# Simultaneous Multi-Object Tracking and Classification via Approximate Variational Inference

**Victor Romero-Cano**

A thesis submitted in fulfilment
of the requirements of the degree of
Doctor of Philosophy

THE UNIVERSITY OF
SYDNEY

Australian Centre for Field Robotics
School of Aerospace, Mechanical and Mechatronic Engineering
The University of Sydney

March 2015

# Declaration

I hereby declare that this submission is my own work and that, to the best of my knowledge and belief, it contains no material previously published or written by another person nor material which to a substantial extent has been accepted for the award of any other degree or diploma of the University or other institute of higher learning, except where due acknowledgement has been made in the text.

**Victor Romero-Cano**

September 16, 2015

# Abstract

Victor Romero-Cano
The University of Sydney

Doctor of Philosophy
March 2015

# Simultaneous Multi-Object Tracking and Classification via Approximate Variational Inference

In modern applications, robots are expected to work in complex dynamic environments and extract meaningful information from low-level, noisy data. In particular, they must build a description of the objects they interact with. This description should be both qualitative and quantitative. The former can be expressed in terms of object classes, while the latter is expressed by the object dynamics.

Qualitative descriptors can be thought of as discrete assignments of object trajectories to category labels that represent different motion patterns in the environment. Obtaining these descriptors along with the kinematic states of the objects, from data, is a challenging task due to the noisy nature of sensor measurements, sensor failure, object occlusions and the presence of objects with infrequent dynamics.

Quantitative descriptors such as locations and velocities are usually obtained using widely known filtering techniques derived from the Kalman filter. Nevertheless, when dealing with measurements originated by multiple objects, associating these measurements with individual objects generates a number of hypotheses that grows combinatorially with the number of measurements, and exponentially with time. Generating these assignments, while also estimating the kinematic state and classes of the objects is a computationally intractable problem.

This thesis proposes a probabilistic model that exploits the correlations between object trajectories and classes and an inference procedure that renders the problem

tractable through a structured variational approximation. The framework presented is very generic, and can be used for various tracking applications. It can handle objects with different and/or infrequent dynamics, such as cars and pedestrians, and it can seamlessly integrate multi-modal features, for example object dynamics and appearance.

# Acknowledgements

Firstly, I would like to thank my supervisors, Juan Nieto and Gabriel Agamennoni, for their support, guidance and encouragement. Their constant availability and advice were vital during this journey. I particularly thank Gabriel for introducing me to the area of approximate inference and the insightful feedback he gave me even after leaving to ETH.

I am thankful to the Rio Tinto Centre for Mine Automation (RTCMA) and the Australian Centre for Field Robotics (ACFR) for supporting me with a post-graduate scholarship.

Many thanks to ACFR's director Prof. Eduardo Nebot and all my friends at ACFR and Usyd, including those who have left: Nasir Ahsan, Suchet Bargoti, Marcos Gerardo-Castro, Prasad Hemakumara, Andres Hernandez, Jonathan Jeyaratnam, Guilherme Maeda, Thierry Peynot, Jeremy Soh, Victor Vera and Francisco Zubizarreta.

Special thanks to Steven Scheding, Raymond Leung, Abhinav Goyal and the RTCMA team; I am very lucky I have spent the last four years surrounded by the most talented people I have ever known. I must thank my friends and peers, Andrew Palmer, Rishi Ramakrishnan, Zachary Taylor, Lloyd Windrim, Tatsumi Uezato and Seong Ho Lee. I will always remember lunchtimes, the delightful conversations, Rishi's *great* ideas and the daily doses of coffee.

Gracias infinitas a mi familia. Mis padres, Adelina y Jorge, y mis hermanos Laura, Yelitza, Luis y Jairo, siempre encontraron la forma de hacerme sentir su presencia a pesar de la distancia.

Finally, I would like to thank Diana, my piece of heaven. There are no words to express my gratitude. Your love kept me afloat.

*Look up at the stars and not down at your feet.*
*Try to make sense of what you see,*
*and wonder about what makes the universe exist.*
*Be curious.*
Stephen Hawking

# Contents

# List of Figures

# List of Tables

# List of Algorithms

# Chapter 1

# Introduction

During the last two decades many attempts have been made towards the development of autonomous vehicles. The Eureka-Prometheus project in the early 1990s [1] and the *Grand DARPA Challenge* in 2005 [2] served as venues for showcasing cars that drove autonomously in highway- and outback scenarios respectively. In the *DARPA Urban Challenge* of 2007 the standards were set even higher, with cars expected to not only drive, but also interact with other cars. However, streets are not only populated by cars, but also by a variety of other participants including pedestrians and cyclists. Therefore, the next step for autonomous cars was clear: they must be able to interact with pedestrians at the very least. Google made significant progress towards this goal in 2010 when one of their platforms drove autonomously in inner-city areas and interacted with both cars and pedestrians for the first time [3].

The introduction of robotic platforms to environments where they need to interact with multiple traffic participants such as cars, pedestrians and cyclists, requires these platforms to have a high degree of Situational Awareness (SA) [4]. In the context of complex dynamic environments, SA can only be ensured by perception systems that provide the robot with geometrically accurate and semantically meaningful representations of the objects in the environment [5, 6]. An example of this is the Bertha Benz experimental vehicle [7], whose motion planning and control module operated using a set of geometric constraints given by a digital map, context rules and *perceived*

*objects*. In between conventional and autonomous cars, Advanced Driving Assistance Systems (ADAS) have emerged as commercial car features that increase driver awareness. ADAS that consider multiple object categories are already on the horizon of car companies such as Volvo, for safety purposes [8], as illustrated in Figure 1.1. Volvo has modified their ADAS so their cars generate a warning and activate auto-breaking once imminent collision with either a pedestrian or a cyclist has been predicted.

**Figure 1.1** – An application example of multi-category object tracking. A perception system with multi-category tracking capabilities can differentiate between cyclists and pedestrians, and track them accordingly. Image extracted from [8].

In spite of the advances in intelligent vehicles, an attentive human driver still has a better driving performance. An attentive human driver can observe the traffic participants, predict their dynamic behaviour, and define a motion policy that accounts for the multi-class nature of these dynamic behaviours. Autonomous cars on the other hand, require further development in terms of interpreting a given traffic scenario and obtaining meaningful behaviour prediction of other traffic participants [7]. On-road behaviour analysis is perhaps one of the newest and therefore least mature areas of Intelligent Transportation Systems (ITS) research [9].

In order to achieve a high degree of SA and therefore be able to perform high-level tasks such as planning, traffic analysis and behaviour prediction, modern robotic

systems must provide answers to two key long-standing questions in mobile autonomy; *what* are the objects in the environment, and *how* are they moving. These two questions are highly correlated: the class of an object should define the way it is expected to move, whereas the way it moves tells us a lot about what it is. The motion of a car, for example, is fundamentally different from that of a pedestrian. Additionally, this correlation allows us to disambiguate approaches that rely only on appearance to classify objects: objects such as cyclists and pedestrians appear visually similar but they are different in their motion.

This thesis presents a novel mathematical framework that captures these correlations and aims to bridge the gap between geometric and semantic representations of dynamic objects. Our framework enables a perception system to describe relevant objects in the environment both quantitatively, through their trajectories, and qualitatively through the assignment of class labels.

The process of estimating multiple trajectories, a.k.a. multi-object tracking, has associated to it the data association problem, which is the problem of assigning object observations from the sensors to hypotheses the system has about the state of the objects. The joint estimation of class assignments, dynamic states and data association results in a computationally intractable problem. The work presented in this thesis proposes a probabilistic model and an inference procedure that renders the problem tractable through a structured variational approximation. The framework presented is very generic, and can be used for various tracking applications. It can handle objects with different dynamics, such as cars and pedestrians, and it can seamlessly integrate multi-modal features, for example object dynamics and appearance, in a computationally efficient way with minimal user input.

The remainder of this chapter provides the aim and scope of the thesis along with the contributions and a brief description of the manuscript's structure.

## 1.1 Aim and Scope

The fundamental contributions of this thesis focus on tracking, data association and trajectory-based object classification. Figure 1.2 shows the major components of a robotic perception system. Lower-level tasks such as sensor selection, deployment and calibration [10, 11], and ego-vehicle localisation [12, 13], are not within the scope of this work.

As a preprocessing step, measurements from the sensing module are converted into disjoint observations by an object-detection module. Once objects of interest are detected, their position coordinates are transformed into a global reference frame and fed to our tracking and classification module. This module associates multiple detections across time, thus creating hypotheses of the state of individual objects while also calculating their class assignment probabilities. For hypothesis validation we use a stereo-vision rig as sensor modality, yet the framework is general enough so that any other sensor modality that provides both depth and appearance information can be used.

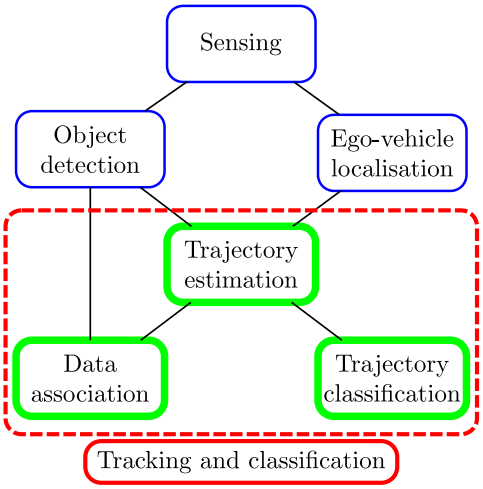



**Figure 1.2** – Aim and scope. Bounding boxes with thicker edges highlight the contributions of this thesis: trajectory estimation (tracking), data association and trajectory classification.

## 1.2 Contributions

This thesis develops a probabilistic framework for joint tracking and classification of moving objects. The inference procedure is efficient and sequential so it can be directly implemented in a robotic platform. The thesis also presents learning techniques for the fitting of the modelling parameters. The specific contributions of this thesis are:

- A holistic probabilistic graphical model that encapsulates the correlation between object classes and object states while also modelling data association.
- The Expectation-Association (EA) algorithm: a new method for performing state estimation, data association, and trajectory-based object classification in a joint probabilistic fashion.
- A novel solution to the data association problem in multi-object tracking that combines appearance and dynamic features in a unified probabilistic framework.
- A Maximum A-Posteriori Expectation Maximisation (MAP-EM) algorithm for automatically estimating the parameters of our model.
- An extensive validation using publicly available data collected in urban environments, and comparisons with state-of-the-art methods.

## 1.3 Thesis Structure

This thesis consists of six further chapters. **Chapter 2** presents the background information needed to motivate the rest of the thesis. A review of the literature that tackles the problem of object trajectory estimation and classification is also provided. **Chapter 3** introduces a novel probabilistic graphical model for describing the correlations between object trajectories and object classes while also accounting for data association ambiguity and objects with unexpected dynamics. The chapter follows a bottom-up approach. A mathematical model for individual trajectories is

first introduced. Then this model is extended so that multiple object categories can be represented. At the end of the chapter, a final version of the model that accounts for unexpected behaviours and data association ambiguities is formulated.

**Chapter 4** presents the EA algorithm, which is an approximate variational method for performing efficient and sequential inference on the model introduced in Chapter 3. **Chapter 5** addresses the automatic learning of the parameters of our model from trajectory data. **Chapter 6** presents the evaluation of the method using publicly available datasets from urban environments. Finally, **Chapter 7** presents the main conclusions and discusses future research directions.

# Chapter 2

# Background and Literature Review

This chapter presents the background material necessary to understand and motivate this thesis. Section 2.1.2 starts by motivating the use of Probabilistic Graphical Models (PGMs) as a modelling paradigm. Section 2.1.1 presents the formulation of multi-object tracking and classification as a state-estimation problem. This section also defines some concepts from probability theory that will be used throughout the text, along with the probability distributions used as the constituting blocks of our model.

After introducing PGMs, Section 2.1.4 gives an overview of approximate inference. Inference procedures efficiently calculate distributions over the variables of interest by integrating the context knowledge provided by the model and the statistical evidence from the data. They are approximate in the sense that assumptions about the form of the output distribution are done so that a tractable solution can be derived. Learning the parameters of the model on which inference is performed, is explained in Section 2.1.5.

Section 2.2.5 presents a review of the techniques used to generate object detections, whereas Section 2.1.6 explains the evaluation metrics used to assess the performance of our multi-object tracking methods.

The chapter ends with a review of the literature. It walks the reader through the major milestones and publications on tracking and trajectory-based object classification, and

gives a summary of the history of tracking and its relevance to perception in robotics and intelligent transportation systems.

## 2.1 Background

### 2.1.1 Probabilistic Graphical Models for Sequential Data

Probabilistic Graphical Models (PGMs) provide a framework based on graph theory [14] that allows us to represent the variables of a system as nodes in a graph and the relationships between these variables as edges that connect them. The state of the variables of the system is represented by means of random variables, whereas the state of the entire system is represented by a joint probability distribution that is a function of factors over the random variables. If no prior information about the relations between variables is available, the PGM would be a fully connected graph.

PGMs enable the introduction of contextual knowledge about the problem, in the structure of the graph. This knowledge is typically introduced through adding or removing edges/relationships between nodes/variables, and encodes conditional independence relationships that provide simpler factorisations of the joint distribution. The number of edges between nodes define a trade-off between representational power and computational tractability. In general, the higher the number of edges, the richer the representation is, but the more complex the running inference on the PGM becomes and vice-versa.

There are two main types of PGMs: undirected PGMs, also known as Markov Random Fields (MRF) and directed PGMs, also known as Bayesian Networks [15]. They are different in that the first one encodes similarity relationships whereas the second one encodes one-way causal relationships. Given the sequential nature of object trajectories, we focus on directed PGMs, i.e. graphs that represent causal relationships by means of directional edges. Applying directed PGMs in the tracking and dynamics-based classification context enables the design of modular approaches that are interpretable and easily extendible.

A Bayesian network is a data structure as shown in Figure 2.1, where variables of interest are represented with nodes, and the relationships (functional dependence) between different variables are represented by means of edges connecting related nodes. Shaded nodes represent observed variables, i.e. observation features. Both nodes and edges are modelled by means of parametric probability distributions, with node distributions being chosen so that they represent the marginal behaviour of the variable, and edge distributions being chosen as conditional distributions that represent the local relationship between neighbouring variables. In particular, the PGM introduced by this thesis models both nodes and edges using Gaussian probability distributions:

$$\begin{aligned} x_0 &\sim \mathcal{N}(x_0; u, V); \\ x_t|x_{t-1} &\sim \mathcal{N}\left(x_{t|t-1}; Fx_{t-1}, Q\right); \\ y_t|x_t &\sim \mathcal{N}(y_t; Hx_t, R). \end{aligned} \tag{2.1}$$

Where $\mathcal{N}(\mu, \Omega)$ represents a multivariate Gaussian distribution with mean $\mu$ and covariance matrix $\Omega$.

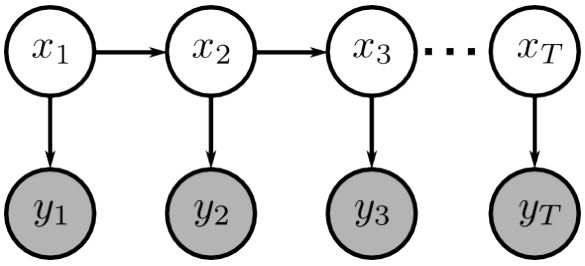



**Figure 2.1** – A linear chain PGM of latent variables with each observed variable conditioned on the state of the corresponding latent variable.

Figure 2.1 illustrates a graph that encodes the conditional independences parametrised by Equations 2.1. The state $x_t$ is conditionally dependent on only the previous state $x_{t-1}$, while the observation $y_t$ is conditionally dependent only on the state $x_t$. In this figure, shaded nodes represent observed variables, unshaded nodes represent hidden variables, and edges represent conditional dependencies between variables. A

graphical model with the graph structure depicted by Figure 2.1 and parametrised by Equations 2.1 is known in the machine-learning literature as a Linear Dynamical System (LDS) and is widely used to model continuous sequential data. We use the LDS as our basic modelling component and extend it in order to deal with multiple and unexpected dynamic behaviours, and data association ambiguity.

### 2.1.2 Multi-Object Tracking and Classification: Modelling Considerations

Due to the multi-class nature of the objects in most dynamic environments, a robotic-perception system must provide, not only quantitative object descriptors in the form of trajectories, but also qualitative ones in the form of class labels. In other words, the system must have a reliable multi-object tracking and classification module in place. This section poses multi-object tracking and classification as a state-estimation problem while it also explains the modelling considerations and concepts that constitute the foundations of our approach.

#### Observation Features

In a similar manner to other estimation problems such as localisation or Simultaneous Localisation and Mapping (SLAM), a solution for the simultaneous tracking and classification problem is expected to process sensor measurements in order to obtain an filtered version of the original measurements, or represent new knowledge. Typically, raw sensor measurements are not fed to the tracking algorithm directly, but are converted into observation features. This is performed by the tracking algorithm, or in a preprocessing step called "object detection". Observation features are a modified version of the sensor measurements that accomplish one or several of the following functions:

- Reduce the dimensionality of the available data. For instance, statistical moments of colour histograms.

- Extract information which has a physical meaning to the user. For instance, centroid locations obtained from point clouds or orientations calculated from centroid pairs.

### Object States

Object tracking algorithms aim at estimating the state of the objects in the field of view of the platform's sensor(s). The typical state space in object tracking is composed of locations and velocities. Nevertheless, variables that model object appearance such as height, width or some statistics of the object's colour and/or texture, can also be desired. In this thesis, object states correspond to a filtered/smoothed version of the observation features, and in the case of position observations, their first derivatives as well. The general form of the state vector we consider throughout this thesis is as follows:

$$x_t = \begin{bmatrix} p_x \\ v_x \\ a_x \\ p_y \\ v_y \\ a_y \\ p_z \\ v_z \\ a_z \\ a_1 \\ \vdots \\ a_n \end{bmatrix}_t , \tag{2.2}$$

where $t$ is the discrete time index, $(p_x, v_x, a_x)$, $(p_y, v_y, a_y)$ and $(p_z, v_z, a_z)$ are the position, velocity and acceleration of the object in the x, y and z coordinates, respectively, and $a_1$ to $a_n$ are appearance variables. This type of state vector is also known in the tracking literature as "extended target state vector" [16].

### Motion Models

In order to develop a framework for estimating the state vector $x_k$, it is necessary to model how this quantity evolves over time. The temporal correlation in object motion is usually modelled using a transition matrix that represents the relationship between the previous kinematic state and the current state of an object. The most common motion models in the tracking literature are the nearly-constant-velocity and nearly-constant-acceleration models, which are particular cases of the Singer model [17]. In the Singer model, the object acceleration is modelled as a first-order stationary Markov process, and its discrete version is given by:

$$\begin{bmatrix} p \\ v \\ a \end{bmatrix}_t = \begin{bmatrix} 1 & T & \frac{T^2}{2} \\ 0 & 1 & T \\ 0 & 0 & \beta \end{bmatrix} \begin{bmatrix} p \\ v \\ a \end{bmatrix}_{t-1} + \begin{bmatrix} \frac{T^2}{2} \\ T \\ 1 \end{bmatrix} w_t, \tag{2.3}$$

where $p$, $v$ and $a$ are position, velocity and acceleration in one dimension respectively, $w_t$ is a zero-mean Gaussian random vector with covariance matrix $Q$, $T$ is the sampling interval and $\beta = e^{-\alpha T}$. $\alpha$ is the reciprocal of the manoeuvre time constant. This manoeuvre time constant is indicative of how long the manoeuvre encoded by the model lasts. Note that a nearly-constant-acceleration motion model can be obtained by setting $\beta$ to one; and a nearly-constant-velocity model can be obtained by setting $\beta$ to zero.

### Observation Model

The observation model represents the functional relationship between object states and observation features. Its linear discrete state-space notation has the form:

$$z_t = Hx_t + v_t, \tag{2.4}$$

where $v_t$ is a zero-mean Gaussian random vector with covariance matrix $R$, and $H$, the observation matrix, is a matrix with as many rows as the number of features and

as many columns as the number of states. In the case that the state variables have a one-to-one relationship with the observation features, $H$ is the identity matrix.

The rest of this section provides the definition of the concepts from probability theory that will be used throughout the thesis as well as the probability distributions used in our model.

### 2.1.3 Probability Distributions

#### Definitions from Probability Theory

Prior Probability Distribution: A probability distribution that represents the uncertainty about the quantity of interest before any evidence about its actual value has been collected.

Posterior Probability Distribution: A probability distribution that represents the uncertainty about the quantity of interest after some evidence about its actual value has been collected.

Inference: The processes of estimating a posterior probability distribution.

Likelihood: The probability of an observation as a function of the model parameters.

#### The Multivariate Gaussian Distribution

The LDS model accounts for the fact that sensors, in general, produce a noisy and incomplete version of the property being measured. It models the actual state $x_t$ as being drawn from a multivariate Gaussian distribution, a widely used model for continuous random variables. Let $\mu$ be an $m$-dimensional vector and $\Sigma$ be an $m$ x $m$ symmetric positive-definite matrix. The random vector $x$ is Gaussian distributed with mean $\mu$ and covariance matrix $\Sigma$ if its Probability Density Function (PDF) is

given by:

$$\mathcal{N}(x;\mu,\Sigma) = (2\pi)^{-m/2} |\Sigma|^{-1/2} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right). \tag{2.5}$$

### The Categorical Distribution

The model presented in this thesis builds on the basic LDS, and extends it in order to account for multiple dynamic behaviours. This is done by means of a mixture model where each component is an LDS that models a particular behaviour. The first module of a mixture model is a categorical random variable with a support equal to the set of expected classes. We will also use categorical random variables to model the assignment between observations and object trajectories. The categorical distribution concerns a random variable that takes values in this finite set. If a random variable $s$ can take one value out of the finite set $\{1, 2, ...j, ..., N_s\}$, where $N_s$ is the number of categories, and $p_j$ is the probability of the event $s = j$, then $s$ is a categorical random variable with a Probability Mass Function (PMF) given by:

$$\mathcal{C}(s;p) = \prod_{j=1}^{N_s} p_j^{\delta(s,j)}, \tag{2.6}$$

where $\delta$ is the Kronecker's delta function, defined as:

$$\delta(x,y) = \begin{cases} 1 & \text{if } x = y \\ 0 & \text{if } x \neq y. \end{cases}$$

Figure 2.2 shows an example of the PMF of a categorical random variable with four classes.

### The Multivariate $t$ Distribution

Another component of the framework we present in this thesis is unexpected behaviour modelling. Unexpected observations can be thought of as data points that

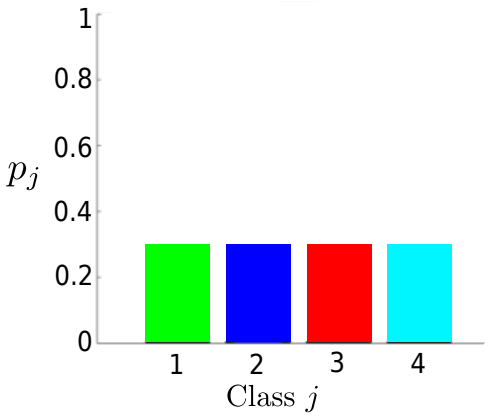



**Figure 2.2** – Probability Mass Function (PMF) of a categorical distribution with four classes.

fall outside the bulk of the data. The multivariate $t$ distribution has properties that make it appropriate for robustly modelling such kind of phenomenon. It has a bell-shaped PDF similar to the Gaussian PDF but with heavier tails, so as illustrated in Figure 2.3, it assigns a high (higher than in the Gaussian distribution) probability density to regions in the variable's domain that are far from the bulk of the data. The weight of the tails is controlled by the degrees of freedom $v$. The smaller $v$ is, the more the probability mass spreads across the sample space.

The random variable $x$ is $t$-distributed with location $\mu$, scale $\Sigma$, and $v$ degrees of freedom if its PDF is as follows:

$$St\left(x;\mu,\Sigma,v\right)=\frac{\Gamma\left(\frac{v+m}{2}\right)}{\Gamma\left(\frac{v}{2}\right)}\frac{1}{\left(v\pi\right)^{\frac{m}{2}}}\frac{1}{\sqrt{\det\left(\Sigma\right)}}\left(1+\frac{1}{v}\left(x-\mu\right)^{T}\Sigma^{-1}\left(x-\mu\right)\right)^{-\frac{m+v}{2}}, \tag{2.7}$$

where $\Gamma\left(\bullet\right)$ is the Gamma function, defined as:

$$\Gamma\left(\alpha\right)=\int_{0}^{\infty}u^{\alpha-1}\mathrm{e}^{-u}du. \tag{2.8}$$

Using the PDF in Equation 2.7 renders marginalisation analytically intractable, however, the Gaussian-mixture interpretation of the $t$ distribution enables the definition of likelihood functions that can be optimised using the EM algorithm [18]. The $t$

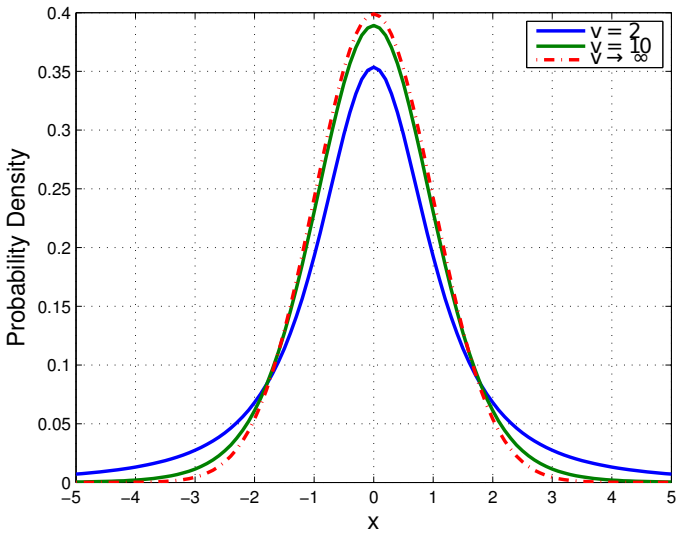



**Figure 2.3** – PDFs of $t$ distributions with different number of degrees freedom.

distribution can be viewed as a mixture of an infinite number of Gaussian random variables with identical means and scaled covariances [19]:

$$\mathcal{N}\left(x;\mu,\Sigma,\omega\right)=\left(2\pi\right)^{-m/2}\left|\Sigma\right|^{-1/2}\omega^{m/2}\exp\left(-\frac{\omega}{2}\left(x-\mu\right)^{T}\Sigma^{-1}\left(x-\mu\right)\right), \tag{2.9}$$

where $w$ is a Gamma-distributed random variable with shape and rate parameters $\alpha=\beta=\frac{v}{2}$:

$$w\sim\mathcal{G}\left(\omega;v/2,v/2\right).$$

### The Gamma Distribution

The choice of parametrisation we have made for the $t$ distribution requires us to draw a precision weight for the scale parameter from a Gamma distribution. This right-skewed distribution is widely used in Bayesian statistics for specifying a prior probability density for the precision parameter of a Gaussian distribution, as it is only defined for positive values.

A continuous random variable $\omega$ is Gamma distributed with shape parameter $\alpha>0$

and rate parameter $\beta > 0$ if its PDF is

$$\mathcal{G}(\omega; \alpha, \beta) = \frac{1}{\Gamma(\alpha)} \beta^{\alpha} \omega^{\alpha-1} \exp(-\beta\omega). \tag{2.10}$$

It is worthwhile noting that the mean of this random variable is:

$$\langle w \rangle = \frac{\alpha}{\beta}. \tag{2.11}$$

Figure 2.4 illustrates the PDF of a Gamma distribution for multiple values of the shape parameter $\alpha = [1, 2, 3]$ and rate parameter $\beta = 1$.

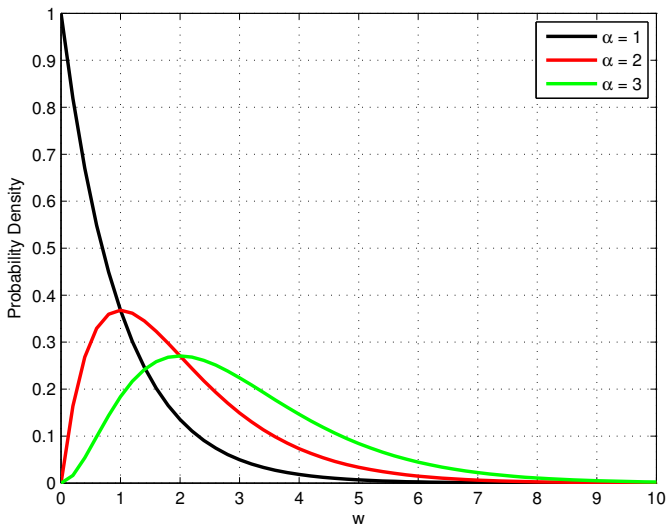



**Figure 2.4** – Probability Density Function (PDF) of a Gamma distribution.

### The Matrix-Variate Normal Distribution

The matrix-variate normal distribution is the generalisation of the multivariate Gaussian distribution to matrix-valued random variables. A real-valued random matrix

$$H \in \mathcal{R}^{n \times m}, \tag{2.12}$$

is distributed according to the matrix-variate normal distribution with a $n \times m$ location parameter $\Lambda$, a $n \times n$ row scale parameter $R$ and a $m \times m$ inverse column scale parameter $\Omega$ if its PDF is

$$\mathcal{N}(H|\Lambda, R, \Omega) = \frac{|\Omega|^{\frac{n}{2}}}{(2\pi)^{\frac{nm}{2}}|R|^{\frac{m}{2}}} etr\left[-\frac{1}{2}(H-\Lambda)^T R^{-1}(H-\Lambda)\Omega\right]. \tag{2.13}$$

$R$ and $\Omega$ are proportional to the among-row covariance and among-column precision matrices respectively. The mean, row covariance matrix and column covariance matrix of a random matrix following this distribution are given by:

$$\begin{aligned} \langle H \rangle &= \Lambda, \\ \left\langle (H-\Lambda)(H-\Lambda)^T \right\rangle &\propto \Sigma, \\ \left\langle (H-\Lambda)^T(H-\Lambda) \right\rangle &\propto \Omega^{-1} \end{aligned} \tag{2.14}$$

**The Inverse-Wishart Distribution**

Estimating covariance matrices is a recurrent problem, not only in statistics, but in many other areas in the sciences and engineering [20–22]. Standard estimation methods such as calculating the sample covariance or maximum likelihood (see Section 2.1.5) are prone to deliver unstable estimates, i.e. covariance matrices that are not positive definite. This issue has been approached by using the Inverse-Wishart distribution as a conjugate prior in order to yield shrinkage of the estimated covariance matrix towards a structure that ensures its positive definitiveness [23, 24].

The inverse Wishart defines a probability distribution over real-valued positive-definite matrices. We say $Q$ follows an inverse Wishart distribution if its PDF is

$$\mathcal{W}^{-1}(Q|\nu\Sigma, \nu) = \frac{\frac{\nu}{2}^{\frac{m\nu}{2}}|\Sigma|^{\frac{\nu}{2}}}{\pi^{\frac{m(m-1)}{4}}\prod_{k=1}^{m}\Gamma\left(\frac{\nu+1-k}{2}\right)}|Q|^{-\frac{\nu+m+1}{2}} etr\left[-\frac{\nu}{2}\Sigma Q^{-1}\right]. \tag{2.15}$$

where $etr(X) \to \exp(tr(X))$. $\Sigma$ and $\nu$ are the scale and the degrees of freedom of the inverse Wishart distribution respectively.

In this thesis, the inverse-Wishart distribution is used to constraint the space of possible realisations of the covariance matrices that constitute our model.

### 2.1.4 Approximate Inference

For LDS models inference can easily be performed using Kalman [25] or particle filtering [26]. However, for richer models, certain interactions between variables can make inference become computationally and/or analytically intractable. In [27, 28] and in Chapter 3 of this thesis, extensions to deal with multiple dynamic behaviours and data association ambiguities on the standard LDS are presented. Accounting for unknown data association results in models on which performing inference is intractable. In this kind of situation, approximate-inference techniques based on either sampling or variational methods are used.

Sampling methods, a.k.a. Monte Carlo approximations [29], allow to perform posterior inference by repeatedly generating samples from the posterior of interest. Sampling methods can be computationally demanding and thus prohibitively slow [15]. They can be slow to converge [30] or fail to converge when not enough data is available in relation to the complexity of the model [31]. The design of efficient proposal distributions for the importance sampling stage make their application to trajectory estimation and data association problems greatly restricted when the size of the state space is large. The work in [32] introduces an approach that performs sampling-based inference on Segmental Switching Linear Dynamical System (S-SLDS) models, where a data-driven approach is used to generate these proposals. This kind of approach can be troublesome if the training data are noisy. Other approaches resort to deterministic approximations such us the Kalman filter in order to generate this proposals [33].

Variational approximations, on the other hand, allow us to perform posterior inference by defining a family of distributions that have the potential of approximating the distribution of interest, and then optimising a measure of fitness to the data as a function of the parameters of the targeted family of distributions. Variational approaches

produce inference procedures that are usually involved, and more difficult to implement if compared to their sampling counterparts, however, due to their deterministic nature, they are faster and are guaranteed to converge [34]. In [27, 28] we introduced the Expectation Association (EA) algorithm, a variational approach to inference on a model for sequential multi-object simultaneous tracking and classification that takes advantage of the computational and theoretical guaranties of variational methods. The inference machinery developed in this thesis is based on this previous work and utilises the variational paradigm as its theoretical foundation.

### Variational Inference

Variational inference, which is also referred to as "deterministic-approximate inference", is a family of methods used to derive flexible approximations for complex probability distributions. Variational inference makes it possible to simplify a PGM into a form where exact inference is tractable.

The fundamental idea is to optimise a measure of discrepancy between a complex distribution $p(x)$ and a simpler distribution (variational distribution) $q(x)$, used to approximate it. The Kullback-Leibler(KL) divergence [35] is a popular measure of this discrepancy and is defined as:

$$KL(q|p) \equiv \langle \ln q(x) - \ln p(x) \rangle_{q(x)} \geq 0. \tag{2.16}$$

Let our complex distribution be the posterior $p(x, a|z)$, and let $q(x, a)$ be the variational approximation. The KL divergence between $q$ and $p$ is:

$$KL(q|p) \equiv \langle \ln q(x, a) - \ln p(x, a|z) \rangle_{q(x,a)} \geq 0. \tag{2.17}$$

By rewriting $p(x, a|z)$ as $p(x, a, z)/p(z)$, we have:

$$KL(q|p) \equiv \langle \ln q(x, a) \rangle_{q(x,a)} - \langle \ln p(x, a, z) - \ln p(z) \rangle_{q(x,a)} \geq 0. \tag{2.18}$$

Since the marginal probability $p(z)$ is independent of both $x$ and $a$, the KL divergence provides a lower bound on $\ln p(z)$:

$$\ln p(z) \geq \langle \ln p(x,a,z) \rangle_{q(x,a)} - \langle \ln q(x,a) \rangle_{q(x,a)} \equiv \mathcal{L}\{q(x,a)\}. \tag{2.19}$$

The approximate inference problem is therefore translated into a two-step procedure. Firstly, a family of distributions $q(x,a)$ is chosen such that the lower bound $\mathcal{L}\{q(x,a)\}$ is computationally tractable. Secondly, the free parameters of $q(x,a)$ are set so that $\mathcal{L}\{q(x,a)\}$ is maximised (see Section 2.1.4).

### Factorised Approximations

Intuitively, the first attempt at reducing the complexity of a probability distribution involves relaxing the dependency between its variables. Approximating a complex distribution with a product of independent factors is an idea borrowed from the *mean field theory* in physics [36]. Let us define the variational distributions as:

$$q(x,a) = q(x)\, q(a). \tag{2.20}$$

Thus, the lower bound $\mathcal{L}\{q(x,a)\}$ in Equation 2.19 can be written as:

$$\mathcal{L}\{q(x,a)\} \equiv \langle \ln p(x,a,z) \rangle_{q(x)q(a)} - \langle \ln q(x) \rangle_{q(x)} - \langle \ln q(a) \rangle_{q(a)}. \tag{2.21}$$

The sub-indices in the expected value expressions indicate the distribution under which the expectation is taken. Let us dissect out the dependence on the factor $q(x)$:

$$\begin{aligned} \mathcal{L}\{q(x,a)\} &\equiv \Big\langle \langle \ln p(x,a,z) \rangle_{q(a)} \Big\rangle_{q(x)} - \langle \ln q(x) \rangle_{q(x)} - \langle \ln q(a) \rangle_{q(a)} \\ &= -\mathrm{KL}\left( q(x) | \exp\left( \langle \ln p(x,a,z) \rangle_{q(a)} \right) \right) - \langle \ln q(a) \rangle_{q(a)}. \end{aligned} \tag{2.22}$$

As shown in Equation 2.22, $\mathcal{L}\{q(x,a)\}$ is, up to a normalisation constant, the KL divergence between $q(x)$ and a distribution proportional to $\exp\left( \langle \ln p(x,a,z) \rangle_{q(a)} \right)$.

Therefore, maximising the lower bound is equivalent to minimising this KL divergence, whose minimum occurs when

$$\ln q(x) = \langle \ln p(x, a, z) \rangle_{q(a)} + \eta(a), \tag{2.23}$$

where $\eta(a)$ is a normalisation constant independent of $x$. Similarly, for $q(a)$, we have

$$\ln q(a) = \langle \ln p(x, a, z) \rangle_{q(x)} + \eta(x). \tag{2.24}$$

By initialising $q(x)$ and $q(a)$, and iteratively updating these distributions using Equation 2.23 and Equation 2.24, the lower bound Equation 2.27 is maximised. Example 2.1 shows a simple example of variational inference on a three-node PGM where we relax a conditional dependence in the posterior, similar to the one we relax in the inference machinery that this thesis develops.

**Example 2.1**

*This is an example of factorised approximate inference on a three-node PGM with a v structure [14]. Similar to the example in Figure* 8.21 *of [15], let us consider a PGM representation of the fuel system on a car. In Figure 2.5, the binary random variable $x$ represents the state of a battery as being either flat ($x = 0$) or charged ($x = 1$); $a$ represents the state of the fuel tank as being either empty ($a = 0$) or full ($a = 1$); $z$ represents the state of an electric fuel gauge that indicates* 0 *for empty and* 1 *for full.*

*The parameters of this PGM are given by the following probability tables:*

$$\begin{aligned} p(x) =& [p(x=0),\ p(x=1)] \\ =& [0.1,\ 0.9]; \\ p(a) =& [p(a=0),\ p(a=1)] \\ =& [0.1,\ 0.9]; \\ p(z=1|x,a) =& [p(z=1|x=0,a=0),\ p(z=1|x=0,a=1),\ p(z=1|x=1,a=0), \\ & p(z=1|x=1,a=1)] \\ =& [0.1,\ 0.2,\ 0.2,\ 0.8]. \end{aligned} \tag{2.25}$$

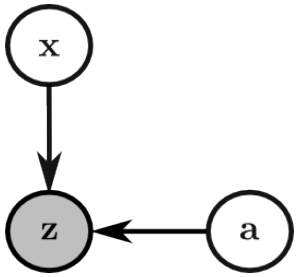
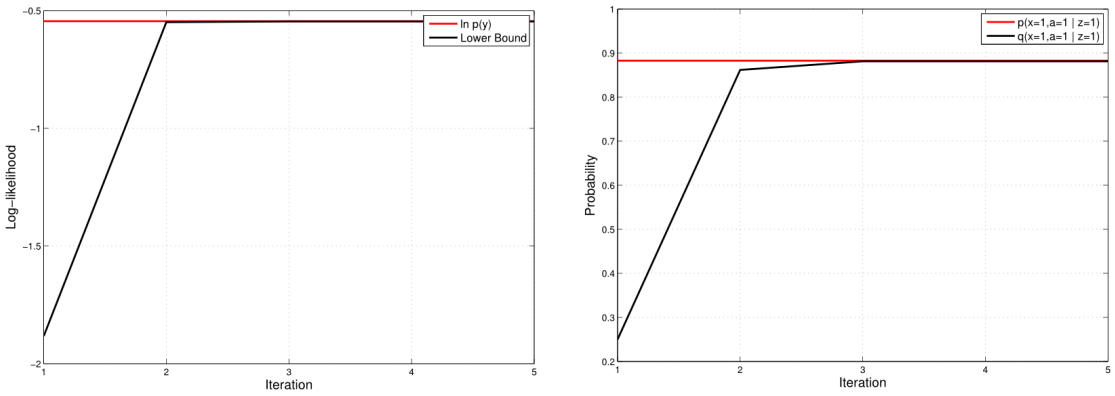



**Figure 2.5** – A three-node PGM with a *v* structure. In this image, shaded and unshaded nodes represent observed and hidden variables respectively.

*According to the notion of d-separation [14], x and a are conditionally dependent given z. To To illustrate the core idea of variational inference, we relax this dependency and approximate the posterior as:*

$$p(x,a|z) \approx q(x)q(a). \tag{2.26}$$

*Initialising $q(x)$ and $q(a)$ as uniform distributions and sequentially updating them as in Equation 2.23 and Equation 2.24 maximises the lower bound (Equation 2.19) on the likelihood as shown in Figure 2.6a*




**(a)** Exact likelihood and its lower bound for Example 2.1

**(b)** Exact and approximate posterior probability $x = 1, a = 1 \quad | \quad z = 1$

**Figure 2.6** – Exact likelihood, lower bound, and an entry of the exact and approximate posterior distribution in Example 2.1. The lower bound in Figure 2.6a increases at each iteration and converges at $t = 3$. Also note that the approximate posterior in Figure 2.6b converges to a value that is very close to the exact one.

### 2.1.5 Model Learning

Once the structure of our PGM has been defined (Section 2.1.1), its parameters must be learned. The underlying idea behind learning a PGM is specifying the set of parameters that define the model. For the model in Example 2.1, model learning boils down to finding the probability tables $p(x)$, $p(a)$ and $p(z|x, a)$ from a set of observations of the variable $z$. In models where all variables can be observed, parameters are learned by maximising the likelihood of the data. These kinds of methods are also called Maximum Likelihood (ML) approaches.

In the presence of hidden variables, calculating the marginal likelihood requires marginalisation over these hidden variables. In general, the summations added by the marginalisation procedure cause the parameters to be tightly coupled, and as a consequence the likelihood cannot be factorised into a product of factors. The *Expectation Maximisation Algorithm* [37] allows us to overcome this problem and calculate the parameters of models with hidden variables in an iterative procedure that optimises a lower bound on the marginal likelihood. PGMs that include hidden variables are also known as Latent Variable Models (LVM).

### The Expectation Maximisation Algorithm

The task of learning the parameters of a probabilistic model can be formulated as an optimisation problem. In the case of variational methods, the cost function is a lower bound on the log-likelihood. Let us define the lower bound on the marginal likelihood as:

$$\ln p(z|\Omega) \geq \langle \ln p(x, z|\Omega) \rangle_{q(x|z)} - \langle \ln q(x|z) \rangle_{q(x|z)} \equiv \mathcal{L}\{q\}. \tag{2.27}$$

The Expectation Maximisation (EM) algorithm is a two-step iterative procedure that, instead of maximising the likelihood function, maximises the expected complete log-likelihood $\langle \ln p(x, z|\Omega) \rangle_{p(x|z)}$. If instead of the exact posterior $p(x|z)$, only the approximate posterior $q(x|z)$ is available, EM aims at maximising the lower bound in Equation 2.27. The Expectation step (E-step) and the Maximisation step (M-step) can be implemented as follows:

**E-step**: Fix the parameters $\Omega$ and find the distribution $q(x|z)$ that maximises Equation 2.27. If we have access to the exact posterior over hidden variables $p(x|z, \Omega^{old})$, we use that distribution obtained with the parameters from the previous iteration.

**M-step**: Find the parameters $\Omega$ that maximise Equation 2.27, which is equivalent to maximising the expected complete log-likelihood $\langle \ln p(x, z|\Omega) \rangle_{q(x|z)}$ with respect to each of the parameters. In [15, 35] it is shown that each cycle of EM increases the marginal log-likelihood.

The EM algorithm has been widely utilised for learning models for spatio-temporal data. For instance, [38] presents an EM algorithm for learning the parameters of an MLDS, whereas [27, 28] uses it for learning the parameters of a model for multi-object tracking and classification.

### The Need for Regularisation

Models whose parameters are estimated using data are prone to over-fitting. This problem is approached using *regularisation*, i.e., by adding a penalty term to the

objective function that constrains the parameter space. The most popular forms of regularisation use L1 [39] or L2 [40] norms. Regularisation is performed by adding a penalty term to the objective function composed by the norm of the model parameters.

$$\begin{aligned} \text{L1: } R(\Omega) &= \|\Omega\|_1 = \sum_j |\Omega_j| \\ \text{L2: } R(\Omega) &= \|\Omega\|_2^2 = \sum_j \Omega_j^2. \end{aligned} \tag{2.28}$$

Bayesian regularisation [41] corresponds to a more general perspective where the model complexity is penalised with a prior distribution over model parameters. Both L1 and L2 norms can be interpreted as particular cases of the Bayesian case with L1 usually being parametrised as a Laplacean prior and L2 as a Gaussian prior [42]. The Bayesian regularisation term has therefore the following general form:

$$\text{Bayesian: } R(\Omega) = -\ln p\left(\Omega\right) \tag{2.29}$$

**The Regularised Expectation Maximisation Algorithm**

The EM algorithm can also be used to obtain Maximum A-Posteriori (MAP) estimates by means of Bayesian regularisation. Here, parameters are regarded as random variables, which contrasts with Maximum Likelihood (ML) procedures where free parameters are obtained. The difference between the MAP-EM and ML-EM algorithms lies in the M-step:

**E-step**: Evaluate $p(x|z, \Omega^{old})$ as in the ML-EM algorithm,

**M-step**: Find the parameters $\Omega$ that maximise:

$$\mathcal{Q}\left(\Omega\right) = \langle \ln p(x, z|\Omega) \rangle_{p(x|z,\Omega^{old})} + \ln p(\Omega). \tag{2.30}$$

The estimates obtained in this version of the M-step are posterior distributions over the parameters rather than point estimates. Additionally, the prior $p(\Omega)$ works as a

regulariser on the EM cost function. It allows us to constrain the parameter space, so that the parameter estimates are robust to outlier observations and less prone to overfitting. This is important when there are a small number of training observations, observations are very noisy, or when some of them are statistical outliers.

### 2.1.6 Evaluation Metrics

Tracking performance is usually evaluated using the *Multiple Object Tracking Accuracy* (MOTA) [43] and the *Mostly-Tracked (MT) / Mostly-Lost (ML) trajectories* metrics [44]. This section explains the intuition behind these metrics.

The MOTA metric is calculated as:

$$MOTA = 1 - \frac{\sum_t (fn_t + fp_t + mme_t)}{\sum_t g_t}, \tag{2.31}$$

where, $fn_t$, $fp_t$, $mme_t$ and $g_t$ are the number of false negatives, false positives, mismatches and ground truth objects respectively for time $t$. The MOTA metric is calculated by assigning to each ground-truth state the closest state from estimated trajectories and then getting the proportion of, firstly, cases when ground-truth states did not have an estimated state assigned to them ($fn_t$), secondly, instances of estimated states that were not assigned to ground-truth states ($fp_t$) and, finally, instances of ground-truth trajectories to which states from different estimated trajectories were assigned ($mme_t$). This metric evaluates the ability of tracking systems to correctly associate detections across time.

The $MT$ and $ML$ metrics provide the percentage of ground-truth trajectories that were covered by the estimated trajectories for more than 80% and less than 20% in length, respectively. These metrics measure the property that, in many cases, objects are correctly tracked during a period of time shorter than its actual life span.

## 2.2 Literature Review

A brief description of the mathematical foundations of the proposed approach was presented in previous sections. This section reviews the use of PGMs for dynamic scene understanding tasks. It also covers the related work on motion detection, tracking and classification.

### 2.2.1 PGMs for Dynamic Scene Understanding

As explained in Section 2.1.1, PGMs provide a flexible modelling framework that makes explicit the structure of the estimation problem, and this can be in turn used to devise powerful and efficient learning and inference techniques. This subsection briefly reviews seminal works on the use of PGMs in the area of dynamic scene understanding.

The work in [45, 46] presents an approach for inferring a scene's topology, geometry and traffic activities from short video sequences. They introduce a generative model that reasons about the 3D layout of a dynamic scene as well as the location and orientation of objects in the scene. This model relates the geometry of an intersection to visual features, such as vehicle tracklets, vanishing points, semantic scene labels, scene flow and occupancy grids. For learning the model, as the partition function of the joint distribution is intractable, they reformulate the problem as a Gibbs random field and learn the model parameters using a sampling based ML approach derived from MCMC called contrastive divergence [47]. They achieve an improvement on state-of-the-art object-detection- and object-orientation estimation by using the context derived from the proposed method.

The authors of [48] developed a method for inferring the interactions between pedestrians and the objects they carry. They introduced a Bayesian network where the hidden states are object types and piecewise interactions, and the observed variables are relative positions and velocities, and appearance observations. The dynamic observations were obtained from a tracking module based on low-level stereo region segmentation

and multi-hypothesis data association, whereas exact inference was performed using the junction-tree algorithm [14]. For the data-association module, the similarity measurements between tracks and observations were obtained by registering point clouds using the Iterative Closest Point (ICP) algorithm [49].

### 2.2.2 Multi-Object Tracking

Multi-Object Tracking (MOT) is a procedure that provides information about what objects of interest are in the environment and their behavioural characteristics [50]. MOT is a well-known problem in the robotics community and many publications on the matter have been produced [51–55]. Most of the current approaches to tracking follow a *tracking-by-detection* methodology, where objects of interest are first detected at each frame [56, 57], then detections are linked to object hypotheses across frames, and finally, trajectories associated to object hypotheses are estimated.

Depending on whether detections are assumed to be perfect measurements of the object state or not, two approaches to tracking exist. The first one defines trajectories in terms of subsets of detections that follow some smoothness constraints [58, 59]. These kinds of approaches are effective at recovering object identities after short-term occlusions, and have an outstanding performance when little or no observation noise is present. However, due to their deterministic nature, their performance usually drops significantly when applied to noisy sensors. Additionally, they do not model unobserved states, therefore queries like object velocities cannot be estimated when only position observations are available.

The second group of approaches to tracking considers object states as latent variables that need to be estimated from incomplete and noisy observations [60–63, 46, 64]. The incomplete and noisy nature of the data can be modelled by means of Latent Variable Models (LVMs). LVMs allow us to estimate quantities that are not directly observed, such as velocities and interactions between scene objects [65, 48]. They can also obtain estimates that are robust against noise, and as it will be shown, they provide tools for probabilistic modelling of unexpected observations.

Approaches based on Random Finite Set (RFS) statistics [66] are also part of this latter group. In RFS approaches, both the object states and number of objects are modelled as random sets. In [67], the Probability Hypothesis Density (PHD) filter was proposed as an approximation of the multi-object Bayes filter using RFS statistics. The Gaussian Mixture PHD (GM-PHD) filter is an implementation where the PHD filter is approximated using Gaussian Mixtures [68].

### The Data Association Problem

When objects in the environment have GPS receivers or identity markers, there is always a known one-to-one correspondence between these objects and the observations obtained from the sensors. However, in most environments where autonomous robots are deployed, the objects that they interact with do not have identity markers, so no deterministic assignment between object hypothesis and new observations is available. This situation introduces one of the most challenging and studied problems in the tracking literature: how do we associate observations to object hypotheses?, known as the *Data Association* (DA) problem.

One of the first attempts to solve the DA problem was the Multi-Hypothesis Tracking (MHT) method [69]. This algorithm maintains a set of hypotheses defined by all the possible associations between observations across time. Each hypothesis corresponds to a sequence of Gaussian Mixtures (GMs), where each mixture represents the state of an object and its observed value, and it is sequentially updated by means of Kalman Filtering. The Kalman filter provides an estimate of the GM at time $t$ as a function of all the associated observations up to time $t$ for a given hypothesis.

Frequently, we would like to estimate each GM based on all the available associated observations. In other words, if $T$ is the size of the hypothesis, we would like to estimate the state at time $t$ based on the observations from time 1 to $T$. The process of obtaining these complete estimates is referred to as *smoothing*. For linear-Gaussian models, smoothing is implemented using the Rauch-Tung-Striebel (RTS) recursions [70].

The number of hypotheses that need to be filtered/smoothed grows combinatorially with the number of objects, and exponentially with time, so pruning heuristics have been used to reduce the computational burden [71] of updating all the object hypotheses. These pruning heuristics may potentially prune correct hypotheses, leading to a degenerate hypothesis tree, where running smoothing leads to wrong estimates of the object states. This issue is caused by the fact that a degenerate tree is equivalent to using the wrong data association for state estimation. In [72] a single-object tracking approach is introduced where this problem is tackled by using forward-backward smoothing where hypotheses are merged rather than pruned.

A different take on the problem of data association was presented in the work of [73]. It introduces the Probabilistic Data Association Filter (PDAF) for single-object tracking, where observations are not discarded but weighted and used to update the object's state. In [74] the *Joint Probabilistic Data Association Filter* (JPDAF) was introduced, which is an extension of PDAF that deals with multiple objects. Additionally, the work in [75] adapted JPDAF so that overlap between objects is described in the context of visual tracking. Although JPDAF is more efficient than MHT, its complexity still grows exponentially with the number of objects. The *Probabilistic Multiple Hypothesis Tracker* (PMHT) [76, 77] is a linear-complexity approach that is based on the Expectation Maximisation algorithm.

Developing methods that deal with the computational complexity of the DA problem is a core topic in both the SLAM and tracking literature. The work in [78] introduced the concept of reversible DA in the context of SLAM for dynamic environments. The method proposed a least-squares-based approach to SLAM that accounts for moving objects and performs robust estimation across multiple time steps. [79] adopted the concept of reversible DA and applied it to multi-object tracking in surveillance applications. The work developed a sampling approach for performing inference that attains online performance thanks to a GPU-based multi-threaded architecture that parallelises detection, sampling, data association and output generation.

This thesis proposes a graphical model and an inference procedure that performs sliding window estimation in close resemblance to the ones used in [78] and [79]

but instead of sampling, it uses a deterministic inference. Probabilistic Graphical Models (PGMs) allow us to approach the tracking problem from a wider perspective. Representing the object tracking problem as a PGM makes it easier to develop efficient inference techniques by introducing expert knowledge into the model. By considering the structure in object trajectories and the representational power of PGMs, the works in [80–83] formulate the DA problem as a factor graph and apply message-passing techniques [84] in order to perform efficient inference on their models.

In many cases, observations about the position of the objects may be accompanied by appearance measurements such as shape, texture or colour. The work in [27, 28] and Chapter 4 of this thesis develops a tracking approach that integrates both appearance from image patches, and dynamics from stereo-vision point clouds. It uses the appearance information to initialise data association probabilities and then updates both these probabilities and the object states with the position measurements.

#### Multiple-model Approaches to Object Tracking

Central to the task of object tracking is the definition of both an observation model and a motion model. The former describes the relationship between observations and states, whereas the latter describes the temporal evolution of the states. Adopting a good model has been shown to result in tracking approaches that outperform any model-free tracking algorithm [85].

There are cases in which only one model is not enough. Multiple-model approaches have been extensively used in environments where individual objects go through multiple motion behaviours/modes (cruising, turning) [86]. They are commonly known as *Interacting Multiple Model* (IMM) methods [87]. These kinds of methods model an object trajectory as a realisation of a Switching Linear Dynamical System (SLDS) [35].

Another use of MM approaches, which has only been recently introduced, is that of tracking multiple objects with different dynamics. In [27, 28] multiple motion models are used to account for the diverse nature of object behaviours (cars, cyclists, pedes-

trians) in urban environments. These kinds of methods model an object trajectory as a realisation of a Mixture of Linear Dynamical Systems (MLDS) [88], where each mixture component describes the dynamic behaviour of one object category.

### 2.2.3 Spatio-temporal Object Classification

Object Classification based on spatio-temporal information has been typically approached using either similarity-based clustering/classification techniques [89–91] or model-based approaches [92–94]. Note that these model-based approaches usually have a clustering component to them in the front-end.

The work in [90] uses an adapted version of Affinity Propagation [95] on a new type of feature they introduced for summarising the shape of laser tracks. They called their feature *Laser Stamps* and also defined an associated measure used to compare the similarity between descriptors.

The work in [92] segments individual trajectories into sub-trajectories by splitting them at points of change in curvature. The set of sub-trajectories are summarised by their principal components, whose coefficients are in turn used to learn a Gaussian Mixture Model (GMM) representing atomic activities. Finally, a Hidden Markov Model (HMM) is used to perform activity classification by modelling trajectories as sequences of the atomic activities. In [93] an approach is presented which is similar to the one introduced by [92], in that it models trajectories as a sequence of atomic actions. However, it includes a spatial component to the classification task by defining points of entry and exit before learning the HMM-based classifier.

#### Outlier Detection

There are cases in which the objects to be classified do not fit any of the considered classes. These objects, usually referred to as outliers or anomalies, can be understood as observations or patterns in the data that do not conform to any of the expected behaviours [96]. In the context of object tracking, anomalies may arise at two different

levels. At an observation level, sensor failures cause *point anomalies*. At an object level, unexpected behaviours generate *trajectory anomalies* [97].

Due to the increasing use of robotic platforms in urban and natural environments, it is desirable that a robot can estimate not only the class of an object, but also whether its behaviour is normal or not according to what has been previously observed. Telling anomalous objects apart is useful and may point out a dangerous interaction or a new behaviour. This process is called *anomaly/novelty detection* [98]. Discovering anomalies from unlabelled data is known as *unsupervised anomaly detection* [96].

The importance of LVMs was explained in Section 2.2.2. From a distribution-theoretic point of view, equipping an LVM with outlier-detection capabilities boils down to including a random variable that explicitly models the abnormality of an observation. The work in [99] introduces the *Robust Probabilistic Multivariate Calibration Model* (RPMC). It is an extension of *Probabilistic Principal Component Analysis* PPCA [100] whose components have a $t$ distribution instead of a Gaussian distribution. In [101], the capabilities of RPMC for dealing with incomplete observations are investigated. By applying the same idea, [102] develops a robust inference procedure for LDS models with heavy-tailed noise.

It should be noted that the above approaches model point anomalies. Most of the literature on anomalous trajectory detection is based on the distance, direction and density of trajectories [97]. There are, to the best of the author's knowledge, no previous approaches in which a robust probabilistic method has been applied to anomaly detection (and robust estimation) in the context of object tracking.

### 2.2.4 Multi-category Object Tracking

The objects that are likely to be encountered by a robot are, in most scenarios, quite diverse. For example, a robot navigating in a city may find cars, pedestrians or cyclists. Assuming that all of them can be tracked using the same model inevitably leads to degradation in the object-classification- and trajectory-estimation performance. Reasoning about object classes enables the robot to use context information

more selectively for tracking. However, multi-object tracking and classification is a chicken-and-egg problem. On one hand, knowledge of object classes can improve tracking. On the other hand, good trajectories serve as features for better classification.

In general, multi-object tracking algorithms estimate object states without reasoning about classes. When this reasoning is required, complete tracks obtained from an independent tracking system are fed into a classifier like those explained in Section 2.2.3. Only few approaches exist that simultaneously perform tracking and classification [103, 104, 32, 105].

In order to classify object trajectories, many approaches to object categorisation (motion pattern discovery) overlook low-level problems such as data association and trajectory estimation. Traditional methods obtain complete object trajectories from a tracking module and then feed them into a classifier which obtains the semantic descriptors. The authors of [103] present a joint optimisation method that uses information from different sensor modalities in order to jointly estimate objects' states and classes. It allows the user to define the degree of correlation between tracking and classification by means of cost weights for errors in both tasks. However, it assumes known data association, which limits its application to environments where objects appearances are very dissimilar or they have identity markers. The work in [104] introduces the theoretical framework on which [103] is based. It argues the framework can be applied to solve the data-association problem when there is no interest in reasoning about classes. However no results are provided and, in addition, it is not clear how the method can be extended to jointly deal with state estimation, data association and classification. The work in [106] presented a Gaussian mixture implementation of the PHD filter that allows objects to switch between multiple motion models. Recently, [107] extended the work in [106] so that tracked objects can also be classified using features of both, the measurements and the tracked objects.

Simultaneous tracking and classification of multiple moving objects with unknown data association is a computationally intractable problem [108]. This is the reason why techniques that classify objects according to their dynamics perform each of

the tasks separately, decoupling state estimation from class assignment [92, 109, 94]. Hence, they neglect the natural correlations between an object's dynamics, environment, and class category. Additionally, most state-of-the-art approaches to MOT either do not assign category labels to tracked objects, or obtain them from an independent process usually based on images [110].

In this thesis we present a framework for the joint estimation of object classes, states, and data association. This has several advantages over previous approaches to both object classification and MOT. First of all, it can boost state-of-the-art appearance-based object classification methods [111, 112] by exploiting motion information and temporal correlations in the data. Secondly, since our approach formulates the problem in terms of a fully probabilistic model, it enables parameter learning. This is in contrast with most MOT approaches, in which the user is expected to empirically set the parameters of the tracker.

### 2.2.5 Vision-based Object Detection in Urban Environments

Object detection constitutes the feature-extraction module of any tracking-by-detection technique and it has been a relevant subject of research in areas like surveillance, ADAS and autonomous driving since the late seventies [1]. Being able to detect dynamic objects such as vehicles, cyclists, or pedestrians, and to estimate their positions and dynamics allows systems to increase their situational awareness. Object-detection schemes detect objects according to three main features: appearance, geometry and motion.

#### Appearance-based Object Detection

This approach relies on template models learned from training data [111, 110]. Training instances are image patches of the objects expected to be in the environment [113]. Some algorithms detect arbitrary objects [114, 115], whereas others specialises on particular categories given by the training set [116–118].

### Geometry-based Object Detection

Most of the approaches based on laser technology [119, 120] fall into this category. Here, object detection is formulated as a clustering problem [121], hence individual objects correspond to a set of points close together in 3D space. For laser data, clustering is usually done by means of optimal assignments computed, for example, by the Hungarian algorithm [122].

### Motion-based Object Detection

The works presented in [123], [124] and [125] highlighted the importance of motion perception and provided the first techniques to calculate a measure of visual motion named *optical flow*. Since then, a large amount of research has been accomplished in order to provide efficient and accurate forms to calculate the motion of multiple moving objects, from visual cues [126–128]. Motion detection on video streams recorded from a static camera, has been successfully performed by learning a static model of the background and then comparing it with the streaming images [129]. However, solving the problem when the camera is moving, presents extra challenging issues. Since the camera motion induces intensity changes in the entire visual field, even static objects appear as moving objects. Therefore, a static model of the background cannot be estimated.

One way to segment independent moving objects from the background, when the camera is mounted on a moving platform, is to compensate for the platform's movement. In order to do that, an estimate of the scene depth must be calculated. In [130] the *Flow Vector Bound constraint* along with the epipolar constraint are used as cues for dense motion detection. For this type of approach, the uncertainty in depth of a point spans through the entire epipolar line or at least a section of it.

In [57], a stereo vision system is used to obtain a dense depth field from which a prediction of the optical flow is estimated. As in [130–132], [57] considers pixel motion as the main cue for object detection. Hence, any sort of moving object can be detected.

## 2.3 Summary

This chapter showed the reader the pathway across which the contributions and context of this thesis extends. It presented the theoretical and technical concepts that the reader should be familiar with in order to understand the following chapters. This includes the definition of simultaneous multi-object tracking and classification as a state estimation problem, and the concepts in probability theory and sequential graphical models from which the individual components of our solution are obtained. Along with the background information, the chapter also reviewed the state-of-the-art literature in multi-object tracking and highlighted the importance of including classification- and outlier detection capabilities into sequential tracking approaches.

# Chapter 3

# A Novel Model for Probabilistic Multi-Object Tracking and Classification

A standard system for dynamic scene analysis has the constituent modules shown in Figure 3.1. The object-detection module generates detections or Regions Of Interest (ROIs) in the field of view of the robot that might correspond to individual entities in the environment. The data-association module groups the detections obtained across time so that those generated by the same object belong to the same group. Subsequently, the trajectory-estimation module, which is usually implemented as a filter, estimates trajectories using the noisy detections. Finally, the trajectory-classification module takes the estimated trajectories, and detects and models motion patterns. The flow of information in the standard dynamic-scene-analysis system just described, is illustrated by the straight black arrows in Figure 3.1.

In most scenarios, due to the physical structure of the environment, context rules and the nature of the objects, there is a correlation between *what* the objects in the environment are, and *how* they are moving. For example, pedestrians and cyclists might look alike, however their motion is fundamentally different. The motion of pedestrians is characterised by random changes in direction, whereas changes in the

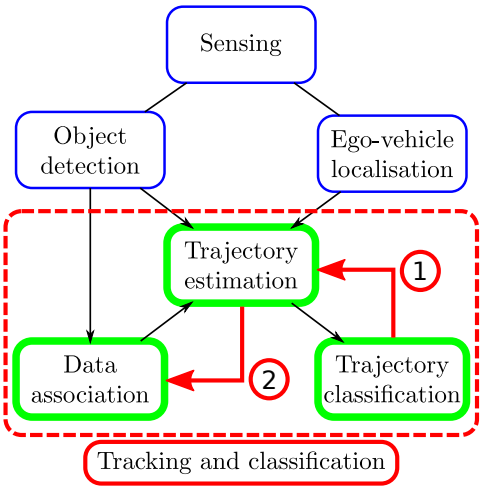



**Figure 3.1** – The general pipeline of a system for dynamic scene analysis. Boxes represent modules of the system; straight black lines represent the standard flow of information, whereas rounded red lines show the new correlations that the model introduced in this thesis represents.

direction of cyclists tend to be smoother. This thesis proposes a mathematical model that represents this correlation succinctly and efficiently (red arrow number one in Figure 3.1). It feeds back trajectory classification to trajectory estimation, and provides an graphical model representation that accounts for the multi-class nature of the object observations.

Another desirable feature of a dynamic-scene-analysis system is the capability of associating observations to estimated trajectories. This association between an observation and an estimated trajectory should be amenable to update, not only at the time the observation was acquired, but also at future times according to the ongoing history of the estimated trajectories. Feeding back information from the trajectory estimation to the data-association module makes it possible to recover object identities after merging or occlusion interactions.

In this chapter, a novel model for the Simultaneous Tracking and Classification (STC) problem is proposed. Section 3.1 and Section 3.2 introduce the modelling considerations used to represent object trajectories and their correlations with motion patterns in the environment, along with data-association ambiguities. Section 3.2.3 presents

the chapter with the equations that implement the model and therefore allows us to calculate the likelihood of the data.

## 3.1 Object State Representation

The main objective of this chapter is to introduce our proposed model for STC. The motion of an object is given by a time-ordered set of states, or trajectory, that is only observable through a sequence of noisy measurements provided by a sensor. We model a trajectory as a hidden Markov process that generates this sequence of sensor measurements. This model is also called a Linear Dynamical System (LDS) [15], and more specifically, it represents a sequence of observations as a linear projection of an underlying Markov process plus noise. An LDS can be represented using the graph in Figure 3.2. The nodes represent the variables of interest and the arrows between them represent Conditional Probability Distributions (CPD) that model the local relationships between variables.

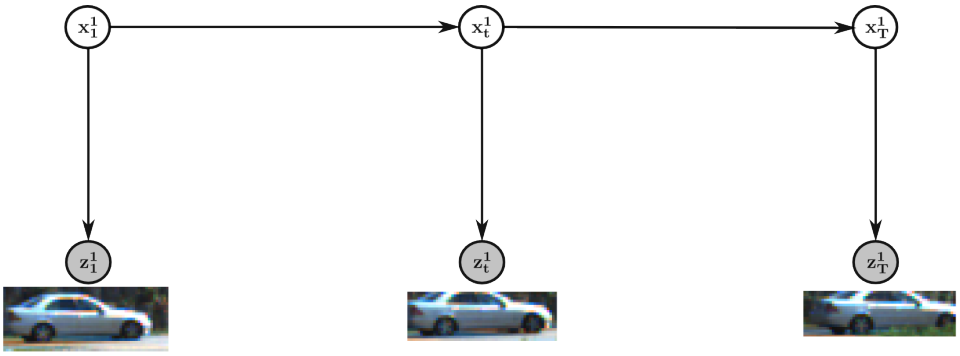



**Figure 3.2** – We can represent the trajectory of an object using a Markov chain of hidden variables with each observation conditioned on the state of the corresponding hidden variable. White and shaded nodes represent hidden and observed variables respectively. Horizontal arrows represent the transition probabilities $p(x_t|x_{t-1})$, whereas the vertical ones represent the likelihood $p(z_t|x_t)$ of each observation. If both the hidden and observed variables are continuous, this graph depicts the PGM of the LDS model

Under the LDS representation, the state sequence of an object in the scene corresponds to a *Directed Acyclic Graph* (DAG). Inference on this sort of model can be

done efficiently and exactly using the sum-product algorithm, which for the LDS model in particular, boils down to a set of forward and backward recursions known as the Kalman filter and smoother without driving inputs [133]. In the context of our application, observations correspond to noisy measurements of object positions (and optionally, shape), whereas hidden states are the actual positions and velocities (and optionally, shape descriptors).

In most environments, states evolve according to underlying class-dependent dynamics. In the urban scenario for example, cars, cyclists and pedestrians follow motion patterns that are different in terms of the velocity and smoothness of their trajectories. To account for different motion models, we have extended the model shown in Figure 3.2 using a Mixture of Linear Dynamical Systems (MLDS) model [38]. This model follows the same intuition behind the Gaussian Mixture Model (GMM) which is designed to account for unobserved groups in point data. The MLDS accommodates unobserved groupings or co-occurring behaviours by augmenting the LDS with a discrete hidden variable, as shown in Figure 3.3. This categorical random variable has a number of values $N_s$ equal to the number of expected classes.

The MLDS can model each observed trajectory and associate it to one of its mixture components. However, it is always possible that a trajectory following an unexpected pattern emerges. An MLDS would associate this trajectory to the closest component and therefore it cannot deal with outliers by itself. To make the approach robust to outliers, our model introduces the use of the t-student distribution, instead of the common Gaussian assumption. Since the t distribution has heavier tails than the Gaussian, it is able to account for events away from the mean. The authors of [134, 135] presented extensions of the HMM and LDS models respectively, in which they modelled the emission/observation conditional PDFs as t-distributions. The resulting models and their respective inference procedures resulted in state estimation approaches that were robust to outliers in the observations.

Our STC model represents each trajectory as a sequence of t-distributed random vectors. Each state in a trajectory is parametrised as a Gaussian random vector whose covariance matrix is weighted by a Gamma-distributed precision weight. The

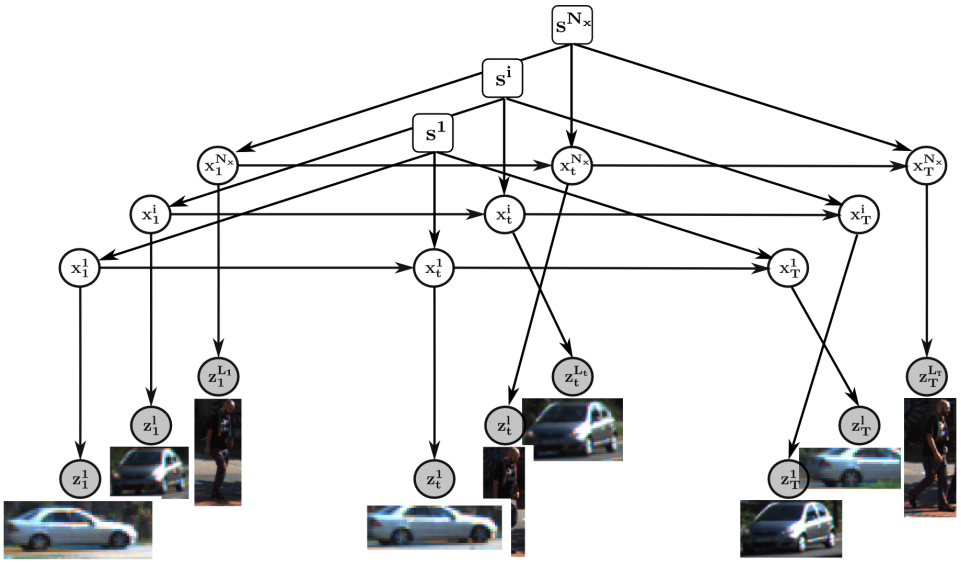


**Figure 3.3** – If both $x_t^i$ and $z_t^l$ are continuous, this graph corresponds to the PGM of an MLDS. In an MLDS a categorical variable $s$ (placed in a squared node) accounts for different dynamics by selecting one LDS per object. In other words, $s^i$ is a categorical random variable that assigns one of $N_s$ LDSs to object $i$. Each LDS models a different object class.

posterior over this weight allows us to detect outlier trajectories, while allowing the state to still be conditionally Gaussian, so inference continues to be tractable.

One of the most difficult problems in multi-object tracking is dealing with data-association ambiguities. For example, when observations from multiple objects get close to each other, knowing which observations belong to which objects becomes uncertain. We model assignments between observations and the state trajectory of tracked objects at each time step using categorical random variables that represent these assignments by means of soft associations. The following section presents a detailed description of the equations that constitute our model.

## 3.2 The Model

Let $x_t^i$ and $z_t^i$ be the hidden state and observation of object $i$ at time $t$, respectively. The hidden process $x_{1:T}^i$ represents the state trajectory of object $i$, whereas $z_{1:T}^i$ is the

sequence of observations or measurements of the object state (See Figure 3.2). The CPDs of our LDS correspond to multivariate Gaussian distributions:

$$\begin{aligned} p\left(x_t^i|x_{t-1}^i\right) &= \mathcal{N}\left(x_t^i; Fx_{t-1}^i, Q\right) \\ p\left(z_t^i|x_t^i\right) &= \mathcal{N}\left(z_t^i; Hx_t^i, R\right) \end{aligned} \tag{3.1}$$

where $F$ is the state-transition matrix; $Q$ is the process-noise covariance matrix; $H$ is the observation matrix or linear mapping between hidden states and observations; and $R$ is the covariance matrix that describes the noise in the sensor. The first and second lines in Eq. (3.1) are commonly referred to as the transition and observation models respectively. These models encode the characteristics of motion behaviours and the noise in the sensor. To convert an LDS into an MLDS we have added the categorical variable $s^i$ which assigns object $i$ to mixture component $j$. When this is done, the transition- and observation models for each mixture component become:

$$p\left(x_t^i \mid x_{t-1}^i, s^i = j\right) = \mathcal{N}(F_j x_{t-1}^i, Q_j) \tag{3.2a}$$

$$p(z_t^i \mid x_t^i, s^i = j) = \mathcal{N}(H_j x_t^j, R_j) \tag{3.2b}$$

where $s^i = j$ is the object category that generates trajectory $i$; and $F_j$, $Q_j$, $H_j$ and $R_j$ are the parameters of the $j$th LDS.

### 3.2.1 Unexpected Trajectory Detection

In order to account for trajectories that have unexpected dynamic behaviours, object states are modelled using a $t$ distribution rather than a Gaussian. The Gamma-distributed random variable $w^{i,j}$ is used to weight the covariance matrices related to the object states in the original MLDS model so that the states are marginally t-student but conditionally Gaussian. This Gamma variable works as a precision weight, thus it decreases with the degree at which the trajectory of object $i$ is inconsistent with model $j$. The closer $w^{i,j}$ is to zero, the more likely it is that $x_{1:T_i}^i$ is an outlier with respect to model $j$. Following the inclusion of the precision weight, the

transition- and observation models for class $j$ become:

$$p\left(x_i^t \mid x_{t-1}^i, s^i = j, \omega^{i,j}\right) = \mathcal{N}(F_j x_{t-1}^i, Q_j/\omega^{i,j}) \tag{3.3}$$

$$p\left(z_t^i \mid x_t^i, s^i = j, \omega^{i,j}\right) = \mathcal{N}(H_j x_t^j, R_j/\omega^{i,j}). \tag{3.4}$$

An LDS parametrised by $F_j$, $Q_j/\omega_i$, $H_j$, $R_j/\omega_i$ has a density function that represents a time-ordered sequence of $t$ distributions. In order to highlight the advantages of the $t$ distribution in the context of trajectory estimation and classification, Example 3.1 illustrates the evolution of posterior precision weights, obtained when learning an MLDS extended as explained in this subsection. The example shows weights corresponding to both inlier- and outlier trajectories.

**Example 3.1**

*Let us consider a modified two-mixture-component MLDS where each component is a Singer model (Equation 2.3) with different target-manoeuvre time constants. We draw a trajectory dataset from this model, and add three outlier trajectories from a noisy oscillator as shown in Figure 3.4.*

*Figure 3.5 depicts the posterior precision weights $\omega^{i,j}$ of both the inlier and outlier trajectories after each training iteration. Note that the outliers' weights have values close to zero, whereas the inliers' ones have larger values. This indicates that, in order to account for the trajectories' deviation from the dynamic behaviours modelled by the underlying MLDS, the covariance matrices of the model need to be inflated.*

### 3.2.2 Data Association

The MLDS allows us to represent both the dynamic ($x_t^i$) and categorical ($s^i$) states of object $i$, given the observation $z_t^i$. In practice, a sensor provides a set of measurements $z_t$ with no assignments to tracked objects. The problem of assigning observations to objects is known in the literature as *data association*.

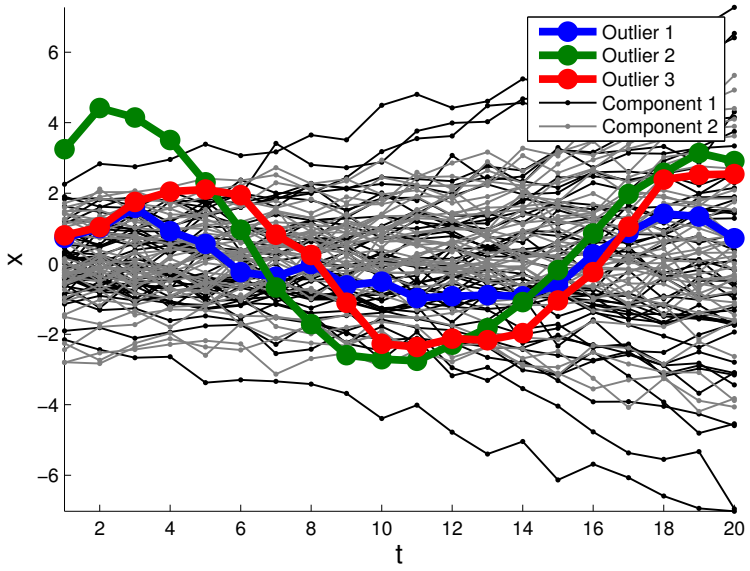
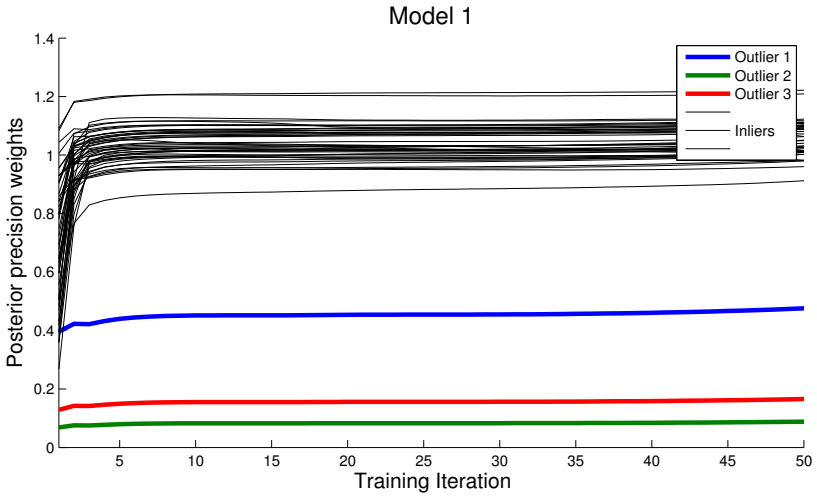



**Figure 3.4** – Set of trajectories drawn from a two-component STC model and three outlier trajectories generated by a noisy oscillator. The mixture components of the STC model are Singer models with varying manoeuvre-time constant parameter.

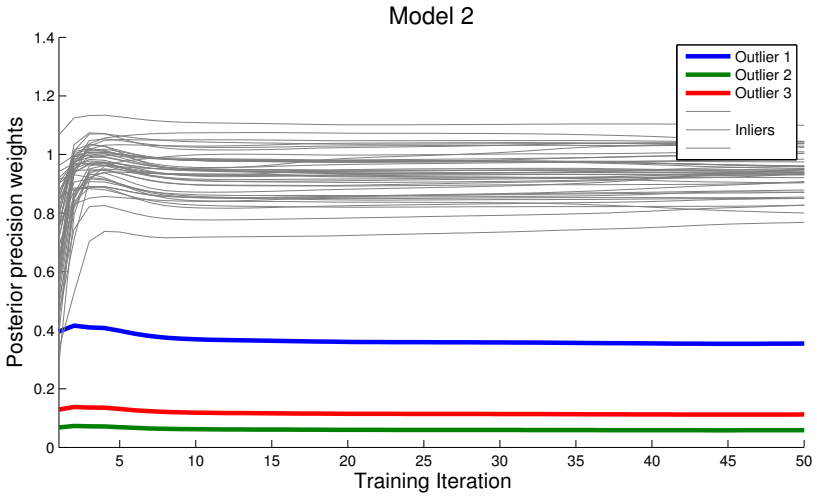
Consider a sequence of observations $z_{1:T} = (z_1, \dots z_t, \dots, z_T)$ with $z_t = (z_t^1 ... z_t^l ... z_t^{L_t})$. These observations are assumed to be generated by $N_x$ different objects. In order to represent the mapping between objects and observations, we define a sequence $a = (a_1, \dots, a_t, \dots, a_T)$ of assignment variables, with $a_t = \{a_t^1, \dots, a_t^l, \dots, a_t^{L_t}\}$. $a_t^l \in \{1, \dots, N_x\}$ is a categorical variable that specifies which object is responsible for generating observation $z_t^l$.

### 3.2.3 Model Overview

Putting together the modules presented above allows us to build a novel model for the multi-object Simultaneous Tracking- and Classification (STC) problem. Figure 3.6 shows the Bayesian-network representation of our generative model. The joint pro-




**(a)** Posterior precision weights for mixture component 1




**(b)** Posterior precision weights for mixture component 2

**Figure 3.5** – Per-trajectory posterior precision weights for each of the mixture components in our modified two-component MLDS model *vs* training iterations.

bability distribution can be written as follows:

$$p\left(s,x,z,a,\omega|\mathbf{\Omega}\right)=\prod_{i=1}^{N_x}\left[p\left(s^i\right)p\left(\omega^i|s^i\right)p\left(x_0^i\middle|s^i\right)\prod_{t=1}^{T_i}p\left(x_t^i\middle|x_{t-1}^i,s^i,\omega^{i,j}\right)\right] \\ \prod_{t=1}^{T}\prod_{l=1}^{L_t}p\left(z_t^l\middle|x_t^{1:N_x},a_t^l,\omega^{1:N_x}\right)p\left(a_t^l\right), \tag{3.5}$$

where the model parameters are given by $\mathbf{\Omega}=[F_{1:N_s},Q_{1:N_s},H_{1:N_s},R_{1:N_s}]$.

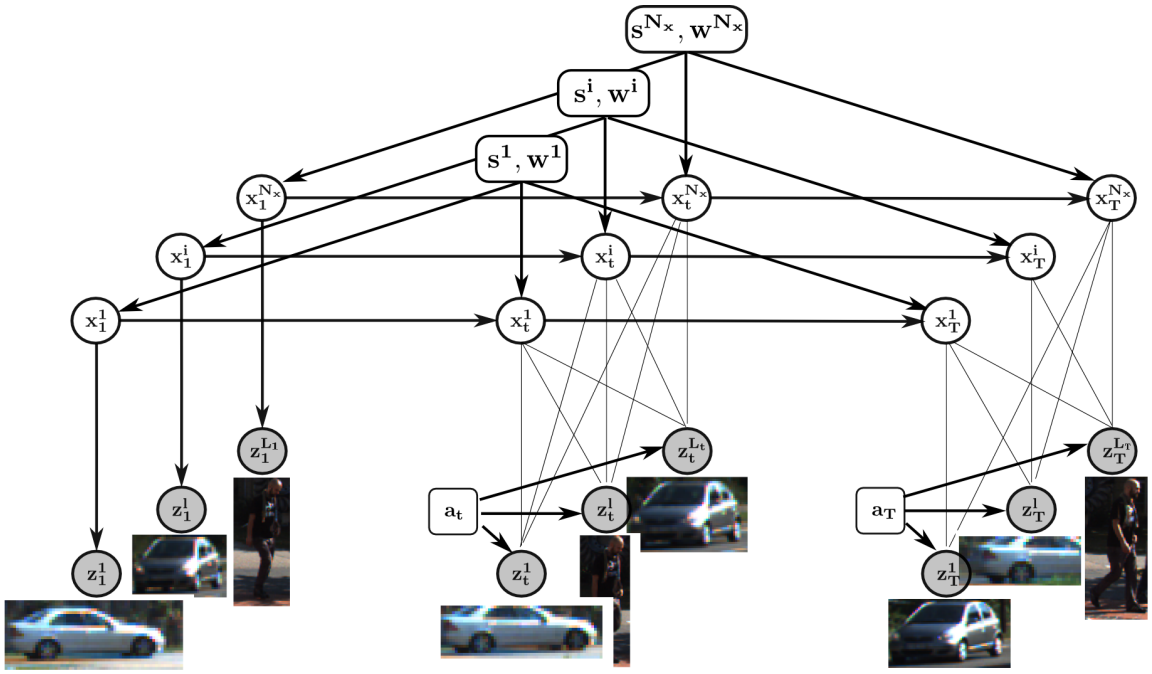



**Figure 3.6** – A graphical model of the simultaneous tracking- and classification problem with unknown data association. Unfilled nodes indicate hidden variables, while filled nodes are observed. $z_t$ represents observations at time $t$; $x_t^i$ models the state of object $i$ at time $t$; $a_t$ defines which observation is assigned to object $i$; $s^i$ chooses the class of object $i$; and $w^i$ is a precision weight that allows to define whether the trajectory $x_{1:T}^i$ follows an expected or unexpected behaviour

In a Bayesian network, random variables and their conditional dependencies are represented by means of a DAG. In our graph, each node $s^i$ is a categorical random variable used for indexing 1 of $N_s$ models. $x_t^i$ is a continuous random variable that models the state of object $i$ at time $t$. $w^{i,j}$ is a precision weight that allows us to define whether the object $i$ is an outlier or not. $z_t^l$ is the $l$th observation made at time $t$. $a_t^l$ is a categorical variable modelling the association between observation $l$ and tracked objects. Finally, $N_x$ is the number of objects in the scene, whereas $L_t$ is the number of observations made at time $t$. Table 3.1 summarises the terminology used in the model and throughout the rest of the thesis.

The factors that compose the complete likelihood of the data under the STC model

**Table 3.1** – Terminology used in our model

| Index | Symbol | Range |
|---|---|---|
| Object | $i$ | $1, \ldots, N_x$ |
| Model | $j$ | $1, \ldots, N_s$ |
| Time step | $t$ | $1, \ldots, T$ |
| Observation | $l$ | $1, \ldots, L_t$ |
| **Variable** | **Symbol** | **Support** |
| Class of object $i$ | $s^i$ | $\{1, \ldots, Ns\}$ |
| State of object $i$ at time step $t$ | $x_t^i$ | $\mathbb{R}^m$ |
| Sequence of object i's states from time $t$ to time $T_i$ | $x_i$ | $\mathbb{R}^m$ |
| Observation $l$ at time step $t$ | $z_t^l$ | $\mathbb{R}^n$ |
| Object i's surrogate observation at time $t$ | $\mathbf{z_t^i}$ | $\mathbb{R}^n$ |
| Sequence of object i's surrogate observations | $\mathbf{z_i}$ | $\mathbb{R}^n$ |
| Association of observation $l$ at time step $t$ | $a_t^l$ | $\{1, \ldots, N_x\}$ |

have the following parametrisations:

$$
\begin{aligned}
p\left(s^i\right) &= \prod_{j=1}^{N_s} p_j^{\delta\left(s^i, j\right)}, \quad p_j = \left[p\left(1\right), ..., p\left(N_s\right)\right], \\
p\left(a_t^l\right) &= \prod_{i=1}^{N_x} p_i^{l\,\delta\left(a_t^l, i\right)}, \quad p_i^l = \left[p\left(a_t^{l,1}\right), ..., p\left(a_t^{l,N_x}\right)\right], \\
p\left(\omega^i | s^i\right) &= \mathcal{G}\left(\omega^{i,j}; \alpha, \beta\right) \\
p\left(x_0^i \middle| s^i = j\right) &= \mathcal{N}\left(x_0^{i,j}; \mu_j, V_j\right) \\
p\left(x_t^i \middle| x_{t-1}^i, s^i = j, \omega^{i,j}\right) &= \mathcal{N}\left(x_t^{ij}; F_j x_{t-1}^{i,j}, Q_j / \omega^{i,j}\right) \\
p\left(z_t^l \middle| x_t^{1:N_x}, a_t^l = i, \omega^{i,j}\right) &= \mathcal{N}\left(z_t^l; H x_t^{i,j}, R_j / \omega^{i,j}\right).
\end{aligned}
$$

By substituting the factors in Equation 3.5 for their respective distributions, we can

rewrite the complete likelihood function as:

$$p\left(s,x,z,a,\omega|\mathbf{\Omega}\right) = \prod_{i=1}^{N_x}\Bigg[\prod_{j=1}^{N_s}\Big[p_j\mathcal{G}\left(\omega^{i,j};\alpha,\beta\right)\mathcal{N}\left(x_0^{i,j};\mu_j,V_j\right)\prod_{t=1}^{T_i}\mathcal{N}\left(x_t^{i,j};F_jx_{t-1}^{i,j},Q_j/\omega^{i,j}\right)$$
$$\prod_{t=1}^{T}\prod_{l=1}^{L_t}\left[p_i^l\mathcal{N}\left(z_t^l;Hx_t^{i,j},R_j/\omega^i\right)\right]^{\delta\left(a_t^l,i\right)}\Big]^{\delta\left(s^i,j\right)}\Bigg]. \tag{3.6}$$

Note that our joint distribution is a mixture model where each component is marginally $t$ and conditionally Gaussian. The model is therefore robust to account for objects whose behaviour deviates from the modelled ones. In order to define the cost function to be optimised by the inference module, we substitute the factor PDFs and apply the logarithm:

$$\begin{aligned}\ln p\left(s,x,z,a,w|\mathbf{\Omega}\right) = &\sum_{i=1}^{N_x}\Bigg[\sum_{j}^{N_s}\delta\left(s^i,j\right)\left(\frac{\upsilon}{2}\ln\frac{\upsilon}{2}+\left(\frac{\upsilon}{2}-1\right)\ln\omega_i-\frac{\upsilon}{2}\omega_i-\ln\Gamma\left(\frac{\upsilon}{2}\right)+\ln p_j+\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\ln p_i^l\right)\\
&+\sum_{j}^{N_s}\delta\left(s^i,j\right)\left(-\frac{1}{2}\left(x_0^{i,j}-\mu_j\right)^T V_j^{-1}\left(x_0^{i,j}-\mu_j\right)-\frac{1}{2}\ln|V_j|\right)\\
&+\sum_{j}^{N_s}\delta\left(s^i,j\right)\left(-\sum_{t=2}^{T_i}\left(\frac{\omega_i}{2}\left(x_t^{ij}-F_jx_{t-1}^{i,j}\right)^T Q_j^{-1}\left(x_t^{ij}-F_jx_{t-1}^{i,j}\right)\right)-\frac{T_i-1}{2}\ln|Q_j|+\frac{m\left(T_i-1\right)}{2}\ln\omega_i\right)\\
&+\sum_{j}^{N_s}\delta\left(s^i,j\right)\left(\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\left(-\frac{\omega_i}{2}\left(z_t^l-H_jx_t^{i,j}\right)^T R_j^{-1}\left(z_t^l-H_jx_t^{i,j}\right)-\frac{1}{2}\ln|R_j|+\frac{n}{2}\ln\omega_i\right)\right)\\
&-\sum_{j}^{N_s}\delta\left(s^i,j\right)\frac{T_i\left(m\right)}{2}\ln\left(2\pi\right)-\sum_{j}^{N_s}\delta\left(s^i,j\right)\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\frac{n}{2}\ln\left(2\pi\right)\Bigg].\end{aligned} \tag{3.7}$$

The complete log-likelihood in Equation 3.7 is given by the sum of the log-likelihoods of $N_x$ trajectories under a STC model parametrised by $\Omega$. The first line is a combination of terms that are functions of the precision weight $\omega_i$ and the prior class $p_j$ and data association $p_i^l$ probabilities. The terms in the second line represent the initial states. The third line represents how well object $i$ complies with the dynamic behaviour represented by the $j^{\text{th}}$ mixture component. The fourth line represents how well object $i$ complies with the observation function and noise modelled by the mixture component $j$. Finally, the fifth line is a sum of normalising terms. Note that the

functional form of our model is very similar to that of an LDS, therefore it inherits its analytical and computational advantages. However, in addition to the properties of the LDS, our new model accounts for (i) multiple dynamics, (ii) outlier detection and (iii) data-association modelling.

## 3.3 Summary

This chapter introduced a probabilistic graphical model that encodes all of the variables in the multi-class tracking scenario. Our complete model can be seen as a Mixture of Linear Dynamical Systems (MLDS) extended in several ways. Firstly, the state of each object is modelled using a $t$ distribution. We introduced a Gamma-distributed auxiliary random variable for weighting the covariance matrices of the MLDS model so that the overall state is conditionally Gaussian. This allows the model, as it will be shown in Chapter 4, to account for anomalous trajectories, while still permitting the use of the efficient inference routines that have been already developed for Gaussian models. Secondly, data-association ambiguities are accounted for by replacing the direct relation between object states and observations with a categorical random variable that represents soft assignments between the observations and all of the existing objects at time $t$.

At the end of the chapter, we presented the complete log-likelihood of the data under the model introduced throughout the chapter. This likelihood will work as the cost function used to estimate both the variables of interest – objects' states and classes, data association, and anomaly score – in Chapter 4 and the parameters of the model in Chapter 5.

# Chapter 4

# The Expectation Association Algorithm

The previous chapter introduced our Probabilistic Graphical Model (PGM) for multi-object tracking and classification. As a result of the parameters in the likelihood function being tightly coupled, performing exact inference on this model is computationally intractable. We must therefore resort to approximate inference methods. The two most common solutions to this problem are Markov Chain Monte Carlo (MCMC) and variational inference. Motivated by the efficiency and convergence properties of variational inference methods, we introduce a new deterministic approximation scheme for estimating object trajectories and classes, while also solving the intractability issue of data association.

This chapter introduces the Expectation Association (EA) algorithm, a variational approach for the multi-object Simultaneous Tracking- and Classification (STC) problem. It starts by motivating the use of variational approximations for the problem at hand. Then it presents a factorised approximation of the posterior of interest and the update equations needed to calculate each of the factors that constitute this approximation. Subsequently, some considerations about the sequential implementation of the inference procedure are highlighted. The chapter concludes with a qualitative comparison between EA and some state-of-the-art tracking methods.

## 4.1 The Lower Bound

Performing inference on our model, introduced in Chapter 3, is equivalent to estimating the posterior distribution $p(s, w, x, a|z)$ over classes, precision weights, object states and associations. In principle, this posterior can be calculated by maximising the likelihood of the data. The log-likelihood of the data, in turn, is obtained by marginalising out the set of hidden variables $(s, \omega, x, a)$ given the STC model parametrised by the parameters $\Theta$ and the observations $z$:

$$\ln p(z) = \ln \sum_{s,a} \iint p\left(s, \omega, x, a, z\right) d\omega dx. \tag{4.1}$$

Unfortunately, this integration is both analytically and computationally intractable due to the coupling between variables induced by the summations. By applying the *d-separation criterion* [15] on our model in Figure 3.6, it can be seen that, although object states and associations are marginally independent, conditioning on the observations (explaining away evidence) introduces statistical dependencies between them. As a result of these dependencies, the posterior is a mixture distribution where the number of components increases combinatorially with the number of objects and exponentially with time.

Given that the exact likelihood function is intractable, an approximation is needed, and for that we use a lower bound instead. Let $q(s, \omega, x, a)$ be a probability density function that approximates the exact posterior $p(s, \omega, x, a|z)$. By expressing $\ln p(z)$ as:

$$\ln \sum_{s,a} \int q\left(s, \omega, x, a\right) \frac{p\left(s, \omega, x, a, z\right)}{q\left(s, \omega, x, a\right)} d\omega dx, \tag{4.2}$$

then applying Jensen's inequality [15] and realising that the logarithm is a convex function, we arrive at a lower bound:

$$\ln p(z) \geq \sum_{s,a} \int q\left(s, \omega, x, a\right) \ln \frac{p\left(s, \omega, x, a, z\right)}{q\left(s, \omega, x, a\right)} d\omega dx = \mathcal{L}\left[q\right]. \tag{4.3}$$

This inequality holds for any choice of $q$. In particular, if $q(s, \omega, x, a)$ equals the true

posterior $p(s, \omega, x, a|z)$, then Equation 4.3 becomes an equality.

### 4.1.1 The Factorised Approximation

If tracked objects are well separated or they have identity markers, the associations are easily obtained and the posterior over object states can be efficiently calculated. Otherwise, the method will have to account for the ambiguities in the data association process. For this case, estimating the object state $x^i$ requires calculating the following marginal:

$$p\left(x^i|z\right) = \sum_a p\left(x^i|a, z\right) p\left(a\right). \tag{4.4}$$

Calculating this marginal requires computations that grow combinatorially with the number of objects and exponentially with time. In order to overcome this computational intractability, approximations are usually made.

The inequality in Equation 4.3 holds for any choice of $q$. In particular, if $q(s, x, a)$ equals the true posterior $p(s, x, a|z)$, then Equation 4.3 becomes an equality. We propose approximating our posterior with a probability density function $q$ that separates classes and states from data associations, so it factorises as follows:

$$q\left(s, \omega, x, a\right) = q(s, \omega, x)q(a). \tag{4.5}$$

Given this factorisation, the posterior of interest is approximated as the product of a state/class distribution and an association distribution. The approximate state/class distribution is a mixture distribution whose complexity increases linearly with time and the number of objects, and not exponentially as is the case for the true posterior. Similarly, the calculation of the distribution over associations also becomes linear in time and in the number of tracked objects

Our approximation assumes that, given the data, state sequences and associations

are statistically independent[1]. This does not imply that the state estimates and data associations are decoupled; in fact, they depend on one another via algebraic equalities — see Equation 4.10 to Equation 4.20. The variational approximation transforms these *statistical* dependencies into *algebraic* constraints. We resolve these constraints by optimising each of the $q$ factors in turn, i.e. by fixing the state estimates and updating the associations, and vice-versa. As shown in Figure 4.1, in the context of our tracking application, object trajectories tend to be temporally coherent, hence, given the data observed up to the current time step, it is possible to recursively estimate states, classes and data association. Example 4.1 illustrates the motivation behind the main assumption in our approximate model.

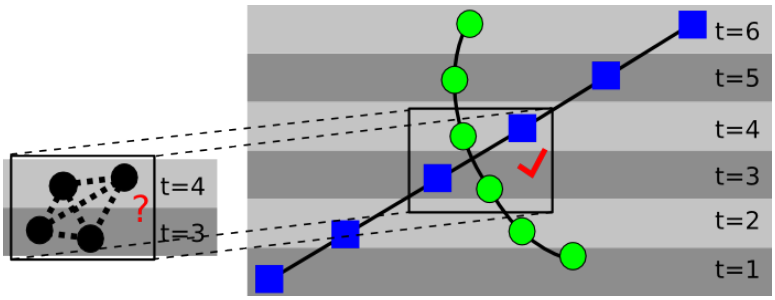



**Figure 4.1** – An intuitive explanation of our variational approximation. We assume that once object trajectories are observed, temporal coherence in the observations make the correlation between classes/states and associations negligible, hence we can assume they are statistically independent. On the left hand side of the image, it would be difficult to choose which observations from time $t = 3$ go with which observations at time $t = 4$. In contrast, as shown at the right hand side, by looking at the observations from time $t = 1$ to $t = 6$ this ambiguity is minimised.

**Example 4.1**

*Let us consider two trajectories sampled from a two-component STC model. In Figure 4.2, stars represent observations, whereas filled circles and squares represent the two different object classes. The true class assignment along with the observations are shown in the first image. The squares/circles that are linked represent one individual object. This example illustrates how the motion history of the objects serves to disambiguate the data association and classify them according to their dynamic behaviour.*

[1]In other words, after having observed the data, any remaining statistical dependencies (e.g. cross-covariances) between state and association variables are not captured by our approximation.

*In iteration 1, two objects are initialised and observations inside the dashed circle are associated to these objects with a high probability. The associations for detections outside the circle are assigned a non-informative uniform distribution. After running the E-step with $q(a)$ initialised as explained before, the resulting trajectories show confident assignments inside the circle and ambiguous ones elsewhere. In iteration 2, we run the E-step using the $q(a)$ updated in the previous iteration. Although the class assignment has not converged to the correct one (note that the colours in iterations 2 and 3 are different), object trajectories fit better the structure of the observations. Convergence is reached in the 3rd iteration where the class assignment is equivalent to the one depicted at the beginning of the image sequence.*

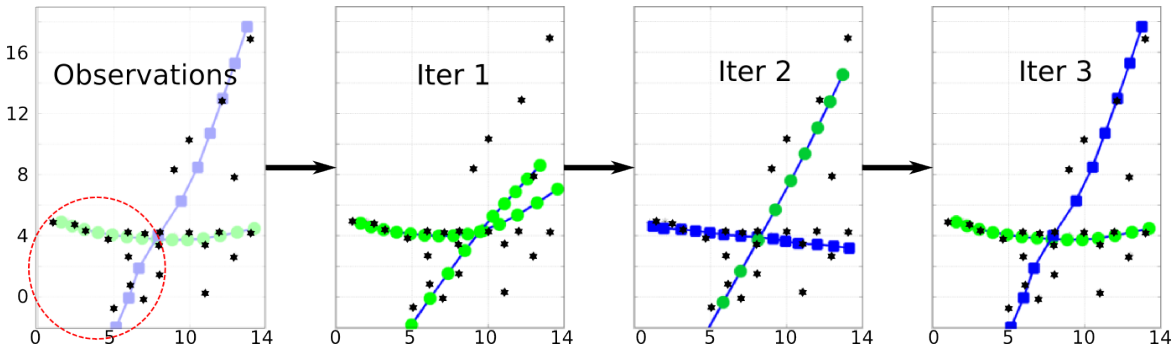



**Figure 4.2** – A synthetic example of state/class estimation from noisy observations using the EA algorithm. The first image depicts the ground-truth trajectories and the observations, including those for which the prior data association is confident (black stars inside the red dotted circle). The last three images show the state estimation (blue connected segments) and class assignment results (coloured markers) after three iterations of the EA algorithm.

---

The initial factorisation in Equation 4.5 results in other factorisations across objects, time and within observations. These are *induced* factorisations, i.e., they do not concede additional accuracy and are exact, given the initial assumption in Equation 4.5. The first set of induced factorisations:

$$q(s, \omega, x) = \prod_{i=1}^{N_x} q_i(s^i, \omega^i, x^i) \tag{4.6}$$

allows us to estimate the state sequence of each object independently from each other and thus we refer to them as the *state factors*. The second set of induced factorisations allows us to have one factor for each association possibility. Therefore, the probability of associating object $i$ to observation $l$ at time $t$ can be updated independently from each other. The approximate marginal over the data association can then be written as:

$$q(a) = \prod_{t=1}^{T} \prod_{l=1}^{N_z} \prod_{i=1}^{N_x} q_t^{l,i}\left(a_t^{l,i}\right). \tag{4.7}$$

We refer to each factor $q_t^{l,i}$ as an *association factor*. Unlike the exact posterior, our approximation in Equation 4.5 is computationally tractable. We can derive the expressions for the factors in Equation 4.6 and Equation 4.7 by maximising Equation 4.3. As explained in [15], the log of the optimal solution for factor $q_i$ is obtained by considering the log of the joint distribution over all variables and then taking the expectation with respect to all of the other factors $q_j$ for $j \neq i$. Our joint distribution is given by Equation 3.7.

Factors $q_i(s^i, \omega^i, x^i)$ and $q_t^{l,i}(a_t^{l,i})$ are iteratively updated as explained in Section 4.2 and Section 4.3 respectively. We call this iterative process the Expectation-Association (EA) algorithm and introduce its batch version with the pseudo-code in Algorithm 1. The next sections explain the two main procedures of our algorithm; the Expectation step (E-step) and the Association step (A-step).

**Algorithm 1** The batch EA algorithm

1: $Model \leftarrow$ Learn model parameters (Chapter 5)
2: $q(a) \leftarrow$ Initialise object-observation association probabilities using appearance (Example 4.2)
3: **procedure E-STEP**($Models$, $q(a)$)
4: $\overline{z}_t^i \leftarrow$ Calculate per-object average observation.
5: $\overline{R}_t^{i,j^{-1}} \leftarrow$ Calculate per-object/per-model observation noise covariance.
6: $\sum_{t=2}^{T_i} l_t^{i,j} \leftarrow$ Run per-object/per-model Kalman Filter and obtain innovation log-likelihoods.
7: $q(x^i|s^i,\omega^i) \leftarrow$ Run RTS smoother and obtain per-model posterior over object states.
8: $q(\omega^i|s^i) \leftarrow$ Calculate posterior over precision weights
9: $q(s^i) \leftarrow$ Calculate marginal over class assignments.
10: **end procedure**
11: **procedure A-STEP**($Models$,$q(s,x)$)
12: $q(a) \leftarrow$ Update the association probabilities.
13: **end procedure**
14: Repeat until convergence.

## 4.2 The Expectation Step (*E-step*)

Each state factor $q_i(s^i,\omega^i,x^i)$ is obtained as the function that maximises the lower bound in Equation 4.3. The optimal state factor $q_i$ is given by the expectation of the complete log-likelihood with respect to the factors $q_{k\neq i}$ and the association factors $q(a)$:

$$\ln q\left(s^i,x^i,\omega^i\right) = \left\langle \ln p\left(s,x,z,a,\omega\right)\right\rangle_{q_{k\neq i}(s^k,x^k,\omega^k),q(a)}. \tag{4.8}$$

Given the conditional independence properties of our model, we can further rewrite $q_i\left(s^i,x^i,\omega^i\right)$ as $q_i\left(s^i\right)q_i\left(\omega^i|s^i\right)q_i\left(x^i|s^i,\omega^i\right)$, so its logarithm is given by:

$$\ln q_i\left(s^i,x^i,\omega^i\right) = \ln q_i\left(s^i\right) + \ln q_i\left(\omega^i|s^i\right) + \ln q\left(x^i|s^i,\omega^i\right). \tag{4.9}$$

As shown in Appendix A, $q\left(x^i|s^i,\omega^i\right)$ has the form of an LDS, hence $\ln q_i\left(s^i,x^i,\omega^i\right)$ can be written as:

$$\ln q\left(s^i,x^i,\omega^i\right)=\ln q\left(s^i\right)+\ln q\left(\omega^i|s^i\right)+\ln LDS\left(x^i,\mathbf{z_i};\mathbf{F_i},\mathbf{Q_i}/\omega^i,\mathbf{H_i},\mathbf{R_i}/\omega^i\right)+\ldots \tag{4.10}$$

where "..." represent additive constants and the parameters of the LDS are:

$$\begin{aligned}\mathbf{F_i}|s^i&=F_j,\\ \frac{\mathbf{Q_i}}{\omega^i}|s^i&=\frac{Q_j}{\omega^{i,j}},\\ \mathbf{H_i}|s^i&=H_j,\\ \frac{\mathbf{R_i}}{\omega^i}|s^i&=\frac{1}{\alpha_t^i\omega^{i,j}}R_j,\end{aligned} \tag{4.11}$$

with

$$\begin{aligned}\alpha_t^i&=\sum_{l=1}^{L_t}\alpha_t^{l,i},\\ \alpha_t^{l,i}&=q(a_t^l=i).\end{aligned} \tag{4.12}$$

The term:

$$\mathbf{z_t^i}=\frac{\sum_{l=1}^{L_t}\alpha_t^{l,i}z_t^l}{\alpha_t} \tag{4.13}$$

is a surrogate observation per object $i$ given the current estimate of $q(a)$. More precisely, it is a weighted average of the observations with weights given by the posterior association probabilities of all of the observations and target $i$. Additionally, since $x^i$ is conditionally Gaussian, the factor $q\left(x^i|s^i,\omega^i\right)$ can be efficiently calculated using the Kalman filter (KF) and the Rauch-Tung-Striebel (RTS) smoother [70]. Note that the observation-noise covariance matrix to be fed to object's $i$ filter/smoother at time $t$ is not a constant matrix any more, but is a function of the association factors. Therefore, we can define the surrogate covariance matrix for object $i$ at time $t$ as follows:

$$\mathbf{R_t^{i,j}} = \frac{1}{\alpha_t^i \omega^{i,j}} R_j. \tag{4.14}$$

In order to obtain the posterior over $q(\omega^i|s^i)$ we marginalise out $x^i$ from the state factor. The approximate posterior over $\omega^i$, conditional on $s^i = j$, is a Gamma distribution with parameters:

$$\alpha_{i,j} = \frac{\upsilon}{2} + \frac{n}{2} \sum_{t=1}^{T_i} \alpha_t^i \tag{4.15}$$

$$\beta_{i,j} = \frac{\upsilon}{2} + \frac{1}{2} \sum_{t=1}^{T_i} \left( \sum_{l=1}^{L_t} \left( \alpha_t^{li} z_t^{lT} R_j^{-1} z_t^l \right) - \mathbf{z_t^i}^T \alpha_t^i R_j^{-1} \mathbf{z_t^i} + \epsilon_t^{ijT} \Sigma_t^{ij} \epsilon_t^{ij} \right) \tag{4.16}$$

where $\Sigma_t^{ij} = H_j \hat{V}_{t|t-1}^{ij} H_j^T + R_j$ and $\epsilon_t^{ij} = \mathbf{z_t^i} - H_j \hat{x}_{t|t-1}^{ij}$ are the innovation statistics given by the Kalman filter. The conditional posterior mean of the precision weight $\hat{\omega}^{i,j}$, is therefore given by

$$\hat{\omega}^{i,j} = \frac{\alpha_{ij}}{\beta_{ij}}. \tag{4.17}$$

Equations 4.15, 4.16 and 4.17 constitute the outlier rejection mechanism of our framework, previously illustrated in Example 3.1. If, as a result of mixture component $j$ being unable to explain sequence $i$, the innovation errors $\epsilon_t^{ij}$ are large, then $\beta_{ij}$ increases and causes $\omega_{ij}$ to drop, effectively down-weighting the entire sequence.

Finally, the posterior assignment probabilities are obtained by further marginalising $\omega^i$ out:

$$\begin{aligned} \ln q(s^i = j) &= \ln \int_0^\infty \exp\left( \ln q\left( s^i = j, \omega^i \right) \right) d\omega^i \\ &= a + \ln b + \ln \Gamma(b) - (b+1) \ln c \end{aligned} \tag{4.18}$$

where

$$a = \ln p_j + \frac{\upsilon}{2} \ln \frac{\upsilon}{2} - \ln \Gamma\left( \frac{\upsilon}{2} \right) - \frac{1}{2} \sum_{t=1}^{T_i} \ln |\Sigma_t^{ij}| + \sum_{t=1}^{T_i} \sum_{l=1}^{L_t} \alpha_t^l p_i^l + \eta_{ij};$$

$$b = \frac{\upsilon}{2} - 1 + \frac{n}{2}\sum_{t=1}^{T_i}\alpha_t;$$
$$c = \frac{\upsilon}{2} + \frac{1}{2}\sum_{t=1}^{T_i}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l^T}R_j^{-1}z_t^l\right) - \mathbf{z_t^i}^T\alpha_t R_j^{-1}\mathbf{z_t^i} + \epsilon_t^{ij^T}\Sigma_t^{ij}\epsilon_t^{ij}\right);$$

One can see that the optimal $q(s, w, x)$ is a Gaussian mixture distribution, with one component for each motion pattern. Note that the marginal over the class assignment variables in Equation 4.18 is obtained by updating the prior over class assignments with a set of terms that are a function of the marginal log-likelihood of the data under the model $j$. This log-likelihood can be obtained as a by-product of the E-step. It is equal to the sum of the innovation log-likelihoods $l_t^{i,j}$, which are computed at each update step.

Note as well, that accumulating these innovation log-likelihoods, after performing filtering with each of the models, allows us to infer the assignment of targets to motion patterns. Furthermore, since Kalman filtering provides these innovation log-likelihoods each time an observation is processed, evidence about class assignments can be sequentially updated. This is fundamental for applying our framework to online tracking.

## 4.3 The Association Step (*A-step*)

The second factor of the factorised approximation is $q(a)$. Its natural logarithm is given by:

$$\ln q(a) = \sum_{t=1}^{T}\sum_{l=1}^{N_z}\sum_{i=1}^{N_x}\ln q\left(a_t^{l,i}\right), \tag{4.19}$$

where $a_t^{l,i}$ is a categorical random variable that is 1, if detection $l$ was generated by object $i$ at time $t$, and 0 otherwise. We obtain each of the sub-factors in Equation 4.19 by maximising Equation 4.3 with respect to $q(a_t^{l,i})$. The solution for the association

sub-factors is given by:

$$\ln q\left(a_t^{l,i}\right) = \ln p\left(a_t^{l,i}\right) + \sum_{j=1}^{N_s} q\left(s^{i,j}\right) \Bigg( - \frac{\hat{\omega}^{i,j}}{2}\left(\left(z_t^l - H_j\hat{x}_t^{i,j}\right)^T R^{j^{-1}}\left(z_t^l - H_j\hat{x}_t^{i,j}\right) + Tr\left(H^{j^T} R^{j^{-1}} H_j Cov\left(\hat{x}_t^{i,j}\right)\right)\right) + \frac{n}{2}\left(\psi(\alpha_{ij}) - \ln(\beta_{ij})\right)\Bigg) + \ldots \tag{4.20}$$

where $\hat{x}_t^{i,j}$ and $\hat{\omega}^{i,j}$ are the smoothed trajectories and the posterior precision weights respectively. Note that $q(a)$ depends on the square of the error between expected and actual observations. Moreover the log-likelihood of assigning object $i$ to observation $l$ at time $t$ decreases when the uncertainty about the state of object $i$ (state covariance) increases. This permits our approach to be robust against spurious observations, even without explicit states to model them, as in classical approaches such as JPDAF, MHT or MCMCDA. Since spurious detections typically support a very small portion of the object's trajectory, they tend to have very weak estimated associations, even if these associations were initialised with a high probability. Object trajectories that were initialised due to spurious observations tend to remain short, as they are promptly removed due to their lack of evidential support. A complete derivation of the association factors is provided in Appendix B.

### 4.3.1 Integration of Multi-modal Features

A key feature of our formulation is its flexibility to integrate multi-modal features when performing data association. In most platforms there are several sources of information to estimate the association between objects and observations. Sensor modalities such as stereo cameras or lasers provide both depth and intensity measurements from which complementary features can be derived.

The prior over associations $p(a_t^l)$ (see Equation 4.20) could for instance be calculated based on appearance features; the inference algorithm would then compute the pos-

terior $q(a_t^l)$ by seamlessly fusing this prior with evidence from the object's dynamics. Example 4.2 illustrates two ways of initialising these association probabilities as found in [136, 27] and an improved approach developed in this thesis.

**Example 4.2**

*In the implementation of our work presented in [27], once an object is time updated, the image of its assigned observation is stored. Then the appearance-based prior over associations is obtained by calculating the normalised cross correlation between the image patches of new detections and those previously assigned to objects. Our proposed approach provides a framework for a more robust initialisation. Similar to [136], we can summarise the image information of individual objects, before and after association ambiguities, using colour histograms as appearance models. We can then sequentially update the appearance model of object i by averaging the histogram of the current assigned image patch (detection) and the current histogram/appearance model.* $p\left(a_t^{l,i}\right)$ *is obtained from the histogram-intersection between appearance models and the histograms of the image patches of current detections. Figure 4.3 illustrates the process.*

## 4.4 The Online EA Algorithm

The factorised approximation we have proposed allows our method to be implemented in a sequential manner. As shown in Equation 4.18, the assignment probabilities are a function of the innovation statistics of the object states under each of the model components. Therefore, when applying our method online, we simply filter each track using each of these model components and accumulate their innovation statistics so that the class-assignment probabilities can be recalculated at each time step. Similarly, the association factors can be sequentially updated due to the fact that they are a function of the current object state.

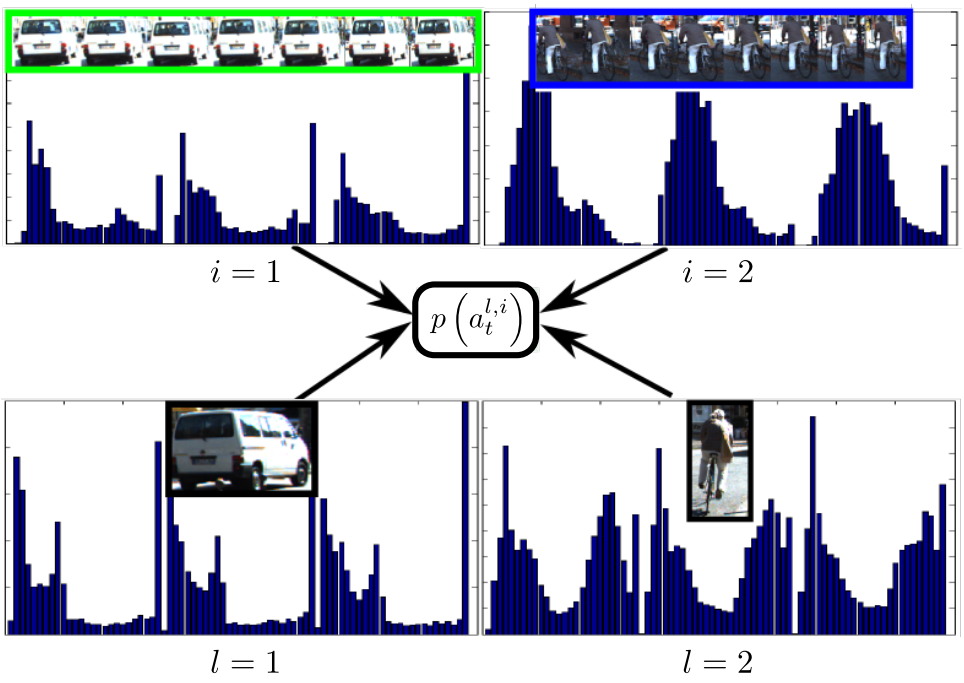



**Figure 4.3** – An appearance-based prior distribution over data associations can be obtained by comparing the colour histogram of tracked objects (indexed by $i$) and the ones for the current detections (indexed by $l$). As shown in the top two images, the histogram of each object $i$ summarises its sequence of associated images. The bottom two images represent the incoming detection images with their corresponding histograms.

In order to perform sequential inference, our implementation accumulates detections in batches that have a size predefined by the user. Once a detection is obtained, it is accumulated and the data that falls in the current sliding window is processed. The entire inference process is summarised in Algorithm 2.

The Kalman forward-backward recursions provide our method with the capability of solving data-association ambiguities without throwing away evidence in ambiguity areas. By forward propagating the filtering densities, followed by backward propagating the smoothed densities, we allow the dynamics of the objects to refine the state estimates and, more importantly, obtain the association between observations and objects as a by-product of the object-state histories.

The approximate state/class distribution for object $i$ has the form of an LDS, hence, as mentioned before, this posterior mode can be calculated using the RTS recursions.

**Algorithm 2** The online EA algorithm

Model ← Learned model
$w$ ← Sliding window provided by the user
EAits ← Number of EA iterations
**for** $t \leftarrow 1$, $T$ **do**
    $d_t$ ← Obtain object detections
    $z_t^{1:N_z}$ ← Obtain raw observations
    $q(a_{t-w+1:t})$ ← Initialise association probabilities in the sliding window

    **for** $k \leftarrow 1, \text{EAits}$ **do**

        **procedure E-STEP**(Model, $q(a_{t-w+1:t})$)
            **for** $i \leftarrow 1,\ N_x$ **do**
                $\mathbf{z^i_{t-w+1:t}}$ ← Calculate surrogate observations.
                $\mathbf{R^{i,s^i}_{t-w+1:t}}$ ← Calculate surrogate noise covariances.
                $q_i(x^i|s^i, \omega^i)$ ← Perform filtering.
                $\sum l^{i,s^i}_{1:t}$ ← Accumulate innovation log-likelihoods.
                $q_i(x^i|s^i, \omega^i)$ ← Perform smoothing.
                $q_i(\omega^i|s^i)$ ← Calculate posterior over precision weights.
                $q_i(s^i)$ ← Calculate posterior class assignment probabilities.
            **end for**
        **end procedure**

        **procedure A-STEP**(Model, $q(x, s, w)$)
            **for** $t_w \leftarrow t - w$, $t$ **do**
                **for** $l \leftarrow 1$, $L_{t_w}$ **do**
                    $q(a^l_{t_w})$ ← Update the association factors.
                **end for**
            **end for**
        **end procedure**

    **end for**

**end for**

The resulting update equations for the forward pass are as follows:

$$\begin{aligned} \overline{x}_t^{i,j} &= F_j \overline{x}_{t-1}^{i,j} + K_t \left( \overline{z}_t^i - H_j F_j \overline{x}_{t-1}^{i,j} \right), \\ \overline{V}_t^{i,j} &= \left( I - K_t H_j \right) P_{t-1} \left( I - K_t H_j \right)^T + K_t \frac{\mathbf{R_t^{i,j}}}{\omega^i} K_t^T, \end{aligned} \tag{4.21}$$

where we have defined:

$$\begin{aligned} P_{t-1} &= F_j \overline{V}_{t-1}^{i,j} {F_j}^T + \frac{Q_j}{\omega^i}, \\ K_t &= P_{t-1} {H_j}^T \left( H_j P_{t-1} {H_j}^T + \frac{\mathbf{R_t^{i,j}}}{\omega^i} \right)^{-1}. \end{aligned} \tag{4.22}$$

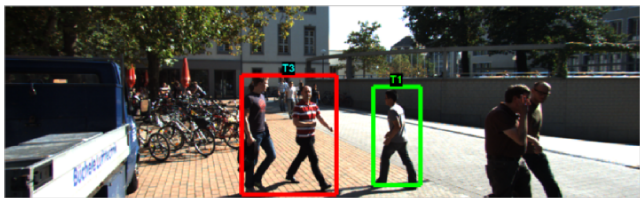
Once the filtering (forward pass) has been done, we calculate the smoothed posterior using the backward recursions:

$$\begin{aligned} \hat{x}_t^{i,j} &= \overline{x}_t^{i,j} + J_t \left( \hat{x}_{t+1}^{i,j} - F_j \overline{x}_t^{i,j} \right), \\ \hat{V}_t^{i,j} &= \overline{V}_t^{i,j} + J_t \left( \hat{V}_{t+1}^{i,j} - P_t \right) J_t^T, \end{aligned} \tag{4.23}$$

where we have defined $J_t = \overline{V}_t^{i,j} {F_j}^T (P_t)^{-1}$. Using these equations we update the object states in the E-step, and calculate in the A-step, association probabilities that consider this enhanced motion history of the objects. Being able to bootstrap the estimated state trajectories with the estimated data association and vice-versa is particularly important in cases of association ambiguity, i.e. when more than one detection is close to an individual object (see Example 4.3) or when an object gets occluded for a small period of time (see Example 4.4).

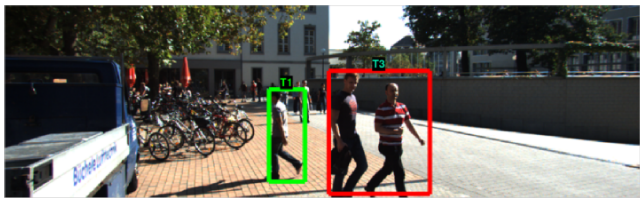
**Example 4.3**

*Consider the case when two objects get close together causing an association ambiguity. The observations obtained at this moment, although uninformative for calculating association probabilities, they still provide evidence about the localisation of the objects. Our method makes use of the evidence in ambiguous areas for localisation purposes and recovers the object identities according to their location history before merging.*




**(a)** Objects $T1$ and $T3$, and their respective detections


**(b)** Objects $T1$ and $T3$ get close, causing an association ambiguity




**(c)** Observations are correctly assigned after the group in (b) splits

**Figure 4.4** – An instance of identity disambiguation.

*As shown in Figure 4.4, although once the grouping occurs (Figure 4.4b), there is no detection associated to object $T1$ with a high probability, EA continues estimating the entire object trajectories, and recalculating the data association based on the observations available up to the current time step. This continuous flow of information between trajectory estimation and data association makes it possible to associate $T1$*

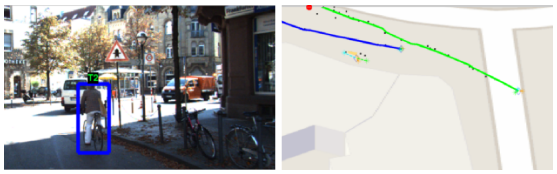
*to the right detection once the grouping is over (Figure 4.4c).*

**Example 4.4**

*A second example, similar in nature to the previous one, is an occlusion. Here, only one of the objects interacting is observed and the occluded object is sufficiently separated before getting occluded so that, no association ambiguity occurs. However, once the occluded object is observed again, it may be difficult to match this observation to its respective track, particularly for vehicles, which suffer drastic appearance changes when they are observed from different perspectives.*

*Occasionally, occlusions may cause new objects to be initialised. However, if the new observations are better explained by already-existing objects, our method naturally recalculates the data associations. This allows us to recover the identity of each object even in the event of temporary occlusions. Figure 4.5 shows an instance of this scenario.*

**(a)** Objects $T1$ (van) and $T2$ (cyclist). Detections and identities on the left and trajectories on the right



**(b)** Object $T1$ is occluded by object $T2$, so the former is not observed

**(c)** Once the van leaves the occlusion, the online association routine assigns the new detection to a new object ($T10$)

**(d)** After smoothing, the detection corresponding to the van is reassigned to object $T1$

**Figure 4.5** – An instance of identity disambiguation after an occlusion. Images with tracked objects are depicted on the left-hand side and object trajectories in a global reference frame on the right (images were cropped to ease visualisation)

## 4.5 Related Work

This section presents a review and highlights the differences between the EA algorithm proposed in this thesis and the state-of-the-art methods for both approximate variational inference and MOT.

### 4.5.1 VBEM

One of the most popular variational methods is the Variational Bayesian Expectation Maximisation (VBEM), proposed by [34]. The EA algorithm may be regarded as a special case of the VBEM algorithm. Both EA and VBEM are mathematically similar, since they seek to approximate the joint posterior distribution by eliminating the coupling between the variables that make this calculation intractable. They both estimate the posterior distribution over variables of interest in an iterative process which optimises a lower bound of the data likelihood. However they are different in that:

- VBEM decouples state variables from model parameters, whereas EA decouples state and class variables from association variables.
- VBEM is completely general, whereas EA is specialised and performs inference efficiently by exploiting the structure of the tracking and classification problem, and by taking advantage of the well-known Rauch-Tung-Striebel smoothing recursions.

### 4.5.2 GM-PHD

The Extended Target GM-PHD (ET-GM-PHD) filter [137] is a Gaussian Mixture implementation of the PHD filter for extended targets – objects that can emit multiple observations per time step [138]. The work in [138] requires each possible grouping/-partition of the observations to be considered in order to update the object's state,

which in practice, is computationally prohibitive. The authors of [137] approach this issue by using only the most probable partitions obtained using a clustering algorithm. In our EA algorithm, the partitions, which correspond to the detections from the object-detection module, are also obtained using a clustering-based approach.

Similar to EA, in ET-GM-PHD both dynamic evolution of each object state and the observation process are modelled using linear Gaussian dynamical models. This allows them to make use of the Kalman filter recursions in order to efficiently update the object states.

The main drawback of PHD-filter-based approaches is that no explicit reasoning about object identities is performed. Therefore, further post-processing is needed in order to obtain the individual state trajectories.

### 4.5.3 IHTLS

The Iterative Hankel Total Least Squares (IHTLS) method [58] performs small track association by means of efficient rank estimation of a Hankel matrix. The Hankel matrix is composed of the raw motion observations, and its rank measures the motion complexity. IHTLS performs dynamics-based data association. It creates tracks from small segments (a.k.a tracklets) that are highly likely to belong to individual objects. Then the objective is to associate together those tracklets that belong to the same trajectory.

Using the tracklets, a Hankel matrix is constructed. This Hankel matrix is incomplete due to object crossings and object occlusions. IHTLS creates a pairwise similarity matrix between tracklets by minimising the rank of the Hankel matrix. Once the similarity measure has been obtained, the final association is formulated and solved as a generalised linear assignment problem [139].

In EA, tracklets are constructed sequentially using a gating procedure. Dynamics-based tracklets association is naturally enforced by our model. As illustrated in Example 4.1, by iterating over trajectory estimation (E-step) and association estimation (A-step), EA converges to trajectories that are spatially smooth.

The main drawback of IHTLS is that it performs tracking directly in the feature/observation space. As explained before, not considering the incompleteness and noisy nature of the data precludes the method from estimating quantities that are not observed (e.g. velocities), and makes inference prone to drops in performance due to unmodeled noise.

### 4.5.4 DC

The Discrete-Continuous (DC) energy minimisation method presented in [140] introduces a discrete-continuous Conditional Random Field (CRF) for object tracking. DC is an optimisation-based approach where the cost function models how well the trajectories follow the detections; it encourages temporally smooth data association, and enforces exclusion constraints. The first exclusion constraint enforces that each object observation should support, at most one trajectory and each trajectory should be assigned, at most, one observation per frame. The second one models the fact that two trajectories should remain spatially separated at all times.

DC is a batch-type tracking approach, and it is not clear how to apply it to sequential object tracking. DC does not consider objects' appearance in its cost function. It also requires an independent tracker to generate initial trajectories for its optimisation procedure. On the other hand, EA is sequential by nature; it makes use of appearance information for calculating associations; and it does not require trajectories to be initialised by an independent tracker.

## 4.6 Summary

This chapter presented a sequential and efficient algorithm for calculating the posterior probability distribution over object states, data association, and precision weights given a set of observations and the model introduced in Chapter 3. We refer to this method as the Expectation Association (EA) algorithm. The intuition behind EA is as follows: *Once a trajectory i has been observed, its states, class assignments, and precision weight can be estimated recursively from a sequence of surrogate observations and noise covariances, which arise from expectations of the data association variables.*

We learn the parameters of our model, introduced in Chapter 3, using a training methodology based on the Expectation-Maximisation (EM) algorithm. As explained in Section 2.1.5, the EM algorithm is a two-step iterative process where the E-step calculates the Expected Sufficient Statistics (ESSs) used to calculate the model parameters in the M-step. Our training methodology uses the EA algorithm, introduced in this chapter, as the inference machinery that implements the E-step.

# Chapter 5

# Model Learning

The previous chapter presented the Expectation-Association (EA) algorithm, a sequential and iterative procedure that performs inference on the Simultaneous Tracking and Classification (STC) model. This model, whose structure encodes the domain knowledge of the problem, is defined by a set of free parameters. This chapter shows how these parameters can be learned from data.

The discrimination power of our STC model depends highly on the observation features we use. Even though utilising a larger number of features allows for better class separability, it also means that a larger number of parameters need to be fitted, and thus the complexity of the model increases. Maximum Likelihood (ML) estimation approaches tend to over-fit complex models to the data, making their generalisation power decrease with the dimensionality of the observation space. Therefore, learning approaches whose generalisation performance scales well with the model complexity are desirable. Additionally, even when parameter fitting should be automatic, most approaches to object tracking do not explicitly learn the parameters of their models, so manual tuning must be performed. Requiring user input to set the parameters makes it difficult to adapt the system to work in different environments and under different conditions, and to be used by non-expert users.

The Expectation Maximisation (EM) algorithm is commonly used to estimate the parameters of models with hidden variables. This chapter presents a method for

estimating the STC model parameters using an iterative procedure based on EM. The Expectation step (E-step) is performed using the EA algorithm. The Maximisation step (M-step) is implemented by the set of equations we derive in this chapter.

## 5.1 Parameter Learning for the STC Model

The parameters of models with latent variables are usually learned by means of ML procedures derived from the EM algorithm. Each EM iteration consists, as explained in Chapter 2, of two nested loops: the E-step and the M-step, which are iterated until convergence. This section presents an EM algorithm where the innermost loop is the E-step and it is carried out by the inference machinery provided by the EA algorithm. Please note that the E-step in the EA algorithm works as a component of E-step introduced in this section for learning purposes. The outermost loop is the M-step and it is the loop where the model parameters are updated.

Ideally, we would like to learn the model parameters by optimising the marginal likelihood of the data, however, due to the complexity added by the marginalisation procedure, a better approach would be to optimise the complete-data likelihood instead. The exact likelihood corresponds to the probability of the observations, where all hidden variables have been marginalised. The complete-data likelihood, on the other hand, is obtained by assuming all the variables in the model are observed and calculating the likelihood of the complete data points. In the case of our STC model, a complete data point would be the set of measurements: $(x^i_{1:t}, z^i_{1:t}, s^i, \omega^i)$.

In practice however, object states $x^i_{1:t}$, class assignment probabilities $s^i$ and precision weights $\omega^i$ are not given during training, thus we cannot calculate the complete-data log-likelihood either. The EM paradigm resorts to the expected value of the complete-data likelihood $\mathcal{Q}_{ML}$ under the posterior distribution of the latent variables. This expectation is given by:

$$
\begin{aligned}
\mathcal{Q}_{ML}(\mathbf{\Omega}) &= \left\langle \ln p\left(s,x,z,w|a,\mathbf{\Omega}\right)\right\rangle_{p\left(s,x,w|a;\hat{\mathbf{\Omega}}\right)} \\
&= \sum_{j=1}^{N_s} \hat{N}_j \ln p_j \\
&\quad - \frac{1}{2}\sum_{j=1}^{Ns} tr\left[V_j^{-1}\left(\eta_j - \xi_j\mu_j^T - \mu_j\xi_j^T + \hat{N}_j\mu_j\mu_j^T\right)\right] - \frac{1}{2}\sum_{j=1}^{Ns}\hat{N}_j \ln|V_j| \\
&\quad - \frac{1}{2}\sum_{j=1}^{Ns} tr\left[Q_j^{-1}\left(\varphi_j - \psi_j F_j^T - F_j\psi_j^T + F_j\phi_j F_j^T\right)\right] - \frac{T_i-1}{2}\sum_{j=1}^{Ns}\hat{N}_j \ln|Q_j| \\
&\quad - \frac{1}{2}\sum_{j=1}^{Ns} tr\left[R_j^{-1}\left(\Lambda_j - \Gamma_j H_j^T - H_j\Gamma_j^T + H_j\Phi_j H_j^T\right)\right] - \frac{T_i}{2}\sum_{j=1}^{Ns}\hat{N}_j \ln|R_j|
\end{aligned}
\tag{5.1}
$$

$$
\begin{aligned}
\hat{N}_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}, \\
\eta_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{P}_1^{ij}, & \Lambda_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=1}^{T_i} z_t^i z_t^{iT}, \\
\zeta_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{x}_1^{ij}, & \Gamma_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=1}^{T_i} z_t^i(\hat{x}_t^{i,j})^T, \\
\varphi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_t^{ij}, & \Phi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=1}^{T_i}\hat{P}_t^{ij}, \\
\psi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t,t-1}^{ij}, & \hat{P}_t^{ij} &= \hat{V}_t^{ij} + \hat{x}_t^{ij}(\hat{x}_t^{ij})^T, \\
\phi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t-1}^{ij}, & \hat{P}_{t,t-1}^{ij} &= \hat{V}_{t,t-1}^{ij} + \hat{x}_t^{ij}(\hat{x}_{t-1}^{ij})^T.
\end{aligned}
\tag{5.2}
$$

The posterior distribution $p\left(s,x,w|a;\hat{\mathbf{\Omega}}\right)$ is obtained in the E-step of the EM algorithm given a previous estimate of the model parameters $\mathbf{\Omega} = [F_j, Q_j, H_j, R_j]$. Note that the expected complete-data log-likelihood $\mathcal{Q}_{ML}$ is a function of $\hat{p}_{ij}$, $\hat{x}_t^{ij}$, $\hat{P}_t^{ij}$, $\hat{P}_{t,t-1}^{ij}$ and $\hat{\omega}^{ij}$, which are the Expected Sufficient Statistics (ESSs) provided by the EA algorithm. What is more, $\hat{x}_t^{ij}$, $\hat{P}_t^{ij}$ and $\hat{P}_{t,t-1}^{ij}$ are given by the conditional expecta-

tions $\langle x_t^i | s^i = j \rangle$, $\left\langle x_t^i {x_t^i}^T | s^i = j \right\rangle$ and $\left\langle x_t^i {x_{t-1}^i}^T | s^i = j \right\rangle$ respectively, which intuitively, represent the estimated trajectory associated to object $i$ according to the mixture component $j$.

In the classic ML M-Step, model parameters are estimated by maximising $\mathcal{Q}_{ML}$ in Equation 5.1. These parameters can have any form, which makes the ML-EM algorithm prone to failure as a result of singularities or degeneracies. During training, covariance matrices might get too close to being singular when for example, none or a very small number of observations support a given assignment to a particular class or mixture component. In order to constrain the parameter space, we propose to impose a prior distribution over the parameters to be learned, which is equivalent to regularising the cost function for each individual model parameter. With this addition, the MAP M-step's objective function $\mathcal{Q}(\mathbf{\Omega})$ is given by the expected complete-data log-likelihood $\mathcal{Q}_{ML}(\mathbf{\Omega})$ –as in the ML case– plus the natural logarithm of the prior over parameters:

$$\mathcal{Q}(\mathbf{\Omega}) = \mathcal{Q}_{ML}(\mathbf{\Omega}) + \ln p(\mathbf{\Omega}). \tag{5.3}$$

Here $p(\mathbf{\Omega})$ is the prior distribution on the model parameters $\mathbf{\Omega}$, and for our model, it factorises as follows:

$$p(\mathbf{\Omega}) = p(F|Q)\, p(Q)\, p(H|R)\, p(R). \tag{5.4}$$

For the prior distributions over process and observation matrices ($p(F|Q)$ and $p(H|R)$), we utilise *matrix-variate Gaussian* [141] distributions conditioned on the process-noise- and observation-noise covariance matrices ($Q$ and $R$) respectively. Using the matrix variate Gaussian distribution allows us to model the $F$ and $H$ as rectangular random matrices. For the prior distributions over covariances matrices ($p(Q)$ and $p(R)$), we use *inverse Wishart* distributions, a family of distributions that allows us to model $Q$ and $R$ as realisations of real-valued positive definite random matrices. Additionally, by parametrising the process- and observation matrices using their respective covariance matrices we perform parameter tying, thus reducing the number of effective parameters that need to be learned. The prior distributions over the pairs

$F/Q$ and $H/R$ are therefore given by:

$$p\left(F|Q\right)p\left(Q\right) = \mathcal{N}_{m\times m}\left(F|\Lambda_f, Q, \Omega_f\right)\mathcal{W}^{-1}\left(Q|\nu_q\Sigma_q, \nu_q\right), \tag{5.5}$$

$$p\left(H|R\right)p\left(R\right) = \mathcal{N}_{m\times n}\left(H|\Lambda_h, R, \Omega_h\right)\mathcal{W}^{-1}\left(Q|\nu_r\Sigma_r, \nu_r\right). \tag{5.6}$$

Substituting Equation 5.5 and Equation 5.6 into Equation 5.4, the prior over parameters is given by:

$$p\left(\mathbf{\Omega}\right) = \mathcal{N}_{m\times m}\left(F|\Lambda_f, Q, \Omega_f\right)\mathcal{W}^{-1}\left(Q|\nu_q\Sigma_q, \nu_q\right)\mathcal{N}_{n\times m}\left(H|\Lambda_h, R, \Omega_h\right)\mathcal{W}^{-1}\left(R|\nu_r\Sigma_r, \nu_r\right). \tag{5.7}$$

Learning the parameters of our model is, in summary, a process that iterates over calculating the posterior via the EA algorithm and then updating the model parameters so that the regularised version of the expected complete-data log-likelihood is maximised. Algorithm 3 presents a pseudo algorithm that summarises our MAP-EM approach.

**Algorithm 3** The MAP-EM algorithm for learning the model parameters

```
p(Ω) ← Initialise prior distribution over parameters
Ω̂ ← Initialise model parameters
EMits ← Number of EM iterations
for i ← 1, EMits do
    procedure E-STEP(z, Ω̂)
        q(x, s, w) = EA(z, Ω̂)
        Return q(x, s, w)
    end procedure
    procedure M-STEP(q(x, s, w), p(Ω))
        Update model parameters as in Section 5.2.
        Return Ω̂ = F̂, Q̂, Ĥ, R̂, μ̂, V̂
    end procedure
end for
```

## 5.2 Update Equations of the Model Parameters

Now that our new objective function has been completely defined, we need to obtain the corresponding objective function for each parameter. To do so, we gather the terms in $\mathcal{Q}(\Omega)$ that are a function of the parameter to be optimised. These terms are extracted from Equation 5.1 and Equation 5.7.

By taking the derivative of the objective function for the transition Matrix $F$ and setting to zero, the update equation for $F_j$ becomes:

$$\begin{aligned} \hat{F}_j &= \underset{F_j}{\operatorname{argmax}}\, \mathcal{Q}(F_j) \\ &= \Big(\Lambda_f \Omega_f + \psi_j\Big) \Big/ \Big(\Omega_f + \phi_j\Big). \end{aligned} \tag{5.8}$$

Note that the terms $\Lambda_f \Omega_f$ and $\Omega_f$ in Equation 5.8, are contributed by the prior over parameters. It can be seen that they work as regularisation terms. In the ML case, very small values for the assignment probabilities $\hat{p}_{ij}$ or a mixture component collapsing onto an individual data point, i.e. $\hat{p}_{i=k,j} \approx 1$, would cause singularities in the complete-data likelihood. Imposing a prior distribution in the MAP case results in an update equation that, either assigns $\Lambda_f$ to $\hat{F}_j$ if $\hat{p}_{ij}$ has very small values, or shifts the ML solution if this one has collapsed onto a particular data point. The larger the contribution from the training data $i$, the smaller the effect of the prior is. Furthermore, the contribution from the training trajectory $i$ is also weighted by the precision weight $\hat{w}_i$, which tends to zero when the trajectory follows a non-modelled dynamic.

By optimising the objective function for the process noise covariance $Q$, we obtain the update equation for $Q_j$:

$$\begin{aligned}\hat{Q}_j =& \underset{Q_j}{\operatorname{argmax}}\, \mathcal{Q}\left(Q_j\right) \\ =& \frac{1}{\nu_q + 2m + 1 + \sum_{i=1}^{N_x} \hat{p}_{ij}\left(T_i - 1\right)} \times \\ & \left(\varphi_j - \psi_j F_j^T - F_j(\psi_j)^T + F_j \phi_j F_j^T + \left(F_j - \Lambda_f\right) \Omega_f \left(F_j - \Lambda_f\right)^T + \nu_q \Sigma_q\right).\end{aligned} \tag{5.9}$$

Following the same line of reasoning of the previous optimisation operations, we derive the update equations for the observation matrix $H_j$ and the observation noise covariance $R_j$:

$$\begin{aligned}\hat{H}_j =& \underset{H_j}{\operatorname{argmax}}\, \mathcal{Q}\left(H_j\right) \\ =& \left(\Lambda_h \Omega_h + \Gamma_j\right) \Big/ \left(\underline{\Omega_h} + \Phi_j\right)\end{aligned} \tag{5.10}$$

$$\begin{aligned}\hat{R}_j =& \underset{R_j}{\operatorname{argmax}}\, \mathcal{Q}\left(R_j\right) \\ =& \frac{1}{\nu_r + n + m + 1 + \sum_{i=1}^{N_x} \hat{p}_{ij} T_i} \Bigg( \Lambda_j - \Gamma_j H_j^T - H_j(\Gamma_j)^T + H_j \Phi_j H_j^T \\ & + \left(H_j - \Lambda_h\right) \Omega_h \left(H_j - \Lambda_h\right)^T + \nu_r \Sigma_r \Bigg).\end{aligned} \tag{5.11}$$

The update equations for the process- and observation-noise covariance matrices (Equation 5.9 and Equation 5.11) involve additions and subtractions of outer-product expectations. Since these operations are numerically non-stable, especially for large matrices, in Equation C.21 and Equation C.26 we provide modified versions of these equations, that are numerically more stable but analytically equivalent.

### 5.2.1 Selecting the Prior Hyper-Parameters

The prior distributions on model parameters are parametrised by a set of hyper-parameters that can be chosen based on Empirical Bayesian Estimation (EBE) [24], Hierarchical Bayesian Estimation (HBE) [23], or expert or contextual knowledge.

With EBE, point estimates of the hyper-parameters are calculated from the data using the EM algorithm. This approach can become overconfident because it uses the data twice, particularly in the presence of outliers, and it does not model the uncertainty in the hyper-parameter values. On the other hand, HBE imposes a prior distribution on the hyper-parameters thus it accounts for the uncertainty in their values at the price of increasing the complexity of the learning procedure.

Object tracking is an application where prior expert knowledge about model parameters is available [17, 85]. This allows us to model the actual motion modes as samples from a distribution parametrised using this knowledge. If we allow the hyper-parameters to be mean- and variability beliefs, mean values can be given by basic models described in the literature (constant velocity, constant acceleration) and variability values can reflect to what extent the objects in the environment are expected to fulfil the assumptions established by these basic models. A constant-velocity model, for example, approximates the motion of a car, better than it approximates the motion of a manoeuvring pedestrian.

We select the matrix blocks in $\Lambda_f$, that correspond to kinematic states ($x$-and-$y$ positions and velocities), as being the transition matrix of a constant-velocity model. And for those states that describe object appearance, we set a constant-position model where the change in the state is assumed to be a Gaussian random variable with zero mean:

$$\Lambda_f = \left[\begin{array}{cc:c} 1 & \Delta T & 0 \\ 0 & 1 & 0 \\ \hdashline 0 & 0 & 1 \end{array}\right], \tag{5.12}$$

where $\Delta T$ is the sampling time. Since we know what variables in the state space are observed, i.e. only first order variables, the blocks in the location hyper-parameter

$\Lambda_h$ are set accordingly:

$$\Lambda_h = \left[\begin{array}{cc:c} 1 & 0 & 0 \\ \hdashline 0 & 0 & 1 \end{array}\right]. \tag{5.13}$$

The scale matrices $\Sigma_q$ and $\Sigma_R$ for the prior over noise covariance matrices have the following form:

$$\nu_q \Sigma_q = \nu_q \left[\begin{array}{cc:c} \Delta T^3/3 & \Delta T^2/2 & 0 \\ \Delta T^2/2 & \Delta T & 0 \\ \hdashline 0 & 0 & 1 \end{array}\right] \sigma_q^2, \tag{5.14}$$

$$\nu_r \Sigma_r = \nu_r \left[\begin{array}{c:c} 1 & 0 \\ \hdashline 0 & 1 \end{array}\right] \sigma_r^2, \tag{5.15}$$

where $\sigma_q$, $\sigma_r$ are arbitrary standard deviation terms for the process- and observation noises. The minimum possible value for the degrees of freedom of an inverse-Wishart distribution is determined by its dimensionality. They define to which extent, the density of the distribution spreads over the space of covariance matrices. Intuitively, the smaller the degrees of freedom is, the more uncertain our knowledge about the true covariance matrix is. The values for the degrees of freedom $\nu_q$ and $\nu_r$, are therefore chosen to be small so that the priors over the covariance matrices are highly dispersed:

$$\begin{aligned} \nu_q &= m + 2 \\ \nu_r &= n + 2 \end{aligned} \tag{5.16}$$

For the particular case of the prior over assignment probabilities $p(s^i)$, these probabilities can be defined by a uniform distribution, which reflects the belief that any of the motion classes can be observed with the same probability. It could also be based on similarity metrics obtained by comparing object detections to precomputed appearance models.

## 5.3 Summary

This chapter developed the update equations for all of the parameters in the model presented in Chapter 3. These parameters are the ones that characterise the motion modes in the environment and the performance of the sensor when measuring the state of different kinds of objects. The EA algorithm introduced in Chapter 4 uses these parameters to perform tracking and classification in a framework that accounts for the multi-class nature of dynamic environments. The update equations we have developed here implement the M-step of our MAP Expectation Maximisation (MAP-EM) approach for automatic parameter estimation. In this approach the complexity of the model parameters has been constrained by means of Bayesian regularisation, which also permits the introduction of expert knowledge about model parameters with uncertainty values attached to it, and prevents singularities in the learning process. The E-step for parameter estimation is performed using the inference machinery provided by the EA algorithm.

# Chapter 6

# Experimental Results

This chapter presents evaluations of our simultaneous tracking and classification approach and comparisons with the state of the art. For the evaluation we use data collected in urban environments, which are extremely challenging due to the large number of objects and variations in their behaviours. The dataset is composed of 21 sequences of stereo images that are part of the public KITTI dataset [142] and contain a significant number of pedestrians and cars interacting in the field of view of a moving platform. The position of the ego-vehicle, along with ground-truth object detections are provided. Detections also convey ground-truth information about the object class and data association across time, which was used for evaluation purposes. We separated the dataset into training/validation- and testing sub-datasets. The training/validation sub-dataset was obtained from 14 sequences out of the 21 that constitute the complete dataset. For the testing sub-dataset, we used the remaining 7 sequences.

Section 6.1 explains the process by which we obtain object detections from stereo-vision data. Section 6.2 provides some details about the implementation of the learning module. A quantitative performance evaluation is presented in Section 6.3 and Section 6.4. Finally, Section 6.5 provides a qualitative evaluation of the classification- and unexpected-behaviour-detection capabilities of the method.

## 6.1 Object Detection and Feature Extraction

The stereo-vision sensor is regarded as one of the most cost-effective sensor modalities. It is composed of two monocular cameras whose field of view overlaps. After a calibration (both extrinsic and intrinsic) process, the 3D geometry of the environment in the overlapping area can be recovered by matching pixels in both images and triangulating using the calibration of the cameras. As a result, both appearance- and geometry information can be obtained. The stereo rig used in the KITTI dataset comprises two 1.4 *Megapixel Point Grey Flea 2* cameras with a baseline of approximately 54cm.

Stereo-based object detection is performed as follows. Firstly, a point cloud of the entire scene is obtained by stereo processing the left and right images at each time step. Since the images are rectified (see Figure 6.1), the `disparity` function from Matlab is used to obtain a point cloud of the environment (see Figure 6.2). The ground of this point cloud is segmented out by fitting a horizontal plane using least squares estimation in disparity image space [143].

**Figure 6.1** – Rectified left and right stereo images. If images are rectified, two corresponding features are horizontally aligned

After segmenting the ground plane, the remaining 3D points are projected onto a polar grid parallel to this plane (see Figure 6.3). Subsequently, areas on the grid with high point density are segmented out and re-projected onto the image plane in order to obtain individual object detections, as shown in Figures 6.4 and 6.5.

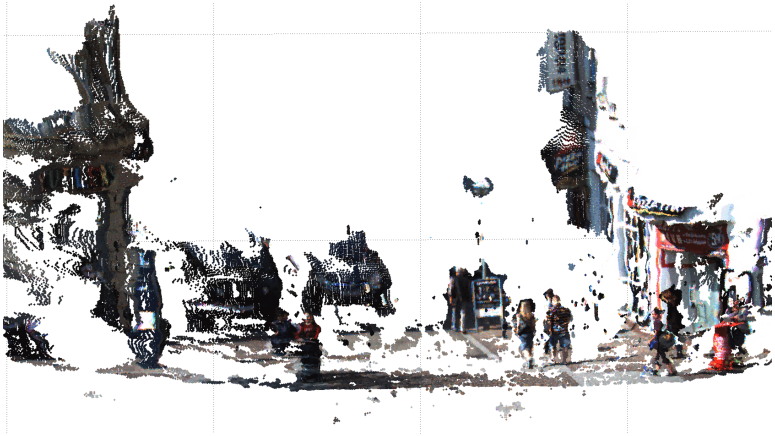


**Figure 6.2** – Point cloud reconstruction obtained from processing stereo images

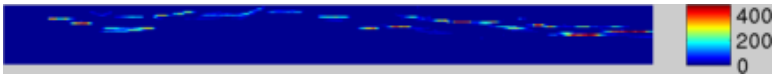



**Figure 6.3** – Polar grid count. The point cloud is projected onto a grid on the ground plane where each cell has the count of the number of points in it

**Figure 6.4** – Segmentation of the polar grid count.

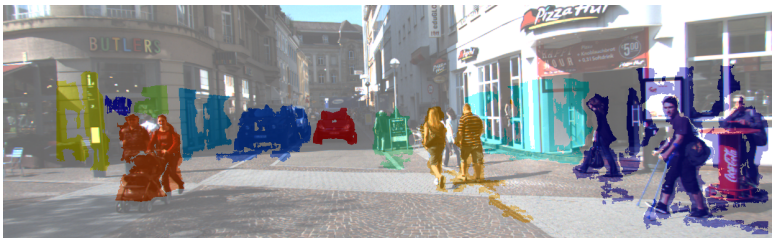


**Figure 6.5** – Detections on the image plane.

### 6.1.1 Feature Extraction

In order to define the sections in the image from where features are to be obtained, we use the 3D detections extracted from the stereo-images as previously explained, and 2D image-based detections. In this thesis, 2D detections are obtained from the KITTI dataset in the first part of the tracking evaluation and using the Randomised Prim's

(RP) algorithm for object proposals generation [114] in the second part. Observations are obtained from the portion of the point cloud given by the intersection between 2D and 3D detections. The motivation behind this is that 2D arbitrary object detection is widely available, however, the 2D bounding boxes generated by the detector contain both foreground and background. Our feature extraction scheme removes the background and allows only points on the objects of interest to be summarised by the observations. As illustrated in Figure 6.6, our feature extraction scheme has the following steps:

1. Obtain the 2D boxes accompanying the dataset.
2. Obtain binary masks by performing 3D object detection on the point cloud.
3. Extract features from the portion of the point cloud designated by the areas where the binary masks fall into the 2D detections.

The set of features extracted from the selected areas on the image represent both the dynamics and the appearance of the objects. In this implementation, dynamics is represented by the coordinates of the detection's centroid in a global reference frame, whereas appearance is represented by the skewness and mean of its width, and the entropy of its colour histogram.

As shown in Figure 6.7, colour and width of the detections are used as the input to our appearance-based feature-extraction module. The first feature is the entropy of the colour histogram. In order to eliminate the background when calculating this feature, we calculate the histogram of each colour band using the mask from the 3D detection. We then calculate the entropy of each histogram by calculating their distance to the uniform distribution. Finally, we calculate the mean of the per-band entropies. This feature captures the property that the colour histograms of cars tend to have peaked modes, whereas cyclists and pedestrians have a wider spectrum.

The other two features extracted are the mean and skew of the object's width from the perspective of the camera. To calculate these features we first extract the patch of $x$ coordinates associated with the detection. We then calculate the difference between

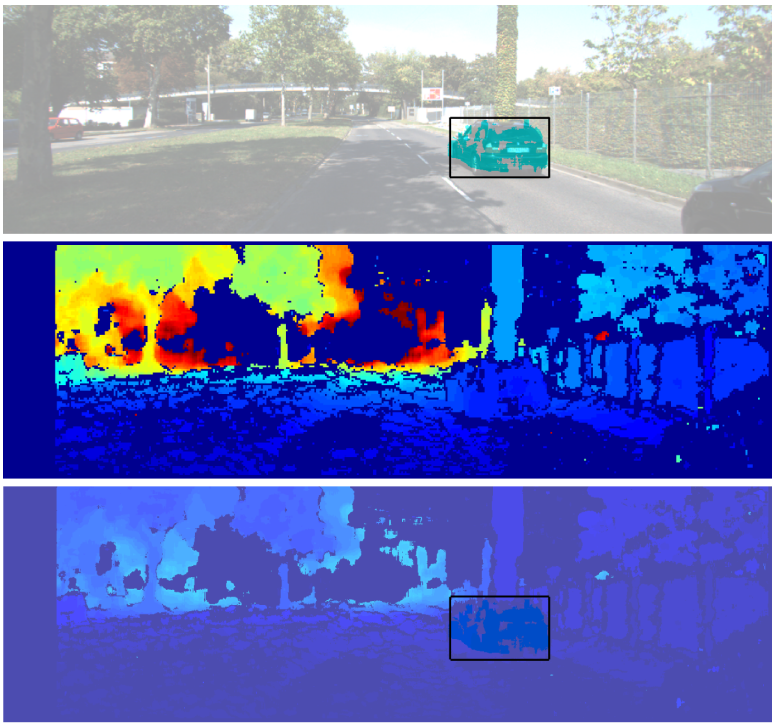
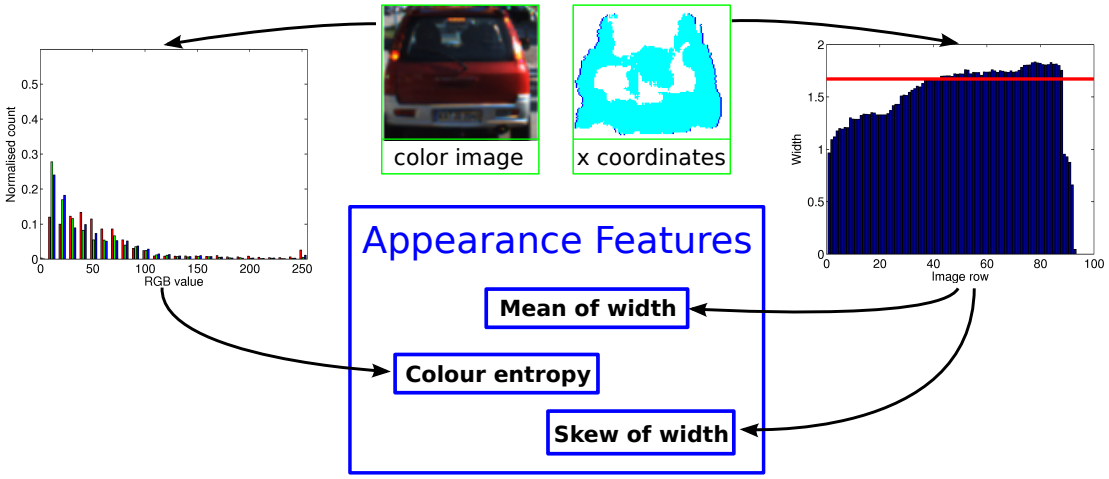


**Figure 6.6** – An example of the feature-extraction scheme. The black bounding box on the top image defines a 2D detection, whereas the binary mask represents its respective 3D detection. Features are extracted from the intersection between 2D- and 3D detections

the max and min values ($x$ coordinates) per row of the point cloud. The mean of this set of widths, besides being different for each of the object classes, tends to change drastically for cyclists according to the perspective of the camera, whereas it changes moderately for cars and pedestrians. The skew on the other hand, tends to be, for cyclists positive and higher than for pedestrians and cars.

The appearance features introduced in this section are by no means expected to provide an optimal performance. The optimal set of features in the context of simultaneous tracking and classification should optimise both inter-class discrimination, for better classification, and inter-object discrimination for better data association, which tend to be opposing characteristics. Feature engineering is considered, in fact, an art more than a science, and further investigation into this topic is outside the scope of the thesis. The process of defining observation features can also be approached by




**(a)** Appearance features - Car


**(b)** Appearance features - Cyclist


**(c)** Appearance features - Pedestrian

**Figure 6.7** – Appearance features obtained from the color and geometry of the detections. The features are the entropy of the colour histogram, and the mean and skew of the width of the detection in the image plane. Each width value is obtained from the difference between the max/min $x$ coordinate values at each row of the structured point cloud associated with this detection

means of feature learning techniques [144–146], which provide ways of automatically obtaining convenient representations of the raw detections. Note that the feature (observation) extraction procedure explained here is used for both learning and inference (tracking).

## 6.2 Model Initialisation and Learning

In this implementation, we use one mixture component per object class. Object classes were chosen to be *Static Car*, *Moving Car*, *Cyclist* and *Pedestrian*, which are the prevalent subsets of all the object categories contained in the KITTI dataset (eight classes in total). Each training instance consists of a sequence of temporally ordered features extracted from the detections at every time step. Figure 6.8 shows a sequence of point cloud segments corresponding to one training sequence. The model parameter $H$ is shared by all of the classes due to the fact that only one sensor is used and the model between states and observation features is known.

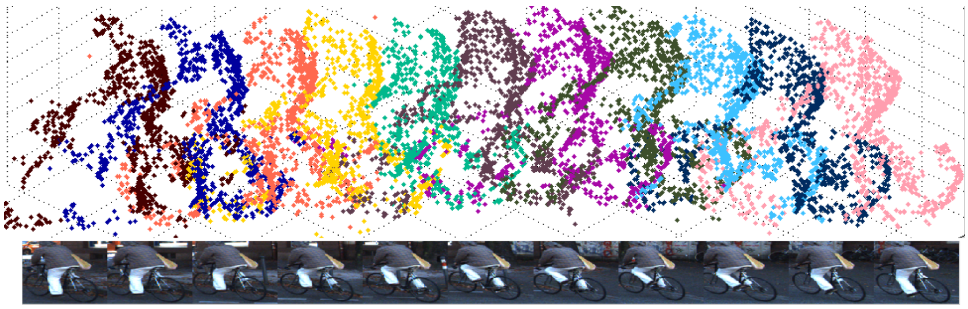


**Figure 6.8** – Sequence of filtered point clouds and image patches that constitute a training instance

The hyper-parameters of the model were initialised as explained in Section 5.2.1. The use of the EM algorithm guarantees that the likelihood increases at every iteration. Figure 6.9 shows a plot of the likelihood of the data for 500 iterations. The learning process was stopped when the change in the likelihood was negligible ($10^{-2}$).

In order to divide the training sequences into training, validation and test sets, three-fold cross-validation was used as follows:

- Randomly partitioned the available sequences into three groups.
- Chose the first two groups for training and validation.
- Used the third set for testing.

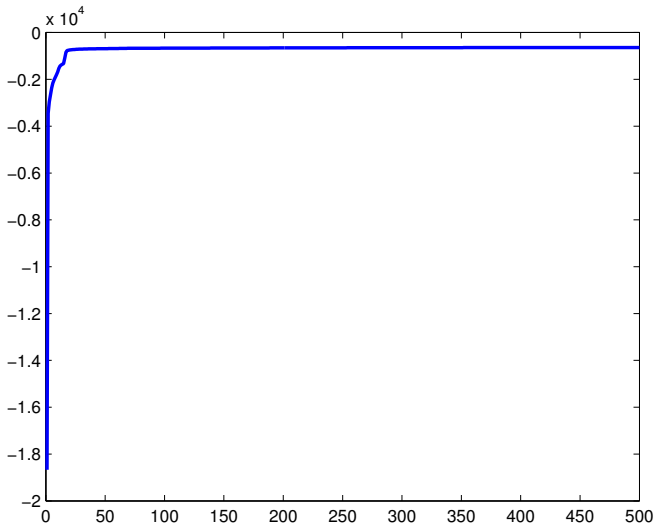



**Figure 6.9** – Likelihood of the data after 500 iterations of MAP-EM algorithm

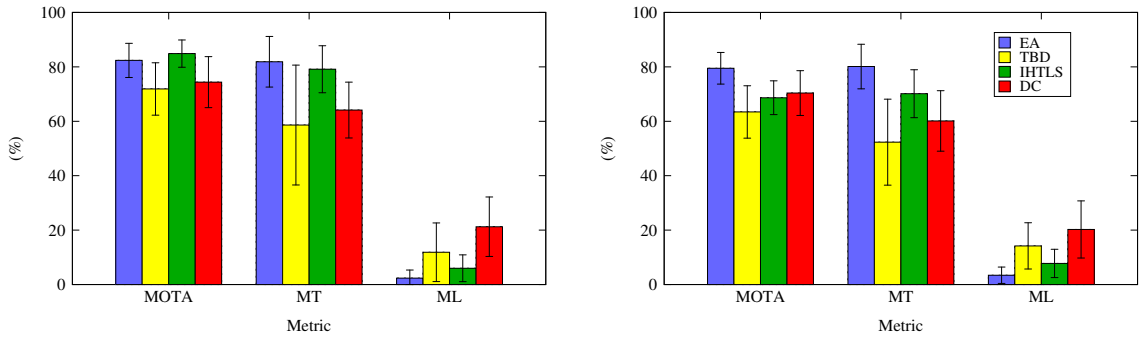
- Measured classification performance on the testing sub-dataset.
- Repeated the process three times.
- Chose the best model.

## 6.3 Performance Evaluation - Tracking

In this section, the EA algorithm is compared with state-of-the-art approaches in multi-object tracking. Discrete-Continuous energy minimisation [140] (DC), the Hungarian method for bipartite matching [46] (the authors call their method Tracking By Detection (TBD)) and Iterative Hankel Total Least Squares [58] (IHTLS) were used as baseline methods. For all cases, the comparisons were run using the code provided by the authors.

Fig. 6.10a reports the performance of our approach and that of the compared methods for detections obtained as explained in Section 6.1. The results were obtained with 7 sequences of the KITTI dataset, that have a total length of approximately 5 minutes.

They show that on average, our method performs better than the others, except for IHTLS. Given how close the MOTA metrics for EA and IHTLS were, we conducted a two-sample t-test for equal means. The null hypothesis is accepted with *p-value* of 0.0165, which means that there is not statistically significant difference between the MOTA metrics for EA and IHTLS. The advantage of our online EA over IHTLS is, however, that our system is sequential and uses past information from small windows (12 frames in our experiments). On the other hand, IHTLS was designed to work offline, so it needs information from entire trajectories across all frames in the sequence.




**(a)** Quantitative evaluation with detections obtained as explained in Section 6.1.1

**(b)** Quantitative evaluation with noisier detections. These detections were obtained as explained in Section 6.3.1

**Figure 6.10** – Quantitative evaluation of tracking performance. Better scores correspond to bigger values of *MOTA* and *MT*, and smaller for *ML*. Figure 6.10a shows the results using the detections provided by the dataset. Figure 6.10a shows the results using noisier detections, thus evaluating the robustness of the compared methods against noise (see Section 6.3.1).

### 6.3.1 Robustness against noise

In addition to the key advantage of online performance, our approach models the state of the objects using hidden variables, whereas IHTLS uses the raw observations. Using raw observations means there is an underlying assumption that they provide perfect and complete measurements of the object states. Under our framework, estimation of the data association is done using a smoothed version of the observations, which

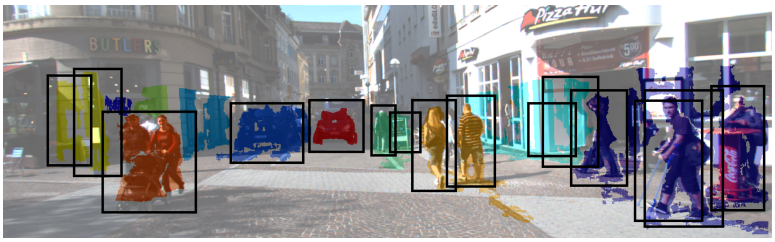


**Figure 6.11** – Sub-partitions on the original detections.

makes our approach more robust to noise. In order to verify this property under a realistic setting, the detection method was modified so that candidate detections from the grid segmentation were further partitioned and provided to the tracker without any preprocessing (see Fig. 6.11). We generated these sub-partitions using the Randomised Prim's (RP) algorithm. Sub-partition methodologies are widely used in order to track objects with different geometries or that are moving close to each other.

The bar graph in Fig. 6.10b shows how the compared methods have an average drop in performance of 10%, whereas the effect on the performance of EA was less than 5% for all of the metrics. Compared against IHTLS, EA performs better, with 99.54% confidence (i.e. with a p-value of 0.0046).

## 6.4 Performance Evaluation - Classification

Table 6.1 and Table 6.2 present the confusion matrices for the classification results using our EA algorithm. In the first table, tracking was performed using position observations only, whereas in the second one appearance features were also included. These tables were built by assigning to each object the class label that it took with the highest frequency while it was in the camera's field of view. Values in the main diagonal represent instances of objects to which the correct class category was assigned.

The confusion matrix in Table 6.1 shows that the class *Cyclist*, and both the classes *Car* and *Pedestrian* overlap strongly. In order to verify the class-overlapping issue,

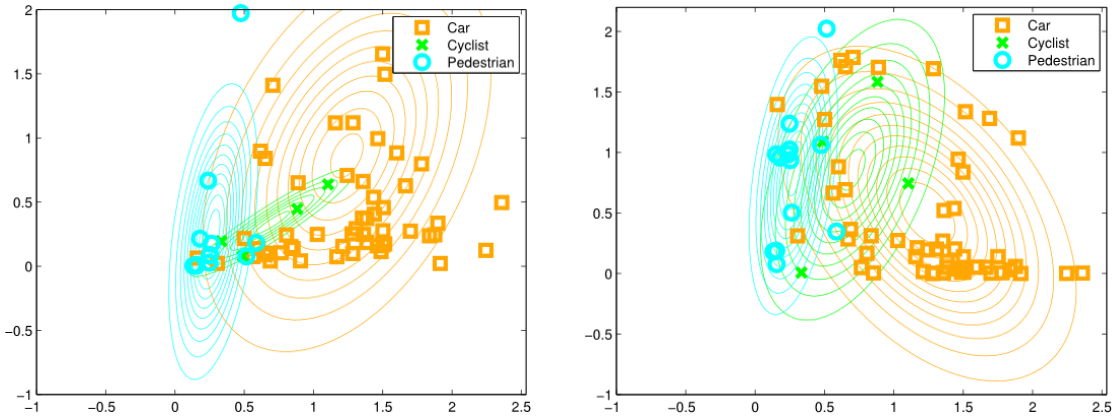
**Table 6.1** – Confusion matrix with the classification results during tracking (%) using position observations only. The last column shows the number of tracked objects per class

| Act.\Pred. | Static | Car | Cyclist | Pedestrian | Total |
|---|---|---|---|---|---|
| **Static** | 71 | 0 | 0 | 29 | 127 |
| **Car** | 0 | 71 | 26 | 3 | 86 |
| **Cyclist** | 6 | 12 | 76 | 6 | 17 |
| **Pedestrian** | 23 | 2 | 5 | 70 | 96 |

**Table 6.2** – Confusion matrix with the classification results during tracking (%) using position and appearance observations. The last column shows the number of tracked objects per class

| Act.\Pred. | Static | Car | Cyclist | Pedestrian | Total |
|---|---|---|---|---|---|
| **Static** | 96 | 0 | 0 | 4 | 127 |
| **Car** | 0 | 89 | 11 | 0 | 86 |
| **Cyclist** | 0 | 6 | 94 | 0 | 17 |
| **Pedestrian** | 6 | 0 | 3 | 91 | 96 |

we evaluated the mean speed, and the speed and heading variances of each instance in the training set. We then fit Gaussian PDFs to these features. Fig. 6.12 shows how classes *Car* and *Pedestrian* are well separated in this feature space, whereas class *Cyclist* overlaps the other two classes. Also, cars and cyclists with low velocities, tend to be classified as pedestrians.

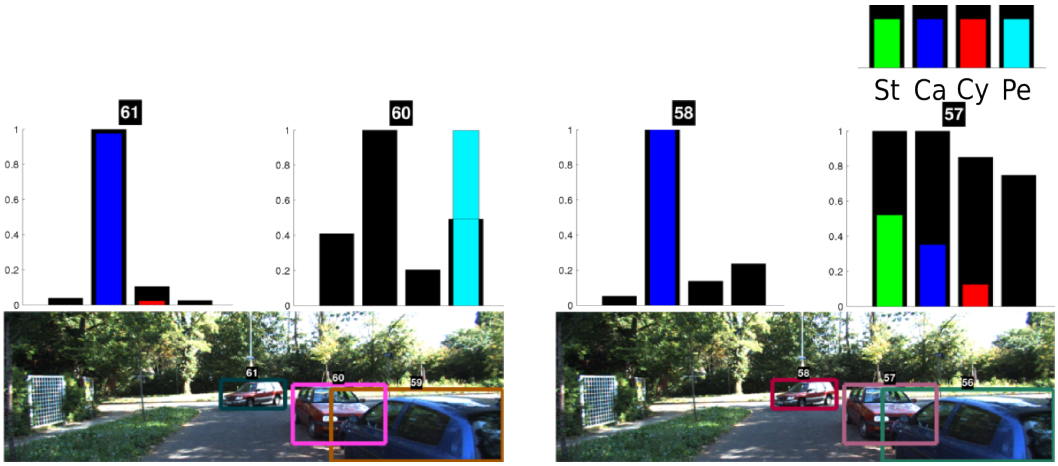



**(a)** Speed means versus (x axis) speed variances (y axis)

**(b)** Speed means versus (x axis) heading variances (y axis)

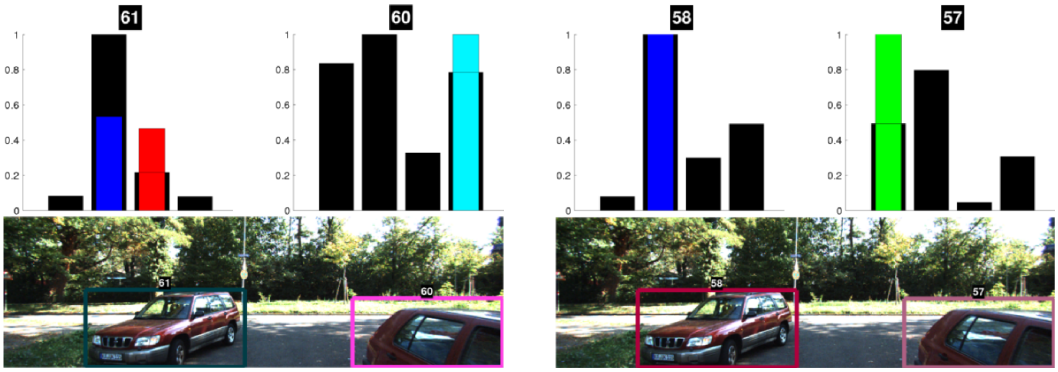
**Figure 6.12** – The issue of class overlapping. Each data point used to fit the per-class Gaussian distributions corresponds to both the speed and heading variance of each training instance versus its mean velocity. These features are commonly used to summarise dynamic behaviour and in our particular case show the similarity between instances of the class *Cyclist* and instances of the other two classes

One way to approach the class-overlapping problem is by adding new features. In our implementations we have evaluated the use of appearance descriptors as a way to boost the performance of simple dynamic descriptors. By obtaining new features from appearance rather than by manipulating the dynamic ones (positions), we ensure that they are conditionally independent given the true states, in other words, that no information is being reused. Using appearance as part of the state space adds discrimination power to our method and thus diminishes class overlapping, as shown in Table 6.2. Figure 6.13 illustrates the improvement in discrimination power obtained by augmenting the observation space with appearance features.




**(a)** STC using positions only - frame 536.

**(b)** STC using both positions and appearance - frame 536.




**(c)** STC using positions only - frame 552.

**(d)** STC using both positions and appearance - frame 552.

**Figure 6.13** – A qualitative comparison between the scenario when STC is performed using position only (scenario 1), and both appearance and position (scenario 2) as observation features. For scenario 1, Figure 6.13a shows how object 60 is wrongly classified as Pedestrian, whereas for scenario 2 in Figure 6.13b, the same object is correctly classified as Static car. Later at frame 552 for scenario 1 the class assignment for object 61 becomes ambiguous, while object 60 is still misclassified. In scenario 2 both objects are confidently and correctly classified.

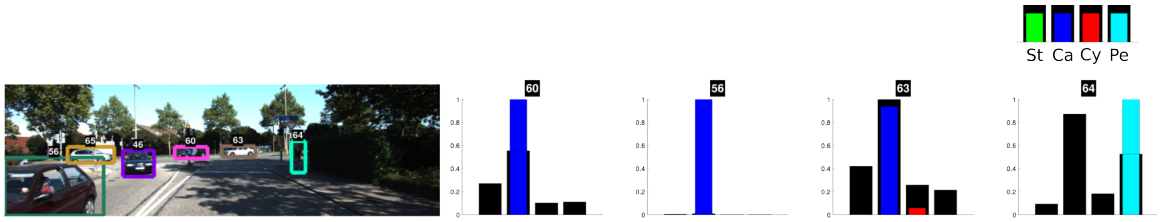
## 6.5 Performance Examples

This section aims to illustrate the performance of the EA algorithm from a qualitative perspective. It presents some examples of the main features of the approach, such as simultaneously tracking and classifying multiple objects and solving association ambiguities due to situations such as occlusions and groupings.

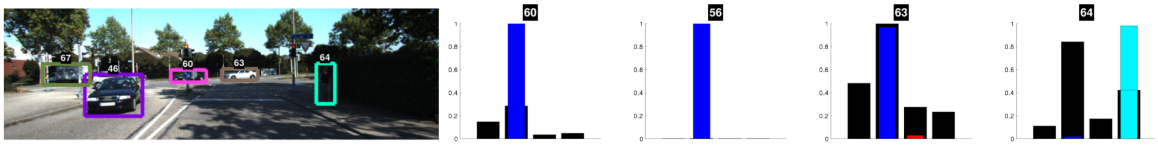
Figure 6.14 shows a group of five cars (Ca) and one pedestrian (Pe) interacting in the field of view of the vehicle. This particular scenario illustrates most of the practical advantages of the EA algorithm:

- **All objects are correctly tracked and classified**. Since the EA algorithm performs inference on a mixture model over trajectories, it can simultaneously track and classify objects that belong to different classes. In Figure 6.14, object 64 is a pedestrian, whereas the other objects are cars, and our method correctly estimates their classes. Note that object 64's precision weight under the class Pe is relatively small. This is likely due to the ambiguity added to the appearance features by the lack of illumination in that part of the scene.

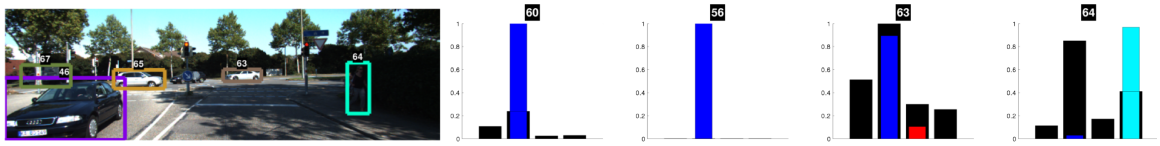
- **Object 65 gets occluded in Figure 6.14b but its identity is recovered as shown in Figure 6.14c**. Identity recovery after occlusions is one of the main practical advantages of our probabilistic approach. Even when an object stops being detected, EA continues predicting its location based on the previous observations, making it possible to associate the detections after the occlusion to their corresponding object, if their temporal evolution supports the dynamic behaviour estimated before the occlusion.




**(a)** Frame 364




**(b)** Frame 368

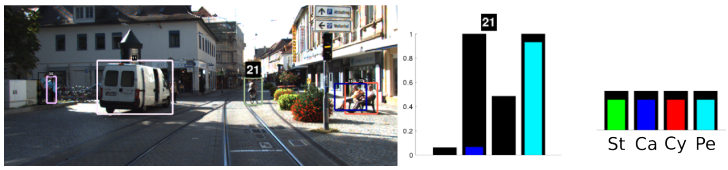



**(c)** Frame 372

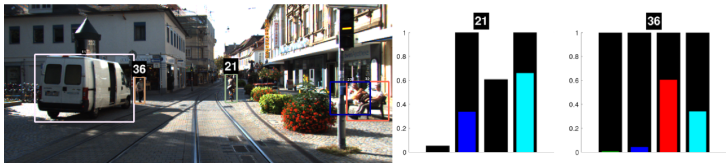
**Figure 6.14** – An exemplar output of the EA algorithm on sequence 01. Object classes are St: Static, Ca: Car, Cy: Cyclist, Pe: Pedestrian. Black bars behind colour bars represent the posterior precision weights per object class. Small weights (less than 0.2 represent outliers). This shows an instance of multi-object classification and occlusion handling

### 6.5.1 Average Convergence of the EA algorithm

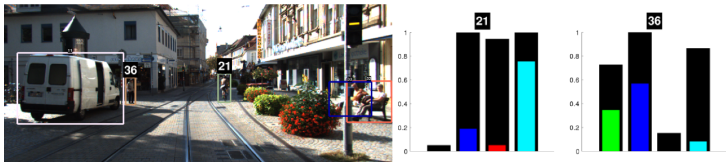
The EA algorithm takes, in average, 1.2 seconds to converge to a confident classification solution. Its dependency on the temporal evolution of the tracked trajectories makes assignment probabilities based on less than 8 frames considerably unreliable. In the experiments, as shown in Figure 6.15, with a frame rate of 10 fps, the EA algorithm takes on average between 8 and 16 frames to obtain stable classification solutions. As a result, a faster accurate classification would require a frame rate of more than 10 fps.



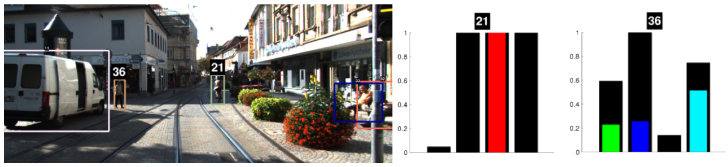

**(a)** Frame 107: Object 21 is updated using observations from only two frames, and it is misclassified as Pe.



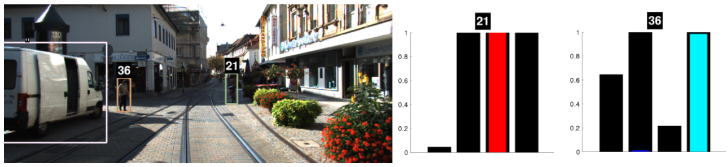

**(b)** Frame 112 - Object 36 is updated with only two frames, and it is misclassified as Cy.




**(c)** Frame 113 - objects have been updated using observations from 7 and 3 frames respectively, and their classification is still erroneous.




**(d)** Frame 115 - Object 21's classification converges to the correct one after processing 10 frames.




**(e)** Frame 118 - Object 36's classification converges to the correct one after processing 8 frames.


**(f)** Frame 132 - Object 36 leaves the sensor's field of view.


**(g)** Frame 172 - Object 21 leaves the sensor's field of view.

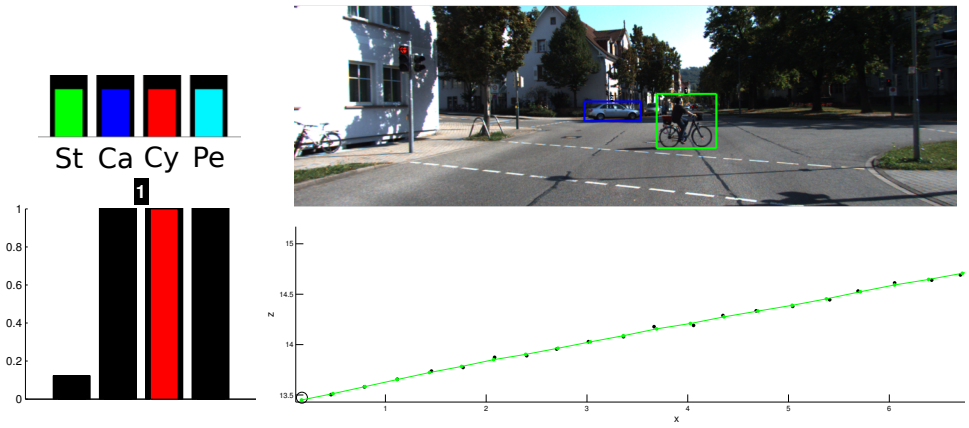
**Figure 6.15** – An example of uncertain classification with a small number of frames. The example considers the class assignments estimated for objects 21 and 36 in sequence 13. It illustrates how our approach requires, on average, more than 8 frames for classification using a video input of 10 fps.

### 6.5.2 Unexpected Objects

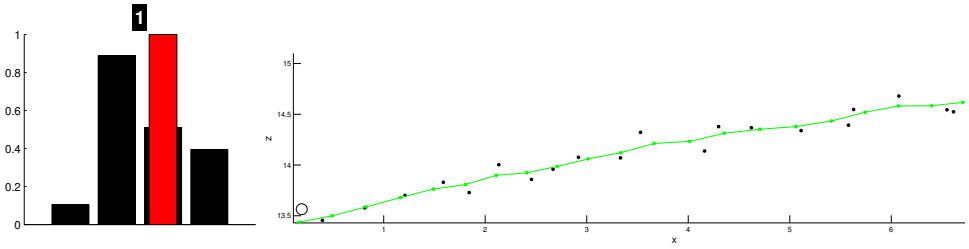
We defined *unexpected objects* as those whose behaviours are different from the previously learned. Due to the scarce nature of unexpected trajectories in the urban dataset (motions are highly structured), no ground truth is available that allows us to quantitatively evaluate the outlier detection capabilities of our method. Therefore this section presents a qualitative evaluation based on simulations and examples from the KITTI dataset. Figure 6.16a illustrates the trajectory of a cyclist estimated by the EA algorithm using the original observations along with the posterior class assignments and precision weights. They reflect how this observation sequence is well explained by our Cy category model. In contrast, Figure 6.16b, Figure 6.16c and Figure 6.16d depict the estimated trajectory after adding i.i.d. Gaussian noise $\epsilon$ to the original observations:

$$\epsilon \sim \mathcal{N}(0, \sigma). \tag{6.1}$$

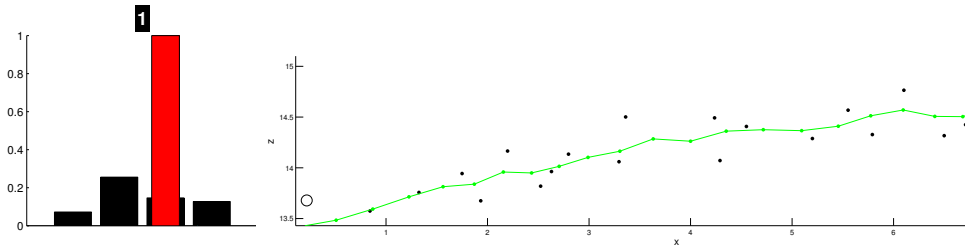
The different simulations in Figure 6.16 correspond to cases where the standard deviation $\sigma$ for the added noise was given values in the set (10cm, 20cm, 50cm). For the case when $\epsilon \sim \mathcal{N}(0, 50\text{cm})$, the precision weights (black bars in Figure 6.16) decrease to values close to zero, which indicates that the trajectory is very likely to follow a non-modelled dynamics. As the dynamics of the estimated trajectory drifts away from the modelled ones, the innovation errors get lager and thus the posterior term $\beta_{i,j}$ increases. Since the posterior precision weights $\hat{\omega}^{ij}$ are inversely proportional to $\beta_{i,j}$ (see Equation 4.17), the larger $\beta_{i,j}$ gets, the smaller $\hat{\omega}^{ij}$ becomes.



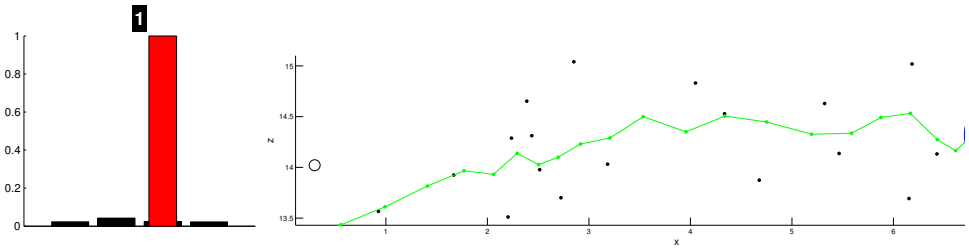

**(a)** Trajectory, class assignments and precision weights estimated from original observations




**(b)** Adding $\epsilon \sim \mathcal{N}(0, 10\text{cm})$ to raw observations.




**(c)** Adding $\epsilon \sim \mathcal{N}(0, 20\text{cm})$ to raw observations.




**(d)** Adding $\epsilon \sim \mathcal{N}(0, 30\text{cm})$ to raw observations.

**Figure 6.16** – A simulated example of unexpected behaviour detection. Coloured and black bars represent posterior assignment probabilities and posterior precision weights respectively. Observations were modified by adding different levels of Gaussian random noise to the observations. The sub-figures show how the precision weights get steadily smaller as the observation noise increases

Figure 6.17 shows examples of objects labelled by our method as having an unexpected behaviour. In this figure, buses were classified as Ca and Pe but with a very small precision weight associated to them. This indicates the unexpected nature of their dynamics/appearance.


**(a)**


**(b)**


**(c)**

**Figure 6.17** – Objects with unexpected behaviours. Coloured bars represent the assignment probabilities whereas the black bars behind them represent the posterior precision weights. Buses in all images were classified as Car in 6.17a and 6.17c, and as Pedestrian in 6.17c. Their precision weights are close to zero, which indicates that the objects follow unexpected behaviours. Although their dynamics might be close to that of a car or a pedestrian in the case of the last object, their appearance features differ from those learned.

### 6.5.3 When does the EA algorithm fail?

A particular kind of situation in which the EA algorithm tends to fail, corresponds to those cases in which two or more objects are initialised when they are close to each other and continue moving in the same vicinity. EA relies on the object trajectory histories to solve future identity ambiguities. Therefore, when the initial association between tracks and observations is ambiguous during the complete span of the tracks, the estimated data association can converge to an erroneous one, originating erroneous identity switches. This is illustrated in Figure 6.18 and Figure 6.19.

**Figure 6.18** – A case of association identity switch due to close initialisation. EA switches the identities of objects 2 and 3 due to their being initialised close to each other and continuing their trajectories in the same vicinity.

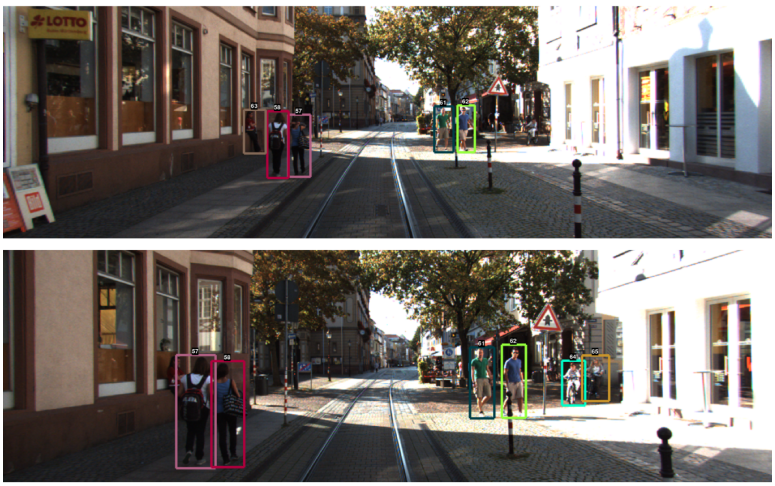


**Figure 6.19** – A case of association identity switch due to close initialisation. EA switches the identities of objects 57 and 58.

All of the baseline approaches included in the performance evaluation suffer the same drawback, except for DC [140]. The cost function of this optimisation-based approach encourages temporally smooth data association, and enforces exclusion constraints, which makes it more robust to situations in which targets are in close proximity to each other. The first exclusion constraint enforces that each object observation should support at most one trajectory, and each trajectory should be assigned at most one observation per frame. The second one models the fact that two trajectories should remain spatially separated at all times. Even though enforcing that objects should not collide, agrees with the physical reality, from the perspective of the object detection module, this is not always the case. Often an individual detection conveys information about multiple trajectories (see Example 4.3).

## 6.6 Summary

This chapter has presented both a quantitative and qualitative evaluation of the framework developed by this thesis. It started with a characterisation of the preprocessing steps for object detection and learning the STC model. It continues with an assessment of the trajectory-estimation- and data-association performance by means of the MOTA, MT and ML metrics. The experiments show that the EA algorithm has a state-of-the-art performance even in the presence of noisy measurements and presents key advantages such as online-robust estimation and classification.

Classification performance has also been evaluated. The results show a good classification performance in general. When using dynamic information only, classification is degraded when objects move at low velocities. Our approach also allows the seamlessly integration of appearance information into the estimation process, improving the discrimination power.

Finally, this chapter also presents a series of examples that illustrate the advantages of our approach. They illustrate the robust behaviour of our approach even when faced with key tracking problems such as occlusion handling and unexpected object detection.

# Chapter 7

# Conclusions

This thesis investigated the simultaneous multi-object tracking- and classification problem. It developed a new framework that formulates the problem as a Probabilistic Graphical Model on which inference is performed via an approximate variational method. This method allows us to alleviate the computational intractability issue brought about by the data association problem.

This chapter presents a summary of the contributions in Section 7.1, future research directions in Section 7.2 and concludes with a summary in Section 7.3.

## 7.1 Summary of Contributions

### 7.1.1 The STC Model

This thesis introduced the Simultaneous Tracking and Classification (STC) model, a PGM that represents the motion of multiple objects that are not equipped with identity markers, therefore it address the data-association problem.

As opposed to most traditional tracking approaches, the new STC model encodes the correlations between object trajectories and object classes and accounts for unexpected object trajectories, which makes it appropriate for multi-class dynamic envi-

ronments. As opposed to most traditional object-classification approaches, the STC model performs trajectory estimation and accounts for data-association ambiguities.

### 7.1.2 The EA Algorithm

This thesis introduced the Expectation Association (EA) algorithm, an approximate inference procedure that allows us to estimate object states and classes simultaneously and efficiently in a sequential manner.

Our system outputs state estimates for all of the objects in the scene and soft assignments of each object to different motion categories or classes. Object classes are estimated even from noisy, incomplete and ambiguous measurements of position and appearance. Our method utilises classic and efficient statistical estimation techniques such as the Kalman filtering and smoothing recursions [70] as subroutines.

### 7.1.3 Efficient Multi-modal Data Association

The approach introduced in this thesis, seamlessly integrate multi-modal features such as appearance and dynamics to solve the data-association problem. Our method allows the user to utilise the technique of their choice in order to initialise the association probabilities, and then updates these probabilities based on the dynamic-state history. This approach to data association allows for identity recovery after occlusion and merging situations. Furthermore, the approach is general enough so any sensor modality can potentially be used.

### 7.1.4 Automatic Parameter Estimation

ML estimation of model parameters is known to be prone to over-fitting. This thesis has introduced a MAP-EM procedure for learning the parameters of the STC model. The introduction of regularisation makes learning more robust, and more importantly, it allows us to safely increase the number of features for describing tracked objects.

### 7.1.5 Unexpected Trajectory Handling

Both the EA algorithm for tracking and classification, and the MAP-EM approach to learning the model parameters, are robust against outlier trajectories. On one hand, the EA algorithm detects unexpected objects during inference. On the other hand, the MAP-EM approach to parameter learning down-weights the contribution of trajectories that have a significant offset to the bulk of the data. Therefore the STC model represents the most common dynamics in the environment even in the presence of outliers.

### 7.1.6 Experiments

The performance of the proposed approach was compared with state-of-the-art approaches and validated using the publicly available KITTI dataset. This dataset provides ground-truth information, which facilitates the evaluation. The results showed that our online approach is more robust to noise than the baseline approaches and achieves state-of-the-art performance even against batch tracking approaches that try to estimate a global solution using the entire observation sequence.

## 7.2 Future Research Directions

Although this thesis has furthered the state-of-the-art in multi-object tracking by highlighting the importance of simultaneous quantitative and qualitative descriptions, the problem of robustly describing moving objects is far from solved. This section points out some limitations of the presented framework, discusses how they can be improved, and presents further research directions.

### 7.2.1 Integrated Object Detection

The current framework totally decouples object detection from object tracking. At each time step the detection module searches the entire field of view of the sensor for

candidate detections. Further research directions include informing the object detection module with the current estimated trajectories, thus adding context information to the detection module.

With very few exceptions ([147, 148] for single-object and [149] for multi-object tracking), most approaches to tracking-by-detection treat detection and tracking as independent modules. Combined object detection, tracking and classification in the multi-object scenario is a very challenging problem, particularly, due to the fact that, more robust and discriminative models are needed. In fact, online learning of robust appearance models for unknown objects is an open problem in the visual-tracking community [150, 151].

### 7.2.2 Object Interaction Modelling

One of the fundamental assumptions that our modelling approach makes is that, given the class assignments of the tracked objects, their motion is independent of each other's. A very promising extension of the model introduced in this thesis, would be one where object interactions such as grouping [65, 48] are accounted for.

Reasoning about object interactions would not only provide robotic platforms with increased situational awareness, but could also be used to increase the robustness of tracking systems to occlusions and sensor failure. The basic intuition in the case of the grouping interaction is that objects moving as a group tend to follow the same dynamics. This information could be used for example to improve the localisation of occluded objects.

Object-interaction reasoning can be achieved by adding to the STC model a random variable whose probability density is parametrised as a function of the object states, and whose scope is given by the possible interactions. The main challenges that would be raised by this extension are: engineering the features for representing these interactions, and ensuring that inference stays tractable.

### 7.2.3 Class Switching

The EA algorithm assumes that object classes remain constant. In the current form of the algorithm, if an object changes its dynamic behaviour (for example, a pedestrian who starts riding his bicycle), the assignment probabilities tend to become uniform across the classes involved in the transition and will slowly skew towards the new class, as the observations supporting the previous assignment leave the estimation window. In order to quickly account for class switching, change-point detection should be included into the framework. Although adding class switching to our model would combinatorially increase the complexity, a variational methodology such as the one presented by [152] could be used to derive an efficient inference procedure. The same kind of methodology could be used to include model parameters as random variables to be inferred, so that objects whose class dynamic model evolves may be accounted for.

## 7.3 Concluding Remarks

This thesis has formulated a novel solution to the simultaneous tracking- and classification problem as performing inference on a PGM, and has presented the structure of the model, an approximate variational procedure for this inference task, and a method for automatically learning the parameters of the model.

By providing both a quantitative and qualitative description of the dynamic environment through the use of simultaneous multi-object tracking and classification, not only the situational awareness of robotic platforms improves, but also the capabilities of higher-level tasks such as path planning, whose performance can be improved with the use of richer contextual information.

# Bibliography


[1] Ernst Dieter Dickmanns. The development of machine vision for road vehicles in the last decade. In *Intelligent Vehicles Symposium (IV)*, pages 268–281, 2002.

[2] Martin Buehler, Karl Lagnemma, and Sanjiv Singh. *The 2005 DARPA Grand Challenge: The Grate Robot Race*. Springer t edition, 2007.

[3] John Markoff. Google cars drive themselves, in traffic, 2010.

[4] Mica R. Endsley and Debra G. Jones. *Designing for Situation Awareness*. 2012.

[5] Cesar Cadena and Jana Kosecka. Recursive Inference for Prediction of Objects in Urban Environments. In *International Symposium on Robotics Research (ISRR)*, pages 1–16, 2013.

[6] M. Zeeshan Zia, Michael Stark, and Konrad Schindler. Towards Scene Understanding with Detailed 3D Object Representations. *International Journal of Computer Vision (IJCV)*, 112:188 – 203, November 2014.

[7] Julius Ziegler, Philipp Bender, Markus Schreiber, Henning Lategahn, Tobias Strauss, Christoph Stiller, Uwe Franke, Nils Appenrodt, Christoph G. Keller, Eberhard Kaus, Ralf G. Herrtwich, Clemens Rabe, David Pfeiffer, Frank Lindner, Fridtjof Stein, Friedrich Erbs, Markus Enzweiler, Carsten Knoppel, Jochen Hipp, Martin Haueis, Maximilian Trepte, Carsten Brenk, Andreas Tamke, Mohammad Ghanaat, Markus Braun, Armin Joos, Hans Fritz, Horst Mock, Martin Hein, and Eberhard Zeeb. Making Bertha Drive - An Autonomous Journey on a Historic Route. *IEEE Intelligent Transportation Systems Magazine (ITSM)*, 6(2):8–20, 2014.

[8] Volvo. Volvo Unveils Cyclist Detection System, 2013.

[9] Sayanan Sivaraman and Mohan Manubhai Trivedi. Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis. *IEEE Transactions on Intelligent Transportation Systems (TITS)*, 14(4):1773–1795, December 2013.

[10] James P Underwood, Andrew Hill, Thierry Peynot, and Steven J Scheding. Error Modeling and Calibration of Exteroceptive Sensors. *Journal of Field Robotics (JFR)*, 27(1):2–20, 2010.

[11] Zachary Taylor, Juan Nieto, and David Johnson. Multi-Modal Sensor Calibration Using a Gradient Orientation Measure. *Journal of Field Robotics (JFR)*, 32(5):675 – 695, 2014.

[12] Davide Scaramuzza and Friedrich Fraundorfer. Visual Odometry. *IEEE Robotics and Automation Magazine*, (December):80–92, 2011.

[13] S. Williams, V. Indelman, M. Kaess, R. Roberts, J. J. Leonard, and F. Dellaert. Concurrent filtering and smoothing: A parallel architecture for real-time navigation and full smoothing. *The International Journal of Robotics Research (IJRR)*, 33(12):1544–1568, July 2014.

[14] Daphne Koller and Nir Friedman. *Probabilistic Graphical Models: Principles and Techniques*. 2009.

[15] Christopher M. Bishop. *Pattern Recognition and Machine Learning*. 2006.

[16] Karl Granstrom, Christian Lundquist, and Umut Orguner. Tracking rectangular and elliptical extended targets using laser measurements. In *IEEE International Conference on Information Fusion (FUSION)*, 2011.

[17] X. RONG LI and VESSELIN P. JILKOV. Survey of Maneuvering Target Tracking . Part I : Dynamic Models. *IEEE Transactions on Aerospace and Electronic Systems (TAES)*, 39(4):1333–1363, 2003.

[18] Chuanhai Liu. ML Estimation of the Multivariate t Distribution and the EM Algorithm. *Journal of Multivariate Analysis (JMA)*, 63(1):296–312, 1997.

[19] Michael Roth. On the Multivariate t Distribution. Technical report, Linkopings University, 2013.

[20] Michael Feldmann, D Franken, and Wolfgang Koch. Tracking of extended objects and group targets using random matrices. *IEEE Transactions on Signal Processing (TSP)*, 59(4):1409–1420, 2011.

[21] Julian F P Kooij, Gwenn Englebienne, and Dariu M Gavrila. A Non-parametric Hierarchical Model to Discover Behavior Dynamics from Tracks. In *European Conference on Computer Vision (ECCV)*, pages 270–283, 2012.

[22] Shijun Sun, Chenglin Peng, Wensheng Hou, Jun Zheng, Yingtao Jiang, and Xiaolin Zheng. Blind source separation with time series variational Bayes expectation maximization algorithm. *Digital Signal Processing (DSP)*, 22(1): 17–33, 2012.

[23] Mathilde Bouriga and O Féron. Estimation of covariance matrices based on hierarchical inverse-Wishart priors. *Journal of Statistical Planning and Inference (JSPI)*, 143(1):795 – 808, 2012.

[24] Colin J. Champion. Empirical Bayesian estimation of normal variances and covariances. *Journal of Multivariate Analysis (JMA)*, 87(1):60–79, October 2003.

[25] Justin Dauwels, Andrew Eckford, Sascha Korl, and Hans-andrea Loeliger. Expectation Maximization as Message Passing - Part I: Principles and Gaussian Messages. *arXiv*, pages 1–14, 2009.

[26] Xiao Li Hu, Thomas B. Schön, and Lennart Ljung. A basic convergence result for particle filtering. *IEEE Transactions on Signal Processing (TSP)*, 7(4): 288–293, 2007.

[27] Victor Romero-Cano, Gabriel Agamennoni, and Juan Nieto. A Variational Approach to Simultaneous Tracking and Classification of Multiple Objects. In *International Conference on Information Fusion (FUSION)*, pages 1 – 8, 2014.

[28] Victor Romero-cano, Gabriel Agamennoni, and Juan Nieto. A Variational Approach to Simultaneous Multi-Object Tracking and Classification. *International Journal of Robotics Research (IJRR)*, 1(1):1–18, 2015.

[29] Christian a Nasseth, Fredrik Lindsten, and Thomas B Schön. Sequential Monte Carlo methods for graphical models. *Advances in Neural Information Processing Systems (NIPS)*, 1(1):1 – 6, 2014.

[30] James Hensman, M Rattray, and ND Lawrence. Fast Variational Inference in the Conjugate Exponential Family. In *Advances in Neural Information Processing (NIPS)*, pages 1–9, 2012.

[31] Fergal Casey, Joshua Waterfall, Ryan Gutenkunst, Christopher Myers, and James Sethna. Variational method for estimating the rate of convergence of Markov-chain Monte Carlo algorithms. *Physical Review E (PR)*, 78(046704):1 – 12, October 2008.

[32] Sang Min Oh, James M. Rehg, Tucker Balch, and Frank Dellaert. Learning and Inferring Motion Patterns using Parametric Segmental Switching Linear Dynamic Systems. *International Journal of Computer Vision (IJCV)*, 77(1-3): 103–124, July 2007.

[33] Christophe Andrieu, Arnaud Doucet, and Roman Holenstein. Particle Markov chain Monte Carlo methods. *Journal of the Royal Statistical Society. Series B: Statistical Methodology (JRSS)*, 72(3):269–342, 2010.

[34] Matthew J Beal and Zoubin Ghahramani. The Variational Bayesian EM Algorithm for Incomplete Data : with Application to Scoring Graphical Model Structures. In *Bayesian Statistics (BS)*, pages 1–10. 2003.

[35] David Barber. *Bayesian Reasoning and Machine Learning*. 2012.

[36] Michael I Jordan. An Introduction to Variational Methods for Graphical Models. *Machine Learning (ML)*, 37(1):183–233, 1999.

[37] A. P. Dempster;, N. M. Laird;, and D. B. Rubin. Maximum Likelihood from Incomplete Data via the EM Algorithm. *Journal of the Royal Statistical Society. Series B: Statistical Methodology (JRSS)*, 39(1):1–38, 1977.

[38] Antoni B Chan and Nuno Vasconcelos. Modeling, clustering, and segmenting video with mixtures of dynamic textures. *IEEE transactions on pattern analysis and machine intelligence (TPAMI)*, 30(5):909–26, May 2008.

[39] MY Park and T Hastie. L1 regularization path algorithm for generalized linear models. *Journal of the Royal Statistical Society (JRSS)*, 69(4):659–677, 2007.

[40] Douglas L. Vail, John D. Lafferty, and Manuela M. Veloso. Feature selection in conditional random fields for activity recognition. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 3379–3384, 2007.

[41] Chris Fraley and Adrian E. Raftery. Bayesian Regularization for Normal Mixture Estimation and Model-Based Clustering. *Journal of Classification (JC)*, 24(1):155–181, 2007.

[42] Andrew Gelman, Aleks Jakulin, Maria Grazia Pittau, and Yu-Sung Su. A weakly informative default prior distribution for logistic and other regression models. *The Annals of Applied Statistics (AAS)*, 2(4):1360–1383, December 2008.

[43] Keni Bernardin and Rainer Stiefelhagen. Evaluating Multiple Object Tracking Performance: The CLEAR MOT Metrics. *EURASIP Journal on Image and Video Processing (JIVP)*, 2008(1):1–10, 2008.

[44] Yuan Li, Chang Huang, and Ram Nevatia. Learning to Associate : HybridBoosted Multi-Target Tracker for Crowded Scene. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 2953–2960, 2009.

[45] Andreas Geiger, Martin Lauer, and Raquel Urtasun. A generative model for 3D urban scene understanding from movable platforms. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1945–1952, Colorado Springs, June 2011. Ieee.

[46] Andreas Geiger, Martin Lauer, Christian Wojek, Christoph Stiller, and Raquel Urtasun. 3D Traffic Scene Understanding from Movable Platforms. *IEEE transactions on pattern analysis and machine intelligence*, pages 1–14, September 2013.

[47] Geoffrey E Hinton. Training products of experts by minimizing contrastive divergence. *Neural computation*, 14:1771–1800, 2002.

[48] Tobias Baumgartner, Dennis Mitzel, and Bastian Leibe. Tracking People and Their Objects. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2013.

[49] Paul Besl and Neil McKay. A Method for Registration of 3-D Shapes. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 14: 239–256, 1992.

[50] N. Kaempchen, B. Schiele, and K. Dietmayer. Situation Assessment of an Autonomous Emergency Brake for Arbitrary Vehicle-to-Vehicle Collision Scenarios. *IEEE Transactions on Intelligent Transportation Systems*, 10(4): 678–687, December 2009.

[51] Oliver Frank, Juan Nieto, Jose Guivant, and Steve Scheding. Multiple Target Tracking using Sequential Monte Carlo Methods and Statistical Data Association. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, volume 00, pages 2718–2723, 2003.

[52] Chieh-Chih. Wang, Charles. Thorpe, Sebastian. Thrun, Martial. Hebert, and H. Durrant-Whyte. Simultaneous Localization, Mapping and Moving Object Tracking. *The International Journal of Robotics Research*, 26(9):889–916, September 2007.

[53] Yang Gu and M. Veloso. Effective Multi-Model Motion Tracking using Action Models. *The International Journal of Robotics Research*, 28(1):3–19, January 2009.

[54] Frank Moosmann and Christoph Stiller. Joint Self-Localization and Tracking of Generic Objects in 3D Range Data. In *IEEE International Conference on Robotics and Automation*, pages 1138–1144, 2013.

[55] Wongun Choi, Caroline Pantofaru, and Silvio Savarese. A General Framework for Tracking Multiple People from a Moving Camera. *IEEE transactions on pattern analysis and machine intelligence*, 35(7):1577–91, July 2013.

[56] M. Bajracharya, B. Moghaddam, A. Howard, S. Brennan, and L. H. Matthies. A Fast Stereo-based System for Detecting and Tracking Pedestrians from a Moving Vehicle. *The International Journal of Robotics Research*, 28(11-12): 1466–1485, July 2009.

[57] Victor Romero-Cano and Juan I Nieto. Stereo-based Motion Detection and Tracking from a Moving Platform. In *Intelligent Vehicles Symposium*, pages 499–504, 2013.

[58] Caglayan Dicle, Octavia I Camps, and Mario Sznaier. The Way They Move : Tracking Multiple Targets with Similar Appearance. In *IEEE International Conference on Computer Vision*, pages 2304–2311, 2013.

[59] Chang Huang, Yuan Li, and Ramakant Nevatia. Multiple target tracking by learning-based hierarchical association of detection responses. *IEEE transactions on pattern analysis and machine intelligence*, 35(4):898–910, April 2013.

[60] Brad Schumitsch, Sebastian Thrun, Leonidas Guibas, and Kunle Olukotun. The Identity Management Kalman Filter ( IMKF ). In *Robotics: Science and Systems*, 2006.

[61] Hirofumi Kanazaki, Takehisa Yairi, Kazuo Machida, Kenji Kondo, and Yoshihiko Matsukawa. Variational Bayes Data Association Filter. *3rd International Conference on Intelligent Sensors, Sensor Networks and Information*, pages 401–406, 2007.

[62] Anton Milan, Stefan Roth, and Konrad Schindler. Continuous Energy Minimization for Multi-Target Tracking. *IEEE transactions on pattern analysis and machine intelligence*, 36(1):1–15, May 2013.

[63] Ernesto Brau, Jinyan Guan, Kyle Simek, Luca Del Pero, Colin Reimer Dawson, and Kobus Barnard. Bayesian 3D tracking from monocular video. In *International Conference on Computer Vision*, pages 3368–3375, 2013.

[64] Karl Granström, Christian Lundquist, Fredrik Gustafsson, and Umut Orguner. Random Set Methods: Estimation of Multiple Extended Objects. *IEEE Robotics & Automation Magazine*, (June):73–82, 2014.

[65] Wongun Choi and Silvio Savarese. A Unified Framework for Multi-Target Tracking and Collective Activity Recognition. In *ECCV*, number i, 2012.

[66] Stephan Reuter, Benjamin Wilking, Jurgen Wiest, Michael Munz, and Klaus Dietmayer. Real-Time Multi-Object Tracking using Random Finite Sets. *IEEE Transactions on Aerospace and Electronic Systems*, 49(4):2666–2678, October 2013.

[67] R.P.S. Mahler. Multitarget bayes filtering via first-order multitarget moments. *IEEE Transactions on Aerospace and Electronic Systems*, 39(4):1152–1178, October 2003.

[68] B.-N. Vo and W.-K. Ma. The Gaussian Mixture Probability Hypothesis Density Filter. *IEEE Transactions on Signal Processing*, 54(11):4091–4104, November 2006.

[69] DB Reid. An algorithm for tracking multiple targets. *Automatic Control, IEEE Transactions on*, (6), 1979.

[70] H. E. Rauch, C. T. Striebel, and F. Tung. Maximum likelihood estimates of linear dynamic systems. *Journal of American Institute of Aeronautics and Astronautics*, 3(8):1445–1450, August 1965.

[71] Wolfgang Koch. On Bayesian Tracking and Data Fusion : A Tutorial Introduction with Examples. 25(7):29–52, 2010.

[72] A Sajana Rahmathullah, Lennart Svensson, and Daniel Svensson. Merging-based forward-backward smoothing on Gaussian mixtures. *International Conference on Information Fusion*, 2014.

[73] Yaakov Bar-Shalom. Tracking in a Cluttered Environment With Probabilistic Data Association *. *Automatica*, 11:451–460, 1975.

[74] Thomas E. Fortmann, Yaakov Bar-Shalom, and Molly Scheffe. Sonar Tracking of Multiple Targets Using Joint Probabilistic Data Association. *Oceanic Engineering,*, 8(3), 1983.

[75] C. Rasmussen and G.D. Hager. Probabilistic data association methods for tracking complex visual objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 23(6):560–576, June 2001.

[76] Roy L Streit and Tod E Luginbuhl. Probabilistic Multi-Hypothesis Tracking. Technical Report February, 1995.

[77] David F. Crouse, Marco Guerreiro, and Peter Willett. A Critical Look at the PMHT. *Journal of Advances in Information Fusion*, 4(2), 2009.

[78] C Bibby and I Reid. Simultaneous localisation and mapping in dynamic environments (SLAMIDE) with reversible data association. *Proceedings of Robotics: Science and Systems*, 2007.

[79] Ben Benfold and Ian Reid. Stable multi-target tracking in real-time surveillance video. *Cvpr 2011*, pages 3457–3464, June 2011.

[80] Jason L. Williams and Roslyn A. Lau. Data association by loopy belief propagation. *International Conference on Information Information Fusion*, 2010.

[81] Jason L. Williams and Roslyn a. Lau. Convergence of loopy belief propagation for data association. *2010 Sixth International Conference on Intelligent Sensors, Sensor Networks and Information Processing*, pages 175–180, December 2010.

[82] Jason L. Williams and Roslyn A. Lau. Approximate evaluation of marginal association probabilities with belief propagation. *arXiv preprint arXiv:1209.6299*, pages 1–14, 2012.

[83] Aleksandr V Segal and Ian Reid. Latent Data Association : Bayesian Model Selection for Multi-target Tracking. In *IEEE International Conference on Computer Vision*, pages 2904–2911, 2013.

[84] Brendan J Frey and Nebojsa Jojic. A comparison of algorithms for inference and learning in probabilistic graphical models. *IEEE transactions on pattern analysis and machine intelligence*, 27(9):1392–416, September 2005.

[85] X. Rong Li and Vesselin P Jilkov. Survey of Maneuvering Target Tracking . Part II : Motion Models of Ballistic and Space Targets. *IEEE Transactions on Aerospace and Electronic Systems*, 46(1), 2010.

[86] X. Rong Li and Vesselin P Jilkov. Survey of Maneuvering Target Tracking. Part V : Multiple-Model Methods. *IEEE Transactions on Aerospace and Electronic Systems*, 41(4), 2005.

[87] E. Mazor, A. Averbuch, Y. Bar-Shalom, and J. Dayan. Interacting Multiple Model Methods in Target Tracking : A Survey. *IEEE Transactions on Aerospace and Electronic Systems*, 34(1):103–123, 1998.

[88] Antoni B. Chan and Nuno Vasconcelos. Mixtures of dynamic textures. In *IEEE International Conference on Computer Vision*, pages 641–647 V. Ieee, 2005.

[89] B.T. Morris and M.M. Trivedi. A Survey of Vision-Based Trajectory Learning and Analysis for Surveillance. *IEEE Transactions on Circuits and Systems for Video Technology*, 18(8):1114–1127, August 2008.

[90] Roman Katz, Juan Nieto, and Eduardo Nebot. Unsupervised Classification of Dynamic Obstacles in Urban Environments. *Journal of Field Robotics*, 27(4): 450–472, 2010.

[91] Alex Teichman and Sebastian Thrun. Group induction. *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 2757–2763, November 2013.

[92] Faisal I Bashir, Ashfaq a Khokhar, and Dan Schonfeld. Object trajectory-based activity classification and recognition using hidden Markov models. *IEEE Transactions on Image Processing*, 16(7):1912–9, July 2007.

[93] Brendan Tran Morris and Mohan Manubhai Trivedi. Trajectory Learning for Activity Understanding : Unsupervised, Multilevel, and Long-Term Adaptive Approach. *IEEE transactions on Pattern Analysis and Machine Intelligence*, 33(11):2287–2301, 2011.

[94] Darío García-García, Emilio Parrado-Hernández, and Fernando Diaz-de Maria. State-space dynamics distance for clustering sequential data. *Pattern Recognition*, 44(5):1014–1022, May 2011.

[95] BJ Frey and Delbert Dueck. Clustering by passing messages between data points. *Science*, (February), 2007.

[96] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A Survey. *ACM Computing Surveys*, 41(3):1–58, July 2009.

[97] Manish Gupta, Jing Gao, Charu Aggarwal, and Jiawei Han. Outlier Detection for Temporal Data. In *Syntheis Lectures on Data Mining and Knowledge Discovery*. 2014.

[98] Marco A. F. Pimentel, David A. Clifton, Lei Clifton, and Lionel Tarassenko. A Review of Novelty Detection. *Signal Processing*, pages 1–64, January 2014.

[99] Yi Fang and Myong K Jeong. Robust Probabilistic Multivariate Calibration Model. *Technometrics*, 50(3):305–316, August 2008.

[100] Michael Tipping and Christopher Bishop. Probabilistic Principal Component Analysis. *Journal of the Royal Statistical Society: Series B*, 61(3):611–622, 1999.

[101] Tao Chen, Elaine Martin, and Gary Montague. Robust probabilistic PCA with missing data and contribution analysis for outlier detection. *Computational Statistics & Data Analysis*, 53(10):3706–3716, August 2009.

[102] Gabriel Agamennoni, Juan I Nieto, Eduardo M Nebot, and Senior Member. *Approximate Inference in State-Space Models With Heavy-Tailed Noise*, volume 60. 2012.

[103] X. Rong Li. Joint tracking and classification based on bayes joint decision and estimation. *10th International Conference on Information Fusion*, pages 1–8, 2007.

[104] X Rong Li. Optimal Bayes Joint Decision and Estimation. In *International Conference on Information Fusion*, volume 3950, 2007.

[105] Gabriel Agamennoni, Juan I Nieto, and Eduardo M Nebot. Estimation of Multivehicle Dynamics by Considering Contextual Information. *IEEE Transactions on Robotics*, 28(4):855–870, 2012.

[106] Syed Ahmed Pasha, Ba-Ngu Vo, Hoang Duong Tuan, and Wing-Kin Ma. A Gaussian Mixture PHD Filter for Jump Markov System Models. *IEEE Transactions on Aerospace and Electronic Systems*, 45(3):919–936, 2009.

[107] Daniel Meissner, Stephan Reuter, Elias Strigel, and Klaus Dietmayer. Intersection-Based Road User Tracking Using a Classifying Multiple-Model PHD Filter. *IEEE Intelligent Transportation Systems Magazine*, 6(April 2014):21–33, 2014.

[108] G. W. Pulford and B. F. La Scala. Multihypothesis Viterbi data association: algorithm development and assessment. *IEEE Transactions on Aerospace and Electronic Systems*, 46(2):583–609, 2010.

[109] D. Vasquez, T. Fraichard, and C. Laugier. Growing Hidden Markov Models: An Incremental Tool for Learning and Predicting Human and Vehicle Motion. *The International Journal of Robotics Research*, 28(11-12):1486–1506, August 2009.

[110] A. Ess, K. Schindler, B. Leibe, and L. Van Gool. Object Detection and Tracking for Autonomous Navigation in Dynamic Environments. *The International Journal of Robotics Research*, 29(14):1707–1725, May 2010.

[111] D. M. Gavrila and S. Munder. Multi-cue Pedestrian Detection and Tracking from a Moving Vehicle. *International Journal of Computer Vision*, 73(1): 41–59, July 2006.

[112] Christian Wojek, Stefan Walk, Stefan Roth, Konrad Schindler, and Bernt Schiele. Monocular Visual Scene Understanding: Understanding Multi-Object Traffic Scenes. *IEEE transactions on pattern analysis and machine intelligence*, August 2012.

[113] Long Leo Zhu, Yuanhao Chen, Yuan Lin, Chenxi Lin, and Alan Yuille. Recursive segmentation and recognition templates for image parsing. *IEEE transactions on pattern analysis and machine intelligence*, 34(2):359–71, February 2012.

[114] Santiago Manen, Matthieu Guillaumin, and Luc Van Gool. Prime Object Proposals with Randomized Prim ' s Algorithm. In *International Conference on Computer Vision*, pages 4321–4328, 2013.

[115] Xiaoyu Wang, M Yang, S Zhu, and Yuanqing Lin. Regionlets for generic object detection. *International Conference on Computer Vision*, 2013.

[116] Pedro F. Felzenszwalb, Ross B. Girshick, David McAllester, and Deva Ramanan. Object detection with discriminatively trained part-based models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–20, 2010.

[117] David Gerónimo, Antonio M López, Angel D Sappa, and Thorsten Graf. Survey of pedestrian detection for advanced driver assistance systems. *IEEE Transactions on Pattern Analysis and Machine Intelligence (TPAMI)*, 32(7): 1239–58, July 2010.

[118] Piotr Dollar, Ron Appel, Serge Belongie, and Pietro Perona. Fast Feature Pyramids for Object Detection. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8828(c):1–1, 2014.

[119] Joop van de Ven, Fabio Ramos, and Gian Diego Tipaldi. An integrated probabilistic model for scan-matching, moving object detection and motion estimation. In *2010 IEEE International Conference on Robotics and Automation*, pages 887–894. Ieee, May 2010.

[120] Christoph Mertz, Luis E. Navarro-Serment, Robert MacLachlan, and Chuck Thorpe. Moving object detection with laser scanners. *Journal of Field Robotics*, 30(2004):17–43, 2013.

[121] B Douillard and J Underwood. On the segmentation of 3D LIDAR point clouds. *International Conference on Robotics and Automation*, 2011.

[122] H. W. Kuhn. The Hungarian Method for the Assignment Problem. *Naval Research Logistics Quarterly*, 2(1-2):83–97, 1955.

[123] Claude L Fennema and William B Thompson. Velocity determination in scenes containing several moving objects. *Computer Graphics and Image Processing*, 9(4):301–315, April 1979.

[124] Berthold K.P. Horn and Brian G. Schunck. Determining optical flow. *Artificial Intelligence*, 17(1-3):185–203, August 1981.

[125] Bruce D Lucas and Takeo Kanade. An Iterative Image Registration Technique with an Application to Stereo Vision. In *Proceedings of Imaging Understanding Workshop*, volume 130, pages 121–130, 1981.

[126] Shireen Y Elhabian, Khaled M El-sayed, and Sumaya H Ahmed. Moving Object Detection in Spatial Domain using Background Removal Techniques - State-of-Art. *Computer*, (2):32–54, 2008.

[127] Steffen Gauglitz, Tobias Höllerer, and Matthew Turk. Evaluation of Interest Point Detectors and Feature Descriptors for Visual Tracking. *International Journal of Computer Vision*, 94(3):335–360, March 2011.

[128] Simon Baker, Daniel Scharstein, J. P. Lewis, Stefan Roth, Michael J. Black, and Richard Szeliski. A Database and Evaluation Methodology for Optical Flow. *International Journal of Computer Vision*, 92(1):1–31, November 2010.

[129] Norbert Buch, Sergio a. Velastin, and James Orwell. A Review of Computer Vision Techniques for the Analysis of Urban Traffic. *IEEE Transactions on Intelligent Transportation Systems (TITS)*, 12(3):920–939, September 2011.

[130] Rahul Kumar Namdev, Abhijit Kundu, K Madhava Krishna, and C V Jawahar. Motion Segmentation of Multiple Objects from a Freely Moving Monocular Camera. In *IEEE International Conference on Robotics and Automation*, 2012.

[131] Ashit Talukder and Larry Matthies. Real-time Detection of Moving Objects from Moving Vehicles using Dense Stereo and Optical Flow. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 3718–3725, Sendai, Japan, 2004.

[132] Hernan Badino and Takeo Kanade. A Head-Wearable Short-Baseline Stereo System for the Simultaneous Estimation of Structure and Motion. In *IAPR Conference on Machine Vision Aplications (MVA)*, 2011.

[133] Frank R Kschischang, Senior Member, Brendan J Frey, and Hans-andrea Loeliger. Factor Graphs and the Sum-Product Algorithm. *IEEE Transactions on Information Theory*, 47(2):498–519, 2001.

[134] Sotirios P Chatzis, Dimitrios I Kosmopoulos, and Theodora a Varvarigou. Robust sequential data modeling using an outlier tolerant hidden Markov model. *IEEE transactions on pattern analysis and machine intelligence*, 31(9): 1657–69, September 2009.

[135] Hao Zhu, Henry Leung, and Zhongshi He. A variational Bayesian approach to robust sensor fusion based on Student-t distribution. *Information Sciences*, 221:201–214, February 2013.

[136] Peter Morton, Bertrand Douillard, and James Underwood. Multi-sensor identity tracking with event graphs. *IEEE International Conference on Robotics and Automation*, pages 4742–4748, May 2013.

[137] Karl Granström, Christian Lundquist, and Omut Orguner. Extended Target Tracking using a Gaussian-Mixture PHD Filter. *IEEE Transactions on Aerospace and Electronic Systems*, 48(4):3268–3286, 2012.

[138] Ronald Mahler. PHD filters for nonstandard targets, I: Extended targets. In *IEEE International Conference on Information Fusion*, pages 448–452, 2009.

[139] Sabine Van Huffel, Haesun Park, and J Ben Rosen. Formulation an Solution of Structured Total Least Norm Problems for Parameter Estimation. *IEEE Transactions on Signal Processing*, 44(10), 1996.

[140] Anton Milan, Konrad Schindler, and Stefan Roth. Detection- and Trajectory-Level Exclusion in Multiple Object Tracking. *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3682–3689, June 2013.

[141] a. P. Dawid. Some Matrix-Variate Distribution Theory: Notational Considerations and a Bayesian Application. *Biometrika*, 68(1):265, April 1981.

[142] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets Robotics: The KITTI Dataset. *The International Journal of Robotics Research*, August 2013.

[143] Hernan Badino. Least Squares Estimation of a Plane Surface in Disparity Image Space. Technical report, 2011.

[144] Geoffrey E Hinton, Simon Osindero, and Yee-Whye Teh. A Fast Learning Algorithm for Deep Belief Nets. *Neural Computation*, 18:1527–1554, 2006.

[145] Lazaros Zafeiriou, Mihalis A Nicolaou, Stefanos Zafeiriou, Symeon Nikitidis, and Maja Pantic. Learning Slow Features for Behaviour Analysis. In *International Conference on Computer Vision*, 2013.

[146] Ruslan Salakhutdinov, Joshua B Tenenbaum, and Antonio Torralba. Learning with Hierarchical-Deep Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 35(8):1958–1971, 2013.

[147] Boris Babenko, Ming-Hsuan Yang, and Serge Belongie. Visual Tracking with Online Multiple Instance Learning. *IEEE transactions on pattern analysis and machine intelligence*, December 2010.

[148] Zdenek Kalal, Krystian Mikolajczyk, and Jiri Matas. Tracking-Learning-Detection. *IEEE transactions on pattern analysis and machine intelligence*, 34(7):1409–1422, December 2012.

[149] Bastian Leibe, Konrad Schindler, Nico Cornelis, and Luc Van Gool. Coupled object detection and tracking from static cameras and moving vehicles. *IEEE transactions on pattern analysis and machine intelligence*, 30(10):1683–98, October 2008.

[150] Cheng-Hao Kuo, Chang Huang, and Ramakant Nevatia. Multi-target tracking by on-line learned discriminative appearance models. In *IEEE International Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 685–692. Ieee, June 2010.

[151] Arnold W M Smeulders, Dung M. Chu, Rita Cucchiara, Simone Calderara, Afshin Dehghan, and Mubarak Shah. Visual tracking: An experimental survey. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 36 (7):1442–1468, 2014.

[152] Gabriel Agamennoni, Stewart Worrall, James R. Ward, and Eduardo M. Nebot. Automated extraction of driver behaviour primitives using Bayesian agglomerative sequence segmentation. In *IEEE International Conference on Intelligent Transportation Systems*, pages 1449–1455, 2014.

# Appendix A

# Complete Derivation of the E-Step

This appendix provides a complete derivation of the E-step of our EA algorithm. Let us write down the log-likelihood of an LDS with its covariance matrices weighted by $\omega^{i,j}$. An LDS parametrised by $\mathbf{A_{i,j}}, \mathbf{Q_{i,j}}/\omega^{i,j}, \mathbf{H_{i,j}}, \mathbf{R_{i,j}}/\omega^{i,j}$ has a log-density function given by:

$$\begin{aligned}\ln \mathrm{LDS}\left(x_i;\cdot\right) = &-\frac{1}{2}\left(x_0^i-\mu_j\right)^T V_j^{-1}\left(x_0^i-\mu_j\right)-\frac{1}{2}\ln|V_j| \\ &-\sum_{t=2}^{T_i}\left(\frac{\omega^{i,j}}{2}\left(x_t^i-\mathbf{F_{i,j}}x_{t-1}^i\right)^T\mathbf{Q_{i,j}}^{-1}\left(x_t^i-\mathbf{F_{i,j}}x_{t-1}^i\right)\right)-\frac{T_i-1}{2}\ln|\mathbf{Q_{i,j}}|+\frac{m\left(T_i-1\right)}{2}\ln\omega^{i,j} \\ &-\sum_{t=1}^{T_i}\left(\frac{\omega^{i,j}}{2}\left(\mathbf{z_t^i}-\mathbf{H_{i,j}}x_t^i\right)^T\mathbf{R_{i,j}}^{-1}\left(\mathbf{z_t^i}-\mathbf{H_{i,j}}x_t^i\right)\right)-\frac{T_i}{2}\ln|\mathbf{R_{i,j}}|+\frac{nT_i}{2}\ln\omega^{i,j} \\ &-\frac{\left(m+n\right)T_i}{2}\ln\left(2\pi\right).\end{aligned} \tag{A.1}$$

Given the conditional independence properties of our model, we can further rewrite $q\left(s_i,x_i,\omega_i\right)$ as $q\left(s_i\right)q\left(\omega_i\right)q\left(x_i|s_i,\omega_i\right)$. Moreover, since the posterior of a Linear Dynamic System (LDS) can be efficiently estimated using the Kalman filtering and smoothing equations, we want to demonstrate that:

$$\begin{aligned}\ln q\left(s_i,x_i,\omega_i\right) &= \ln q\left(s_i\right)+\ln q\left(\omega^{i,j}|s_i\right)+\ln q\left(x_i|s_i=j,\omega^{i,j}\right)=\left\langle\ln p\left(s,x,z,a,\omega\right)\right\rangle_{q_{k\neq i}(s_k,x_k,\omega_k),q(a)} \\ &= \ln q\left(s_i\right)+\ln q\left(\omega_i|s_i\right)+\ln LDS\left(x_i,\mathbf{z_i};\mathbf{F_{i,j}},\mathbf{Q_{i,j}}/\omega^{i,j},\mathbf{H_{ji}},\mathbf{R_{i,j}}/\omega^{i,j}\right)+\eta_i.\end{aligned} \tag{A.2}$$

## A.1 Object i's State: $q\left(x_i|s_i=j,\omega^{i,j}\right)$

Lets us write down the expectation in Equation A.2 (Expectations are underlined):

$$
\begin{aligned}
\ln q\left(s_i,x_i,\omega_i\right) = &\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\frac{\upsilon}{2}\ln\frac{\upsilon}{2}+\left(\frac{\upsilon}{2}-1\right)\ln\omega^{i,j}-\frac{\upsilon}{2}\omega^{i,j}-\ln\Gamma\left(\frac{\upsilon}{2}\right)+\ln p_j+\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\underline{\underline{q(a_t^l=i)}}p_i^l\right)\\
&+\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\underbrace{-\frac{\omega^{i,j}}{2}\left(x_1^{i,j}-\mu_j\right)^T V_j^{-1}\left(x_1^{i,j}-\mu_j\right)-\frac{1}{2}\ln|V_j|}_{f_i(x;\,s)}\right)\\
&+\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\underbrace{-\sum_{t=2}^{T_i}\left(\frac{\omega^{i,j}}{2}\left(x_t^{i,j}-F_j x_{t-1}^{i,j}\right)^T Q_j^{-1}\left(x_t^{i,j}-F_j x_{t-1}^{i,j}\right)\right)-\frac{T_i-1}{2}\ln|Q_j|+\frac{mT_i}{2}\left(\ln\omega^{i,j}-\ln\left(2\pi\right)\right)}_{f_p(x;\,s,\,w)}\right)\\
&+\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\underbrace{\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\underline{\underline{q(a_t^l=i)}}\left(-\frac{\omega^{i,j}}{2}\left(z_t^l-H_j x_t^{i,j}\right)^T R_j^{-1}\left(z_t^l-H_j x_t^{i,j}\right)\right)}_{f_o(x;\,s,\,w)}\right)\\
&+\boxed{\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\underline{\underline{q(a_t^l=i)}}\left(-\frac{1}{2}\ln|R_j|+\frac{n}{2}\ln\omega^{i,j}-\frac{n}{2}\ln\left(2\pi\right)\right)\right)}.
\end{aligned}
\tag{A.3}
$$

It is evident that $f_i(x;s)$ and $f_p(x;s,w)$ in Equation A.3, are equivalent to the portion of the log-likelihood of an LDS corresponding to both the initial conditions and the process model (first and second line of Equation A.1). For this reason, we only need to modify $f_o(x;s,w)$ in order to demonstrate the equality in the last line of Equation A.2. We distribute the sum over associations for object $i$ and complete the square as follows:

$$
\begin{aligned}
&f_o(x;s,w) =\\
&=-\sum_{t=1}^{T_i}\frac{\omega^{i,j}}{2}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l\,T}R_j^{-1}z_t^l\right)-\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l\,T}R_j^{-1}H_j x_t^{i,j}\right)-\sum_{l=1}^{L_t}\alpha_t^l\left(x_t^{i,j\,T}H_j^T R_j^{-1}z_t^l\right)+\sum_{l=1}^{L_t}\alpha_t^l\left(x_t^{i,j\,T}H_j^T R_j^{-1}H_j x_t^{i,j}\right)\right)\\
&=-\sum_{t=1}^{T_i}\frac{\omega^{i,j}}{2}\left(\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^{l\,T}}{\alpha_t}\alpha_t R_j^{-1}\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t}-\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^{l\,T}}{\alpha_t}\alpha_t R_j^{-1}H_j x_t^{i,j}-x_t^{i,j\,T}H_j^T\alpha_t R_j^{-1}\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t}+x_t^{i,j\,T}H_j^T\alpha_t R_j^{-1}H_j x_t^{i,j}\right)\\
&\quad-\frac{nT_i}{2}\ln\left(2\pi\right)-\frac{T_i}{2}\ln|\frac{1}{\alpha_t}R_j|+\frac{nT_i}{2}\ln\omega^{i,j}\\
&\quad-\boxed{\sum_{t=1}^{T_i}\frac{\omega^{i,j}}{2}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l\,T}R_j^{-1}z_t^l\right)\right)}-\boxed{\left(\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^{l\,T}}{\alpha_t}\alpha_t R_j^{-1}\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t}\right)+\frac{nT_i}{2}\ln\left(2\pi\right)+\frac{T_i}{2}\ln|\frac{1}{\alpha_t}R_j|-\frac{nT_i}{2}\ln\omega^{i,j}}
\end{aligned}
\tag{A.4}
$$

Where $\alpha_t^l = q(a_t^l = i)$ and $\alpha_t = \sum_{l=1}^{L_t} \alpha_t^l$. Terms inside the blue box were added and subtracted in order to complete the square, whereas terms inside black boxes, in Equations A.3 and A.4, are constant terms that belong to the original complete-data log-likelihood but were left out in order to complete the square.

By comparing $f_o(x;s,w)$ with Equation A.1, we show that $f_i(x;s)$, $f_p(x;s,w)$ and $f_o(x;s,w)$ together correspond (up to a constant factor) to the PDF of an LDS parametrised by the following parameters:

$$\begin{aligned}
\mathbf{z_t^i} &= \frac{\sum_{l=1}^{L_t} \alpha_t^l z_t^l}{\alpha_t} \\
\mathbf{F_{i,j}} &= F_j \\
\frac{\mathbf{Q_{i,j}}}{\omega^{i,j}} &= \frac{Q_j}{\omega^{i,j}} \\
\mathbf{H_{ji}} &= H_j \\
\frac{\mathbf{R_{i,j}}}{\omega^{i,j}} &= \frac{1}{\alpha_t \omega^{i,j}} R_j
\end{aligned} \tag{A.5}$$

As a result, we can write the variational factor $\ln q\left(s_i, x_i, \omega_i\right)$ as follows:

$$\begin{aligned}
\ln q\left(s_i,x_i,\omega_i\right) = &\sum_j^{N_s} \delta\left(s_i,j\right)\left(\frac{\upsilon}{2}\ln\frac{\upsilon}{2} + \left(\frac{\upsilon}{2}-1\right)\ln\omega_i - \frac{\upsilon}{2}\omega^i - \ln\Gamma\left(\frac{\upsilon}{2}\right) + \ln p_j + \sum_{t=1}^{T_i}\sum_{l=1}^{L_t} \underline{\underline{q(a_t^l=i)}} p_i^l\right) \\
&+ \underbrace{\sum_j^{N_s} \delta\left(s^i,j\right)\left(\mathrm{LDS}\left(x^i, \frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t}; F_j, \frac{Q_j}{\omega^i}, H_j, \frac{1}{\alpha_t\omega^i}R_j\right)\right)}_{\ln q\left(x^i|s^i,\omega^i\right)} + \eta_i,
\end{aligned} \tag{A.6}$$

where

$$\begin{aligned}
\eta_i = \sum_j^{N_s} \delta\left(s^i,j\right)\Bigg(&\sum_{t=1}^{T_i}\sum_{l=1}^{L_t} \alpha_t^l\left(-\frac{1}{2}\ln|R_j| + \frac{n}{2}\ln\omega^i - \frac{n}{2}\ln\left(2\pi\right)\right) \\
&- \sum_{t=1}^{T_i}\frac{\omega^i}{2}\left(\sum_{l=1}^{L_t}\alpha_t^l\left({z_t^l}^T R_j^{-1} z_t^l\right) - \frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^{l\,T}}{\alpha_t}\alpha_t R_j^{-1}\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t}\right) \\
&+ \frac{nT_i}{2}\ln\left(2\pi\right) + \frac{T_i}{2}\ln|\frac{1}{\alpha_t}R_j| - \frac{nT_i}{2}\ln\omega^i\Bigg)
\end{aligned} \tag{A.7}$$

From Equation A.6 we can obtain the PDF of $x^i$ and $\omega^i$ for target class $s^i = j$:

$$\begin{aligned}\ln q\left(s^i=j,x^i,\omega^i\right) =& \frac{\upsilon}{2}\ln\frac{\upsilon}{2}+\left(\frac{\upsilon}{2}-1\right)\ln\omega^i-\frac{\upsilon}{2}\omega^i-\ln\Gamma\left(\frac{\upsilon}{2}\right)+\ln p_j+\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l p_i^l \\ &\underbrace{+\,\text{LDS}\left(x^i,\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t};F_j,\frac{Q_j}{\omega^i},H_j,\frac{1}{\alpha_t\omega^i}R_j\right)}_{\ln q\left(x^i|s^i=j,\omega^i\right)}+\eta_i(s^i=j),\end{aligned} \tag{A.8}$$

## A.2 Object i's Precision Weight: $q\left(\omega^i|s^i\right)$

In order to obtain the posterior over $\omega^i$ we need to marginalise out $x^i$. We start by rewriting A.8. For that, we group all the terms that depend on $\omega^i$ together and define the constant $\eta_{i,j}$:

$$\begin{aligned}\ln q\left(s^i=j,x^i,\omega^i\right) =& \frac{\upsilon}{2}\ln\frac{\upsilon}{2}+\left(\frac{\upsilon}{2}-1-\frac{nT_i}{2}+\frac{n}{2}\sum_{t=1}^{T_i}\alpha_t\right)\ln\omega^i \\ &-\left(\frac{\upsilon}{2}+\frac{1}{2}\sum_{t=1}^{T_i}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l\,T}R_j^{-1}z_t^l\right)-\mathbf{z_t^i}^T\alpha_t R_j^{-1}\mathbf{z_t^i}\right)\right)\omega^i-\ln\Gamma\left(\frac{\upsilon}{2}\right) \\ &+\ln p_j+\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l p_i^l \\ &\underbrace{+\,\text{LDS}\left(x^i,\frac{\sum_{l=1}^{L_t}\alpha_t^l z_t^l}{\alpha_t};F_j,\frac{Q_j}{\omega^i},H_j,\frac{1}{\alpha_t\omega^i}R_j\right)}_{\ln q\left(x^i|s^i=j,\omega^i\right)}+\eta_{i,j},\end{aligned} \tag{A.9}$$

where:

$$\eta_{i,j}=\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l\left(-\frac{1}{2}\ln|R_j|-\frac{n}{2}\ln(2\pi)\right)+\frac{nT_i}{2}\ln(2\pi)+\frac{T_i}{2}\ln|\frac{1}{\alpha_t}R_j| \tag{A.10}$$

In order to obtain the posterior over $\omega^i$, we marginalise out $x^i$:

$$\begin{aligned}q\left(s^i=j,\omega^i\right)=\int_{x^i}\exp\left(\ln q\left(s^i=j,x^i,\omega^i\right)\right)dx^i \propto& f(\omega^i)\int_{x^i}q\left(x^i|s^i=j,\omega^i\right)dx^i \\ \propto& f(\omega^i)\prod_{t=1}^{T_i}l_t^i, \\ \propto& f(\omega^i)\prod_{t=1}^{T_i}\mathcal{N}\left(\hat{\mathbf{y}}_\mathbf{t}^\mathbf{i}|H_j\hat{x}_{t|t-1}^{i,j},\frac{H_j\hat{V}_{t|t-1}^{i,j}H_j^T+R_j}{\omega^i}\right)\end{aligned} \tag{A.11}$$

where $l_t^i = p\left(\mathbf{z_t^i}|\mathbf{z_{1:t-1}^i}\right)$, which is the marginal likelihood of the state at time $t$, is obtained as a sub-product of the filtering routine (the innovation likelihood). We can

then write down $\ln q(s^i = j, \omega^i)$ as:

$$\begin{aligned}\ln q(s^i = j, \omega^i) =& \frac{\upsilon}{2}\ln\frac{\upsilon}{2} + \left(\frac{\upsilon}{2} - 1 - \frac{nT_i}{2} + \frac{n}{2}\sum_{t=1}^{T_i}\alpha_t\right)\ln\omega^i \\ &- \left(\frac{\upsilon}{2} + \frac{1}{2}\sum_{t=1}^{T_i}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l\,T}R_j^{-1}z_t^l\right) - \mathbf{z_t^i}^T\alpha_t R_j^{-1}\mathbf{z_t^i}\right)\right)\omega^i - \ln\Gamma\left(\frac{\upsilon}{2}\right) \\ &+ \ln p_j + \sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l p_i^l \\ &+ \sum_{t=1}^{T_i}\ln\mathcal{N}\left(\mathbf{\hat{y}_t^i}|H_j\hat{x}_{t|t-1}^{i,j}, \frac{H_j\hat{V}_{t|t-1}^{i,j}H_j^T + R_j}{\omega^i}\right) + \eta_{i,j},\end{aligned} \tag{A.12}$$

Next we expand the innovation log-likelihood for object $i$ in order to obtain the remaining terms that are a function of $\omega^i$ as follows:

$$\ln\mathcal{N}\left(\mathbf{\hat{y}_t^i}|H_j\hat{x}_{t|t-1}^i, j, \frac{\Sigma_t^{i,j}}{\omega^i}\right) = -\frac{\omega^i}{2}\epsilon_t^{i,j\,T}\Sigma_t^{i,j}\epsilon_t^{i,j} - \frac{1}{2}\ln|\Sigma_t^{i,j}| - \frac{n}{2}\ln(2\pi) + \frac{n}{2}\ln\omega^i \tag{A.13}$$

where $\Sigma_t^{i,j} = H_j\hat{V}_{t|t-1}^{i,j}H_j^T + R_j$. Next we distribute the terms in the innovation so that Equation A.12 becomes:

$$\begin{aligned}\ln q(s^i = j, \omega^i) =& \frac{\upsilon}{2}\ln\frac{\upsilon}{2} + \left(\frac{\upsilon}{2} - 1 + \frac{n}{2}\sum_{t=1}^{T^i}\alpha_t\right)\ln\omega^i \\ &- \left(\frac{\upsilon}{2} + \frac{1}{2}\sum_{t=1}^{T_i}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l\,T}R_j^{-1}z_t^l\right) - \mathbf{z_t^i}^T\alpha_t R_j^{-1}\mathbf{z_t^i} + \epsilon_t^{i,j\,T}\Sigma_t^{i,j}\epsilon_t^{i,j}\right)\right)\omega^i \\ &- \ln\Gamma\left(\frac{\upsilon}{2}\right) \\ &- \frac{1}{2}\sum_{t=1}^{T_i}\ln|\Sigma_t^{i,j}| + \ln p_j + \sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l p_i^l + \eta_{i,j},\end{aligned} \tag{A.14}$$

where:

$$\eta_{i,j} = \sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l\left(-\frac{1}{2}\ln|R_j| - \frac{n}{2}\ln(2\pi)\right) + \frac{T_i}{2}\ln|\frac{1}{\alpha_t}R_j|. \tag{A.15}$$

Note that two terms in Equation A.14 are a function of $w^i$ and $\ln w^i$. That indicates that the approximate posterior over $w^i$ is conditionally Gamma with parameters given

by:

$$\boldsymbol{\alpha_{i,j}} = \frac{\upsilon}{2} + \frac{n}{2}\sum_{t=1}^{T_i}\alpha_t, \tag{A.16}$$

$$\boldsymbol{\beta_{i,j}} = \frac{\upsilon}{2} + \frac{1}{2}\sum_{t=1}^{T_i}\left(\sum_{l=1}^{L_t}\left(\alpha_t^l z_t^{l^T} R_j^{-1} z_t^l\right) - \mathbf{z_t^i}^T \alpha_t R_j^{-1} \mathbf{z_t^i} + \epsilon_t^{i,j^T} \Sigma_t^{i,j} \epsilon_t^{i,j}\right) \tag{A.17}$$

## A.3 Object i's Posterior Assignment Probability $q\left(s^i = j\right)$

Finally, we marginalise out $\omega^i$. If we define the auxiliary variables as:

$$\begin{aligned}
a &= \frac{\upsilon}{2}\ln\frac{\upsilon}{2} - \ln\Gamma\left(\frac{\upsilon}{2}\right) - \frac{1}{2}\sum_{t=1}^{T_i}\ln|\Sigma_t^{i,j}| + \ln p_j + \sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\alpha_t^l p_i^l + \eta_{i,j};\\
b &= \frac{\upsilon}{2} - 1 + \frac{n}{2}\sum_{t=1}^{T_i}\alpha_t;\\
c &= \frac{\upsilon}{2} + \frac{1}{2}\sum_{t=1}^{T_i}\left(\sum_{l=1}^{L_t}\alpha_t^l\left(z_t^{l^T} R_j^{-1} z_t^l\right) - \mathbf{z_t^i}^T \alpha_t R_j^{-1}\mathbf{z_t^i} + \epsilon_t^{i,j^T}\Sigma_t^{i,j}\epsilon_t^{i,j}\right);
\end{aligned}$$

we obtain:

$$\begin{aligned}
\ln q(s^i = j) &= \ln\int_0^\infty \exp\left(\ln q\left(s^i = j, \omega^i\right)\right) d\omega^i\\
&= a + \ln b + \ln\Gamma(b) - (b+1)\ln c
\end{aligned} \tag{A.18}$$

# Appendix B

# Complete Derivation of the Association Step (A-Step)

This Appendix presents the derivation of the association factors $q_t^{l,i}(a_t^{l,i})$

$$\ln q\left(a_t^{l,i}\right) = \left\langle \ln p\left(s, x, \omega, z, a\right)\right\rangle_{q(s,x,\omega),q_{k\neq t}^{m\neq l,n\neq i}(a_i^{l,i})} \tag{B.1}$$

After taking the terms that are a function of $a_t^{l,i}$ from the complete log-likelihood in Equation 3.7, the association factors are given by:

$$\begin{aligned}\ln q\left(a_t^{l,i}\right) \propto \sum_{j=1}^{N_s} q\left(s^{i,j}\right) \Bigg( & \ln p\left(a_t^{l,i}\right) \\ & - \frac{1}{2}\Big( \Big\langle \omega_i \left(z_t^l - H^j x_t^{i,j}\right)^T R^{j^{-1}} \left(z_t^l - H^j x_t^{i,j}\right) - n \ln \omega_i \Big\rangle_{q_i(s_i,x_i,w_i)} \\ & + \ln |R_j| + n \ln(2\pi)\Big)\Bigg)\end{aligned} \tag{B.2}$$

$$\begin{aligned}\ln q\left(a_t^{l,i}\right) \propto \sum_{j=1}^{N_s} q\left(s^{i,j}\right) \Bigg( & \ln p\left(a_t^{l,i}\right) \\ & - \frac{\hat{\omega}^{ij}}{2}\left(\left(z_t^l - H^j \hat{x}_t^{i,j}\right)^T R^{j^{-1}} \left(z_t^l - H^j \hat{x}_t^{i,j}\right) + Tr\left(H^{j^T} R^{j^{-1}} H^j Cov\left(\hat{x}_t^{i,j}\right)\right)\right) \\ & + \frac{n}{2}\left(\psi(\alpha_{ij}) - \ln(\beta_{ij})\right) \\ & - \frac{1}{2}\left(\ln|R_j| + n\ln(2\pi)\right)\Bigg)\end{aligned} \tag{B.3}$$

# Appendix C

# Complete Derivation of the Model Parameter Update Equations

## C.1 The Complete-Data Log-Likelihood

$$p\left(s,x,z,a,\omega|\mathbf{\Omega}\right)=\prod_{i=1}^{N_x}\left[p\left(s_i\right)p\left(\omega_i|s_i\right)p\left(x_0^i|s_i\right)\prod_{t=1}^{T_i}p\left(x_t^i|x_{t-1}^i,s_i,\omega^i\right)\right]\\ \prod_{t=1}^{T}\prod_{l=1}^{L_t}p\left(z_t^l|x_t^{1:N_x},a_t^l,\omega^{1:N_x}\right)p\left(a_t^l\right) \tag{C.1}$$

The model parameters are given by $\mathbf{\Omega}=[F_{1:N_s},Q_{1:N_s},H_{1:N_s},R_{1:N_s}]$. In order to constrain the parameter space, we regularise the learning of the model parameters by imposing a prior distribution over them. This prior is shared by all the mixture components of the model.

$$p\left(\mathbf{\Omega}\right)=p\left(F|Q\right)p\left(Q\right)p\left(H|R\right)p\left(R\right) \tag{C.2}$$

### C.1.1 Prior Over Hidden Variables

$$
\begin{aligned}
p\left(s_i\right) &= \prod_{j=1}^{N_s} p_j^{\delta(s_i,j)}, \quad p_j = \left[p\left(1\right), ..., p\left(N_s\right)\right], \\
p\left(a_t^l\right) = \left[p\left(a_t^{l,1}\right), \cdots, p\left(a_t^{l,N_x}\right)\right], \quad p\left(a_t^{l,i}\right) &= \prod_{i=1}^{N_x} p_i^{l\,\delta\left(a_t^{l,i},i\right)}, \quad p_i^l = \left[p\left(a_t^{l,1}\right), ..., p\left(a_t^{l,N_x}\right)\right], \\
p\left(\omega_i|s_i\right) &= \mathcal{G}\left(\omega^i;\alpha,\beta\right) \\
p\left(x_0^i\middle|s^i=j\right) &= \mathcal{N}\left(x_0^{ij};\mu_j,V_{ij}\right) \\
p\left(x_t^i\middle|x_{t-1}^i,s^i=j,\omega^i\right) &= \mathcal{N}\left(x_t^{ij};F_j x_{t-1}^{ij},Q_j/\omega^i\right) \\
p\left(z_t^l\middle|x_t^{1:N_x},a_t^l=i,\omega^i\right) &= \mathcal{N}\left(z_t^l;Hx_t^{ij},R_j/\omega^i\right).
\end{aligned}
$$

PDFs for the the priors over the hidden variables:

$$
\begin{aligned}
\mathcal{G}\left(\omega_i;\alpha,\beta\right) &= \frac{1}{\Gamma\left(\alpha\right)}\beta^\alpha \omega_i^{\alpha-1}\exp(-\beta\omega_i), \quad \alpha=\beta=\frac{\upsilon}{2} \\
\mathcal{N}\left(x_0^{ij};\mu_j,V_{ij}\right) &= \left(2\pi\right)^{-m/2}\left|V_{ij}\right|^{-1/2}\exp\left(-\frac{1}{2}\left(x_0^{ij}-\mu_j\right)^T V_{ij}^{-1}\left(x_0^{ij}-\mu_j\right)\right) \\
\mathcal{N}\left(x_t^{ij};F_j x_{t-1}^{ij},Q_j/\omega_i\right) &= \left(2\pi\right)^{-m/2}\left|Q_j\right|^{-1/2}\omega_i^{m/2}\exp\left(-\frac{\omega_i}{2}\left(x_t^{ij}-F_j x_{t-1}^{ij}\right)^T Q_j^{-1}\left(x_t^{ij}-F_j x_{t-1}^{ij}\right)\right) \\
\mathcal{N}\left(z_t^l;H_j x_t^{ij},R_j/\omega_i\right) &= \left(2\pi\right)^{-n/2}\left|R_j\right|^{-1/2}\omega_i^{n/2}\exp\left(-\frac{\omega_i}{2}\left(z_t^l-H_j x_t^{ij}\right)^T R_j^{-1}\left(z_t^l-H_j x_t^{ij}\right)\right).
\end{aligned}
$$

### C.1.2 Prior Over Parameters

For the regularisation term of the state transition matrix F and the process noise covariance Q, we utilise the *Matrix Variate Normal* [141] and the *inverse Wishart* distributions respectively.

$$
p\left(F|Q\right)p\left(Q\right) = \mathcal{N}_{m\times m}\left(F|\Lambda,Q,\Omega\right)\mathcal{W}^{-1}\left(Q|\nu\Sigma,\nu\right). \tag{C.3}
$$

The PDFs of the factors to the right of the equal sign in Eq. C.3 are given by:

$$\mathcal{N}(F|\Lambda,Q,\Omega)=\frac{|\Omega|^{\frac{m}{2}}}{(2\pi)^{\frac{m^2}{2}}|Q|^{\frac{m}{2}}}etr\left[-\frac{1}{2}(F-\Lambda)^TQ^{-1}(F-\Lambda)\Omega\right] \tag{C.4}$$

where $etr(X)\rightarrow\exp(tr(X))$. $\Lambda$ and $\Omega$ are the mean and among-column covariance respectively. We set the among-row covariance to be equal to the process noise covariance $Q$.

$$\mathcal{W}^{-1}\left(Q|\nu\Sigma,\nu\right)=\frac{\frac{\nu}{2}^{\frac{m\nu}{2}}|\Sigma|^{\frac{\nu}{2}}}{\pi^{\frac{m(m-1)}{4}}\prod_{k=1}^{m}\Gamma\left(\frac{\nu+1-k}{2}\right)}|Q|^{-\frac{\nu+m+1}{2}}etr\left[-\frac{\nu}{2}\Sigma Q^{-1}\right]. \tag{C.5}$$

$\nu$ and $\Sigma$ are the degrees of freedom and the scale of the inverse Wishart prior respectively. The prior over the pair $H,R$ is defined similarly. Therefore, the prior over parameters is given by:

$$p\left(\mathbf{\Omega}\right)=\mathcal{N}_{m\times m}\left(F|\Lambda_f,Q,\Omega_f\right)\mathcal{W}^{-1}\left(Q|\nu_q\Sigma_q,\nu_q\right)\mathcal{N}_{n\times m}\left(H|\Lambda_h,R,\Omega_h\right)\mathcal{W}^{-1}\left(R|\nu_r\Sigma_r,\nu_r\right) \tag{C.6}$$

### C.1.3 The Complete-Data Likelihood Function

$$p\left(s,x,z,a,\omega|\mathbf{\Omega}\right)=\prod_{i=1}^{N_x}\left[\prod_{j=1}^{N_s}\left[p_j\mathcal{G}\left(\omega^i;\alpha,\beta\right)\mathcal{N}\left(x_0^{ij};\mu_j,V_{ij}\right)\prod_{t=1}^{T_i}\mathcal{N}\left(x_t^{ij};F_jx_{t-1}^{ij},Q_j/\omega^i\right)\right.\right.$$
$$\left.\left.\prod_{t=1}^{T}\prod_{l=1}^{L_t}\left[p_i^l\mathcal{N}\left(z_t^l;Hx_t^{ij},R_j/\omega^i\right)\right]^{\delta\left(a_t^l,i\right)}\right]^{\delta\left(s^i,j\right)}\right]$$

We proceed to write down the complete-data log-likelihood of our model:

$$\begin{aligned}
\ln p\left(s,x,z,a,w|\mathbf{\Omega}\right) = & \sum_{i=1}^{N_x}\Bigg[\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\frac{\upsilon}{2}\ln\frac{\upsilon}{2}+\left(\frac{\upsilon}{2}-1\right)\ln\omega_i-\frac{\upsilon}{2}\omega_i-\ln\Gamma\left(\frac{\upsilon}{2}\right)+\ln p_j+\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\ln p_i^l\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(-\frac{1}{2}\left(x_0^{ij}-\mu_j\right)^T V_j^{-1}\left(x_0^{ij}-\mu_j\right)-\frac{1}{2}\ln|V_{ij}|\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(-\sum_{t=2}^{T_i}\left(\frac{\omega_i}{2}\left(x_t^{ij}-F_j x_{t-1}^{ij}\right)^T Q_j^{-1}\left(x_t^{ij}-F_j x_{t-1}^{ij}\right)\right)-\frac{T_i-1}{2}\ln|Q_j|+\frac{m\left(T_i-1\right)}{2}\ln\omega_i\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\left(-\frac{\omega_i}{2}\left(z_t^l-H_j x_t^{ij}\right)^T R_j^{-1}\left(z_t^l-H_j x_t^{ij}\right)-\frac{1}{2}\ln|R_j|+\frac{n}{2}\ln\omega_i\right)\right)\\
& -\sum_{j}^{N_s}\delta\left(s_i,j\right)\frac{T_i\left(m\right)}{2}\ln\left(2\pi\right)-\sum_{j}^{N_s}\delta\left(s_i,j\right)\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\frac{n}{2}\ln\left(2\pi\right)\Bigg].
\end{aligned} \tag{C.7}$$

Using the trace trick

$$u^T\Sigma^{-1}v = trace\left(u^T\Sigma^{-1}v\right) = trace\left(\Sigma^{-1}vu^{T\text{`}}\right), \tag{C.8}$$

we rewrite the complete-data log-likelihood as follows:

$$\begin{aligned}
\ln p\left(s,x,z,a,w|\mathbf{\Omega}\right) = & \sum_{i=1}^{N_x}\Bigg[\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\frac{\upsilon}{2}\ln\frac{\upsilon}{2}+\left(\frac{\upsilon}{2}-1\right)\ln\omega_i-\frac{\upsilon}{2}\omega_i-\ln\Gamma\left(\frac{\upsilon}{2}\right)+\ln p_j+\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\ln p_i^l\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(-\frac{1}{2}\mathrm{tr}\left(V_j^{-1}\left(x_0^{ij}x_0^{ij^T}-x_0^{ij}\mu_j^T-\mu_j x_0^{ij^T}+\mu_j\mu_j^T\right)\right)-\frac{1}{2}\ln|V_j|\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(-\sum_{t=2}^{T_i}\frac{\omega_i}{2}\mathrm{tr}\left(Q_j^{-1}\left(x_t^{ij}x_t^{ij^T}-x_t^{ij}{x_{t-1}^{ij}}^T F_j^T-F_j x_{t-1}^{ij}x_t^{ij^T}+F_j x_{t-1}^{ij}{x_{t-1}^{ij}}^T F_j^T\right)\right)\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(-\frac{T_i-1}{2}\ln|Q_j|+\frac{m\left(T_i-1\right)}{2}\ln\omega_i\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\left(-\frac{\omega_i}{2}\mathrm{tr}\left(R_j^{-1}\left(z_t^l z_t^{l^T}-z_t^l x_t^{ij^T}H_j^T-H_j x_t^{ij}z_t^{l^T}+H_j x_t^{ij}x_t^{ij^T}H_j^T\right)\right)\right)\right)\\
& +\sum_{j}^{N_s}\delta\left(s_i,j\right)\left(\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\left(-\frac{1}{2}\ln|R_j|+\frac{n}{2}\ln\omega_i\right)\right)\\
& -\sum_{j}^{N_s}\delta\left(s_i,j\right)\frac{T_i\left(m\right)}{2}\ln\left(2\pi\right)-\sum_{j}^{N_s}\delta\left(s_i,j\right)\sum_{t=1}^{T_i}\sum_{l=1}^{L_t}\delta(a_t^l,i)\frac{n}{2}\ln\left(2\pi\right)\Bigg].
\end{aligned} \tag{C.9}$$

We obtain the expected complete-data log-likelihood of our model by calculating the expectation of Equation C.9 under the posterior and keeping only those terms that

are a function of the model parameters:

$$\begin{aligned}
\mathcal{Q}_{ML}(\mathbf{\Omega}) &= \langle \ln p\,(s,x,z,w|a,\mathbf{\Omega})\rangle_{p(s,x,w|a,;\hat{\mathbf{\Omega}})} \\
&= \sum_{i=1}^{N_x}\left[\sum_{j=1}^{N_s}\hat{p}_{ij}\,(\ln p_j)\right] \\
&- \sum_{i=1}^{N_x}\left[\sum_{j=1}^{N_s}\hat{p}_{ij}\left(\frac{1}{2}tr\left(V_j^{-1}\left(\hat{P}_1^{ij}-\hat{x}_1^{ij}\mu_j^T-\mu_j(\hat{x}_1^{ij})^T+\mu_j\mu_j^T\right)\right)-\frac{1}{2}\ln|V_j|\right)\right] \\
&- \sum_{i=1}^{N_x}\left[\sum_{j=1}^{N_s}\hat{p}_{ij}\left(\sum_{t=2}^{T_i}\frac{\hat{w}_i}{2}tr\left(Q_j^{-1}\left(\hat{P}_t^{ij}-\hat{P}_{t,t-1}^{ij}F_j^T-F_j(\hat{P}_{t,t-1}^{ij})^T+F_j\hat{P}_{t-1}^{ij}F_j^T\right)\right)+\frac{T_i-1}{2}\ln|Q_j|\right)\right] \\
&- \sum_{i=1}^{N_x}\left[\sum_{j=1}^{N_s}\hat{p}_{ij}\left(\sum_{t=1}^{T_i}\frac{\hat{w}_i}{2}tr\left(R_j^{-1}\left(z_t^iz_t^{iT}-z_t^i(\hat{x}_t^{i,j})^TH_j^T-H_j\hat{x}_t^{i,j}z_t^{iT}+H_j\hat{P}_t^{ij}H_j^T\right)\right)+\frac{T_i}{2}\ln|R_j|\right)\right],
\end{aligned} \tag{C.10}$$

In a more compact form:

$$\begin{aligned}
\mathcal{Q}_{ML}(\mathbf{\Omega}) = &\sum_{j=1}^{N_s}\hat{N}_j\ln p_j \\
&-\frac{1}{2}\sum_{j=1}^{Ns}tr\left[V_j^{-1}\left(\eta_j-\xi_j\mu_j^T-\mu_j\xi_j^T+\hat{N}_j\mu_j\mu_j^T\right)\right]-\frac{1}{2}\sum_{j=1}^{Ns}\hat{N}_j\ln|V_j| \\
&-\frac{1}{2}\sum_{j=1}^{Ns}tr\left[Q_j^{-1}\left(\varphi_j-\psi_jF_j^T-F_j\psi_j^T+F_j\phi_jF_j^T\right)\right]-\frac{T_i-1}{2}\sum_{j=1}^{Ns}\hat{N}_j\ln|Q_j| \\
&-\frac{1}{2}\sum_{j=1}^{Ns}tr\left[R_j^{-1}\left(\Lambda_j-\Gamma_jH_j^T-H_j\Gamma_j^T+H_j\Phi_jH_j^T\right)\right]-\frac{T_i}{2}\sum_{j=1}^{Ns}\hat{N}_j\ln|R_j|
\end{aligned} \tag{C.11}$$

$$\begin{aligned}
\hat{N}_j &= \sum_{i=1}^{N_x}\hat{p}_{ij} & \phi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t-1}^{ij} \\
\eta_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{P}_1^{ij} & \Lambda_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}z_t^iz_t^{iT} \\
\zeta_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{x}_1^{ij} & \Gamma_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}z_t^i(\hat{x}_t^{i,j})^T \\
\varphi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_t^{ij} & \Phi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_t^{ij}, \\
\psi_j &= \sum_{i=1}^{N_x}\hat{p}_{ij}\hat{w}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t,t-1}^{ij}
\end{aligned} \tag{C.12}$$

where

$$\hat{P}_t^{ij} = \hat{V}_t^{ij} + \hat{x}_t^{ij}(\hat{x}_t^{ij})^T \tag{C.13}$$

$$\hat{P}_{t,t-1}^{ij} = \hat{V}_{t,t-1}^{ij} + \hat{x}_t^{ij}(\hat{x}_{t-1}^{ij})^T. \tag{C.14}$$

Note that for learning purposes, we assume known data association, which is provided by the KITTI tracking dataset. We obtain the objective function for each parameter by extracting the terms in the M-step objective function that are a function of the parameter to be optimised. Note that terms are extracted from Equation C.10 and Equation C.15. The regularisation terms, which are contributed by the prior over parameters, were highlighted.

## C.2 Regularised Model Parameters Learning

We learn the parameters of our model from training data. We derive a MAP-estimation procedure. Substituting the PDFs in Equation C.6 and applying the natural logarithm, the log-prior can be expressed as:

$$\begin{aligned}
\ln p\left(\boldsymbol{\Omega}\right) = & \frac{m}{2}\ln|\Omega_f| - \frac{m^2}{2}\ln\left(2\pi\right) - \frac{m}{2}\ln|Q| - \frac{1}{2}tr\left[(F-\Lambda_f)^T Q^{-1}(F-\Lambda_f)\Omega_f\right] \\
& + \frac{m\nu_q}{2}\ln\frac{\nu_q}{2} + \frac{\nu_q}{2}\ln|\Sigma_q| - \frac{m(m-1)}{4}\ln\pi + \sum_{k=1}^{m}\ln\Gamma\left(\frac{\nu_q+1-k}{2}\right) - \frac{\nu_q+m+1}{2}\ln|Q| + tr\left[-\frac{\nu_q}{2}\Sigma_q Q^{-1}\right] \\
& + \frac{n}{2}\ln|\Omega_h| - \frac{nm}{2}\ln\left(2\pi\right) - \frac{m}{2}\ln|R| - \frac{1}{2}tr\left[(H-\Lambda_h)^T R^{-1}(H-\Lambda_h)\Omega_h\right] \\
& + \frac{n\nu_r}{2}\ln\frac{\nu_r}{2} + \frac{\nu_r}{2}\ln|\Sigma_r| - \frac{n(n-1)}{4}\ln\pi + \sum_{k=1}^{n}\ln\Gamma\left(\frac{\nu_r+1-k}{2}\right) - \frac{\nu_r+n+1}{2}\ln|R| + tr\left[-\frac{\nu_r}{2}\Sigma_r R^{-1}\right].
\end{aligned} \tag{C.15}$$

The M-step objective function $\mathcal{Q}$ for estimating the posterior mode is given by expected complete-data log-likelihood, which for our MAP case is given by:

$$\mathcal{Q}(\boldsymbol{\Omega}) = \mathcal{Q}_{ML}(\boldsymbol{\Omega}) + \ln p\left(\boldsymbol{\Omega}\right). \tag{C.16}$$

## C.3 Transition Matrix F

The objective function[2] for the transition Matrix $F$ is given by:

$$\begin{aligned}\mathcal{Q}\left(F_j\right) =& -\frac{1}{2}tr\Bigg(Q_j^{-1}\Big(-\sum_{i=1}^{N_x}\hat{w}_i\hat{p}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t,t-1}^{ij}F_j^T - F_j\sum_{i=1}^{N_x}\hat{w}_i\hat{p}_{ij}\sum_{t=2}^{T_i}(\hat{P}_{t,t-1}^{ij})^T + F_j\sum_{i=1}^{N_x}\hat{w}_i\hat{p}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t-1}^{ij}F_j^T \\ &+ \underline{\left(F_j-\Lambda_f\right)\Omega_f\left(F_j-\Lambda_f\right)^T}\Big)\Bigg) \\ =& -\frac{1}{2}tr\Bigg(Q_j^{-1}\Big(-\left(\sum_{i=1}^{N_x}\hat{w}_i\hat{p}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t,t-1}^{ij} + \underline{\Lambda_f\Omega_f}\right)F_j^T - F_j\left(\sum_{i=1}^{N_x}\hat{w}_i\hat{p}_{ij}\sum_{t=2}^{T_i}(\hat{P}_{t,t-1}^{ij})^T + \underline{\Omega_f\Lambda_f^T}\right) \\ &+ F_j\left(\sum_{i=1}^{N_x}\hat{w}_i\hat{p}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t-1}^{ij} + \underline{\Omega_f}\right)F_j^T\Big)\Bigg).\end{aligned} \tag{C.17}$$

Maximising Equation C.17:

$$\begin{aligned}\hat{F}_j =& \underset{F_j}{\operatorname{argmax}}\,\mathcal{Q}\left(F_j\right) \\ =& \Big(\underline{\Lambda_f\Omega_f} + \sum_{i=1}^{N_x}\hat{w}_{ij}\hat{p}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t,t-1}^{ij}\Big)\Big/\Big(\underline{\Omega_f} + \sum_{i=1}^{N_x}\hat{w}_{ij}\hat{p}_{ij}\sum_{t=2}^{T_i}\hat{P}_{t-1}^{ij}\Big) \\ =& \Big(\underline{\Lambda_f\Omega_f} + \psi_j\Big)\Big/\Big(\underline{\Omega_f} + \phi_j\Big).\end{aligned} \tag{C.18}$$

## C.4 Process Noise Covariance Q

The objective function for the process noise Covariance $Q$ is obtained in a similar way:

$$\begin{aligned}\mathcal{Q}\left(Q_j\right) =& -\frac{1}{2}tr\Bigg(Q_j^{-1}\Bigg(\Bigg(\sum_{i=1}^{N_x}\hat{w}_{ij}\hat{p}_{ij}\sum_{t=2}^{T_i}\Big(\hat{P}_t^{ij} - \hat{P}_{t,t-1}^{ij}F_j^T - F_j(\hat{P}_{t,t-1}^{ij})^T + F_j\hat{P}_{t-1}^{ij}F_j^T\Big)\Bigg) \\ &+ \underline{\left(F_j-\Lambda_f\right)\Omega_f\left(F_j-\Lambda_f\right)^T + \nu_q\Sigma_q}\Bigg)\Bigg) \\ &- \frac{1}{2}\left(\underline{\nu_q + 2m + 1} + \sum_{i=1}^{N_x}\hat{p}_{ij}\left(T_i-1\right)\right)\ln|Q_j|.\end{aligned} \tag{C.19}$$

[2] the regularisation terms, which are contributed by the prior over parameters, are underlined

By optimising Equation C.19, we obtain the update equation for $Q_j$:

$$\begin{aligned}
\hat{Q}_j =& \underset{Q_j}{\operatorname{argmax}}\, \mathcal{Q}\left(Q_j\right) \\
=& \frac{1}{\underline{\nu_q + 2m + 1} + \sum_{i=1}^{N_x} \hat{p}_{ij}\left(T_i - 1\right)} \Bigg( \\
& \left( \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=2}^{T_i} \left( \hat{P}_t^{ij} - \hat{P}_{t,t-1}^{ij} F_j^T - F_j (\hat{P}_{t,t-1}^{ij})^T + F_j \hat{P}_{t-1}^{ij} F_j^T \right) \right) \\
& + \underline{\left(F_j - \Lambda_f\right) \Omega_f \left(F_j - \Lambda_f\right)^T + \nu_q \Sigma_q} \Bigg) \\
=& \frac{1}{\underline{\nu_q + 2m + 1} + \sum_{i=1}^{N_x} \hat{p}_{ij}\left(T_i - 1\right)} \Bigg( \varphi_j - \psi_j F_j^T - F_j (\psi_j)^T + F_j \phi_j F_j^T \\
& + \underline{\left(F_j - \Lambda_f\right) \Omega_f \left(F_j - \Lambda_f\right)^T + \nu_q \Sigma_q} \Bigg).
\end{aligned} \tag{C.20}$$

The choice of auxiliary variables in Equation C.12 made the derivation of the update equations simpler, however, the final form in Equation C.20 involves subtractions of positive definite matrices, which are numerically non-stable operations. More stable update equations can be obtained by rewriting the equation as follows:

$$\begin{aligned}
\hat{Q}_j =& \frac{1}{\nu_q + 2m + 1 + \sum_{i=1}^{N_x} \hat{p}_{ij}\left(T_i - 1\right)} \Bigg( \\
& \left( \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=2}^{T_i} \left( \left( \hat{x}_t^{ij} - F_j \hat{x}_{t-1}^{ij} \right)\left( \hat{x}_t^{ij} - F_j \hat{x}_{t-1}^{ij} \right)^T + \hat{V}_t - \hat{V}_{t,t-1}^{ij} F_j^T - F_j (\hat{V}_{t,t-1}^{ij})^T + F_j \hat{V}_{t-1}^{ij} F_j^T \right) \right) \\
& + \left(F_j - \Lambda_f\right) \Omega_f \left(F_j - \Lambda_f\right)^T + \nu_q \Sigma_q \Bigg)
\end{aligned} \tag{C.21}$$

## C.5 Observation Matrix H

$$\begin{aligned}
\mathcal{Q}\left(H_j\right) =& -\frac{1}{2} tr \Bigg( R_j^{-1} \Big( - \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} z_t^i (\hat{x}_t^{i,j})^T {H_j}^T - H_j \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \hat{x}_t^{i,j} {z_t^i}^T + H_j \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \hat{P}_t^{ij} H_j^T \\
& + \underline{\left(H_j - \Lambda_h\right) \Omega_h \left(H_j - \Lambda_h\right)^T} \Big) \Bigg) \\
=& -\frac{1}{2} tr \Bigg( R_j^{-1} \Big( - \Big( \underline{\Lambda_h \Omega_h} + \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} z_t^i (\hat{x}_t^{i,j})^T \Big) H_j^T - H_j \Big( \underline{\Omega_h \Lambda_h^T} + \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \hat{x}_t^{i,j} {z_t^i}^T \Big) \\
& + H_j \Big( \underline{\Omega_h} + \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \hat{P}_t^{ij} \Big) H_j^T \Big) \Bigg).
\end{aligned} \tag{C.22}$$

Maximising Equation C.22:

$$\begin{aligned}\hat{H}_j =& \underset{H_j}{\operatorname{argmax}}\, \mathcal{Q}\left(H_j\right) \\ =& \Big(\underline{\Lambda_h \Omega_h} + \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} z_t^i (\hat{x}_t^{i,j})^T\Big) \Big/ \Big(\underline{\Omega_h} + \sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \hat{P}_t^{ij}\Big) \\ =& \Big(\underline{\Lambda_h \Omega_h} + \Gamma_j\Big) \Big/ \Big(\underline{\Omega_h} + \Phi_j\Big).\end{aligned} \tag{C.23}$$

## C.6 Observation Noise Covariance R

$$\begin{aligned}\mathcal{Q}\left(R_j\right) =& -\frac{1}{2} tr\Bigg(R_j^{-1}\Bigg(\Bigg(\sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \Big(z_t^i z_t^{i^T} - z_t^i (\hat{x}_t^{i,j})^T H_j^T - H_j \hat{x}_t^{i,j} z_t^{i^T} + H_j \hat{P}_t^{ij} H_j^T\Big)\Bigg) \\ &+ \underline{\left(H_j - \Lambda_h\right) \Omega_h \left(H_j - \Lambda_h\right)^T + \nu_r \Sigma_r}\Bigg)\Bigg) \\ &- \frac{1}{2}\left(\underline{\nu_r + n + m + 1} + \sum_{i=1}^{N_x} \hat{p}_{ij} T_i\right) \ln |R_j|\end{aligned} \tag{C.24}$$

Maximising Equation C.24:

$$\begin{aligned}\hat{R}_j =& \underset{R_j}{\operatorname{argmax}}\, \mathcal{Q}\left(R_j\right) \\ =& \frac{1}{\underline{\nu_r + n + m + 1} + \sum_{i=1}^{N_x} \hat{p}_{ij} T_i} \Bigg( \\ & \Bigg(\sum_{i=1}^{N_x} \hat{w}_{ij}\hat{p}_{ij} \sum_{t=1}^{T_i} \Big(z_t^i z_t^{i^T} - z_t^i (\hat{x}_t^{i,j})^T H_j^T - H_j \hat{x}_t^{i,j} z_t^{i^T} + H_j \hat{P}_t^{ij} H_j^T\Big)\Bigg) \\ & + \underline{\left(H_j - \Lambda_h\right) \Omega_h \left(H_j - \Lambda_h\right)^T + \nu_r \Sigma_r}\Bigg) \\ =& \frac{1}{\underline{\nu_r + n + m + 1} + \sum_{i=1}^{N_x} \hat{p}_{ij} T_i} \Bigg( \Lambda_j - \Gamma_j H_j^T - H_j (\Gamma_j)^T + H_j \Phi_j H_j^T \\ & + \underline{\left(H_j - \Lambda_h\right) \Omega_h \left(H_j - \Lambda_h\right)^T + \nu_r \Sigma_r}\Bigg)\end{aligned} \tag{C.25}$$

Similar to Equation C.21, a numerically more stable version of Equation C.25 is given by:

$$\begin{aligned}\hat{R}_j =& \frac{1}{\nu_r + n + m + 1 + \sum_{i=1}^{N_x} \hat{p}_{ij} T_i} \Bigg( \\ & \left( \sum_{i=1}^{N_x} \hat{w}_{ij} \hat{p}_{ij} \sum_{t=1}^{T_i} \left( \left( z_t^i - H_j \hat{x}_t^{i,j} \right) \left( z_t^i - H_j \hat{x}_t^{i,j} \right)^T + H_j \hat{V}_t^{ij} H_j^T \right) \right) \\ & + \left( H_j - \Lambda_h \right) \Omega_h \left( H_j - \Lambda_h \right)^T + \nu_r \Sigma_r \Bigg)\end{aligned} \tag{C.26}$$

## C.7 Initial State

Following the same line of reasoning of the previous optimisation operations, the ML posterior (regularisation can be done as with the other model parameters) for the mean and covariance of the initial state of each model $j$ is given by:

$$\begin{aligned}\hat{\mu}_j &= \frac{\sum_{i=1}^{N_x} \hat{p}_{ij} \hat{x}_1^{ij}}{\sum_{i=1}^{N_x} \hat{p}_{ij}} \\ &= \frac{\zeta_j}{\hat{N}_j}\end{aligned} \tag{C.27}$$

$$\begin{aligned}\hat{V}_j &= \frac{\sum_{i=1}^{Nx} \hat{p}_{ij} \left( \hat{P}_1^{ij} - \hat{x}_1^{ij} \mu_j^T - \mu_j (\hat{x}_1^{ij})^T - \mu_j \mu_j^T \right)}{\sum_{i=1}^{Nx} \hat{p}_{ij}} \\ &= \frac{\eta_j - \zeta_j \mu_j^T - \mu_j \zeta_j^T + \hat{N}_j \mu_j \mu_j^T}{\hat{N}_j}\end{aligned} \tag{C.28}$$

# Appendix D

# Implementation Details

This appendix provides details about the implementation of the EA algorithm. The following script presents the actual high level functions used in the code:

```
EA_obj=EA(Models);
for t=1:numel(frames)
    d = Get_Detections();
    EA_obj = EA_obj.update_t(d,t);
    EA_obj = EA_obj.run_EA();
    [EA_obj,detect_obj_assig] = EA_obj.ManageObjects(d);
    EA_obj = EA_obj.UpdateAppearance(d,detect_obj_assig);
end
```

Initially, an EA object is created using the class constructor `EA()`. At each time step the function `Get_Detections()` calculates object detections. The method `update_t()` adds the incoming detections to the current list of observations and calculates the initialisation of the association factors by comparing the appearance models of the existing objects with those of the current detections. Subsequently, `run_EA()` runs the EA algorithm on a time window predefined by the user. Finally, `ManageObjects()` decides which objects will not be updated any more, whereas `UpdateAppearance()`

updates the appearance model of the objects to which at least one detection was confidently assigned. A detailed pseudo-code of the tracking process is summarised in Algorithm 2.