UNIVERSITY OF CAPE TOWN

MASTERS THESIS

Aircraft state estimation using cameras and passive radar

Author:
Benjamin DE CHARMOY

Supervisor:
Dr. Fred NICOLLS

A thesis submitted in fulfillment of the requirements for the degree of Masters of Science

in the

Digital Image Processing Lab
Electrical Engineering

15th October 2018

Declaration of Authorship

I, Benjamin DE CHARMOY, declare that this thesis titled, 'Aircraft state estimation using cameras and passive radar' and the work presented in it are my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

_____

Date:

_____

'For every complex problem there is an answer that is clear, simple and wrong.'

H. L. Mencken

UNIVERSITY OF CAPE TOWN

Abstract

Engineering and the Built Environment

Electrical Engineering

Masters of Science

Aircraft state estimation using cameras and passive radar

by Benjamin DE CHARMOY

Multiple target tracking (MTT) is a fundamental task in many application domains. It is a difficult problem to solve in general, so applications make use of domain-specific and problem-specific knowledge to approach the problem by solving sub-tasks separately. This work puts forward a MTT framework (MTTF) which is based on the Bayesian recursive estimator (BRE). The MTTF extends a particle filter (PF) to handle the multiple targets and adds a probabilistic graphical model (PGM) data association stage to compute the mapping from detections to trackers.

The MTTF was applied to the problem of passively monitoring airspace. Two applications were built: a passive radar MTT module and a comprehensive visual object tracking (VOT) system. Both applications require a solution to the MTT problem, for which the MTTF was utilized.

The VOT system performed well on real data recorded at the University of Cape Town (UCT) as part of this investigation. The system was able to detect and track aircraft flying within the region of interest (ROI). The VOT system consisted of a single camera, an image processing module, the MTTF module and an evaluation module. The world coordinate frame target localization was within $\pm 3.2\,\mathrm{km}$ and these results are presented on Google Earth. The image plane target localization has an average reprojection error of ± 17.3 pixels. The VOT system achieved an average area under the curve value of 0.77 for all receiver operating characteristic curves. These performance figures are typical over the $\pm 1\,\mathrm{hr}$ of video recordings taken from the UCT site.

The passive radar application was tested on simulated data. The MTTF module was designed to connect to an existing passive radar system developed by *Peralex Electronics Pty Ltd*. The MTTF module estimated the number of targets in the scene and localized them within a 2D local world Cartesian coordinate system.

The investigations encompass numerous areas of research as well as practical aspects of software engineering and systems design.

Acknowledgements

I thank Dr. Fred Nicolls for supervising me and helping with many aspects of this project. Peralex Electronics, the University of Cape Town and the Wilhem Frank Trust are acknowledged for financial support that made it all possible.

Contents

List of Figures

Acronyms

ADS-B automatic dependent surveillance–broadcast.

AOA angle of arrival.

ARD amplitude-range-Doppler.

AUC area under the curve.

BN Bayesian network.

BRE Bayesian recursive estimator.

BREs Bayesian recursive estimators.

BRG blue-red-green.

CAF cross ambiguity function.

CFAR constant false alarm rate.

CPEM calibration and pose estimation module.

CPHD cardinalized probability hypothesis density.

CPM camera processing module.

CRFs conditional random fields.

DOA direction of arrival.

EKF extended Kalman filter.

ESS effective sample size.

FAR false alarm rate.

FG factor graph.

FGs factor graphs.

FGs Factor graphs.

FISST finite set statistics.

FOV field of view.

FPR false positive rate.

GIS geographic information system.

GMM Gaussian mixture model.

GMMs Gaussian mixture models.

GPS global positioning system.

GUI Graphical user interface.

GUI graphical user interface.

HMC Hamiltonian Monte Carlo.

HMM hidden Markov model.

HoG histogram of orientated gradients.

IID independent and identically distributed.

JPDAF joint probabilistic data association filter.

KF Kalman filter.

KFs Kalman filters.

MAP maximum a posteriori.

MC Monte Carlo.

MHT multiple hypothesis tracking.

MHT Multiple hypothesis tracking.

MLE maximum likelihood estimate.

MRFs Markov random fields.

MTT Multiple target tracking.

MTT multiple target tracking.

MTTF MTT framework.

PDAF probabilistic data association filter.

PDF probabilty density function.

PDFs probability density functions.

PF particle filter.

PFs particle filters.

PGM probabilistic graphical model.

PGMs probabilistic graphical models.

PGMs Probabilistic graphical models.

PHD probability hypothesis density.

PR passive radar.

PRS (*Peralex*) passive radar system.

RANSAC random sampling consensus.

RFS random finite sets.

RGNF recursive Gauss-Newton filter.

ROC receiver operating characteristic.

ROC Receiver operating characteristic.

ROI region of interest.

RR residual resampling.

SIS sequential importance sampling.

SMC sequential Monte Carlo.

SNR signal-to-noise ratio.

SVM support vector machine.

TPR true positive rate.

UCT University of Cape Town.

VIM video input module.

VOT visual object tracking.

VOT Visual object tracking.

WGS84 world geodetic system 1984.

Chapter 1

Introduction

This work sets out to achieve two main objectives: propose a multiple target tracking (MTT) framework and build two prototype applications that use the MTT framework to a solve real world problem. The problem around which this work is centered is that of passively tracking aircraft within a region of interest (ROI). Passive systems are ones which have no active component. They do not interrogate the scene like traditional radar systems do. Passive systems simply observe the scene and produce a form of scene understanding based on these observations. In both use cases presented here, the number and position of aircraft are estimated. This work is aimed at exploring the MTT problem in a general sense and building the practical applications which validate the proposed framework. The cases in which these systems are used originate from the military and the aerospace industry. In such applications the passive nature of the system makes it harder for adversaries to detect, giving vantage to passive systems. There is also an increasing need for monitoring airspace, and passive systems are often cheaper and do not make use of valuable bandwidth which, like physical airspace, is finite and limited.

Airspace over modern cities is a finite and valuable resource. There are many large companies such as *Amazon* and *Uber* aiming to exploit these spaces as a transport route for goods and potentially people. Coupled with the large uptake of high-end consumer drones like those produced by *Dji*, these airspaces are going to get more congested. As these airspace transport routes fill up, monitoring is going to become a paramount concern. As of today, companies such as *Dji*, *Altitude Angle* and *Airbus* are all developing airspace monitoring systems. It is estimated that this year alone over one million drones were sold. It is clear that air transport is a massive industry and monitoring it will be crucial to ensure safety and governance. Though this project is aimed at monitoring larger aircraft, the visual object tracking (VOT) application can easily be applied to smaller craft such as drones. As policies emerge it is undoubtedly going to be the case that all aircraft will be fitted with a transponder, as is the current case with manned aircraft using automatic dependent surveillance–broadcast (ADS-B). However, redundant systems will be sought and adversaries will not adhere to policy and the governance of the airspace. Thus there

will be a need for a cheap, reliable fall-back system as well as one that can detect aircraft intending to go unseen. An aspect of this project aims to provide a prototype airspace monitoring system in the hope that this can contribute to progress in the world as we adapt and embrace new technologies.

The multiple target tracking (MTT) framework is built under the theoretical guide of Bayesian reasoning. The multiple target tracking task is cast as a Bayesian inference problem, by which the posterior probability of a set of hypotheses about the target locations is updated as new evidence becomes available via one or many sensing modalities.

The first application is a comprehensive visual object tracking (VOT) system that uses a single static camera to detect and track aircraft that fly through the region of interest (ROI). The VOT system is made up of a number of modules. A high level system overview is shown below in figure 1.1.

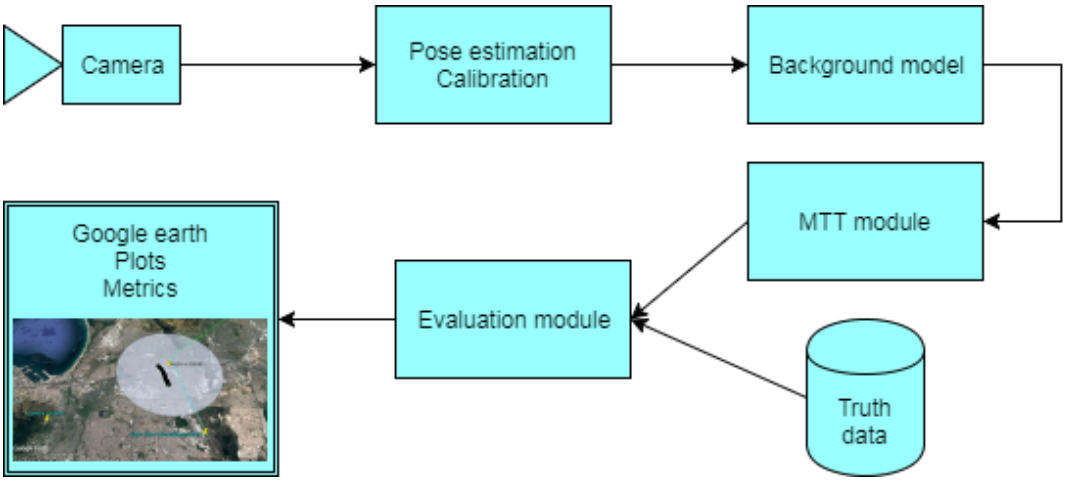


FIGURE 1.1: visual object tracking system overview.

The second application is a bistatic passive radar multiple target tracking module. The passive radar system that is used was developed by Peralex Electronics in South Africa. This work includes the building of the MTT module for this system and outputs aircraft positions on Google earth. The passive radar system overview is shown below in figure 1.2.

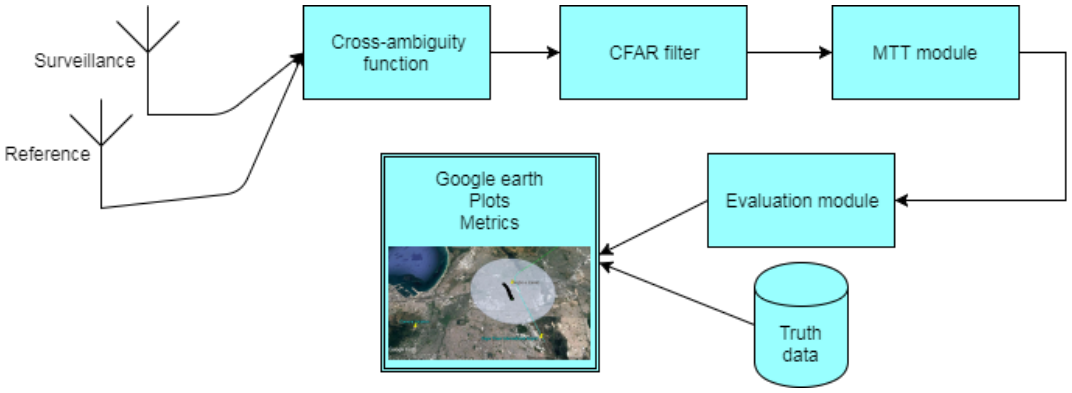


FIGURE 1.2: (*Peralex*) passive radar system system overview. The reference antenna receives the direct path signal from the illuminator. The surveillance antenna receives the time delayed and Doppler shifted signal echoed off the target and other objects in the scene. These two signals are processed by applying the cross ambiguity function and the result is passed through a constant false alarm rate filter to form detections. These detections are fed to the MTT module.

A complete description of the bistatic passive radar system is given in Tong's PhD thesis [56], as he did the pioneering work on this system. Section 1.2.2 gives an overview of bistatic passive radar.

Both applications use the same multiple target tracking (MTT) module, which is built as a Bayesian recursive estimator (BRE) that holds and updates the posterior distribution over the collective multi-target state vector as detection sets become available. To update the posterior distribution correctly, an association between the trackers and detections needs to be made. This association is the crux of many MTT algorithms as the computational complexity grows with the number of detections made. The MTT framework is the main contribution of this work. The MTT framework solves the MTT data association problem by casting it as a Bayesian inference problem by which the probability of a set of hypotheses about the target locations is updated as new evidence becomes available.

The posterior distribution over the collective state is held as a set of weighted particles that occupy the state space $\mathcal{S}$. A reduced state and measurement space called a label space is constructed and all targets and detections are mapped on to this reduced dimensional space. Then a graph is constructed that describes the joint distribution as a product of three potentials: unary, configuration and association. An inference algorithm is used to determine the maximum a posteriori (MAP) label assignment which provides the detection to target assignments. Based on this assignment labelling, the posterior is updated and the recursion is repeated for each measurement.

Those are the two main objectives of this work: building a multiple target tracking (MTT) framework and giving the theoretical derivations credibility by providing two applications that are realizations of the framework. It centers around the real

world problem of passive airspace monitoring and aircraft tracking and the two applications address and solve this problem. The remainder of this chapter provides further detail: defining the problem more strictly in section 1.1, highlighting the applications and how they were tested, having a brief look that the results in section 1.2 and lastly, providing a document road map in the concluding section 1.4.

1.1 Problem definition

This section describes the set of problems that need to be addressed to make inroads into a prototype solution to the airspace monitoring and aircraft tracking problem. This problem is simply stated as keeping a count and a state estimate (positions and velocities) of the number of aircraft in a region of interest (ROI) in the sky. In order to estimate the cardinality of targets and their respective states, two sub-problems need to be jointly solved. The target state estimation problem is described in section 1.1.1 and the cardinality and association problem is described in section 1.1.2. To give direction to the investigation, some aims are defined and these are stated in section 1.1.3.

1.1.1 State estimation problem

The state estimation problem is described as follows: consider a state vector S from the space $\mathcal{S}$ that obeys some stochastic evolution $S_{new} = f(S_{old}) + v$ where $f(.)$ is an arbitrary function that maps $\mathcal{S} :\longleftarrow \mathcal{S}$ and v is a random variable drawn from a distribution V. After observing Z the observation vector from the space $\mathcal{Z}$ is related to the state S by $Z = h(S) + w$ where $h(.)$ is a function that maps $\mathcal{Z} :\longleftarrow \mathcal{S}$ and w is a random variable drawn form a distribution W. The problem is to estimate the probabilty density function (PDF) describing S given all the information up to the current time. This problem is framed by the Bayesian interpretation as $p(S|Z) \propto p(Z|S)p(S)$. In the two applications dealt with in this work the state estimation problem is that of estimating the state of an aircraft given all the observations made by a staring camera or a passive radar system.

1.1.2 Multiple Target Tracking problem

The multiple target tracking problem is framed as follows: given a set responses from a sensor or a number of sensors, calculate the posterior probability of the system state, where the state is made up of concatenated single target states as well as a background model state. Both the value of the target state and the number of target states need to be estimated. This concatenated or stacked state vector is referred to as the collective state. Both applications require a solution to this problem as they aim to estimate the state of multiple targets as well as a background model. Thus, the state estimation problem is extended to estimate the collective state and the cardinality or number of single target states within the collective state. The multiple

target tracking (MTT) problem refers to the problem of jointly estimating the number of targets and their states or trajectories from noisy sensor measurements [61].

1.1.3 Aims

Multiple target tracking in the proposed context has many facets. To focus the effort a few aims are defined below:

1 *Develop a coherent and concise framework for solving the state estimation problem and extending it to the multiple target tracking problem.*

2 *Develop an image processing pipeline that implements the practical operations in a visual object tracking (VOT) system.*

3 *Develop a signal processing pipeline that implements the multiple target tracking framework.*

4 *Apply the multiple target tracking realization to the visual aircraft tracking problem.*

5 *Apply the multiple target tracking realization to the passive radar aircraft tracking problem.*

1.2 Tracker realizations and method overview

This section gives an overview of the two tracker realizations: the visual object tracking system and the passive radar system.

1.2.1 Visual tracker method overview

The visual tracking system built for this investigation has the following system components:

1 Camera calibration unit.

2 Camera pose estimation unit.

3 Image processing pipeline and tracker implementations.

4 File and TCP/IP video source input capabilities.

5 Output to Google earth via .kml files.

This system was deployed at the University of Cape Town and data sets were generated as part of this work. Pose estimation is achieved by solving the perspective-n-point problem and this is done using the random sampling consensus (RANSAC) method which is a standard approach. The implementation in the OpenCV library is used. The output of the first block is a calibrated image as well as the camera extrinsics that map world coordinates to image plane coordinates. The background model is

a Gaussian mixture model (GMM) and the output of this block is a statistical model of the pixel processes that make up the background. Based on the current frame and the background model, the detection map stage applies a threshold and passes the set of detections to the MTT module. The MTT module updates the collective multi-target state vector based on the set of detections. The majority of the work presented here is held in the MTT module and it outputs an estimate of target locations. These locations are plotted on Google Earth. See chapter 3 for details on the data processing pipeline, the site and physical setup as well as a description of the coordinate frames and transforms used to anchor the system as a real world application.

1.2.2 Passive radar tracker method overview

Passive radar is a class of radar system that makes use of non-cooperative sources of illumination, such as commercial broadcast and communication signals, in order to detect and track objects/targets [36]. Figure 1.3 illustrates a typical bistatic passive radar setup. An illuminator, typically an FM radio broadcast tower, emits a signal. The passive radar system has two antennae, a reference antenna and a surveillance antenna.

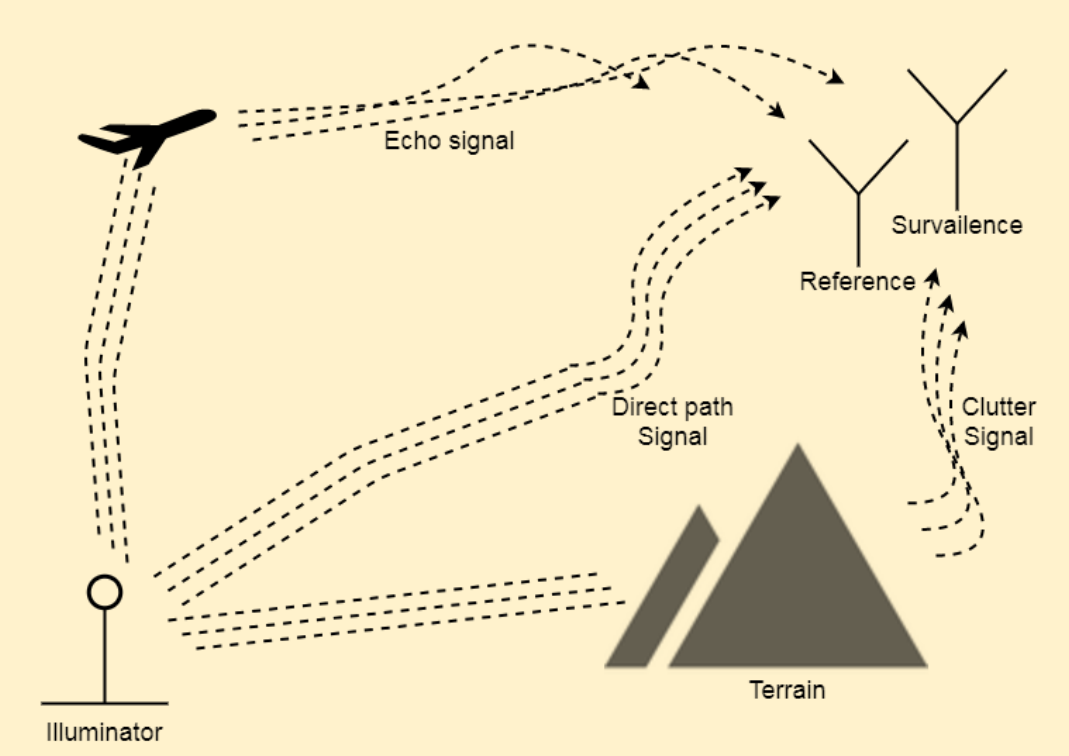


FIGURE 1.3: Bistatic passive radar setup.

The reference antenna has line of sight to the illuminator and the surveillance antenna surveys a ROI. The reference signal is the direct path signal from the illuminator, and the surveillance signal is the echo returned from the surface of a potential target (see

figure 1.2). Reflected signals from the surface of a target within the ROI are cross-correlated with multiple synthetically Doppler-shifted versions of the reference signal. These synthetic Doppler shifts range over predetermined intervals based on the expected targets of interest velocities. The process of shifting and cross-correlating these two signals is specified by the cross ambiguity function (CAF). The CAF of these two signals is computed and produces an amplitude-range-Doppler (ARD) surface over the specified bistatic range and Doppler bins. A constant false alarm rate (CFAR) filter is applied to the ARD surface and a set of detections is produced. The detections are given to the MTT module which updates the collective multi-target state vector and outputs position estimates which are plotted on Google Earth. This dissertation is concerned with the MTT module and it was only this module that was built as part of this work. Jackson's work [25] details the geometric setup and its relating consequences. The system utilised in this work comes out of Tong's PhD thesis [56] and he provides further details regarding the processing pipeline.

1.3 Overview of contributions and results

The main contribution of this work from a conceptual and theoretical point of view is the development of a multiple target tracking (MTT) framework following the Bayesian approach. This framework handles the birth and death of trackers and solves the data association problem with a probabilistic graphical model (PGM) that uses a reduced dimension label space on which to calculate the maximum a posteriori (MAP) assignments. This is not a novel approach and the framework under development is compared to those in the literature in section 2.4.The framework holds a background model as well as a collective state for all trackers. The framework is general purpose in that it is sensor and system agnostic. Provided that a prior of the form $p(S_k|S_{k-1})$ and a likelihood function $\mathcal{L}(S_k, Z_k) \propto p(Z_k|S_k)$ can be specified, the MTT framework can be applied. These contributions are detailed in chapters 5 and 6.

The main practical contribution is the system architecture and implementation of a passive airspace monitoring system. This has a complete set of modules for calibration, world-aligned pose estimation, streaming live video over TCP/IP, recording and playback. Although it is a prototype, the system engineering provides a significant practical contribution. The VOT system as it stands is a useful airspace monitoring tool. In its current form it is used as an aid and a human operator would still be needed to verify the targets. This operational drawback is primarily due to the single camera configuration; however, this is inexpensive and still provides utility.

The practical details are summarized in chapter 3. The code is hosted on and accompanying software documentation can be found in [9].

The system was able to estimate the number of targets in a scene (in both the visual

and passive radar cases) and track these targets. In scenarios with high false alarm rates the MTT algorithm did birth and destroy an excessive number of trackers. This could be perceived as a failure in certain applications, especially when actions that incur a cost would be taken based on these false assertions. However, in most security and monitoring systems there is still a person in the loop and so this application would simply be an aid.

Anecdotal evidence of the system outperforming a fatigued human operator follows. During the painstaking task of labelling an hour of video recordings, the author missed a small aircraft. Upon running the recording through the VOT system, it picked up and tracked the aircraft. After the system made the initial detection and drew the operator's attention to it, it was easy for the operator to determine it was in fact an aircraft and of interest.

For the case of a five targets under observation of the passive radar (PR) system, the following simulation results were obtained. The number of targets present in a simulated scenario is given by N_t. The expected number of false alarms per scan is given by u and governed by a Poission distribution. A binomial distribution with parameter d dictates the probability of detection. A full explanation of the simulation test bed and results is covered in chapter 7.

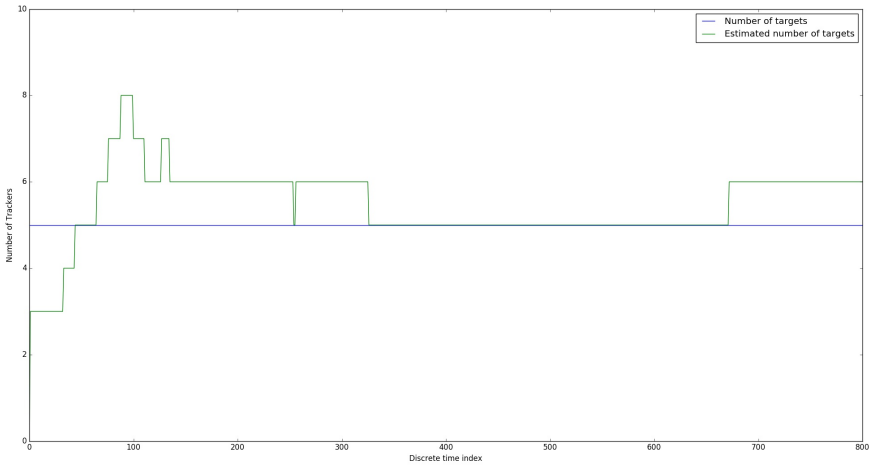


FIGURE 1.4: Cardinality estimation with simulation parameters $N_t = 5$, $u = 10$ and $d = 0.8$. The number of targets is N_t, expected number of false alarms per scan is u and probability of detection is d.

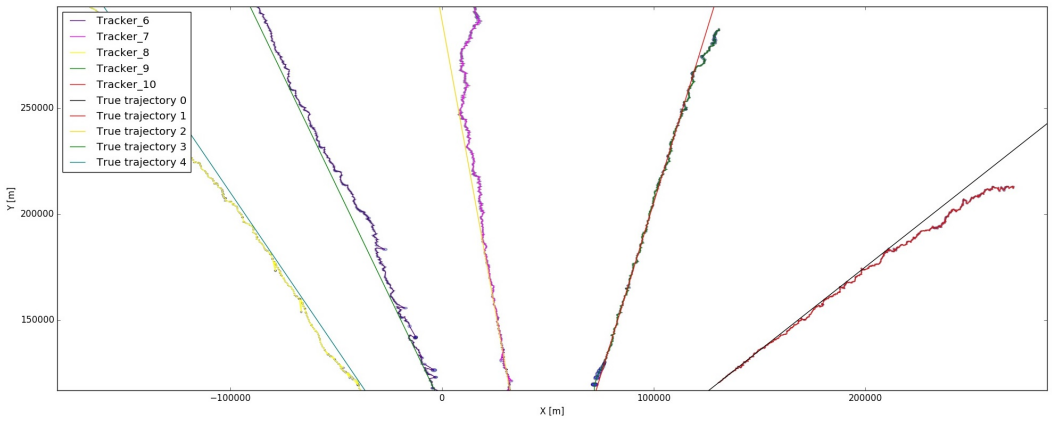


FIGURE 1.5: Passive radar simulated scene with simulation parameters $N_t = 5$, $u = 10$ and $d = 0.8$.

Figure 1.4 shows that the number of estimated targets remains reasonable. Despite there being ten false detections per scan and five true targets the MTTF curbs the number of proposed targets to around five. Figure 1.5 shows the five simulated trajectories and the five longest persisting trackers. On this simple test case the MTTF was able to track the multiple targets and birth and destroy short lived trackers which were needed to explain away the sets of false alarms.

The visual object tracking (VOT) system was able to autonomously detect and track aircraft in the region of interest (ROI). An important feature of any practical real world airspace monitoring system would be to be able to run free of human guidance and simply make alerts when targets are present. Approximate trajectories for each target over a number of trials were obtained and these closely resemble the true trajectories. Below is a typical result from the VOT system.

FIGURE 1.6: A typical frame from 08162017_MVI_BA6419 sequence.

Figure 1.6 shows a frame from a video as input to the VOT system. This frame has an aircraft in it which is en route to land at Cape Town international airport. The VOT system detected and tracked this aircraft as it flew through the ROI. Figure 1.7 shows the VOT system's target state estimate plotted against the typical flight paths of this aircraft.

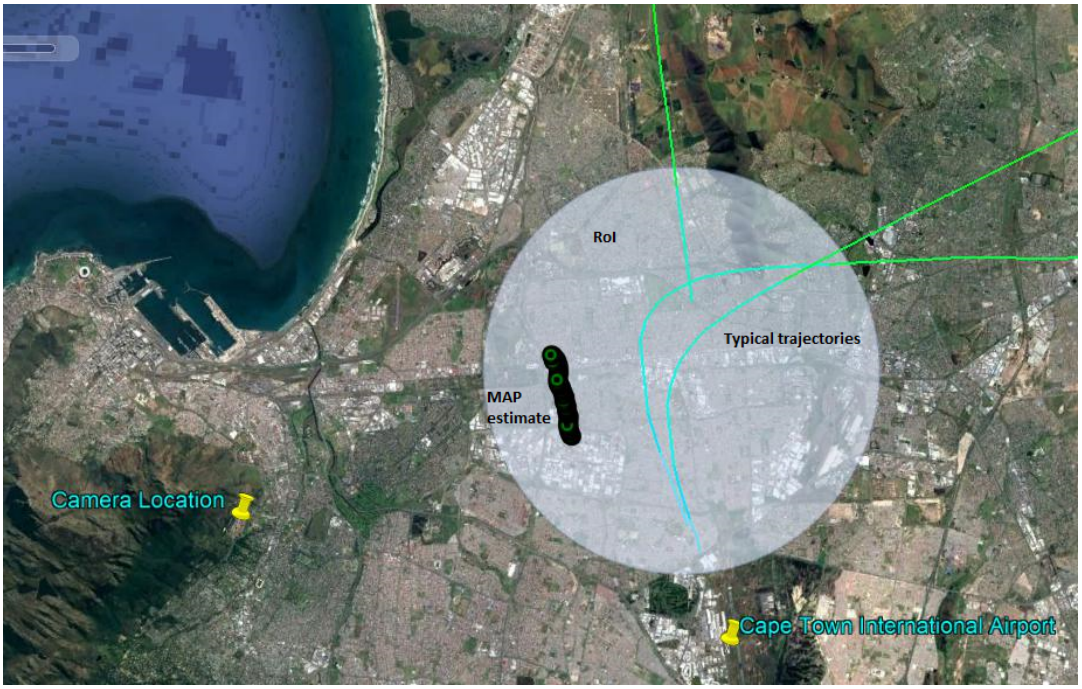


FIGURE 1.7: Tracker plotted on Google Earth. This is the system's output and the human operator can look at this track together with the video feed to determine if further actions are required.

The VOT system does not accurately estimate the aircraft's position as it is using only a single camera and there is information loss due to the 3D to 2D projection. This is inherent in the system owing to its single camera design but with a more constrained prior, more accurate pose estimation and a stronger motion model it would be possible to improve these results. The VOT system is able to estimate target airspeed that was within the typical range for aircraft within this ROI.

Further comments regarding the passive radar (PR) system and the visual object tracking (VOT) system's performance, as well as a discussion on future work, can be found in chapter 9.

1.4 Document roadmap

This section describes the structure of the document and outlines the development of the investigation.

Chapter 2 provides a review of the literature and techniques that were drawn to produce this work. It covers aspects of probability theory. It reviews and comments on common approaches to system state tracking and how they have been applied to both visual object tracking and passive radar tracking. Specific attention is paid to the Bayesian recursive estimator and its approximate implementation, namely the particle filter. Predominant multiple target tracking algorithms are reviewed and related to the MTTF. A short introduction of probabilistic graphical models with an emphasis on the notation of factor graphs is presented. The chapter concludes with a review of background foreground segmentation and how it is used in VOT systems.

Chapter 3 details the experimental setup and system architecture of the VOT system. It describes the site and gives a high-level view of the system and graphical user interface. The technical aspects of the experiment, such as the camera calibration and pose estimation and resulting measurement model, are described at the end of this chapter.

Chapter 4 describes the image processing module of the visual object tracking system. The background modelling and detection process is covered in detail and an illustrative example is presented. Alternative detection methods that were attempted during this investigation are summarised at the end of this chapter.

Chapter 5 outlines the development of the MTT framework. It describes the Bayesian recursive estimator and presents all the assumptions and design choices that were made in pursuing a realizable implementation. The particle filter is presented as it applies to multiple target tracking. The approach taken to birthing and terminating trackers as well as handling multiple trackers is specified in detail. An illustrative two-aircraft example is defined and used to clarify the internal workings of the MTT framework.

Chapter 6 presents the data association stage. The graphical structure of the association problem is presented and the two-aircraft scenario is extended to explain the association mechanism used.

Chapter 7 shows how the MTT framework was applied to the task of bistatic passive radar aircraft tracking. A simulation environment was created and two scenarios are reported on. This chapter gives an overview of applying the MTTF to an actual problem and validates its correctness on two simple cases.

Chapter 8 describes the visual object tracking experiment and criteria against which its results were evaluated. The results are presented for the UCT site dataset which was curated as part of this investigation. These findings are discussed, citing conditions under which the system performs well and what pathologies it suffers from.

Chapter 9 highlights the achievements of the study and presents insights and recommendations regarding future work in the area of passive airspace monitoring.

Chapter 2

Literature review

The literature pertaining to state estimation, target tracking, visual target tracking, passive radar tracking and data association is extensive. This chapter offers a summary of the papers and techniques that were drawn from to produce this work. Section 2.1 gives reference to probability theory and highlights aspects that are utilized in the remainder of this work. Section 2.2 gives an overview of the categories of trackers in both the visual and passive radar application domains. Section 2.3 details some specific commonly used and often referred to Bayesian tracking algorithms. It details the particle filter (PF) as it is the chosen Bayesian recursive estimator (BRE) realization and highlights the numerous variations and implementation techniques. In section 2.4 three predominant multiple target tracking algorithms are reviewed and compared to the MTT framework (MTTF). Section 2.5 provides an overview of probabilistic graphical models and how they have been applied to the data association problem within the context of the MTT framework. Section 2.6 gives a brief summary of the approaches that have been used in background modeling and image segmentation. The visual object tracking (VOT) application makes use of a background model as is common in VOT systems. The last section highlights the recent work done in this domain.

2.1 Probability theory

This section simply states a few probability theory results, terms and definitions to give context and common ground for the remainder of the work. Gelman [19] provides a good reference and clearly states the Bayesian view. The main theorem around which this work pivots is Bayes' theorem:

$$p(s|z) = \frac{p(z|s)p(s)}{p(z)}. \tag{2.1}$$

This simple equation allows for rewriting the conditional probability of s given z, the system state given an observation, as a product of the probability of an observation given state, a prior on state and the probability of evidence. This enables making claims about the state (often called hidden or latent variables) even though it was

never observed directly. This result forms the basis of all Bayesian recursive estimators. The left-hand-side of equation 2.1 is called the posterior distribution. The first term of the numerator is called the likelihood and the second term is the prior. The denominator is referred to as the probability of evidence.

The joint probability distribution $p(s, z)$ is a function over the joint event space of s and z. All joint distributions can be factored according to the general product rule shown here with three random variables for clarity:

$$p(x_0, x_1, x_2) = p(x_2|x_1, x_0)p(x_1|x_0)p(x_0). \tag{2.2}$$

The general product rule applied to the joint of two events, a state s and an observation z, is $p(s, z) = p(z|s)p(s)$. Once an observation has been observed the conditional probability $p(s|z)$ can be computed as $p(s|z) = \frac{p(s,z)}{p(z)}$. Thus Bayes' rule is a simple result based on the above two properties, the general product rule and conditional probability, as shown below:

$$p(s, z) = p(z|s)p(s) \tag{2.3}$$

$$p(s|z) = \frac{p(s, z)}{p(z)} \tag{2.4}$$

$$p(s|z) = \frac{p(z|s)p(s)}{p(z).} \tag{2.5}$$

Marginalization is the procedure of summing out a subset of variables over a probability distribution. Thus, given the joint probability $p(x, y)$ where $x \in \mathcal{X}$ and $y \in \mathcal{Y}$, the marginal probability $p(x)$ is calculated as follows:

$$p(x) = \int_{\mathcal{Y}} p(x, y)dy \tag{2.6}$$

$$= \int_{\mathcal{Y}} p(x|y)p(y)dy. \tag{2.7}$$

In the discrete case, which is how it is used in this work, the integral becomes a summation as shown below:

$$p(x) = \Sigma_{y\in\mathcal{Y}}\, p(x, y) \tag{2.8}$$

$$= \Sigma_{y\in\mathcal{Y}}\, p(x|y)p(y). \tag{2.9}$$

Since it involves summing over subsets of variables, marginalization is often referred to as the sum rule. This is a fundamental procedure and when distributions become complicated and of high dimension it is often a computation that is desired but not tractable. This is the primary reason for shifting to a low-dimensional label space for the data association stage. See section 2.5 and chapter 6 for further details.

The Markov property is an assumption on the manner in which the state vector

and the governing dynamics of a system are defined. The Markov property asserts that the system state at time k, namely s_k, depends only on the the system state at time $k-1$ and not on any previous states. This may seem to be restrictive but the state can always be extended to ensure this condition is met. The assumption holds true when the state is complete. A complete state holds all the necessary system information to compute the next system state and does not require any knowledge of state evolution up until that time. This assumption greatly simplifies multivariate state estimation problems and makes these models suitable to simple probabilistic graphical models as shown in section 2.5.

Definition 1 (Markov property) *The conditional probability distribution of future states of the process depends only upon the present state and not on the sequence of events that preceded it.*

The state of a system as well as the resulting measurements are governed by probabilistic laws. The state transition is governed by $p(s_t|s_{0:t-1}, z_{1:t-1})$ as this is the probability distribution from which s_t is generated. Thurn [55] frames this notion very clearly and the following remarks simply echo them. Under the Markov assumption $p(s_t|s_{0:t-1}, z_{0:t-1}) = p(s_t|s_{t-1})$ which shows that the history of measurements $z_{0:t-1}$ can be discarded without losing information provided the previous state is known. This leads to the next useful fact, that of conditional independence. This means that certain variables are independent of each other if a third group of variables are known. Conditional independence is used extensively in chapter 6 where the data association algorithm is described. Conditional independence as defined by Koller [32] is stated below.

Definition 2 (Conditional independence) *Let X, Y and Z be sets of random variables. X is conditionally independent of Y given Z in a distribution p if*
$p(X{=}x, Y{=}y \mid Z{=}z) = p(X{=}x \mid Z{=}z)p(Y{=}y \mid Z{=}z)$
for all values $x \in X, y \in Y$ and $z \in Z$

The data association stage uses probabilistic graphical models (PGMs) and these models depict conditional independences by their structure. Using the general chain rule the joint probability distribution can be factored into a product of factors known as potentials. This factorization adheres to the graph structure and allows generic inference techniques to be applied. Literature and supporting results for using graphical models as an interpretation of joint probability distributions are presented and discussed in sections 2.5 and 6.4. The following works [31, 32, 26] are good introductions to the fundamentals of graphical models and their usefulness.

2.2 Review summary of trackers

As described in the problem statement in section 1.1.1, the state estimation task aims to track the evolution of a system's state over time. There are two broad categories

into which approaches fall. These are state space and tracking by assignment methods. The literature pertaining to these approaches is summarized and compared in sections 2.2.1 and 2.2.2. Section 2.2.3 shows the state of current visual object tracking (VOT) research and highlights where this work fits in as a visual object tracking framework. Section 2.2.4 looks at the passive radar tracking literature and in particular the work done by the University of Cape Town radar group and the *Council for Scientific and Industrial Research*.

2.2.1 State space methods

The state space approach holds at its the core a fully specified state vector (complete in the Markov process sense; see section 2.1) and a set of governing dynamics. By defining the system as a Markov process, it can represented by a Bayesian network (BN), an example of which is shown in figure 2.1.

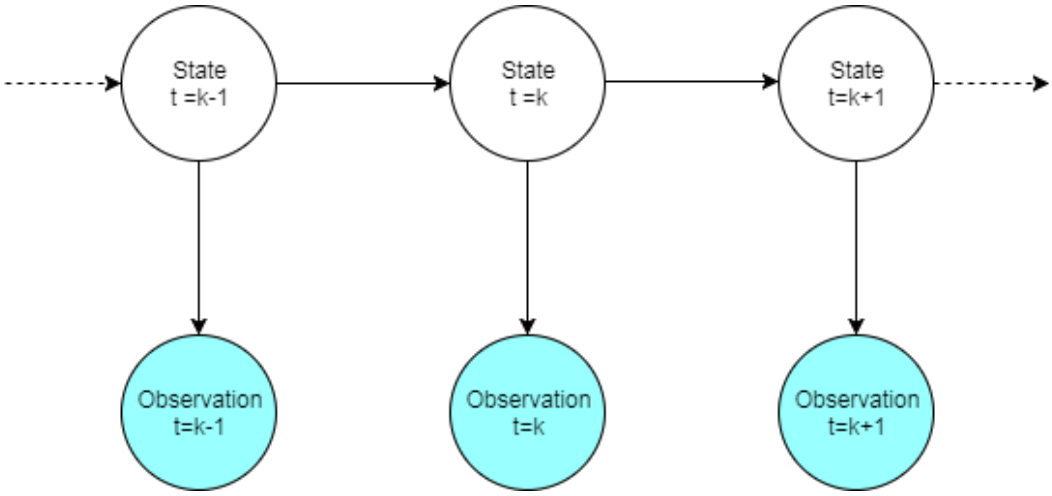


FIGURE 2.1: Bayesian network (BN) showing a system that obeys the Markov assumption.

The example system depicted in figure 2.1 leads to the following state evolution process: $p(State_k) = p(State_k|State_{k-1})$.

The state evolution process allows for embedding the dynamics of the target into the tracking algorithm. This has an advantage over pure assignment trackers (see section 2.2.2) as it provides a principled mechanism through which prior knowledge can be included in the tracker. As stated earlier, the framework used as a guiding principle throughout this investigation is Bayesian. The Bayesian principle follows directly from Bayes' rule (equation 2.1) and simply states that inferences about the latent variables of a system can be computed given the history of observed variables. In order to apply Bayes' rule the latent variables and the observed variables are defined as random variables. Random variables have an associated distribution from which they are drawn. State space models hold the distributions over the latent and observed variables. Bayes' rule guides the construction of the state space model and the defining functions that link observations and latent states.

The characteristics of a Bayesian tracker [50] are:

1 *Prior Distribution.* There must be a prior distribution on the state of the targets. If the targets are moving, the prior distribution has to include a probabilistic description of the motion characteristics of the target. Usually the prior is given in terms of a stochastic process for the motion of the target.

2 *Likelihood Functions.* The information in sensor measurements, observations, or contacts must be characterized by likelihood functions.

3 *Posterior Distribution.* The basic output of a Bayesian tracker is a posterior probability distribution on the (joint) state of the target(s). The posterior at time t is computed by combining the motion updated prior at time t with the likelihood function for the observation(s) received at time t.

State space methods give a clear and unambiguous framework for incorporating all three characteristics of a Bayesian tracker. The multiple target tracking (MTT) framework developed and implemented in this work (MTTF) has all these attributes and uses the state space formulation of the tracking problem. See chapter 5 for a complete explanation.

2.2.2 Tracking-by-assignment

The reason for including tracking-by-assignment into this brief literature review was because this field has performed some very informative work on the data association problem. In essence this approach is centered on the data association problem and the author derived many insights into this problem when reviewing and implementing these methods. The tracking-by-assignment problem is the task of finding assignments between sets of object candidate observations produced at regular intervals [30]. The goal is to build up an object track by linking observations over time.

Tracking-by-assignment is a popular approach when there is no need for querying the system for values of hidden or latent states. Thus, if the task were simply to track in the image plane, as is the case for many applications in biology, then the tracking-by-assignment paradigm is well suited. These applications often require a count of the objects of interest and the duration for which they were observable [30]. It is often practical to use a matching approach and create a track by joining the associated detections. This approach lacks a temporal structure so if the underlying process has a strong temporal characteristic, such as an aircraft as opposed to cells floating in a turbulent fluid, then the dynamics of the system cannot be exploited.

The tracking-by-assignment paradigm has many advantages as it is often simple and easy to implement, so real-time applications on mobile platforms can benefit from such approaches. However, the performance hinges on the ability of the detector. In the case of image-based tracking, if there is a high false alarm rate (FAR) then

these approaches fail and systematic coherent approaches to make improvements are difficult. The result is to use ad-hoc, very domain-specific solutions to improve performance.

Tracking-by-assignment solves the difficult data association problem. The data association problem is as follows. Multiple objects are present in a scene. These objects make up the set $\{O_i\}_{i=0}^{i=N_o}$. A sensor monitoring the scene can detect a subset of the objects and produces reports of these detections as well as false alarms. The detection set is $\{Z_j\}_{j=0}^{i=N_d}$. The problem of data association is to determine which observation from the set Z corresponds to which object in the set O. Many papers pose the data association problem as a probabilistic graphical model and employ graph-theoretic results to compute the maximum a posteriori assignments between objects and detections [12, 11, 64]. The MTT framework developed in this work makes use of the PGM-based data association approach described in the tracking-by-assignment literature. Section 2.5 details the probabilistic graphical models, and the data association used in this work is put forward in chapter 6.

2.2.3 Visual trackers

Visual object tracking (VOT) is a vast field of research. The application areas are varied and so the techniques that are well-suited to visual object tracking are difficult to specify in general. Various applications are able to sidestep difficulties based on some domain-specific assumptions.

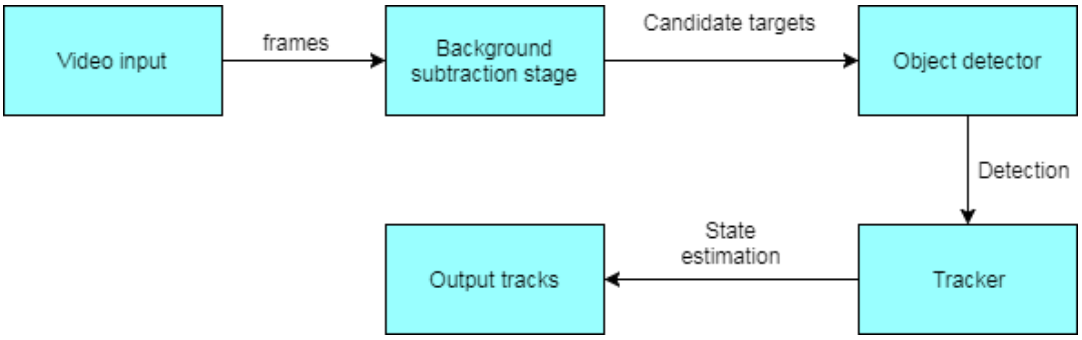


FIGURE 2.2: Typical VOT system.

A large portion of the machine vision research community approaches the tracking problem by building a good object detector and then feeding the output to a generic tracking algorithm such as a Kalman filter (KF). This leads to building up an image processing pipeline that has a classifier of some description which reports detections to the tracking stage. Figure 2.2 shows a typical visual object tracking system processing pipeline. Trucco provides a good survey and introduction to the video tracking landscape [57]. It describes the tracking task as a motion estimation problem and a matching problem. The first problem is predicting where the target will be in the next frame. The second problem is locating the target in the current frame. This

division is very useful as a means of expressing the tracking task as two smaller sub-tasks. However, this work endeavors to blur this line as the act of locating a target in the current frame should be informed by the latest prediction. As will become clear in later chapters, the framework developed here models the complete system and updates both stages in a coherent way. It is commonplace (in the computer vision community) that trackers are initialized by hand and then an attempt is made to track for as long as possible [3, 27]. The best performing trackers in this paradigm are adaptive discriminative trackers. Kalal (in [27]) details this approach, however it relies on two aspects that the current work does not have. These are that the tracker is manually initialized by a bounding box annotation on the first frame, and that the target occupies a significant number of pixels in the frame which allows strong discriminative features to be extracted. These assumptions are very strong, and though they lead to the development of new techniques the direct applicability to real world problems is lacking. A tracker that requires a human to initialize has very limited use cases. It should be noted that the assumption that the target will occupy a large number of pixels is also very limiting. In the aircraft tracking problem addressed in this work the target typically occupies very few pixels and so popular discriminatory approaches fail. Section 4.2 describes how the prevailing VOT object detector approaches fared on the aircraft tracking task posed in this work.

2.2.4 Passive radar trackers

The work presented here does not tackle the passive radar problem. It merely builds on a MTT module to an existing system, namely the (*Peralex*) passive radar system (PRS). Aspects of this work that pertain to passive radar is just an example realization of the multiple target tracking framework applied to the PRS. This is a bistatic radar, which means it consists of a single receiver and a single illuminator. This system is combined with an angle of arrival antenna array. Thus the measurements consist of bistatic range, bistatic range-rate and the angle of arrival (AOA). Passive radar measurements are formed by the process described as follows. The direct path or reference signal is cross correlated with the surveillance path over multiple time and frequency shifts. The resulting function is referred to as the cross ambiguity function (CAF). The cross ambiguity function is visualized for an interval of bistatic ranges and bistatic Doppler shifts as an amplitude-range-Doppler (ARD) map. Detections are determined by running a constant false alarm rate (CFAR) filter over this amplitude-range-Doppler surface and detecting peaks. These detections are passed to the MTT module developed in this work. For a complete description of the system and signal processing stages refer to Tong's PhD thesis [56] where the pioneering work is developed. Figure 2.3 shows a graphical description of the system.

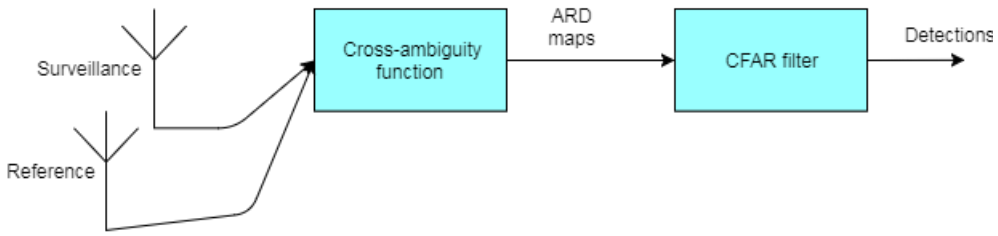


FIGURE 2.3: A schematic of the passive radar system for tracking aircraft.

Maasdorp [36] developed a multi-static Doppler-only tracker using the recursive Gauss-Newton filter (RGNF). The Doppler-only tracker is notable for a few reasons. It uses the same signal processing stage as this work, although it was in a multi-static configuration. The Massdorp tracker bypassed the data association stage by making use of automatic dependent surveillance–broadcast (ADS-B) data. Even though results were based on field trials the use of truth data in the form of ADS-B reports removed the need for data association which is a key component in all multiple target tracking tasks. Chapter 7 describes the MTT framework as it was applied to the (*Peralex*) passive radar system.

2.3 Bayesian recursive estimators

This section describes the Bayesian recursive estimator (BRE). Section 2.3.1 details the constraints under which computing the optimal BRE solution is tractable. Section 2.3.2 covers a practical approximation to the BRE, namely the particle filter (PF), when the constraints required for the optimal cases are violated.

Bayesian recursive estimators are a class of estimators that make full use of the Bayesian framework. The Bayesian framework uses probability theory to represent and manipulate all forms of uncertainty in the model [20]. The general form of Bayesian recursive estimators is as follows:

$$bel(s_k) = \eta p(z_k|s_k) \int p(s_k|s_{k-1}) bel(s_{k-1}) ds_{k-1}. \tag{2.10}$$

This states that the current belief of a system's state $bel(s_k)$ depends on the previous belief about the system state, the state evolution model and the likelihood of the observations. Since the state dynamics and observation model are a fully-specified probability function, the updating of the system state belief is consistent with probability theory and principled. The inverse of probability of evidence is η and is needed as a normalizing constant to interpret the $bel(s_k)$ as a valid probability. The Bayesian recursive estimator is a consistent framework in which to work. However, computing all the necessary terms when the probability density functions (PDFs) are no longer conjugate distributions and the transforms governing the state evolution and observation

models are no longer linear is an intractable problem. The set of equations below portray the state estimation problem in a general form that the following sections will reference:

$$S_k = f_k(S_{k-1}, v_{k-1}) \quad \text{and} \tag{2.11}$$

$$Z_k = h_k(S_k, w_k). \tag{2.12}$$

Equations 2.11 and 2.12 show the evolution of state and the measurement model respectively. The current state S_k is mapped to the next state S_{k-1} by $f_k(.,.)$ where v_{k-1} is drawn from an independent and identically distributed (IID) process noise model V. A measurement prediction Z_k is produced by the measurement model $h_k(.,.)$. The parameter w_k is drawn from an independent and identically distributed measurement noise model W. The output of a Bayesian estimator is the posterior probability $p(S_k|Z_{1:k})$. This allows for not only the first moment estimate (mean) but also an associated uncertainty (covariance) to be calculated. From the posterior a number of other measures could be deemed valuable but the BRE computes the optimal posterior given all prior knowledge and all observed data. The motivation for ensuring state estimation is done in a calibrated manner is because it is often part of a greater system in which decisions are required.

This posterior is computed recursively by a prediction and update step. The prediction step uses the state transition model given by equation 2.11 and the known process of V. It is computed using the Chapman-Kolmogorov equation:

$$p(S_k|Z_{1:k-1}) = \int p(S_k|S_{k-1})p(S_{k-1}|Z_{1:k-1})dS_{k-1}. \tag{2.13}$$

The update step uses the latest observation Z_k and updates the posterior based on this latest piece of information. This is done by using the observation model given by equation 2.12 and the known process W. Using Bayes' rule the posterior is updated as follows:

$$p(S_k|Z_{1:k}) = \frac{p(Z_k|S_k)p(S_k|Z_{1:k-1})}{p(Z_k|Z_{1:k-1})}. \tag{2.14}$$

All Bayesian recursive estimators (BREs) adhere to this prediction, (measurement) update, prediction, (measurement) update recursion.

2.3.1 Optimal Bayesian recursive estimators (BREs)

This section looks at filters that compute the true Bayesian recursive estimator. The Kalman filter (KF) and grid-based filter are briefly described and the assumptions under which optimality is guaranteed are clearly stated.

Kalman Filter

The Kalman filter (KF) makes four assumptions:

1 The mapping from measurement space $\mathcal{Z}$ to state space $\mathcal{S}$ is linear. That is, $h_k(S_k, w_k)$ from equation 2.12 is linear.

2 The evolution of state is both Markovian and linear. That is, $f_k(S_k, v_k)$ is linear and the state is complete in the sense of the Markov assumption in definition 1.

3 The uncertainty in the measurement process is Gaussian and independent and identically distributed.

4 The uncertainty in the state dynamics is Gaussian and independent and identically distributed.

These assumptions allow a rewrite of equation 2.11 as

$$S_k = F_k S_{k-1} + v_{k-1} \tag{2.15}$$

and equation 2.12 as

$$Z_k = H_k S_k + w_k. \tag{2.16}$$

Using the common Kalman filter notation the covariance of v_{k-1} is Q_{k-1} and the covariance of w_k is R_{k-1}. The matrices F_k and H_k define the linear functions governing state transitions and the observation model respectively. Since the probability distributions are Gaussian they are parametrized by a mean vector $m_{k|k-1}$ and a covariance matrix $P_{k|k-1}$. Taking the linear state transition model (equation 2.15) and the linear observation model (equation 2.16) and using the Bayesian recursive estimator prediction equation 2.13 and the Bayesian recursive estimator update equation 2.14, the Kalman filter (KF) is derived to be the following recursive relationship [2]:

$$(prior)\, p(S_{k-1}|Z_{1:k-1}) = \mathcal{N}(S_{k-1}, m_{k-1|k-1}, P_{k-1|k-1}) \tag{2.17}$$

$$(prediction)\, p(S_k|Z_{1:k-1}) = \mathcal{N}(S_k, m_{k|k-1}, P_{k|k-1}) \tag{2.18}$$

$$(update)\, p(S_k|Z_{1:k}) = \mathcal{N}(S_k, m_{k|k}, P_{k|k}). \tag{2.19}$$

This is the Bayesian recursive estimator under the linear Gaussian constraints. Although it achieves optimality in a Bayesian sense the tracking task often breaks these constraints, as happens in both the visual tracking and passive radar tracking applications dealt with later in this work.

All probability distributions can be illustrated as a probabilistic graphical model (PGM) and the posterior of a Kalman filter is no different. Figure 2.4 depicts a kalman filter as a probabilistic graphical model. This directed graphical model shows the conditional independences, which variables are observed, which are latent and that the Markov assumption holds true.

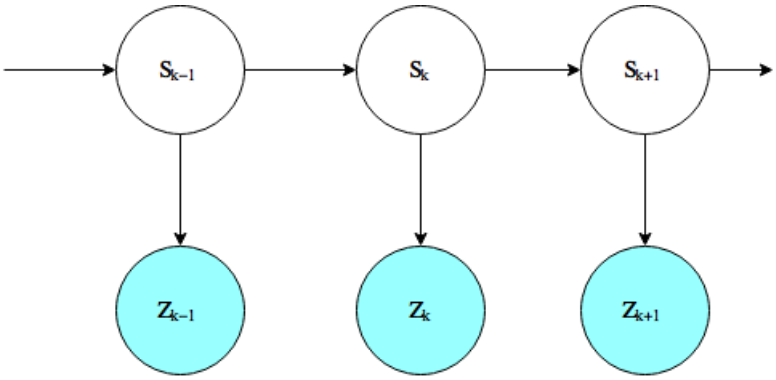


FIGURE 2.4: A Kalman filter (KF) depicted as a hidden Markov model (HMM).

Further details regarding probabilistic graphical models are given in section 2.5. The second optimal Bayesian recursive estimator (BRE) realization is grid-based estimation. This class of filters are described below.

Grid-based estimation

Grid-based filters are able to perform an exact Bayesian recursive estimator (BRE) recursion provided that the state space is discrete and consists of a finite set of states S. Given these two assumptions, the posterior PDF can be held by a complete collection of discrete states $S = [S^i]_{i=1}^{NumStates}$ and an associated weight ω^i:

$$p(S_{k-1}|Z_{1:k-1}) = \Sigma_{i=1}^{NumStates} \omega_{k-1|k-1}^{i} \delta(S_{k-1} - S_{k-1}^{i}). \tag{2.20}$$

The Bayesian recursive estimator prediction and update steps can be computed directly by iterating over all states provided that the transition model $p(S_k^i|S_{k-1}^j)$ and the likelihood function based on the observation model $p(Z_k|S_k^i)$ are both known. The restrictions are very strong but provide a nice way to see the problem as a whole.

2.3.2 Suboptimal Bayesian recursive estimators

The Bayesian recursive estimator solves the estimation problem, but computing it is often not possible. This leads to approximating the BRE, and there are two categories of approaches. The first approach is to form an approximate system model and then solve that new problem. This is the approach taken by the extended Kalman filter (EKF) in which a linearization is done and the rest of the computation is similar to the classical Kalman filter. The second approach is to approximate the probabilities. This is the approach of the sequential Monte Carlo (SMC) particle-based filters. A review of sequential Monte Carlo methods is provided below.

Particle filters

This section gives a review of the numerous variations and designs of particle filters and highlight the aspects of these sequential Monte Carlo (SMC) methods that make online Bayesian estimation possible.

In the Monte Carlo framework, an estimate of the mean of a set of N samples from a distribution $p(s)$ under the function $f(s)$ is given by

$$\mathcal{E}(f) = \frac{1}{N}\Sigma_{i=1}^{N} f(s_i). \tag{2.21}$$

However, there is often no way to sample from the underlying distribution $p(s)$, making the computation of equation 2.21 impossible. Fortunately there is an approach called importance sampling whereby a proposal density $q(s)$ is sampled from instead of the true distribution $p(s)$. Below is a summary of the importance sampling approach and follows the work done by Doucet [14, 22]. Provided that the support of the proposal distribution satisfies

$$q(s) > 0 \Rightarrow p(s) > 0, \tag{2.22}$$

the Monte Carlo expectation equation 2.21 can be rewritten as

$$\mathcal{E}(f) \approx \frac{1}{N}\Sigma_{i=0}^{N} \frac{p(s^i)}{q(s^i)} f(s^i). \tag{2.23}$$

Let $w^i \triangleq \frac{p(s^i)}{q(s^i)}$ be the set of importance weights. Note that in reducing this ratio to a set of weights the need to be able to evaluate p and q exactly is not required, and knowing them up to a normalizing constant is sufficient as the set of weights can always be renormalized as follows:

$$w^i = \frac{w^i}{\Sigma_{j=0}^{N} w^j}. \tag{2.24}$$

In the particle filter (PF), which uses sequential importance sampling (SIS), the posterior is held by a set of weighted samples S_k at time instant k. Thus the posterior is estimated as

$$p(S_k|Z_{1:k}) \approx \Sigma_{i=0}^{N} w_k^i \delta(s_k - s_k^i) \tag{2.25}$$

where the importance weights are

$$w_k^i \propto \frac{p(s_k^i|Z_{1:k})}{q(s_k^i|Z_{1:k})}. \tag{2.26}$$

By substituting the approximate posterior equation 2.25 into the Bayes' rule for updating the posterior, the sequential Monte Carlo weight update is given by

$$w_k^i \propto w_{k-1}^i \frac{p(Z_k|s_k^i)p(s_k^i|s_{k-1}^i)}{q(s_k^i|s_{k-1}^i, Z_k)}. \tag{2.27}$$

There are a number of design choices and algorithmic pathologies. These are now described and further implementation details are given in chapter 5.

The choice of a proposal density $q(s)$ determines how effectively the particles represent the true posterior. A poor choice of $q(s)$ will lead to an inefficient use of particles, as a large portion of them will get assigned with small weights and so add little description to the posterior. A common choice of $q(s)$ is the prior $p(s_k^i|s_{k-1}^i)$, which is what the particle filters in this work uses. However, even with a good choice of $q(s)$ the variance of importance weights can only increase [13]. This leads to the same issue as a poorly chosen $q(s)$ and is known as the degeneracy problem. The mass of the posterior is held by fewer and fewer particles, which results in a poor approximation as well as wasted computational effort. This problem is partially solved by a resampling step. The decision as to when resampling should take place is given by the following:

$$\hat{N}_{\text{eff}} = \frac{1}{\Sigma_{i=1}^{N}(w_k^i)^2}, \tag{2.28}$$

where $\hat{N}_{\text{eff}}$ is an approximation to the effective number of particles [13] and resampling is done if $\hat{N}_{\text{eff}} < N_{\text{threshold}}$.

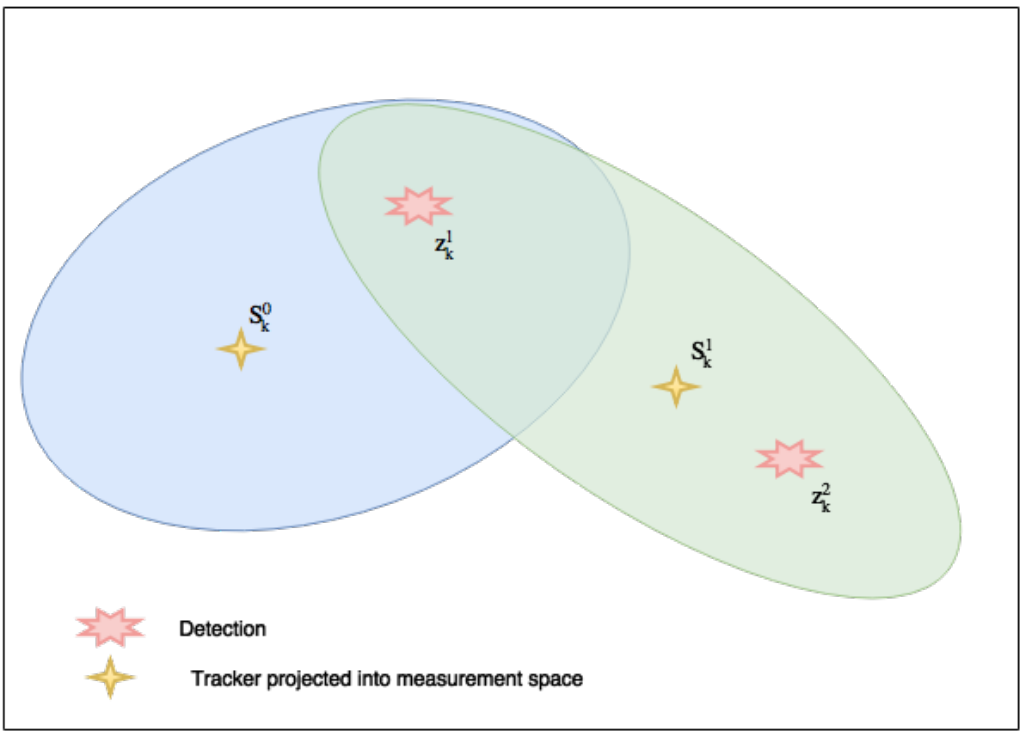
The choice of resampling algorithms, of which there are many, needs to be made. There are various theoretical and practical trade-offs and a very complete and recent summary of these is given by [34]. This work is not replicated here and the particular techniques used for the two applications are detailed in chapters 5, 7 and 8.

2.4 Review of predominant MTT algorithms

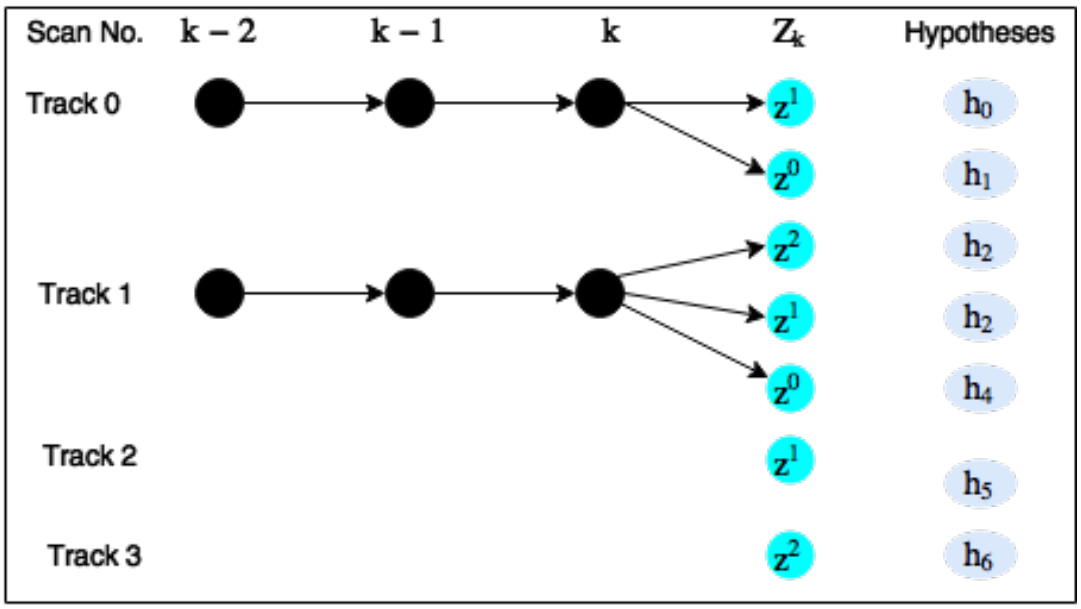
There are three predominant approaches to the multiple target tracking (MTT) problem. The first two, multiple hypothesis tracking (MHT) and the joint probabilistic data association filter (JPDAF), are widely used in academia and industry while the third, the random finite sets (RFS) tracker, is a more recent approach and is making significant impact. This section gives a brief overview of these approaches and closes with comments on how the multiple target tracking framework developed in this work compares to these three methods.

The multiple hypothesis tracking initially formulated in its hypotheses oriented manner by Reid [46] has since seen a few restatements. A track oriented approach and the most cited is that proposed by Bar-Shalom [4]. The MHT algorithm uses deferred decision logic [6], which allows for using more than one scan interval set of measurements

in the association of tracks to detections. This is done by building up a hypothesis tree as shown in figure 2.5b.



(A) Multiple hypothesis tracking hypothesis gating scheme. The gating scheme limits the possible tracker detection associations.



(B) Multiple hypothesis tracking hypothesis tree. At time instant k there are two trackers present in the scene and two detections are made z^1, z^2. Based on the gating in figure 2.5a tracker_0 (S^0) can only be associated with one detection z^1 or assigned a missed detection z^0. Tracker_1 (S^1) can be associated with either detection z^1 or z^2 or the missed detection z^0. For each tracker the hypotheses are shown as well as the case for birthing new trackers. These hypotheses are denoted by h_i.

Using the hypothesis tree the MHT algorithm reduces the association uncertainty, at each time step, by searching over previous association possibilities. In order to curb

the computational complexity that arises from the rapid growth of hypotheses the MHT algorithm applies gating and N-scan pruning. Gating is performed to remove infeasible measurement to track associations. This is done in the measurement space and simply puts a threshold on how far away a measurement and predicted measurement can be. Figure 2.5b shows the gating ellipses. Scan pruning reduces the number of tracks by eliminating hypotheses that have low probability. Under these two techniques numerous approaches have been taken [61]; however, they are necessary for all but the trivial MTT case. Since at each scan instant a hypotheses is created for the birth of a new track and only the tracks with the highest posterior probability are propagated, the MHT approach handles a time-varying number of targets. Once the most probable hypothesis has been selected a standard single target BRE such as a Kalman filter can be used on the associated measurements for each track to estimate the states and trajectories of individual targets. A full evaluation of all hypotheses in the MHT algorithm is not tractable and the heuristic gating and pruning techniques are needed for all real-world applications.

The joint probabilistic data association filter (JPDAF) is probably the most popular formulation and is widely used in both civilian and military applications. It is the natural extension of the probabilistic data association filter (PDAF) which propagates a single mode posterior for single target tracking, where it weights the importance of each measurement according to its association probability. This mitigates the risk of wrong assignments and is know to perform well in practical applications. The JPDAF extends the PDAF to multiple targets by extending the state and computing the joint association probabilities. This computation increases exponentially with the number of targets and the number of detections [62]. The JPDAF assumes a fixed number of targets, however [42] shows a way to extend this to varying number of targets and [60] makes use of sequential Monte Carlo methods to approximate the joint association density. The JPDAF makes soft assignments according to the joint association density. It updates the posterior distribution over all states based on the weighted contribution of these soft assignments.

Lastly, and in the author's opinion the most general, is the random finite sets (RFS) approach which was established and popularized by Mahler in this book [38]. Random finite sets are a generalization of random variables. That is, both the number of elements in the set and the elements are random variables. A random finite set is simply a random variable that takes values as (unordered) finite sets [62]. Mahler [37] put forward finite set statistics (FISST) which formalizes the notion of distributions over RFS. This enables a straightforward analogy from the single target Bayesian recursive estimator (BRE) to the case for multiple targets. The theoretical details are less straightforward and can be found in Mahler's work [37]. The Bayesian RFS formulation avoids the conceptual and technical inconsistencies caused by explicit data associations between objects and measurements [62]. Practical implementations of the RFS approach are limited and under strict Gaussian assumptions the probability

	Cardinality estimation	Target birth & death	Full BRE	Association stage
MHT	No	Yes	No	Yes
JPDAF	No	No	No	Yes
RFS	Yes	Yes	Yes	No
MTTF	Yes	Yes	No	Yes

TABLE 2.1: Table comparing MTT algorithms.

hypothesis density (PHD) filter and the cardinalized probability hypothesis density (CPHD) filter, which are first moment approximations, have been shown to work. A comprehensive simulation-based comparison between the MHT and CPHD filters [53] was conducted and the results are compelling. The CPHD filter reacts quicker to changes in cardinality due to targets disappearing and appearing [53]. The PHD filter is derived using FISST by Mahler [39]. The CPHD filter which jointly propagates the PHD and the cardinality distribution [61] has better performance but at a higher computational cost. The use of RFS for describing the multiple target tracking problem is both intuitive and effective, as seen by the work cited above. It extends the BRE to the multi-target case which is the goal of the MTT framework developed in this work.

The work presented here was borne out of a combination of objectives: to build a practical MTT system to monitor airspace and to develop a coherent MTT framework by extending the Bayesian recursive estimator to the multi-target case. The three algorithms described above and the MTT framework developed here are compared in table 2.1.

The MTT framework is most similar to the joint probabilistic data association filter in that a notion of an association distribution is constructed during the association stage. Both these methods use a single scan to compute the data association decision for a sensor sample instance. The framework developed here allows for many trackers to evolve using the same reported detection. Allowing ambiguous associations to occur is done as the measurement space $\mathcal{Z}$ is a projection of the state space $\mathcal{S}$, and so targets detected close to one another does not mean they are close in the state space. Holding multiple trackers in the collective state vector to which the same set of detections are associated can be interpreted as exploring multiple hypotheses aimed at explaining the same set of observations. In this way the MTTF is similar to the MHT approach as both are able to explore differing interpretations of sets of temporally spaced observations. The MTTF reduces these hypotheses as the configuration potential in the data association graphical model penalizes targets that are close in the state space $\mathcal{S}$. The RFS approach is superior to all methods looked at in this study, in the sense of adhering to the Bayesian recursive estimator. The practical implementations that exist are not fully developed in the literature and it remains to be seen whether this theoretically-sound approach performs well on real-world applications such as those explored here.

2.5 Probabilistic graphical models

Probabilistic graphical models are a useful tool for describing multivariate distributions. They provide a visual representation that is very general and describes the structure of a multivariate distribution in such a way that higher level reasoning and interpretation can be easily made. The book by Koller [31] offers a very good introduction and foundation of these techniques. Graphical models can be represented by many graph structures and these go by different names: Bayesian networks, Markov random fields (MRFs), conditional random fields (CRFs), factor graphs (FGs) and numerous other variations. These structures all have various advantages and disadvantages, and references [54] and [43] illustrate the similarities, interpretations and strengths of one type of graphical model representation over another. The data association stage developed in this work draws heavily on the use of these structures and in particular on factor graphs (FGs).

2.5.1 Factor graphs

Factor graphs are a bipartite graph with nodes X that correspond to random variables and nodes F that correspond to factors. The edges in the graph link variable nodes to factor nodes corresponding to the variables that appear in the arguments of the factor nodes F.

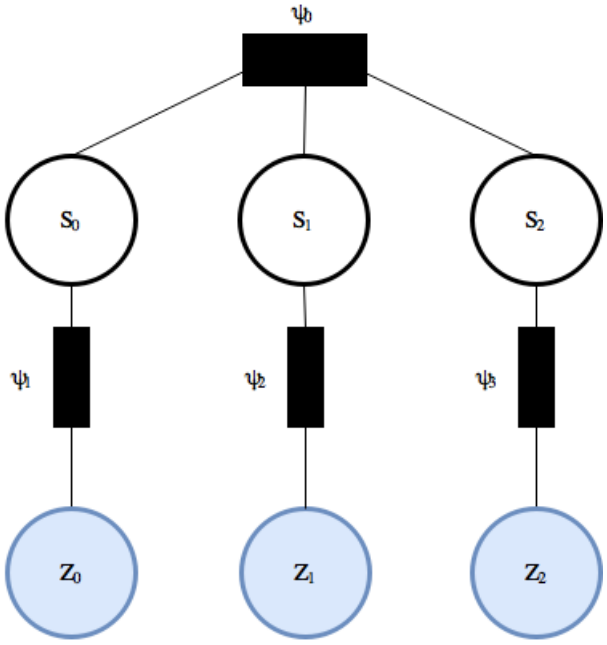


FIGURE 2.6: A generic factor graph structure. This graph factors the joint distribution as a product of four potentials,

$$p(S_0, S_1, S_2, Z_0, Z_1, Z_2) = \frac{1}{Z}\psi_0(S_0, S_1, S_2)\psi_1(S_0, Z_0)\psi_2(S_1, Z_1)\psi_3(S_2, Z_2),$$

where Z is a normalizing constant.

More formally, a factor graph $G = (V, E)$ is a bipartite graph with variable nodes and factor nodes:

- **Variable nodes** Vertices in the graph are depicted as clear circles. Each random variable in the distribution corresponds to a variable node on the graph.
- **Factor nodes** Vertices in the graph are depicted as solid rectangles. Each factor of the distribution correspond to a factor node on the graph.

Thus the joint probability $p(s_0, s_1, s_2, a_0, a_1, a_2)$ factors according to graph G shown in figure 2.6 as

$$p(s_{0:2}, a_{0:2}) = \frac{1}{Z}\psi_0(s_0, a_0)\psi_1(s_1, a_1)\psi_2(s_2, a_2)\psi_3(s_0, x_1, s_2). \tag{2.29}$$

The use of a graphical model enables one to give structure to the joint distribution that arises from domain or problem-specific knowledge. There is recent and promising work on learning graph structures as well as their parameters. This is called structure learning or simply model selection and is covered by [32, 15]. The work presented here does not make use of structure learning as the problem lends itself to predetermined structures. The system that is being tracked is a physical process and it is clear that observations are generated by the system, so causality enables one to make sensible graph structures that factor the joint densities in a coherent manner. This is not a new approach and numerous authors draw the links between classical Kalman filters (KFs) and their hidden Markov model (HMM) graphical structure. See section 2.3 and in particular figure 2.4.

2.5.2 Inference on factor graphs

There are numerous approaches to performing inference in graphical models. The aim of any inference technique is to infer one or more of the following problems [63]:

1 Computing the likelihood of data.

2 Computing the marginal distribution over a particular subset of variables.

3 Computing the conditional distribution for two disjoint subsets of variables.

4 Computing the mode of the density. This is the maximum a posteriori (MAP) estimate.

There are various techniques to calculate the quantities listed above. These range from brute force exact solutions using variable elimination to approximate solutions using variational methods, Monte Carlo sampling or message passing algorithms. The Coursea course *Probabilistic Graphical Models* run by D. Koller presents this material in a very accessible manner. The following [40, 31, 32] provide details on these inference techniques as well as the advantages and disadvantages each approach

poses. The approach briefly described below is that of belief propagation or sum-product message passing. This technique is used in chapter 6 to solve for the maximum a posteriori (MAP) label assignments of detections to trackers.

The sum-product message passing algorithm was first proposed by Pearl [44]. It works by passing messages (approximate values) along edges between nodes.

2.6 Review of background foreground segmentation

Understanding a scene given a sequence or video stream is a difficult and unsolved task. A simpler subset of scene understanding is background subtraction and foreground detection. There have been many and varied attempts at solving this segmentation problem. Despite being a crucial step in nearly all computer vision applications there is no complete solution. This work makes use of a statistical approach to the segmentation problem using Gaussian mixture models (GMMs). This section gives a brief overview and describes the pertinent mechanics of this approach. For an excellent summary of numerous approaches as well as many implementations see [8, 49] and the relating github repository [48]. These methods have a range of advantages and disadvantages and the choice varies depending on the application. Systems that have little computing power available, such as those running on mobile platforms, might benefit from using simple frame difference techniques. Applications using a stationary camera but needing to handle a dynamic scene benefit from using statistical methods as they can adapt to changes in the scene. Statistical methods provide an interpretable and calibrated classification score. The score, $p(z_i = background|m, \theta)$, is the probability of pixel z_i being a background pixel given the particular model m and its parameters θ. This is very informative to the next stages of the processing pipeline, especially if decisions are being made based on that information.

The manner in which the background-subtraction stage fits into the visual object tracking (VOT) system is best shown graphically as is done in chapter 3. A Gaussian mixture model (GMM) is a model used to describe the underlying generative process that produces the pixel values. Each pixel is viewed as a separate process and the Gaussian mixture model describing it is of the following form:

$$p(z^{(p)}|Z_T, BG) \approx \Sigma_{m=1}^{M} \hat{\pi}_m \mathcal{N}(z^{(p)}; \hat{\mu}_m, \hat{\sigma}_m \mathcal{I}). \tag{2.30}$$

Each pixel $\left[z^{(p)}\right]_{p=0}^{p=numPixels}$ is described by a mixture of Gaussian distributions with estimated parameters $\hat{\mu}_m$ (mean) and $\hat{\sigma}_m$ (variance) and mixing weights $\hat{\pi_m}$. The parameters are estimated based on a training set of frames Z_T which are captured over a duration of T time intervals. Parameter estimation is done recursively based on the results of [67].

Recent applications of GMMs to background subtraction have made some key improvements that enable handling dynamic backgrounds that change over time. These mechanisms

are described in [68] whereby the training set is continuously updated and the parameters as well as the number of mixture components M are continually re-estimated.

The problem with a Gaussian mixture model (GMM) pixel-wise background model $p(z^{(p)}|Z_T, BG)$ is that there is no spatial reasoning held within the model. Each pixel process is estimated independently while it is clear that most objects close together in an image are in fact the same object in the scene and should be classed as such. The works that best latch onto spatial relationships between pixels are conditional random fields (CRFs) and Markov random fields (MRFs). In brief, these models specify a conditional probability or potential between pixels sharing a common neighbourhood and so nearby pixels are labelled coherently. Sutton gives a clear and concise introduction to conditional random fields and how they can be used to solve the image segmentation problem [52]. Further comments on the various methods based on experimentation using the datasets captured for this study are made in section 4.2.

2.7 Conclusion

This concludes the review of pertinent literature and techniques that this investigation draws on. A description of the Bayesian framework and its main theoretical pillars were put forward. Then a fairly comprehensive overview of state estimation techniques from closed form recursive solutions to sequential Monte Carlo (SMC) based approximate solutions were described. A basic yet sufficient introduction to probabilistic graphical models was put forward together with notations and conventions. Lastly, background subtraction was described with particular attention paid to Gaussian mixture models, which is the approach used in the visual object tracking application presented here.

Chapter 3

Experimental setup and system architecture

This chapter describes the experimental setup and gives an overview of the visual object tracking (VOT) system that was built. An experiment was set up to evaluate the efficacy of the developed MTT framework. The airspace in the region of the Cape Town international airport was monitored using a static camera and this data was processed by the VOT application. Thereafter the output target tracks were compared to truth data obtained from *FlightRadar24* [18], to provide a qualitative measure of the success of the MTTF approach and visual object tracking system. Since an aim of this project was to build a fully functioning application to solve the general problem of passively monitoring airspace, this chapter describes the practical engineering aspects needed to achieve a complete visual object tracking system. Section 3.1 details the site at which all experiments were carried out. This gives the application context and the performance can only be viewed through the understanding of the site and conditions under which the system was tested. Section 3.2 details the system architecture. It gives a system overview and demonstrates the graphical user interface (GUI) which ties all the components together into a deployable, functional system. The calibration and pose estimation details are provided in section 3.3 and the resulting measurement model is derived in section 3.3.3. Concluding remarks on the experimental setup are given in section 3.4 and deficiencies in the setup are highlighted.

3.1 Site description

The camera is situated on the roof of the Menzies building at the University of Cape Town (UCT). A typical view from this vantage point and the overall setup is shown in figure 3.1.

The site offers a favourable view over the airspace commonly used by commercial airliners when arriving at the Cape Town international airport. A map of the area

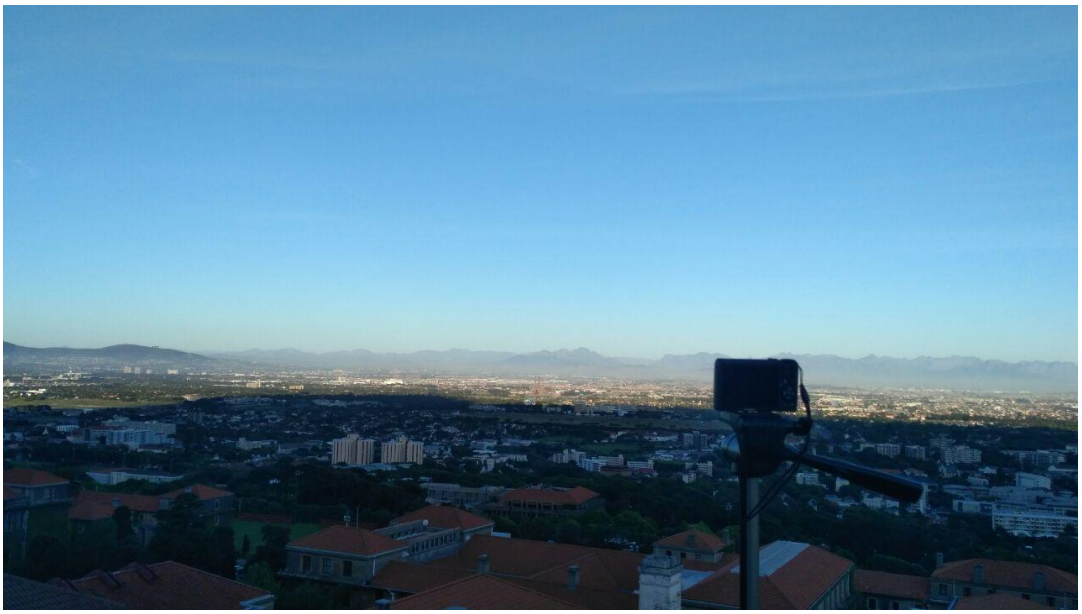


FIGURE 3.1: View of the camera setup.

with typical flight paths is shown in figure 3.2. The camera is pointed at 65° from North.

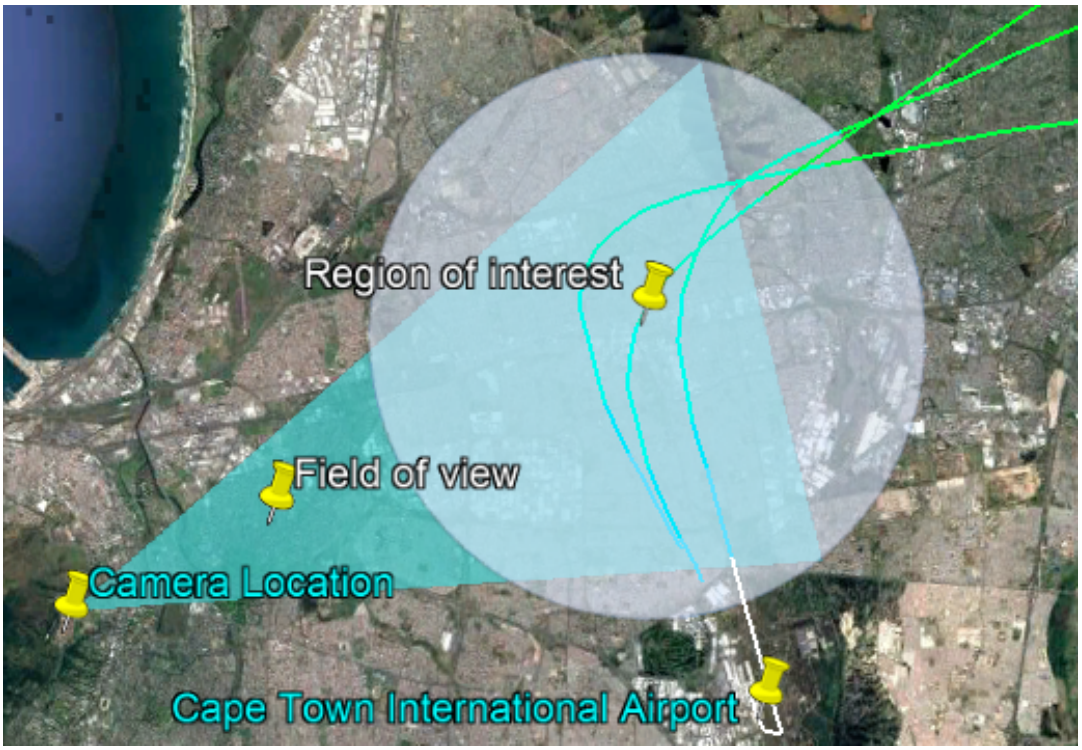


FIGURE 3.2: Site layout and typical flight paths

The camera used for all the experiments was a Sony S100. It was set to record at $23\,\text{fps}$ with a fixed zoom and a frame resolution of 1920×1080 pixels. The system architecture and the system setup consisting of camera calibration and pose estimation

are described below.

3.2 System architecture and graphical user interface

This section describes the visual object tracking system architecture from a functional perspective. The system is made up of number software processing modules and data queues. Figure 3.3 shows how these are connected. It also highlights how new modules could be built and plugged in. This section concludes with a view of the VOT system from a user perspective, showing it being used as a passive airspace monitoring system deployed at the University of Cape Town.

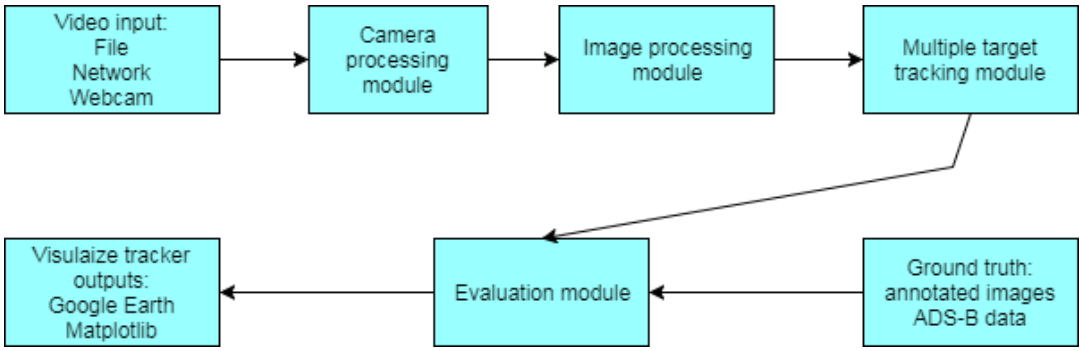


FIGURE 3.3: Visual object tracking processing pipeline.

The video input block connects to an IP camera over TCP/IP or reads a video file as a stream and pushes the data frame by frame to the camera processing module (CPM). The CPM wraps the current frame into a data structure, called a camera frame, and adds metadata (timestamp, frame count, pan, tilt and zoom levels) and enters it onto a queue. The image processing module takes a camera frame off the queue, applies the segmentation algorithm and passes the detections onto the multiple target tracking module. The MTT module solves the data association problem and updates the current set of trackers. It births and destroys trackers as needed to explain the observations it has received. It then writes all estimates to file at each time interval as a set of image plane (u, v) pixel locations, world and local world coordinates and the associated confidence intervals for all these estimations. The evaluation module then reads all outputs as well as the truth data, both annotated image plane positions and ADS-B world coordinate data acquired from [18]. Estimation errors are computed and the visualization module outputs these to Google Earth and plots are made using the Python library *Matplotlib* [23].

All the modules mentioned above, together with a calibration and pose estimation application and a passive radar MTT module, were integrated into a graphical user interface (GUI). Figure 3.4 shows the Qt [45] based application.

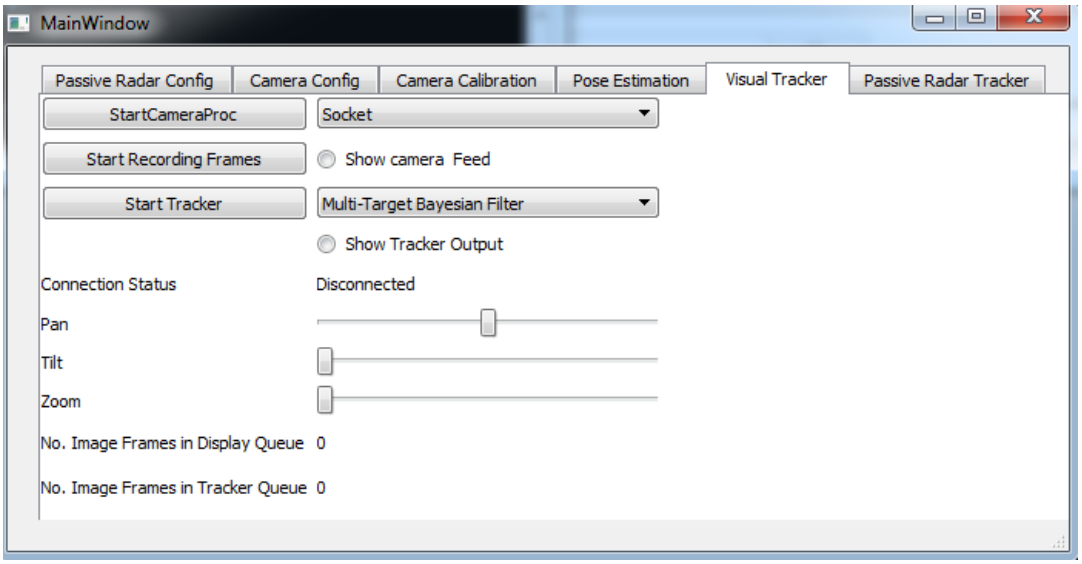


FIGURE 3.4: Graphical user interface overview.

All practical aspects of this work are implemented in this application and the final product is a multi-sensor airspace monitoring system.

3.3 Camera calibration and pose estimation

In computer vision, estimating the camera pose from a set of n 3D to 2D point correspondences is a fundamental and well-understood problem [24].

The visual object tracking (VOT) system has an integrated calibration and pose estimation module (CPEM). This module connects to the video input module (VIM) in file or network configuration, and the CPEM computes the intrinsic and extrinsic parameters. Details of these two processes are provided below: calibration in section 3.3.1 and pose estimation in section 3.3.2. These sections are brief as the literature is well established [21] and off the shelf implementations from the *OpenCV* library [24] were used to perform these operations. It is included here to give further clarification about the experiments that were conducted. The GUI of the calibration and pose estimation module is shown in figure 3.5. The CPEM establishes the measurement model and is specified in section 3.3.3.

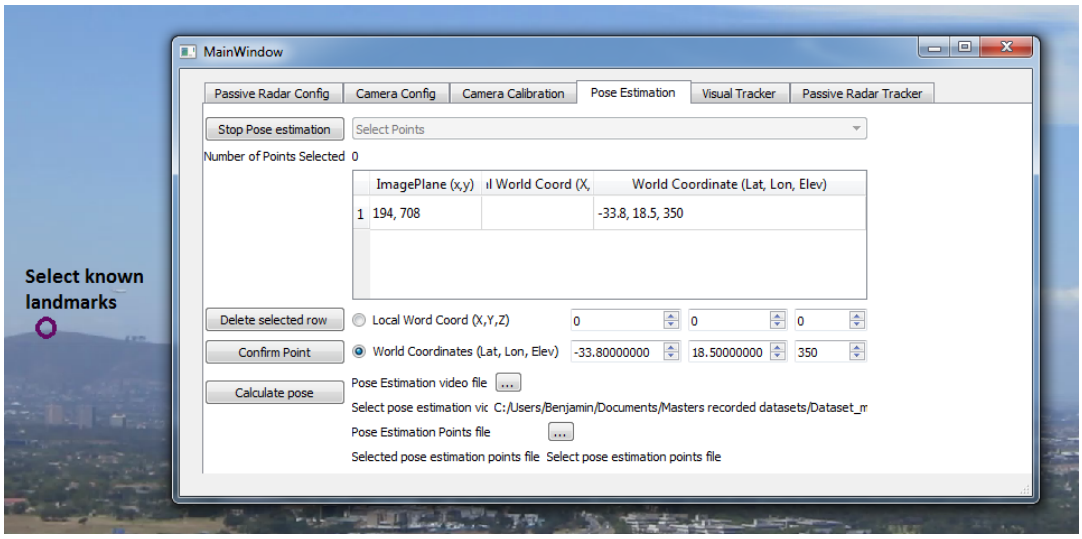


FIGURE 3.5: Graphical user interface of the calibration and pose estimation module (CPEM).

3.3.1 Camera calibration

Camera calibration is the process of estimating the parameters of the lens and image sensor. These are collectively referred to as the intrinsic parameters. Intrinsic properties are those internal to the camera and are constant for all sites. These are the focal length and the principal point, and are held in the 3×3 matrix

$$K = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix}, \tag{3.1}$$

where f_x and f_y are the the focal lengths measured in pixels and (c_x, c_y) is the principal point. The camera matrix K as well as the distortion coefficients for radial and tangential distortions were determined using *OpenCV* [24] built-in functions. This was done to correct for lens distortions and was achieved in the standard manner using a 7×9 checkerboard.

3.3.2 Camera pose estimation

The extrinsic characteristics of the camera are needed to map points in the 3D world to points on the image plane. This mapping is the measurement equation of the camera and will be referred to as $H(.)$. Thus, a point in the local world coordinate frame (X, Y, Z) is projected onto the image plane to a point (u, v) by the measurement equation

$$[u, v]^T = H(X, Y, Z). \tag{3.2}$$

The state vector s contains the local world position (x, y, z) and $H(s)$ is a function of the state s. To determine this transform, a set of at least six point correspondences is required. The set consists of local world points and their corresponding image

points. The calibration and pose estimation module (CPEM) allows a user to select known points in the image and enter the corresponding world coordinates; see figure 3.5. Once a minimum of six points have been selected in this manner, a local world Cartesian coordinate system is defined with the camera at the origin. Although, a minimum of three points are required to solve the remaining degrees of freedom (as the camera is calibrated) the random sampling consensus (RANSAC) approach used here performs better with multiple point pairs and the system was designed to use a minimum of six. The C++ implementation of *GeographicLib* [29] with the world geodetic system 1984 (WGS84) was used to define this local world coordinate system. The CPEM uses the RANSAC point-n-perspective implementation to compute rotation matrix R and translation vector t. The final measurement equation $H(s)$ in homogeneous coordinates with scale factor is

$$H(s) = \text{scale} \begin{bmatrix} u \\ v \\ 1 \end{bmatrix} = \begin{bmatrix} f_x & 0 & c_x \\ 0 & f_y & c_y \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} r_{11} & r_{12} & r_{13} & t_1 \\ r_{21} & r_{22} & r_{23} & t_2 \\ r_{31} & r_{32} & r_{33} & t_3 \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \\ 1 \end{bmatrix}. \tag{3.3}$$

The scale factor is needed to renormalize the homogeneous coordinate and retrieve the image plane coordinate (u, v).

3.3.3 Measurement model

The measurement model is approximated by a Gaussian distribution in the (u, v) image plane given by

$$p(z|s) = \frac{1}{(2\pi)^{b/2} \left|\Sigma_{\text{obs}}\right|^{1/2}} e^{-\frac{(H(s)-z)^T \Sigma_{\text{obs}}^{-1} (H(s)-z)}{2}}. \tag{3.4}$$

The mapping from state space to measurement space $\mathcal{H} : \mathcal{S} \rightarrow \mathcal{Z}$ is formed by the camera intrinsic matrix K combined with the rotation matrix R and the translation vector t to give the projection matrix P,

$$P = K\left[R|t\right]. \tag{3.5}$$

A conversion from the 2D image plane coordinate to the 3D homogenous representation is needed and the (u, v) pixel coordinates can be computed by dividing each component by the third component. The projection is described by

$$\left[u, v, 1\right]^T = P\left[X, Y, Z\right]^T \tag{3.6}$$

In the multi-tracker case the system state is made up of the collective state of all trackers. The collective state is $S = \{s_i\}_{i=0}^{i=N_t}$ where s_i is an individual trackers state

and N_t is the number of trackers held in the collective state. The log likelihood function $\mathcal{L}_{log}(S, Z)$, which describes how likely a measurement is given the system is in a particular state S, is given by

$$\mathcal{L}_{log}(S, Z) = \Sigma_{m=0}^{m=N_d} \Sigma_{i=0}^{i=N_t} \log(p(z_m|s_i) I(m, i)) \tag{3.7}$$

where $I(m, i)$ is an indicator function which evaluates to one if the m^{th} detection is assigned to the i^{th} tracker and zero otherwise. Chapter 6 describes this process in detail. The emphasis here is to explain how the mapping from state space to measurement space was achieved and how it is used to formulate the likelihood function.

3.4 Conclusion

This concludes the system engineering and practical components of the visual object tracking (VOT) system. The VOT system is a complete, deployable software application that has all the necessary components for a functioning passive airspace monitoring system. This application was built in its entirety by the author for this dissertation. The experiment site was described and the problem of passive airspace monitoring is presented as a use-case example which is the focus of this investigation.

It is noted that this experimental setup has deficiencies. Truth data is only gathered for some aircraft as provided by [18] and the accuracy of this information is unknown. The truth data collected is sufficient to validate the VOT system as a whole. However, for determining accurate performance characteristics this setup lacks curated reliable truth data. Setting up field trials in a manner such that all needed truth data can be collected requires significant time and monetary resources which could not be justified in relation to the scope of this project.

The next chapter expands on the functional explanation of the VOT system by describing the image processing pipeline.

Chapter 4

Image processing module

This chapter describes the image processing module that the visual object tracking system uses. Figure 3.3 illustrates where this module fits within the overall VOT system's processing pipeline. The purpose of this module is to read in camera frames off the camera processing module (CPM) queue and output a likelihood map. This likelihood map is a single channel image where each pixel is a value between zero and one. Each pixel value indicates how likely the physical process that generated the light ray corresponding to that pixel is part of the background. This likelihood map is the output of a Gaussian mixture model of the background process viewed as a pixelwise independent process. Section 4.1 details the Gaussian mixture model and sets up the notation used throughout the document. It shows how the likelihood map is generated and how the thresholding is done to produce a set of detections per frame. A review of three discriminative object detectors that were investigated as possible solutions to detecting aircraft in the scene is given in section 4.2. These approaches were not used in the final application but are recorded here as it provided further insights into the VOT problem. The same section highlights the limitations of the selected approach and the need for a multiple target tracking algorithm for the visual object tracking application is made clear.

4.1 Background subtraction using a GMM

Section 2.6 gave a review of the state-of-the-art background segmentation algorithms. The concept of a Gaussian mixture model is also introduced there. This section covers the GMM background subtraction technique in detail and links it to the first application, namely visual tracking of aircraft. Visual object tracking makes use of information received from an unknown probabilty density function (PDF) and only samples (pixels) from this PDF are available. These samples are recorded by a camera which has its own intrinsic characteristics; see section 3.3.1. Thus the value of a pixel through time is viewed as a stream of samples from some unknown density. All the unknown pixel densities are approximated using a Gaussian mixture model (GMM) with M components. The GMM model fits parameters to approximate the underlying density per pixel location.

The value of a pixel at time t in blue-red-green (BRG) channel order is denoted by $z_{(u,v)}$ where (u, v) are the pixel coordinates and the value is a tuple of blue, red and green channel values. The likelihood of a pixel being labelled as background is given by

$$p(B|z_{(u,v)}) = \frac{p(z_{(u,v)}|B)p(B)}{p(z_{(u,v)})} \tag{4.1}$$

and the likelihood of a pixel being labelled foreground is given by

$$p(F|z_{(u,v)}) = \frac{p(z_{(u,v)}|F)p(F)}{p(z_{(u,v)})}. \tag{4.2}$$

In order to determine which is more likely, a ratio of the likelihood of background $p(B|z_{(u,v)})$ to the likelihood of foreground $p(F|z_{(u,v)})$ is used. If the ratio of likelihoods given by

$$\frac{p(B|z_{(u,v)})}{p(F|z_{(u,v)})} = \frac{p(z_{(u,v)}|B)p(B)}{p(z_{(u,v)}|F)p(F)} \tag{4.3}$$

is greater than one, then the pixel $z_{(u,v)}$ is defined as background and vice versa. Since a foreground model is not used to determine whether a pixel is part of the background, a comparison of $p(z_{(u,v)}|B)$ to a threshold is needed. Another consequence of not having an explicit foreground model is that every sample is used to update the GMM. This means the model held per pixel given a history data sequence of $\mathcal{D}$ is

$$p(z_{(u,v)}|\mathcal{D}, B, F) = \Sigma_{m=0}^{M} \pi_m \mathcal{N}(z_{(u,v)}; \vec{\mu}_m, \sigma_m^2 \mathcal{I}), \tag{4.4}$$

where π_m is a mixing weight and $\Sigma_{m=0}^{M} \pi_m = 1$. Each Gaussian component has mean $\vec{\mu}_m$ and covariance $\sigma_m^2 \mathcal{I}$ where $\mathcal{I}$ is the identity matrix, as it is assumed that the covariance between channels is zero and so the resulting matrix is diagonal.

The assumption is made that the background features persist for longer than the foreground features. Thus, the component weight π_m is larger for the background components. The background model $p(z_{(u,v)}|B)$ is approximated as the C largest components

$$p(z_{(u,v)}|B) \approx \Sigma_{m=0}^{C} \pi \mathcal{N}(z_{(u,v)}; \vec{\mu}_m, \sigma_m^2 \mathcal{I}), \tag{4.5}$$

where

$$C = \underset{c}{\operatorname{argmin}} \ (\Sigma_{m=0}^{c} \pi_m < (1 - c_f)) \tag{4.6}$$

and the components have been sorted by descending weight. The parameter c_f dictates what portion of the scene can be attributed to the foreground without affecting the background model [66]. The GMM implementation used in the VOT system was taken from *OpenCV* [24] and the likelihood map of foreground is denoted by $\mathcal{I}_{fg}$. The likelihood map $\mathcal{I}_{fg}$ is a 2D matrix of similar dimensions to the input frame and its elements are referred to in pixel coordinates as $i_{fg}(u, v)$.

4.1.1 Generating detections

From the likelihood map output of the GMM background model a set of detections or contacts are formed. These are pixel locations that have a foreground score greater than the detection threshold. Each frame produces a set of detections according to

$$\{Z_k | Z_k \in \mathcal{I}_{fg}, i_{fg}(u,v) > 0.5\}. \tag{4.7}$$

If neighbouring pixels exceed the threshold only the first pixel is deemed a detection and the rest are ignored. The first pixel is determined by scanning row by row from top left to bottom right of the likelihood map. Pixels are neighbours if they are within 32 pixels of each other.

If one views each frame as a sample from the underlying distribution then this is not a strong generative model, as it is incapable of generating samples that look like the training set of samples. This is illustrated by figure 4.1. This weakness arises from three shortcomings in this approach. Firstly, the physical process is not explicitly modelled at all. Thus the GMM cannot include prior knowledge about the scene as the scene itself is not explicitly modelled. Secondly, the spatial relationship between pixels has no bearing on the model. This is intuitively a problem as objects in the scene often project to neighbouring pixels. The notion of neighbourhoods of pixels is not captured. Lastly, the model generates samples as from one of the modes describing the pixel process. A pixel value drawn from the model is a maximum a posteriori (MAP) estimate from all the Gaussian mixtures that model a particular pixel. This leads to a blurred image which is never observed in reality. Consider a scene which consists of blue sky with the occasional white cloud. The Gaussian mixture model could fit a two-mode model with the mean of the first component centred on the blue value and the second component centred on the white value. If the model is queried for an estimate of pixel values, one of two values for each pixel in that region could be produced. A blue value drawn from the blue-centred component or a white value centred on the white-valued component. Since there is no spatial relation between pixels there is no reason for the model to produce, under any of these two schemes, an image that correctly reflects the real world. This leads to a poor generative model and yet a GMM approach was chosen for the VOT application. Despite this clear and known limitation of the GMM-based background models, it works surprisingly well in practical applications such as this. This was a system design choice and since the scope of this work does not extend to a full investigation to remedy these shortcomings, section 9.2.3 provides recommendations and the authors insights.

(A) Input frame to be processed.

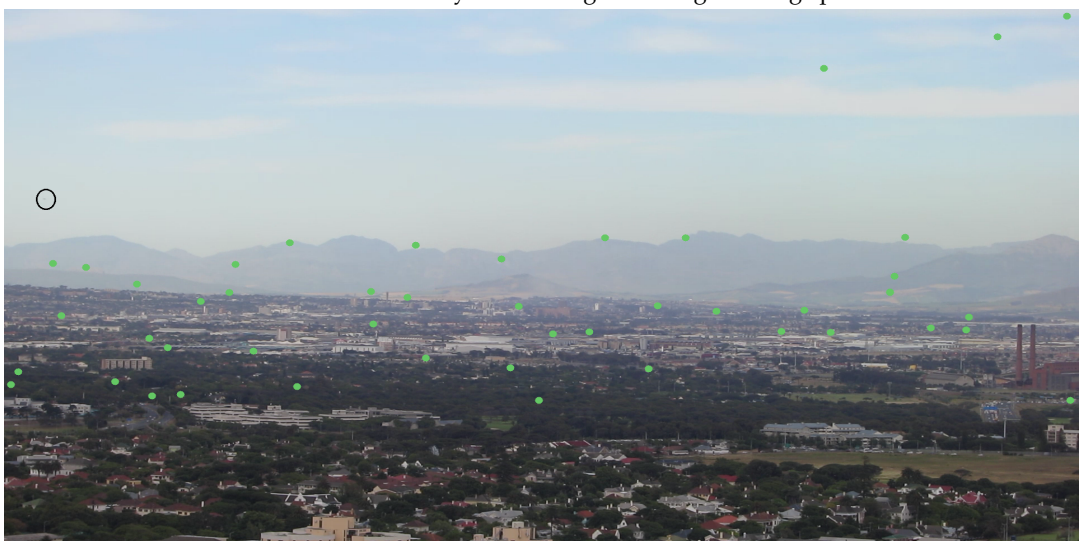
(B) Sample from the background model showing the blurred and unrealistic nature of these samples.

(C) Potential target locations.

FIGURE 4.1: A typical sample from the Gaussian mixture background model.

After applying threshold to the likelihood map a set of detections are generated. A typical set of detections are shown in figures 4.2a and 4.2b. There are typically 0 to 50 detections per frame.

(A) A typical collection of 10 detections plotted on the input frame. Green dots show detection locations and the one encircled by a black ring is the target's image plane location.



(B) A collection of 47 detections plotted on the input frame. Green dots show detection locations and the black ring encircles the target. In this frame all reported detections where false alarms.

FIGURE 4.2: Typical collection of detections for a single frame.

This concludes the image processing aspect of this investigation. A few comments on the shortfalls of this approach are highlighted above. Alternatives that were explored before selecting such a simple image processing pipeline are discussed in section 4.2.

4.2 Alternative detection methods

When building the visual object tracking (VOT) system three attempts were made to build a target appearance model. In the computer vision community it is common for a VOT system to incorporate a target appearance model. These often take the form of a discriminative model with which target candidates are classified. The typical VOT system architecture is shown in figure 2.2. This architecture was explored for much of the time spent on this project. The three target appearance models utilized were a bank of image patches for template matching, a histogram of orientated gradients (HoG) feature-based support vector machine (SVM) and a convolutional neural network.

The template matching object detector used a bank of five image templates and these were correlated with candidate target locations. This approach worked well when the candidate image patch either contained the aircraft or was untextured such as blue sky. However, any clouds or buildings resulted in high correlation scores and so false alarms were numerous.

The HoG SVM was trained on a set of 500 images of aircraft. Each image was decomposed into a HoG feature vector and it was in this feature space that the SVM was trained. This approach was more robust than template matching but it too gave false alarms and it was an uncalibrated classifier. This meant that the score (distance from the decision boundary) associated with the labelling output was not a probability, which made it hard to integrate into a Bayesian framework in a coherent manner.

A five-layer fully connected convolution stage followed by a pooling stage neural network was investigated as a binary classifier. It was trained on a set of 500 samples. This was implemented using *tinyDNN* [16]. The neural network worked but it also did not provide a readily-available calibrated probability on the label output. It worked better than the other two methods but it had a large number of parameters and so a much larger training set would have been required. It was found that the network's performance changed dramatically on test cases that were significantly different to the samples used in the training set. Future work should investigate transfer learning [65], which would allow the use of publicly available datasets and then a smaller problem-specific one would not be such a limitation.

All these approaches were abandoned as they were found to be not robust enough and did not fit into the Bayesian recursive estimator (BRE) in their current forms. The main limitation of making an appearance model in the VOT application investigated here is that the target occupies only a few pixels and there is very little discriminative information held in such small target appearances.

Since the target occupies so few pixels (see figure 4.3), the approach taken was to

FIGURE 4.3: An example of a aircraft image template or training sample used to train the convolutional neural network or SVM classifier. These images are 32×16 pixels.

only model the background. As shown in section 4.1 this process generates a set of detections and these are then passed on to the MTT module. Removing the brittle target appearance model and relying solely on the GMM background model reduces the complexity of the image processing module. This decision increased the number of false and true detections per frame. This design decision forced the need for an multiple target tracking algorithm. The task of building a robust discriminative target appearance model was traded for a higher-order data association task. This was a critical system design choice and further insights and recommendations addressing this are put forward in section 9.2.3.

4.3 Conclusion

This concludes the description and discussion regarding the image processing module. The Gaussian mixture background model was selected as the application lent itself to this approach. In a static camera application such as this VOT system the lack of spatial constraints in the model appears to have a limited impact on its performance. A fixed camera together with a fairly static scene allowed the per pixel scoring scheme to retain a usable amount of global consistency. It is the static camera assumption that allows for such useful results from a very simple and openly naive approach.

Chapter 5

Trackers

This chapter outlines the development of the MTT framework (MTTF). This chapter starts by setting up the notation for a single target Bayesian recursive estimator. The BRE is extended to the multiple target case. It highlights the need for non-linear filters and a data association stage in order to solve the MTT problem.

Section 5.1 shows how the BRE is used for multiple target tracking. Section 5.2 details the particle filter (PF) which is the implementation approach selected to solve the MTT problem. The method of birthing and terminating trackers is dealt with in section 5.2.2. The complete MTT framework is presented in section 5.3 and an illustrative two-aircraft example is given in section 5.4.

5.1 Bayesian recursive estimator

This section describes the Bayesian recursive estimator and shows how it was used for the multiple target tracking case. The difficulties of implementing the optimal Bayes' solution are discussed. The approximations and assumptions used to develop the MTT framework (MTTF) are clearly stated in this section.

Below is a development of the full BRE under two assumptions, namely sensor independence and the Markov property. Sensor independence means that reports from the sensor at time k depend only on the system's state at time k and not on the previous sensor reports. This is not strictly true for all sensors and is an assumption made by the system designer. The Markov (see definition 1) assumption means that the model assumes that the state is complete. Again, this is a system design choice and an evaluation of the model output against real-world truth data is the only manner in which these design choices can be justified.

Enforcing the Markov property assumption limits the adaptability of the estimator; however, the state vector can be enriched to cater for a particular dimension of adaptability. As an example consider a system that is modelled with a constant velocity while the true physical process has an acceleration component acting on one or more of its states. The estimator will perform poorly as it cannot generate

good predictions from a transition function which does not include the acceleration component. If the specified uncertainty in the motion model is too limiting the estimator will fail outright.

The Bayesian recursive estimator does not require these two assumptions, however the derivation that follows does and it clearly states where in the derivation these assumptions are applied. The goal of the BRE is to compute the posterior distribution of the system's state given all available data. The belief about the system $bel(s_k)$ is the starting point of the derivation:

$$bel(s_k) = p(s_k|z_{0:k}) = p(s_k|z_k, z_{0:k-1}). \tag{5.1}$$

Applying Bayes' rule gives

$$p(s_k|z_{0:k}) = \frac{p(z_k|s_k, z_{0:k-1})p(s_k|z_{0:k-1})}{p(z_k|z_{0:k-1})}. \tag{5.2}$$

Let the probability of evidence $p(z_k|z_{0:k-1}) = \eta$. Applying the sensor independence assumption gives

$$p(s_k|z_{0:k}) = \eta p(z_k|s_k)p(s_k|z_{0:k}), \tag{5.3}$$

which can be expanded based on the total probability theorem. The expansion produces the following expression:

$$p(s_k|z_{0:k}) = \eta p(z_k|x_k) \int p(s_k|s_{k-1}, z_{0:k})p(s_{k-1}|z_{0:k-1}))ds_{k-1}. \tag{5.4}$$

After applying the Markov property assumption and substituting in equation 5.3 the belief about the system's state can be updated by the following recursion:

$$bel(s_k) = \frac{p(z_k|s_k) \int p(s_k|s_{k-1})bel(s_{k-1})ds_{k-1}}{p(z_k|z_{0:k-1})}. \tag{5.5}$$

The recursion is split into a prediction step and an update step. The prediction step is

$$p(s_k|z_{k-1}) = \int p(s_k|s_{k-1})p(s_{k-1}|z_{k-1}))ds_{k-1} \tag{5.6}$$

and the update step is

$$p(s_k|z_k) = \frac{p(z_k|s_k)p(s_k|s_{k-1})}{p(z_k|z_{k-1})}. \tag{5.7}$$

This formulation can be simply extended to the multi-target case. The state vector is extended to $S_k = \left[\mathbf{s}^0, \mathbf{s}^1, ..., \mathbf{s}^{N_t}\right]$ where N_t is the number of targets. The observation vector is extended to $Z_k = \left[z^0, z^1, ..., z^{N_d}\right]$ where N_d is the number of detections. The practical issue that arises for the multiple target tracking (MTT) case is how to specify the multi-dimensional transitions and likelihood densities. The transition density

$$p(S_k|S_{k-1}) = f(S_{k-1}) + \omega_k \tag{5.8}$$

needs to model the interaction between multiple targets, and the likelihood

$$\mathcal{L}(Z_k, S_k) \propto p(Z_k|S_k) \tag{5.9}$$

needs to represent the true multiple target measurement likelihood. The Bayesian recursive estimator does not require an explicit association between detections and targets. It does not even require the thresholding of the observation to form a countable set of detections. All that is needed is the ability to compute the likelihood $p(Z_k|S_k)$ on the full observation and the collective state [33]. It is at this point that the many multiple target tracking approaches make assumptions and proceed in different directions. Firstly the multiple target state transition density and secondly the multiple target likelihood will be discussed.

The most common assumption applied to the multiple tracker state is to treat all trackers independently. This assumption is made in the multiple hypothesis tracking (MHT) and probabilistic data association filter (PDAF) approaches and is utilized in the formulation of this MTTF. This implies that the complete state vector is separated into sub-vectors and these hold the state of each individual target. Each of these individual state vectors are updated independently using $p(s_k|s_{k-1})$. This assumption is valid provided the trackers are not close to one another and the targets do not interact. In chapter 6 the data association stage deals with interacting trackers by specifying a configuration potential.

Approaches for specifying the multiple target likelihood function bifurcate into associative and non-associative methods. The first method thresholds the sensor output to produce a countable set of detections. Then one of the following assumptions are made:

1. each detection can be associated with zero or one tracker per scan, or
2. each detection can be associated with zero or any number of trackers per scan.

The associative method is used in this work and is discussed in detail in chapter 6. The non-associative method does not threshold the sensor output and simply specifies a full likelihood function on the entire sensor output. This is often difficult to specify and is computationally exhaustive. The non-associative approach is commonly used in the track-before-detect paradigm and is put forward clearly by Stone [51]. It is reported to behave better in low signal-to-noise ratio (SNR) environments [33] and deals with merged measurements implicitly. The work on random finite sets (RFS) cited in section 2.4 is a recent non-associative approach to updating the multi-target state using a complete likelihood function.

As mentioned, the multiple target tracking framework developed in this work takes the first of the two approaches just described, namely the associative approach. It is in this decision that the framework departs from the true Bayesian recursive

estimator. Functional similarities are drawn between the complete BRE and the MTTF in section 5.3. The BRE is the optimal estimator, provided one can specify all its components described above. Since this is non-trivial for the real-world MTT airspace monitoring task tackled in this work, what follows is a set of approximations that enable a realizable implementation. This implementation takes the form of a particle filter (PF) described in section 5.2 and an association stage described in chapter 6.

5.2 Particle Filter

This section details how a particle filter is used as part of the solution to the multiple target tracking (MTT) problem. As stated in section 1.1.2 the MTT problem aims to calculate the posterior probability over the collective state vector that contains multiple target state vectors as well as the background model. The collective state vector consists of $\mathbf{S} = \left(\mathbf{B}, \mathbf{s}^0, \mathbf{s}^1, ...\mathbf{s}^{N_t}\right)$ where $\mathbf{s}^j$ holds the state of a single tracker indexed by j and $\mathbf{B}$ holds the state of the background model. The assumption of independent targets and background model made in the previous section holds true and as a result transition densities of each can be updated independently. Section 4.1 describes how the background state $\mathbf{B}$ is updated. For the rest of this section, the background model components are left out so as not to complicate the notation.

The single tracker state $\mathbf{s}^j$ is made up of a set of weighted particles. Thus, for each $\mathbf{s}^j$ we let $\mathbf{s}^j = \left[s^i_{0:k}, \omega^i_k\right]_{i=0}^{N_s}$ and this consistent for all trackers making up the collective state with $j = 0, 1, 2, ...N_t$. The discrete time index is k. Each particle has an index i which indexes into the samples up to the total sample count N_s. Each target has an index j which indexes into the set of targets up to N_t, the total number of targets.

Particle filters approximate probability densities as a set of weighted samples, called particles. A particle is denoted as $\{s, \omega\}$, where s is the location of the particle in the state space $\mathcal{S}$ and ω is the importance or weight of that particle. The particle filter holds the posterior distribution $p(\mathbf{S}|\mathbf{Z}_{1:k})$ over the system state vector as this set of weighted points in the state space. For each subset $\{\mathbf{s}^j\}_{j=0}^{N_s}$ the posterior is described and updated according to these equations:

$$p(\mathbf{s}^j_k|Z_{1:k}) \approx \Sigma_{i=0}^{N_s} \omega^i_k \delta(s^*_k - s^i_k) \tag{5.10}$$

$$\omega^i_k \propto \omega^i_{k-1} \frac{p(Z_k|s^i_k)p(\mathbf{s}_k|\mathbf{s}_{k-1})}{q(s^i_k|s^i_{k-1}, Z_k)}. \tag{5.11}$$

As $N_s \to \infty$ the probability $p(\mathbf{S}_k|\mathbf{Z}_{1:k})$ approaches the true posterior distribution. The importance density is $q(s^i_k|s^i_{k-1}, Z_k)$ from which proposal samples are drawn. There are now a few design decisions to make. Firstly, what importance density to choose and secondly, what resampling algorithm to use. In section 2.3.2 it was stated

that the optimal importance density is [2]

$$q(\mathbf{s}_k|s^i_{k-1}, \mathbf{z}_k)_{optimal} = p(\mathbf{s_k}|s^i_{k-1}, \mathbf{z}_k), \tag{5.12}$$

which gives

$$\omega^i_k \propto \omega^i_{k-1} p(\mathbf{z}_k|\mathbf{s}^i_{k-1}) \tag{5.13}$$

$$= \omega^i_{k-1} \int p(\mathbf{z_k}|\mathbf{s}'_k) p(\mathbf{s}'_k|s^i_{k-1}) d\mathbf{s}'_k. \tag{5.14}$$

This optimal importance density has two drawbacks [13]. Firstly, it requires sampling from $p(\mathbf{s}_k|\mathbf{s}_{k-1}, \mathbf{z}_k)$. Secondly, it needs to be able to evaluate the integral over the new state. It has been shown in [2] that if $\mathbf{s}_k$ is a member of a finite set then the integral becomes a sum and sampling from $p(\mathbf{s}_k|\mathbf{s}_{k-1}, \mathbf{z}_k)$ is possible. The second class has been shown to have an analytic solution for when the observation model is linear. Since neither of these is the case making a suboptimal choice for the importance density is required. In the MTT framework a proposal density equal to the kinematic prior $p(\mathbf{s}_k|\mathbf{s}_k)$ is used. This variation of particle filter is often referred to as a bootstrap filter. The result of such a choice is

$$\omega^i_k \propto \omega^i_{k-1} p(\mathbf{z}_k|s^i_k). \tag{5.15}$$

The next design choice is the resampling algorithm. This combats the impoverishment problem detailed in section 2.3.2. Resampling removes particles with very little weight or importance, and new particles are added in regions of the state space near to existing particles with significant weight. Section 2.3.2 gives reference to existing resampling approaches. Resampling is done when there is a significant portion of the particle population with low weights. The measure of filter degeneracy is commonly done by using an estimate of the effective sample size (ESS) calculated by

$$\hat{N}_{\text{eff}} = \frac{1}{\Sigma^N_{i=1}(w^i_k)^2}. \tag{5.16}$$

The resampling algorithm used in this work is residual resampling (RR). The particle filter algorithm used in the MTTF is given in algorithm 1.

Algorithm 1 Particle Filter - sequential importance sampling with resampling

$\mathbf{s}_{k-1}^{j} \leftarrow \left[s_{k-1}^{i}, \omega_{k-1}^{i}\right]_{i=0}^{N_s}$
for $i = 0 : N_s$ **do**
  – `Draw` $s_k^i \sim p(s_k^i | s_{k-1}^i)$
  –`Assign particle weight` $\omega_k^i = \omega_{k-1}^i p(z_k | s_k^i)$
$\Omega_{total} =$ `SUM` $\left[\omega_k^i\right]_{i=0}^{N_s}$
for $i = 0 : N_s$ **do**
  $\omega_k^i = \frac{\omega_k^i}{\Omega_{total}}$
`Calculate` $\hat{N}_{\text{eff}}$ ▷ see equation 5.16
$\hat{N}_{\text{eff}} = \frac{1}{\Sigma_{i=0}^{N_s}(\omega_k^i)^2}$
if $\hat{N}_{\text{eff}} < N_{\text{threshold}}$ **then**
  – `Resample` ▷ see algorithm 2

In the resampling step a set of N_s particles $\mathbf{s}_k^j$ with varying weights are resampled to form a set of N_s particles $\mathbf{s}_{(R)}^j$ with uniform weights. In general, as long as resampling reduces the variance of the particles it will combat the impoverishment problem of particle-based approaches. The residual resampling algorithm [10] is detailed in algorithm 2.

Algorithm 2 Resampling

1: **procedure** RESIDUAL RESAMPLING
2: $\mathbf{s}_k^j \leftarrow \left[s_k^i, \omega_k^i\right]_{i=0}^{N_s}$
3: $\mathbf{s}_{(R)}^j \leftarrow \left[s_{new}^i, \omega_{new}^i\right]_{i=0}^{N_s}$
4: NumberOfResampledParticles $= 0$
5: **for** $i = 0 : N_s$ **do**
6: ParticleImportanceValue $= \omega_k^i \times N_s$
7: **for** index $= 0 :$ ParticleImportanceValue **do**
8: $\mathbf{sr}^j$[NumberOfResampledParticles] $= \mathbf{s}_k^j[i]$
9: NumberOfResampledParticles $++$
10: **if** (NumberOfResampledParticles $== N_s$) **then**
11: Goto line 14 ▷ re-weight
12: **for** $i =$ NumberOfResampledParticles $: N_s$ **do**
13: $\mathbf{s}_{(R)}^j[i] = \mathbf{s}_k^j[0]$
14: **for** $i = 0 : N_s$ **do**
15: $\omega_{new}^i = \frac{1}{N_s}$
16: *return* $\mathbf{s}_{(R)}^j$

5.2.1 Handling multiple targets

In order to handle multiple targets the posterior distribution must be multi-modal, where each mode describes the state of a target. The particle filter is extended to allow for this by viewing subsets of the state as pertaining to a particular target. Thus the decomposition of $\mathbf{S}_k$ as $\mathbf{S}_k = \left(\mathbf{s}^0, \mathbf{s}^1, ..., \mathbf{s}^{N_t}\right)$ together with a separation of the state dynamics gives rise to a view that is well described as an ensemble of particle filters. This gives the framework power to describe how targets can interact as well as place priors on target configurations. The configurations considered was simply penalizing targets that were close together. Factor graphs were the chosen representation as apposed to directed graphs as they map well to software implementation [1].

To illustrate the framework in its totality and show how it follows the Bayesian recursive estimator (BRE) in principle, the graphical model in figure 5.1 is used. Figure 5.1 describes a joint probability density over both state and observations

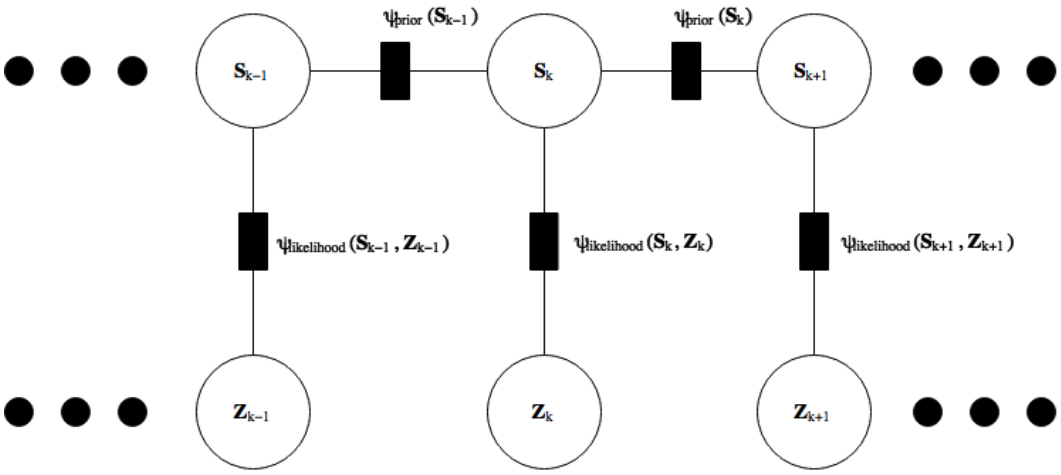


FIGURE 5.1: Factor graph overview.

$p(\mathbf{S}_k, \mathbf{Z}_k) = \eta\psi_{\text{prior}}(\mathbf{S}_{k-1})\psi_{\text{likelihood}}(\mathbf{S}_k, \mathbf{Z}_k)$. The set of detections $\mathbf{Z}_k$ is an observed variable and the posterior probability $p(\mathbf{S}_k|\mathbf{Z}_k)$ is the one of interest, which is also represented by the same factor graph (figure 5.1) up to a scale factor. Loeliger shows that this holds in general and one can pass from a prior model to a posterior model by observing some variables, and that this does not change the factor graph structure [35]. The conditioned posterior, described by the factor graph, is defined as a product of two potentials as

$$p(\mathbf{S}_k|\mathbf{Z}_k) \propto p(\mathbf{S}_k, \mathbf{Z}_k) \tag{5.17}$$

$$= \eta\psi_{prior}(\mathbf{S}_k, \mathbf{S}_{k-1})\psi_{likelihood}(\mathbf{S}_k, \mathbf{Z}_k), \tag{5.18}$$

where η is a normalizing constant. This reflects the posterior probability terms in the Bayesian recursive estimator (BRE). Equation 5.5 is also a product of two factors containing the same terms. Thus, the graph structure proposed in 5.1 is consistent

with the BRE formulation, representing the posterior probability as a product of two terms namely, $posterior \propto likelihood \times prior$. However, the use of a graph has not provided any way to proceed as the factors $\psi_{prior}(.,.)$ and $\psi_{likelihood}(.,.)$ are still not specified and this was simply a way to re-frame the problem.

The state $\mathbf{S}_k$ is made up of N_t target state vectors $\left[\mathbf{s}^0, \mathbf{s}^1, ..., \mathbf{s}^{N_t}\right]$ and the expanded view of a single time slice of the graphical model is illustrated in Figure 5.1.

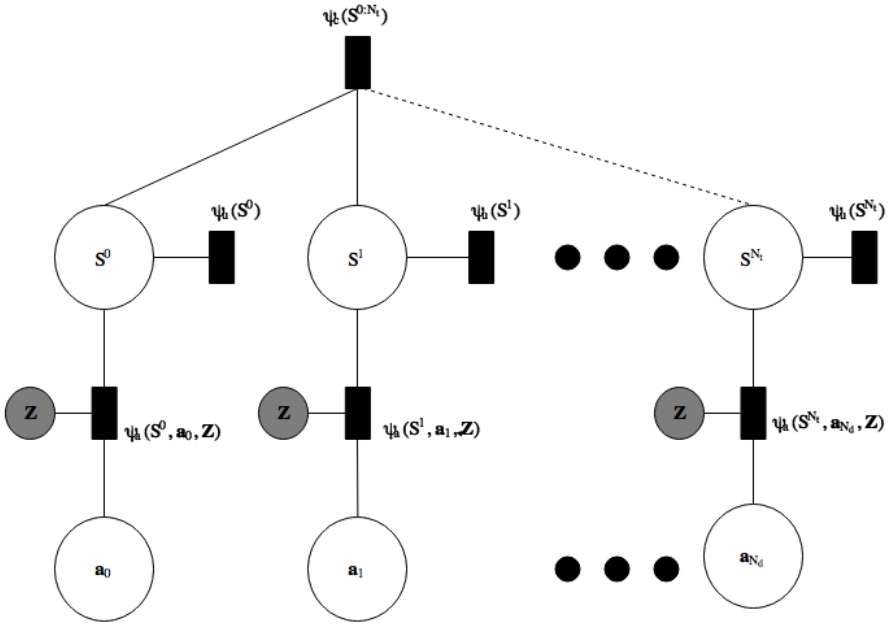


FIGURE 5.2: Factor graph expanded on a single time slice.

The factor graph (figure 5.2) shows how the components of the state vector as well as the components of the observation vector are split and what factors capture these subsets of variables. The posterior is defined as

$$p(\mathbf{S}_k|\mathbf{Z}_k) = \psi_p(\mathbf{S}_k, \mathbf{S}_{k-1})\Pi_{j=0}^{j=N_t}\left[\psi_u(\mathbf{S}^j)\Pi_{m=0}^{m=N_d}\psi_l(\mathbf{S}^j, z^{0:m})\right], \tag{5.19}$$

where $\psi_u(.)$ is the unary potential indicating how probable that portion of the state is. The likelihood potential $\psi_l(.,.)$ is a subset of the state vector given the current observation. The goal is to compute the posterior probability over the state. To get a step closer to this goal one needs to compute these two factors. It is at this stage that further progress requires two assumptions:

1. each subset of the state vector $\mathbf{s}^j$ generates at most one subset of the observed vector $\mathbf{z}^m$, and

2. a subset of the observed vector $\mathbf{z}^m$ can be attributed to many trackers.

Based on these two assumptions, the problem is associating which subset of the state vector generated which subset of the observed vector. This is now framed

as a data association problem and the goal is to find the maximum a posteriori (MAP) assignment of state subset vectors to observation subset vectors. This data association stage is dealt with in detail in chapter 6.

The result from the data association stage is a maximum a posteriori assignment of detections to trackers. This is achieved by defining an assignment vector $\mathbf{a}$, where $\mathbf{a} = [a_u]_{u=0}^{u=N_d}$ and $a_u \in [0 : N_t]$. Thus, $\mathbf{a}[j = 0] = 2$ means that target j caused the subset of observation vector at index 2, namely $\mathbf{z}^{(2)}$.

5.2.2 Birth and death of trackers

This section describes the birth and death of trackers. First the process of birthing a new tracker is described. Thereafter, a description of tracker termination is defined.

In order to estimate the number of targets in a region of interest (ROI), the multiple target tracking framework uses a prior distribution on the number of trackers per unit volume of the ROI. This is a Poisson distribution defined as

$$p_{\text{birth}}(i|\mu_{\text{expected number of trackers}}) = \frac{e^{-\mu}\mu^i}{i!}, \tag{5.20}$$

where μ is the mean number of trackers per unit volume. At each time step (sensor report interval) a new tracker is spawned if a detection is not assigned to an existing tracker and a draw from the Poisson distribution dictates that a new tracker is needed. The tracker is initialized in the local world coordinate frame Gaussian distributed along the ray of the detection. As the distance from the camera cannot be determined the variance along this direction is large compared to the other two dimensions. The mean vector of this distribution is $\vec{\mu}_{\text{birth}}$ and the covariance matrix is Σ_{birth}. Figure 5.3 shows an example Gaussian distribution from which initial particles are drawn.

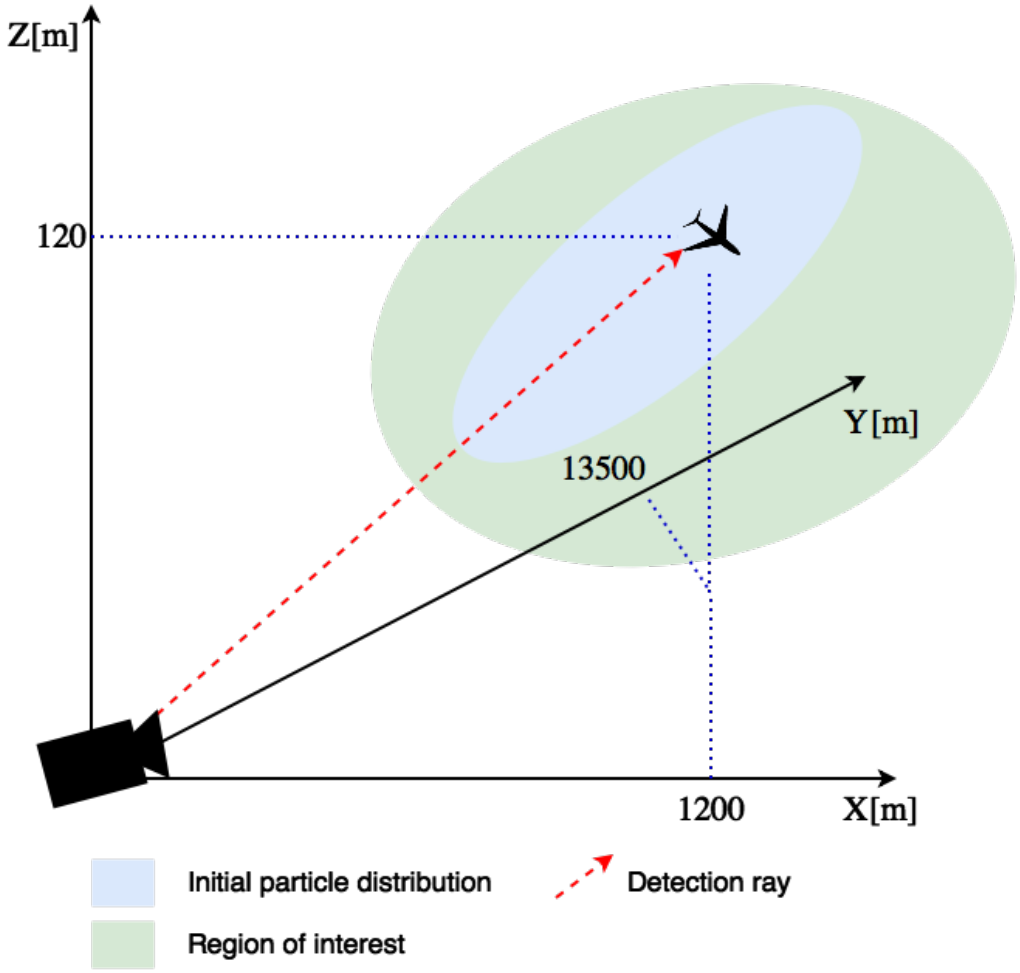


FIGURE 5.3: Initial tracker particle distribution along the detection ray.

The tracker birth process is formalized below.

Algorithm 3 Birth tracker

procedure BIRTH TRACKER
  $i \sim p_{\text{birth}}(N_t | \mu_{\text{expected number of trackers}})$
  if a is not complete **then** ▷ Not all detections are assigned a tracker
    if $i < N_t$ **then**
      $\mathbf{s}_k^{j=N_t} \sim \mathcal{N}(\vec{\mu}_{\text{birth}}, \Sigma_{\text{birth}})$ ▷ Gaussian distributed along detection ray.
      $\left[\omega_i^{N_t} = \frac{1}{N_t}\right]_{i=0}^{i=N_s}$

The termination of a tracker is an important part of any long running multiple target tracking (MTT) algorithm. The MTTF terminates trackers based on the probability of the tracker. A measure of the probability of the tracker is compared to a termination threshold, namely $\Theta_{terminate}$. The following algorithm determines if a tracker is terminated.

Algorithm 4 Terminate tracker

1: **procedure** TERMINATE TRACKER
2: **for** $j = 0 : N_t$ **do**
3: **if** $p(\mathbf{s}_k^j) < \Theta_{\text{terminate}}$ **then**
4: Terminate tracker j

This concludes the description of the individual parts of the MTTF. The next section details the complete multiple target tracking framework and what it outputs.

5.3 Complete MTT framework (MTTF)

This section ties together the whole MTT framework and details what it outputs. The complete MTTF is described below in algorithm 5.

Algorithm 5 multiple target tracking (MTT)

1: **procedure** INITIALIZE MTT
2: `Birthtrackers` ▷ see algorithm 3
3: $S^j \leftarrow \left[s_{new}^i, \omega_{new}^i\right]_{i=0}^{N_s}$
4: **procedure** PREDICT ▷ propagate the particles
5: **for** $j = 0 : N_t$ **do**
6: $\mathbf{s}_k^j = f(\mathbf{s}_{k-1}, \omega)$
7: **procedure** DATA ASSOCIATION
8: `Compute label space probabilities.`
9: `Construct the graph.`
10: `Compute the maximum aposteriori label assignment.`
11: **procedure** UPDATE ▷ Reweight the particles based on the current observation and update the background model
12: **for** $a_{index} = 0 : N_t$ **do**
13: $\omega_k^{a_{index}} = \omega_{k-1}^{a_{index}} p(\mathbf{Z}_k[a_{index}] | s_k^{a_{index}})$
14: `Update the background model B` ▷ see section 4.1.1

It is re-emphasized that all probabilty density functions are held as weighted particles. This means that at each time step k the $p(\mathbf{s}^j)$ is held as a sampled distribution and a number of measures can be drawn from it. The most common measures are the mean and MAP estimates. The mean state estimate is given by

$$\bar{s} = \Sigma_{i=0}^{N_p} \omega_i s_i \tag{5.21}$$

and the MAP estimate by

$$s_{\text{map}} = s_i \ : \ i = \underset{i}{\text{argmax}}\, \omega_i. \tag{5.22}$$

The particles are approximating the full conditional posterior and if it is assumed to be a Gaussian distribution then the variance can be estimated by

$$s^2 = \Sigma_{i=0}^{N_p} \omega_i (s_i - \bar{s})^2. \tag{5.23}$$

This is the variance of the posterior at an instant in time and is different to the filter variance which is determined by running a Monte Carlo (MC) simulation with a number of particle filters and then finding the variance in the mean estimates of all these filters. Depending on the application, the particles might be approximating a multi-modal distribution and so taking the first and second moments (mean and variance) as measures of the system state would not be wise as that implies a maximum likelihood estimate (MLE) estimate of Gaussian parameters.

This concludes the development of the MTT framework (MTTF) which is an approximation of a Bayesian recursive estimator (BRE). It is approximate for the following reasons. Firstly, the posterior distribution is approximated as a set of weighted samples. Secondly, the transition density treats all trackers independently. Although the application explored in this work does not deal with interacting targets it is noted as a step away from the goal of a Bayes' optimal solution. The likelihood function also assumes independence between trackers and an associative approach is taken to evaluate the probability of the observation given the current state. Full details of the association stage is given in chapter 6. The association stage computes a single scan maximum a posteriori (MAP) assignment of trackers to detections and, provided all targets remain far enough away from each other, this will be a good approximation to the Bayes' optimal case. These design choices do move the MTT framework (MTTF) further into the ad-hoc realm of trackers; however, this was done purposefully and the approximations are not too crude. By the assumption which allowed for assigning a detection to multiple trackers the framework can explore various hypotheses given a set of detections. In this way the proposed framework is close to the the multiple hypothesis tracking approach.

The presentation thus far has described how the BRE is used to recursively compute the posterior distribution over the collective state space. The posterior distribution is appoximated by a set of weighted particles and updated by the methods of the PF. A real world system of any use would need to provide the operator with actionable information. Two useful questions are asked of the MTTF: *Is there a target present within the ROI?* and *Where are the targets?* The BRE answers neither of these and so measures about the posterior are required. To answer both of these questions the particle population of each tracker is used to fit a Gaussian distribution. Answering the first question each tracker within the ensemble of trackers, making up the collective state, are queried. A measure of confidence is defined as the norm of the diagonal of the covariance matrix of the Gaussian distribution describing the particle population of each tracker. This score can then be thresholded and used as a decision boundary

for classifying target presence or absence. Now the operator has an adjustable threshold that can be set to achieve the desired false alarm rate regarding this binary decision. The second question regarding target location is answered by simply taking the mean of the distribution which together with the covariance matrix providing a measure of confidence. An example scenario is described next to provide an overview of the steps just described.

5.4 PF — two-aircraft example

This section defines a two-aircraft scenario. It restates the MTTF algorithm in a graphical manner and ties it to a specific scenario. The example is described as a VOT use case but analogies could be made to the passive radar case.

Consider the following example: two aircraft are flying across a scene, at which a camera is pointed. The aircraft fly into and out of the field of view of the camera. The camera makes a set of measurements (images) $\mathbf{Z}$ at a constant frame rate. The task is to track the two targets as they pass through the region of interest (ROI). This scenario is depicted in figure 5.4.

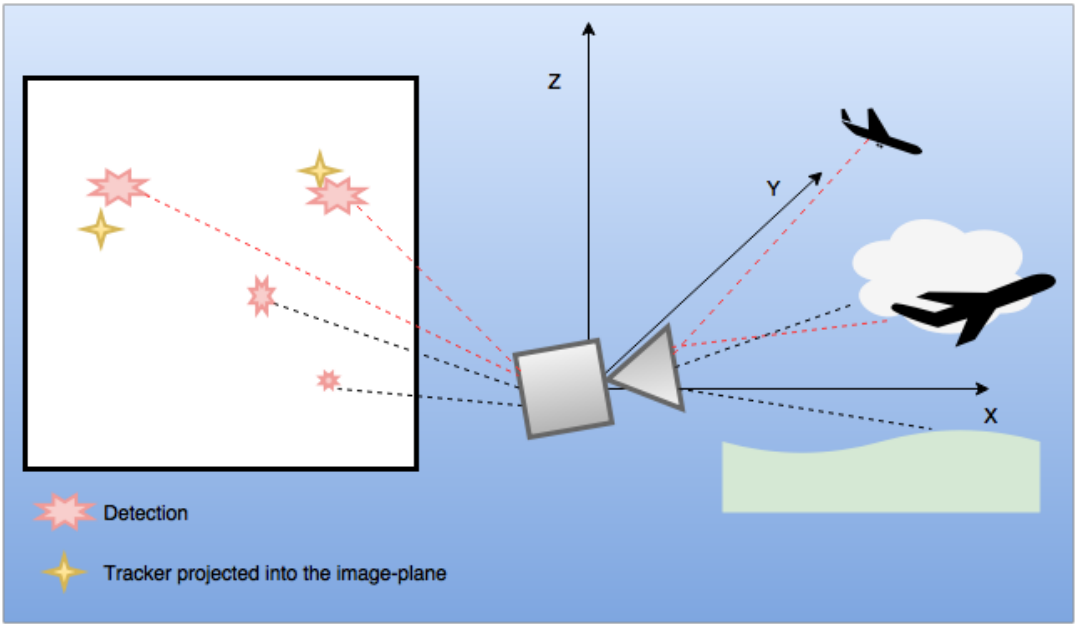


FIGURE 5.4: Two target example captured at time instance k at which there are four detections reported. The state space in local world coordinates is $S = (x, y, z)$and Z is the observation space in image plane coordinates (u, v).

Each pixel value is a measurement along a particular ray originating from some physical object at a moment in time. The Gaussian mixture model (GMM) that represents the background is a generative model. Thus, each pixel value measured by the camera is scored against the model as $p(pixel_{i,j} = BG|pixel_{i,j} = z_{i,j})$. This scoring is explained in section 2.6 and implementation details are given in section

4.1.1. The target state-space models are also generative; however, they generate estimate detection locations ($\tilde{Z}_k$) in the image plane. The measurement model $H(S)$ is used to compute these estimated detection locations by projecting the local world coordinate of each tracker onto the image plane. The target models do not generate pixel values as they do not have an appearance model. The reasons for stripping out the appearance model is given in section 4.2. The task of the PF is to recursively estimate the posterior distribution, $p(\mathbf{S}_k|\mathbf{Z}_k)$. Figure 5.5 shows how the PF holds the posterior distribution as a weighted sum of particles and propagates them according to the motion model.

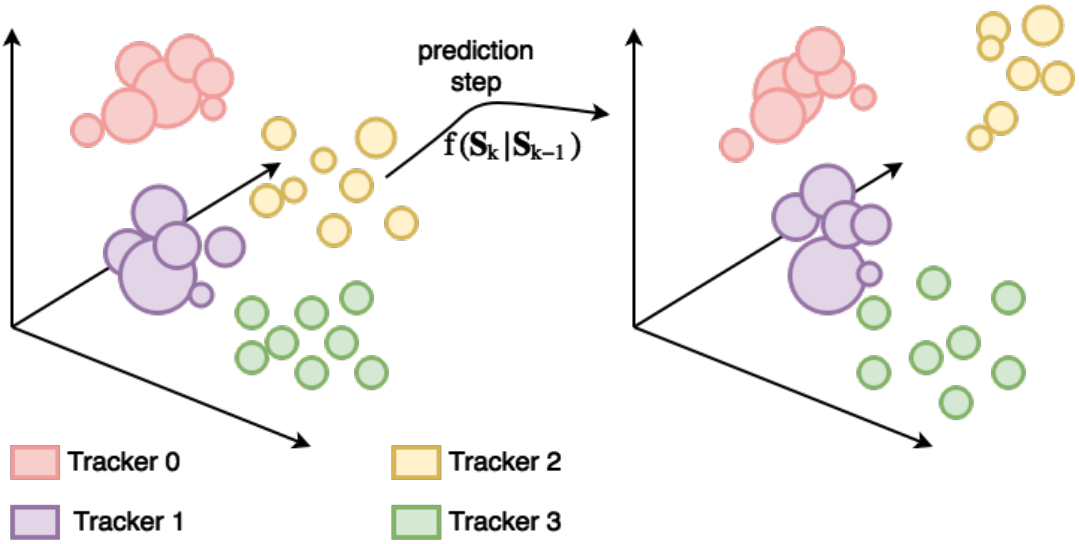


FIGURE 5.5: The particle filter internal representation of the posterior distribution. Each colour represents a different tracker: *Tracker_0*, *Tracker_1*, *Tracker_2* and *Tracker_3*. The size of each particle represents the importance or weight of each particle. The motion model $f(\mathbf{S}_k|\mathbf{S}_{k-1})$ propagates the particles which performs the prediction step. There are four trackers, however the ROI only contains two targets.

The PF estimates the posterior distribution which is the sets of weighted particles. Figure 5.5 shows the posterior with four trackers represented by the four clusters of particles. The particles of trackers 0 and 1 are tightly grouped with large weights and represent peaky distributions whereas trackers 2 and 3 are spread out and represent flatter distributions. A Gaussian distribution can be fitted to each of these particle clusters. The sample covariance matrix represents a measure of confidence a tracker holds about its mean target location estimate. The MTTF can hold a posterior distribution with more modes than there are actual targets. However, the confidence associated with each tracker differs as shown by the particle distributions. By using the norm of diagonal elements of the covariance matrix, $\|\text{diag}(\Sigma_i)\|$, for each tracker a confidence measure can be obtained. This measure is computed and used in section 8.3.4 to generate receiver operating characteristic curves. Figure 5.6 shows the mapping of the sample mean of the particle distribution of each tracker to the measurement

space $\mathcal{Z}$. The detections are also plotted in this measurement space. The data association problem is graphically portrayed in $\mathcal{Z}$ space as trackers need to be associated with a detection. An illustration of the association table generated by the data association stage is used to perform the update step. A resampling step is then applied and the resulting posterior is computed.

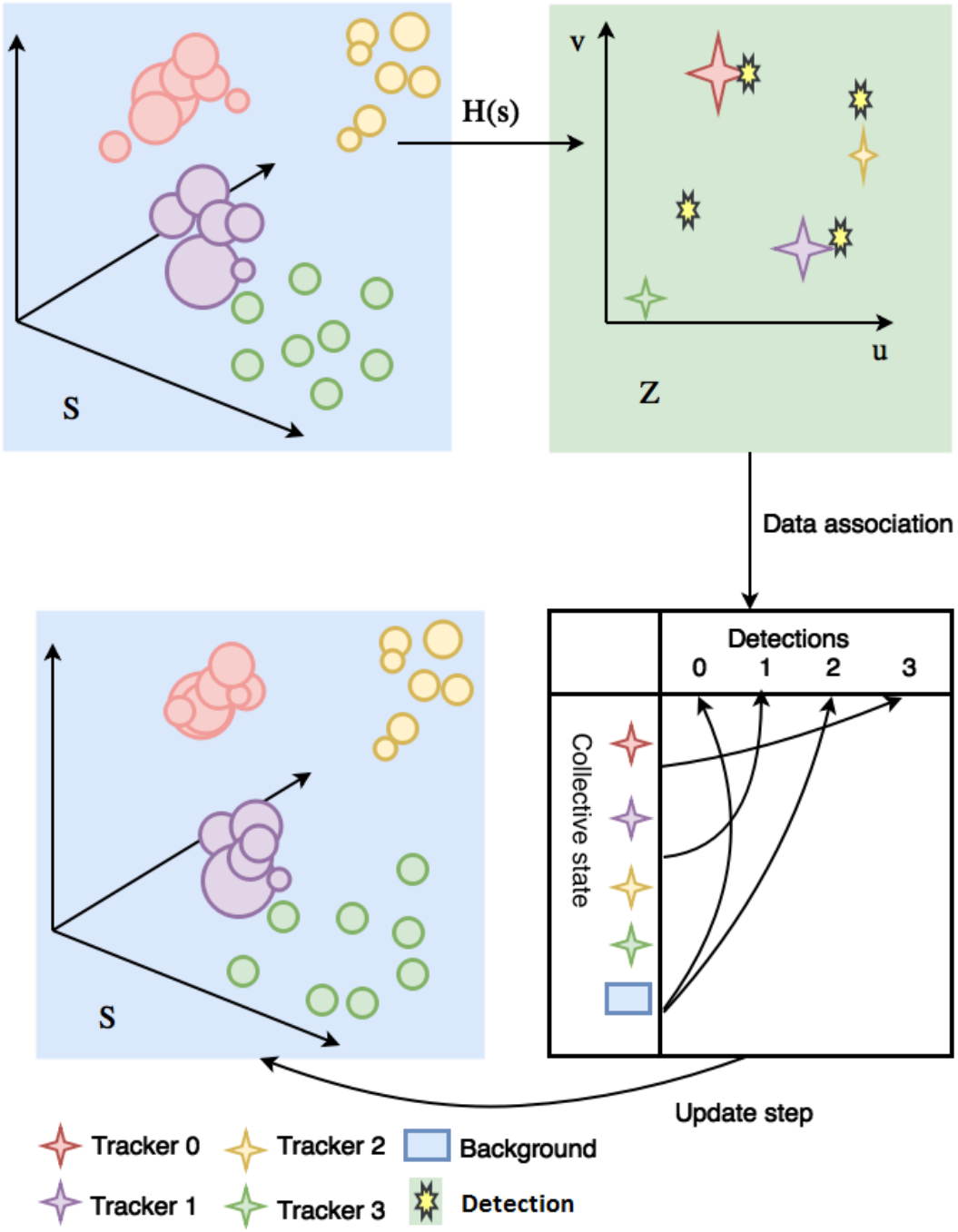


FIGURE 5.6: The particle filter internal representation of the posterior distribution as it is mapped to the measurement space $\mathcal{Z}$. There are 36 detections, 4 trackers and two targets. The data association stage generates an association table as shown. The updated and resampled posterior is depicted in the bottom left of the image.

The two-aircraft scenario showed the MTTF internal representation and how it is updated as observations are made. It is important to observe that the posterior distribution is the output of the BRE — PF in this case. When questions are asked of the MTTF like: *How many targets are in the ROI?* and *Where are the targets?*. Further measures are derived from the particle representation. Namely each cluster is assumed to be Gaussian and the mean is used as the tracker's estimated location. A binary classifier can be applied to the size of the norm of the diagonal of the covariance matrix of each tracker to determine if the tracker is confidently tracking a target. The VOT application aims to provide usable information by answering these two questions and chapter 8 displays these results.

The data association problem has been posed graphically in figure 5.6. It is a critical stage in the MTTF algorithm and is detailed in full in the next chapter.

Chapter 6

Data Association

This chapter tackles the data association problem by means of a probabilistic graphical model (PGM). It is necessary to solve the data association problem because a full likelihood function is difficult to specify and is intractable to compute for the full BRE MTT problem. The likelihood function is broken down into independent factors where each factor takes in an associated pair of tracker state S_i and detection z_j. This chapter draws on the fundamentals and notation put forward in section 2.5. A standard inference algorithm is applied to the proposed graphical structure and a MAP assignment estimate of detections to trackers is determined.

Section 6.1 states the need for a data association stage. It provides a high level interpretation of what the proposed data association approach achieves. It also expands on the a two-aircraft example to clarify various details. Section 6.2 describes the embedding of the state and association vectors into a lower-dimensional space. Section 6.3 shows the resulting factor graph (FG) representation and the potentials that describe the joint probability distribution $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$. The two-aircraft example is elaborated upon and a fully worked solution is given. Section 6.4 provides the details on the probabilistic inference performed on the data association factor graph. Lastly, section 6.5 concludes the association stage and comments on its outcomes.

6.1 Introduction

Data association is the term used to describe the methods of associating an observation or sensor reading, with a process. When modelling the underlying generative process the data association problem is determining which process generated which observed measurement. The need for a data association stage occurs when it is either not possible to specify or not tractable to work with the complete likelihood $p(Z|S)$.

The Bayesian approach to data association is used here, where the goal is not to simply make hard assignments but rather to put a distribution over the possible assignments. Ideally this PDF, the posterior over the states and assignments given the observations $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$, should be incorporated into the update step of the tracker

which holds the posterior PDF over the state of the system $p(\mathbf{S}_k|\mathbf{Z}_{1:k})$. Since this involves computing the joint PDF over the entire set of possible states and possible assignments, the combinatorial complexity explodes and we hit a computational wall. This leads to three necessary assumptions for reducing the dimension of the state and observation vectors.

Assumptions:

1. A detection is generated from a target model $p(\mathbf{S})$ or the background model $p(\mathbf{Z}|pixel)$.

2. A target generates either one detection or none.

3. A detection can be assigned to more than one tracker.

There is no restriction on assigning a detection to multiple trackers. This allows for multiple interpretations of a single detection in cases were there is some ambiguity. The configuration potential $\psi_c(.)$ is used to combat having many trackers locking onto the same target. Trackers that are close together in the state space $\mathcal{S}$ will be subject to this influence on each other and only the most probable one will be assigned detections at this point. By having multiple trackers evolve using the same set of detections for a number of time steps allows multiple hypotheses to be explored before the most probable tracker wins out.

The two-aircraft example described by figures 5.5 and 5.6 has one opaque step — data association. Figure 6.1 provides a high level description of the data association stage for the two-aircraft example.

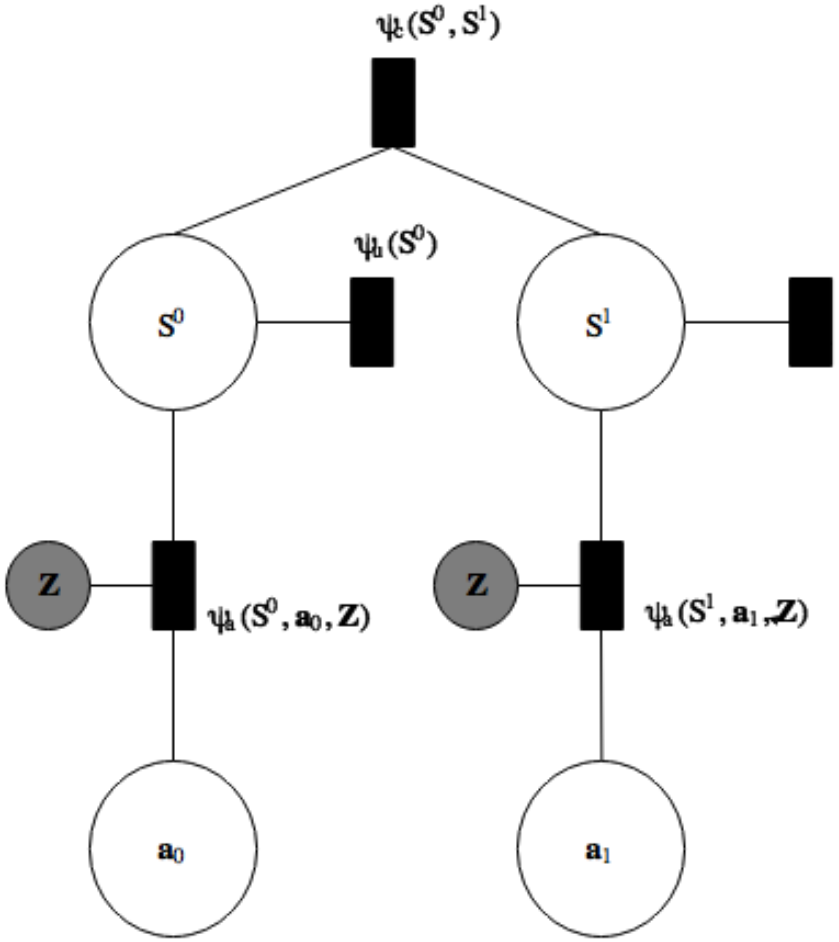


FIGURE 6.1: Data association graphical model for the two-aircraft example.

The FG depicted in figure 6.1 has four variable nodes S^0, S^1 and a_0, a_1 which refer to *tracker_0*, *tracker_1* and association vector a_0, a_1 respectively. There are five factor nodes represented by the five rectangular vertices and these potentials represent factors of the joint distribution $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$. A potential is any non-negative function of its connected variable node arguments. The joint distribution is then defined to be proportional to the product of these potentials,

$$p(S^0, S^1, a_0, a_1) = \frac{1}{Z_{\text{constant}}} \psi_c(S^0, S^1)\psi_u(S^0)\psi_u(S^1)\psi_a(S^0, a_0, \mathbf{Z})\psi_a(S^1, a_1, \mathbf{Z}). \quad (6.1)$$

Since the potentials are over discrete variables they can be defined by a table of non-negative values. Section 6.3.1 provides a worked example. Potentials are not probabilities. They represent the relative "compatibility" between different assignments to the potential [41]. Section 6.3 specifies these potentials completely. The purpose of these factors is in the way we can construct non-negative functions to take on higher values for more probable configurations. The potentials were constructed to assert the following properties in the joint distribution. Trackers close together

are unlikely. Trackers that have a dispersed particle distribution are less likely than ones with tightly clustered particles. Trackers which generate detection estimates far from observed detections are less likely. These three characteristics are encapsulated by the configuration potential $\psi_c(.)$, the unary potential $\psi_u(.)$ and the association potential $\psi_a(.)$ respectively. The remainder of this chapter details how the state and association vectors are embedded into discrete label spaces which allow the tabulation on the potentials and computing of the map assignments.

6.2 Embedding the state and association vectors

In order to cast the data association problem as a discrete graphical model the state and association vectors are embedded into a label space. This label space is discrete and was chosen to have as small a dimension as possible without making further assumptions. Since the observation vector is by definition an observed variable, the data association task is set up as determining a joint probability distribution over the state and association vectors conditioned on the current set of observations. Thus the goal is to estimate $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$. The label spaces are defined by equations 6.2 - 6.2 listed below:

$$\text{(State label space)}\tilde{\mathcal{S}} = [0, 1] = [\text{NotTarget, Target}]\,,$$
$$\text{(Background label space)}\tilde{\mathcal{B}} = [0, 1] = [\text{NotBackground, Background}]\,,$$
$$\text{(Association label space)}\tilde{\mathcal{O}} = [0, 1, 2, ..., N_d]\,,$$

where N_d is the total number of detections. To make notation simpler the label spaces $\tilde{\mathcal{S}}$ and $\tilde{\mathcal{O}}$ are combined into $\tilde{\Theta} = \tilde{\mathcal{S}} \cup \tilde{\mathcal{O}}$. Note that all symbols with a "˜" are defined on or denote a label space. The definition of a detection or contact is given in section 4.1. In a fully Bayesian approach each pixel would be deemed a detection and this association step would assign it to the background model if that is the maximum a posteriori assignment. This would increase the dimension of the association vector $\tilde{\mathbf{a}}$ to $N_d(=$ Number of pixels) and would make the implementation somewhat more difficult and require more computation. Instead the output of the background model is thresholded to produce a set of detections as detailed in section 4.1.1.

The set of pixels that are above the detection threshold make up the observation set $\mathbf{Z}_k = \left[z^0, z^1, ..., z^{N_d}\right]$. This is noted as a suboptimal step as a large portion of the pixels are assigned to the background model without consideration of the plausible target locations which are held by the collective tracker state.

For each target there is a posterior distribution $p(S_k|\mathbf{Z}_{1:k})$ which is held as a set of weighted particles. Each distribution is reduced to a point estimate on a boolean label space. This makes the embedded state space vector $\tilde{S} = \left[\tilde{\mathbf{s}}^0, \tilde{\mathbf{s}}^1, ..., \tilde{\mathbf{s}}^{N_t}\right]$. Associated with each boolean label space state vector is a potential $\psi_u(\tilde{\mathbf{s}^i})$. The definition of these unary potentials is given in section 6.3.

The association vector $\tilde{\mathbf{a}} = [\tilde{a}_0, \tilde{a}_1, ..., a_{\tilde{N}_t}]$ holds the mapping from target to detection. To illustrate how this indexing works, consider again the two-aircraft example in figure 5.4. In this snapshot of the scene at time instant k there are four detections $\mathbf{Z}_k = \left[z^0, z^1, z^2, z^3\right]$. The size of the association vector is N_t and is defined as $\tilde{\mathbf{a}} = \left[\tilde{a}^0, \tilde{a}^1\right]$. Assume that the maximum a posteriori (MAP) label assignment of $\tilde{\Theta} = \left[\tilde{S}^0, \tilde{S}^1, \tilde{a}^0, \tilde{a}^1\right] = [1, 0, 0, 3]$. This would imply that

- $\mathbf{S}^0$ tracker zero is in state $\tilde{S}[1] = Target$.
- $\mathbf{S}^1$ tracker one is in state $\tilde{S}[0] = NotTarget$.
- $\tilde{a}^0$ association index for target zero is zero. Thus, update $\mathbf{S}^0$ with the detection $\mathbf{Z}_k[\tilde{a}^0] = \mathbf{Z}_k[0] = z^0$.
- $\tilde{a}^1$ association index for tracker one is three. Thus, update $\mathbf{S}^1$ with the detection $\mathbf{Z}_k[\tilde{a}^1] = \mathbf{Z}_k[3] = z^3$.

This concludes the description of how the posterior of each tracker (sub-state) vector $p(S_k^i|\mathbf{Z}_k)$ is mapped to the state label space $\tilde{S}$. The association vector is constructed over the association label space $\tilde{\mathcal{O}}$ and the interpretation of the maximum a posteriori (MAP) label assignments were also described. The next section (6.3) shows how the graphical model is constructed.

6.3 Data association cast as a probabilistic graphical model

The following probabilistic graphical model (PGM) describes the data association problem (figure 6.2). The joint probability distribution $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$ is factored into

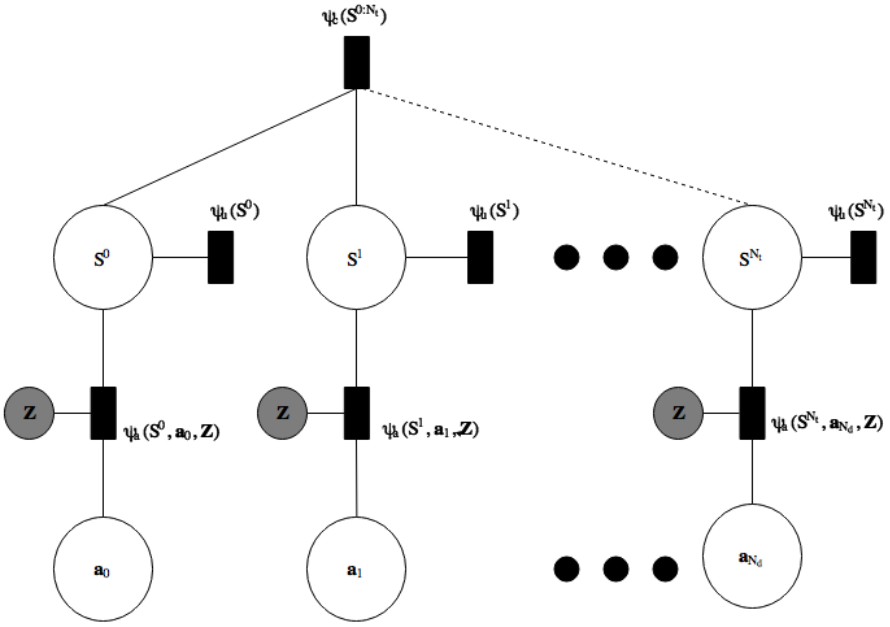


FIGURE 6.2: Factor graph describing the data association problem

three types of potentials. The first is $\psi_u(\mathbf{s}^j)$, the unary potential of variable $\mathbf{s}^j$. This captures the probability of the j^{th} tracker. The second potential is $\psi_a(\mathbf{s}^j, a_i, \mathbf{Z})$, the association potential. This captures the probability of associating the observation $\mathbf{Z}[\mathbf{a}[i]]$ with the j^{th} tracker. The last potential is called the configuration potential, $\psi_c(\mathbf{s}^{0:N_t})$. The plausibility of the target configuration can be inserted into the joint probability distribution via this potential.

The posterior of each sub-state vector is mapped to the prior of a state label on the space $\tilde{\mathcal{S}}$ according to the mapping $I : \mathcal{S} \to \tilde{\mathcal{S}}$. For each sub-state vector the single target component, denoted by a superscript j, the prior over the label space is defined as

$$p(S^j = 1) = \frac{1}{N_s - 1} \exp\left(-\Sigma_{i=0}^{N_s}(\mathbf{s}_i^j - \bar{\mathbf{s}^j})^T(\mathbf{s}_i^j - \bar{\mathbf{s}^j})\right), \tag{6.2}$$

$$p(S_l^j = 0) = 1 - p(s_l^j = 1) \quad \text{and} \tag{6.3}$$

$$\psi_u(\tilde{s}^j) = p(\mathbf{s}^j). \tag{6.4}$$

The factor over state and associations given current observations $\psi_a(S^j, \mathbf{a}_m, \mathbf{Z})$ is defined as

$$\psi_a(S^j, \mathbf{a}, \mathbf{Z}) = \begin{cases} \frac{\exp(-(\|h(\mathbf{s}^j) - \mathbf{Z}[a[m]]\|))}{\sigma_{observation}} & \text{for } S^j = 1 \\ 1 - \frac{\exp(-(\|h(\mathbf{s}^j) - \mathbf{Z}[a[m]]\|))}{\sigma_{observation}} & \text{for } S^j = 0. \end{cases} \tag{6.5}$$

The configuration potential is where the shape of the joint PDF around co-located targets can be manipulated and is defined as

$$\psi_c(\mathbf{S}^{0:N_t}) = \begin{cases} \Pi_{j=0}^{j=N_t}\Pi_{k=0}^{k=N_t}(1 - \exp(-(\|\mathbf{s}^j - \mathbf{s}^k\|))) & \text{for } S^j = S^k = 1 \text{ and } (j \neq k) \\ p(S^j)p(S^k) & \text{for } (j \neq k). \end{cases} \tag{6.6}$$

These potentials form an unnormalized product of factors that describe the true posterior. The factorization stipulated by the graphical model is

$$p(\tilde{\mathbf{S}}_k, \tilde{\mathbf{a}}|\mathbf{Z}_k) = \frac{1}{Z_{constant}}\Pi_{j=0}^{N_t}\psi_u(\mathbf{S}^j)\psi_a(\mathbf{S}^j, \mathbf{a}, \mathbf{Z})\psi_c(\mathbf{S}^{0:N_t}). \tag{6.7}$$

This simple, yet notationally cumbersome, approach is made clearer below in section 6.3.1 by means of continuing the two-aircraft example.

6.3.1 Data association example — two-aircraft

Consider the two-aircraft scenario again in which two aircraft are being tracked in a 3D coordinate system. They each have a population of weighted particles describing the posterior probability of their state. The collective state would be $\mathbf{S} = \left[\mathbf{S^0}, \mathbf{S^1}\right]$. At time k a set of detections $\mathbf{Z} = [z_0, z_1, z_2, z_3]$ are reported.

The data association procedure embeds the illustrated scenario (see figure 5.4) as the following tabulated potential functions. Tables 6.1 and 6.2 hold the tabulated unary potential for *tracker_0* and *tracker_1* respectively.

	$S^0 = 0$	$S^0 = 1$
$\psi_u(S^0)$	0.2	0.8

TABLE 6.1: Tabulated unary potential for S^0, $\psi_u(S^0)$.

	$S^1 = 0$	$S^1 = 1$
$\psi_u(S^1)$	0.3	0.7

TABLE 6.2: Tabulated unary potential for S^1, $\psi_u(S^1)$.

Tables 6.3 and 6.4 hold the tabulated association potentials for trackers 0 and 1.

$\mathbf{a_0}$	$S^0 = 0$	$S^0 = 1$
0	0.3	0.9
1	0.23	0.09
2	0.5	0.039
3	0.32	0.009

TABLE 6.3: Tabulated association potential for S^0, $\psi_a(S^0, a_0)$.

$\mathbf{a_1}$	$S^1 = 0$	$S^1 = 1$
0	0.3	0.01
1	0.23	0.09
2	0.5	0.9
3	0.32	0.009

TABLE 6.4: Tabulated association potential for S^1, $\psi_a(S^1, a_1)$.

Table 6.5 holds the tabulated configuration potential for the two trackers.

	$S^0 = 0$	$S^0 = 1$
$S^1 = 0$	0.01	0.5
$s^1 = 1$	0.3	0.3

TABLE 6.5: Tabulated configuration potential for two trackers S^0, S^1, $\psi_c(S^0, S^1)$.

Probabilistic inference in performed on the graph structure shown in figure 6.2 and the maximum a posteriori (MAP) assignment of labels is found. In this example the maximum a posteriori assignment of labels Θ was $\Theta = [1, 1, 0, 3]$. This is interpreted

as $S^0 = \Theta[0] = 1 =$ Target, $S^1 = \Theta[1] =$ Traget, $\mathbf{a}[0] = \Theta[2] = 0$ which implies that tracker 0 is assigned to detection 0 and $\mathbf{a}[1] = \Theta[3] = 3$ which implies that tracker 1 is assigned to detection 3.

6.4 Inference and MAP estimation

Now the data association problem is an inference problem over discrete spaces where all factors can be represented as a table across all possible variable combinations. Recall that the label assignments are over the association vector $\tilde{\mathbf{a}}$ and the collective state vector $\tilde{S}$. These are referred to as the parameters and given the notation $\tilde{\Theta}$. The computational inference problem to be solved is item 4 of the list in section 2.5.2, the maximum a posteriori (MAP) assignment. Here the maximum a posteriori estimation is computed which uses the mode as a point estimate of the posterior distribution. Ideally, item 3 of the list from section 2.5.2 should be computed as the Bayesian approach calls for the conditional distribution $p(\mathbf{S}, \mathbf{a}|Z) = p(\Theta|\mathbf{Z})$. In order to compute that one needs $p(\mathbf{Z}) = \int_\Theta p(\theta|Z)p(\theta)$, the probability of evidence term. To achieve this requires either conjugate priors, as they allow analytical solutions to the integral, or a well-designed sampling method such as Hamiltonian Monte Carlo (HMC) [5]. The framework departs from the true Bayesian approach here and computes the maximum a posteriori (MAP) assignment as follows:

$$\hat{\theta}_{MAP} = \underset{\Theta}{\operatorname{argmax}}\ p(\theta|Z) \tag{6.8}$$

$$= \underset{\Theta}{\operatorname{argmax}}\ \frac{p(Z|\theta)p(\theta)}{p(Z)} \tag{6.9}$$

$$(keep\ \theta\ terms) = \underset{\Theta}{\operatorname{argmax}}\ p(Z|\theta)p(\theta) \tag{6.10}$$

$$= \underset{\Theta}{\operatorname{argmax}}\ \prod_{z_i \in O} p(z_i|\theta)\ p(\theta). \tag{6.11}$$

The construction of the graph and the MAP assignment computation was implemented using *OpenGM* [1]. Inference was performed using a message passing algorithm called belief propagation. For papers and technical documents that describe this class of algorithms, refer to section 2.5. For specific implementation details, refer to the two application chapters. Refer to chapter 7 for how it was applied to bistatic passive radar tracking and to chapter 8 for how it was applied to visual object tracking.

The initial problem of associating subsets of the observed vector $\mathbf{Z} = [z_0, z_1, ..., z_m]$ to subsets of the state vector $\mathbf{S}$ was solved by proposing a PDF $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$ which could be expressed as a product of factors described by the graphical model depicted in figure 6.2. An optimization algorithm, namely message passing, was applied to the

resulting graphical structure and the maximum a posteriori estimate of the proposed PDF $p(\mathbf{S}, \mathbf{a}|\mathbf{Z})$ was used as the optimal labelling and hence the tracker to detection mapping was determined. The factors were designed to assert three properties which resulted from the assumptions stated in 6.1.

6.5 Conclusion

This concludes the data association approach used in this work. It reformulates the problem in an embedded label space (see section 6.2) and casts it as a probabilistic graphical model as illustrated in figure 6.2. A generic inference algorithm is used to compute the maximum a posteriori assignment estimate as it is the chosen measure taken from the full posterior. The output of the data association stage is a MAP assignment of labels on $\tilde{\Theta}$ which includes both $\tilde{\mathbf{a}}$ and $\tilde{\mathbf{S}}$. This labelling is then used to update the Bayesian recursive estimator (BRE). Chapters 5 and 6 put forward the MTT framework. It strives to adhere to the BRE but practical approximations and assumptions were required and these were clearly stated. The next two chapters (7 and 8) describe two practical applications of the MTTF which validate the approximations and assumptions used in its derivation.

Chapter 7

Passive Radar MTT experiments and results

This chapter outlines the application of the multiple target tracking framework as it applies to passive radar, in particular the bistatic (*Peralex*) passive radar system (PRS). It builds on chapters 5 and 6 and details the particle filter implementation and data association stage. The MTT module was integrated into the existing PRS. Unfortunately a field test with accurate truth data collection was not feasible and so simulated data was generated to test the MTTF. Arguing the usefulness of a basic simulation for validating the MTTF is not the aim of this chapter. The aim is rather to show the components of the MTTF that are problem specific and which are general to a broader class of MTT problems. Simulation experiments were carried out on two different environment configurations and the performance of the MTT module is analysed.

The bistatic passive radar system is reviewed and the simulation environment is detailed in section 7.1. Section 7.2 describes the particle filter (PF) implementation. Section 7.2.4 describes the data association stage in the context of assigning passive radar detections (output of CFAR stage) with trackers. The simulation environment and results and discussion are presented in section 7.3.

7.1 Brief system and simulation environment overview

The (*Peralex*) passive radar system (PRS) is fully described in Tong's PhD thesis [56]. This work appends a multiple target tracking (MTT) module to the output of the constant false alarm rate (CFAR) stage of that system. The following illustration 7.1 is a basic overview of the signal processing pipeline.

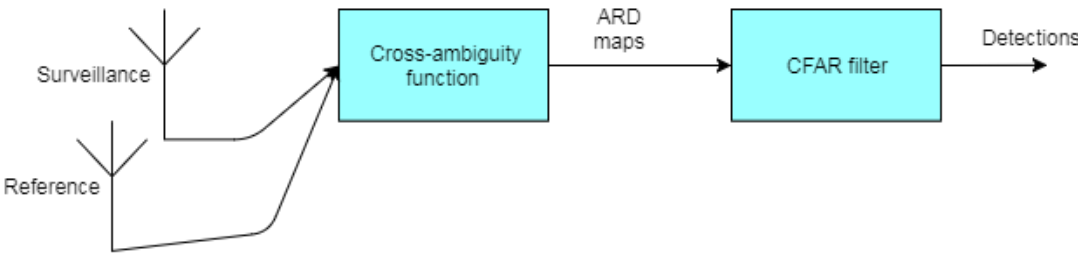


FIGURE 7.1: Passive radar processing pipeline.

The PRS uses a second stage direction of arrival (DOA) module. This enables target localization in a 2D coordinate system as the bistatic range ellipse is intersected by the DOA line. The bistatic range ellipse and DOA line are shown in figure 7.2. The bistatic range is defined as the distance from the transmitter T_x to the target and back to the receiver R_x. The detections sent to the MTT module take the following form $z_i = [r, \dot{r}, \phi]$ where r is the bistatic range, $\dot{r}$ is the bistatic range rate and ϕ is the azimuth. Thus the input to the multiple target tracking passive radar module is a set of detections $Z_k = [z_0, z_1, ..., z_{N_d}]$ where $z_i = [r, \dot{r}, \phi]$. The MTT framework aims to estimate the number of targets and their collective state $S_k = \left[s_k^0, s_k^1, ..., s_k^{N_t}\right]$. As the passive radar system is in a bistatic configuration tracking is limited to a 2D plane. However, the framework does not impose such limitations and the state vector and likelihood function $L(S_k, Z_k)$ can be extended accordingly. Figure 7.2 shows the geometry of the bistatic radar system.

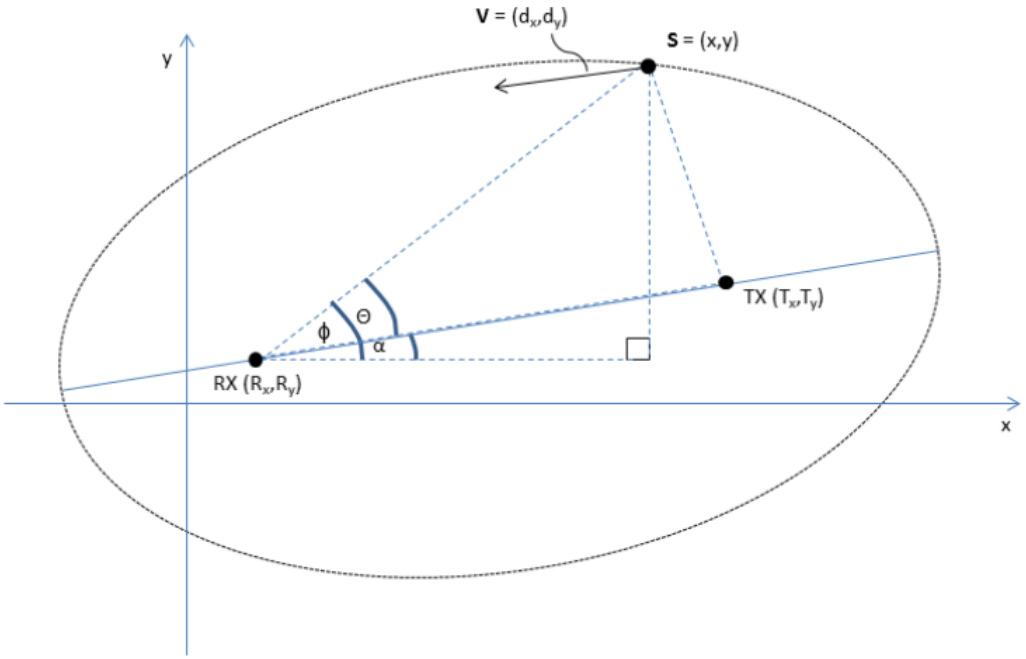


FIGURE 7.2: Geometry of the bistatic passive radar system in a two-dimensional Cartesian plane[17]. A point target is at position $S = (x, y)$ and travelling with a velocity of $V = (d_x, d_y)$. The receiver and transmitter are $R_X(R_x, R_y)$ and $T_X(T_x, T_y)$ respectively. The angle measured by the DOA module is ϕ.

The ellipse shown in figure 7.2 is defined by the $R_X(R_x, R_y)$ and $T_X(T_x, T_y)$ foci and the bistatic range. The DOA is defined as ϕ. The measurement model $h(s)$ which maps $\mathcal{Z} :\longleftarrow \mathcal{S}$ based on figure 7.2 is given in section 7.2.3.

The simulations use the setup described by figure 7.2 and the transmitter and receiver are located 140km apart. The parameters describing the simulation environment are as follows:

- N_t is the number of targets.
- u is the expected number of false alarms per unit area.
- d is the probability of detection.

The false alarm rate is governed by a Poisson distribution $p_{FA}(i|u) = \frac{e^{-u}u^i}{i!}$ and at each scan interval an integer i is drawn from this distribution and that determines the number of spurious detections that are generated uniformly across the ROI. A Bernoulli distribution governs whether or not the true target detection is in the set of detections. For every scan a draw from

$$p_d(b|d) = \begin{cases} d & \text{if } b = \text{target present} \\ 1-d & \text{if } b = \text{target absent} \end{cases} \tag{7.1}$$

is used to indicate the true target's presence or absence in the set of detections passed to the MTT module.

7.2 MTTF passive radar implementation

This section gives the implementation details of the MTTF that were used in the simulations described in section 7.3. This section serves to clarify which aspects of the MTTF are problem-specific and how a practical realization interfaces with the structure of the general framework described in chapters 5 and 6.

Section 7.2.1 details the PF design choices, the state vector and system dynamics, and the measurement model.

7.2.1 Particle filter implementation

As mentioned in section 2.3.2 there are a number of design decisions when implementing a particle filter. The first is choosing a proposal distribution and for this the kinematic prior was chosen. Thus, the weight update recursion becomes

$$\omega_k^i = \omega_{k-1}^i \frac{p(z_k^{a(i)}|s_k^i)p(s_k^i|s_{k-1}^i)}{q(.|s_k^i, z_k^{a(i)})} \tag{7.2}$$

$$\text{where} \quad q(.|s_k^i, z^{a(i)}) = p(s_k^i|s_{k-1}^i) \tag{7.3}$$

$$\omega_k^i = \omega_{k-1}^i p(z_k^{a(i)}|s_k^i). \tag{7.4}$$

The second is choosing a resampling algorithm for which residual resampling was selected; see algorithm 2. The motion model, referred to here as the transition density is $p(s_k^i|s_{k-1}^i) = f(s_k^i) + w$ where $p(s_k^i)$ is the system dynamics and w is the system noise.

7.2.2 State vector and system dynamics

The collective state vector is $\mathbf{S}$ and similar to previously this is composed of tracker states $\mathbf{s}^j$. Equations 7.5 and 7.6 show the state vectors and equation 7.7 describes the associated dynamics. The collective state vector is given by

$$\mathbf{S} = \left[\mathbf{s}^0, \mathbf{s}^1, ..., \mathbf{s}^{N_t}\right] \tag{7.5}$$

$$\text{where} \quad \mathbf{s}^j = \left[x, y, \dot{x}, \dot{y}\right], \tag{7.6}$$

and x and y are the position coordinates with time derivatives $\dot{x}$ and $\dot{y}$. The system dynamics are

$$\mathbf{s}_k^j = A\mathbf{s}_{k-1}^j + w \tag{7.7}$$

$$\text{where} \quad A = \begin{bmatrix} 1 & 0 & \delta t & 0 \\ 0 & 1 & 0 & \delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{7.8}$$

$$\text{and} \quad w \sim N\left(\mu_{\text{sys}}, \Sigma_{\text{sys}}\right). \tag{7.9}$$

The system noise w is additive Gaussian noise with mean μ_{sys} and covariance Σ_{sys}.

7.2.3 Measurement model

At each scan interval a set of measurements are reported. This set is $\mathbf{Z}_k = \{z_k^0, z_k^1, ..., z_k^{N_d}\}$. A detection is related to a target state vector as follows

$$z_k = \begin{bmatrix} r \\ \dot{r} \\ \phi \end{bmatrix} = \begin{bmatrix} D_R + D_T \\ \frac{\Delta R_x \dot{x}_k + \Delta R_y \dot{y}_k}{D_R} + \frac{\Delta T_x \dot{x}_k + \Delta T_y \dot{y}_k}{D_T} \\ \text{atan2}(\Delta R_y, \Delta R_x) \end{bmatrix} \tag{7.10}$$

where D_R and D_T are the distances from the receiver to the target and from the transmitter to the target respectively. The change in the x coordinate from the receiver to the target is $\Delta R_x = R_x - x$ and in the y coordinate is $\Delta R_y = R_y - y$.

This concludes the particle filter instantiation details for the passive radar application. The data association potentials are detailed next.

7.2.4 Data Association Implementation

The full description of the data association stage was provided in chapter 6. The potentials were defined as follows for the bistatic passive radar simulations:

$$\psi_u(s^i) = \frac{N_{\text{eff}}}{N_p}, \tag{7.11}$$

$$\psi_a(s^i, \mathbf{a}_m, \mathbf{Z}) = \exp(-(\frac{\|h(s^i) - \mathbf{Z}[\mathbf{a}[m]]\|}{1000})) \quad \text{and} \tag{7.12}$$

$$\psi_c(\mathbf{S}^{0:N_t}) = \Pi_{j=0}^{j=N_t} \Pi_{k=0}^{k=N_t} (1 - \exp(-(\|\mathbf{S}^j - \mathbf{S}^k\|))) \quad j \neq k. \tag{7.13}$$

The passive radar simulation environment described above shows the aspects of the MTTF which are problem specific. These are the state and associated dynamics as well as the measurement model. The graph structure used in the data association stage remains unchanged. The generality of the MTTF was a design goal and in

this respect it was achieved. The simulated environment is simplistic but remains realistic as a test bed for an MTT module as it fits into the existing system. The following section gives the simulation results with a brief discussion.

7.3 Simulations — results and discussion

All simulations used $N_p = 2000$ where N_p is the number of particles. The results are analysed according to two criteria: position error and a cardinality estimate. The position error is the Euclidean distance between the true target's x, y coordinate and the tracker's mean position estimate. The cardinality estimate is the number of trackers present in the ROI as estimated by the MTTF compared to the number of true targets. It is calculated as follows:

$$E_{\text{cardinality}} = \frac{\Sigma_{i=0}^{N_{\text{scans}}} \mathcal{I}(S)}{\Sigma_{i=0}^{N_{\text{scans}}} \mathcal{I}(S^*)} \tag{7.14}$$

where $\mathcal{I}(.)$ is an indicator function that evaluates to the number of trackers in the sets S (estimated set) and S^* (true set).

Section 5.3 gives an overview of how the presented results relate to the output of the BRE, namely the posterior distribution. The results and discussion of the two simulations are presented below in sections 7.3.1 and 7.3.2.

7.3.1 Simulation 1

The simulation parameters are $N_t = 1$, $u = 0$ and $d = 0.6$. Figure 7.3 shows each tracker's estimated and ground truth trajectory.

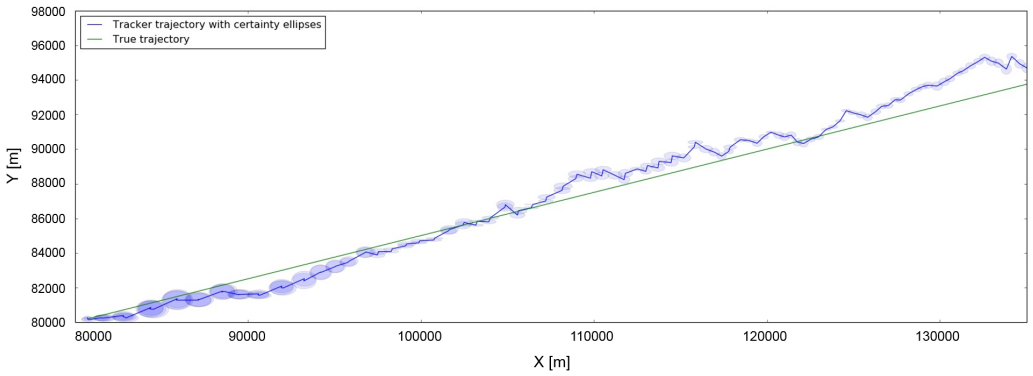


FIGURE 7.3: Tracker trajectory with covariance error ellipse and the ground truth of the target.

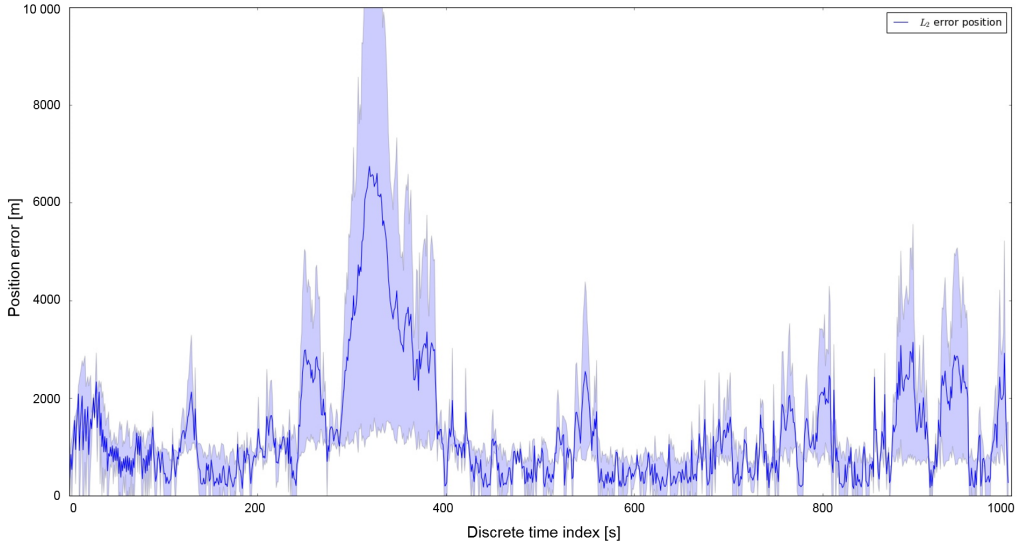


FIGURE 7.4: Position error of the mean estimate from the tracker and one standard deviation band.

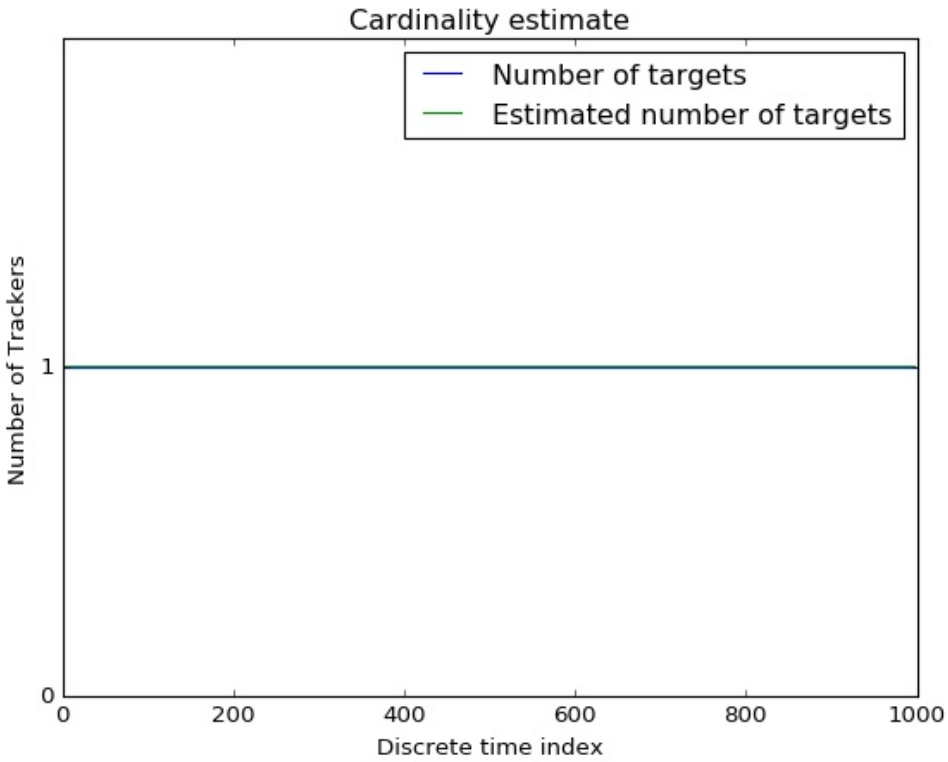
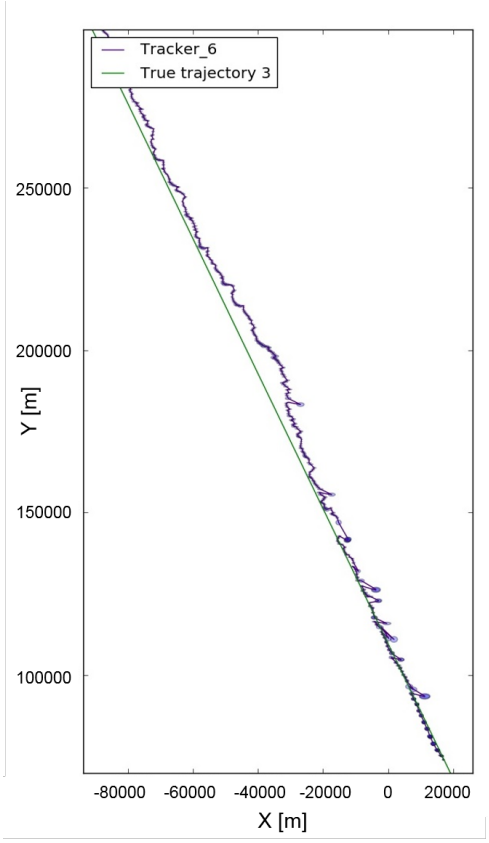


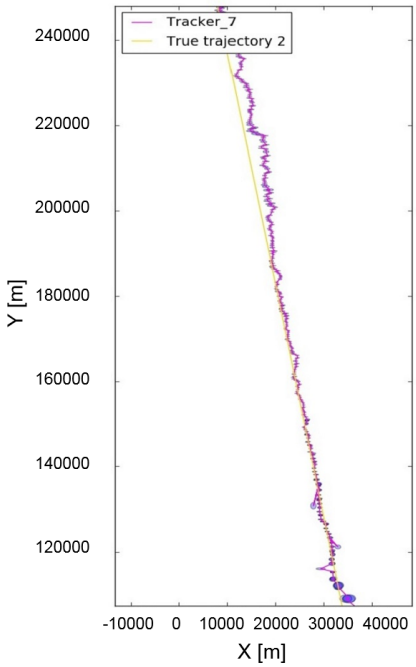
FIGURE 7.5: The cardinality estimate of the MTTF was $E_{\text{cardinality}} = 1.0$.

7.3.2 Simulation 2

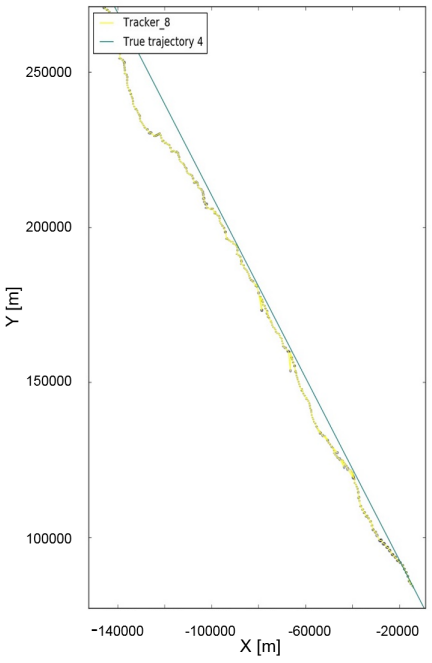
The simulation parameters are $N_t = 5$, $u = 10$ and $d = 0.8$. Figure 7.6 shows each tracker's estimated and ground truth trajectories.



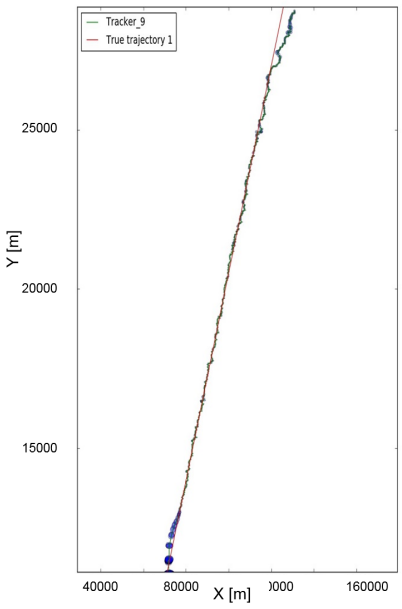
(A) Tracker 6's trajectory with covariance error ellipses plotted against the true target path.



(B) Tracker 7's trajectory with covariance error ellipses plotted against the true target path.

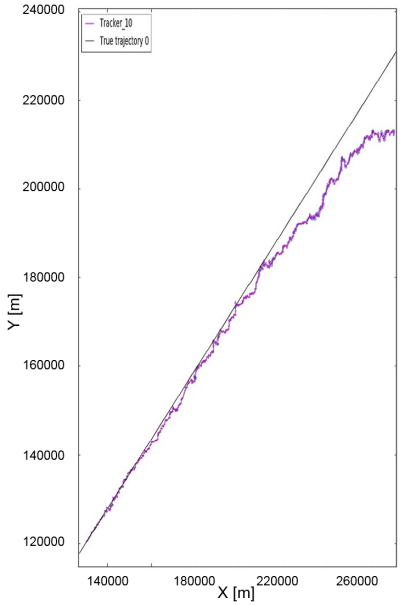


(C) Tracker 8's trajectory with covariance error ellipses plotted against the true target path.



(D) Tracker 9's trajectory with covariance error ellipses plotted against the true target path.

Figure 7.7 shows the position error and first standard deviation band.



(E) Tracker 10's trajectory with covariance error ellipses plotted against the true target path.

FIGURE 7.6: The five simulated targets and associated trackers.

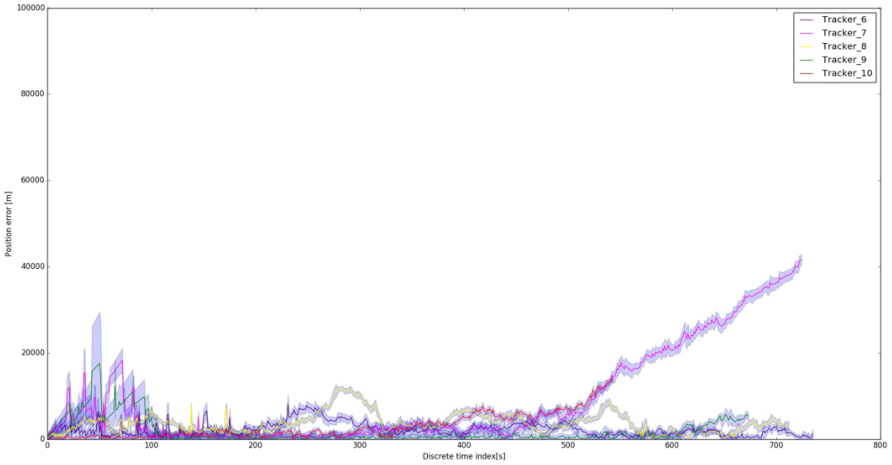


FIGURE 7.7: Position error and the first standard deviation band for all 5 long running trackers.

Figure 7.8 shows the cardinality estimate of the MTT module. The highest number of spurious trackers at any point in time was 3 giving a total of 8 trackers. This then settles down to five trackers and then gets pushed up to 6 until the end of the simulation. This shows that the MTTF does not create excessive numbers of trackers despite the large number of false alarms.

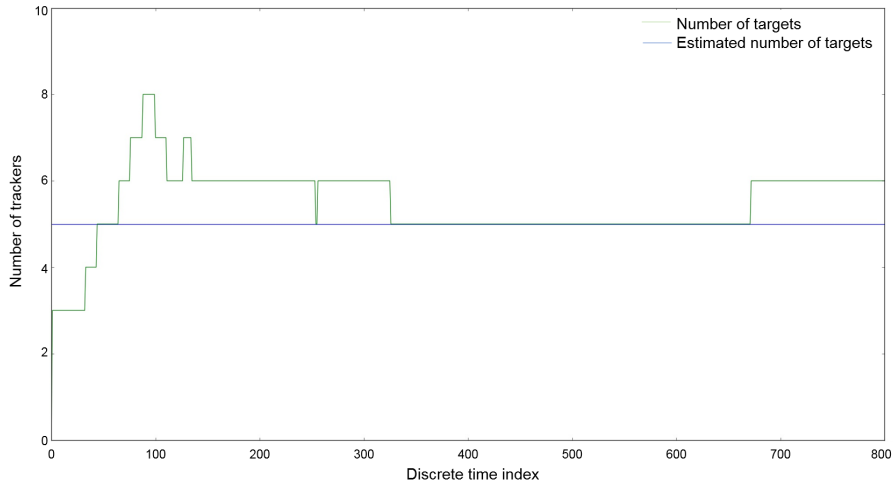


FIGURE 7.8: The cardinality estimate of the MTTF for the second simulation was $E_{cardinality} = 1.1$.

These results show that the MTTF is capable of simultaneously tracking multiple targets. It can birth and destroy trackers as is required by any long term tracking system. These simple simulations show the ease with which the MTTF can be adapted to differing MTT tasks. This is precisely what was done in this investigation whereby the MTTF was originally tested as a module in VOT system and then applied to the passive radar case. The generality of this framework gives confidence to the levels at which abstractions were made. One being the graphical structure used in the data association stage. The structure was determined by the assumptions put forward in chapter 6 and as long as the assumptions hold the graph will accurately describe and solve the data association problem. The other aspect of generality is from a software and system abstractions point of view. The modules and class hierarchies that compose the system are well suited to the MTT task. Though these aspects of the project have a minor role in this report it deserves mention from a practical software and system engineering perspective.

This concludes the first use case application for passive airspace monitoring. The visual object tracking experiment is presented next and it provides more convincing results as it is a complete real world airspace monitoring system.

Chapter 8

Visual tracking experiment

This chapter describes the conditions under which the VOT system was tested and the criteria against which it was evaluated. Section 8.1 gives an overview of the experimental setup and geographic location, and comments on the criteria against which the system will be measured. Section 8.2 gives a summary of all implementation parameter values, describing the state vector, motion model and the factors used in the data association stage. The results of the visual object tracking system on a series of videos are shown in section 8.3. These are discussed in the last section (8.4) of this chapter.

8.1 Overview

The visual object tracking (VOT) system makes use of a single static camera. The camera is mounted at the University of Cape Town and is pointed at the airspace North of the Cape Town international airport. Figure 8.2a shows the scenario layout and a typical view of the region of interest (ROI). The camera is calibrated and the projection matrix was determined using fixed known locations in the scene (see section 3.3 for details).

The Bayesian recursive estimator outputs a posterior distribution. In this work the PF implementation of the BRE holds this as sets of weighted particles. The goal of the BRE is to shift probability mass to regions that have higher probability of having an aircraft. The goal of the VOT system is to alert users to the presence of an aircraft within the ROI and track its location. These are two different goals and this must be carefully addressed when evaluating the system. Figure 8.1 shows the posterior distribution held by the PF.

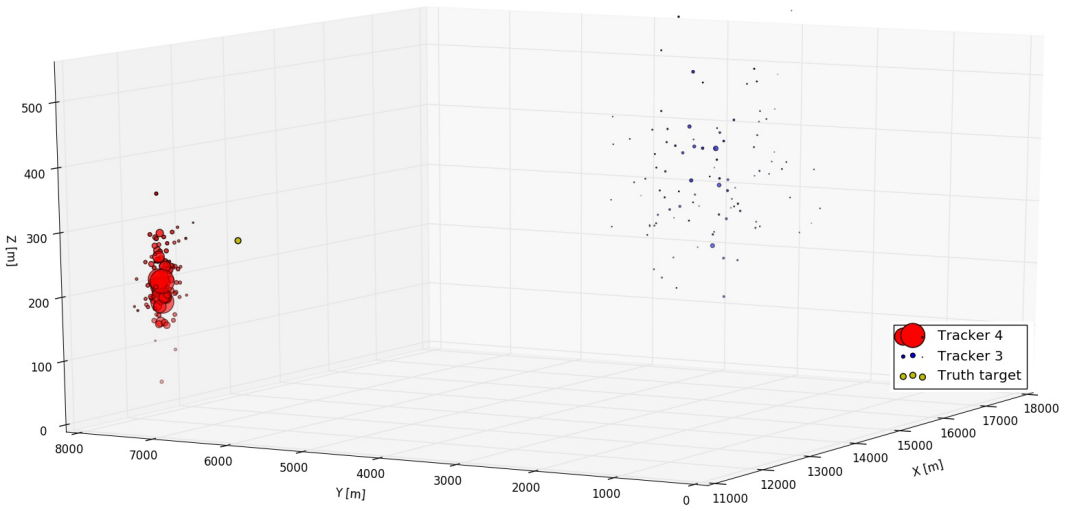


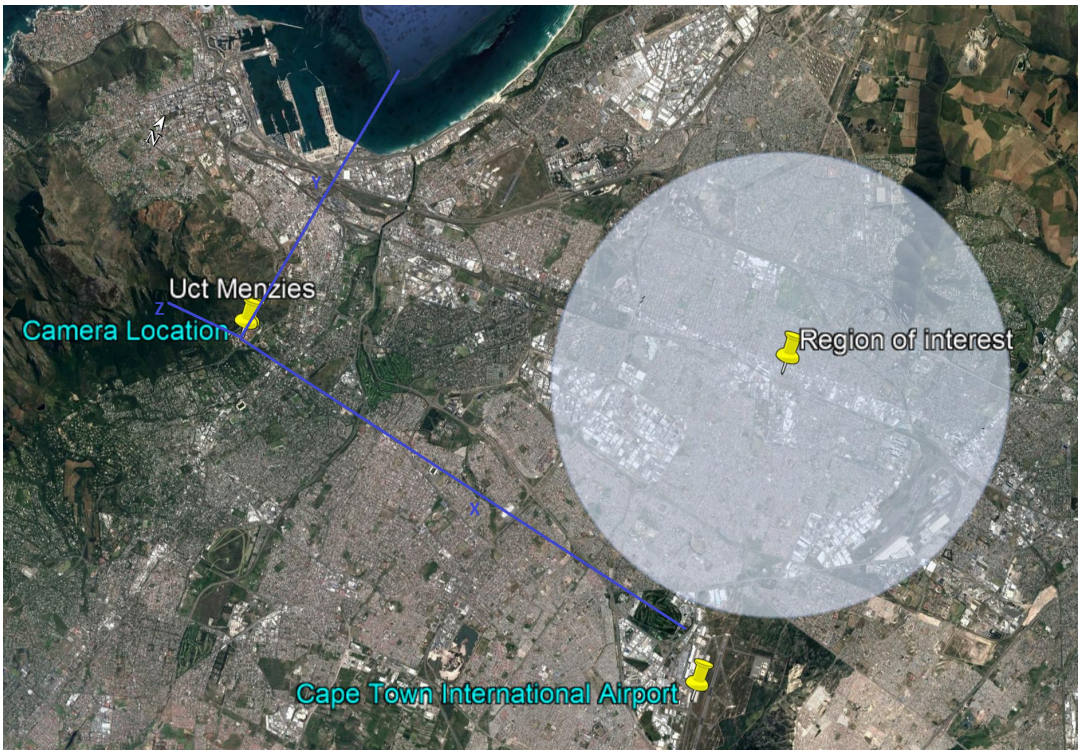
FIGURE 8.1: The posterior distribution as output by the PF.

At this time instant there are two trackers active and a single true target present. The posterior distribution has two modes. The first, *tracker_4*, is peaky and close to the target location. The second, *tracker_3*, is flat. Determining how well this posterior distribution describes the true scene is difficult to score. Further discussion on this is provided in sections 8.3.3 and 9.2.2. The evaluations that follow are from the VOT system point of view. To determine if an aircraft is present within the ROI each tracker is queried as to how confidently it is tracking a target. Thus each tracker is viewed as a binary classifier and ROC curves are plotted. These results are shown in sections 8.3.3 and 8.3.4. When position estimates are quoted for a tracker they were determined by fitting a Gaussian distribution to a set of particles and reporting the mean and standard deviation. With this distinction between the output of the BRE and the desired outputs of the VOT system the results of the experiment are presented.

The tracker is evaluated against four criteria: local world coordinate state estimation, image plane target localization estimation, overall cardinality estimation and as an aircraft detector. The first is done by comparing the tracker's estimate of the target's 3D X_w, Y_w, Z_w local Cartesian coordinates to the truth data obtained from [18]. This will be referred to as the local world frame error. The second criterion measures how well the tracker localizes the target in the image plane. The 3D local world target estimate is projected onto the 2D image plane and this is compared to the true target locations determined by manual annotations. This error will be referred to as the image plane reprojection error. The third criterion measures how well the MTT tracker estimates the true number of targets present in the scene. This is known as the cardinality error $E_{\text{cardinality}}$. The last criterion is how well the VOT system detects the presence of an aircraft and this performance is presented as receiver operating

characteristic (ROC) curves. All four of these criteria are computed for each of the videos recorded from the UCT site.

Below are two images showing the geographic location of the camera, the region of interest and a typical frame from the dataset used to test the system.



(A) The VOT system layout with a North referenced local world coordinate system

(B) Typical frame from the VOT system.

FIGURE 8.2: (A) The visual object tracking system layout showing the camera location at the University of Cape Town and the ROI being the airspace 7.5km North of the Cape Town international airport. (B) A typical frame from the visual object tracking system deployed at the University of Cape Town, South Africa.

8.2 Tracker implementation summary

This section describes the design details of the MTT framework (MTTF) as it is applied to the visual object tracking system. The collective MTT state vector is $\mathbf{S} = \left[\mathbf{B}, \mathbf{s}^0, \mathbf{s}^1, ..., \mathbf{s}^{N_t}\right]$ where B is the background model and s^i is the state of the i^{th} tracker. The state consists of X, Y, Z local world coordinates and their time derivatives. The prior is $p(\mathbf{s}_k^i|\mathbf{s}_{k-1}^i) = A\mathbf{s}_{k-1}^i + w_k$ where A is the constant velocity model

$$A = \begin{bmatrix} 1 & 0 & \delta t & 0 \\ 0 & 1 & 0 & \delta t \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \tag{8.1}$$

and δt is $\frac{1}{23}$[s]. The process noise is modelled as the Gaussian distribution $w_k \sim \mathcal{N}(\mathbf{0}, \Sigma)$ with $\Sigma = \text{diag}(3.38, 3.38, 0.2)$. The likelihood function

$$L(s, Z) = \exp(-(\|h(s) - z\|_2)) \tag{8.2}$$

where $h(s)$ is the projection from the 3D local coordinate frame to the 2D image plane. Th L_2 norm is the Euclidean distance between the tracker's projected location and the measured detection in the (u, v) image plane. The tracker's birth rate is governed by a Poisson distribution $p(i|\mu) = \frac{e^{-\mu}\mu^i}{i!}$ where the expected value is $\mu = 1$ and i is the number of proposed targets in the scene. Trackers are destroyed when their PF holding the approximate posterior distribution degenerates owing to numerous successive missed detections. The data association stage is described in detail in chapter 6. The data association graph structure is shown in figure 6.2. The three potentials whose product estimates the posterior distribution are the unary potential $\psi_u(.)$, the association potential $\psi_a(.,.)$ and the configuration potential $\psi_c(.)$. These potentials were defined as follows for the visual object tracking system:

$$\psi_u(s^i) = \frac{N_{\text{eff}}}{N_p}, \tag{8.3}$$

$$\psi_a(s^i, \mathbf{a}_m, \mathbf{Z}) = \exp(-(\frac{\|h(s^i) - \mathbf{Z}[\mathbf{a}[m]]\|}{100})) \quad \text{and} \tag{8.4}$$

$$\psi_c(\mathbf{S}^{0:N_t}) = \Pi_{j=0}^{j=N_t}\Pi_{k=0}^{k=N_t}(1 - \exp(-(\|\mathbf{S}^j - \mathbf{S}^k\|))) \quad j \neq k. \tag{8.5}$$

The following results are from the multiple target tracking framework performing visual tracking of aircraft.

8.3 VOT results

The truth data for the image plane tracking evaluation was generated by annotating the frames manually. The annotations are a point in the center of the target and this was compared to the tracker's estimate projected onto image plane. The local world coordinate frame performance analysis made use of actual flight data from the aircraft, attained from *Flight Radar 24* [18]. The dataset used for this analysis is made up of one hour of video recordings, composed of twenty-three video files. The results from a single file, namely sequence 08112017_MVI_BA6419, are reported in detail under the relevant subsections. This sequence was not selected for any particular reason but it is indicative of the general result. The results are separated into four sections: image plane re-projection errors in section 8.3.1, local world frame localization errors in section 8.3.2 and cardinality errors in section 8.3.3. The last performance metric is the receiver operating characteristic (ROC) curve and the associated area under the curve (AUC) score which is discussed in section 8.3.4. This measure indicates how reliable the VOT system is at detecting aircraft within the region of interest. An analysis of the results on the entire dataset are presented in section 8.3.5 and the "best" and "worst" performing sequences are analysed further.

8.3.1 Reprojection error

The reprojection is the Euclidean distance in the image plane, measured in pixel units, between the true location of the target and the projected estimated location of the target from the VOT system. The reprojection error for the 08112017_MVI_BA6419 sequence is shown in figure 8.3.

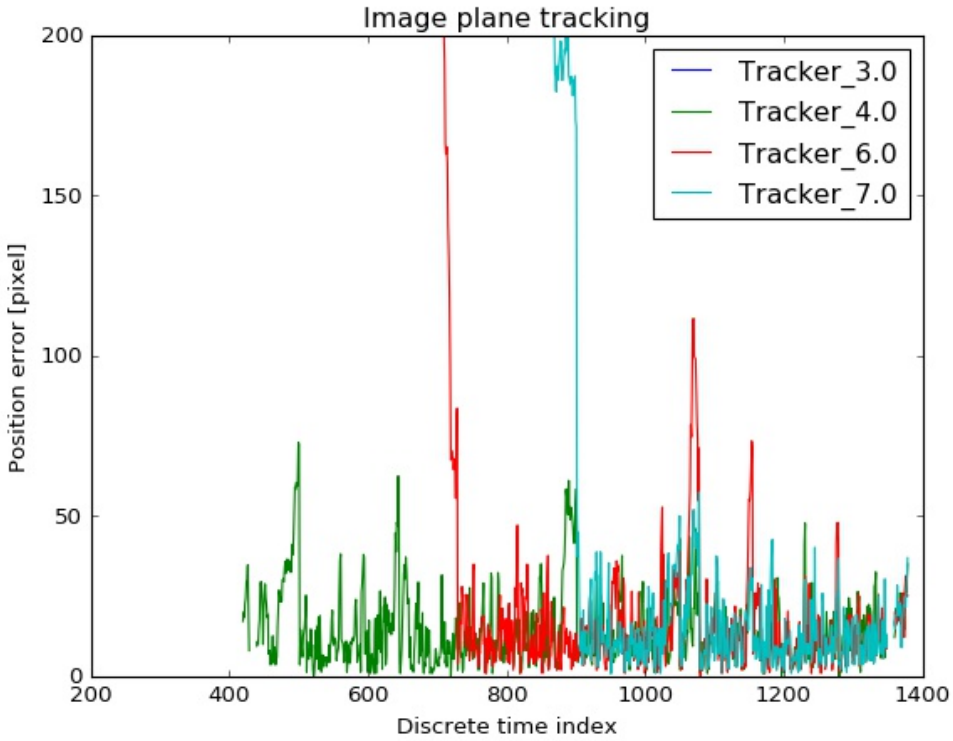


FIGURE 8.3: The re-projection error for the sequence 08112017_MVI_BA6419. Tracker 4 is the longest persisting. Trackers 6 and 7 converge to the same image frame positions. The mean and standard deviation error for each tracker is as follows:
tracker $3(\mu = 623.2\,\text{pixel}, \quad \sigma = 131.6\,\text{pixel})$,
tracker $4(\mu = 27.1\,\text{pixel}, \quad \sigma = 73.5\,\text{pixel})$,
tracker $6(\mu = 18.6\,\text{pixel}, \quad \sigma = 25.8\,\text{pixel})$,
tracker $7(\mu = 91.9\,\text{pixel}, \quad \sigma = 135.8\,\text{pixel})$.
The missing trackers were birthed erroneously and attempted to track false alarms. This characteristic of the MTT algorithm is captured by the cardinality metric.

For all video sequences similar results were produced. The average mean reprojection error across all sequences was $\mu_{\text{all sequences}} = 40.3\,\text{pixels}$ where the image dimensions are 1920×1080. Trackers are allowed to converge to the same image-plane location as the association stage assumes the same detection can be assigned to many trackers. The configuration potential in the graphical model assigns low scores to trackers that are nearby in the state space $\mathcal{S}$. The projective transform clearly allows for separation in the state space and closeness in the measurement space $\mathcal{Z}$.

8.3.2 World frame localization errors

The local world frame position estimate, mean and one standard deviation band for X, Y, Z are shown in figures 8.4, 8.5 and 8.6 respectively. The mean estimate together with the truth aircraft trajectory are shown in figure 8.7.

World frame localization is the tracker's output in a world frame coordinate system. There is the local world coordinate frame and the world coordinate frame, see chapter 3 for details. The longest persisting tracker in the BA6419 sequence is tracker 4. The local world frame estimates and one standard deviation band for tracker 4 are shown below. The tracker outputs its local world frame position estimate with a confidence band of one standard deviation. This is shown in figures 8.4, 8.5 and 8.6.

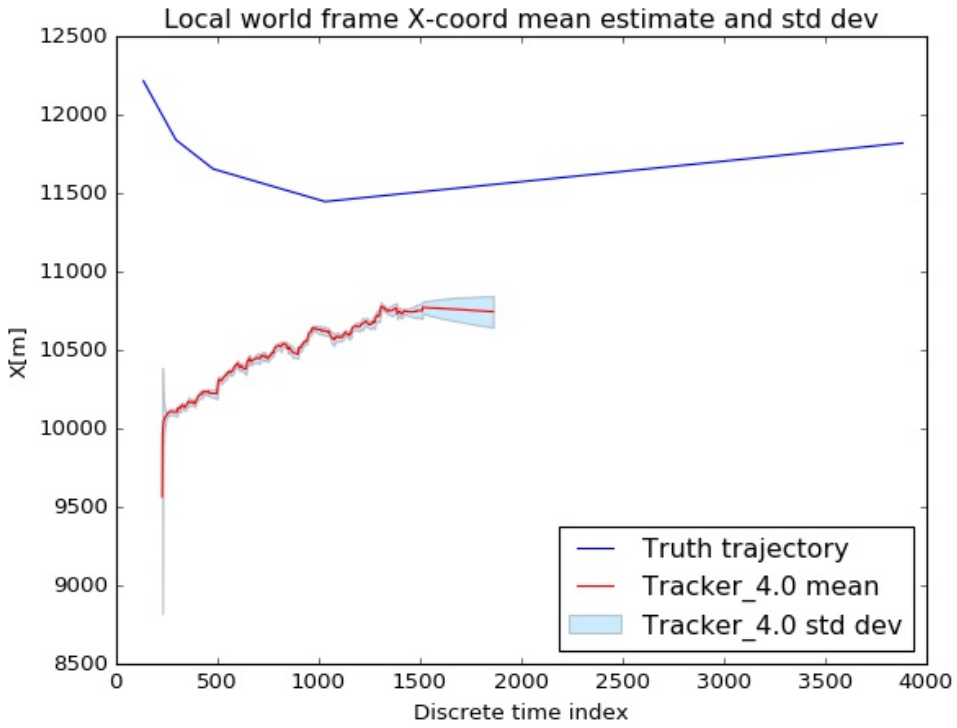


FIGURE 8.4: Local world frame X position estimate for sequence 08112017_MVI_BA6419.

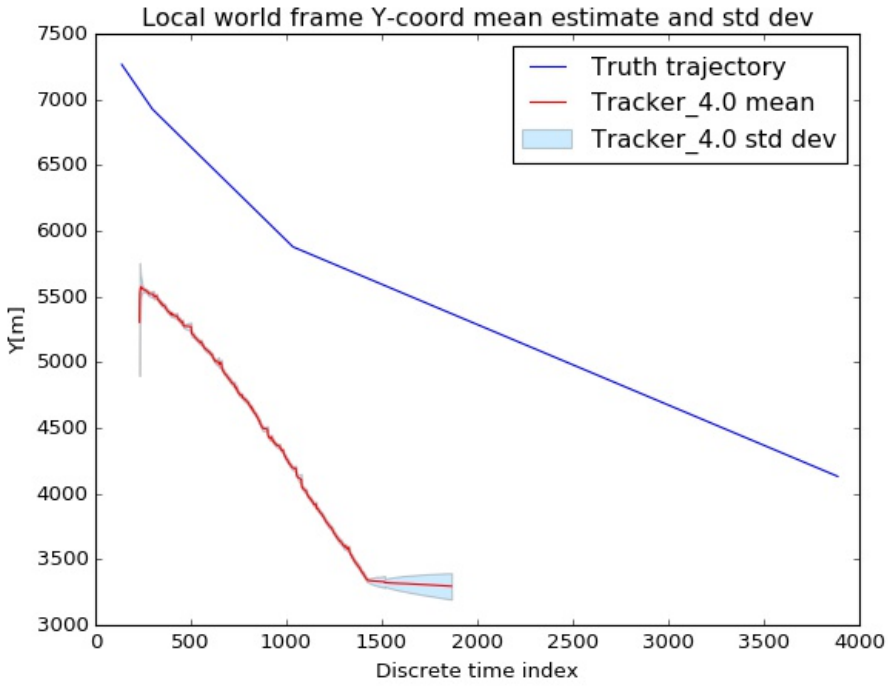


FIGURE 8.5: Local world frame Y position estimate for sequence 08112017_MVI_BA6419.

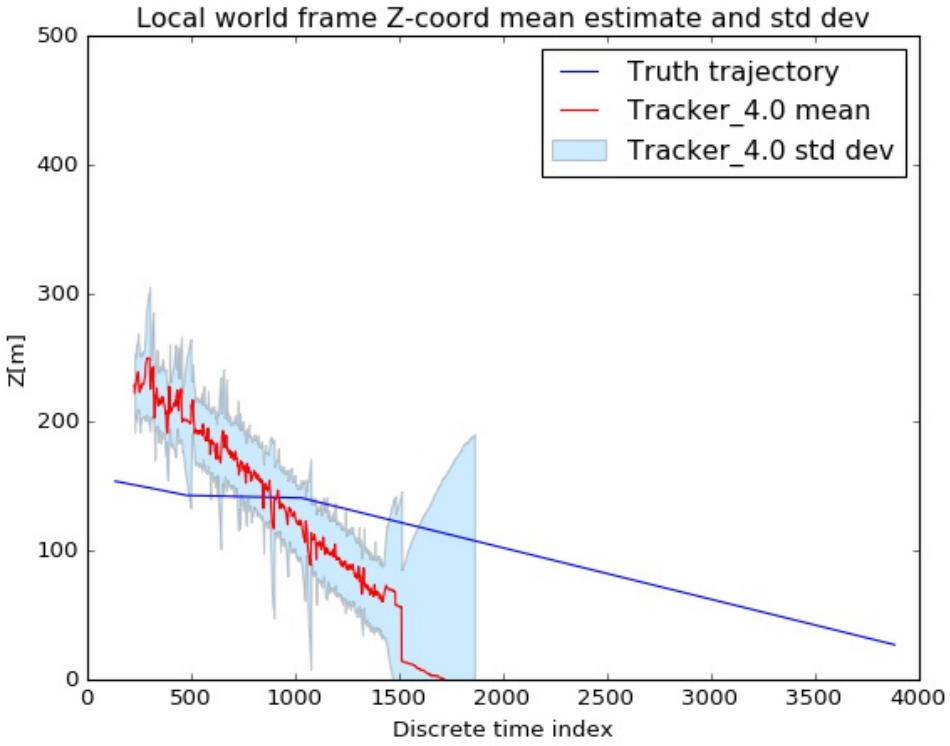


FIGURE 8.6: Local world frame Z position estimate of aircraft in 08112017_MVI_BA6419 by *Tracker_4*. After frame 1356 the aircraft exited the FOV and the tracker was no longer given detections (at least ones that supported the current posterior) and as a result the entries in the covariance matrix get larger. This occurs in all three spatial dimensions as depicted here and in figures 8.4 and 8.5.

The average standard deviation for tracker 4 on the BA6419 sequence is $\bar{\sigma}_X = 50.6\,\mathrm{m}$, $\bar{\sigma}_Y = 41.5\,\mathrm{m}$, $\bar{\sigma}_Z = 75.6\,\mathrm{m}$. On all video sequences the average standard deviation of the filter's estimate tracking a true target is $\bar{\sigma}_X = 180.6\,\mathrm{m}$, $\bar{\sigma}_Y = 52.5\,\mathrm{m}$, $\bar{\sigma}_Z = 43.3\,\mathrm{m}$. Figure 8.7 shows the error between the filter's estimate and the aircraft's true trajectory.

The VOT system's results are now compared to the true flight trajectories. Figure 8.7 shows the mean estimate of tracker 4 and the true aircraft (BA64119) trajectory. The average mean errors are $\bar{\mu}_{X_{truth}-X_{est}} = -1250\,\mathrm{m}$, $\bar{\mu}_{Y_{truth}-Y_{est}} = -1942\,\mathrm{m}$, and $\bar{\mu}_{Z_{truth}-Z_{est}} = -8.3\,\mathrm{m}$ between tracker 4's mean estimate and the true trajectory.

FIGURE 8.7: Aircraft BA6419's true trajectory and *Tracker_4's* mean estimate plotted on the local world frame.

The average local world frame mean estimate errors are $\bar{\mu}_{X_{truth}-X_{est}} = 3421.5\mathrm{m}$, $\bar{\mu}_{Y_{truth}-Y_{est}} = 2457.0\mathrm{m}$, $\bar{\mu}_{Z_{truth}-Z_{est}} = 20.0\mathrm{m}$. The final

output of the VOT system is the mean position estimate of the tracker converted to global positioning system (GPS) coordinates. Figure 8.8, shows a Google Earth plot of the visual object tracking system's aircraft position estimation in the context of the whole scene.

FIGURE 8.8: Aircraft BA6419's trajectory and *Tracker_4's* estimated location plotted on Google Earth. The true trajectory is the continuous green to blue line which enters and exits the ROI and terminates at Cape Town international airport. The estimated locations are the discrete green circles that begin and end within the ROI.

8.3.3 Cardinality estimates

The cardinality estimates for the BA6419 sequence are depicted below in figure 8.9. This figure shows how many targets the VOT system estimates there are present in the ROI, and the assigned *Tracker_ID*. The true number of targets in the ROI during this experiment was one. Similar results were generated for all video sequences. The cardinality estimate is generally an overestimate. The average number of spurious or extra targets estimated by the system across all sequences was found to be 2.7. The cardinality score is calculated as follows:

$$E_{\text{cardinality}} = \frac{\Sigma_{i=0}^{N_{\text{frames}}} \mathcal{I}(S)}{\Sigma_{i=0}^{N_{\text{frames}}} \mathcal{I}(S^*)} \tag{8.6}$$

where $\mathcal{I}(.)$ is an indicator function that evaluates to the number of trackers in the sets S (estimated set) and S^* (true set).

FIGURE 8.9: Cardinality estimate for sequence BA6419. This image shows the lifetimes of the various trackers. The associated error is computed as the ratio of the sum of the number of frames for which each tracker is present to the number of frames the true target is present. For this sequence the cardinality error $E_{\text{cardinality}} = 2.7$.

This measure is somewhat naive as it does not ask the system how confident it is that all trackers are actively tracking a target. The “best" way to see how well a MTT algorithm is performing would be to determine the Wasserstein metric [53] (earth mover’s distance) between the true posterior and the posterior held by the PF. The Wasserstein metric is computed as how much probability mass needs to be moved to go from one distribution to another and is thus a measure of closeness between distributions. This means that many targets with little probability mass should not be too harshly penalised when judging the performance of a MTT algorithm. Further comments on the evaluation module and the evaluation of MTT algorithms in general are made in section 9.2.2.

To determine how well the system “detects" aircraft each tracker is queried as to how confidently it is tracking an actual target. This produces a detection score for each tracker for every sensor scan interval and is presented in the next section as receiver operating characteristic curves and associated area under the curve (AUC) scores, as is typically done to characterise a detector.

8.3.4 Receiver operating characteristic (ROC) and area under the curve (AUC)

The ROC curve is used to measure the performance of a binary classifier, or detector, as the detection threshold is varied. In this work the detection stage is very poor and the raw detections provide very little information. The MTT module processes these detections using the data association stage and collective state particle filter. To illustrate how reliable the VOT system is at detecting a target the ROC curves and AUC score are shown. At each time instant in the sequence the tracker is queried as to how likely it is to be tracking an actual target. This score is the norm of the diagonal of the covariance matrix associated with that tracker’s posterior distribution. Thus, as the particles group together after receiving sequential detections that support the tracker’s motion model, the sample covariance of the PF decreases. This assumes that the posterior is a Gaussian distribution (discussed in chapter 5) and the covariance matrix describes the concentration of the probability mass. The annotated frame’s truth data is used to decide if the tracker is in fact locked onto an actual target.

The receiver operating characteristic curve and associated area under the curve were computed for each video sequence. Figure 8.10 shows the ROC curve for *tracker_4* during this sequence. The AUC was significantly greater than 0.5, which a random

decision mechanism would produce. The false positive rate (FPR) and true positive rate (TPR) are defined as

$$\text{FPR} = \frac{\text{FP}}{\text{FP} + \text{TN}}, \tag{8.7}$$

$$\text{TPR} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{8.8}$$

where FP are false positives, TN are true negatives, TP are true positives and FN are false negatives. These rates are computed for a varying threshold value applied to the tracker's output score and the resulting plot is the ROC curve.

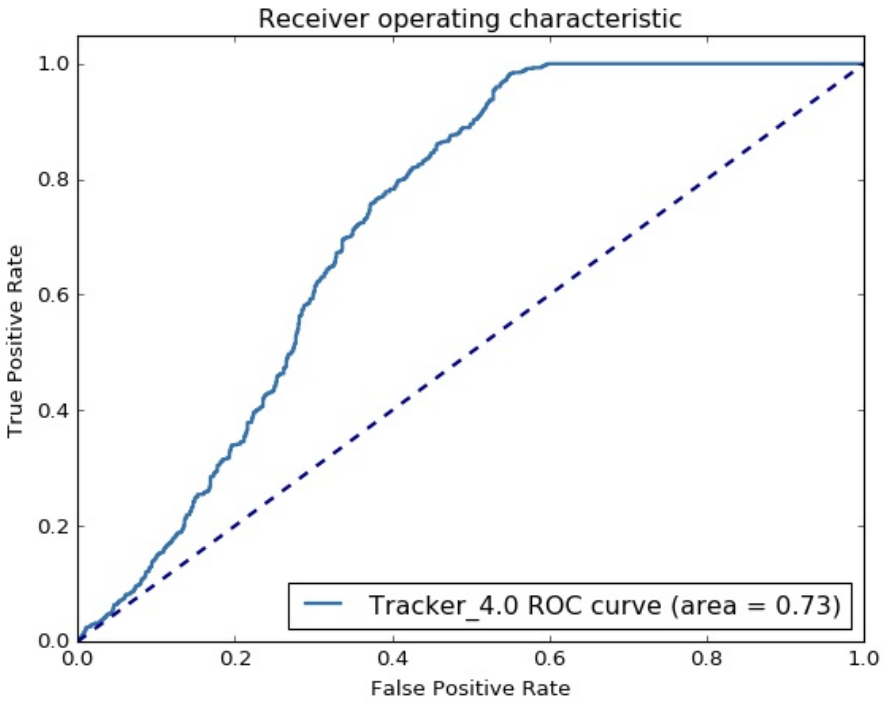


FIGURE 8.10: The receiver operating characteristic (ROC) curve and associated area under the curve (AUC) score for sequence BA6419.

The ROC curve shows that by varying the detection threshold of the tracker's score the system can provide reliable alerts as to when an aircraft travels within the region of interest. A curve that lies above the diagonal dotted line is useful. It shows that the system performs better than random guessing and the threshold can be set to achieve any performance characteristic that lies on the ROC curve. Similar results were obtained of all sequences and the mean AUC was 0.77 with a standard deviation of 0.13.

The results presented above pertain to a specific sequence namely, 08112017_MVI_BA

6419. These results are indicative of how the system performed over the entire dataset. Table 8.1, shows a summary of all of the results.

8.3.5 Summary of results

This section provides a summary of the VOT system's results as a whole in table 8.1. Some of the results are then put into "best" and "worst" performing groups. The sequences that make up each of these groups are then analysed and presented as what optimal conditions are and what modes of failure exist in the current system.

Seq. name	$\bar{\mu}_{X_{truth}-X_{est}}[m]$	$\bar{\mu}_{Y_{truth}-Y_{est}}[m]$	$\bar{\mu}_{Z_{\text{truth}}-Z_{\text{est}}}[m]$	$\bar{\mu}_{\text{re-proj. error}}[pixel]$	Cardinality error	AUC
08112017_MVI_BA6419	-1250	-1942	-8.3	13.3	2.7	0.73
08112017_MVI_1605	-3100	-1446	22	25.6	1.5	0.90
08112017_MVI_1607	-2523	-2409	-71.8	19.2	2.8	0.89
08112017_MVI_1626	-4482	-1330	23	15.5	3.2	0.94
16112017_MVI_1537	-1791	-3303	73	12.3	3.4	0.72
16112017_MVI_1540	-2290	-2724	-40	16.2	2.2	0.77
16112017_MVI_1544	-3942	-3228	-56	14.8	1.6	0.69
16112017_MVI_1559	-1799	-2466	-47	17.0	1.0	0.83
16112017_MVI_1601	-1772	-2388	-10	15.3	2.5	0.87
16112017_MVI_1605	-2521	-2227	-1	13.3	1.3	0.88
16112017_MVI_1608	-3831	-3036	-21	21.6	2.3	0.83
16112017_MVI_1612	-1631	-3445	-47	14.4	2.9	0.80
16112017_MVI_1614	-3178	-2235	1	13.4	1.7	0.76
16112017_MVI_1618	-2897	-3117	-76	12.3	1.0	0.75
16112017_MVI_1621	-2556	-2697	-35	32.8	4.0	0.32
16112017_MVI_1624	-2838	-2578	-65	13.9	1.2	0.67
16112017_MVI_1628	-2883	-2762	-73	13.7	2.4	0.67
17112017_MVI_1651	-2230	-2468	198	21.7	4.4	0.78
17112017_MVI_1654	-2883	-2542	-8	19.1	3.1	0.75
17112017_MVI_1657	-4311	3148	51	15.3	1.9	0.85
17112017_MVI_1701	-2164	-3023	119	15.2	1.8	0.88
17112017_MVI_1717	-3253	-2317	32	23.5	2.4	0.74
17112017_MVI_1722	-4401	-2678	16	17.6	3.0	0.69
mean	-2805	-2578	47	17.3	2.4	0.77
standard deviation	918.4	541	65.7	5.0	0.9	0.13

TABLE 8.1: Summary of VOT results. The sequences highlighted in red belong to the "worst" performing group and the sequences highlighted in green belong to the "best" performing group.

The summary of results in table (8.1) shows that the average reprojection error on over an hour of recordings is $17.3\,\text{pixels}$. The average altitude estimate error is $47\,\text{m}$ and the average local world frame X and Y position estimate errors are $-2805\,\text{m}$

and $-2578\,\mathrm{m}$ respectively. This gives an average position error of $3810\,\mathrm{m}$. The localization performance is poor (as expected) but the ROC curves and AUC scores are promising. The next two paragraphs explain the conditions under which the system performs poorly and under which it behaves well.

Worst performing sequences. The red coloured rows in the results table 8.1 were selected as the worst performing sequences for various reasons: poor cardinality estimate, low AUC score and poor altitude estimate.

On reviewing the data it was found that birds were present for much of the 17112017_MVI_1651 sequence. The birds were detected and tracked as well as possible: see figure 8.11. This is what caused the high cardinality estimate of 4.4. The ROC curves for all trackers on this sequence is shown in figure 8.12.

FIGURE 8.11: Presence of birds in sequence 17112017_MVI_1651.

The ROC curves for the sequence shows that the excessive number of birthed trackers did not gain high confidence regarding tracking an actual target, and it was only *Tracker_15* that tracked the true target.

The sequence 17112017_MVI_1701 was reviewed and it was found that the target was successfully tracked for ± 250 frames and then a helicopter entered the scene. There is no world frame truth data for the helicopter but it was annotated as a real target in the image-plane truth dataset. The helicopter was substantially closer to the camera and higher in the scene, thus the resulting altitude estimate Z was higher. Frames 800 and 1100 of the sequence are shown in figure 8.13. These frames show that *Tracker_3* correctly tracks the commercial airliner and *Tracker_12* picks up and

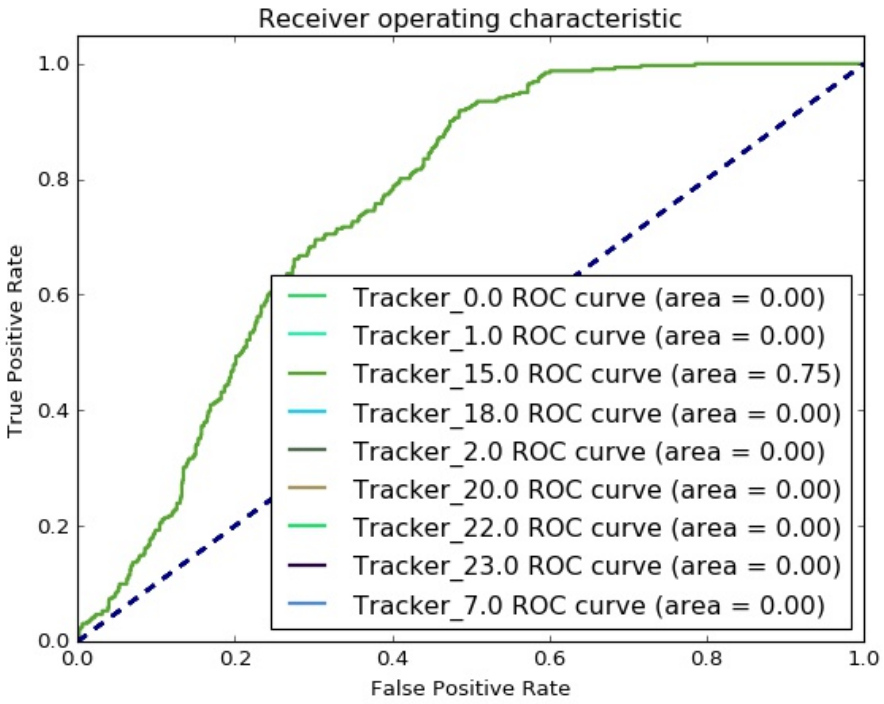


FIGURE 8.12: ROC curve for sequence 17112017_MVI_1651.

tracks the helicopter. Then both trackers lose their respective targets and *Tracker_3* migrates to the helicopter target. The error was made in the evaluation module and on inspection of the results the altitude error ($\bar{\mu}_{Z_{\text{truth}}-Z_{\text{est}}}$) for *tracker_3* during the frames it was tracking the original target was $16\,\text{m}$. It is noted that trackers should not migrate between targets and this was a system error that was not correctly dealt with by the evaluation module.

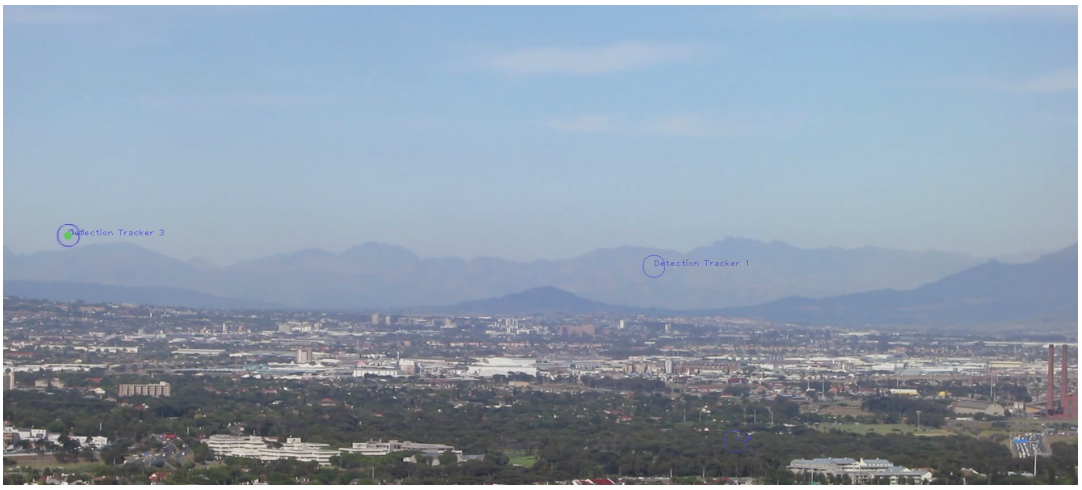


FIGURE 8.13: Frame 200 of sequence 17112017_MVI_1701 *Tracker_3* detected and tracked the target until frame 800.

FIGURE 8.14: Frame 800 — a helicopter enters the scene. *Tracker_12* tracks it and *Tracker_3* loses the original target.

FIGURE 8.15: Frame 1700 *Tracker_3* migrated to a new target and is tracking the helicopter.

The sequence 16112017_MVI_1621 shows a target that is exceedingly faint. Figure 8.17a shows it entering the scene against a blue sky. The tracker successfully detects

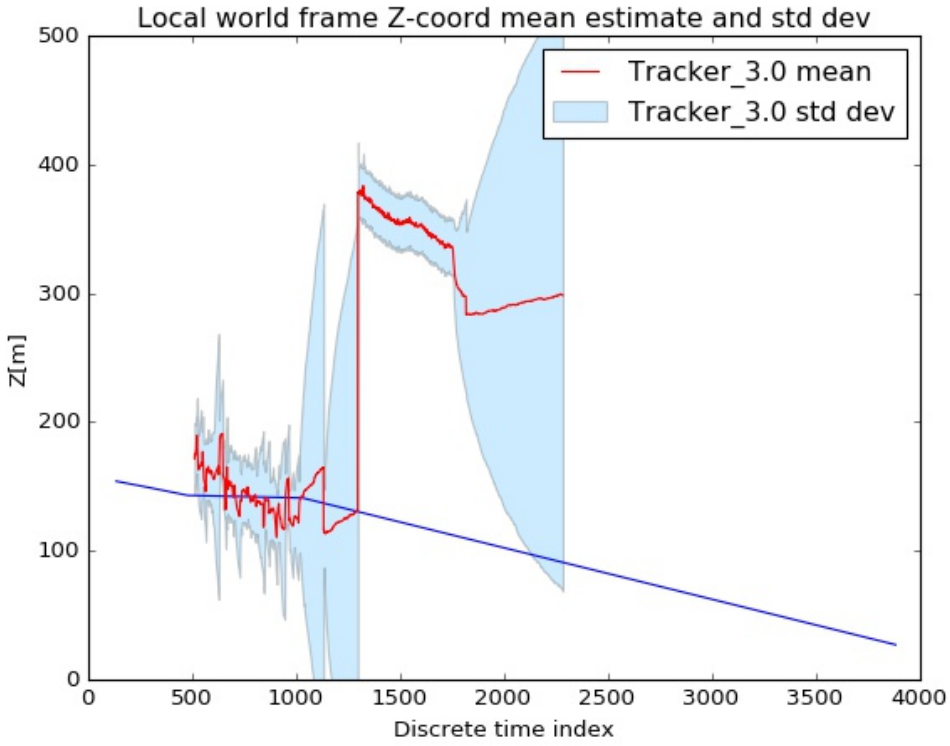


FIGURE 8.16: The altitude profile for *Tracker_3* and the truth data of the original target. The evaluation module failed to detect that *Tracker_3* had migrated targets and thus produced a large altitude error when compared to the truth trajectory of the original target.

and then loses it and continues to miss many detections. Despite continuing to produce a good mean estimate of the target's position and reprojection error the successive missed detections caused the entries in the covariance matrix to grow and the tracker gave poor confidence scores. This resulted in the very low AUC score.

(A) Faint target in sequence 16112017_MVI_1621 against the blue sky

(B) Faint target in sequence 16112017_MVI_1621 against a white cloud.

FIGURE 8.17: Faint target in sequence 16112017_MVI_1621.

The failure on a very faint target is predictable. There is little use in trying to combat this pathology as there is so little information received at the sensor level. Attempting to solve this robustly is guaranteed to be a difficult task.

Best performing sequences. The green coloured rows in table 8.1 were deemed to be the best-performing sequences. These were analysed further and it was found that they had observably less camera shake and fewer consecutive missed detections. The target was also in the field of view for more frames; under these conditions the trackers settled and performed well.

8.4 VOT discussion

The visual object tracking (VOT) system was able to detect and track aircraft flying through the region of interest. The results show that the world frame localization of the aircraft is not very accurate. The VOT system's inaccuracies result from a number of sources. Firstly, using a single camera leads to measurement ambiguities which can be resolved provided the tracker gets a substantial number of correct detections. This leads to the second source of errors — the data association stage. The data association problem is in itself intractable and the assumptions and approximations (see 6) lead to further inaccuracies. Thirdly, insensitivities in the background model lead to missed detections, causing trackers to drift or lose their target altogether. Lastly, the camera pose estimation makes use of data from Google Earth, the accuracy of which is unknown, and the corresponding point selection is done by hand (see chapter 3). However, across all videos the system never failed to detect and track the target for part of the trajectory through the ROI.

Upon reviewing the two groups of trackers, the high performers and the poor performers, it was found that the main reason for poor performance was the presence of birds in the scene at a distance from the camera such that they persist for a number of frames. The system only models the background. Therefore all pixels that cannot be explained by the GMM model have to be explained by a dynamic tracker model. The birds persisted in the field of view and trackers locked on; thus the cardinality estimate increased.

The second pathology was a large altitude error. This was caused by a helicopter entering the FOV and the true target was lost. The same tracker migrated to the new target. Both these failures could be argued as system design failures. This was owing to the lack of a target appearance model, so the only way the system could deal with persistent detections was to track them. Section 9.2.3 addresses this recommendation.

From the analysis of the sequences on which the tracker performed well, it was observed that they had less camera shake and far fewer consecutive missed detections.

Despite the highlighted shortcomings of the visual object tracking (VOT) system it shows promise as a passive airspace monitoring system. It reliably detected all aircraft entering the FOV and even "bested" an operator when scanning the scene for aircraft.

The next chapter concludes the investigation and provides insights and recommendations for future work.

Chapter 9

Conclusions

This chapter draws the investigation to a close. It states the achievements of this work and provides a critical review of relevant aspects of the project. Section 9.2.1 reviews the experimental method. Section 9.2.2 comments on the MTT framework and highlights some of its shortcomings. The problem of passively monitoring airspace is discussed and ideas and recommendations for future work are presented in section 9.2.3.

9.1 Achievements

This thesis detailed the problem of passively monitoring airspace. Two sensing modalities were investigated, namely passive radar and visual camera sensors. Both these modalities generate large amounts of data per measurement scan, and once processed a number of detections are made. These sets of detections held many false alarms and missed detections as the true number of targets in the scene is unknown. This characteristic of the sensors necessitated the need for a multiple target tracking algorithm to track targets in clutter. Based on the literature discussed in chapter 2, a MTT framework was derived. This framework was successfully implemented by creating the multiple target tracking module of the passive radar system as well as in the VOT system. The complete visual object tracking system comprised a calibration and pose estimation module, an image processing module, the multiple target tracking module as well as an evaluation module and visualizations via Google Earth. A large annotated dataset of over an hour of video recordings was curated and used for testing the visual object tracking system. The VOT system solves the real and increasingly important problem of airspace monitoring.

9.2 Critical reviews, future work and recommendations

This section highlights a number of critical insights and retrospectives. The main purpose of this section is to guide the reader who aims to improve upon this work or build a similar system.

9.2.1 System review

This section reviews the experimental method pertaining to the visual object tracking system. The first shortfall of the experiments is the validity and accuracy of the truth data. The data received from *Flight Radar 24* [18] was intermittent and some files had missing or corrupted data. The data received was sufficient to validate the prototype system and show that such a system could provide a solution to the passive airspace monitoring problem.

The pose estimation was done by manually identifying image and world frame point correspondences using Google Earth. This procedure could be improved upon by using more accurate geographic information system (GIS) data.

9.2.2 Multiple target tracking framework review

The MTT framework described in this work shows promise and was successfully demonstrated on two use cases. The strong point of this approach is its generality. This was achieved by making use of probabilistic graphical models. PGMs are a contemporary research topic and there are established general algorithms that apply to a broad range of graphical structures. By specifying suitable potentials it is possible to re-use this framework for many other multiple target tracking applications.

As a full Bayesian recursive estimator solution to the MTT problem this work is lacking, as many assumptions and approximations were required. The primary assumption was approximating the posterior distribution by a set of weighted particles and extracting a MAP data association assignment per sensor scan interval. This work also fell short on fully integrating the background model with the collective state model. If this could be achieved, the random finite sets (RFS) approach would be applicable and the need for a data association stage would fall away.

The evaluation module is a crucial part of developing any practical system. For this project an incorrect assumption about the reliability and availability of the truth data meant that all the results presented here are subject to the unknown quality of the data obtained from *Flight Radar 24* [18]. However, aside from the quality of the truth data, MTT performance metrics are complex and difficult to define. The results presented here were justifying the practical performance and usability of the system. However, the theoretical optimal solution to the MTT problem is difficult to define and so even with perfect information it is unclear what constitute sensible metrics. Recent work suggests the Wassernstein metric together with other convoluted measures; see [53]. The lack of such a evaluation scheme is seen as a significant failure in this work. That being said the results are significant and further work in this direction is warranted. Suggested directions for future work are presented next.

9.2.3 Future work and recommendations

In order to improve this work and provide valuable contributions to the fields of computer vision and Bayesian filtering within the context of airspace monitoring, the following recommendations are made.

Firstly, to improve the accuracy of the world frame target estimation, the single static camera system should be extended to a multi-camera configuration. The current software architecture can accommodate this but accurate camera synchronization would be required.

Secondly, a better source of truth data would greatly speed up development time as validating the correctness and accuracy of the system would enable faster iterations over ideas and approaches. The author recommends achieving this by simulating a simple scene with targets moving through it. Initial and promising work was started in this regard using *Blender* [7], a 3D modelling and simulation tool. Unfortunately and unwisely it was abandoned. This would provide a good test-bed for any MTT framework and truth data would be accurate and readily available. There have also been recent advancements in photo-realistic rendering [28], which have been used to generate training data for vision systems. This approach could provide insights into better ways to do background modelling and target detection. These resources will provide the reader with a good starting point if further work on this topic is to be undertaken [58, 47, 59].

Thirdly, the background model used in this work has a number of limitations which were discussed in chapter 4. Incorporating spatial constraints into the model is the obvious next step, however achieving this is not trivial. Markov random fields (MRFs) and conditional random fields (CRFs) provide one such avenue but retaining a temporal component in these models greatly increases the computational complexity. A number of comments and recommendations in the context of this work were put forward in section 4.2.

Fourth, the data association stage can be removed by defining a multi-target likelihood function. This would bring the solution closer to the full Bayesian recursive estimator (BRE). It is unclear how to implement this, but the work done by Mahler using random finite sets (RFS) is leading the field in this endeavour. See section 2.4 for relevant works and framing in the MTT context.

Fifth, an appearance model should be added. As discussed in chapter 4 this does not work reliably for small image patches, but upon reviewing the failure cases it appears that such a model will be needed. This would require actively controlling the camera and zooming in to classify the object while tracking. This poses numerous technical hurdles: calibration and keeping the background model up to date. As discussed, integrating this into a Bayesian framework would require the classifier to have calibrated outputs. More work is needed here.

The field of passive radar is well established and extensive. This work built an MTT module for the (*Peralex*) passive radar system (PRS). Similar recommendations are made for this application. Readily available truth data would provide much needed feedback and validation. A more realistic simulation environment would also be beneficial.

As a whole, this investigation has achieved its aims (section 1.1.3). A multiple target tracking framework was developed and it was applied to two use cases. The first is a visual object tracking (VOT) system composed of an image processing module, the MTT module and an evaluation module. The VOT system achieved aims one through four identified in section 1.1.3. The second use case is incorporating the MTT module into an existing bistatic passive radar system, thus achieving the fifth aim mentioned in section 1.1.3 and validating the multiple target tracking framework. In spite of the various shortcomings listed above, this thesis covered a wide range of work: researching a suitable approach to multiple target tracking, implementing a system capable of processing two sensing modalities, and tracking aircraft in the region of interest. The shortcomings were fully expressed throughout this report and a detailed set of recommendations and future directions was presented.

Bibliography

[1] Bjoern Andres, Thorsten Beier and Jh Kappes. 'OpenGM 2.0 Manual'. In: (2017). URL: http://hci.iwr.uni-heidelberg.de/opengm2/download/opengm-2.0.2-beta-manual.pdf.

[2] M.S. Arulampalam et al. 'A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking'. In: *IEEE Transactions on Signal Processing* 50.2 (2002), pp. 174–188. ISSN: 1053587X. DOI: 10.1109/78.978374. arXiv: arXiv:1011.1669v3. URL: http://ieeexplore.ieee.org/xpl/login.jsp?tp=%7B%5C&%7Darnumber=978374%7B%5C&%7Durl=http://ieeexplore.ieee.org/iel5/78/21093/00978374.

[3] B. Babenko, Ming-Hsuan Yang and S. Belongie. 'Robust Object Tracking with Online Multiple Instance Learning'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 33.8 (Aug. 2011), pp. 1619–1632. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2010.226. URL: http://vision.ucsd.edu/%7B~%7Dbbabenko/data/miltrack%7B%5C_%7Dcvpr09.pdf%20http://ieeexplore.ieee.org/document/5674053/.

[4] Yaakov Bar-Shalom, Fred Daum and Jim Huang. 'The probabilistic data association filter'. In: *IEEE Control Systems Magazine* 29.6 (2009), pp. 82–100. ISSN: 08880611. DOI: 10.1109/MCS.2009.934469.

[5] Michael Betancourt. 'A Conceptual Introduction to Hamiltonian Monte Carlo'. In: (2017). arXiv: 1701.02434. URL: http://arxiv.org/abs/1701.02434.

[6] S.S. Blackman. 'Multiple hypothesis tracking for multiple target tracking'. In: *IEEE Aerospace and Electronic Systems Magazine* 19.1 (2004), pp. 5–18. ISSN: 0885-8985. DOI: 10.1109/MAES.2004.1263228. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1263228.

[7] Blender. *Blender 3D simulation and modeling*. 2017. URL: https://www.blender.org/ (visited on 21/09/2017).

[8] Thierry Bouwmans. 'Traditional and recent approaches in background modeling for foreground detection: An overview'. In: *Computer Science Review* 11-12 (2014), pp. 31–66. ISSN: 15740137. DOI: 10.1016/j.cosrev.2014.04.001. URL: http://dx.doi.org/10.1016/j.cosrev.2014.04.001.

[9] Benjamin de Charmoy. *Multiple Target Tracking Framework*. 2017. URL: https://github.com/BenjaminDev/Masters2016.

[10] Z .H. E. Chen. 'Bayesian Filtering: From Kalman Filters to Particle Filters, and Beyond'. In: *Statistics* 182.1 (2003), pp. 1–69. ISSN: 00220949. DOI: 10.1.1.107.7415. arXiv: 10.1.1.107.7415. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.107.7415%7B%5C&%7Drep=rep1%7B%5C&%7Dtype=pdf.

[11] Z. M. Chen. 'Efficient Multi-Target Tracking Using Graphical Models'. PhD thesis. 2008, pp. 1–104. URL: http://ssg.mit.edu/ssg%7B%5C_%7Dtheses/ssg%7B%5C_%7Dtheses%7B%5C_%7D2000%7B%5C_%7D2009/Chen%7B%5C_%7DMEng%7B%5C_%7D5%7B%5C_%7D08.pdf.

[12] C.Y. Chong. 'Graph approaches for data association'. In: *FUSION, 2012 15th International Conference on* (2012), pp. 1578–1585. URL: http://ieeexplore.ieee.org/xpls/abs%7B%5C_%7Dall.jsp?arnumber=6290494.

[13] Arnaud Doucet, Simon Godsill and Christophe Andrieu. 'On sequential Monte Carlo sampling methods for Bayesian filtering'. In: *Statistics and Computing* 10.3 (2000), pp. 197–208. ISSN: 09603174. DOI: 10.1023/A:1008935410038. URL: http://link.springer.com/article/10.1023/A:1008935410038%7B%5C%%7D5Cnhttp://link.springer.com/10.1023/A:1008935410038.

[14] Arnaud Doucet and Am Johansen. 'A tutorial on particle filtering and smoothing: fifteen years later'. In: *Handbook of Nonlinear Filtering* December (2011), pp. 656–704. ISSN: 01677152. DOI: 10.1.1.157.772. URL: http://automatica.dei.unipd.it/tl%7B%5C_%7Dfiles/utenti/lucaschenato/Classes/PSC10%7B%5C_%7D11/Tutorial%7B%5C_%7DPF%7B%5C_%7Ddoucet%7B%5C_%7Djohansen.pdf.

[15] Mathias Drton and Marloes H. Maathuis. 'Structure Learning in Graphical Modeling'. In: (2016). ISSN: 2326-8298. DOI: 10.1146/annurev-statistics-060116-053803. arXiv: 1606.02359. URL: http://arxiv.org/abs/1606.02359.

[16] Riba Edgar et al. *tiny-dnn: header only, dependency-free deep learning framework in C++14*. 2017. URL: http://tiny-dnn.readthedocs.io.

[17] Peralex Electonics. 'Project Note 01: Tracking'. In: (2016).

[18] FlightRadar24. *Flight Radar 24 live air traffic*. 2017. URL: https://www.flightradar24.com (visited on 27/10/2017).

[19] Andrew Gelman et al. *Bayesian Data Analysis*. 2004, p. 696. ISBN: 978-1-4398-9820-8. DOI: 10.1007/s13398-014-0173-7.2.

[20] Zoubin Ghahramani. 'Bayesian nonparametrics and the probabilistic approach to modelling'. In: *Philosophical Transactions of the Royal Society A* 371.1984 (2013), pp. 1–27. ISSN: 1364-503X. DOI: 10.1098/rspa.00000000. arXiv: arXiv:1301.3724v1.

[21] Richard Hartley and Andrew Zisserman. *Multiple view geometry in computer vision*. Cambridge university press, 2003.

[22] Mcelory Hoffmann, Karin Hunter and Ben Herbst. 'The hitchhiker's guide to the particle filter'. In: *Parsa* (2008). URL: http://appliedmaths.sun.ac.za/%7B~%7Dherbst/research/publications/particle%7B%5C_%7Dfilter%7B%5C_%7DPrasa2008.pdf.

[23] J. D. Hunter. 'Matplotlib: A 2D graphics environment'. In: *Computing In Science & Engineering* 9.3 (2007), pp. 90–95.

[24] Itseez. *Open Source Computer Vision Library*. https://github.com/itseez/opencv. 2015.

[25] M. C. Jackson. 'The geometry of bistatic radar systems'. In: *Communications, Radar and Signal Processing, IEE Proceedings F* 133.7 (1986), pp. 604–612. ISSN: 0143-7070. DOI: 10.1049/ip-f-1:19860097.

[26] Michael I Jordan. *Graphical models*. Tech. rep. 1996, pp. 1–35. URL: https://www.cs.cmu.edu/%7B~%7Daarti/Class/10701/readings/graphical%7B%5C_%7Dmodel%7B%5C_%7DJordan.pdf.

[27] Zdenek Kalal, Krystian Mikolajczyk and Jiri Matas. 'Tracking-learning-detection'. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 34.7 (2012), pp. 1409–1422. ISSN: 01628828. DOI: 10.1109/TPAMI.2011.239.

[28] Simon Kallweit et al. 'Deep Scattering: Rendering Atmospheric Clouds with Radiance-Predicting Neural Networks'. In: *ACM Trans. Graph. (Proc. of Siggraph Asia)* 36.6 (2017). DOI: 10.1145/3130800.3130880. URL: http://doi.acm.org/10.1145/3130800.3130880.

[29] Charles Karney. *GeographicLib*. https://sourceforge.net/p/geographiclib/. 2017.

[30] B. Kausler. 'Tracking-by-Assignment as a Probabilistic Graphical Model with Applications in Developmental Biology'. PhD thesis. 2013, pp. 1–112. URL: http://archiv.ub.uni-heidelberg.de/volltextserver/15296/.

[31] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models- Principles and Techniques*. Vol. 53. 1989, p. 160. ISBN: 9788578110796. DOI: 10.1017/CBO97811. arXiv: arXiv:1011.1669v3.

[32] Daphne Koller et al. 'Graphical Models in a Nutshell'. In: *Introduction to Statistical Relational Learning* (2007), p. 43. DOI: 10.1.1.146.2935. URL: http://www.robotics.stanford.edu/%7B~%7Dkoller/Papers/Koller+al:SRL07.pdf.

[33] Chris Kreucher, Keith Kastella and Alfred O. Hero III. 'Tracking Multiple Targets Using a Particle Filter Representation of the Joint Multitarget Probability Density'. In: *In Signal and Data Processing of Small Targets* 5204 (2004), pp. 258–269. DOI: 10.1117/12.502696.

[34] Tiancheng Li, Miodrag Bolic and Petar M. Djuric. 'Resampling Methods for Particle Filtering: Classification, implementation, and strategies'. In: *IEEE Signal Processing Magazine* 32.3 (May 2015), pp. 70–86. ISSN: 1053-5888. DOI: 10.1109/MSP.2014.2330626. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=7079001.

[35] Hans Andrea Loeliger et al. 'The factor graph approach to model-based signal processing'. In: *Proceedings of the IEEE* 95.6 (2007), pp. 1295–1322. ISSN: 00189219. DOI: 10.1109/JPROC.2007.896497.

[36] Francois de Villiers Maasdorp. 'Doppler-only Target Tracking for a Multistatic Radar Exploiting FM Band Illuminators of Opportunity'. PhD thesis. University of Cape Town, 2015.

[37] Ronald Mahler. 'Random Set Theory for Target Tracking and Identification'. In: *Data Fusion Hand Book* (2001), pp. 14/1–14/33. DOI: 10.1201/9781420038545.ch14. URL: http://dsp-book.narod.ru/HMDF/2379ch14.pdf%7B%5C%%7D5Cnhttp://www.crcnetbase.com/doi/abs/10.1201/9781420038545.ch14.

[38] Ronald P. S. Mahler. *Statistical Multisource-Multitarget*. Artech House, 2007. ISBN: 9781596930926.

[39] Ronald P.S. Mahler. 'Multitarget Bayes Filtering via First-Order Multitarget Moments'. In: *IEEE Transactions on Aerospace and Electronic Systems* 39.4 (2003), pp. 1152–1178. ISSN: 00189251. DOI: 10.1109/TAES.2003.1261119.

[40] K. P. Murphy. 'An introduction to graphical models'. In: *Rap. tech* 96.May (2001), pp. 1–19. ISSN: 01621459. DOI: 10.1198/jasa.2001.s425. arXiv: arXiv:1608.07891v1. URL: http://pubs.amstat.org/doi/abs/10.1198/jasa.2001.s425%7B%5C%%7D5Cnhttp://www.denizyuret.com/ref/murphy/intro%7B%5C_%7Dgm.pdf.

[41] Kevin P Murphy. 'Undirected Graphical Models (Markov Random Fields)'. In: *Machine Learning: A Probabilistic Perspective* 2 (2012), pp. 661–705.

[42] D. Mušicki and R. Evans. 'Joint Integrated Probabilistic Data Association - JIPDA'. In: *Proc. Fusion 2002* I (2002), pp. 1120–1125. DOI: 10.1109/ICIF.2002.1020938.

[43] S. Nowozin and C .H. Lampert. 'Structured learning and prediction in computer vision'. In: *Foundation and trends in computer graphics and vision* 6.3-4 (2011), pp. 185–365.

[44] Judea Pearl. 'Reverend Bayes on inference engines: A distributed hierarchical approach'. In: *Proceedings of the AAAI National Conference on AI* (1982), pp. 133–136. ISSN: 19326203. DOI: 10.1.1.294.5479. arXiv: arXiv:1206.1208v2. URL: http://scholar.google.com/scholar?hl=en%7B%5C&%7DbtnG=Search%7B%5C&%7Dq=intitle:Reverend+Bayes+on+Inference+Engines%7B%5C#%7D0.

[45] Qt. *Qt*. https://www.qt.io/. 2017.

[46] Donald Reid. 'An algorithm for tracking multiple targets'. In: *IEEE Transactions on Automatic Control* 24.6 (1979), pp. 843–854. ISSN: 0018-9286. DOI: 10.1109/TAC.1979.1102177. URL: http://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=4046312%7B%5C%%7D5Cnhttp://ieeexplore.ieee.org/lpdocs/epic03/wrapper.htm?arnumber=1102177.

[47] Ashish Shrivastava et al. 'Learning from Simulated and Unsupervised Images through Adversarial Training'. In: *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3.4 (2016). DOI: 10.1109/CVPR.2017.241. arXiv: 1612.07828. URL: http://arxiv.org/abs/1612.07828.

[48] Andrews Sobral. 'BGSLibrary: An OpenCV C++ Background Subtraction Library'. In: *IX Workshop de Visão Computacional (WVC'2013)*. Rio de Janeiro, Brazil, June 2013. URL: https://github.com/andrewssobral/bgslibrary.

[49] Andrews Sobral and Antoine Vacavant. 'A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos'. In: *Computer Vision and Image Understanding* 122 (2014), pp. 4–21. ISSN: 10773142. DOI: 10.1016/j.cviu.2013.12.005.

[50] Lawrence Stone. 'Likelihood Ratio Detection and Tracking'. In: *2009 12th International Conference on Information Fusion*. July. 2009, p. 10.

[51] Lawrence D. Stone. 'Multisensor Data Fusion, 2 Volume Set'. In: *Handbook of Multisensor Data Fusion* 3 (2001), pp. 1–30. DOI: 10.1201/9781420038545. URL: http://www.crcnetbase.com/doi/book/10.1201/9781420038545.

[52] Charles Sutton and Andrew McCallum. 'An Introduction to Conditional Random Fields'. In: *Machine Learning* 4.4 (2011), pp. 267–373. ISSN: 1935-8237. DOI: 10.1561/2200000013. arXiv: 1011.4088. URL: http://arxiv.org/abs/1011.4088%7B%5C%%7D5Cnhttp://www.arxiv.org/pdf/1011.4088.pdf.

[53] Daniel Svensson, Johannes Wintenby and Lennart Svensson. 'Performance evaluation of MHT and GM-CPHD in a ground target tracking scenario'. In: *2009 12th International Conference on Information Fusion* (2009), pp. 300–307.

[54] Y. Teh and Michael I. Jordan. 'Hierarchical Bayesian nonparametric models with applications'. In: *Bayesian Nonparametrics* (2009), pp. 1–48. ISSN: 0162-1459. DOI: 10.1198/016214506000000302.

[55] Sebastian Thrun. 'Probabilistic robotics'. In: *Communications of the ACM* 45.3 (2002), pp. 1999–2000. ISSN: 00010782. DOI: 10.1145/504729.504754. arXiv: arXiv:1011.1669v3. URL: http://portal.acm.org/citation.cfm?doid=504729.504754.

[56] Craig Andrew Tong. 'A Scalable Real-time Processing Chain for Radar Exploiting Illuminators of Opportunity'. PhD thesis. 2014.

[57] Emanuele Trucco and Konstantinos Plakas. 'Video tracking: A concise survey'. In: *IEEE Journal of Oceanic Engineering* 31.2 (2006), pp. 520–529. ISSN: 03649059. DOI: 10.1109/JOE.2004.839933.

[58] UnRealCV. *Synthetic Computer Vision 3D simulation and modeling*. 2017. URL: https://github.com/unrealcv/synthetic-computer-vision (visited on 02/09/2017).

[59] V. S. R. Veeravasarapu, Constantin Rothkopf and Ramesh Visvanathan. 'Adversarially Tuned Scene Generation'. In: (2017). ISSN: 1701.00405. DOI: 10.1109/CVPR.2017.682. arXiv: 1701.00405. URL: http://arxiv.org/abs/1701.00405.

[60] Jaco Vermaak, Simon J. Godsill and Patrick Perez. 'Monte carlo filtering for multi target tracking and data association'. In: *Aerospace and Electronic Systems, IEEE Transactions on* 41.1 (2005), pp. 309–332. ISSN: 0018-9251. DOI: 10.1109/TAES.2005.1413764.

[61] Ba-ngu Vo et al. 'Multi-Target Tracking'. In: *Wiley Encyclopedia of Electrical and Electronics Engineering* (2015). DOI: 10.1002/047134608X.W8275.

[62] Ba-Tuong Vo. 'Random Finite Sets in Multi-Object Filtering'. PhD thesis. 2008, p. 254. URL: http://ieeexplore.ieee.org/xpls/abs%7B%5C_%7Dall.jsp?arnumber=1561884.

[63] Martin J. Wainwright and Michael I. Jordan. 'A Variational Principle for Graphical Models'. In: *New Directions in Statistical Signal Processing* March (2005).

[64] Jason L. Williams and Roslyn A. Lau. 'Data association by loopy belief propagation'. In: *2010 13th International Conference on Information Fusion* (2010), pp. 1–8.

[65] Jason Yosinski et al. 'How transferable are features in deep neural networks?' In: *Advances in neural information processing systems* (2014), pp. 3320–3328. ISSN: 10495258. arXiv: 1411.1792. URL: http://arxiv.org/abs/1411.1792.

[66] Zoran Zivkovic. 'Improved adaptive Gaussian mixture model for background subtraction'. In: *Proceedings of the 17th International Conference on Pattern Recognition, 2004. ICPR 2004.* 2.2 (2004), pp. 28–31. ISSN: 1051-4651. DOI: 10.1109/ICPR.2004.1333992.

[67] Zoran Zivkovic and Ferdinand van der Heijden. 'Recursive unsupervised learning of finite mixture models.' In: *IEEE Trans. Pattern Analysis and Machine Intelligence* 26.5 (2004), pp. 651–6. ISSN: 0162-8828. DOI: 10.1109/TPAMI.2004.1273970. URL: http://www.ncbi.nlm.nih.gov/pubmed/15460286.

[68] Zoran Zivkovic and Ferdinand Van Der Heijden. 'Efficient adaptive density estimation per image pixel for the task of background subtraction'. In: *Pattern Recognition Letters* 27.7 (2006), pp. 773–780. ISSN: 01678655. DOI: 10.1016/j.patrec.2005.11.005.