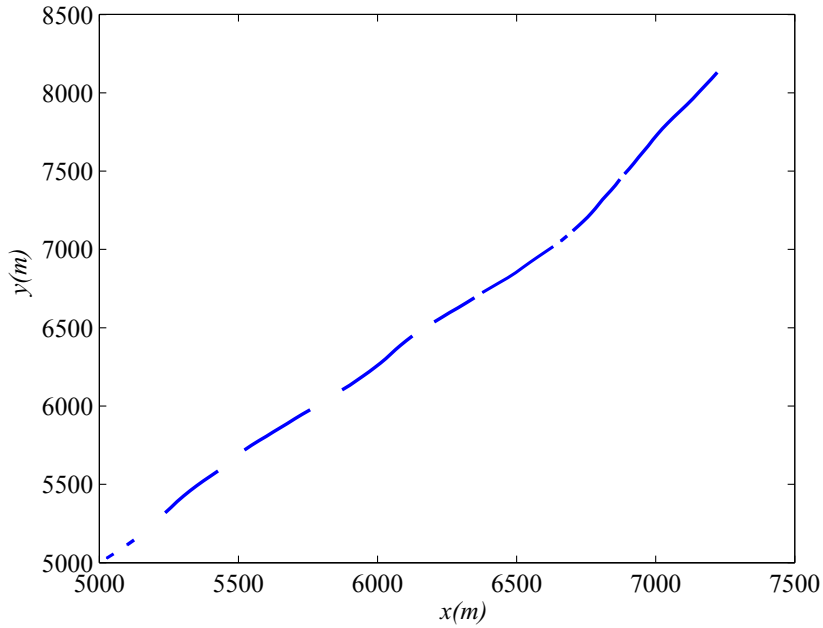

Figure 4.11: The true track of a single target with fragmentation.

Estimated track fragments are generated by the tracking algorithm when tracking noisy observations as depicted in Figure 4.11. These track fragments will will ordinarily be deleted by the M/N track maintenance algorithm when a sufficient amount missed detections (due to occlusions occur).

4.3.2 Tracking results with track fragmentation

The generated target with the fragmentation incurred as described in the previous subsection was tracked using the tracking algorithm described earlier.

Tracking the noisy observations generated from the true target track fragments resulted in a 100% track loss. The track was repeatedly dropped, and then reinitialised at a later stage, only to be dropped again. The estimated track fragments for the true target track in Figure 4.11 are shown in Figure 4.12. The original true track is also shown in Figure 4.12, along with a number of observations shown as dots.

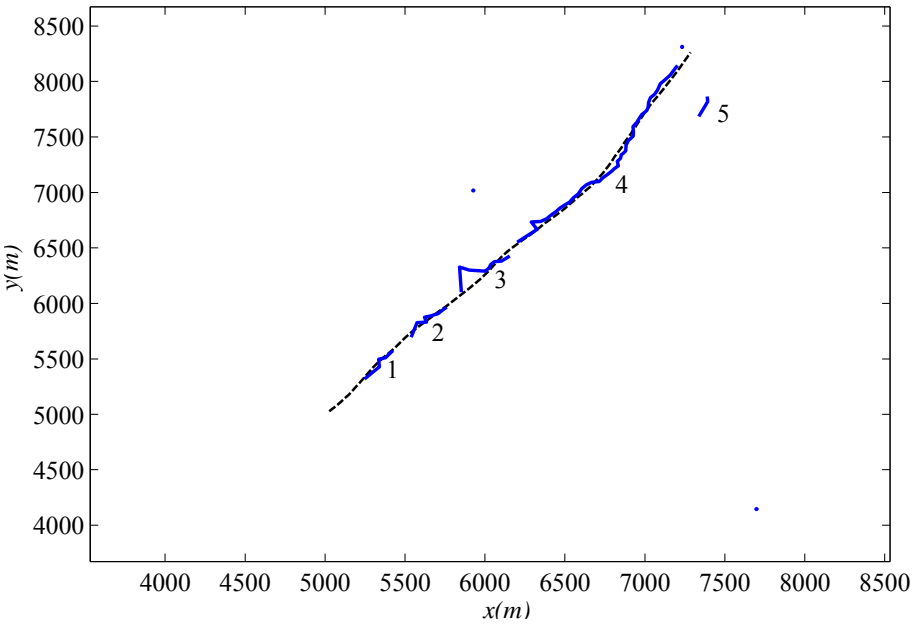

Figure 4.12: The estimated track fragments created by tracking the noisy observations generated from the true target track fragments. The track fragments have been numbered and correlates the the track fragment numbers shown in Figure 4.14.

The track fragments are superimposed over the true target track. These track fragments will usually be deleted, but in this study certain features of the track fragments are retained as described in the previous chapter. These features can be used to stitch the track fragments together.

4.3.3 Track stitching results

Attention is now shifted to the actual stitching results of the track fragments, using the sequential Viterbi approach discussed earlier.

4.3.3.1 Possible associations

The possible track fragment to track fragment associations for the estimated track fragments shown in Figure 4.12 are determined by the track gating algorithm discussed in the previous chapter.

Figure 4.13 shows a possible track fragment to track fragment association through the gating procedure taken from the scenario in Figure 4.12. The gate around the predicted position of the earlier track becomes larger as the prediction is made for more time steps. This is because of the increase in the Kalman propagator covariance. The increase in gate size accounts not only for uncertainty in the target position, but also for small deviations in the assumed straight line trajectory.

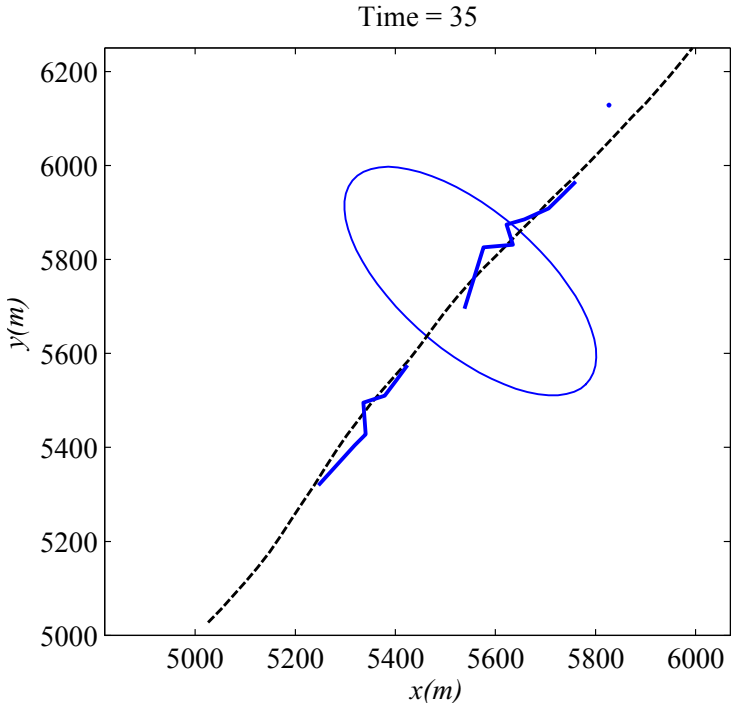

Figure 4.13: The starting position of a later track falls within the gate around the predicted position of an earlier track, resulting in a possible association.

As can be seen from Figure 4.13 the starting position of a later track falls within the gate around

the predicted position of an earlier track, making an association between these two track fragments possible.

The number of associations obtained from the tack fragment to track fragment gating procedure can be reduced by exploiting the epoch information of the track fragments. Figure 4.14 shows epochs of each of the track fragments for the scenario in Figure 4.12.

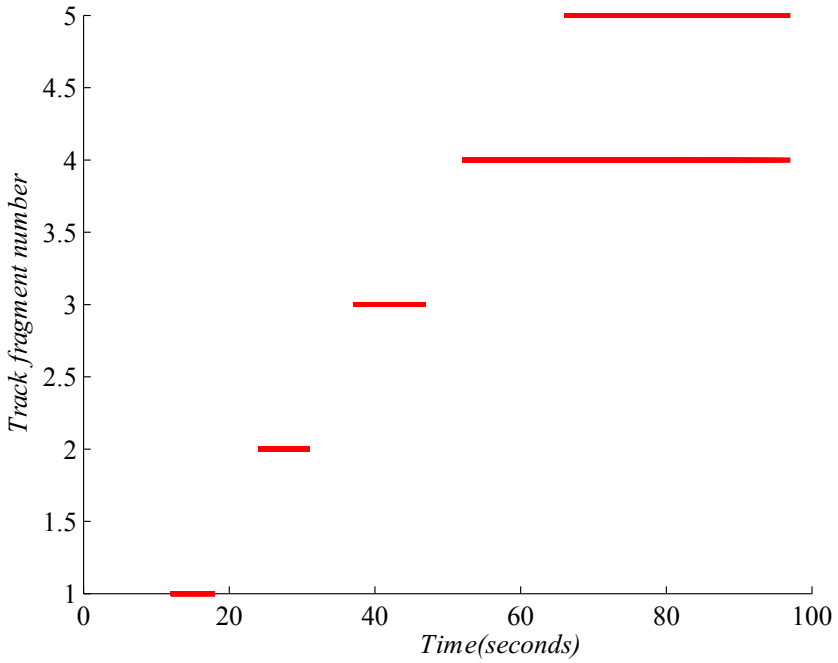

Figure 4.14: The track fragment epochs. The target track fragments are numbered from 1 to 4, while the false track is numbered as track 5.

As can be seen from Figure 4.14, the false track fragment (track fragment 5) overlaps in time with a true track fragment (track fragment 4), rejecting such an association.

Utilising the gating information of the track to track associations and the epoch information of the track fragments, a binary association matrix can be constructed. For the scenario presented in Figure 4.12, the binary association matrix is shown in Figure 4.15. In Figure 4.15, white blocks denote possible associations, while black blocks denotes an impossible association. The white blocks on the diagonal denote the *new track* hypotheses for each of the track fragments. These associations are represented by the horizontal arrows in the track trellis shown in the previous chapter or alternatively by the arrows returning to the same node in the state diagram in the previous chapter.

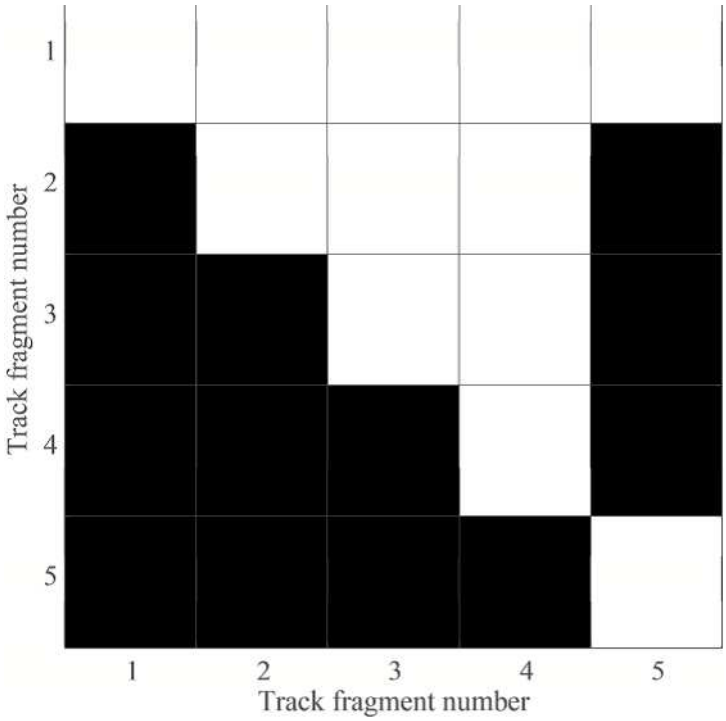

Figure 4.15: The association matrix for the scenario in Figure 4.12. The target track fragments are numbered from 1 to 4, while the false track is numbered as track 5.

4.3.3.2 Stitching results

The track trellis model along with the sequential Viterbi data association model, discussed in the previous chapter is now used to perform track stitching. The missing data is linearly interpolated to produce the stitched track. The stitched tracks with the missing data linearly interpolated for the scenario in Figure 4.12 is shown in Figure 4.16.

As can be seen from Figure 4.16, the sequential Viterbi algorithm correctly stitches the track fragments together, resulting in a continuous target track for the true target. The false track is deemed to have been created by a new target, and no associations to this track fragment has been made.

4.3.4 Multiple targets

Attention is now shifted to the multiple target case. In this subsection, two crossing targets with fragmented true target tracks are simulated. The targets are simulated in such a way as to increase the ambiguity when performing track fragment to track fragment associations. Three cases are simulated

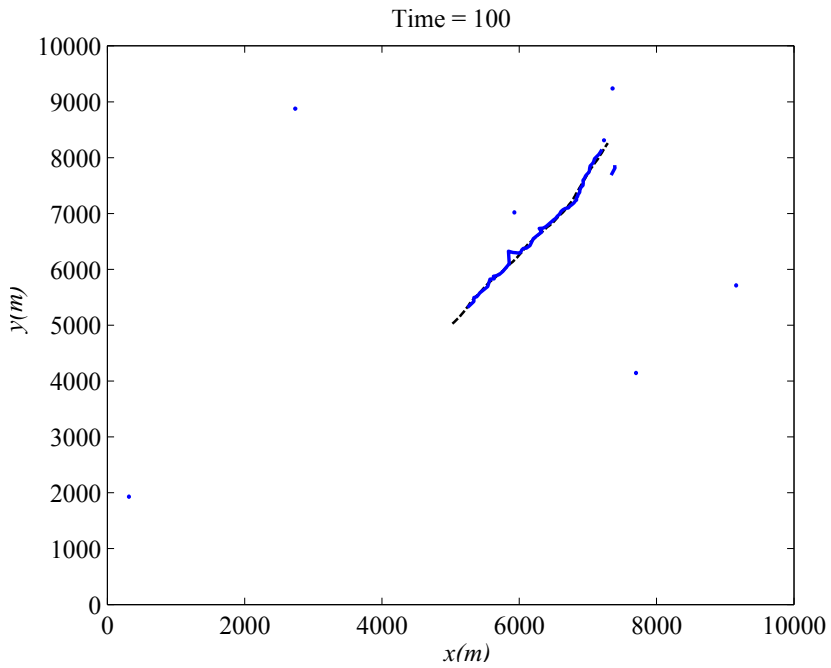

Figure 4.16: The stitched tracks with the missing data linearly interpolated.

to show the performance of the sequential Viterbi track stitching algorithm.

4.3.4.1 Case 1: Targets crossing almost perpendicularly

In this case the two true targets were initialised with the following initial state vectors:

$$x_{0_1} = [5000,\ 5000,\ 25,\ 25]^\top \text{and}$$

$$x_{0_2} = [8000,\ 5000,\ -25,\ 25]^\top.$$

These state vectors have the form described earlier in equation 2.3. Figure 4.17 shows the true target tracks of two targets crossing almost perpendicularly. In this case there exists very little ambiguity in associating the track fragments after the crossing.

The tracking algorithm was applied to the scenario shown in Figure 4.17 and the track fragment features where retained to perform track stitching. Figure 4.18 shows the estimated track fragments after being tracked by the tracking algorithm.

Again, the epochs of the track fragments are of interest. The epochs are shown in Figure 4.19 for the estimated track fragments shown in the scenario of Figure 4.18.

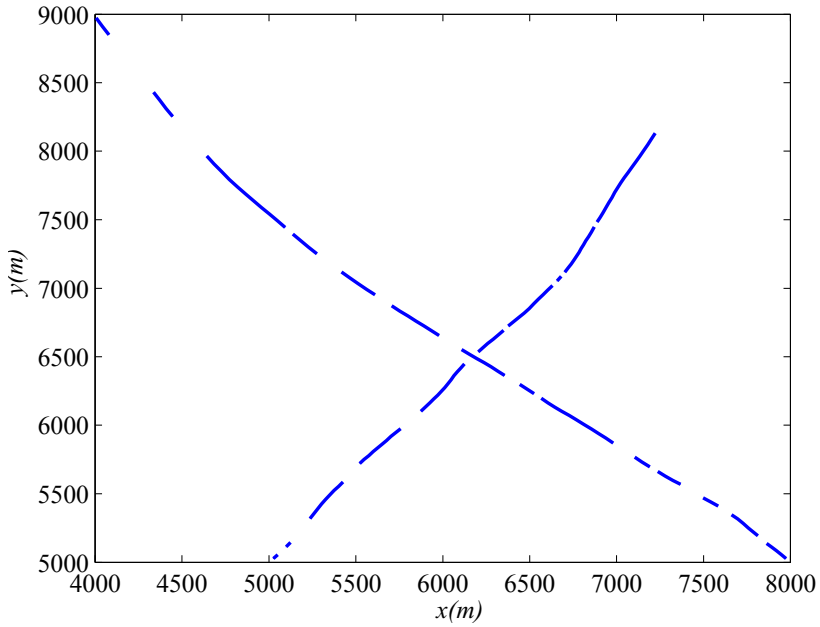

Figure 4.17: The true track fragments for two targets.

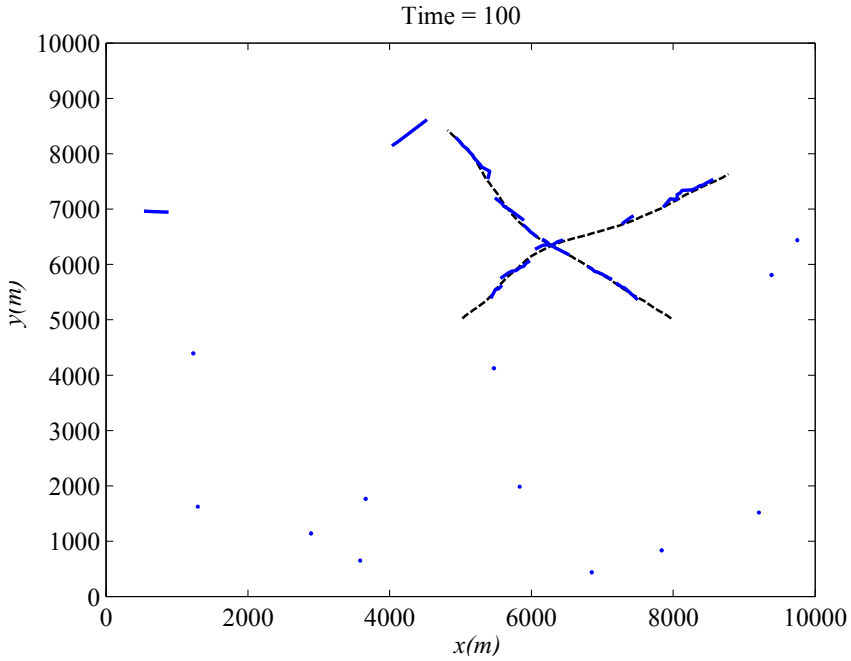

Figure 4.18: The true track fragments for two targets in case 1.

As can be seen as from Figure 4.19, the number of overlapping track fragment epoch periods increased substantially in the multiple target case. The overlapping occurrences in track fragment epoch information is consequently used to reduce the number of valid associations.

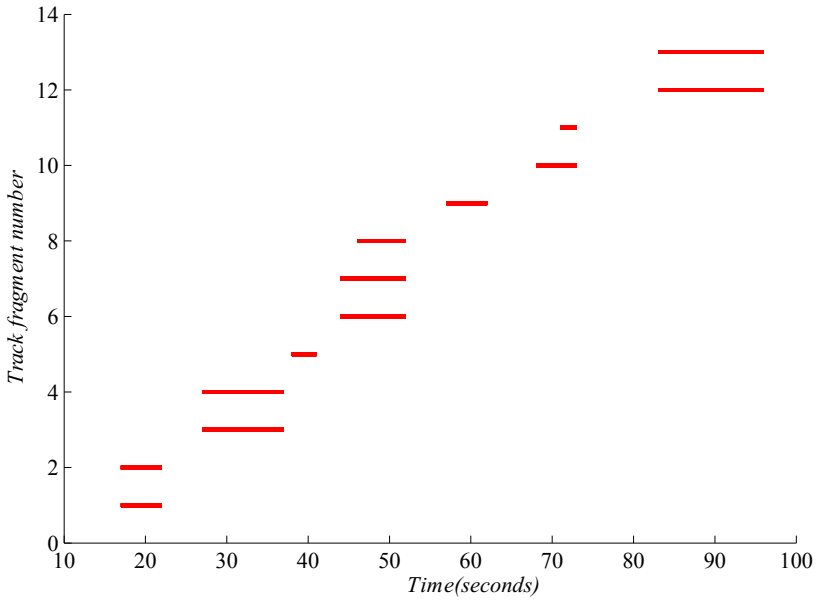

Figure 4.19: The epochs of the of the track fragments for the two targets.

As discussed in the the single target case, the track fragment are also gated, centred around the predicted position from each track fragment. Using the valid associations and time track fragment epoch information, a binary association matrix for the scenario in Figure 4.18 can be constructed as in the single target case. This is shown in Figure 4.20.

Again the white blocks indicate valid associations while the black blocks indicate impossible associations. When comparing Figure 4.20 and Figure 4.15, it can be seen that the number of associations grows exponentially with time, the number of targets and the amount of track fragments.

Using the track trellis model described in the previous chapter and the above information, the most probable sequence of track fragments can be determined for this case. Again the missing data is linearly interpolated. The result of applying the sequential Viterbi track stitching algorithm to the scenario in Figure 4.18 and linearly interpolating the missing data is shown in Figure 4.21.

As can be seen from Figure 4.21, the sequential Viterbi track stitching algorithm correctly stitches the track fragments together.

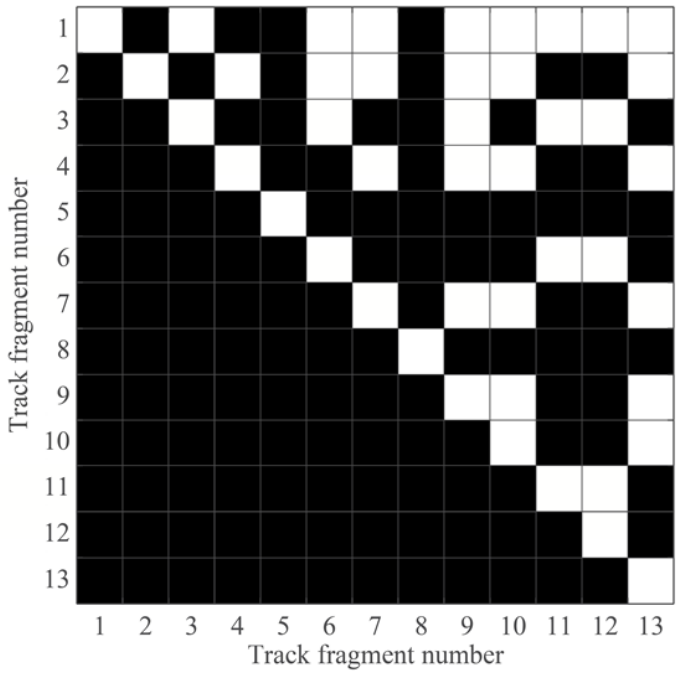

Figure 4.20: The possible associations for the scenario in Figure 4.18.

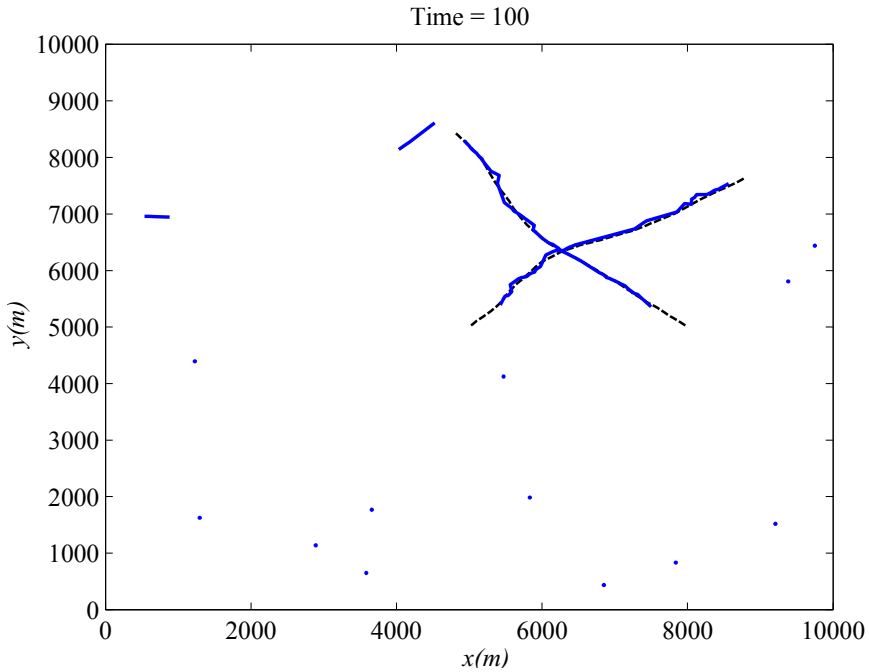

Figure 4.21: The stitched tracks with the missing data linearly interpolated, for the scenario in Figure 4.18.

4.3.4.2 Case 2: Targets crossing at an acute angle

In this case the two true targets were initialised with the following initial state vectors:

$$x_{0_1} = [5000,\ 5000,\ 20,\ 30]^\top \text{and}$$

$$x_{0_2} = [7000,\ 5000,\ -20,\ 30]^\top.$$

Figure 4.22 shows the true track fragments for the two targets crossing at an acute angle, thereby increasing the ambiguity in associating track fragments to track fragments.

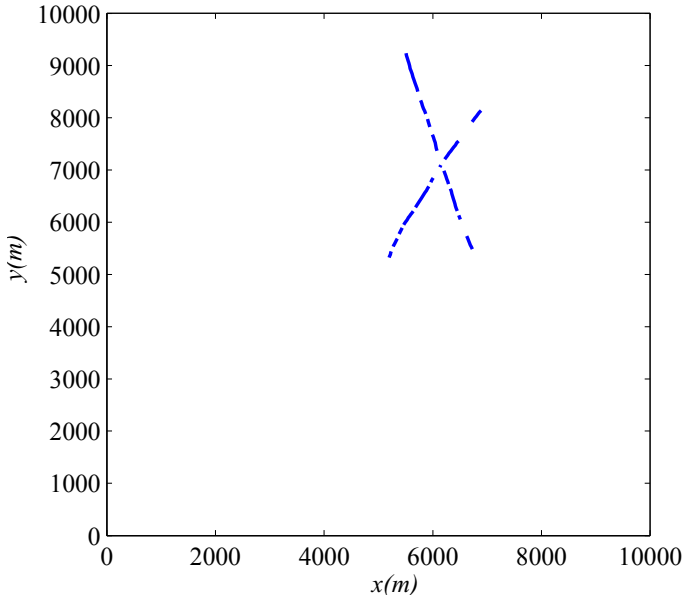

Figure 4.22: The true track fragments for two targets in case 2.

The estimated track fragments produced when applying the tracking algorithm to the scenario in Figure 4.22 is shown in Figure 4.23.

The result of applying the sequential Viterbi track stitching algorithm to the scenario in Figure 4.22 and linearly interpolating the missing data is shown in Figure 4.24.

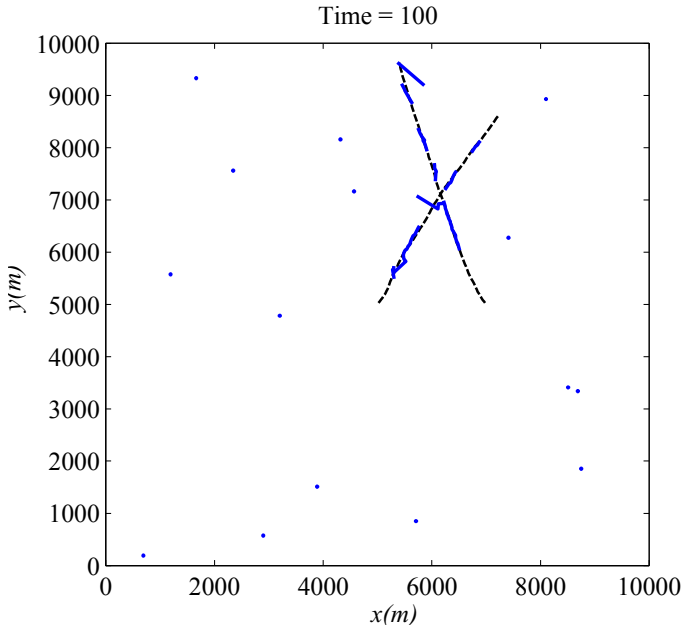

Figure 4.23: The estimated target tracks for the scenario in Figure 4.22.

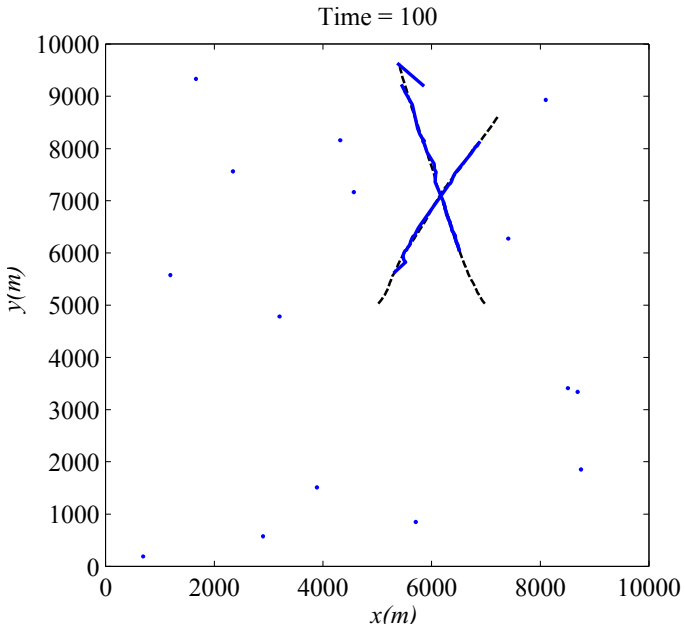

Figure 4.24: The stitched tracks with the missing data linearly interpolated, for the scenario in Figure 4.23.

As can be seen from Figure 4.24, the sequential Viterbi algorithm correctly stitched the track fragments together in this case, even with the added ambiguity

4.3.4.3 Case 3: Targets moving crossing and moving parallel to each other

The final case that is considered is when very high ambiguity exists when performing track fragment to track fragment association. In this case the two true targets were initialised with the following initial state vectors:

$$x_{0_1} = [5000,\ 5000,\ 5,\ 35]^\top \text{and}$$

$$x_{0_2} = [5500,\ 5000,\ -5,\ 35]^\top.$$

Figure 4.25 shows the true track fragments for the two targets moving parallel to each other after crossing.

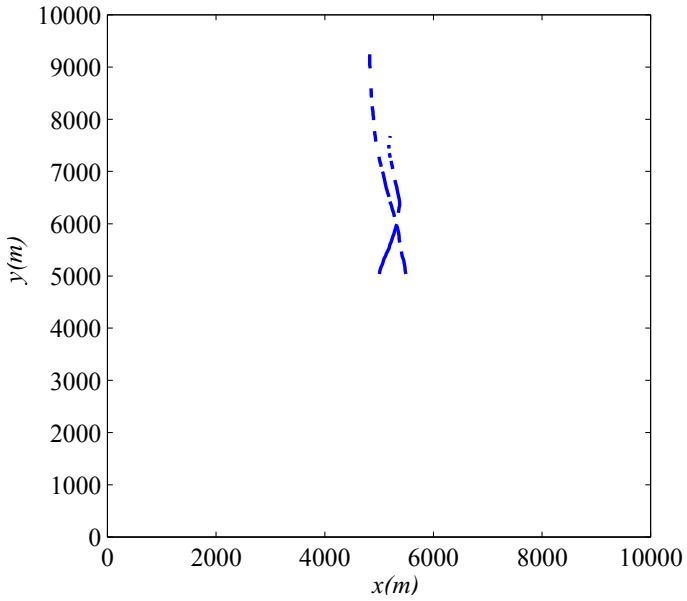

Figure 4.25: The true track fragments for two targets in case 2.

The estimated track fragments produced when applying the tracking algorithm to the scenario in Figure 4.25 is shown in Figure 4.26.

As can be seen from Figure 4.26, the way that the targets cross, and the few track fragments available before and after the crossing leads to an extremely high level of ambiguity in associating track fragments to each other.

The result of applying the sequential Viterbi track stitching algorithm to the scenario in Figure 4.25 and linearly interpolating the missing data is shown in Figure 4.27.

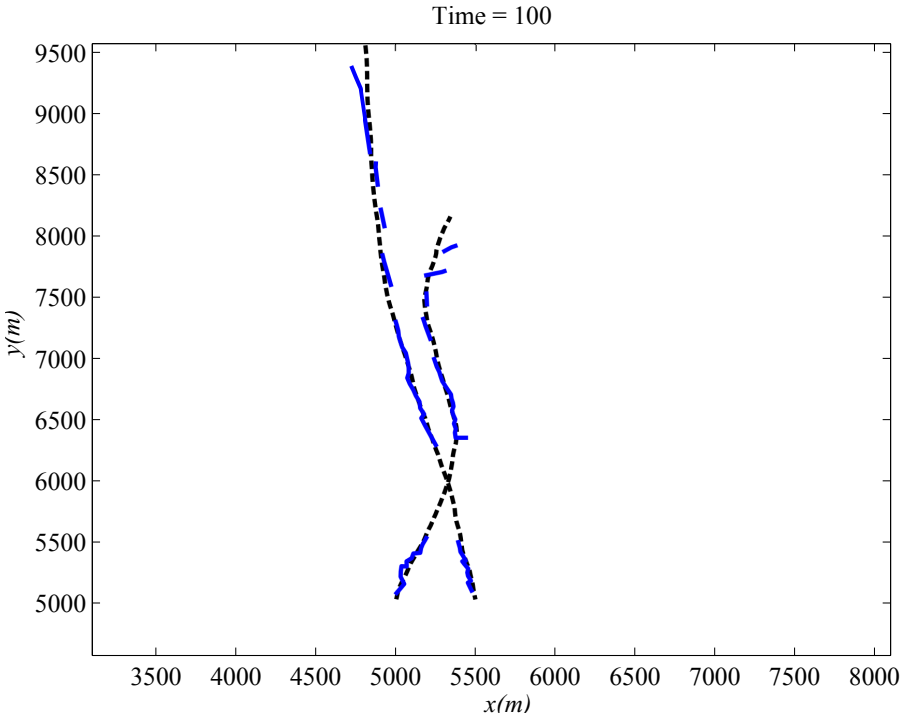

Figure 4.26: The estimated target tracks for the scenario in Figure 4.25.

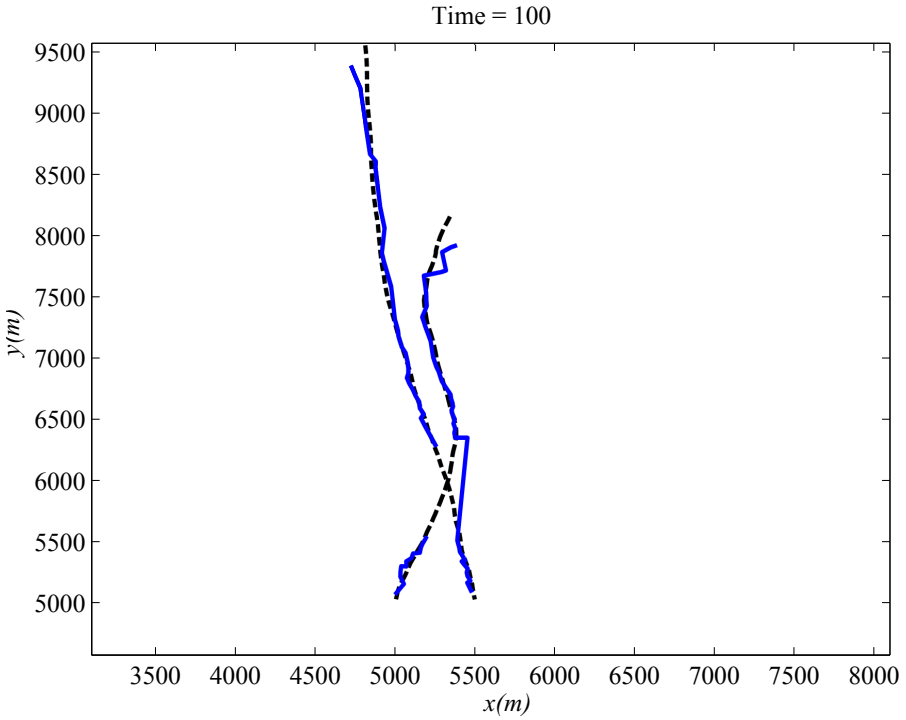

Figure 4.27: The stitched tracks with the missing data linearly interpolated, for the scenario in Figure 4.26.

As can be seen from Figure 4.27 track fragment coalescence occurs in this case because of the lack of track fragments at the crossing point causing the track fragment to track fragment association to be incorrect.

The MHT approach correctly stitched the track fragments in this case as can be seen in Figure and 4.28.

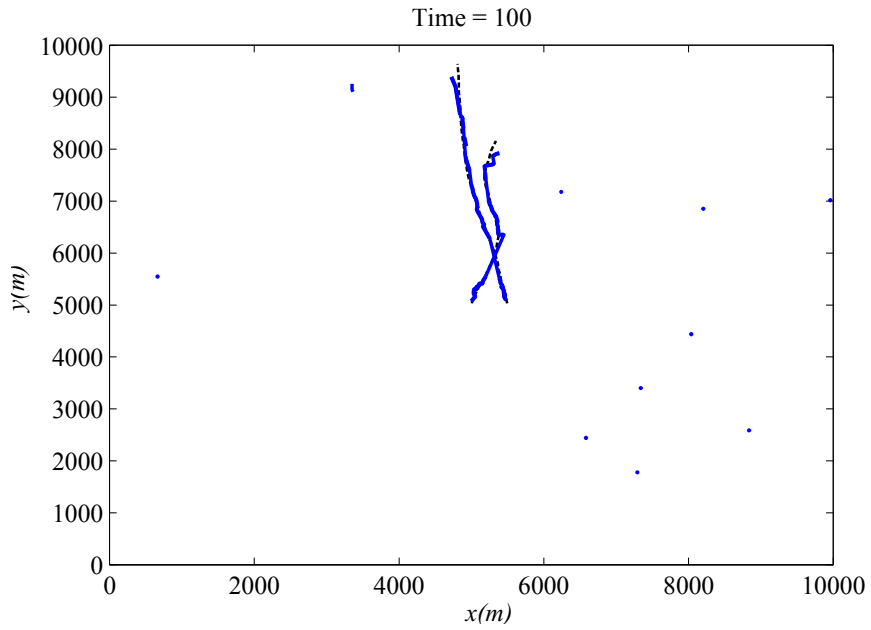

Figure 4.28: The stitched tracks with the missing data linearly interpolated, for the scenario in Figure using MHT track stitching.

It turns out that the Viterbi algorithm succeeds at this level of ambiguity when there are a sufficient number of track fragments where the ambiguity is observed. In Figures 4.29 and 4.30 two targets are crossing multiple times, but because of a sufficient amount of track fragments the Viterbi track stitching algorithm succeeds.

4.4 SEQUENTIAL VITERBI TRACK STITCHING VS. MULTIPLE HYPOTHESIS TRACK STITCHING

In order to evaluate the performance and effectiveness of the developed sequential Viterbi track stitching algorithm, a comparison in effectiveness and efficiency is made with the multiple hypothesis track stitching algorithm. In both implementations the MATLAB code used to simulate the algorithms

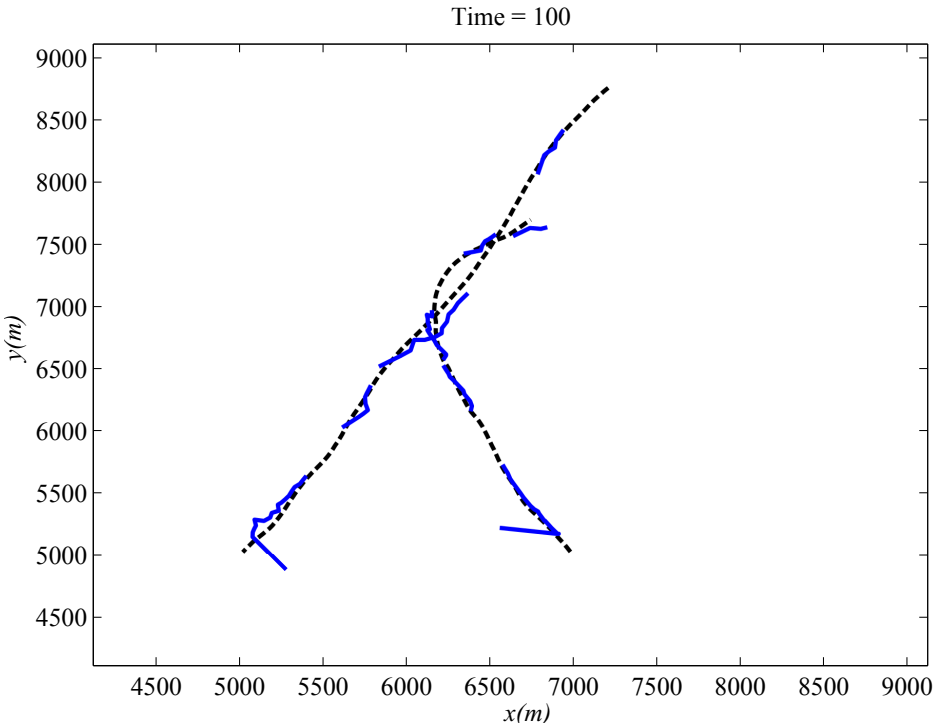

Figure 4.29: The unstitched track fragments where a sufficient amount of track fragments are present.

optimised and vectorised as far as possible using the MATLAB code profiler to ensure comparable sets of results.

The simulation parameters for the multiple hypothesis track stitching algorithm are shown in Table 4.5. The transition probabilities used to create the track fragments remained the same as in the pre-

Table 4.5: Simulation parameters for the Viterbi track stitching algorithm

Parameter symbol	Description	Parameter value
β_{NT}	New track rate	11×10^{-11}
β_{FT}	False track rate	9×10^{-11}
ε_{MHTS}	Maximum deviation allowed for Auction algorithm in MHTS track fragment association	0.5

vious section. The chosen maximum number of hypotheses parameter, N_h and the chosen pruning interval, N_P, were varied to evaluate the effect that these parameters have on the efficiency and performance of the MHTS algorithm. The results of these variations over 150 Monte Carlo simulations are

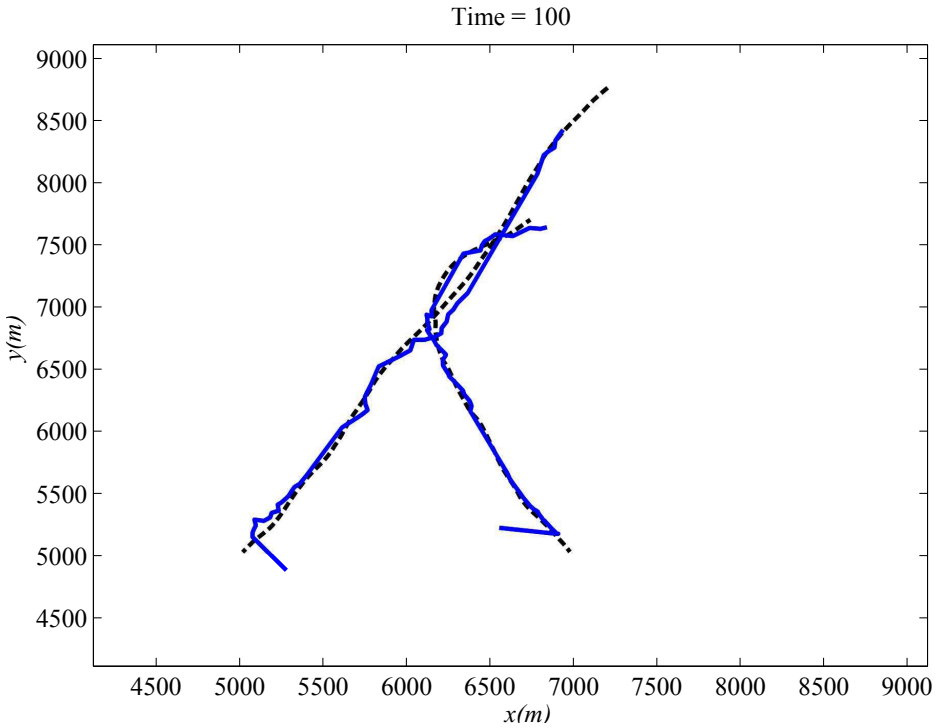

Figure 4.30: The stitched track fragments using the Viterbi track stitching algorithm for the case in Figure 4.29

shown in Table 4.6 for each of the three cases presented earlier. Only the most probable hypothesis is considered when determining the performance of the algorithm for every simulation run. As can be seen from the Table 4.6, the maximum number of hypothesis considered has a substantial effect on the performance of the multiple hypothesis track stitching algorithm. The pruning time also influences the performance of the algorithm, but to a lesser extent. In all cases in table 4.6, a substantial increase is recorded in the execution time when the maximum number of hypotheses are increased and when the pruning time is increased.

The fusing and pruning times influencing the sequential Viterbi track stitching algorithm were varied to evaluate the effect that these parameters have on the efficiency and performance of the algorithm. The results of these variations over 150 Monte Carlo simulations are shown in Table 4.7.

As can be seen from Tables 4.6, and 4.7, when all hypotheses are considered by the multiple hypothesis track stitching algorithm, the multiple hypothesis track stitching algorithm performs better than the sequential Viterbi track stitching algorithm. This increase in performance however, requires an enumeration of all the hypothesis, decreasing the efficiency of the algorithm severely. When compa-

Table 4.6: Evaluation of the multiple hypothesis track stitching algorithm.

Case	N_h	N_P (time steps)	**Average execution time over 100 time steps (seconds)**	**Correct track stitching probability**
Case 1	10	15	22.32 ± 3.32	0.83 ± 0.16
	10	5	20.14 ± 3.89	0.79 ± 0.19
	5	15	16.34 ± 3.84	0.72 ± 0.26
	5	5	14.38 ± 4.03	0.68 ± 0.33
Case 2	10	15	23.03 ± 4.14	0.85 ± 0.15
	10	5	21.85 ± 3.79	0.76 ± 0.23
	5	15	15.27 ± 3.47	0.69 ± 0.26
	5	5	13.51 ± 4.21	0.67 ± 0.23
Case 3	10	15	24.82 ± 3.38	0.71 ± 0.29
	10	5	22.04 ± 3.21	0.67 ± 0.32
	5	15	16.13 ± 4.04	0.58 ± 0.35
	5	5	14.01 ± 4.19	0.56 ± 0.28

ring the two algorithms in cases where the simulation times are close to each other, it can be seen that the sequential Viterbi algorithm performs marginally better.

Finally, the simulation variables where chosen in such a way as to obtain similar execution times between the two algorithms as

$$N_{P_{Vi}} = 15 \text{ time steps}, \tag{4.2}$$

$$N_F = 10 \text{ time steps}, \tag{4.3}$$

$$N_{P_{MHT}} = 5 \text{ time steps}, \tag{4.4}$$

$$N_h = 5 \text{ hypotheses}. \tag{4.5}$$

Here the $N_{P_{MHT}}$ and $N_{P_{Vi}}$ denote the purging interval for the MHT and Viterbi algorithms respectively, while N_F denotes the fusing interval for both algorithms and N_h denotes the number of hypotheses in the MHT algorithm. The level of ambiguity was set to the first level as described previously.

This setup allowed for a fair comparison between the performances of the two algorithms. Figure 4.31 shows the mean and 1-σ covariance Gaussian ellipse for the two cases. From the means it

Table 4.7: Evaluation of the sequential Viterbi track stitching algorithm.

Case	N_P (time steps)	N_F (time steps)	**Average execution time over 100 time steps (seconds)**	**Correct track stitching probability**
Case 1	15	10	14.22 ± 4.14	0.79 ± 0.21
	15	5	11.23 ± 4.11	0.78 ± 0.21
	5	10	9.76 ± 4.21	0.76 ± 0.23
	5	5	8.12 ± 4.15	0.72 ± 0.27
Case 2	15	10	14.53 ± 4.23	0.81 ± 0.19
	15	5	12.34 ± 4.17	0.73 ± 0.25
	5	10	9.93 ± 4.12	0.74 ± 0.23
	5	5	8.87 ± 4.09	0.70 ± 0.26
Case 3	15	10	13.32 ± 4.13	0.68 ± 0.29
	15	5	11.87 ± 4.09	0.65 ± 0.31
	5	10	9.96 ± 4.12	0.62 ± 0.30
	5	5	7.27 ± 4.03	0.59 ± 0.33

can be seen that the sequential Viterbi track stitching algorithm performs somewhat better than the multiple hypothesis track stitching approach for similar execution times. From Figure 4.31 it can further be seen that the MHT based algorithm resulted in more cases even higher execution times still produced a low track stitching probability. This indicates that the Viterbi based algorithm produces more consistent results than that of the MHT based algorithm where similar execution times are considered.

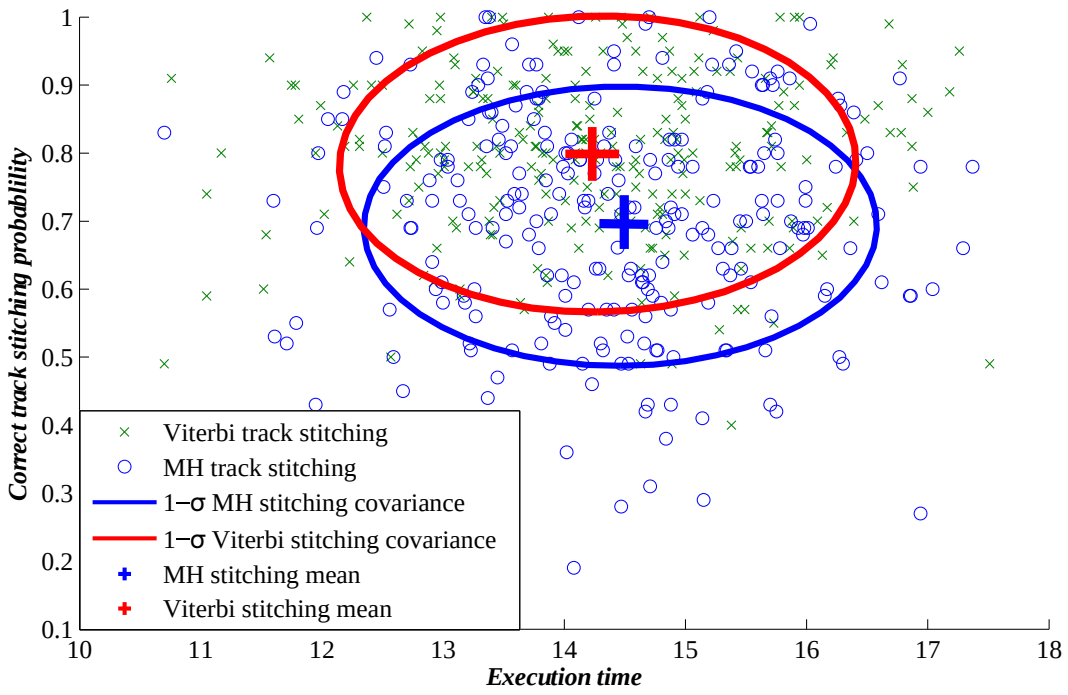

Figure 4.31: A comparison between MH track stitching and sequential Viterbi track stitching.

CHAPTER 5

CONCLUSION

"Now this is not the end. It is not even the beginning of the end. But it is, perhaps, the end of the beginning."

Winston Churchill (1874-1965)

In this study it was shown that track fragment stitching can be performed by using a graphical model and a Viterbi algorithm approach. The track fragments were modelled as a Markov model, and restructured into a time dependent lattice (track graph). A sequential Viterbi algorithm was developed to solve the track graph, and determine the most probable set of track fragments.

It was shown that the developed algorithm performed somewhat better when compared against a hypothesis based multiple hypothesis track stitching algorithm. It was also shown that an increase in possible combinations and the maximum number of hypotheses resulted in long execution times in the multiple hypothesis track stitching algorithm. However, when reducing the maximum number of hypotheses and the number of combinations the execution time of the algorithm improved, while the performance decreased.

A similar analysis was performed on the sequential Viterbi track stitching algorithm. It was shown that the sequential Viterbi algorithm performs somewhat better than the MHT track stitching algorithm for similar execution times, however, the Viterbi algorithm produces more consistent results where, the MHT based algorithm resulted in more cases where higher execution times still produced a low track stitching probability. This indicated that the Viterbi based algorithm produces more consistent results than that of the MHT based algorithm where similar execution times are considered.

5.1 FUTURE WORK

Future work includes evaluating the algorithm with real world data, and higher target densities and includes modelling the track fragments using a more general graphical model. These models can then be solved by using more advanced algorithms [15], [4], [19].

Future work includes extending this work into a three dimensional space with a higher target density and also further comparing the track stitching algorithms considered in this study to other track stitching algorithms in the literature. In this study a simple linear interpolation of the missing positions was performed. Future work includes considering more advanced smoothing algorithms to interpolate in between the track fragments that satisfy the motion model [10], [11].

REFERENCES

[1] R. Ivey, J. Horn, and R. Merket, "Long-duration fused feature learning-aided tracking," *Multisensor, Multisource Information Fusion: Architecture, Algorithms and Applications*, Apr. 2012.

[2] Z. Chen, L. Chen, M. Cetin, and A. Willsky, "An efficient message passing algorithm for multi-target tracking," in *12th International Conference on Information Fusion*, Jul. 2009, pp. 826 –833.

[3] Y. Bar-Shalom and X. Li, *Multitarget-Multisensor Tracking:Principles and Techniques*. YBS Publishing, 1995.

[4] Z. Wu, T. Kunz, and M. Betke, "Efficient track linking methods for track graphs using network-flow and set-cover techniques," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2011, pp. 1185–1192.

[5] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE Transactions on Information Theory*, vol. 13, no. 2, pp. 260 –269, Apr. 1967.

[6] D. Reid, "An algorithm for tracking multiple targets," *IEEE Transactions on Automatic Control*, vol. 24, no. 6, pp. 843 – 854, Dec. 1979.

[7] C. Chong, G. Castanon, N. Cooprider, S. Mori, R. Ravichandran, and R. Macior, "Efficient multiple hypothesis tracking by track segment graph," in *12th International Conference on Information Fusion*, Jul. 2009, pp. 2177 –2184.

[8] A. A. Markov, *The theory of algorithms.* Acad. Sci. USSR, 1951.

[9] S. Mori and C. Chong, "Performance analysis of graph-based track stitching," in *16th International Conference on Information Fusion*, 2013, pp. 196–203.

[10] T. Ding, M. Sznaier, and O. Camps, "Fast track matching and event detection," in *IEEE Conference on Computer Vision and Pattern Recognition*, vol. vol.2, Jun. 2008, pp. 1–8.

[11] T. Ding, O. Camps, and M. Sznaier, "A rank minimization approach to fast dynamic event detection and track matching in video sequences," in *46th IEEE Conference on Decision and Control*, Dec 2007, pp. 4122–4127.

[12] R. Lublinerman, M. Sznaier, and O. Camps, "Dynamics based robust motion segmentation," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, June 2006, pp. 1176–1184.

[13] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of basic Engineering*, vol. 82, no. 1, pp. 35–45, 1960.

[14] G. Taylor, A. Kanaujia, K. Ramnath, and N. Haering, "A portable geo-aware surveilance system for vehicles," in *12th IEEE International Conference on Computer Vision*, vol. vol.2, Sept. 2009, pp. 1267–1274.

[15] A. Abrantes, J. Marques, and J. Lemos, "Long term tracking using Bayesian networks," in *International Conference on Image Processing*, vol. 3, June 2002, pp. 609–612 vol.3.

[16] S. Zhang and Y. Bar-Shalom, "Track segment association for GMTI tracks of evasive move-stop-move maneuvering targets," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 47, no. 3, pp. 1899–1914, July 2011.

[17] S. Blackman and R. Popoli, *Design and Analysis of Modern Tracking Systems*. Artech House, 1999.

[18] S. Blackman, *Multiple-Target tracking with radar applications*, Dedham, Ed. Artech House, 1986.

[19] G. Castanon and L. Finn, "Multi-target tracklet stitching through network flows," in *IEEE Aerospace Conference*, Mar. 2011, pp. 1 –7.

[20] D. Salmond, "Tracking and guidance with intermittent obscuration and association uncertainty," in *16th International Conference on Information Fusion*, 2013, pp. 691–698.

[21] G. D. Forney, "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268 – 278, Mar. 1973.

[22] T. E. Fortmann, Y. Bar-Shalom, and M. Scheffe, "Sonar tracking of multiple targets using joint probabilistic data association," *IEEE Journal of Oceanic Engineering*, vol. 8, no. 3, pp. 173–184, 1983.

[23] Y. Bar-Shalom and X. Li, *Estimation with Applications to Tracking and Navigation*. Wiley-Interscience, 2001.

[24] M. Sharpe, *General theory of Markov processes*. Access Online via Elsevier, 1988, vol. 133.

[25] C. M. Bishop, *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Secaucus, NJ, USA: Springer-Verlag New York, Inc., 2006.

[26] R. J. Elliott, *Hidden Markov models: estimation and control*. New York: Springer-Verlag, 1995, vol. 29.

[27] B. S. Everitt, *Introduction to Latent Variable Models*, H. Toutenburg, Ed. Chapman and Hall, London, 1984.

[28] L. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257 –286, Feb. 1989.

[29] Y. Ephraim, D. Malah, and B.-H. Juang, "On the application of hidden Markov models for enhancing noisy speech," *IEEE Transactions on Acoustics, Speech and Signal Processing*, vol. 37, no. 12, pp. 1846–1856, 1989.

[30] M. Stanke and S. Waack, "Gene prediction with a hidden Markov model and a new intron submodel," *Bioinformatics*, vol. 19, no. suppl 2, pp. ii215–ii225, 2003.

[31] X. R. Li and V. P. Jilkov, "Survey of maneuvering target tracking. Part I. Dynamic models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 39, no. 4, pp. 1333 – 1364, Oct. 2003.

[32] X. Li and V. Jilkov, "Survey of maneuvering target tracking. Part III. Measurement models." *Proceedings of SPIE Conference on Signal and Data Processing*, pp. 423–466, 2001.

[33] M. Mallick and B. L. Scala, "Comparison of single-point and two-point difference track initiation algorithms using position measurements," *Acta Automatica Sinica*, vol. 34, no. 3, pp. 258 – 265, 2008.

[34] C. Morefield, "Application of 0-1 integer programming to multitarget tracking problems," *IEEE Transactions on Automatic Control*, vol. 22, no. 3, pp. 302 – 312, Jun. 1977.

[35] H. W. Kuhn, "The Hungarian method for the assignment problem," *Naval Research Logistic Quarterly*, vol. 2, pp. 83–97, 1955.

[36] R. Jonker and A. Volgenant, "A shortest augmenting path algorithm for dense and sparse linear assignment problems," *Computing*, vol. 38, pp. 325 –340, 1987.

[37] J. Munkres, "Algorithms for the assignment and transportation problems," *Journal of the Society for Industrial and Applied Mathematics*, vol. 5, pp. 32–38, 1957.

[38] D. Bertsekas, "A distributed asynchronous relaxation algorithm for the assignment problem," in *24th IEEE Conference on Decision and Control*, vol. 24, 1985, pp. 1703–1704.

[39] Y. Bar-Shalom, S. Blackman, and R. Fitzgerald, "Dimensionless score function for multiple hypothesis tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, pp. 392–400, 2007.

[40] I. J. Cox and S. Hingorani, "An efficient implementation and evaluation of Reid's multiple hypothesis tracking algorithm for visual tracking," in *Pattern Recognition. Proceedings of the 12th IAPR International Conference on Computer Vision & Image Processing.*, vol. 1, 1994, pp. 437–442 vol.1.

[41] R. Danchick and G. E. Newnam, "Reformulating Reid's MHT method with generalised Murty K-best ranked linear assignment algorithm," *IEE Proceedings on Radar, Sonar and Navigation*, vol. 153, no. 1, pp. 13–22, 2006.

[42] G. Pulford, "Multi-target Viterbi data association," in *9th International Conference on Information Fusion*, Jul. 2006, pp. 1 –8.

[43] B. Vo, S. Singh, and A. Doucet, "Sequential Monte Carlo implementation of the PHD filter

for multi-target tracking," in *6th International Conference on Information Fusion*, 2003, pp. 792–799.

[44] J. Wolf, A. Viterbi, and G. Dixon, "Finding the best set of K paths through a trellis with application to multitarget tracking," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 25, no. 2, pp. 287 –296, Mar. 1989.

[45] R. L. Streit, "How to count targets given only the number of measurements," in *16th International Conference on Information Fusion*, 2013, pp. 1678–1685.

[46] J. Bucklew and R. Radeke, "On the Monte Carlo simulation of digital communication systems in Gaussian noise," *IEEE Transactions on Communications*, vol. 51, no. 2, pp. 267–274, 2003.

ADDENDUM A

DERIVATIONS

A.1 DERIVATION OF THE RECURSIVE KALMAN FILTER

In this section the Kalman filter equations will be derived. The Kalman filter can algorithm can be summarised as in Figure A.1.

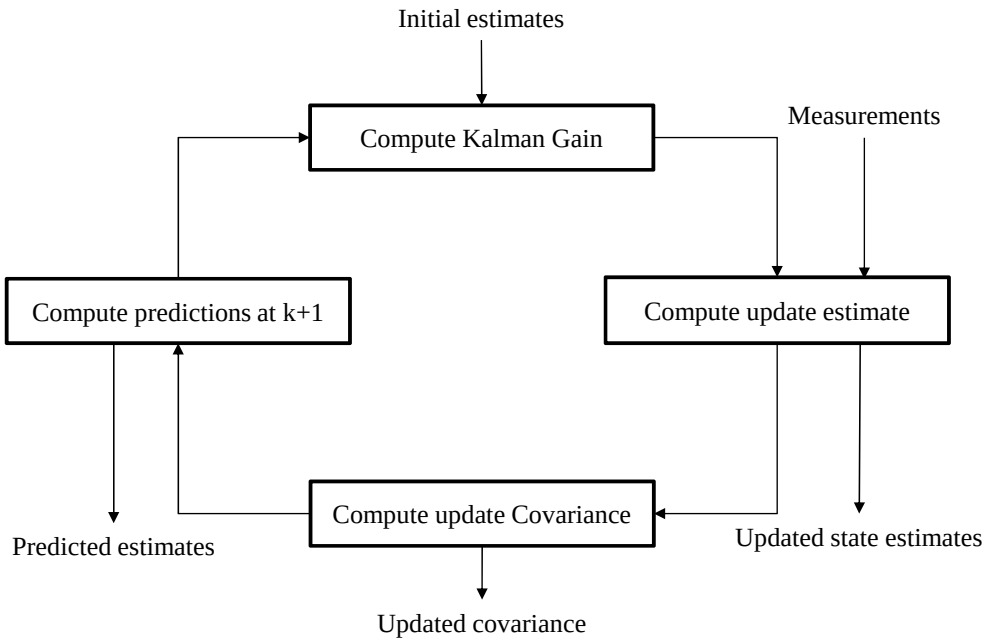

Figure A.1: The recursive Kalman filter algorithm.

The Kalman filter equations are given by:

1. Prediction equations:

(a) State extrapolation (*a priori*):

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1} \tag{A.1}$$

(b) Covariance extrapolation (*a priori*):

$$\boldsymbol{P}_{k|k-1} = \boldsymbol{\Phi}_k \boldsymbol{P}_{k-1|k-1} \boldsymbol{\Phi}_k^\top + \boldsymbol{Q}_k \tag{A.2}$$

2. Update equations:

(a) Kalman gain computation:

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top \left[\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top + \boldsymbol{R}_k \right]^{-1} \tag{A.3}$$

(b) State update (*a posteriori*):

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \left[\boldsymbol{z}_k - \boldsymbol{H}_k \boldsymbol{x}_{k|k-1} \right] \tag{A.4}$$

(c) Covariance update (*a posteriori*):

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_{k|k-1} \tag{A.5}$$

The following model is assumed:

1. The state transition model is given by

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{w}_k, \tag{A.6}$$

where $\boldsymbol{w}_k$ represents the Gaussian process noise, and $\boldsymbol{\Phi}_k$ represents a transition matrix dependent on the sample period.

2. The measurement model is given by

$$\boldsymbol{z}_k = \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k, \tag{A.7}$$

where $\boldsymbol{v}_k$ represents the Gaussian observation noise and $\boldsymbol{H}_k$ represents a transition matrix.

3. The following assumptions are also made about the Gaussian noise in the model:

(a) The expectation of the initial state vector is given by

$$\mathbb{E}\left[\boldsymbol{x}_{0|0}\right] = \boldsymbol{\mu}_0^x, \tag{A.8}$$

where $\boldsymbol{\mu}_0^x$ is the mean of the first state vector.

(b) The expectation of the process noise is zero for all time steps, k, i.e.

$$\mathbb{E}\left[\boldsymbol{w}_k\right] = 0 \forall k. \tag{A.9}$$

(c) The expectation of the observation noise is zero for all time steps, k, i.e.

$$\mathbb{E}\left[\boldsymbol{v}_k\right] = 0 \forall k. \tag{A.10}$$

(d) The covariance of the process noise is given by

$$\mathbb{E}\left[\boldsymbol{w}_k, \boldsymbol{w}_l\right] = \boldsymbol{Q}\delta_{kl}, \tag{A.11}$$

where δ_{kl} is the Kronecker delta.

(e) The covariance of the process noise is given by

$$\text{cov}\{\boldsymbol{v}_k, \boldsymbol{v}_l\} = \boldsymbol{R}\delta_{kl}. \tag{A.12}$$

(f) The initial covariance is determined by the initial state vector as

$$\text{cov}\{\boldsymbol{x}_{0|0}, \boldsymbol{x}_{0|0}\} = \boldsymbol{P}_{0|0}. \tag{A.13}$$

(g) The process noise and the measurement noise are uncorrelated and are independent of the initial state. Therefore,

$$\text{cov}\{\boldsymbol{w}_k, \boldsymbol{v}_l\} = 0 \forall k, \tag{A.14}$$

$$\text{cov}\{\boldsymbol{x}_0, \boldsymbol{w}_k\} = 0 \forall k, \tag{A.15}$$

$$\text{cov}\{\boldsymbol{x}_0, \boldsymbol{v}_l\} = 0 \forall l. \tag{A.16}$$

4. It is assumed at time k, that the unbiased estimate of the state $\hat{\boldsymbol{x}}_{k|k-1}$ is available, from time $k-1$.

5. The error term $\tilde{\boldsymbol{x}}_{k-1|k-1} = \boldsymbol{x}_{k-1} - \hat{\boldsymbol{x}}_{k-1|k-1}$ (where $\boldsymbol{x}_{k-1}$ is the true state at the previous time step) has zero mean, and covariance $\boldsymbol{P}_{k-1|k-1}$.

6. It is also assumed that the measurement $\boldsymbol{z}_k$ is available at time k.

The equation for the updated (*a posteriori*) state estimate (equation A.4) is derived first.

The goal is to find an unbiased, minimum variance estimator of the state at time k in the recursive form given by

$$\hat{\boldsymbol{x}}_{k|k} = \boldsymbol{K}_k^\top \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \boldsymbol{z}_k. \tag{A.17}$$

The restriction that the estimator $\hat{\boldsymbol{x}}_{k|k}$ needs to be unbiased leads to

$$\frac{\mathbb{E}\left[\hat{\boldsymbol{x}}_{k|k} - \boldsymbol{x}_k\right]}{\tilde{\boldsymbol{x}}_k} = 0. \tag{A.18}$$

By substituting equation A.17 into equation A.18 leads to

$$\boldsymbol{K}_k^\top \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \boldsymbol{z}_k - \boldsymbol{x}_k = 0, \tag{A.19}$$

where the term $\tilde{\boldsymbol{x}}_k$ has been cancelled and the expectation has been ignored for now. By adding and subtracting the term $\boldsymbol{K}_k^\top \boldsymbol{x}_{k-1}$ and substituting in equation A.7 for $\boldsymbol{z}_k$ leads to

$$\boldsymbol{K}_k^\top \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \left(\boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k\right) - \boldsymbol{x}_k - \boldsymbol{K}_k^\top \boldsymbol{x}_{k-1} + \boldsymbol{K}_k^\top \boldsymbol{x}_{k-1} = 0. \tag{A.20}$$

Substituting in equation A.6, factorising and rearranging the terms results in

$$\boldsymbol{K}_k^\top \left(\hat{\boldsymbol{x}}_{k-1|k-1} - \boldsymbol{x}_{k-1}\right) + \boldsymbol{x}_{k-1}\left(\boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{\Phi}_k - \boldsymbol{\Phi}_k + \boldsymbol{K}_k^\top\right) + \boldsymbol{w}_k \left(\boldsymbol{K}_k \boldsymbol{H}_k - \boldsymbol{I}\right) + \boldsymbol{K}_k \boldsymbol{v}_k = 0. \tag{A.21}$$

Taking the expectation and comparing to equation A.18 results in

$$\mathbb{E}\left[\hat{\boldsymbol{x}}_{k|k} - \boldsymbol{x}_k\right] = \mathbb{E}\left[\boldsymbol{x}_{k-1}\right]\left(\boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{\Phi}_k - \boldsymbol{\Phi}_k + \boldsymbol{K}_k^\top\right) = 0. \tag{A.22}$$

This requires that

$$\boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{\Phi}_k - \boldsymbol{\Phi}_k + \boldsymbol{K}_k^\top = 0, \tag{A.23}$$

or in the factorised form that

$$\boldsymbol{K}_k^\top = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right) \boldsymbol{\Phi}_k. \tag{A.24}$$

Thus, the unbiased estimator is given by

$$\hat{\boldsymbol{x}}_{k|k} = \left(\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k\right) \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{K}_k \boldsymbol{z}_k, \tag{A.25}$$

or equivalently

$$\hat{\boldsymbol{x}}_{k|k} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1} + \boldsymbol{K}_k \left(\boldsymbol{z}_k - \boldsymbol{H}\boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1} \right), \tag{A.26}$$

which leads to

$$\hat{\boldsymbol{x}}_{k|k} = \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \left(\boldsymbol{z}_k - \boldsymbol{H}\hat{\boldsymbol{x}}_{k|k-1} \right). \tag{A.27}$$

Which is the state update equation given by equation A.4.

The the *a posteriori* estimate covariance, $\boldsymbol{P}_{k|k}$ is derived next.

Using the factorised form of equation A.27 in the estimation error equation, given by $\tilde{\boldsymbol{x}}_{k|k} = \boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k}$, results in

$$\begin{aligned}
\tilde{\boldsymbol{x}}_{k|k} &= (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \boldsymbol{z}_k - \boldsymbol{x}_k \\
&= (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \hat{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k (\boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{v}_k) - \boldsymbol{x}_k \\
&= \hat{\boldsymbol{x}}_{k|k-1} - \hat{\boldsymbol{x}}_{k|k-1} \boldsymbol{K}_k \boldsymbol{H}_k + \boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{x}_k + \boldsymbol{K}_k \boldsymbol{v}_k - \boldsymbol{x}_k \\
&= \left(\hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{x}_k \right) - \boldsymbol{K}_k \boldsymbol{H}_k \left(\hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{x}_k \right) + \boldsymbol{K}_k \boldsymbol{v}_k \\
&= (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \left(\hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{x}_k \right) + \boldsymbol{K}_k \boldsymbol{v}_k \\
&= (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \tilde{\boldsymbol{x}}_{k|k-1} + \boldsymbol{K}_k \boldsymbol{v}_k
\end{aligned} \tag{A.28}$$

Now taking the covariance on both sides, and noting that $\boldsymbol{v}_k$ is uncorrelated with all other terms, results in

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k) \boldsymbol{P}_{k|k-1} (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k)^\top + \boldsymbol{K}_k \boldsymbol{R}_k \boldsymbol{K}_k^\top \tag{A.29}$$

This formula is valid for any value of $\boldsymbol{K}_k$ and can be further simplified into equation A.5, when $\boldsymbol{K}_k$ is the optimal Kalman gain as shown later.

The next goal is to find the optimal Kalman gain, $\boldsymbol{K}_k$. This is found by minimising the expected value of the square of the magnitude of the error term, given by

$$\mathbb{E}\left[|\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k|k}|^2 \right]. \tag{A.30}$$

This is equivalent to minimising the trace of the *a posteriori* estimate covariance matrix $\boldsymbol{P}_{k|k}$.

By expanding equation A.29 the following is obtained

$$\boldsymbol{P}_{k|k} = \boldsymbol{P}_{k|k-1} - \boldsymbol{P}_{k|k-1} \boldsymbol{K}_k^\top \boldsymbol{H}_k^\top - \boldsymbol{K}_k \left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1} \boldsymbol{H}_k^\top + \boldsymbol{R}_k \right) \boldsymbol{K}_k^\top. \tag{A.31}$$

The optimal Kalman gain, $\boldsymbol{K}_k$ is determined by taking the matrix derivative of the trace of $\boldsymbol{P}_{k|k}$ with respect to $\boldsymbol{K}_k$ and setting it equal to zero,

$$\frac{\partial \text{tr}\left(\boldsymbol{P}_{k|k}\right)}{\partial \boldsymbol{K}_k} = -2\left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1}\right)^\top + 2\boldsymbol{K}_k\left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top + \boldsymbol{R}_k\right) = 0. \tag{A.32}$$

Solving for $\boldsymbol{K}_k$ yields the optimal Kalman gain (equation A.3),

$$\boldsymbol{K}_k = \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top\left(\boldsymbol{H}_k \boldsymbol{P}_{k|k-1}\boldsymbol{H}_k^\top + \boldsymbol{R}_k\right)^{-1}. \tag{A.33}$$

Substituting equation A.33 into equation A.29 results

$$\boldsymbol{P}_{k|k} = (\boldsymbol{I} - \boldsymbol{K}_k \boldsymbol{H}_k)\boldsymbol{P}_{k|k-1}. \tag{A.34}$$

Which is the updated *a posteriori* estimate covariance (equation A.5).

Finally, the equation for the *a priori* covariance $\boldsymbol{P}_{k|k-1}$ can be obtained by noting that the extrapolated state estimate is given by,

$$\hat{\boldsymbol{x}}_{k|k-1} = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1}. \tag{A.35}$$

The extrapolated state estimate error is then,

$$\tilde{\boldsymbol{x}}_{k|k-1} = \hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{x}_k = \boldsymbol{\Phi}_k \hat{\boldsymbol{x}}_{k-1|k-1} - \boldsymbol{\Phi}_k \boldsymbol{x}_{k-1} + \boldsymbol{w}_k. \tag{A.36}$$

By taking the expectation on both sides, and noting that, $\hat{\boldsymbol{x}}_{k|k-1} - \boldsymbol{x}_k$ has covariance $\boldsymbol{P}_{k|k-1}$, $\hat{\boldsymbol{x}}_{k-1|k-1} - \boldsymbol{x}_{k-1}$ has covariance $\boldsymbol{P}_{k-1|k-1}$ and $\boldsymbol{w}_k$ has covariance $\boldsymbol{Q}_k$, the following is obtained,

$$\begin{aligned}
\boldsymbol{P}_{k|k-1} &= \mathbb{E}\left[\tilde{\boldsymbol{x}}_{k|k-1}\tilde{\boldsymbol{x}}_{k|k-1}^\top\right] \\
&= \boldsymbol{\Phi}_k \mathbb{E}\left[(\hat{\boldsymbol{x}}_{k-1|k-1} - \boldsymbol{x}_{k-1})(\hat{\boldsymbol{x}}_{k-1|k-1} - \boldsymbol{x}_{k-1})^\top\right] + \mathbb{E}\left[\boldsymbol{w}_k \boldsymbol{w}_k^\top\right] \\
&= \boldsymbol{\Phi}_k \boldsymbol{P}_{k-1|k-1}\boldsymbol{\Phi}_k^\top + \boldsymbol{Q}_k.
\end{aligned} \tag{A.37}$$

Which is the *a priori* covariance (equation A.2).

A.2 DERIVATION THE HYPOTHESIS PROBABILITY IN MULTIPLE HYPOTHESIS TRACKING

In this section the hypothesis probability used in multiple hypothesis target tracking is derived. Suppose θ_k is an association hypothesis about a measurement set $\boldsymbol{Z}_k = \{\boldsymbol{z}_k^1, \boldsymbol{z}_k^2, \boldsymbol{z}_k^3 \ldots\}$ and that a hypothesis for the past is available. By the fundamental theorem of target tracking [3], the *posterior* probability of θ_k is given by

$$P(\theta_k|\boldsymbol{Z}_{0:k}) \propto \beta_{FA}^{m_k^{FA}} \beta_{NT}^{m_k^{NT}} \left[\prod_{j\in\mathcal{T}_D} P_D^j p_{k|k-1}^j(\boldsymbol{z}_k^{\theta_k^{-1}(j)})\right]\left[\prod_{j\in\mathcal{T}_{ND}} (1 - P_D^j P_G^j)\right]. \tag{A.38}$$

Where, $\beta_{FA}^{m_k^F A}$ denotes the false alarm rate, $\beta_{NT}^{m_k^N T}$ denotes the new track rate, P_D^j denotes the detection probability of the jth target, P_G^j denotes the gate probability of the jth target and $p_{k|k-1}^j(\boldsymbol{z})$ denotes the predicted measurement density of jth target. The set $\mathcal{T}_D$ denotes the indices of targets which were assigned a measurement by θ_k^i and $\mathcal{T}_{ND}$ denotes the set of indices of targets which were not assigned a measurement by θ_k^i. Since there is a single hypothesis for the past, the term,

$$C_j = \left[\prod_{j\in\mathcal{T}_{ND}}(1-P_D^jP_G^j)\right]\left[\prod_{j\in\mathcal{T}_D}P_D^jP_G^j\right] = \prod_{j=1}^{n_T}(1-P_D^jP_G^j) \tag{A.39}$$

is constant. Substituting equation A.39 into equation A.38 results in

$$P(\theta_k|\boldsymbol{Z}_{0:k}) \propto \beta_{FA}^{m_k^{FA}}\beta_{NT}^{m_k^{NT}}\left[\prod_{j\in\mathcal{T}_D}\frac{P_D^jp_{k|k-1}^j(\boldsymbol{z}_k^{\theta_k^{-1}(j)})}{(1-P_D^jP_G^j)}\right]. \tag{A.40}$$

For multiple hypothesis tracking, let $\Theta_k^l = \{\theta_k, \Theta_{k-1}^i\}$, where Θ_k^l is the new set of hypotheses extended from the original previous hypothesis Θ_{k-1}^i. The *posterior* probabilities in the set Θ_k^l are given by

$$P(\Theta_k^l|\boldsymbol{z}_{0:k}) \propto p(\boldsymbol{z}_k|\Theta_k^l,\boldsymbol{z}_{0:k-1})P(\theta_k|\Theta_{k-1}^i,\boldsymbol{z}_{0:k-1})P(\Theta_{k-1}^i|\boldsymbol{z}_{0:k-1}). \tag{A.41}$$

This can be extended to

$$P(\Theta_k^l|\boldsymbol{z}_{0:k}) \propto \beta_{FA}^{m_k^{FA}}\beta_{NT}^{m_k^{NT}}\left[\prod_{j\in\mathcal{T}_D}P_D^jp_{k|k-1}^j(\boldsymbol{z}_k^{\theta_k^{-1}(j)})\right]\left[\prod_{j\in\mathcal{T}_{ND}}(1-P_D^jP_G^j)\right]P(\Theta_{k-1}^i|\boldsymbol{z}_{0:k-1}) \tag{A.42}$$

by using equation A.38. Substituting in equation A.39 and using equation A.40 results in

$$P(\Theta_k^l|\boldsymbol{z}_{0:k}) \propto \beta_{FA}^{m_k^{FA}}\beta_{NT}^{m_k^{NT}}\left[\prod_{j\in\mathcal{T}_D}\frac{P_D^jp_{k|k-1}^j(\boldsymbol{z}_k^{\theta_k^{-1}(j)})}{(1-P_D^jP_G^j)}\right]C_iP(\Theta_{k-1}^i|\boldsymbol{z}_{0:k-1}). \tag{A.43}$$

Which is the *posterior* distribution of the new set of hypotheses extended from the original set of hypotheses given by Θ_{k-1}^i.