Behavioral Recognition and multi-target tracking in partially observed environments

Arsène Fansi Tchango

UNIVERSITÉ DE LORRAINE

Département de formation doctorale en informatique **École doctorale IAEM Lorraine**
UFR STMIA

Reconnaissance comportementale et suivi multi-cible dans des environnements partiellement observés

THÈSE

présentée et soutenue publiquement le 04 décembre 2015

pour l'obtention du

Doctorat de l'université de Lorraine

(spécialité informatique)

par

Arsène Fansi Tchango

Composition du jury

Rapporteurs :	François Brémond	Directeur de recherche, INRIA Sophia Antipolis
	Séverine Dubuisson	Maître de conférences, Université Pierre et Marie Curie
Examinateurs :	James Crowley	Professeur, Institut National Polytechnique de Grenoble
	Jean-François Mari	Professeur, Université de Lorraine
Directeur de thèse :	Alain Dutech	Chargé de recherche, INRIA Nancy - Grand Est
Membres de la famille :	Vincent Thomas	Maître de conférences, Université de Lorraine
	Laurent Navarro	Ingénieur de recherche, Thales

Laboratoire Lorrain de Recherche en Informatique et ses Applications — UMR 7503

THALES

Inria INVENTEURS DU MONDE NUMÉRIQUE

Loria Laboratoire lorrain de recherche en informatique et ses applications

Mis en page avec la classe thloria.

Acknowledgments

First and foremost, I would like to thank my advisors. Vincent, Olivier, Fabien and Alain are just outstanding people with incredible scientific background and real interpersonal skills. During these four last years, they helped me go through all my paperwork, both scientific and administrative, and God knows how numerous they are when you are a foreigner. I have appreciated every discussion we had, both professional and personal. Although they were not experts of the main topic discussed in this thesis, I would like to highlight their capacity to rapidly embrace the subject and provide me with wise guidance. From Vincent, I have learned from his pedagogical abilities which helped me improve my presentation skills as well as my scientific dissemination skills. Without any doubt, Olivier, equipped with an implacable scientific rigor, helped me improve my writing skills and has mainly contributed to my scientific development. Regarding Fabien, his wide panorama of the industrial world definitely helped me to better combine academic and industrial expectations around this thesis. He lavished me the required motivation to surpass myself in order to satisfy both parties. Finally, I owe Alain the improvement of my oral presentation skills.

Secondly, I also want to acknowledge the jury's members of this thesis who have accepted to evaluate my works. Particularly, the reviewers Sévérine DUBUISSON and François BREMOND provide me with insightful comments in order to improve and broadcast the works that have been done in this document. In the same line, I would like to thank the examiners for their scientific rigor and their constructive questions during the defense.

Next, I would like to highlight the environment in which this thesis has been developed, at least from the academic point of view. INRIA Nancy Grand-Est / LORIA is not only one of the most important research center in France, but also a place where a multicultural and keen community of researchers/engineers are encouraged to interact. I have learned a lot from these interactions and I would particularly acknowledge the support of the MAIA/LARSEN team, especially François CHARPILLET, Véronique CONSTANT as well as Laurence FELICITE. In the same line, I would like to thank, for their friendship and help, my fellow PhD students and post-docs including Jilles, Manel, Mohamed, Nassim, Mihai, Abdellah, Inaki, Thomas, and many others.

All the achievements of this thesis would not been realized without the participation of the ThereSIS Lab at THALES, the second environment in which this thesis has been developed. I met there wonderful people who did not hesitate, despite their high workload, to dedicate some of their time to my scientific concerns. This includes Laurent from who I get lo of my inspirations, Christophe, Philippe, Jean-Benoît, Martin, Willy, Alexandre, Majorie, Edith and many others.

Finally, I want to acknowledge my wife, Nina Michelle, for their patience, support and understanding during this thesis. She, and more recently my son Johan Edward, have been the main motivation of all the efforts I have realized during this thesis. And there is no word here to describe how grateful I am to have you by my side. I would also like to thank all my Family, especially my parents, brothers and sisters, to continuously keep watch on me and offer me the perfect environment and state of mind for running up this thesis.

To Johan Edward, Nina Michelle and all my Family.

Contents

List of Figures

List of Tables

List of Algorithms

List of Abbreviations

AR	Activity recognition
BF	Bayesian filter
BN	Bayesian network
CFG	Context-free grammar
CPD	Conditional probability density
DA	Data association
DBN	Dynamic Bayesian network
EA	Evolutionary algorithm
EKF	Extended Kalman filter
FJAH	Feasible joint association hypothesis
GNN	Global nearest neighbor
HMM	Hidden Markov model
IS	Importance sampling
IT-MCJPDAF	Interacting target - Monte Carlo - joint probabilistic data association filter
JPDA	Joint probabilistic data association
JPDAF	Joint probabilistic data association filter
KF	Kalman filter
MC	Monte Carlo
MC-JPDAF	Monte Carlo - joint probabilistic data association filter
MHT	Multiple hypothesis tracker
MPSM	Microscopic pedestrian simulation models
MRF	Markov random field
MTT	Multi-target tracking
PF	Particle filter
PNF	Past-now-future
POMDP	Partially observable Markov decision process
RPF	Regularized particle filter
SCFG	Stochastic context free grammar
SIS	Sequential importance sampling
SMC	Sequential Monte Carlo
STAR	Simultaneous tracking and activity recognition
STT	Single-target tracking
UKF	Unscented Kalman filter

List of Publications

A. Fansi Tchango, V. Thomas, O. Buffet, F. Flacher, and A. Dutech, "Simulation-based behavior tracking of pedestrians in partially observed indoor environments," in *Proceedings of the 13th International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS-14)*, May 2014.

A. Fansi Tchango, V. Thomas, O. Buffet, A. Dutech, and F. Flacher, "Simultaneous tracking and activity recognition (STAR) using advanced agent-based behavioral simulations," in *Proceedings of the 21st European Conference on Artificial Intelligence (ECAI)*, August 2014, pp. 1105–1106.

A. Fansi Tchango, V. Thomas, O. Buffet, F. Flacher, and A. Dutech, "Towards the usage of advanced behavioral simulations for simultaneous tracking and activity recognition," in *Proceedings of the 7th European Starting AI Researcher Symposium (STAIRS)*, August 2014, pp. 121–130.

A. Fansi Tchango, V. Thomas, O. Buffet, A. Dutech, and F. Flacher, "Tracking multiple interacting targets using a joint probabilistic data association filter," in *Proceedings of the 17th International Conference on Information Fusion (FUSION)*, July 2014, pp. 1–8.

Chapter 1

Introduction

1.1 Examples and Motivations

Let us consider a complex environment, e.g. a subway station, in which pedestrians may evolve and perform several types of actions. The environment is equipped with a sensor network which has eventually been configured to partially cover areas therein. The sensors provide noisy location data about pedestrians when the latter are detected within covered areas. Additionally, the environment contains objects with which a pedestrian may interact in order to fulfill his goal. For example, in a subway station, these objects include ticket machines, ticket barriers, ATMs, and escalators. A passenger within the station may need to interact with these objects either for moving to the train platform, or for leaving the station.

Given such an environment, one of the major concerns of several applications is to automatically retrieve, on the sole basis of data provided by the sensor network, relevant information about what is going on there, either for surveillance [Haritaoglu et al., 2000], assistance [Zouba et al., 2009, Dubois and Charpillet, 2013], smart homes [Krumm et al., 2000] or advertising purposes. For example, knowing that a passenger has bought a ticket to a given destination, let us say Paris, and that he is approaching an advertising panel, then proposing an advertisement centered on the city of Paris will likely catch up his attention.

In this thesis, we are interested in a supervision system for surveillance purposes. More specifically, we would like to determine whether or not the pedestrians present in the scene behave *"normally"*, i.e., as expected in the environment. Thus, the problem of interest consists in estimating the pedestrians' behaviors while relying on observation data received from sensors. This includes, in case the environment is not fully under sensory coverage, inferring the behaviors of the underlying pedestrians when the latter are within a non-covered area.

1.2 Problematic

The problem we are interested in broadly consists in determining the state of a partially observed dynamical system on the sole basis of possibly noisy observations (or measurements) received from a sensor network connected to the system. In the literature, this class of problems is referred to as "*filtering problems*". Moreover, when the objective is to estimate the behaviors of a set of targets (here, pedestrians), the term *behavioral tracking* can be used to refer to corresponding

filtering problems. In such a case, the dynamical system under investigation is made up of the set of targets evolving in the environment.

Behavioral tracking is generally a challenging problem even in the simplest case where the focus is on a single pedestrian. This is probably because the behavior exhibited by a pedestrian, besides being *individual*, is *adaptive*. By *individual*, we mean that such a behavior depends on internal (i.e., personal) properties characterizing the underlying pedestrian as for example the stamina, the age, and/or the education level. For example, in the subway station scenario introduced previously, a less civic passenger would be likely to cheat on the ticket barriers.

By *adaptive*, we mean that the pedestrian behavior is sensitive to the environment in which the pedestrian is immersed. For example, considering the subway station scenario, let us assume we are in presence of a hungry passenger willing to get a train. Also, let us assume that he owns enough money for purchasing some food, together with the required tickets to cross the ticket barriers. In presence of a vending machine within the environment, the passenger will likely modify his main motivation to get some food from the machine before continuing to the train platform. This behavior would never occur if the vending machine were not there. In such a situation and under the assumption that there is no food stores in the station, either the utility of taking the train is dominant and the passenger will move to the platform, or the need for food is dominant and the passenger will leave the station to get something to eat. This illustration perfectly shows how objects, by the simple fact of their presence in the environment, influence pedestrian behaviors.

Another facet of the influences of environment objects on pedestrian behaviors comes from their dynamical nature. Indeed, the states of objects in the environment are not static over time and thus, they are subject to dynamical changes — either from exogenous events or from actions initiated by the pedestrians — which may subsequently affect the pedestrian behaviors. For example, in the subway station scenario, if a fire alarm is suddenly launched, pedestrians present in the environment would likely want to exit the station whatever their current behaviors.

The environment objects are not the only environmental factors which may affect the behavior of a given pedestrian. In the general case where there are multiple pedestrians evolving together in the environment, the behavior of a given pedestrian may be influenced by the presence of another one in his surrounding. For instance, in the subway station scenario, the behavior of a passenger willing to cheat on the ticket barriers will be affected by the presence of controller agents. We use the generic term *target interactions* to refer to this form of behavioral influences between pedestrians. Target interactions add another layer of complexity to the behavioral tracking problem. Indeed, it is required to reason on the behaviors of all the pedestrians at once rather than focusing on the behavior of each pedestrian on an individual basis. This may rapidly prove to be computationally impractical if no special attention is paid when designing a solution to such a problem.

1.3 Literature snapshot and current limitations

From the description made in Section 1.2, a solution to the behavioral tracking problem could, roughly speaking, be qualitatively assessed on the basis of the following criteria:

- **Pedestrian characterization**: the solution should offer an interface allowing to define specific properties related to each pedestrian.

- **Contextual modeling**: the solution should be able to represent and reason on the environmental context in which the pedestrians are immersed. This includes, among others, the objects present in the environment (e.g., vending machine), the service they offer (e.g., food/drinks) as well as their usage by a given pedestrian.

- **Environmental changes**: the solution should integrate, within its inference process, dynamical modifications occurring in the environmental context. These modifications are either caused by pedestrians themselves or by external factors.

- **Interaction management**: the solution should be able to efficiently handle pedestrian's interactions while keeping the computational workload at a reasonable level.

The simplest solution to the behavioral tracking problem consists in using human operators behind a set of monitoring screens (e.g., the sensors are cameras) with the aim of analyzing the scene context and detecting abnormal people behaviors. This is not a trivial task to perform, even for humans, and misinterpretations are very common. One reason for this issue is that the operators are required to remain concentrated over long periods of time, thus making the task inadequate.

In the literature, the filtering problem has usually been addressed using the Bayesian filtering framework [Diard et al., 2003], and, within this framework, automated solutions have been proposed to handle the behavioral tracking problem [Ke et al., 2007, Aggarwal and Ryoo, 2011]. Two principal aspects of the pedestrian behavior have generally been addressed. The first one concerns the *motion* and deals with the trajectory followed by the considered pedestrian. On the other hand, the second aspect is related to the *activity*, in the general sense, performed by the underlying pedestrian within his environment.

Depending on which aspect they focus on, there are solutions in the literature whose objective is either motion tracking [Khan et al., 2003, Pellegrini et al., 2009, Luber et al., 2010, Tastan and Sukthankar, 2011], or activity recognition [Dollar et al., 2005, Niebles et al., 2008, Liu et al., 2009]. Nevertheless, moving pedestrians are usually driven by an inner motivation in relation with the activity they are performing in the environment. Therefore, location and activity are contextually dependent and considering both problems simultaneously definitely presents a significant advantage as the knowledge regarding the location may help improving the estimation of the activity and inversely. On the basis of this thought, solutions have been proposed in the literature [Wilson and Atkeson, 2005, Manfredotti et al., 2011, Cattelani et al., 2014] under the general framework of *Simultaneous Tracking and Activity Recognition* where tracking is used to improve activity recognition and vice-versa. However, with respect to the above-mentioned criteria, all these methodologies generally present at least one of the following limitations:

- pedestrian characterization: the tracked pedestrians are simply characterized by very basic properties (e.g., location, velocity, activity) [Pellegrini et al., 2009, Tastan and Sukthankar, 2011] and there is no means to represent internal features (e.g., stamina) governing the intrinsic nature of the humans in the real world;

- contextual modeling: the approaches designed to capture the context of the environment in which the tracking process are rather limited. They usually do not provide enough details for explicitly dealing with objects present in the environment, especially, their usage and/or their ability to directly influence pedestrian behaviors by the different services they propose

[Wilson and Atkeson, 2005, Niebles et al., 2008, Manfredotti et al., 2011];

- environmental changes: most solutions are usually designed to cope with situations in which the environment context is assumed static over time [Dollar et al., 2005, Wilson and Atkeson, 2005, Luber et al., 2010];

- interaction management: we roughly distinguish two types of methodologies. The first one concerns methodologies which make problem-specific assumptions regarding the nature of interactions between targets. While these assumptions allow to manage interactions with low computational complexity, the derived solutions usually cannot be generalized to other problems in practice [Khan et al., 2003]. On the other hand, approaches belonging to the second category embed more generic interaction models. However, they usually suffer from high computational complexity [Cattelani et al., 2014].

1.4 Contributions

In this thesis, we are interested in developing a generic solution to the behavioral tracking problem capable of coping with the current limitations of the state of the art. To this end, we articulate our work around two main axes.

The first axis focuses on issues related to the top three criteria introduced previously (pedestrian personalization, contextual modeling, and environmental changes). As described in the literature, we share the idea of performing both the motion tracking and the activity recognition simultaneously. However, unlike existing approaches, we propose an innovative solution — within the Bayesian filtering framework — in which advanced agent-based behavioral simulators [Shao and Terzopoulos, 2007] are used as a baseline of the inference process. These simulators, roughly speaking, aim at realistically reproducing human behaviors within virtual environments using situated and autonomous agents equipped with sensing capabilities. The advantages of the proposed solution reside in that, through these simulators, it is possible to:

- represent each pedestrian being tracked by a virtual agent and define, besides basic properties such as the location and/or the velocity, high-level and intuitive internal properties characterizing its dynamics (e.g., thirst level);

- reason on richer contextual models by virtually representing, within the simulators, the environment in which the tracking process is taking place as well as the different objects therein;

- simulate the dynamical changes occurring in the real environment within the simulator so as to take into account their effects on the behaviors of virtual agents representing the pedestrians being tracked.

The second axis of this thesis focuses on the management of target interactions. Unlike existing approaches, we seek for a solution relying on a generic model of interactions while avoiding, at the same time, the issues related to the computational complexity. We propose a factored algorithm — for tracking multiple interacting targets — which can be viewed as a collection of collaborating (sub-)filtering processes, each one dedicated to a single target. The particularity of our approach

resides in that, for reducing the complexity without much degrading the quality of behavior estimates, we design heuristics for aggregating, on an individual basis, the beliefs related to the behaviors of all the targets into few representative states so that the latter are the only piece of information needed by each (sub-)filtering process for subsequently updating the corresponding belief.

1.5 Document Structure

The remainder of this document is organized in three parts described as follows:

- **Part I**: the first part is dedicated to the formal introduction of the filtering problem as well as the description of the mathematical background related to the Bayesian filtering framework;
- **Part II**: the second part focuses on the special case of a single target (pedestrian) and demonstrates the advantages of using advanced agent-based behavioral simulators in the context of behavioral tracking;
- **Part III**: the third part of the document is devoted to the general case of multiple targets with a particular attention to the management of target interactions. It describes the algorithm/heuristics proposed for efficiently handling target interactions within a behavioral inference process.

At the end, some conclusions and directions for future research are presented.

Part I

Bayesian Filtering

Chapter 2

The Filtering Problem

Contents

As previously stated in Chapter 1, the problem of interest in this thesis globally falls into the category of problems known in the literature as filtering problems. The purpose of this chapter is to formally introduce the filtering problem in the general case of a dynamical system as well as the mathematical background related to the Bayesian filter [Bayes, 1763] [Jeffreys, 1973] [Diard et al., 2003], a theoretically optimal method for addressing such a problem from the Bayesian perspective.

2.1 Partially Observed Dynamical System: The Filtering Problem

A dynamical system is a system whose internal state, represented by a random variable $\mathbf{x}$, evolves over time causally, that is, the next state of the system depends only on past and present states. Additionally, when the knowledge of past states carries no additional information that would help determine the future state of a dynamical system and what simply matters is the current state, it is said that the dynamical system verifies the *Markov property* [Markov, 1954]. In the rest of this document, we are interested in dynamical systems meeting this property[1]. More particularly, we focus on those systems whose dynamics can be represented by:

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{w}_t), \forall t \geq 1, \tag{2.1}$$

where $\mathbf{x}_t$ and $\mathbf{x}_{t-1}$ represent the state of the system at time t and $t-1$ respectively, $\mathbf{f}$ is a deterministic function modeling the system dynamics, and $(\mathbf{w}_t)_{t\geq 1}$ is a noisy process representing exogenous factors that may affect the evolution of the system. From a probabilistic perspective, the system's dynamics can be characterized by a probability distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ often called the *state-transition model*. In such a model, the noise $(\mathbf{w}_t)_{t\geq 1}$ from Equation 2.1 is represented by the *non-deterministic* characteristic of the model. Usually, a knowledge Bel_0 regarding the system's initial state at time $t = 0$, also called *prior*, is provided in form of a probability distribution $p(\mathbf{x}_0)$. Formally, such a distribution defines the possible values of the initial state of the system and their corresponding probabilities.

A partially observed dynamical system is a system in which it is not possible to have the complete access to the internal state; instead, sensors are used to obtain noisy (partial) measurements also referred to as observations[2] represented by a random variable $\mathbf{z}$. It is said that the system's state is "*hidden*". The observation $\mathbf{z}_t$ received from the system at a given time t can be modeled as:

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t, \mathbf{v}_t), \forall t \geq 1, \tag{2.2}$$

where $\mathbf{x}_t$ represents the state of the system at time t, $\mathbf{h}$ is a known deterministic observation function of the sensor network, and $(\mathbf{v}_t)_{t\geq 1}$ is a noisy process modeling the imperfection of the sensors used. It is assumed that both noisy processes $(\mathbf{w}_t)$ and $(\mathbf{v}_t)$ are mutually independent. From a probabilistic perspective, the observation process can be characterized by a probability distribution $p(\mathbf{z}_t|\mathbf{x}_t)$ often called the *observation model* which allows to relate the noisy measurement obtained to the system's state. A graphical representation of the evolution of a partially observed dynamical system is illustrated in Figure 2.1.

[1] The term "*dynamical system*", when not specified, will be used to refer to systems verifying the Markov property.

[2] The terms "*measurement*" and "*observation*" will be used interchangeably throughout this document.

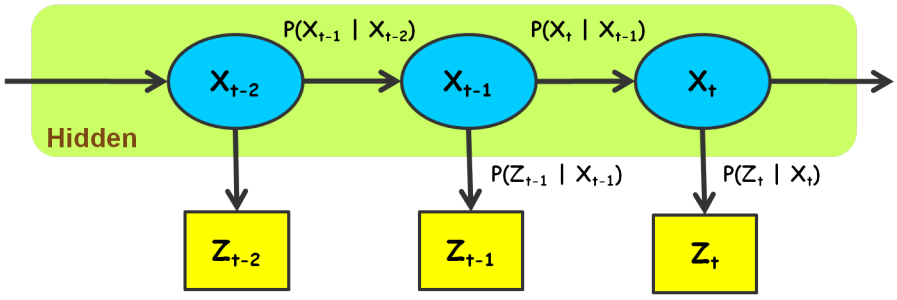


Figure 2.1: The evolution of a partially observed dynamical system.

The filtering problem consists in inferring the current hidden state of a partially observed dynamical system from the observations received up to the current time step and from the prior knowledge about the system. In other words, from a probabilistic perspective, the aim of the filtering problem is to compute the probability distribution over the current state $\mathbf{x}_t$ conditioned on all past measurements $\mathbf{z}_{1:t}$ and initial state knowledge Bel_0. This probability is called the *belief* on the state and is defined as:

$$Bel_t(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0),\ \forall t \geq 1. \tag{2.3}$$

In the next section, we discuss the Bayesian filter, an optimal method for recursively solving the filtering problem.

2.2 Bayesian Filter

The Bayesian filter[Jeffreys, 1973] is a generic method to efficiently address the filtering problem. It relies on the Bayes theorem which describes how new evidences can be used to update previous beliefs. The Bayesian filter is a recursive method in the sense that the belief $Bel_t(\mathbf{x}_t)$ on the system's state at time t is computed from the belief $Bel_{t-1}(\mathbf{x}_{t-1})$ at time $t-1$ when the observation $\mathbf{z}_t$ is received. It handles received measurements in a sequential way rather than in a batch mode; and hence, it is suitable for an online setting in which an estimate of the system's state is required at every time step as a new measurement is obtained. Therefore, as suggested by the Markov property, there is no need to store previously received data for further processing.

In order to build the belief $Bel_t(\mathbf{x}_t)$ at each time step t, the Bayesian filter proceeds in two steps as illustrated in Figure 2.2: the *prediction* and the *correction* steps. In the *prediction step*, the previous belief $Bel_{t-1}(\mathbf{x}_{t-1})$ is modified according to the system's dynamics $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ to obtain the predicted belief $Bel_{t|t-1}(\mathbf{x}_t)$. $Bel_{t|t-1}(\mathbf{x}_t)$ predicts the system's state at time t based on the previous belief state, before taking into account the observation data $\mathbf{z}_t$ at time t. This is expressed as:

$$\begin{aligned} Bel_{t|t-1}(\mathbf{x}_t) &= p(\mathbf{x}_t|\mathbf{z}_{1:t-1}, Bel_0) \\ &= \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) Bel_{t-1}(\mathbf{x}_{t-1}) d\mathbf{x}_{t-1}. \end{aligned} \tag{2.4}$$

In the *correction step*, as the observation data $\mathbf{z}_t$ is received from the sensors, it is used to correct the predicted belief $Bel_{t|t-1}(\mathbf{x}_t)$ via the Bayes law, therefore leading to the updated belief

$Bel_t(\mathbf{x}_t)$. This is expressed as:

$$\begin{aligned} Bel_t(\mathbf{x}_t) &= p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0) \\ &= \frac{p(\mathbf{x}_t, \mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)} \\ &= \frac{p(\mathbf{z}_t|\mathbf{x}_t)p(\mathbf{x}_t|\mathbf{z}_{1:t-1}, Bel_0)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)} \\ &= \frac{1}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)} p(\mathbf{z}_t|\mathbf{x}_t) Bel_{t|t-1}(\mathbf{x}_t), \end{aligned} \tag{2.5}$$

where $p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)$, viewed as a normalizing constant, is given by

$$p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0) = \int p(\mathbf{z}_t|\mathbf{x}_t) Bel_{t|t-1}(\mathbf{x}_t) d\mathbf{x}_t. \tag{2.6}$$

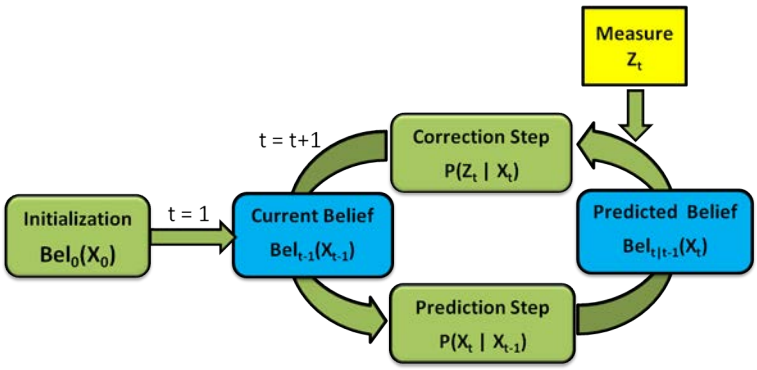


Figure 2.2: Graphical representation of the Bayesian filter. During the prediction step, the state-transition model is used to compute the predicted belief. As the current observation is received, the predicted belief is updated using the observation model leading to the current belief on the system's state.

The application of the Bayesian filter requires the computations of integrals as described in Equations 2.4 and 2.5. However, these computations, in cases of variables defined in multidimensional space and/or systems with strong nonlinearity, are unfortunately intractable in practice. Nevertheless, solutions have been derived to efficiently approximate the Bayesian filter under particular conditions. Among them, one can name the grid-based Bayesian filter [Diard et al., 2003] which subdivides the probability distribution function into a discrete grid. The grid-based filter is appropriate when the cardinality of the state space of the system under consideration is small in such a way that it is possible to enumerate all the states. Another derivative is the well known Kalman filter (KF) [Kalman, 1960] which is used to analytically compute the belief regarding the state of linear systems with additive Gaussian noises. In what follows, we present the details of the Kalman filter.

2.3 Kalman Filter

Let us consider a system whose dynamics $\mathbf{f}(\mathbf{x}_{t-1}, \mathbf{w}_t)$ is linear in the system's state $\mathbf{x}_{t-1}$. Also, we assume that the observation model function $\mathbf{h}(\mathbf{x}_t, \mathbf{v}_t)$ is linear in $\mathbf{x}_t$. Moreover, it is assumed that

both the noise processes $(\mathbf{w}_t)_{t>0}$ and $(\mathbf{v}_t)_{t>0}$ are characterized by zero mean Gaussian distributions with known covariances $\mathbf{Q}_t$ ($\mathcal{N}(0, \mathbf{Q}_t)$) and $\mathbf{R}_t$ ($\mathcal{N}(0, \mathbf{R}_t)$) respectively. Additionally, they have an additive pattern within the system's dynamics and the observation model. Therefore, Equations 2.1 and 2.2 can be respectively rewritten as

$$\begin{aligned} \mathbf{x}_t &= \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{w}_t) \\ &= \mathbf{F}\mathbf{x}_{t-1} + \mathbf{w}_t, \forall t \geq 1, \end{aligned} \tag{2.7}$$

$$\begin{aligned} \mathbf{z}_t &= \mathbf{h}(\mathbf{x}_t, \mathbf{v}_t) \\ &= \mathbf{H}\mathbf{x}_t + \mathbf{v}_t, \forall t \geq 1, \end{aligned} \tag{2.8}$$

where $\mathbf{F}$ and $\mathbf{H}$ are known matrices characterizing the system's dynamics and the observation model respectively.

Under these assumptions and considering that the knowledge regarding the initial state of the system is given by a Gaussian distribution, it can be shown [Kalman, 1960] that both the beliefs $Bel_{t|t-1}(\mathbf{x}_t)$ and $Bel_t(\mathbf{x}_t)$ can be represented by Gaussian distributions whose parameters are sequentially computed by the Kalman filter. The Kalman filter proceeds recursively via the same two steps (*prediction* and *correction*) as in the Bayesian filter and, in each step, the computed belief is expressed in form of a Gaussian density. Assuming that the previous belief $Bel_{t-1}(\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_{t-1|t-1}, \Sigma_{t-1|t-1})$, the different steps are analytically described in what follows:

- **the prediction step:**

$$\begin{aligned} Bel_{t|t-1}(\mathbf{x}_t) &= \mathcal{N}(\mathbf{x}_{t|t-1}, \Sigma_{t|t-1}) \text{ where} \\ \mathbf{x}_{t|t-1} &= \mathbf{F}\mathbf{x}_{t-1|t-1}, \text{ and} \\ \Sigma_{t|t-1} &= \mathbf{Q}_t + \mathbf{F}\Sigma_{t-1|t-1}\mathbf{F}^T; \end{aligned} \tag{2.9}$$

- **the correction step:**

$$\begin{aligned} Bel_t(\mathbf{x}_t) &= \mathcal{N}(\mathbf{x}_{t|t}, \Sigma_{t|t}) \text{ where} \\ \mathbf{x}_{t|t} &= \mathbf{x}_{t|t-1} + \mathbf{K}_t(\mathbf{z}_t - \mathbf{H}\mathbf{x}_{t|t-1}), \\ \Sigma_{t|t} &= \Sigma_{t|t-1} - \mathbf{K}_t\mathbf{H}\Sigma_{t|t-1}, \text{ and} \\ \mathbf{K}_t &= \Sigma_{t|t-1}\mathbf{H}^T[\mathbf{H}\Sigma_{t|t-1}\mathbf{H}^T + \mathbf{R}_t]^{-1}. \end{aligned} \tag{2.10}$$

In the above equations, $\mathbf{K}_t$ is called the Kalman gain. It is interesting to point out that the covariance matrices $\Sigma_{t|t-1}$ and $\Sigma_{t|t}$ together with the gain $\mathbf{K}_t$ do not depend on the received observation $\mathbf{z}_t$ and therefore, they can be precomputed to speed up the filtering process in real time scenarios.

2.4 Other Approaches

In the previous section, we introduced the Kalman filter, a derivative of the Bayesian filter, which is appropriate in presence of linear systems with additive Gaussian noises. Unfortunately, the

Kalman filter is not suitable for nonlinear systems. Approximate solutions have been developed to tackle the filtering problem in case of nonlinear systems either analytically or using Monte Carlo methods [von Neumann and Ulam, 1940, Bauer, 1958]. Here, we present a brief overview of these approaches.

2.4.1 Analytical Approaches

Analytical solutions usually proceed by approximating the nonlinear system's dynamics by a linear function. By doing so, these solutions aim to exploit the intrinsic linearity properties during the computation of integrals within the filtering procedure. These solutions assume, as in the case of the Kalman filter, the Gaussian form of the probability distribution over the system's state. Among the existing methods, we can name the *Extended Kalman filter* (EKF) [Jazwinski, 1970, Einicke and White, 1999] and the *Unscented Kalman filter* (UKF) [Julier and Uhlmann, 1997, 2004]. While the quality of the results obtained with these analytical solutions is quite satisfactory in the context of weakly nonlinear systems, they badly perform in case of strong non-linearity (i.e., multi-modal distributions) as stated in [Einicke, 2012].

2.4.2 Monte Carlo Approaches

Unlike analytical approaches, Monte Carlo solutions do not try to approximate, by a linear function, the dynamics of the system under consideration. Moreover, they do not make any assumptions regarding the form of the probability distribution over the system's states. Instead, they consider representing the belief $Bel_t(\mathbf{x}_t)$ using a set of points — also referred to as samples or particles — from the state space, and they simply proceed by simulating, over time, the evolution of these points using the function characterizing the system dynamics. Monte Carlo approaches, therefore, are able to handle multi-modal distributions and they are more appropriate when the dynamics of the system is strongly nonlinear. Further details regarding these approaches are discussed in Chapter 3.

2.5 Conclusion

In this chapter, we formalized the generic filtering problem within a partially observed dynamical system. We presented the general framework of the Bayesian filter which provides an optimal solution to the filtering problem. Unfortunately, this solution is intractable unless in special case of linear systems with additive Gaussian noises for which the Kalman filter is an ideal solution. Approximate solutions have been developed to tackle the filtering problem in case of nonlinear systems either analytically or using Monte Carlo methods. Analytical solutions usually proceed by approximating the nonlinear system's dynamics by a linear function. However, they poorly perform in case of strong non-linearity (multi-modal distributions). On the other hand, Monte Carlo solutions do not make any assumptions regarding the form of the probability distribution and they are more appropriate when the dynamics of the system is strongly nonlinear (multi-modal distributions). The next chapter is devoted to the description of the general framework of the Sequential Monte Carlo (SMC) methods for the filtering problem also known as *particle filters*.

Chapter 3

Particle Filtering

Contents

In the previous chapter, we introduced the Kalman filter, a derivative of the Bayesian filter suitable in case of linear systems with additive Gaussian noises. However in practice, several problems, including the one we are interested in within this thesis, concern nonlinear systems with multi-modal distributions. As described in Section 2.4, approximate solutions to the filtering problem have been developed in case of nonlinear systems, and most notably the Monte Carlo approaches. The purpose of this chapter is to describe the mathematical background of these approaches. After introducing the concept of a Monte Carlo method, we present the general framework of Sequential Monte Carlo (SMC) approaches for the filtering problem also known as *particle filters*.

3.1 Monte Carlo methods

According to Sawilowsky [Sawilowsky, 2003], a Monte Carlo method is a technique that can be used to solve a mathematical or statistical problem which relies on repeated random sampling for obtaining numerical results. For example, let us consider the problem of approximating a probability distribution $p(x)$, $x \in \mathcal{X}$. Also, let us assume that it is possible to sample N independent random variables $x^i \sim p(x), i = 1 \cdots N$. Then, for sufficiently large N, the probability distribution $p(x)$ can be approximated by the following expression:

$$\hat{p}(x) = \frac{1}{N} \sum_{i=1}^{N} \delta_{x^i}(x),$$

where $\delta_{x^i}(x)$ represents the Dirac function located at x^i.

Now, assume that we are interested in approximating the expectation of a function $f : \mathcal{X} \mapsto \mathbb{R}$, given by

$$E(f) = \int f(x)p(x)dx.$$

Using a Monte Carlo approach, the following equation provides an approximation of $E(f)$:

$$\hat{E}(f) = \frac{1}{N} \sum_{i=1}^{N} f(x^i).$$

It may not be obvious to directly sample from $p(x)$ in many problems. In such cases, the Importance Sampling, which is described in the next section, can be a solution.

3.2 Importance Sampling Principle

When it is difficult to sample from the probability density $p(x)$, one can use the Importance Sampling (IS) principle [Bergman, 1999] for approximating $p(x)$. In order to do this, the IS principle relies on another probability density $q(x)$ from which it is easy to generate samples and which verifies the following property:

$$\forall x \in \mathcal{X}, p(x) > 0 \text{ implies } q(x) > 0. \tag{3.1}$$

Then, based on N samples $\{x^i\}_{i=1}^N$ drawn from $q(x)$ ($x^i \sim q(x), i = 1, \cdots, N$), the approximation $\hat{p}(x)$ of $p(x)$ is given by

$$\hat{p}(x) = \sum_{i=1}^{N} w^i \delta_{x^i}(x),$$

where $w^i \propto \frac{p(x^i)}{q(x^i)}$ is the normalized importance weight of the i^{th} sample. $q(x)$ is called the *importance density* or the *proposal density*.

3.3 Sequential Importance Sampling

Sequential importance sampling (SIS) [Arulampalam et al., 2002] is the basis for most SMC methods developed over the past decades. It is a technique for estimating the posterior distribution $p(\mathbf{x}) = p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}, Bel_0)$ [3] regarding the complete trajectory of the state of a partially observed dynamical system characterized by Equations 2.1 and 2.2 using Monte Carlo simulations. The key idea is to represent the required posterior distribution $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}, Bel_0)$ by a finite set of random samples $\{\mathbf{x}_{0:t}^i\}_{i=1}^N$ with associated weights $\{w_t^i\}_{i=1}^N$.

In order to approximate $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}, Bel_0)$, SIS uses a proposal density $q(\mathbf{x}) = q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ from which the samples are generated. Therefore, using the IS principle, the posterior distribution $p(\mathbf{x})$ can be approximated by

$$p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) \approx \hat{p}(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = \sum_{i=1}^{N} w_t^i \delta_{\mathbf{x}_{0:t}^i}(\mathbf{x}_{0:t}), \text{ where} \tag{3.2}$$

$$w_t^i \propto \frac{p(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t})}{q(\mathbf{x}_{0:t}^i|\mathbf{z}_{1:t})}. \tag{3.3}$$

Assuming that the proposal density $q(\mathbf{x}) = q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ is of the form

$$q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}) = q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t}) q(\mathbf{x}_{0:t-1}|\mathbf{z}_{1:t-1}), \tag{3.4}$$

then one can sample $\mathbf{x}_{0:t}^i$ from $q(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ by sequentially sampling $\mathbf{x}_t^i$ from $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})$ at each time t. Given that the system verifies the Markov property, $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t})$ could be ideally chosen in such a way that it depends only on the system state $\mathbf{x}_{t-1}$ at the previous time step as well as the observation $\mathbf{z}_t$ at the current time step, that is, $q(\mathbf{x}_t|\mathbf{x}_{0:t-1}, \mathbf{z}_{1:t}) = q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t)$ [Arulampalam et al., 2002]. As a result, at each time t, one can easily estimate the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ regarding the current state of the system by using the set of random weighted samples $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$ where w_t^i is derived from Equation 3.3 and its value is given by

$$w_t^i \propto w_{t-1}^i \frac{p(\mathbf{z}_t|\mathbf{x}_t^i) p(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_t^i|\mathbf{x}_{t-1}^i, \mathbf{z}_k)}. \tag{3.5}$$

[3] For simplification purposes, the term Bel_0 may subsequently be omitted throughout this document. Therefore, the terms $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t}, Bel_0)$ and $p(\mathbf{x}_{0:t}|\mathbf{z}_{1:t})$ are interchangeably used to refer to the posterior distribution of a dynamical system.

Algorithm 3.1: SIS Algorithm

Algorithm SIS($\{\mathbf{x}_{t-1}^i, w_{t-1}^i\}_{i=1}^N$, $\mathbf{z}_t$)
  for $i = 1 : N$ **do**
    Draw $\mathbf{x}_t^i \sim q(\mathbf{x}_t^i | \mathbf{x}_{t-1}^i, \mathbf{z}_t)$
    Assign the sample's weight, w_t^i, according to Equation 3.5
  return $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$

Details of how to obtain Equation 3.5 can be obtained in [Arulampalam et al., 2002]. Therefore, the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is approximated as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \hat{p}(\mathbf{x}_t|\mathbf{z}_{1:t}) = \sum_{i=1}^{N} w_t^i \delta_{\mathbf{x}_t^i}(\mathbf{x}_t). \tag{3.6}$$

The SIS approach can be viewed as a recursive propagation of the weights and support points (samples) as each measurement is received sequentially. A one step pseudo-code of the SIS algorithm is described in Algorithm 3.1.

A common problem with the SIS algorithm is the *degeneracy phenomenon* [Liu and Chen, 1995] where after a few iterations, all but one sample will have negligible weights (see Figure 3.1). As a consequence, this leads to a poor approximation of the posterior distribution since only one sample effectively contributes to the approximation. One solution to reduce this phenomenon is the introduction of the *resampling stage* within the SIS algorithm, leading to a new class of algorithms commonly called *particle filter* that we describe in the next section.

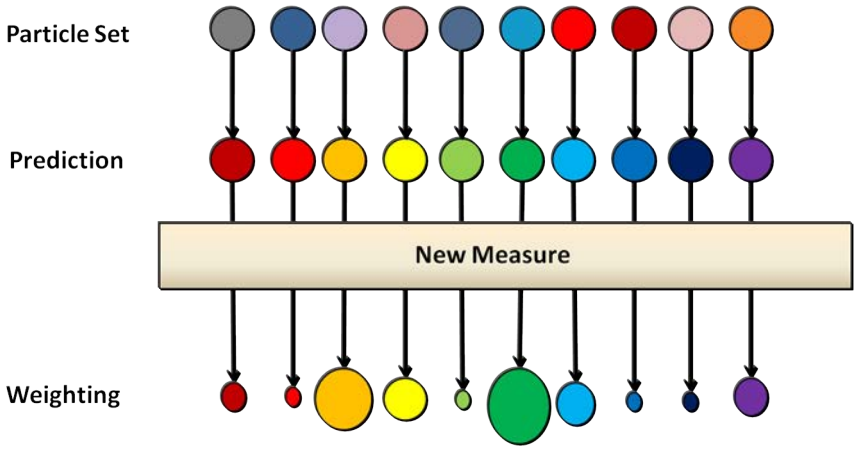


Figure 3.1: Graphical illustration of the degeneracy phenomenon of the SIS algorithm. After the prediction step, when a new measure is received, the weighting step takes place from which only few particles have non negligible weights. The state (value) and the weight of a particle are respectively symbolized by its color and its radius. Inspired from [Parker, 2013].

3.4 Particle Filter

Like the SIS algorithm, a particle filter (PF) is an approximation of the Bayesian filter using a Monte Carlo approach. As stated in Section 2.1, the objective of a filtering problem is to estimate the belief $Bel_t(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0)$ of a partially observed dynamical system characterized by Equations 2.1 and 2.2. The PF algorithm is mainly derived from the SIS algorithm; therefore, it approximates the belief $Bel_t(\mathbf{x}_t)$ in Equation 2.5 by a finite set of weighted samples, also referred to as particles, $\mathcal{S}_t = \{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$. Each particle represents an hypothesis regarding the actual state of the system and its associated weight represents the likelihood of this hypothesis with respect to the reality. Therefore, particles with high weights are near the modes of the posterior distribution while those with low weight are near the tail. The particle set $\mathcal{S}_t$ is typically computed from the previous set $\mathcal{S}_{t-1}$ and the current observation $\mathbf{z}_t$ in three steps [Arulampalam et al., 2002]:

- **prediction:** a sample $\mathbf{x}_{t|t-1}^i$ is generated from each sample $\mathbf{x}_{t-1}^i$ of the set $\mathcal{S}_{t-1}$ using a proposal density function $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t)$.

- **weight assignment:** each predicted sample $\mathbf{x}_{t|t-1}^i$ is assigned an importance weight w_t^i computed according to

$$w_t^i = w_{t-1}^i . \frac{p(\mathbf{z}_t|\mathbf{x}_{t|t-1}^i).p(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)}.$$

  Once computed, the importance weights are normalized.

- **resampling:** it consists in deleting or duplicating particles according to their weights. This is usually done by generating a new set of particles $\{\mathbf{x}_t^j\}_{j=1}^N$ from an approximate discrete representation of $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ given by

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} w_t^i \delta_{\mathbf{x}_{t|t-1}^i}(\mathbf{x}_t) \tag{3.7}$$

  so that $p(\mathbf{x}_t^j = \mathbf{x}_{t|t-1}^i) = w_t^i$. At the end, each resulting particle $\mathbf{x}_t^j$ is assigned a weight $w_t^j = 1/N$.

An illustration of how a particle filter works is shown in Figure 3.2.

Several resampling strategies have been proposed in the literature. Among them, the *multinomial resampling* strategy [Efron and Tibshirani, 1993] is presented to be the simplest approach [Douc et al., 2005]. The multinomial resampling strategy consists in sampling from a uniform distribution $\mathcal{U}((0, 1])$ N numbers $\{k_i\}_{i=1}^N$. Then each particle $\mathbf{x}_{t|t-1}^i$ is replaced by the particle $\mathbf{x}_t^i = \mathbf{x}_{t|t-1}^{L(k_i)}$ where $L(k_i)$ is the unique integer m such that $\sum_{u=1}^{m-1} w_t^u < k_i \leq \sum_{u=1}^{m} w_t^u$.

While the resampling step reduces the effects of the degeneracy phenomenon, it actually adds an extra noise to the variance of the estimator of the posterior distribution. The *residual resampling* strategy [Liu and Chen, 1998] has been proposed to reduce this additional noise with respect to the multinomial resampling strategy [Douc et al., 2005]. In order to obtain the new set of particles, this strategy proceeds via two steps. First, it duplicates each particle

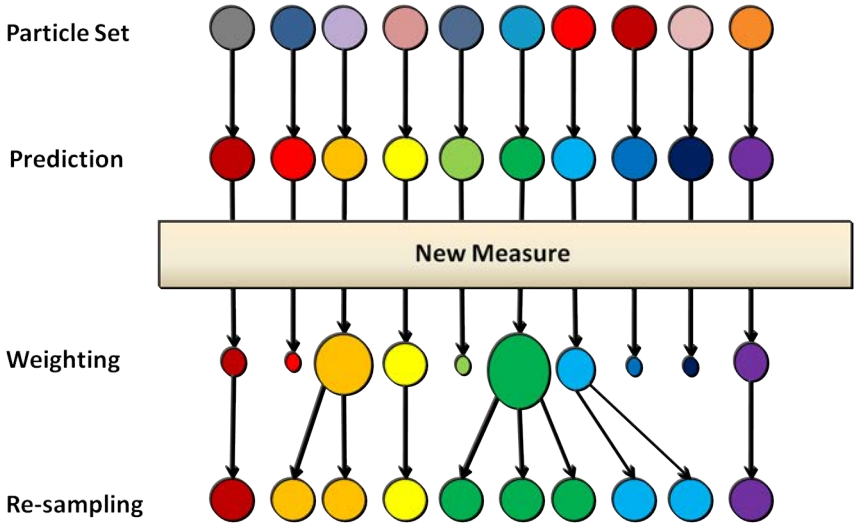


Figure 3.2: Graphical illustration of a Particle filter. After the prediction step, when a new measure is received, the weighting step takes place and is followed by the resampling step which consists in duplicating particles with high weights and deleting particles with low weights.

$\mathbf{x}^i_{t|t-1}$, $i = 1, \cdots, N$, n_i times where $n_i = \lfloor N w^i_t \rfloor$ with $\lfloor . \rfloor$ being the operator which returns the largest integer not greater its argument. Then, the rest of $N - \sum_{i=1}^{N} n_i$ particles, that are still needed for completing to N the size of the new particle set, are randomly drawn from the multinomial distribution $\mathcal{M}(N - \sum_{i=1}^{N} n_i, N w^1_t - n_1, \cdots, N w^N_t - n_N)$.

Because the resampling step does add an extra noise to the posterior distribution's estimator, it is suggested [Arulampalam et al., 2002] to perform this operation only when a significant degeneracy is observed. A criterion named *effective sample size* has been proposed [Kong et al., 1994] to measure the degeneracy of the filter and it is defined by

$$N_{eff_t} = \frac{N}{1 + Var_q(w^*_t)}, \tag{3.8}$$

where w^*_t is the **true weight** of a particle and $Var_q(.)$ means the variance operator that is computed with respect to the proposal density q. Then using a threshold $N_{T_{eff}}$, the resampling is performed whenever $N_{eff_t} < N_{T_{eff}}$. However, since it is difficult to compute the true weight in most problems, one can use another measure $\hat{N}_{eff_t}$ as a good estimate of the effective sample size [Arulampalam et al., 2002] and which is defined by

$$\hat{N}_{eff_t} = \frac{1}{\sum_{i=1}^{N} (w^i_t)^2}. \tag{3.9}$$

The choice of the proposal density $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t)$ plays a crucial role in the efficiency of the filter. Besides the fact that it must me the condition defined by Equation 3.1, two factors have to be taken into account when designing such a density: (1) the ability to easily sample from it, and (2) the ability to evaluate it in order to compute the weight of the different particles. One of the choice that is commonly made is $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$. While this assumption simplifies the computation of the importance weights, its drawback resides in the fact that the proposal density does not take into account the current observation $\mathbf{z}_t$. However, it is usually

sufficient for obtaining good filtering performances [Arulampalam et al., 2002].

A one step pseudo-code of a generic particle filter is described in Algorithm 3.2.

Algorithm 3.2: Generic Particle Filter

Algorithm PF($\{\mathbf{x}_{t-1}^i, w_{t-1}^i\}_{i=1}^N$, $\mathbf{z}_t$)
  for $i = 1 : N$ **do**
    Draw $\mathbf{x}_{t|t-1}^i \sim q(.|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$
    Assign the particle's weight, w_t^i, according to Equation 3.5
  Calculate total weight: $W = \sum_{i=1}^N w_t^i$
  for $i = 1 : N$ **do**
    Normalize: $w_t^i = \frac{w_t^i}{W}$
  Compute $\hat{N}_{eff_t}$ according to Equation 3.9
  if $\hat{N}_{eff_t} < N_{T_{eff}}$ **then**
    $\{\mathbf{x}_t^i\}_{i=1}^N =$ Resample $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)$*
    for $i = 1 : N$ **do**
      $w_t^i = \frac{1}{N}$
  else
    for $i = 1 : N$ **do**
      $\mathbf{x}_t^i = \mathbf{x}_{t|t-1}^i$
  return $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$
(*) The **Procedure** Resample $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)$ performs the resampling operation.

3.5 Regularized Particle Filter

When the underlying system's dynamics function is subject to little noise, then the particle filter, as described above, is not appropriate anymore. This is because duplicating a particle, at the resampling step, will lead to a particle that will evolve quite similarly as the original one, thus leading in the long term to the sample impoverishment phenomenon [Arulampalam et al., 2002]. Regularized particle filters (RPFs) have been introduced [LeGland et al., 1998, Musso et al., 2001] with the aim to prevent this phenomenon.

The main idea of RPFs is to resample from a continuous approximation of the probability density function $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ instead of its discrete approximation (see Equation 3.7), hence producing a new set of particles with N different particles. The continuous approximation of the posterior distribution is usually computed as follows (see Figure 3.3):

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \hat{p}_\lambda(\mathbf{x}_t|\mathbf{z}_{1:t}) = \sum_{i=1}^{N} w_t^i . K_\lambda(\mathbf{x}_t - \mathbf{x}_t^i), \tag{3.10}$$

where $K_\lambda(\mathbf{x}) = \frac{1}{\lambda^{n_x}} K(\frac{\mathbf{x}}{\lambda})$ is the rescaled kernel density, $K(.)$ is the considered kernel function, λ is the kernel bandwidth and n_x is the dimension of the state space. A kernel K could be any symmetric density function such that $K > 0$, $\int K(\mathbf{x})d\mathbf{x} = 1$, $\int \mathbf{x}K(\mathbf{x})d\mathbf{x} = 0$ and

$\int ||\mathbf{x}||^2 K(\mathbf{x}) d\mathbf{x} < \infty$. Examples of such a kernel include Gaussian kernel, Epanechnikov kernel and cosine kernel. Usually, the covariance matrix of the underlying samples is often used as additional information when designing the kernel. For example, this matrix can be used to define the covariance matrix of a Gaussian kernel.

In the field of density estimation, the "mean integrated squared error" (MISE), which is defined as

$$E||p_n - p||_2^2 = \int (p_n(x) - p(x))^2 dx,$$

is usually used to evaluate the quality of an estimate $p_n(.)$, obtained on the basis of n independent samples, with respect to the unknown density $p(.)$. Focusing on the RPF approach, it has been shown that, when the kernel is Gaussian, the optimal bandwidth that minimizes the MISE is given [Silverman and Green, 1986] by

$$\lambda_{opt} = \left(\frac{4}{(n_x + 2)N} \right)^{\frac{1}{n_x+4}}. \tag{3.11}$$

In practice, when the densities are multi-modal, it is suggested to choose λ as $\lambda_{opt}/2$ [Silverman and Green, 1986]. The pseudo-code describing the RPF is provided in Algorithm 3.3.

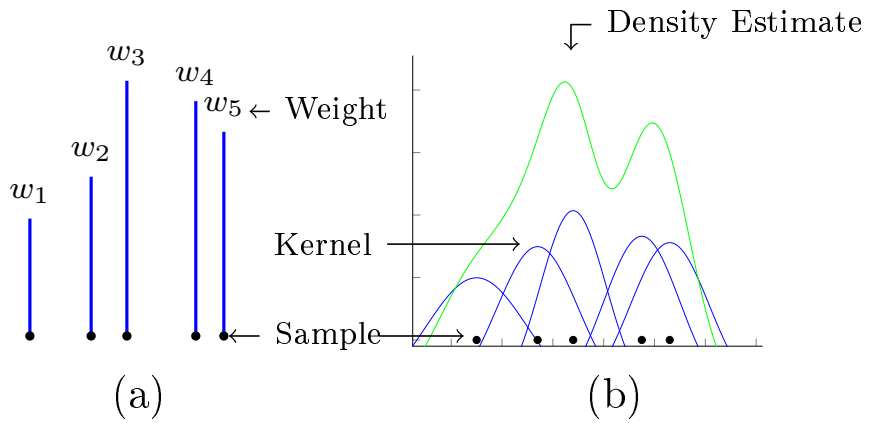


Figure 3.3: Regularization process: (a) Weighted empirical measure; (b) Regularized measure. The new samples will be drawn from the green distribution represented in (b) instead of the discrete distribution obtained in (a).

3.6 Conclusion

In this chapter, we described Monte Carlo solutions to the filtering problem for the general case of nonlinear systems. Specifically, we introduced the generic particle filter algorithm as an approximation of the Bayesian filter in which the posterior distribution or belief over the system's state is represented by a finite set of weighted particles. The particle filter does not make any assumptions regarding the system's dynamics and therefore it is recommended for strongly nonlinear systems. In the remainder of this document, the concepts introduced in this chapter will be intensively used for addressing the specific problem of pedestrian behavioral tracking.

Algorithm 3.3: Regularized Particle Filter

Algorithm `RPF`$(\{\mathbf{x}_{t-1}^i, w_{t-1}^i\}_{i=1}^N, \mathbf{z}_t)$
  for $i = 1 : N$ **do**
    Draw $\mathbf{x}_{t|t-1}^i \sim q(.|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$
    Assign the particle's weight, w_t^i, according to Equation 3.5
  Calculate total weight: $W = \sum_{i=1}^N w_t^i$
  for $i = 1 : N$ **do**
    Normalize: $w_k^i = \frac{w_t^i}{W}$
  Compute $\hat{N}_{eff_t}$ according to Equation 3.9
  if $\hat{N}_{eff_t} < N_{T_{eff}}$ **then**
    Compute the empirical covariance matrix S_t of $\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N$
    Compute D_t such that $D_t \times D_t^T = S_t$
    $\{\mathbf{x}_t^i\}_{i=1}^N =$ `Resample` $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)^*$
    for $i = 1 : N$ **do**
      Draw $\epsilon^i \sim K$ (from the kernel)
      $\mathbf{x}_t^i = \mathbf{x}_t^i + \lambda D_t \epsilon^i$
      $w_t^i = \frac{1}{N}$
  else
    for $i = 1 : N$ **do**
      $\mathbf{x}_t^i = \mathbf{x}_{t|t-1}^i$
  return $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$
(*) The **Procedure** `Resample` $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)$ performs the resampling operation.

Part II

Single-Target Behavioral Tracking

Chapter 4

Algorithms for estimating pedestrian's behavior

Contents

4.1 Introduction

Estimating pedestrian's behaviors using a sensor network is of great importance in several application domains including, among others, surveillance systems, patient monitoring systems, smart homes, elderly care assistance, and systems involving human interactions with electronic devices. For instance, in a public place like an airport or a subway station, it is interesting for an automated surveillance system to detect abnormal or suspicious passengers' behaviors. These abnormal behaviors may be of different kinds ranging, for example, from "cheating on a ticket barrier" to "setting a trash can on fire" or "abandoning a luggage". Another example in which one can see the utility of estimating pedestrian behaviors takes place in an elderly care center. In such a center, a monitoring system must be able to detect the fall of an elderly patient and automatically raise an alarm to inform the nurses.

Generally speaking, the problem of estimating the behavior of a single pedestrian can be viewed as a filtering problem as described in Section 2.1. In this case, the system under investigation is the considered pedestrian within his environment and the only and possibly noisy information available from the system are data collected by the sensors used. Two principal behavior's aspects have been associated with the notion of "pedestrian behavior" in the literature [Helbing and Molnár, 1995, Ke et al., 2007, Aggarwal and Ryoo, 2011]:

- **The motion**. The first one focuses on the pedestrian motion and deals with the trajectory followed by the concerned target.
- **The activity**. The second aspect is related to the activity, in the general sense, performed by the pedestrian within his environment.

Depending on which aspects they are focusing on, approaches developed for tackling the problem of pedestrian behavior understanding can be roughly classified into three categories:

- **Motion tracking**. This category regroups methods dedicated to the motion tracking or localization problem. Such methods, in order to work, require as input a predictive model of the pedestrian motion. Several motion models exist in the literature ranging from simple models such as *the constant velocity model* [Fod et al., 2002] to more realistic models which are able to deal with dynamic obstacle avoidance such as *the social force model* [Helbing and Molnár, 1995].
- **Activity recognition**. This category concerns approaches designed to deal with the activity recognition problem independently of the target location within the environment. Activities can be classified into low-level activities (e.g., gestures, atomic actions) and high-level activities (e.g., human-object interactions) [Aggarwal and Ryoo, 2011]. When focusing on recognizing high-level activities, context-based knowledge is usually provided to express how these activities are related to low-level ones. Depending on how this knowledge is expressed, we can distinguish graphical model based approaches, syntactic approaches and description-based approaches.
- **Simultaneous tracking and activity recognition**. This category contains methods which simultaneously address both the problems of motion tracking and activity recognition. Based on the idea that moving pedestrians are driven by an inner motivation in

relation with the activity they are performing in the environment, these methods try to exploit the intrinsic dependency between location and activity to improve the estimate of the location from the knowledge they have regarding the activity and conversely.

This chapter aims at presenting a non-exhaustive review of the approaches belonging to each of the above mentioned categories and it is organized as follows: motion tracking and activity recognition solutions are described in Section 4.2 and Section 4.3 respectively while solutions belonging to the third category are described in Section 4.4.

4.2 Motion Tracking

In this section, we are interested in methods that have been developed to address the problem of pedestrian behavior estimation from the sole point of view of trajectory tracking.

Generally, in methods designed for motion tracking, the considered pedestrian's behavior is represented by physical attributes in relation with his motion such as the position, the velocity, and/or the acceleration. Moreover, a predictive pedestrian motion model is required as an input to the inference process. In the following sections, we will discuss motion models found in the literature and associated works in which they were used for tracking purposes.

4.2.1 Simple motion models

In the literature, some works [Fod et al., 2002, Cui et al., 2005, Arras et al., 2008] make simple assumptions regarding the human motion and consider that it can be represented by a constant velocity model modulated by some noise. While this assumption simplifies the computation of the estimate of the target location via the use of Kalman filters [Kalman, 1960] (see Section 2.3), it does not often reflect the reality. Indeed, constant velocity models consider that a given pedestrian will continue to move in the direction in which he was last observed (see Figure 4.1). This is, of course, not always the case as, for example, in presence of obstacles.

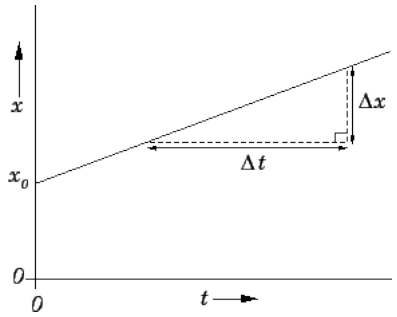


Figure 4.1: Constant velocity model: displacement versus time of a body moving with constant velocity in one dimension. $\mathbf{x}_0$ represents the initial position of the body at time $t = 0$ and the path followed by the body over time is linear.

4.2.2 Motion models handling static obstacles

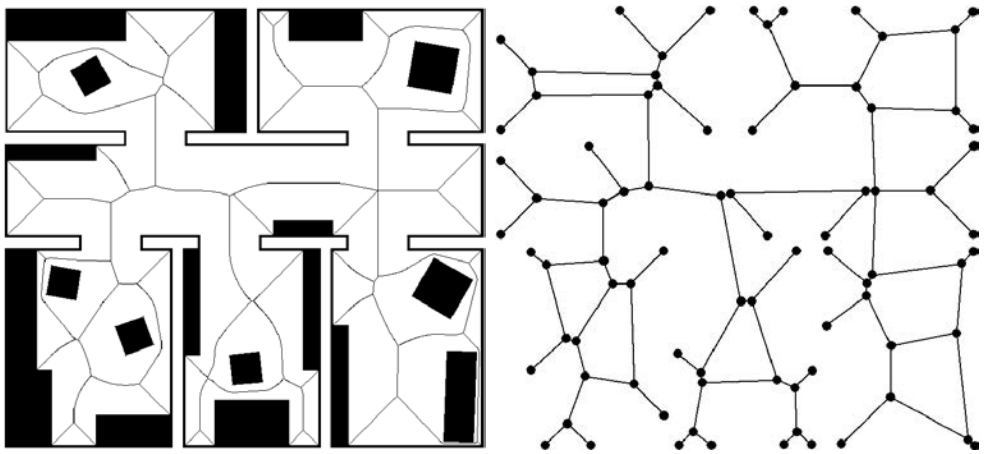
In order to deal with obstacles, a general approach consists in taking into account the environment while performing motion path planning to the destination point targeted by the pedestrian. The

most popular method for performing path planning is the well known *A* Algorithm* [Hart et al., 1968] which allows to compute a least-cost path from a given initial point to another one in the environment while avoiding obstacles.

In [Liao et al., 2003], the authors proposed a Voronoi graph based motion model which constrains the location of moving targets to lie on edges of a Voronoi graph extracted from a map of the environment. The Voronoi graph (see Figure 4.2) provides a natural discretization of the environment on which the authors apply unsupervised learning techniques to derive typical motion patterns of people walking through the environment. Their model is then combined with a particle filter for tracking purposes.

Bennewitz et al. [2005] proposed a model based on groups of trajectories represented as closely-spaced sequences of waypoints. Their model represents people's motion patterns learned from a collection of trajectories, and is used to classify to which group a tracked trajectory belongs; thus enabling to predict an individual's motion. More specifically, from each pattern, a hidden Markov model is derived and used as the baseline for motion prediction.

With respect to constant velocity models, both path planning and Voronoi motion model have the advantage of integrating the topology of the environment. However, as pointed out in [Tastan and Sukthankar, 2011], these methods are not based on actual human profiles and cannot be easily generalized. Moreover, these models as well as the one proposed by Bennewitz et al. do not take into account dynamic obstacles (e.g., presence of other pedestrians [4]) in the surrounding of the considered pedestrian.



(a) (b)

Figure 4.2: Illustration of a generalized Voronoi diagram. The discretization of the environment into a Voronoi diagram represented by a set of cells of various shapes (Fig 4.2a). In Fig 4.2b, the generalized Voronoi graph corresponding to the Voronoi diagram in Fig 4.2a is depicted. From [Wallgrün, 2005].

[4] Even if, in this section, we are focusing on the problem of inferring a single pedestrian behavior, we found interesting to discuss generic motion models whose dynamic evolution may consider the presence of other targets in the environment.

4.2.3 Motion models handling dynamic obstacles

Analytical microscopic motion models capable of handling dynamic obstacles have been proposed in the literature [Henderson, 1974, Helbing, 1992]. However, their numerical solution is very complex and simulations are usually preferred. Most microscopic pedestrian simulation models (MPSMs), instead of performing global path planning, rely on the idea that a walker makes a decision regarding the path to follow based on local features. These features, which include the effects of static obstacles as well as those of neighboring pedestrians, are represented by a set of attractive and repulsive fields which define the characteristics of the motion observed such as the velocity, the acceleration, or the orientation. Depending on the underlying model used for the computation, we may regroup the MPSMs into two categories: *cellular based* and *physics based* models. The cellular based models discretize the environment into a regular grid from which each cell can be occupied by at most one pedestrian. Then, computations regarding a given pedestrian dynamics are performed based on a set of rules depending on the properties of neighboring cells. Models belonging to this category are the *benefit cost cellular model* [Gipps and Marksjo, 1985, Teknomo, 2002] and the *cellular automata model* [Yang et al., 2003, Blue and Adler, 2001]. These models present the advantages to be computationally simple. However, the discrete nature of the environment gives the impression that pedestrians are jumping from one cell to another, thus leading to non realistic movements. On the other hand, physics based models do not discretize the environment and, instead, rely on physical forces for explaining motion patterns observed from a pedestrian's movement. Models falling within this category include the *magnetic force model* [Okazakia and Matsushitaa, 1993] and the *social force model* [Helbing and Molnár, 1995, Arkin, 1998, Reynolds, 1999, Lakoba et al., 2005].

In the following sections, we will first describe the above mentioned motion models and then we will present an overview of works in the literature based on these models for pedestrian tracking purposes.

4.2.3.1 Cellular-based motion models

Benefit Cost Cellular Motion Model

This model, proposed by Gipps and Marksjo [1985], subdivides the environment into a regular square grid of dimension such that at most one pedestrian can occupy a cell (e.g., $0.5 \times 0.5m^2$ per cell). The model hypothesizes the existence of repulsive forces between pedestrians so that, as the subject approaches another pedestrian, the "potential energy" of his position rises and the "kinetic energy" of his speed drops while deflecting him from a straight line. A *cost score*, representing the repulsive effect of close pedestrians, is assigned to each cell based on the proximity to other pedestrians in the environment. The cost score C_i of a cell i generated by a pedestrian — located in a cell k — moving to that cell i is approximately inversely proportional to the square of the separation of the two cells and it is computed as

$$C_i = \frac{1}{(\Delta - \alpha)^2 + \beta}, \text{ where} \tag{4.1}$$

- Δ is the distance between cell i and the pedestrian in cell k,

- $\alpha = 0.4$, is a constant slightly below the size of the cell (0.5m),
- $\beta = 0.015$, is a constant number used to moderate changes in score for cells close to the pedestrian.

When the fields[5] of several pedestrians overlap, the score in each cell is the sum of the score generated by each individual. The cost score is balanced against a *gain score* made by moving towards the pedestrian's destination point. The gain score depends on the angle of deviation from the desired path rather than the distance of the considered cell from the destination and it is computed as

$$\begin{aligned} P(\sigma_i) &= K\cos(\sigma_i)|\cos(\sigma_i)|, \\ &= \frac{K(X_i - S)(D - S)|(X_i - S)||(D - S)|}{|(X_i - S)|^2|(D - S)|^2}, \text{ where} \end{aligned} \tag{4.2}$$

- $P(\sigma_i)$ is the gain score for moving to cell i with respect to the destination point and it is equal to zero when the pedestrian does not move and remains static,
- σ_i is the angle of deviation from the straight line to the immediate destination when moving to cell i,
- K is a constant of proportionality to enable the gain of moving in straight line to be balanced against the costs of approaching other pedestrians too closely,
- X_i is the location coordinates of cell i,
- S is the location coordinates of the target cell (cell containing the considered pedestrian),
- D is the location coordinates of the destination point.

The result of the balancing, also known as the *net benefit* is computed for all the nine neighbor cells (including the actual location of the pedestrian) as

$$B_i = P(\sigma_i) - C_i. \tag{4.3}$$

Finally, the pedestrian moves to the cell having the maximum net benefit among his surrounding.

The principal advantage of this model resides in its simplicity. However, the scoring system is arbitrary and does not have any physical meaning, therefore making the model difficult to be used for explaining real-world phenomena.

Cellular Automata Motion Model

As for the benefit cost cellular motion model, the cellular automata motion model subdivides the environment into a regular grid. The grid structure can form different patterns depending on

[5] The field of a pedestrian is composed of neighboring cells surrounding the cell he currently occupies.

the problem at hand. The basic forms are triangular grid, rectangular grid and hexagonal grid. Each cell is characterized by an internal state. The state of a cellular automaton is completely specified by the values of the states at each cell. A cellular automaton evolves in discrete time steps, with the value of the state at one cell being affected by the values of states at cells in its neighborhood on the previous time step. A set of *local rules* [Wolfram, 1986] is used to update the overall state of the cellular automaton at each time step.

In [Blue and Adler, 2001], the authors focus on designing a pedestrian motion model for a bi-directional walkways using cellular automata. The state of each cell is either occupied or unoccupied and at most one pedestrian may be within a cell. They consider three fundamental elements of pedestrian movements: lane changing, forward movement (breaking, acceleration) and conflict mitigation (deadlock avoidance). They designed a set of rules based on these three elements. Cells are updated in parallel in two stages. In the first update stage, the next lane of each pedestrian is computed based on current conditions. The lane that best promotes forward movements is chosen from the neighborhood set consisting of left, same, and right lanes. Once the lanes of all pedestrian are computed, pedestrians are moved to new cells. In the second update stage, the speed of each pedestrian is computed based on his desired speed and the available gap ahead (number of unoccupied cells). A well known cellular automata motion model in which the transition probability of cells are not fixed but vary dynamically is the *floor field model* [Burstedde et al., 2001].

Like the benefit cost cellular motion model, the cellular automata motion model is also computationally simple to design. However, the discrete nature of the automata leads to non realistic motion movements.

4.2.3.2 Physics-based motion models

Magnetic Force Model

The magnetic force model has been introduced in [Okazakia and Matsushitaa, 1993]. It mainly relies on the convention from the magnetic field theory stating that *"two charges of the same sign, e.g., two positive charges, repel each other, while two charges with opposite sign attract each other"*. This phenomenon is called *the electromagnetic interaction*. Based on this fact, pedestrians as well as obstacles are represented by *positive poles* while *negative poles* are assumed located at the destination points of each pedestrian. Therefore, pedestrians are "attracted" by their destination point while avoiding at the same time obstacles and other pedestrians. Using Coulomb's laws, magnetic forces are computed on each pedestrian as the effect of other magnetic poles present in the environment. Considering two magnetic pole (one being the considered pedestrian), the mutual magnetic force is obtained as

$$\mathbf{F} = \frac{kq_1q_2}{r^3}\mathbf{r}, \text{ where} \tag{4.4}$$

- $\mathbf{F}$ is the vector representing the magnetic force,
- k is the Coulomb's constant,
- q_1 is the intensity of the magnetic pole representing the considered pedestrian,

- q_2 is the intensity of the second magnetic pole,
- $\mathbf{r}$ is the vector from the pedestrian to the magnetic pole, and
- r is the distance between the pedestrian and the magnetic pole (length of $\mathbf{r}$).

Another force acts on a pedestrian to avoid the collision with another pedestrian. In the example of Figure 4.3 where Pedestrian A tries to avoid the collision with Pedestrian B, such a force exerts acceleration $\mathbf{a}$ on Pedestrian A in such a way to modify the direction of his relative velocity RV (with respect to pedestrian B) to the direction of line AC where the latter represents a contacting line from the position of Pedestrian A to the circle around Pedestrian B (the comfort area of the pedestrian). The norm of acceleration $\mathbf{a}$ is calculated using the following equation

$$||\mathbf{a}|| = ||\mathbf{V_A}||.\cos(\alpha).\tan(\beta), \text{ where} \tag{4.5}$$

- $\mathbf{a}$ is the acceleration applied to modify the direction of Pedestrian A in order to avoid Pedestrian B,
- $\mathbf{V_A}$ is the velocity of Pedestrian A,
- α is the angle between $\mathbf{RV}$, the relative velocity of Pedestrian A to Pedestrian B, and $\mathbf{V_A}$,
- β is the angle between $\mathbf{RV}$, the relative velocity of Pedestrian A to Pedestrian B, and AC.

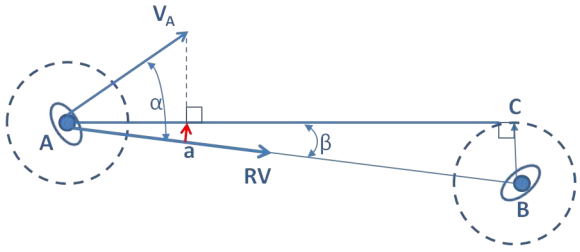


Figure 4.3: Collision Avoidance in Magnetic Force Model: Additional force $\mathbf{a}$, represented in red arrow, acts on Pedestrian A to avoid the collision with Pedestrian B.

At the end, the resulting force affecting a given pedestrian consists in a sum of all the forces generated from other magnetic poles and it decides of the velocity of the pedestrian. In complex plans from which it is not trivial for pedestrians to reach their destination point, it is sometimes useful to artificially define special points on the walls (also referred to as *corner points*) as temporary goals which lead them to their final destination.

Social Force Model

The social force model uses the concept of *social forces* to represent different factors that affect the movement of a pedestrian. It was first introduced by Helbing et al. [Helbing, 1991, Helbing and Molnár, 1995, Helbing and Vicsek, 1999]. In this model, a pedestrian is assumed subjected

to forces due to the influence of the environment (static obstacles) and the presence of other people. The resultant of all these forces acts upon the considered pedestrian by modifying the motion direction together with the speed when necessary. The model relies on the hypothesis that each pedestrian is willing to reach a certain destination point at a certain target time.

More formally, given Pedestrian i with mass m_i, he may wish to move comfortably with a certain *intended speed* s_i in an *intended direction* characterized by a unit vector $\mathbf{e}_i$. Assuming that his actual velocity is $\mathbf{v}_i$, he will therefore adapt his velocity accordingly during a given amount of time τ_i also referred to as *relaxation time*. This change of velocity is modeled by the personal force $\mathbf{F}_i^{pers}$ computed as

$$\mathbf{F}_i^{pers} = m_i . \frac{s_i \mathbf{e}_i - \mathbf{v}_i}{\tau_i}. \tag{4.6}$$

In presence of obstacles or other people, a given pedestrian may want to avoid them and thus, he might not be able to preserve the intended speed and direction. These repulsive effects from other entities are represented by global force $\mathbf{F}_i^{soc}$ obtained by a summation of individual forces generated by all the considered entities. This is modeled by

$$\mathbf{F}_i^{soc} = \sum_{j \in \mathcal{P} \setminus \{i\}} \mathbf{F}_{i,j}^{soc} + \sum_{o \in \mathcal{O}} \mathbf{F}_{i,o}^{soc}, \tag{4.7}$$

where $\mathcal{P}$ and $\mathcal{O}$ are respectively the set of all pedestrians and the set of all static obstacles in the environment. The effects of these individual forces decrease proportionally to the distance of their sources and they are represented by

$$\mathbf{F}_{i,k}^{soc} = a_k e^{\frac{r_{i,k} - d_{i,k}}{b_k}} \mathbf{n}_{i,k}, \text{ where} \tag{4.8}$$

- k represents either a person or an obstacle,
- a_k and b_k respectively represent the magnitude and the range of the force,
- $d_{i,k}$ is the distance between the centers of mass of Pedestrian i and Entity k (shortest distance to the closest face in case of obstacle),
- $r_{i,k}$ is the sum of the diameter of Pedestrian i and the diameter of Entity k,
- $\mathbf{n}_{i,k}$ is the normalized vector pointing from Entity k to Pedestrian i.

Because of the limited field of view of humans, influences from other entities might not be isotropic. The effect of entities located behind the considered pedestrian will have a weaker influence. Thus, in order to take into account the effect of perception (it is assumed that the perception or the sight of an individual is characterized by an effective angle of 2φ ahead), a direction-dependent weight $w(i,k)$ is introduced to modulate each force and it is defined as

$$w(i,k) = \begin{cases} 1 & \text{if } \mathbf{e}_i . \mathbf{F}_{i,k}^{soc} \geq \|\mathbf{F}_{i,k}^{soc}\| . \cos\varphi, \\ c & \text{Otherwise } (c \text{ is a constant s.t. } 0 < c < 1). \end{cases} \tag{4.9}$$

This leads to a new expression of $\mathbf{F}_{i,k}^{soc}$ in which these weights are integrated:

$$\mathbf{F}_i^{soc} = \sum_{j \in \mathcal{P} \setminus \{i\}} w(i,j)\mathbf{F}_{i,j}^{soc} + \sum_{o \in \mathcal{O}} w(i,o)\mathbf{F}_{i,o}^{soc}. \tag{4.10}$$

Finally, using Equations 4.6 and 4.10, the global force exerted on Pedestrian i is then

$$\mathbf{F}_i = \mathbf{F}_i^{pers} + \mathbf{F}_i^{soc}. \tag{4.11}$$

Using $\mathbf{F}_i$, the Pedestrian i's motion can be modeled using the following basic equation

$$\frac{d}{dt}\mathbf{v}_i = \frac{\mathbf{F}_i}{m_i} + fluctuations, \tag{4.12}$$

where the *fluctuation term* is added to take into account random variations of the motion behavior. An illustration of all the forces is shown in Figure 4.4.

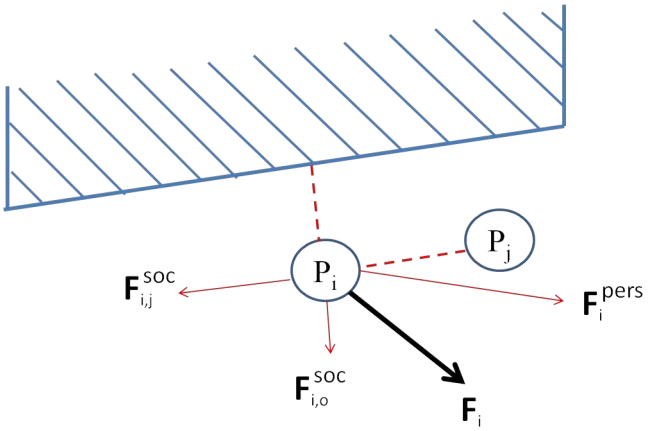


Figure 4.4: Illustration of the Social Force Model. Pedestrian P_i is the one of interest. Red arrows represent forces from the wall ($\mathbf{F}_{i,o}^{soc}$) and from the Pedestrian P_j ($\mathbf{F}_{i,j}^{soc}$) as well as the personal force towards intended direction ($\mathbf{F}_i^{pers}$). The resulting force ($\mathbf{F}_i$) is represented in black.

4.2.4 Overview of pedestrian tracking related works

This section presents a non exhaustive overview of pedestrian tracking works in the literature that used cellular-based and physics-based motion models as baseline of their filtering process.

In [Suppitaksakul et al., 2006], the authors describe a pedestrian tracking system that uses a movement prediction technique associating a cellular automaton and a neural network. While the cellular automaton captures the pedestrian movement, the derived patterns are used to train the neural network for future position estimation. The effectiveness of the technique depends on the following factors: the speed of pedestrians and the movement patterns.

Ali and Shah [2008] are interested in tracking pedestrians in emergency situations. To achieve their goal, they combine a floor field approach [Burstedde et al., 2001] with a dedicated evacuation model of the environment under consideration. However, their approach suffers from the discrete nature of the motion model in achieving good performances.

In their work, Luber et al. [2010] combine the social force model proposed in [Helbing and Molnár, 1995] with a generic Bayesian tracker in order to predict realistic human motions. They consider forces from the environment by maintaining a short-term environment model and computing different repulsion forces and physical constraints from static obstacles. They claimed to be able to deal efficiently with occlusion events of two persons approaching each other as the motion models prevent them to collide together.

Pellegrini et al. [2009] propose a motion model inspired from the social force model for pedestrian trajectory prediction in the context of visual tracking. Their model is designed for walkers with short-term prediction in mind for obstacle avoidance. In their approach, the velocity of a given pedestrian is accounted for in his energy potential. This is done by estimating the closest future distance in his space-time trajectory which is then used as an additional potential (force); thus enabling the system to plan ahead to some extent. The model is trained with videos recorded from birds-eye view at busy locations, and applied as a motion model for tracking from a vehicle-mounted camera.

In the same line, Leal-Taixé et al. [2011] combine the social force model with a global optimization scheme to obtain a robust tracker able to work in crowded scenarios. They claim that their approach achieves good performances in terms of recovering pedestrian tracks after occlusion events.

Finally, Tastan and Sukthankar [2011] show how prior knowledge of human perception and locomotion limitations can be leveraged to enhance path prediction and tracking in indoor environments for pervasive computing applications. They demonstrate an approach for path prediction based on a model of visually guided steering proposed in [Fajen et al., 2003] and that has been validated on human obstacle avoidance data. They claim that their approach outperforms standard motion models in a particle filter tracker during occlusion periods of greater than one second and results in a significant reduction of the tracking error (sum-of-square differences).

Having realistic motion models is important in pedestrian tracking, however, focusing solely on pedestrian motion models does not provide any asset for reasoning on people intent. Identifying the activities performed by a pedestrian plays an important role in inferring his behavior.

4.3 Activity Recognition

This section focuses on methods that have been developed in the literature for addressing the pedestrian behavior estimation problem from the sole point of view of activity recognition. The following presentation is mainly inspired from the literature review made by Aggarwal and Ryoo [2011].

The idea behind human activity recognition is to automatically identify activities that are undertaken by one or more humans (also referred to as agents or targets) from a sequence of observations. Depending on their complexity, human activities can be classified into different levels of granularity. We may distinguish, in a hierarchical order, *gestures* (e.g., raising a leg, turning the face), *actions* (e.g., walking, facing right) and *interactions*. The interactions can be further subdivided into two categories depending on the entities involved. These are *human-human interactions* (e.g., fighting, queuing) and *human-object interactions* (e.g., buying a ticket, crossing a barrier). These different levels are clearly not independent and, the higher the level, the more contextual information is needed. Also, when reasoning at a group scale, the term

group activity is sometimes used to refer to activity performed by a group composed of multiple pedestrians (e.g., having a meeting). From now in this section, we will use the concept of "low-level activities" to refer to activities limited to gestures and actions while by "high-level activities", we will refer to other types of activities (interactions). Although the remainder of this section essentially addresses approaches designed for recognizing high-level activities, the following section presents a brief overview of literature regarding the activity recognition problem and discusses works emphasizing on low-level activities.

4.3.1 Overview

In the literature, methodologies developed for the problem of activity recognition can be organized into two main categories depending on whether they are designed to identify low-level or high-level activities. The following sections are dedicated to each of these categories.

4.3.1.1 Low-level activities

Methodologies for recognizing low-level activities usually represent and recognize the activities directly from the sequence of received observation data. Depending whether they treat received data as a block or on a sequential basis, low-level activity methodologies can further be subdivided into two groups: **space-time approaches** and **sequential approaches**.

In space-time approaches [Campbell and Bobick, 1995, Chomat et al., 2000, Bobick and Davis, 2001, Sheikh et al., 2005, Ke et al., 2007], the observation data, let us say a video input, is viewed like a collection of 2-D images in chronological order and is processed as a particular 3-D XYT space-time volumetric data obtained by concatenating the 2-D images (XY) along time (T). From the volumetric data, features are extracted and compared to features previously extracted from training videos representing each activity the system wishes to identify. Finally, the activity with the highest similarity with respect to the extracted features is defined as the one the input video corresponds to. An illustration of a volumetric representation of a sequence of image is shown in Figure 4.5.

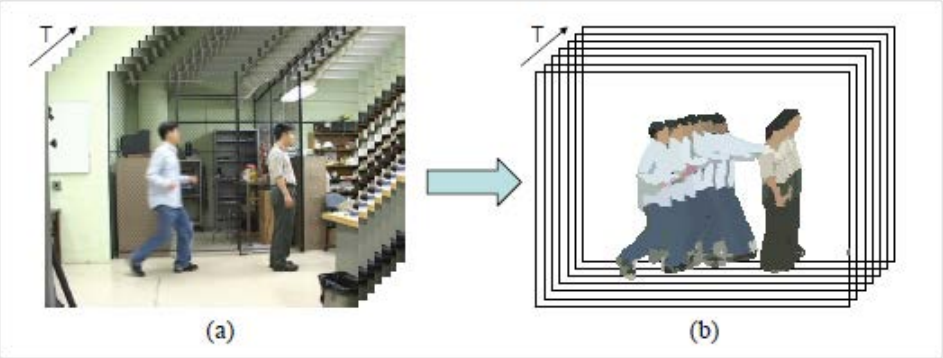


Figure 4.5: Volumetric representation of a sequence of images: (a) entire images and (b) features (foreground blob images) from a "punching" sequence. From [Aggarwal and Ryoo, 2011].

On the other hand, sequential approaches consider the input video as a sequence of frames. From each frame, a set of features (a feature vector) is extracted. Then, in order to identify a given activity, sequential approaches look after the *activity class* (a specific sequence characterizing

the activity) within the set of extracted features. Most of the time, models are imposed to the temporal dynamics of the extracted features. Generally, one model is designed for each activity and is statistically trained to generate a sequence of feature vectors representing the related activity class. A typical example of such a model is the hidden Markov models (HMMs). Figure 4.6 shows an example of an HMM for the action "stretching an arm". Once trained, these models are then used for calculating the probability for a given input sequence to be generated by the activity associated to them. This is done by recursively computing the likelihood (the posterior probability) of the features extracted with respect to the activity class. Some of the works falling in this category are [Yamato et al., 1992, Bobick and Wilson, 1997, Schlenzig et al., 1994, Dubuisson et al., 2012].

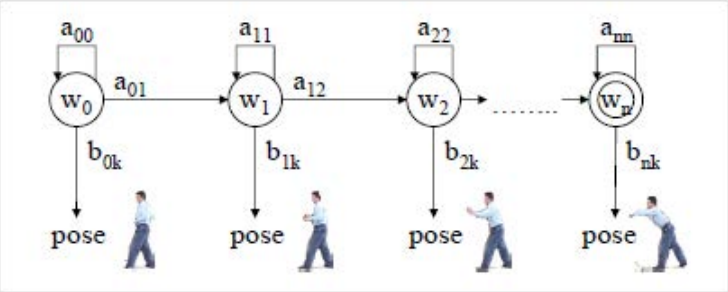


Figure 4.6: An example of HMM for the action "stretching an arm". Each actor image represents a pose with observation probability b_{jk} for its state w_j. From [Aggarwal and Ryoo, 2011].

For instance, Dubuisson et al. [2012] are interested in estimating the movements performed by articulated objects such as the human body. They use a dynamic Bayesian network (DBN) to represent the dependencies among different parts of the considered object (see Figure 4.7). A DBN is an extension of a HMM, composed of multiple conditionally independent hidden nodes that generate observations at each time frame directly or indirectly. Subsequently, they consider the use of Bayesian filter (especially, particle filter) for estimation purposes and they design an approach — called *combinatorial resampling* — for efficiently performing, at low computational cost, the resampling stage within the corresponding high dimensional state space while exploiting the structure of the underlying DBN.

4.3.1.2 High-level activities

Methodologies for recognizing high-level activities usually represent the high-level activities in terms of other simple activities. The motivation is to let the simpler sub-activities (also referred to as sub-events) to be recognized first, and then use them for the recognition of higher-level activities. For example, the activity of "fighting" can be recognized by successively detecting a sequence of several "punching" and "kicking" interactions. Each sub-event (e.g., punching) can further continuously be decomposed into other sub-events until it is not anymore possible to perform a decomposition. Thus, the architecture of such methodologies are usually composed of multiple layers in a hierarchical setup with the lowest layer being devoted to the identification of low-level activities and the highest layer being used for the inference of the corresponding high-level activities. In order for these hierarchical methods to work properly, intermediate layers have to leverage the context-based knowledge regarding the relationship between the low-level activities identified in the bottom layer and the activities they aim to identify. Depending on how this knowledge is provided, these methodologies can be organized into three groups:

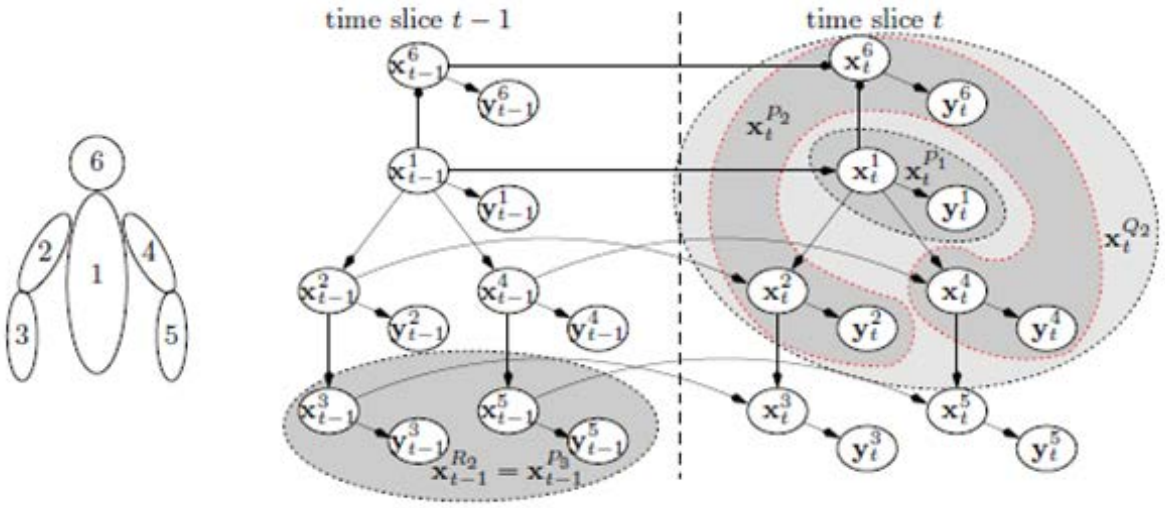


Figure 4.7: DBN representation of human-body dependencies. On the left, a human body: part 1 corresponds to the torso, parts 2 and 3 to the left arm, parts 4 and 5 to the right arm and part 6 to the head. On the right, the corresponding DBN: to the i^{th} part corresponds a pair of state and observation variables (x_t^i,y_t^i). The arrows show the dependencies between variables, including between different time slices. From [Dubuisson et al., 2012].

- graphical model based approaches,
- syntactic approaches,
- description-based approaches.

In what follows, graphical model based approaches are addressed in Section 4.3.2 while syntactic approaches and description-based approaches are respectively discussed in Sections 4.3.3 and 4.3.4.

4.3.2 Graphical model based approaches

Methodologies belonging to this category use probabilistic graphical models to represent the context-based knowledge of the activities they aim to identify. HMMs and DBNs are examples of such graphical models. Both fall within the general category of Bayesian Networks. A Bayesian network (BN) [Pearl, 1988] is a graphical model that represents a set of random variables and their conditional dependencies (causality effects) via a directed acyclic graph with local conditional probability densities (CPDs). However, a generic DBN is able to encode more complex dependence relationships than an HMM. One of the most fundamental hierarchical form under this category is the multi-layered HMMs [Oliver et al., 2002, Nguyen et al., 2005]. In this approach, the bottom HMM is used to recognize atomic actions which are then used as observations for the HMM that is one level higher and so on. An example of a two-layer HMM is depicted in Figure 4.8. Works in which classical DBNs are used to encode domain knowledge can be found in [Intille and Bobick, 1999, Park and Aggarwal, 2003, Damen and Hogg, 2009].

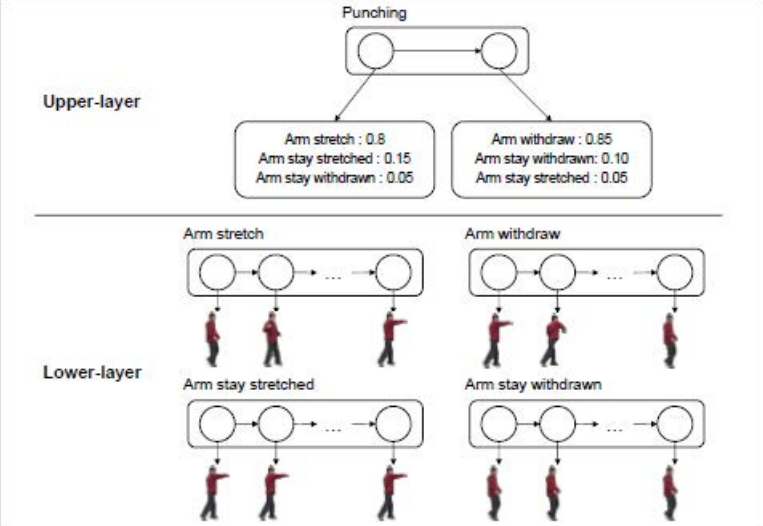


Figure 4.8: An example of a two-layers HMM for recognizing an activity "punching". The bottom layer HMM is used to recognize atomic actions like "stretching" and "withdrawing". The upper HMM used recognition results of the lower layer HMM as observations for recognizing the "punching" activity. From [Aggarwal and Ryoo, 2011].

Besides complex human actions, DBNs have also been used to improve the recognition of human-object interactions [Moore et al., 1999, Gupta and Davis, 2007, Ryoo and Aggarwal, 2007]. In this context, they are designed to express relationships and dependencies between the objects and the human activities by representing, for instance, the way a human interacts with a given object. Therefore, the recognition of objects can benefit the activity recognition while the activity recognition may help identifying objects. For example, with an object called "ATM", one should expect the interaction "withdrawing money" while with the object "ticket machine", one should expect the interaction "Buying tickets". The knowledge of this information suggests a different treatment of the object "ATM" with respect to the object "ticket machine" although they may have the same rectangular form. In [Moore et al., 1999], the authors conduct human-object interaction recognition using DBNs based on scene context derived from other objects in the scene. Generally the structure of the DBNs is provided by domain experts. Due to their inherent Markovian property, DBNs can only be used to model sequential activities. However, in case of large networks, CPDs learning or expert hand-tunings can be extremely difficult to perform.

Other types of probabilistic graphical models, different from DBNs, have been used in the literature. These include, amongst other, propagation nets ($P-net$) [Shi et al., 2004, 2006] and Petri nets [Castel et al., 1996, Ghanem, 2007, Albanese et al., 2008]. The structure of a $P-net$ is similar to that of an HMM with the main difference that a $P-net$ allows the activation of multiple state nodes simultaneously; therefore a $P-net$ is able to represent activities with concurrent as well as sequential sub-events; however learning a $P-net$ model is a very complex and extremely costly task. Similarly, Petri nets [Petri, 1966] are useful in expressing sequencing, synchronization and concurrency [David and Alla, 1994] however they suffer from the disadvantage to be manually schematized for representing the model structure as automatic learning methods do not exist yet.

4.3.3 Syntactic approaches

Syntactic approaches represent human activities as a string of symbols, each of them corresponding to a sub-event. These symbols are part of a whole set of structural rules having syntactic meaning and which govern the composition and the generation of high-level activities (strings) from symbols. This set of rules can be viewed as grammars in language modeling which specify how sentences (activities) can be constructed from words (symbols). Also, from a grammar, one can be able, using parsing techniques, to recognize if a sentence (an activity performed in a video) conforms to the rules (activity model) associated with the considered grammar. This set of rules governs the composition of clauses, phrases, and words in any given natural language. One of the earliest work using the concept of grammars for activity recognition is by Brand [1996] who uses a simple and non-probabilistic grammar for hand manipulation recognitions in sequences containing disassembly tasks. Ryoo and Aggarwal [2006] introduce the context-free grammar (CFG) formalism for high-level human activity recognition. In CFG, it is only necessary to enumerate the list of atomic actions that need to be detected and a set of rules that define higher-level activities of interest. Once CFG rules have been defined, efficient algorithms [Earley, 1970] can be used to parse them for activity recognition in real-time setup.

CFGs are deterministic in nature, and thus, they expect perfect accuracy in lower-level activity identification. As algorithms for low-level primitives are mostly probabilistic, stochastic context-free grammars (SCFGs) have been proposed as an extension of the CFGs in probabilistic cases and were used in several works [Ivanov and Bobick, 2000, Moore and Essa, 2002, Minnen et al., 2003]. For instance, Ivanov and Bobick [2000] use the SCFGs to model the semantics of activities whose structure was assumed to be known. In [Joo and Chellappa, 2006], the authors propose the attribute grammars as an extension of the SCFG in which they attach, to each atomic event, semantic tags or additional attributes enabling the recognition of more descriptive activities. For example, in case the exact location in which an event occurred is important for describing the event, attribute grammars allow to associate the location attribute to the event when formalizing the grammar rules. An example of a SCFG is depicted in Figure 4.9.

```
Fighting  ->  Punching              : 0.3
          |   Punching Fighting      : 0.7

Punching  ->  stretch withdraw      : 0.8
          |   stretch stay_withdrawn : 0.1
          |   stay_stretched withdraw : 0.1
```

Figure 4.9: An example of a simplified Stochastic Context-Free Grammar for representing and recognizing the "fighting" activity. "Fighting" is defined as a sequence of "punching" actions which itself can be decomposed into "stretching" and "withdrawing". From [Aggarwal and Ryoo, 2011].

Although suitable for representing sequential activities, syntactic approaches assume that all observations are parsed by applying their production rules. Therefore, it is required to provide a set of rules for all possible events, even for a large domain. Moreover it has been proved that automatically learning the rules of a grammar from training data is extremely difficult in the general case [de la Higuera, 2000].

4.3.4 Description-based approaches

A description-based methodology is a methodology that explicitly maintains a description of the high-level activity in terms of sub-events composing the activity together with their temporal, spatial and logical relationships. Thus, an activity is modeled as an occurrence of its sub-events satisfying certain relations (specified in the description).

Usually, a *time interval* is associated with a sub-event for representing temporal relationships. Allen's interval algebra [Allen, 1983] has been widely used in the literature for this purpose [Pinhanez and Bobick, 1998, Nevatia et al., 2003, Vu et al., 2003a]. In [Pinhanez and Bobick, 1998], the authors make use of Allen's temporal predicates to model sophisticated temporal ordering constraints such as past, now and future, leading to the PNF (past-now-future) network.

Most of the time, the descriptions are performed through precompiled domain specific scenario models [Vu et al., 2003a]. These models can be of various forms including logical rules [Rota and Thonnat, 2000, Tran and Davis, 2008, Neumann and Möller, 2008], declarative programming procedures [Vu et al., 2003b, Nevatia et al., 2003], and ontologies [Town, 2004, Nevatia et al., 2004, Hartz and Neumann, 2007]. Logical rules have the advantage of being able to easily express the domain knowledge. Also, the results of high-level reasoning can be easily presented in a human readable format. However, they may require an extensive enumeration of all the logical rules by a domain expert which may be different from one setting to the other.

Declarative programming procedures rely on description languages for modeling each scenario of interest. For example, in [Vu et al., 2003b], the authors design a representation language to describe human activities. In their language, a scenario ω is made of four parts:

- a set of actors variables (characters and objects) involved in the ω,
- a set of sub-scenarios that composed ω,
- a set of *forbidden* sub-scenarios (i.e., scenarios not occurring during ω),
- and a set of constraints (temporal, atemporal and forbidden) of ω.

They use Allen's temporal predicates, spatial predicates, and logical predicates to specify the scenario constraints. Moreover, with the *sub-scenario* notion, their language is able to handle activities with any levels of hierarchy. Figure 4.10 represents a model of a composed scenario "Bank attack" which involves two actors, a cashier and a robber. For the recognition, they rely on Bayesian networks for identifying elementary scenarios while HMMs are used for the identification of composite scenarios. However, as mentioned in [Aggarwal and Ryoo, 2011], the expressiveness capacity of their language is limited since only conjunctive predicates are allowed when concatenating multiple temporal relationships (i.e., only *and* is allowed, not *or*).

In [Zouba et al., 2010], the above described description language has been extended to address complex activity recognition involving several physical objects of different types in a scene observed by video cameras and environmental sensors and which is lasting over an extended period of time. The resulting description language has been used for the monitoring of elderly activities at home.

Unlike logical rules and/or declarative programming procedures which focus on describing specific scenarios of interest, ontologies try to centralize representations of activities independently

```
Scenario(Bank_attack,
   Characters((cashier:Person), (robber:Person))
   SubScenarios(
     (cas_at_pos,inside_zone,cashier,"Back_Counter")
     (rob_enters, changes_zone, robber,
           "Entrance_zone", "Infront_Counter")
     (cas_at_safe, inside_zone, cashier, "Safe")
     (rob_at_safe, inside_zone, robber, "Safe") )
   ForbiddenSubScenarios(
     (any_in_branch, inside_zone, any_p,"Branch"))
   Constraints(
     Temporal ((rob_enters during cas_at_pos)
               (rob_enters before cas_at_safe)
               (cas_at_pos before cas_at_safe)
               (rob_enters before rob_at_safe)
               (rob_at_safe during cas_at_safe))
     Atemporal((cashier ≠ robber))
     Forbidden((any_p ≠ cashier) (any_p ≠ robber)
               (any_in_branch during rob_at_safe))))
```

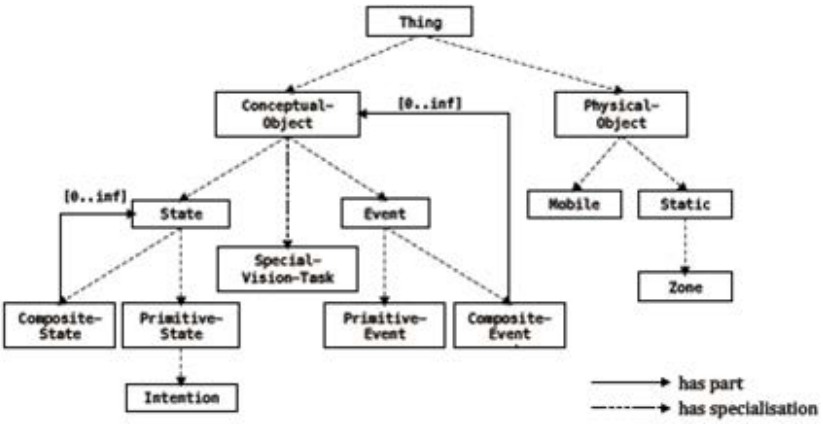
Figure 4.10: Example of declarative programming procedures - "Bank attack" scenario composed of four steps: (1) the cashier is at his/her position behind the counter, (2) the robber enters the bank and moves toward the front of the counter then (3) both of them arrive at the safe door and (4) nobody else in the branch during the attack. From [Vu et al., 2003b].

of the scenario in which they will be recognized. For example, in [Bohlken et al., 2013], the authors describe a generic framework for model-based behavior interpretation and its application to monitoring aircraft service activities. To serve the general purpose of scene interpretation, they use OWL-DL (Ontology Web Language Description Logics), the standardized web ontology language, for the representation of the contextual model. Also, they rely on the SWRL (Semantic Web Rule Language) extension of OWL to express application constraints (e.g., temporal constraints) between different concepts (e.g., objects, characters) within the model. An ontology, described using the OWL-DL, usually comprises a domain-independent *upper model* with concept types which are generally useful for scene interpretation, and a *domain model* defined in relation with the application domain. In their approach, the described (i.e., in OWL-DL) models are automatically converted into a java-based operational scene interpretation system which accepts tracked objects as input and delivers high-level activity descriptions as output. The interpretation (inference) process uses Beam Search [Manber and Myers, 1993] — an optimization of the best-first search algorithm for graph exploration — for exploring the interpretation space and, it is guided by a probabilistic model which is designed in such a way to correspond to the hierarchy of the different concepts defined within the ontology. An example of such an ontology is shown is Figure 4.11.

While description-based approaches are able to represent and recognize human activities with complex temporal structures (sequential and/or concurrent sub-events), their major drawback resides in their inability to compensate for the failures to identify low-level sub-events as it was the case for CFGs. That is, most of the description-based approaches have a deterministic high-level component [Aggarwal and Ryoo, 2011].

4.4 Simultaneous Tracking and Activity Recognition

In the previous section, we have seen, for example in [Moore et al., 1999], how by integrating knowledge regarding the object in use, one can leverage such information to easily recognize



(a)

```
Turnaround
    Arrival
        Arrival-Preparation
            GPU-Enters-GPU-Zone
            GPU-Stopped-Inside-GPU-Zone
                GPU-Stopped
                GPU-Inside-GPU-Zone
            Drop-Chocks
        Airplane-Enters-ERA
        Airplane-Stopped-Inside-ERA
            Airplane-Stopped
            Airplane-Inside-ERA
    Services
        Passenger-Activity
            Passenger-Stairs-Enters-PS-Zone
            Passenger-Stairs-Stopped-Inside-PS-Zone
                Passenger-Stairs-Stopped
                Passenger-Stairs-Inside-PS-Zone
        Unload-Right
            Unload-Right-AFT
                Loader-Enters-Right-AFT-LD-Zone
                Transporter-Enters-Right-AFT-TS-Zone
                Unload-Motion-Right-AFT-Belt
                Transporter-Leaves-Right-AFT-TS-Zone
                Loader-Leaves-Right-AFT-LD-Zone
            Unload-Right-FWD
                Loader-Enters-Right-FWD-LD-Zone
                Transporter-Enters-Right-FWD-TS-Zone
                Unload-Motion-Right-FWD-Belt
                Transporter-Leaves-Right-FWD-TS-Zone
                Loader-Leaves-Right-FWD-LD-Zone
        Refuelling
            Tanker-Enters-Tanking-Zone
            Tanker-Stopped-Inside-Tanking-Zone
                Tanker-Stopped
                Tanker-Inside-Tanking-Zone
            Pumping-Operation
        Load-Right
            Load-Right-AFT
                Loader-Enters-Right-AFT-LD-Zone
                Transporter-Enters-Right-AFT-TS-Zone
                Load-Motion-Right-AFT-Belt
                Transporter-Leaves-Right-AFT-TS-Zone
                Loader-Leaves-Right-AFT-LD-Zone
            Load-Right-FWD
                Loader-Enters-Right-FWD-LD-Zone
                Transporter-Enters-Right-FWD-TS-Zone
                Load-Motion-Right-FWD-Belt
                Transporter-Leaves-Right-FWD-TS-Zone
                Loader-Leaves-Right-FWD-LD-Zone
    Departure
        Start-Beacon
        Pushback
```

(b)

Figure 4.11: Example of an ontology for the for aircraft servicing domain: (a) the upper model and (b) the Domain model describing an *an aircraft turnaround* and its constituting activities. From [Bohlken et al., 2013].

the underlying interaction between a human and the considered object. In this context, the recognition of objects can benefit the activity recognition while the activity recognition may help the identification of objects. Similarly, moving pedestrians are driven by an inner motivation in relation with the activity they are performing in the environment. Therefore, there is an intrinsic dependency between the location and the activity within the environment and thus, it is possible to improve the estimation of the pedestrian behavior by leveraging this contextual information. This section focuses on methods which seek to take advantage of such a contextual knowledge.

Usually, an environment is structured in such a way that only a finite set of activities can be performed in a given area of the environment. For illustration purposes, let us consider a home composed of a living room and a kitchen. Activities like *"cooking"* and *"opening the fridge"* can only be realized in the kitchen while activities like *"watching TV"* and *"sitting on the sofa"* can only be observed in the living room. Integrating this additional location dimension as useful information for activity identification is the basic principle of methods which simultaneously address motion tracking and activity recognition when trying to infer pedestrian behaviors. In such methods, the knowledge regarding the location of the pedestrian can help improving the estimation of the current activity while the belief regarding the activity can benefit in improving the estimates of the pedestrian location. Therefore, we are in a loopy architecture that can be represented as depicted in Figure 4.12.

Figure 4.12: The benefits of joint motion tracking and activity recognition approach: the knowledge regarding the location data are used for improving the estimation of the underlying activity and conversely.

One of the precursor work falling in this category is by Bruce and Gordon [2004]. They characterize a pedestrian by both his current location and his final goal (destination point) within the environment. Then, they use a particle filter based on a path planner (see Section 4.2.2) to automatically infer the pedestrian goal and improve the motion prediction. A limitation of their method is that, by using a path-planner-based motion model, they are not able to deal with dynamic obstacles. Moreover, they assimilate pedestrian activity as physical points in the environment which is not sufficient in most situations. Additionally, causality effects are nonexistent as it is not possible to infer, from previous visited points, where a pedestrian is likely going to move thereafter.

Wilson and Atkeson [2005] employ the generic term "simultaneous tracking and activity recognition" (STAR) to formally refer to approaches — as, previously, in [Vu et al., 2003a,b] — in which people tracking is used to improve activity recognition and vice-versa. They consider the problem of automatic health monitoring in an instrumented home for elderly people using binary sensors. An illustration of such an instrumented home is shown in Figure 4.13. In order to represent the contextual knowledge within the environment, they use dynamic Bayesian networks (DBNs) as depicted in Figure 4.14 in such a way to put in ralation the room in which a given target is and his current activity. They subsequently learn the probabilities characterizing such a model and rely on a particle filter for inference purposes. Finally, they perform some experiments in a real setup. One limitation of their method is that the activities to be recognized are

rudimentary (i.e., whether or not a pedestrian is moving, or, in which room a pedestrian can be found).

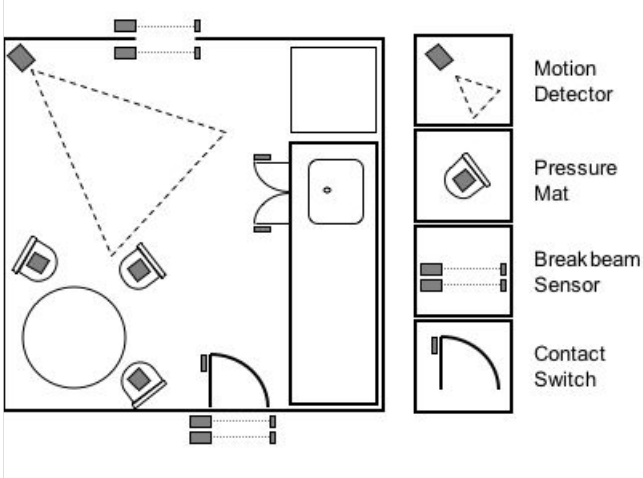


Figure 4.13: Example of instrumented home for automatic health monitoring. Grey squares represent sensors. From [Wilson and Atkeson, 2005].

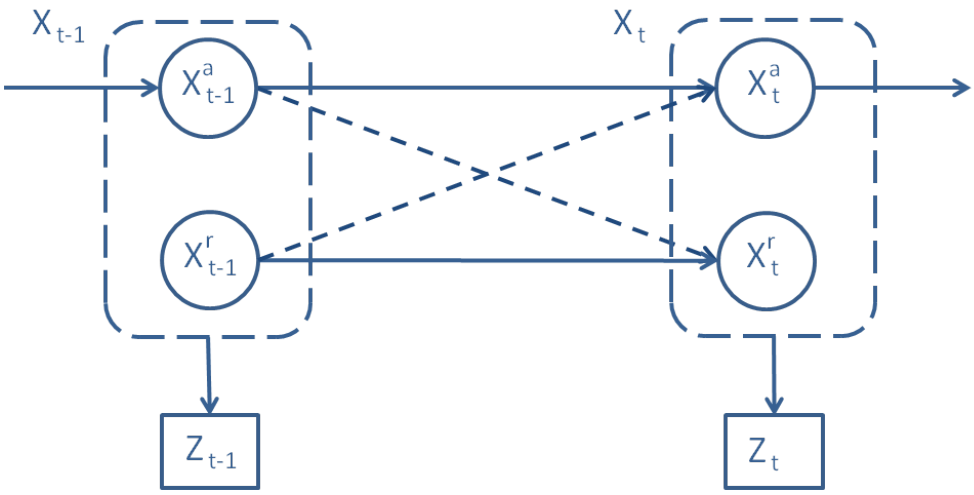


Figure 4.14: Example of DBN describing room-level tracking and activity recognition. From [Wilson and Atkeson, 2005]. $\mathbf{X}^r_{\cdot}$ represents the room in which the target can be found while $\mathbf{X}^a_{\cdot}$ models the activity of the target (whether he is walking or not). Arrows indicate causal influences through time.

In [Manfredotti et al., 2011], the authors try to identify the type of relations between pedestrians while tracking their locations. To this end, they introduce the concept of *relational state* characterizing each target. A relational state of a target is composed of two parts: the state of its proper attributes and the state of its relations with other targets. Then, they investigate the use of DBNs on these relational states (leading to a model that they called *relational DBNs*) to represent useful contextual knowledge regarding the dynamics of the relations between targets over time (see Figure 4.15). Finally, they used the particle filter for inference purposes. They demonstrate that the explicit recognition of the relation between targets improves the estimation

of their behaviors. Also, they argue that their approach has significant advantages over methods that do not use relations. However, one drawback of their method is the necessity to learn the parameters of the relational DBN which is not obvious, particularly as it involves several entities. Also, DBNs, by nature, are limited to describe activity sequences. Moreover, although it is theoretically possible to incorporate further activity-related information (e.g., temporal data such as the duration of an activity) within the DBNs using additional random variables, one may easily obtain, in a real setup, a quite large data structure which may prove to be difficult to manage in practice. This is mainly because of the strict first-order assumption governing the DBNs. Despite all these limitations, their approach is still interesting.

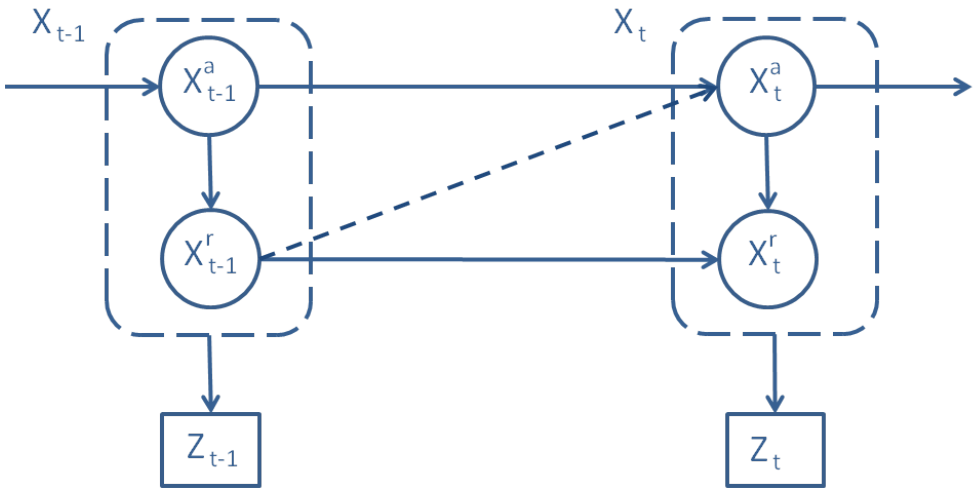


Figure 4.15: Relational transition model. From [Manfredotti et al., 2011]. $\mathbf{X}_{\cdot}^{a}$ represents the proper attributes of the targets while $\mathbf{X}_{\cdot}^{r}$ models the relation with other targets. Arrows indicate probabilistic dependence between variables.

4.5 Conclusion

In this chapter, we considered two behavioral aspects associated to the notion of '*pedestrian behavior* (the location and the activity) and we presented different methodologies that exist in the literature for handling the pedestrian behavior inference problem. These methodologies can be classified into three main groups depending on which aspects of the behavior they are focusing on: (1) motion tracking approaches which focus only to the location, thus dealing with the localization problem; (2) activity recognition approaches which only consider the activity performed by the pedestrian; and (3) simultaneous tracking and activity recognition (STAR) approaches which consider both the location and activity data. Methodologies belonging to this latter group tend to exploit the inherent synergy between location and activity in order to improve the estimation of activity from the knowledge regarding the location data and inversely. Most of the works belonging to this third group rely on graphical models, and most notably dynamic Bayesian networks (DBNs), to express contextual knowledge that are later on leveraged during the inference process. The structure of such models is usually provided by domain experts or learning from training data. However, DBNs, by nature, are limited to describe information regarding sequential activities. Moreover, in case of complex scenes, graphical models usually require a lot of nodes and therefore, they may reveal to be difficult to manage in practice. In

the next chapter, we address the problem of recognizing high-level pedestrian activities (human-object interactions) within an environment equipped with several objects. We opt for a STAR approach, but instead of using graphical models for context-based knowledge representations, we rely on behavioral simulations of autonomous agents within the considered environment.

Chapter 5

Inferring pedestrian behaviors from agent-based simulations

Contents

5.1 Introduction

This chapter is devoted to the design of an original approach for pedestrian behavioral tracking. As previously stated (cf. Chapter 1), this problem consists in inferring the behaviors of pedestrians within an environment on the sole basis of observation data from a sensor network. We consider the general case in which, on one hand, the environment may contain areas which are not under sensory coverage and, on the other hand, it is possible for a pedestrian to make use of different objects (interactions with objects) in the environment to serve his own purposes. Additionally, we particularly focus our attention on high-level activities and, thereby, low-level activities, such as gestures, are not concerned in this chapter.

In this work, we consider the simultaneous tracking and activity recognition (STAR) framework (cf. Section 4.4) whose principle consists in leveraging, during the inference procedure, contextual knowledge to improve the estimations, in terms of location and activity, of the pedestrian behaviors (knowledge related to activities are used to improve locations' estimates and conversely). Such a knowledge can be centered, for example, on a target (e.g., the nature of the target as for instance *police officer*, *passenger* or *controller*), a group of targets (e.g., nature of their relationship as for instance *family* or *friends*), a location (e.g., list of activities accessible from a given location), an activity (e.g., places where the activity can occur, duration of the activity), or a causality rule (e.g., list of activities usually performed after performing a given activity).

5.1.1 The STAR approach and current limitations

STAR methodologies presented so far in the literature [Wilson and Atkeson, 2005, Manfredotti et al., 2011] mostly rely on graphical models and principally on dynamic Bayesian networks (DBNs) to encode the contextual knowledge of the environment. While DBNs are suitable for describing event sequences and causality effects, they are usually designed in such a way to *neglect* enough details for explicitly dealing with objects present in the environment, especially, their usage (e.g., the duration of a given event along the time). This restriction appears to be insufficient in practical situations as for example in case of an interaction with an object (e.g., withdrawing cash from an ATM) which may involve a certain duration. Moreover, as pointed out in Section 4.3.2, capturing the global scene's contextual information with DBNs (and more generally graphical models) in real-world scenarios with complex environments would yield a quite large data structure which may prove to be difficult to manage in practice.

Besides these intrinsic limitations of the DBNs, the above-mentioned methodologies have been designed under the assumption that the context of the considered environment is not subject to dynamical changes caused by either pedestrian actions or exogenous events. While this assumption serves well the purposes of these works, it cannot be generalized in everyday life (e.g., an escalator failure and/or a fire alarm within a subway station change the environmental context and influence the behaviors of the passengers therein).

5.1.2 Agent-based behavioral simulators: an alternative to graphical models

To overcome the above limitations, we propose to rely on advanced agent-based behavioral simulators [Shao and Terzopoulos, 2007] for modeling the contextual knowledge of the environment.

Nowadays, the use of such simulators to generate context-dependent realistic human behaviors in indoor environments has become very popular in a wide range of application domains such as crisis management [Hanisch et al., 2003], urban planning [Stylianou et al., 2004, Osaragi, 2004] and virtual training. These simulators aim at realistically reproducing human behaviors within complex environments using situated virtual agents equipped with sensing capabilities and, to this end, they allow to virtually represent, in a finer way, the details of the environment under consideration together with the objects therein as well as the related services.[6] Therefore, it is possible to easily describe, through these simulators, the behavioral process of a pedestrian when interacting with a given object in the environment as well as the duration of such an interaction.

Although DBN's extensions have recently been proposed in order to incorporate the duration of an event [Donat et al., 2010, Wang et al., 2011], agent-based behavioral simulators present the advantage of naturally embedding agents' behavioral models which are characterized not only by basic attributes (e.g., position, velocity), but also by inner action-selection mechanisms [Tu and Terzopoulos, 1994, Funge et al., 1999] responsible for creating and executing navigational plans depending on the environmental context. Additionally, in an effort for integrating the effects an environment can have on the behavior exhibited by an agent, works (from the situated artificial intelligence field) have focused on coupling such action-selection mechanisms with sensori-motor loops [Maes, 1994, Meyer, 1997] in such a way that the internal state of an agent depends on his perceptions (information sensed from the environment). On this basis, as the behavioral model aims at determining the action a given agent should undertake based on its current internal state, it is possible for the considered agent to be adaptive and therefore automatically take into account the dynamical changes occurring in the environment.

5.1.3 Contributions

Having in mind the capacities of agent-based behavioral simulators, the solution we propose to the behavioral tracking problem consists in integrating such a behavioral simulator, as a predictive block, within the Bayesian filtering framework (see Chapter 2), and for such purposes, the considered simulator aims at providing hypotheses regarding the real world. In this chapter, we do not intend to describe how to design a behavioral model for a given environment, but how to exploit such a model (assumed designed upstream by experts) for inferring pedestrian behaviors.

Using together real-world data sets and simulations is becoming commonplace in the field of human-simulation validation [Osaragi, 2004, Vizzari and Manenti, 2012]. While the main purpose of these works is to end up with a model that simulates a real phenomenon well, our objective is to leverage such a model for behavioral inference purposes. Also, there have been some methodologies [Antonini et al., 2004a,b, Butenuth et al., 2011] relying on simulations within the context of tracking and behavior estimation. However, these approaches mainly focus on crowd properties and are neither interested in determining individual activities, nor in considering challenges due to occlusions — the major concern of our work.

This chapter is organized as follows. Section 5.2 briefly introduces agent-based behavioral simulators and describes the principles of such simulators while emphasizing on their utility for solving a STAR problem. In Section 5.3, we present the proposed solution and discuss the

[6] The set of services related to an object corresponds to the different utilities of that object within the considered environment. For instance, in a subway station, a vending machine can be used for buying either food or drinks. Similarly, the ticket machine is used for buying tickets.

implementation details in Section 5.4. To this end and for simplicity, we consider the special case of single target tracking. Then, Section 5.5 focuses on experimental evaluations of the proposed approach both on simulated and real data. Finally, Section 5.6 identifies some important research lines induced by the described work.

5.2 Autonomous agent-based behavioral simulation

In this section, we describe the general principles together with the architecture governing autonomous pedestrians simulators. Because each pedestrian has his own characteristics (in terms of perception, decision-making, and interaction), we particularly focus on simulators which, in their conceptual approach, (1) model each pedestrian as a **unique individual** via virtual characters or agents, (2) allow these virtual agents to sense their environment and integrate their perceptual data in their decision-making process, and (3) make possible for virtual characters to use (i.e., interact with) different objects present in their environment to serve their own purposes. In other words, these are simulators which rely on self-animated models of individual characters that incorporate human-like abilities useful to the purpose of animating virtual pedestrians. Therefore, are not concerned in this section *"crowd animation"* simulators in which one character is not equipped with a decision-making module and simply follows another one according to a fluid-like model.

Given an environment, humans usually perform activities in relation with the objects (or corresponding services) available in their surrounding. For instance, one can only execute the action "buy a ticket" if there is a ticket machine or a counter from which one can buy a ticket. The proper objective of autonomous pedestrians behavioral simulators is to realistically reproduce human-like behaviors within the simulated environment. When the simulated environment is a perfect reproduction of a real building, the importance of realism is such that the aim for these simulators is to produce virtual animations from which one cannot pinpoint important differences with observations captured from real life. For achieving such a level of realism, three dimensions are of crucial importance when designing such a simulator [Shao and Terzopoulos, 2007]:

- **The representation of the services**. The first dimension is the ability to incorporate, within the virtual world, all the services that may be of particular interest for a given pedestrian. This dimension includes the description of object interactions available in the underlying environment together with information regarding how these interactions are performed from a pedestrian point of view. More details are provided in Section 5.2.1.

- **The selection of agent actions**. The second dimension, which is discussed in Section 5.2.2, concerns the ability for a virtual character to express, in a coherent and persistent way, its desire for a given service in the environment so as to work towards the satisfaction of its goal. This implies an effective design of action-selection mechanisms which need to rapidly adapt to external factors from the environment.

- **Planning**. Finally, the third dimension, presented in Section 5.2.3, regards the ability to intelligently guide an agent towards a service in the environment capable to satisfy its desire. This involves a high-level planner responsible for determining the optimal matching between "desire" and "service" while taking into account the actual situation of the environment and the considered agent's characteristics.

In what follows, we provide a brief overview of each of these dimensions (Section 5.2.1— Section 5.2.3). Finally, Section 5.2.4 presents a brief summary while emphasizing on the role of such simulation systems within the STAR framework.

5.2.1 Environment's Service Representation

The ability to model services available in the environment is an important factor for pedestrian simulators wishing to achieve a high degree of realism. These services are mostly represented via dedicated objects situated in the environment with which pedestrians may interact to satisfy their needs; hence, the prominence of agent-object interactions within these simulators.

From a designer point of view, dealing with agent-object interactions implies incorporating, within the simulator, knowledge regarding how these interactions are performed. This knowledge is then leveraged by virtual agents, when appropriate, to correctly execute expected actions while interacting with a given object so as to give the illusion of realism. The more detailed the provided knowledge, the more realism. Let us consider the example of using an elevator. A typical associated knowledge can be described as the following sequence: reaching the front of the elevator, searching for the call button, pressing the call button, waiting for door opening, entering the elevator, searching for destination buttons, specifying the destination, waiting for door opening at your destination, and exiting the elevator.

When the simulator is designed for a predefined set of tasks, this knowledge can be inserted within each agent's model. However, when the simulator aims to be used generically in different settings without updating the core of the system, such an approach is not plausible at the risk of having an extremely complex and unmaintainable agent model. In [Kallmann and Thalmann, 1998], the authors introduce the *Smart Object* concept, a feature modeling approach, to overcome this issue. In such an approach, an object includes within its description not only intrinsic properties, but also interaction information. Intrinsic properties refer to properties that are part of the object design such as physical properties (e.g., form, weight) as well as internal mechanisms (e.g., movement descriptions of non-static parts — the movement of an escalator). On the other hand, interaction information (also known as interaction features) aims at helping an agent to perform each possible interaction with the object. This information can be of various types. For example, the location of a given interaction part (e.g., the lift call button), the expected motion of the agent body part (e.g., hand shape and gestures, approach direction), or the object's movements resulted from the actions of an agent (e.g., the door opening process of an elevator). In some cases, it may be interesting to specify, within an object description, the possible states that the object may have (e.g., ascending or descending service for an escalator) together with authorized activities (interactions) associated with each of these states.

Consequently, using the Smart Object approach, a virtual agent does not need to maintain, within its knowledge, information regarding the usage of objects, but it has to retrieve such data from the object offering the service it is interested in. Also, in different settings, one only needs to specify the interaction features applicable for his specific case without modifying the underlying model of the agent. These interesting properties have led the Smart Object approach to be widely used in the literature [Jorissen et al., 2005, Shao and Terzopoulos, 2007, Bandini et al., 2009]. Again, be aware that the level of realism achieved by such simulation systems mostly depends on the specified interaction-features.

This way of providing relevant information via "smart objects" to facilitate the consideration of agent-object interactions within a simulation system fits with the logic behind describing the

processes that may occur in the environment. An illustration of an XML-based basic description of services offered by an elevator is depicted in Listing 5.1. In Listing 5.1, an elevator is characterized by three variables — namely *destination_list*, *received_command* and *current_level* — representing respectively the list of future destinations of the elevator, a command as specified by an agent, and its current level. It has two states: *Open* and *Close*. In the *Close* state, an agent can interact with the object from the outside by pressing the call button. This operation causes the update of the destination list. Also, in this state, when the destination list is not empty and the elevator is therefore moving, there is a background procedure which consists in verifying if the current level of the elevator is part of its destination list. If it is the case, the elevator stops and its state transits to *Open*. In the *Open* state, agents can either enter in or exit from the elevator. Entering agents can then specify their destinations, which has the effect of updating the list of destinations of the elevator. Also, there is a procedure which is launched after 10 seconds to automatically close the elevator's door. Figure 5.1 shows a virtual agent interacting with an elevator. For a complete example of designing an automatic door service using the Smart Object approach, please refer to [Kallmann and Thalmann, 1998].

Listing 5.1: Basic description of services offered by an elevator

```
<smartobject name="Elevator" color="(0.5,0.5,0.5)" form ="elevator_open.obj" volume="cabin"
  default_state="Close" rally_point="(0.5,0.0,0.0)" radius="0.9">
  <!-- ============================ VARIABLES ========================== -->
  <variables>
    <var name="destination_list" value="" role="the list of all future destinations"/>
    <var name="received_command" value="" role="a destination as specified by the user"/>
    <var name="current_level" value="" role="the current level of the elevator"/>
  </variables>
  <!-- ============================= STATES ============================ -->
  <state name="Close">
    <targets>
      <nop id="init"/>
      <move id="search" name="searching call button" destination="(0.75,0.75,0)"
        orientation="(-1,0,0)"/>
      <interact id="call" name="pressing call button" result="received_command"
        duration="1000" animation="pressing">
        <update var="destination_list" add="received_command"/>
      </interact>
    </targets>
    <transitions>
      <transition from="init" to="search"/>
      <transition from="search" to="call"/>
    </transitions>
    <procedures>
      <procedure id="Automatic_Opening" name="deserving destination list"
        Activated="NOT_EMPTY destination_list">
        <trigger id="checkArrival" name="checking if condition is satisfy">
          <condition var="current_level" operator="in" var="destination_list">
            <changestate to="open" duration="10000" animation="opening"/>
          </condition>
        </trigger>
      </procedure>
    </procedures>
  </state>
  <!-- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -->
  <state name="Open">
    <targets>
      <nop id="init">
        <update var="destination_list" remove="current_level"/>
      </nop>
      <move id="enter" name="enter the elevator" destination="(-4.0,0,0)" radius="0.5"/>
      <interact id="destination" name="pressing destination button" duration="1000"
        result="received_command" animation="pressing2">
```

```
        <update var="destination_list" add="received_command"/>
      </interact>
      <move id="exit" name="exit the elevator" destination="(4.0,0,0)" radius="0.5"/>
    </targets>
    <transitions>
      <transition from="init" to="enter"/>
      <transition from="enter" to="destination"/>
      <transition from="init" to="exit"/>
    </transitions>
    <procedures>
      <procedure id="Automatic_Closure" name="elevator closure" elapsedtime="10000" >
        <trigger id="AutomaticWait" name="Waiting for automatic closure of the elevator">
          <changestate to="Close" duration="10000" animation="closure"/>
        </trigger>
      </procedure>
    </procedures>
  </state>
</smartobject>
```

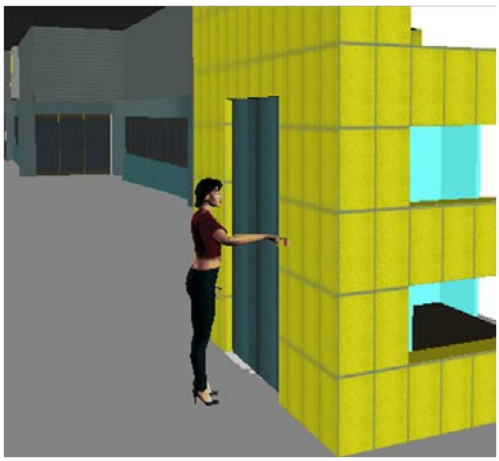


Figure 5.1: Illustration of agent-object interaction with an elevator. From [Kallmann and Thalmann, 1998].

5.2.2 Action Selection Mechanisms

Like real humans, autonomous virtual pedestrians must be able to take, in a coherent and persistent way, their own decisions in real-time regarding what they are willing to achieve in the environment (the service they are interested in). Such a decision-making process must be individual for each agent in order to enhance personal characteristics. Moreover, it should depend not only on agent internal variables but also on various factors and opportunities coming from the environment in which the agent is immersed. In the "virtual humans" field, the *action selection mechanism* is responsible for choosing, for a given agent, the appropriate action to execute at each moment in time. In order to obtain a high degree of autonomy, an action-selection mechanism, besides being individual, needs to be reactive and proactive [Nareyek, 2000]. Pro-activity refers to anticipatory, change-oriented and self-initiated behavior in situations. On the other hand, reactivity is about responding to events occurring in the environment.

In [Bryson, 2000], the authors, after performing a study over a wide range of architectures, conclude that a hierarchical architecture best promotes the pro-active capacity when designing an action-selection mechanism. Similarly, as suggested in [Tyrell, 1993], the reactive capacity can

efficiently be achieved using a *free flow hierarchy*, a typical instance of a hierarchical architecture. A **free-flow hierarchy** is a hierarchical action selection architecture in which all nodes in the hierarchy can influence the subsequent behavior of the agent. Nodes higher in the hierarchy express weighted preferences for nodes lower in the hierarchy via transfer functions. This process propagates throughout the whole hierarchy, and as a result, instead of making a decision at each layer, a decision is only made at the lowest (i.e. action) level where the most highly preferred (or activated) node is chosen as the resulting action.

For illustration purposes, a free flow hierarchy is depicted in Figure 5.2. It mainly contains four layers:

- *Environmental information*, which corresponds to data originated from the environment. They can be categorized into two types. On one hand, we have perceptions that are directly sensed by the agent (e.g., alarm, fire). On the other hand, we have stimuli which correspond to retro-effects of actions initiated by the agent (e.g., satisfaction received when drinking some water).

- *Internal variables*, which represent physiological and psychological states of the agent (e.g., stress, thirst). They have their proper dynamics, which allows to maintain homeostasis properties, that is, a capacity to stay in a comfort area, generally a physiological viable zone [Meyer, 1998]. They may naturally evolve over time and/or can be affected by environmental data sensed by the agent.

- *Motivations*, which are dedicated to the representation of how a virtual actor should behave according to the environmental information and the states of the internal variables.

- *Actions*, which are defined in the lowest layer of the hierarchy. They are potential candidates to the outcome of the selection process. They correspond to objectives (goals) that need to be achieved by the virtual agent in the simulated environment. They can also be viewed as desires that the agent would like to satisfy in the environment.

One important characteristic of an action selection mechanism is its ability to preserve the homeostasis properties [Cannon, 1932] of the internal variables. This is achieved by choosing the most appropriate action that will maintain internal variables in their comfort zone. In [Meyer, 1998], threshold systems have been introduced to adapt the values of motivation nodes in such a way that the internal variables maintain their homeostasis properties. Such a system defines, for the associated variable (let us say i) and corresponding motivation represented by a scalar variable (let us say M), two threshold values (T_1 and T_2 with $T_1 < T_2$) delimiting three zones: the comfort zone, the tolerance zone and the danger zone. Basically, when the value of the variable i is less than the first threshold T_1 (comfort zone), the selection mechanism does not pay attention to the motivation M. If the value of the variable i lies between the two thresholds (tolerance zone), the motivation M holds the value of the internal variable. Finally, if the value of the variable i is greater than T_2 (danger zone), the value of the motivation is amplified in such a way that the action associated to the motivation M will have more chances to be selected by the overall mechanism, resulting to the decrease of the value of the variable i. An illustration of a threshold system is depicted in Figure 5.3 while an example of associated function characterizing

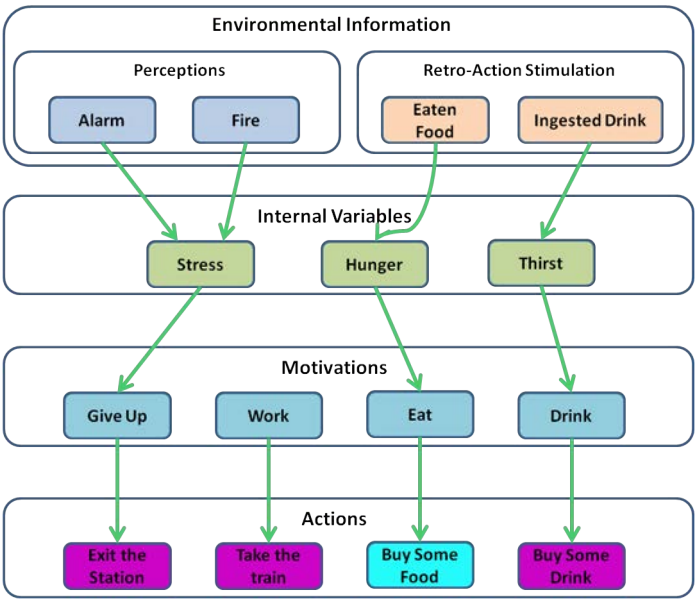


Figure 5.2: Example of a simple action selection mechanism based on a free-flow hierarchy for the subway station. Decisions are made only at the lowest level of the structure. The "Buy some food" node is the most activated one and, hence, it is chosen as output of the selection process.

such a system is formulated as

$$M = \begin{cases} 0 & \text{if} \quad i < T_1, \\ i & \text{if} \quad T_1 \leq i \leq T_2, \\ 2^i & \text{if} \quad T_2 < i. \end{cases} \tag{5.1}$$

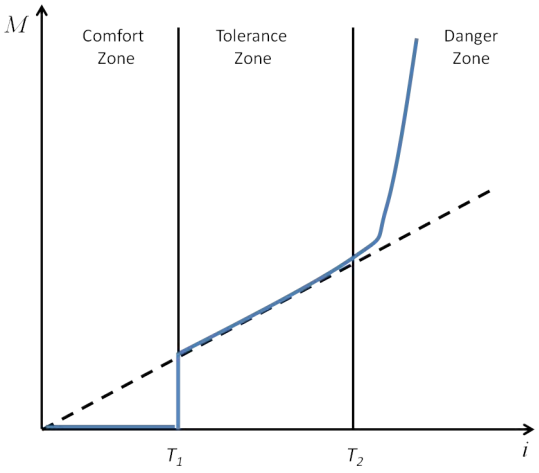


Figure 5.3: Example of threshold system for maintaining the homeostasis properties of the internal variables.

Another important aspect of action selection mechanisms is the persistence over time of the selected actions so as to avoid inadvertent oscillations between virtual agent motivations. Hysteresis functions (see Equation 5.2) are mainly used to keep at each instant, a fraction of a given

motivation from the previous iterations. They can be expressed as [De Sevin and Thalmann, 2005]

$$M_t = (1 - \alpha)M_{t-1} + \alpha C, \tag{5.2}$$

where M_t is the motivation value at time t, C is the motivation value resulting from the threshold system and α is the hysteresis parameter with $0 \leq \alpha \leq 1$. Using the hysteresis functions allows to maintain the chosen action as the most activated one until the corresponding internal variables return within their comfort zone. For an example illustrating a complete design and the evaluation of an action selection mechanism based on a free-flow hierarchy, please refer to [De Sevin and Thalmann, 2005].

5.2.3 High-level Planners

Once it is possible to represent the different services available in the environment using smart objects (cf. Section 5.2.1) and, for autonomous virtual agents to express their desires in a persistent and efficient way via action selection mechanisms (cf. Section 5.2.2), the next concern, from the designer point of view, is to determine how to realistically satisfy the current desire of an agent.

High-level planners [Geib et al., 1994, Geib, 1994] are responsible for determining the optimal plan to satisfy the desire exhibited by an agent based on the services available in the environment and the agent's own characteristics. Roughly speaking, a high-level planner has two purposes:

- *computation of the sequence of subgoals* necessary to satisfy the agent's desire. For illustration purposes, let us consider the subway station example introduced previously and an agent whose resulting desire from the action selection mechanism depicted in Figure 5.2 is "Take the train". A sequence of subgoals that can be generated is as follows: *buy ticket, cross ticket barriers, wait at the platform*. Generally, a subgoal involves a service in the environment. In such a case, it is the task of the high-level planner to search for the suitable object providing such a service in accordance with the proper characteristics of the agent and the environment's state. For instance, considering the previous example, let us assume that there are two machines from which the agent can buy a ticket: Machine A (which is the closest to the agent's location) and Machine B. Under normal circumstances, the agent will choose Machine A to buy its ticket. However, if there is a queue at Machine A and the agent is in a hurry, it would likely opt for the service provided by Machine B.

- *agent navigation*: Once the objects providing the services for satisfying the agent's desire have been identified, the agent needs to move towards these objects to perform the required interactions while avoiding obstacles. For such purposes, realistic motion models described in Section 4.2 can be used to handle the navigation of the agents in the environment.

5.2.4 Summary

The last three sections were devoted to a presentation of the main dimensions related to agent-based behavioral simulators. Putting together these dimensions results in a pipeline as illustrated in Figure 5.4. Basically, given a virtual agent, external factors from the environment (contextual

data) affect its internal state from which a desire is expressed at each point of time using an action selection mechanism. The expressed desire is then matched with a sequence of subgoals (in relation to the services available in the environment) whose execution allows to satisfy the desire. Finally, once the subgoals have been computed and the corresponding objects have been identified, the trajectories towards these objects are executed by the agent. Therefore, agents move in **self-explanatory trajectories** since it is possible to build, from the observed trajectories, a set of diagrams which may explain the agent's motivations (interactions to perform). The so described pipeline explicitly materializes the existing synergy between the activity and the location of the agents and, for this reason, the underlying simulators reveal to be useful in practice within the STAR framework.

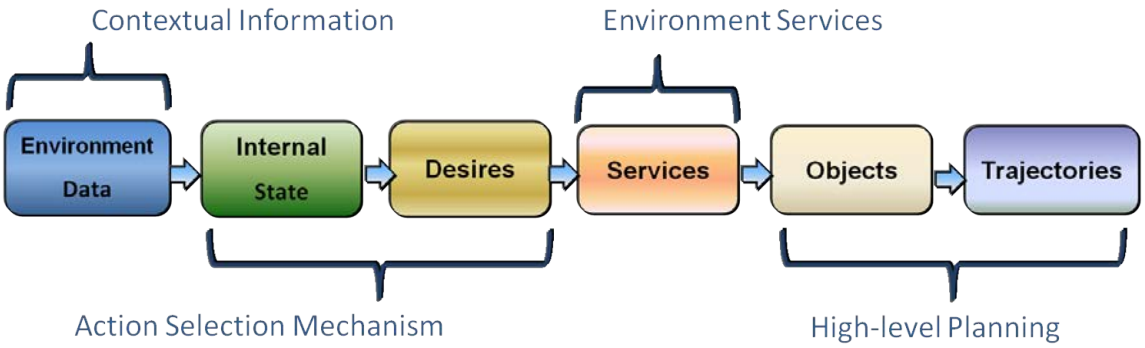


Figure 5.4: Overview of a behavioral simulator of autonomous pedestrians.

Coming back to our initial problem, we are interested in inferring the behaviors of moving pedestrians within a complex environment. In the following section, we propose and describe a novel approach in which we leverage advanced pedestrian behavioral simulators. As previously mentioned, our solution consists in using the behavioral simulation system as a predictive block within the Bayesian filtering framework (see Chapter 2), and for such purposes, it aims at providing hypotheses regarding the real world.

5.3 Agent-Based Behavior Tracking

In the previous section, we introduced the main principles of agent-based behavioral simulators and we emphasized on their utility within a STAR paradigm. In this section, we detail how such a simulator can be leveraged, through particle filters, for estimating moving people behaviors within an environment. We start by presenting the process overview of our approach together with the assumptions made. After that, we describe the underlying models (system dynamics and observation model) as required by the Bayesian filtering framework (see Section 2.1).

5.3.1 Process Overview

The solution we propose for estimating moving pedestrian behaviors is illustrated by Figure 5.5. In this approach, the simulator aims at artificially reproducing human-like behaviors for autonomous characters in a virtual environment simulating the real one. The pedestrians under

tracking in the real environment are represented by virtual agents within the simulator. As previously stated, a particle filter is used for inference purposes. In this filter, a set of hypotheses regarding the internal states of the virtual agents is considered. Then, based on these hypotheses, the simulator is used, during the prediction step of the filter, to simulate the behavior of the different agents and thus, to estimate the belief regarding both the aimed location and the activity of the tracked entities.

The observation data received from the camera network (deployed in the real environment) are subsequently used during the correction step of the filtering process to refine the computed belief and therefore to discard unlikely hypotheses as well as to update particle weights. These observation data are typically noisy location estimates of the detected targets usually obtained after a video analysis step. Thereby, we are not concerned with low-level activities (gestures) as we are interested in inferring high-level activities from the sole basis of observed trajectories. We assume that the video analysis is performed by an external module and thus, it is not part of the discussion. The camera network can totally or partially cover the environment. However, we are aware of areas under sensory coverage. Besides, we assume that it is possible to retrieve, at each time step, the states of the objects in the real environment. Finally, in order for the virtual agents to take into account dynamical changes happening within the real environment and to behave accordingly, the simulator is updated with changes regarding the states of the real objects (e.g., escalator failures) as well as exogenous events (e.g., fire, alerts) in such a way to preserve the coherence between the real world and the simulated ones.

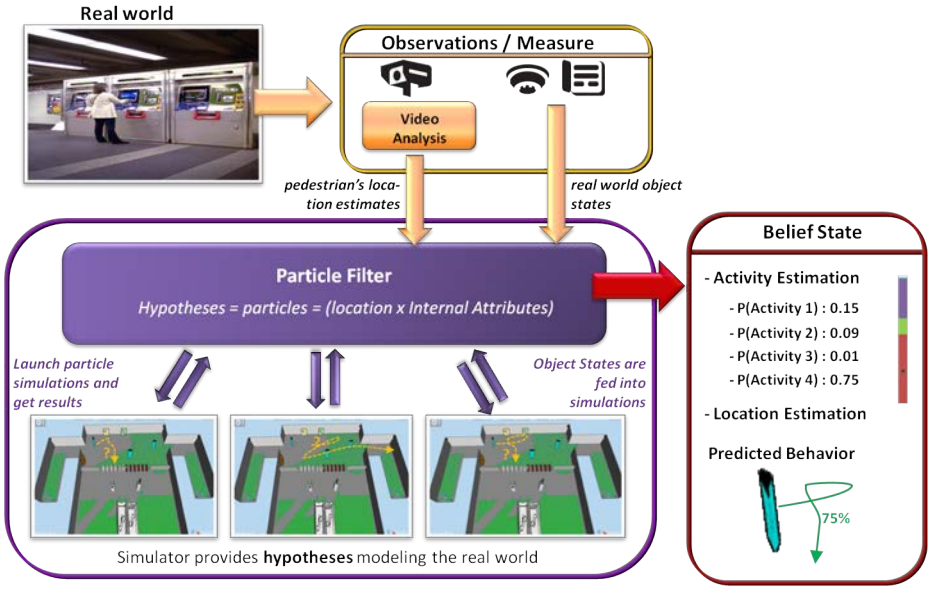


Figure 5.5: Process overview of the proposed approach: The behavioral simulator is used as a predictive block in the filtering procedure. Observations from the camera network are used to refine the belief regarding the behavior of the tracked entity. Also, the states of objects in the real environment are plugged into the virtual environment in order to preserve a coherence between both worlds.

After presenting the global architecture of our solution, the next section is dedicated to the description of the different models used in the particle filter. For simplicity, we center our description on the special case of a single target tracking scenario. The general case of multi-target tracking will be addressed later in this document (cf. Chapters 6 and 7).

5.3.2 System Dynamics

In this section, we describe the system dynamics used within the particle filter as shown in Figure 5.5. Considering a single target, our system is characterized by two elements at each time step t:

- $\mathbf{x}_t$, the state of the tracked target. It contains attributes that are taken into account within its action-selection mechanism, that is, all attributes that may play a role in the selection of the actions to be performed by the agent. These attributes can be split into two categories. The first category regroups spatial attributes (e.g., position, velocity, orientation) while the second one contains internal attributes representing for example the psychological and physiological traits (e.g., the thirst level, stress) as well as the resources (e.g., tickets, money) owned.

- $\mathbf{E}_t$, the state of the environment. It includes the state of the objects present therein. For instance, an escalator may be characterized by its current status (e.g., is it out of service or not) and, when applied, the direction (e.g., *up*, *down*) in which it transports pedestrians. $\mathbf{E}_t$ is assumed known at each time step.

As we assume the state $\mathbf{E}_t$ of the environment is known at each time step, we only need to consider the agent's dynamics. Since the simulator is responsible for simulating the virtual agent decision-making and navigational features, this dynamics is fully encoded within the simulator and can be represented by

$$\mathbf{x}_{t+1} \sim f(\mathbf{x}_t, \mathbf{E}_t), \tag{5.3}$$

where f is the stochastic function implemented by the simulator taking as input the state $\mathbf{x}_t$ of the agent together with the environment state $\mathbf{E}_t$ and modifying the agent's inner attributes.

On this basis, and because of the Smart Object approach (see Section 5.2.1), a particle within our system only contains information related to the characteristics of the tracked target and does not embed information related to (simulated) objects in the environment. Indeed, it is always possible, when simulating a particle, to retrieve from an object all the information needed for interacting with it. This has the advantage of reducing the dimension of the state space.

5.3.3 Observation Model

As stated before, the environment is equipped with a sensor network which provides noisy observation data to the system at each time step. In this work, we assume that the sensor network is characterized by a zero-mean Gaussian noise $(\mathbf{v}_t)_{t>0} \sim \mathcal{N}_{0;Q_v}(v_t)$ with a covariance matrix Q_v. The observation data, represented by the random variable $\mathbf{z}_t$, depends on whether the agent is within a covered area or not. Also, because of the noise, it is possible for the sensor network to not detect the agent even if the latter is within a covered area. This is likely the case when the agent is close to the boundaries of an area which is not under sensory coverage. Therefore, we distinguish two cases:

- the agent is within a non-covered area: $\mathbf{z}_t = \varnothing$;[7]

[7] $\varnothing$ refers to the fact that the agent is undetected by the sensor network.

- the agent is within a covered area: we consider $\tilde{\mathbf{z}}_t = h(\mathbf{x}_t) + \mathbf{v}_t$ with h being a function returning the location data from the state representation $\mathbf{x}_t$; because of the noise, the location data represented by $\tilde{\mathbf{z}}_t$ can fall within an area which is not under sensory coverage and thus, we have

$$\mathbf{z}_t = \begin{cases} \tilde{\mathbf{z}}_t & \text{if } \tilde{\mathbf{z}}_t \text{ is within a covered area,} \\ \varnothing & \text{otherwise.} \end{cases} \tag{5.4}$$

As illustrated in Equation 5.4, the agent may still be undetected even within covered areas. Next, we are interested in modeling the probability φ of such an event. As the observation noise is assumed Gaussian, φ can be assimilated to the portion of the Gaussian belonging to the area not covered by the sensor network. Such a portion is then approximated following the procedure described below:

1. First, we approximate the Gaussian noise by considering a region centered around P, the position of the agent, with a given radius r_h (e.g., a circle). See Figure 5.6 for illustration purposes.
2. Then, we discretize the considered region and compute the probability, with respect to P, for each cell (center) to belong to the region.
3. Finally, φ is estimated as the proportion $\hat{\varphi}$ of the region's cells belonging to the areas of the environment which are not under sensory coverage, that is

$$\hat{\varphi} = \frac{\sum \text{prob. of region's cells in non-covered areas}}{\sum \text{prob. of all region's cells}}. \tag{5.5}$$

Putting together all the different cases, the observation model can thus be summarized as follows:

$$p(\mathbf{z}_t|\mathbf{x}_t) = \begin{cases} 1 & \text{if } \mathbf{z}_t = \varnothing \text{ and } \neg cov(\mathbf{x}_t), \\ \hat{\varphi} & \text{if } \mathbf{z}_t = \varnothing \text{ and } cov(\mathbf{x}_t), \\ \mathcal{N}_{0;Q_v}(\mathbf{u}_t) & \text{otherwise,} \end{cases} \tag{5.6}$$

where $\mathbf{u}_t = \mathbf{z}_t - h(\mathbf{x}_t)$ and $cov(\mathbf{x}_t)$ indicates that the location data carried by $\mathbf{x}_t$ falls within an area covered by the sensor network.

We have now described the different models — system dynamics and sensor model — as required by the particle filter for performing the inference process. In the next section, we discuss the implementation details of the proposed approach.

5.4 Implementation

This section is dedicated to the implementation details of the system presented in Section 5.3. We start by presenting the agent-based behavioral simulator used for implementing the proposed approach. Then, we focus on managing target-object interactions within the filtering process and particularly at the resampling stage where particles are duplicated (created from existing ones). Finally, we present the architecture of the designed system so as to cope with real-time constraints.

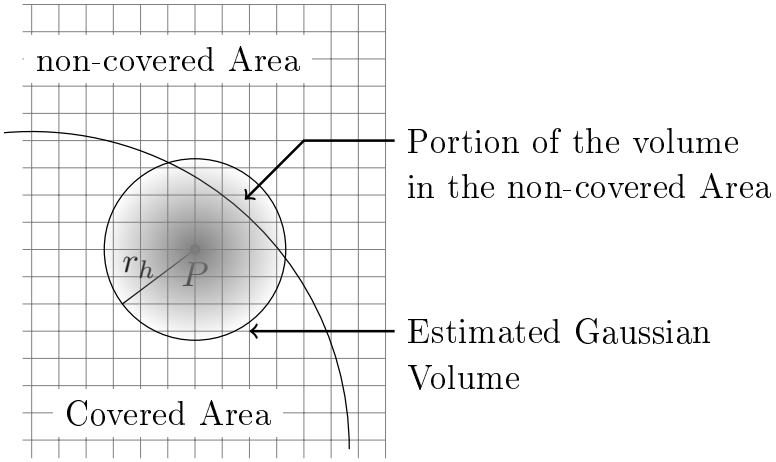


Figure 5.6: Approximation of φ: P is the agent position. The ratio is computed with respect to all cells in the considered region.

5.4.1 Simulator

The simulator used for implementation purposes is SE-Star [Navarro et al., 2015], a Thales[8] proprietary synthetic environment engine capable of modeling adaptive behaviors for autonomous agents including navigation features and interactions with objects. It is based on the concept of smart objects (see Section 5.2.1) and navigation is handled via a combination of A* path planning and the social force motion model as presented in Section 4.2. Furthermore, each virtual agent is characterized by an action-selection mechanism responsible for determining the action the agent has to undertake in order to satisfy its internal motivation. The selection mechanism is represented as a motivational tree based on a free-flow hierarchy approach (see Section 5.2.2). The mechanism uses hysteresis functions (see Equation 5.2) as well as threshold systems (see Equation 5.1) to preserve the homeostasis properties of the agent's internal variables.

However, SE-Star is a simulator with little randomness in behavior model dynamics. Therefore, running several simulations from a given agent state will generate identical results in terms of exhibited behaviors. Such processes with no or little noise are not appropriate when using a particle filter as they will lead to the sample impoverishment phenomenon [Arulampalam et al., 2002] described in Section 3.5. To tackle this issue, we implement a **regularized particle filter** (RPF) using Gaussian kernel (see Section 3.5) in order to explicitly introduce the randomness (via noise) within the filtering process. As previously explained, the main idea behind RPFs is to vary the particle set by resampling from a continuous approximation of the probability density function obtained via the provided kernel.

5.4.2 Interactions with objects

Recalling from Section 5.2.1, interactions with objects play an important role in the satisfaction of the agent desire. These interactions can take several time units in order to be achieved and, at their end, may provide the agent with some useful resources (e.g., money with an ATM, ticket

[8] Thales is a French multinational company specialized in critical information systems for security markets.

with a ticket machine) for proceeding with its plan. However, managing efficiently these interaction processes within a particle filter appears to be a challenging task, and most particularly at the resampling step when the particles, represented by virtual agents, are currently interacting with a given object.

In SE-Star, an agent in interaction with a smart object follows an interaction procedure provided by the considered object. Also, it is not possible to manually modify the current level of the execution of such a procedure and any attempt will completely interrupt the interaction. Under these conditions, the problem we faced here is that, if a special attention is not provided during the resampling, the newly generated particles could simply restart the interaction procedure again; hence the necessity to reproduce and obtain particles that are already in the process of interaction too.

Dealing with this issue (i.e., generating particles which are in interaction) per se is not difficult, at least from the theoretical point of view, since one needs to maintain, within the particles, the information regarding both the state of the agent and the state of the interaction (e.g., elapsed time since the beginning of the interaction, the current step of the interaction procedure). This solution suggests, however, that it is possible to retrieve, from the inner state of the underlying system (and therefore, the state of the particles), information regarding the objects on which the interactions are performed. This is in stark contrast with the solution proposed in Section 5.3 where a particle, by definition, only represents the state of the agent which contains nothing else than its own characteristics.

Nevertheless, it is possible to deduce from the particle's state whether or not the agent is currently interacting with an object. Based on this indicator function, a naive solution to the "object interaction" problem consists in not resampling from particles that are in interaction (thereby maintaining them in the resulting particle set), thus allowing them to completely achieve their current interaction process. While this solution seems appropriate, it will introduce, in the worst case (the true target has finished the interaction process) distance lags between the observation received and the positions of the particles as they are stuck performing the interaction procedure; thus yielding to poor results (we recall that the correction step of the filtering process is based on the likelihood of the received observation data and the location of the particles).

In our implementation, we design an alternate solution which consists in using a threshold approach characterized by a *minimum distance of acceptance*, let us say d_{min}. In such a solution, it is always possible to resample (RPF) from a particle that is in interaction, hence obtaining a new particle. The new particle is kept in the resulting particle set if its distance to the original one is greater than the threshold d_{min} (we consider that the interaction is finished); otherwise the original particle is kept (and thus, the interaction will be pursued). By doing so, since the resampling step occurs after the correction step which integrates the observation received from the sensors, it is possible to efficiently manage noisy measurements stating that the target is far away from an object he was previously interacting with while he is still in interaction.

5.4.3 System Architecture

One of the major constraints under which the proposed solution has been implemented is to guarantee a response time of the system that allows to cope with real-time conditions. In an effort for satisfying this constraint, we design a distributed version of the filtering system [Brun et al., 2002], as illustrated in Figure 5.7, which is composed of:

- **a master node** in which the weighting as well as the resampling steps of the filtering process are performed;

- **several slave nodes** which contain an instance of the simulator and are in charge of the computation of the evolution (simulation) of the particle set over time. More specifically, a slave node may *physically* correspond to a CPU node of a computer which executes the corresponding instance of the simulator. Thereby, a physical machine can contain more than one slave node. Additionally, at each time step t, the particles are uniformly spread over the slave nodes. Thus, a slave node may be in charge of simulating more than one particle.

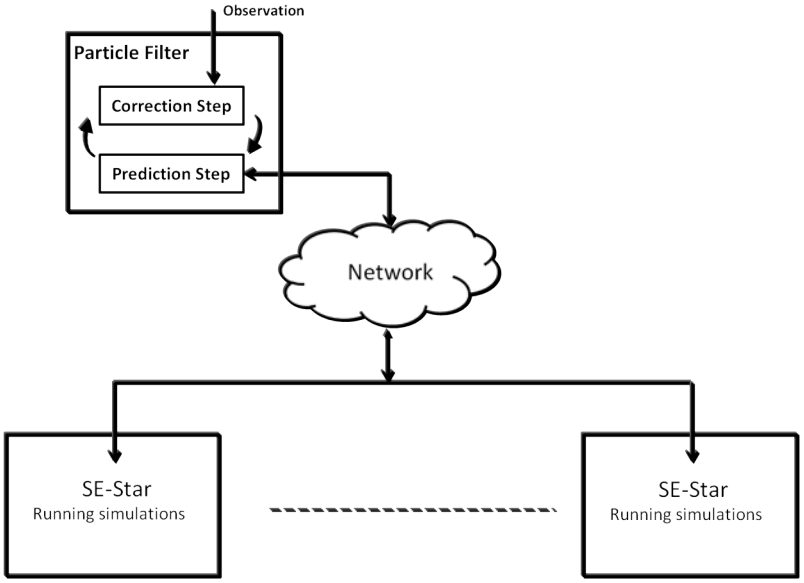


Figure 5.7: Implementation Architecture - A distributed particle filter. In the prediction step, the evolution of the particles is performed in parallel over the network.

Under this basis, the prediction step of the filtering process is performed in parallel over all the available slave nodes. Given the workload of the simulation system, this has the advantage of accelerating the execution time of the overall filter. The more the slave nodes, the better the response time of the filtering process. Despite the highlighted improvements, the response time of the system within this architecture is limited by incompressible computational factors such as the computation time required for simulating a particle, or network communications. In our implementation, we opt for an approach in which the resampling operation is centralized and performed on the master node. This approach has the interesting feature of decreasing the estimation error as the number of slave nodes increases. However, it requires that all the particles are returned back to the master node. Solutions have been proposed to operate local re-distributions on the subset of particles independently on each slave node [Brun et al., 2002]. But, in such a solution, a more complex procedure has to be performed in order to guarantee good filtering estimates.

5.5 Experimental Evaluations

This section is devoted to the experimental evaluation of the behavioral tracking system described in Section 5.3. For assessing the portability of the system, experiments have been conducted within a virtual environment as well as a real one. In what follows, we describe the experimental setups and the results obtained in both cases. But, first, let us introduce the performance criteria on which we rely for evaluating the quality of our system.

5.5.1 Performance metrics

The purpose of the system described in Section 5.3 consists in determining the behavior exhibited by a pedestrian under tracking. As stated previously, such a behavior is characterized by the location of the target as well as the activity he is performing within the environment. Therefore, the assessment of the system should take into account both characteristics. To this end, we use the following criteria:

- **the mean squared error on the trajectory (T-MSE)** — it is a metric commonly used for comparing two trajectories. In our setup, it measures the average, over time, of the squared distance between the estimated and the true target locations when available. Its expression, for one trajectory, is given by

$$T\text{-}MSE = \frac{\sum_{t=1}^{T} dist(pos(t) - pos'(t))}{T},$$

  where $pos(t)$ is the exact position of the tracked target at time t, $pos'(t)$ corresponds to the estimated location from the filter at the same time step and $dist(.,.)$ is a function computing the Euclidean distance between its arguments.

- **the similarity over the activity** — it measures the portion of time the best hypothesis of the system correctly matches the tracked target activity. In other words, this metric allows us to evaluate the behavioral similitude between the real activity currently undertaken by the underlying target with respect to the one computed by the system. This is done according to the following formula

$$A\text{-}Similarity = \frac{\sum_{t=1}^{T} 1_{Act(t)=Act'(t)}}{T},$$

  where $Act(t)$ is the exact activity performed by the target at time t and $Act'(t)$ is the most likely activity computed by the system at the same time step.

- **the robustness over the activity** — it measures the portion of time the system contains a valid hypothesis regarding the actual activity of the tracked target even if the concerned hypothesis is not the best one. It is calculated as

$$A\text{-}Robustness = \frac{\sum_{t=1}^{T} 1_{Pr(Act(t))>0}}{T},$$

  where $Act(t)$ is defined as previously, and $Pr(Act(t))$ is the probability of $Act(t)$ within the system.

As it can be noticed, the first metric focuses on the location estimation while the two others focus on the activity estimation. The following section is dedicated to the proper evaluation of the system on virtual scenarios. Real-world evaluations will be discussed later in Section 5.5.3.

5.5.2 Virtual-world based experiments

The main interest of performing evaluations in virtual environments resides in the ability to simulate events which hardly occur in normal circumstances in real life (e.g., fire, machine failures). The objective of this section is to demonstrate the performances of our system on designed virtual scenarios representing challenging situations in everyday life. We start by presenting, in Section 5.5.2.1, the setup under which the experiments are carried out. Then, from Section 5.5.2.2 to Section 5.5.2.4, we describe different scenarios on which the system is evaluated as well as the performances achieved.

5.5.2.1 Experimental Setup

In this section, we describe the environment, the behavioral model, the sensor network, the system parameters as well as the initial belief used in the experimental evaluation.

Environment

The environment in which virtual-world based experiments are performed represents a two-floor subway station (see Figure 5.8) equipped with a train door (with a train parked), an escalator connecting the two floors, a camera network, an ATM (in light green), a ticket machine (yellow), a vending machine (brown), ticket barriers (white), and exit barriers (red). A passenger (agent) may interact with these objects in order to fulfill his objectives. These interactions are usually not instantaneous and may require a certain duration depending on the agent's demand. For example, the time spent when withdrawing money from the ATM depends on the amount required. In the station, the cost of a ticket is 3 (money) units. In the meantime, drink/food can be purchased from the vending machine with 1 unit.

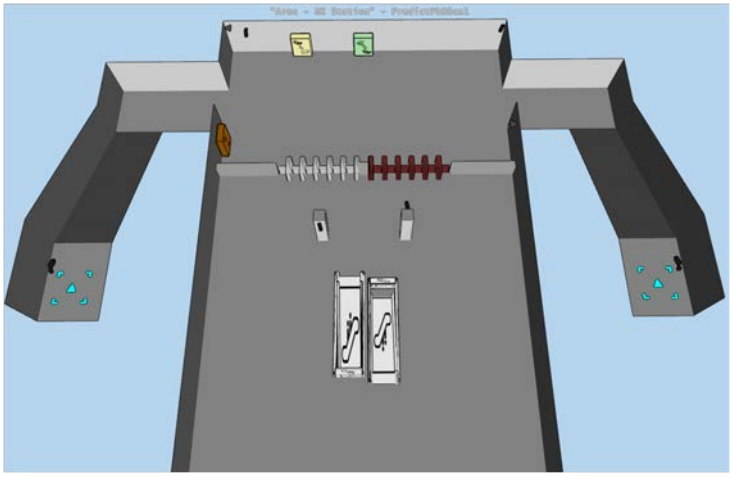


Figure 5.8: Virtual subway station.

Behavioral Model

Next, we consider a simplified model of a passenger for this station. The model assumes that a passenger may own money and/or tickets and that, during its lifetime, he may be motivated by the following main objectives: **taking the train**, **drinking**, **eating** and **leaving the station**. Depending on the resources owned, a passenger can exhibit various behaviors in relation with his current motivation. For example, a passenger willing to take a train may first buy a ticket if he does not already have a valid one. Also, he may be interested in getting some cash from the ATM if he does not have enough money[9] for buying the ticket. It is our task to infer the corresponding behavior solely from the passenger's observed trajectory. Despite the simplicity of the underlying model, tracking in such an environment remains challenging.

The passenger model has been designed in SE-Star, where each motivation is characterized by a numerical attribute taking values in the interval $[0, 2]$. The dynamics of these attributes along the time are encoded within the simulator using hysteresis functions and threshold mechanisms (see Equation 5.1 with $T_1 = 0.1$ and $T_2 = 0.70$). Besides, the model includes two integer attributes representing the resources owned (money and tickets). In the model, the resource attributes can only be modified after an interaction of the agent with an object in the environment.

Sensor Network

Regarding the camera network present in the environment, we consider two configurations, as illustrated in Fig. 5.9, under which the experiments will be conducted:

- **sensor configuration 1**: Here, the camera network is set to cover areas occupied by the ATM, the ticket machine and the vending machine;
- **sensor configuration 2**: Here, the camera network is set in such a way that the ATM, the ticket machine as well as the vending machine are not anymore under sensory coverage.

By considering both configurations, our intention is to assess the robustness of the tracking system in case of occlusions, specially within areas with high degree of interactions.

The sensor network is assumed to be subjected to observation noise characterized by a zero-mean Gaussian distribution whose standard deviation is set to 0.8, 0.8 and $0.1m$ (meters) for the x, y and z coordinates respectively.

System Parameters

When not explicitly specified, the system parameters are configured as follows:

- the radius r_h for approximating the error at the edge of covered/non-covered areas is set to $0.5m$;
- the number N of particles used to represent the belief regarding the target behavior is set to 2000;

[9] We assume that the ticket machine does not accept electronic cards.

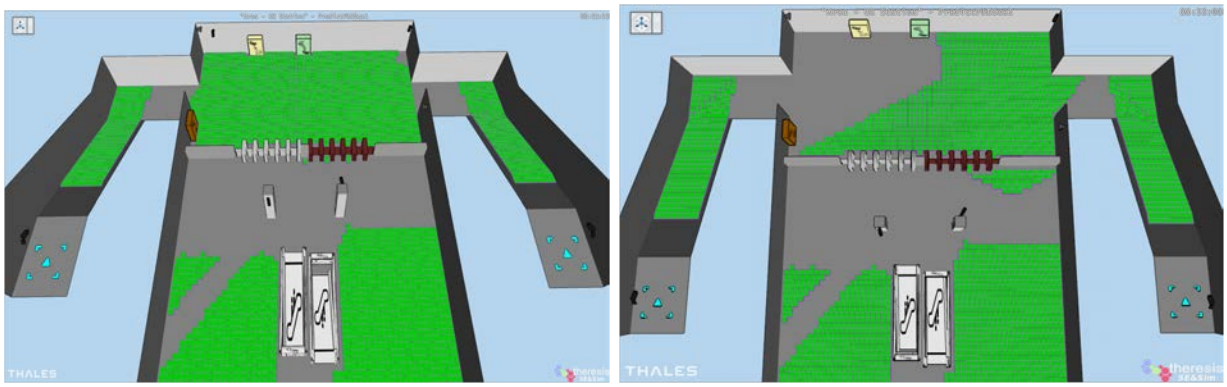


Figure 5.9: Sensor network configurations 1 (left) and 2 (right). Green squares represent areas under coverage.

- the threshold N_T, under which the resampling step is performed within the filtering process, is set to $0.75N$;
- the minimum distance of acceptance d_{min} used when resampling from a particle in interaction is set to $0.5m$;
- the experiments are performed on 2.0 GHz Intel Xeon E5-2650 CPU machines running under Ubuntu Linux (16 cores per machines). We have at disposal 4 such a machine which allow us to launch up to 48 slaves nodes (12 nodes per machine).

Initial Belief

For all the experiments conducted in the above described virtual environment, a time step within the tracking system corresponds to 0.5 seconds in real time and the system is initialized with the following belief regarding the passenger model's attributes:

- a passenger has 30% chances to own a number of tickets (resp. an amount of money) chosen uniformly in $\{1, \cdots, 3\}$ (resp. $\{1, \cdots, 5\}$), and nothing otherwise;
- the attribute related to the motivation "taking the train" is initialized using the Gaussian distribution $\mathcal{N}(0.8; 0.1)$ [10];
- the drinking/eating (resp. leaving) related attribute has 15% chances to be initialized using the Gaussian distribution $\mathcal{N}(1.40; 0.1)$ (resp. $\mathcal{N}(0.40; 0.1)$), otherwise it is set to 0;
- the velocity of a passenger is initialized using the Gaussian distribution $\mathcal{N}(1.42; 0.15)$.

The following sections are dedicated to the evaluation of the system on various virtual scenarios.

[10] At the initialization, when the number drawn from the provided distribution does not belong to the interval characterizing the underlying attribute, it is rejected and a new draw is performed.

5.5.2.2 Scenario 1: target with a fixed motivation

The purpose of this experiment is to highlight the inference capacity of our system on a simple scenario where the target under tracking is motivated, along the time, by a single objective within the environment. While the target motivation remains unchanged during the scenario, the activities he may undertake are highly correlated to the resources owned (amount of money, number of tickets). This information (the resources owned by the passenger) is unknown to the system, thus making the behavioral tracking process challenging. This section is organized as follows. We start by providing a description of the scenario. Then, we present the results obtained under each sensor network configuration introduced in Section 5.5.2.1. Finally, a performance analysis of the system is presented.

Scenario description

This scenario is about a passenger entering the station and whose motivation consists in taking the train. However, he does not have money, nor a valid ticket. Figure 5.10 shows the trajectory performed by the passenger and it serves as ground truth for the experiment. First, the passenger withdraws, from the ATM, some cash with which he buys a ticket at the ticket machine. He then crosses a ticket barrier before moving toward the platform where he enters the train.

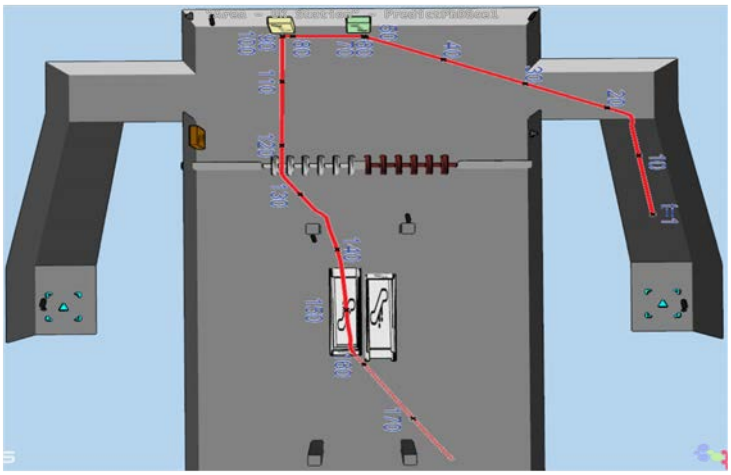
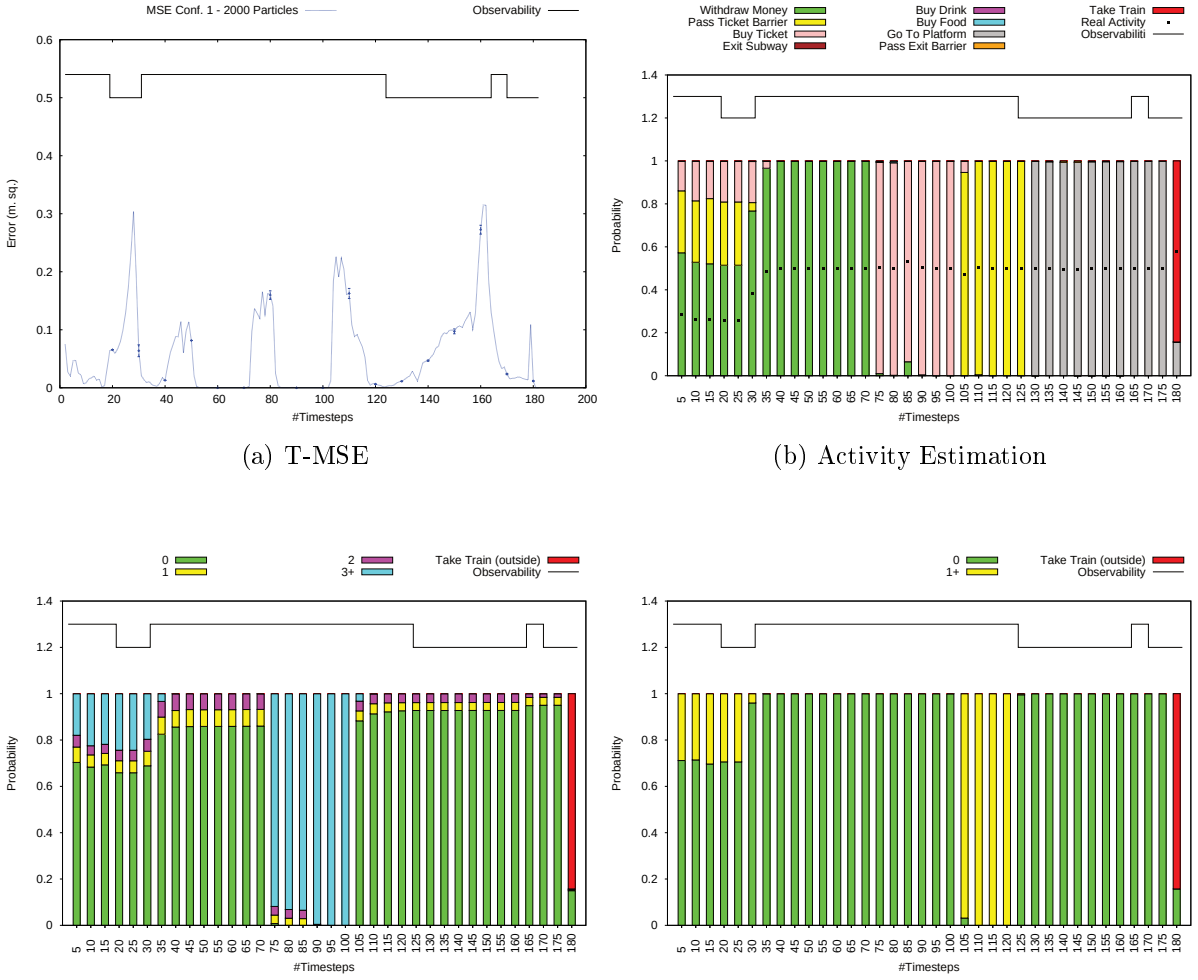


Figure 5.10: Passenger's trajectory (and corresponding time steps) in Scenario 1. The dashed line represents the portion of the trajectory performed in the second floor.

Sensor network configuration 1

The results presented in Figure 5.11 are an aggregation of the system outcomes obtained after carrying out 50 runs under sensor network Configuration 1. Figures 5.11a and 5.11b respectively show the T-MSE of the passenger location's estimate and the estimation of his activity over time. Additionally, Figures 5.11c and 5.11d illustrate the estimation of the different resources (the amount of money and the number of tickets) owned by the passenger.

From the results obtained, it appears that the passenger almost spends two-thirds of the time under sensory coverage while satisfying his motivation. Under these circumstances, the system is well predicting his behavior with an A-Similarity of 99.25%($\pm$0.38%) and an A-Robustness of



(a) T-MSE

(b) Activity Estimation

(c) Money Estimation

(d) Ticket Estimation

Figure 5.11: Scenario 1 - Experimental results under Configuration 1. Regarding Figure 5.11b, each bar represents the belief over the activities (modeled as slices on the bar) inferred by the system and the black point on each bar represents the true passenger's activity. Similarly, each bar in Figure 5.11c (resp. Figure 5.11d) represents the belief over the amount of money (resp. number of tickets) owned by the passenger. The label "1+" (resp. "3+") symbolizes numbers greater than or equal to 1 (resp. 3). For all the figures, the line on top represents time steps in which the target was either observed (up) or not (down).

100%. Having a close look at Figure 5.11a, the quality of the inference process can be appreciated in general thanks to the low values of the T-MSE over the time. Also, because the trajectory and, to some extent, the interactions mostly occur under sensory coverage, the system is quite confident, with its estimations resulting in very low variances. However, two phenomena can be pointed out in this graph: (1) periods of time where the T-MSE is null and (2) the presence of abrupt leaps usually at the end of those periods. In what follows, we provide an explanation of these phenomena over the time:

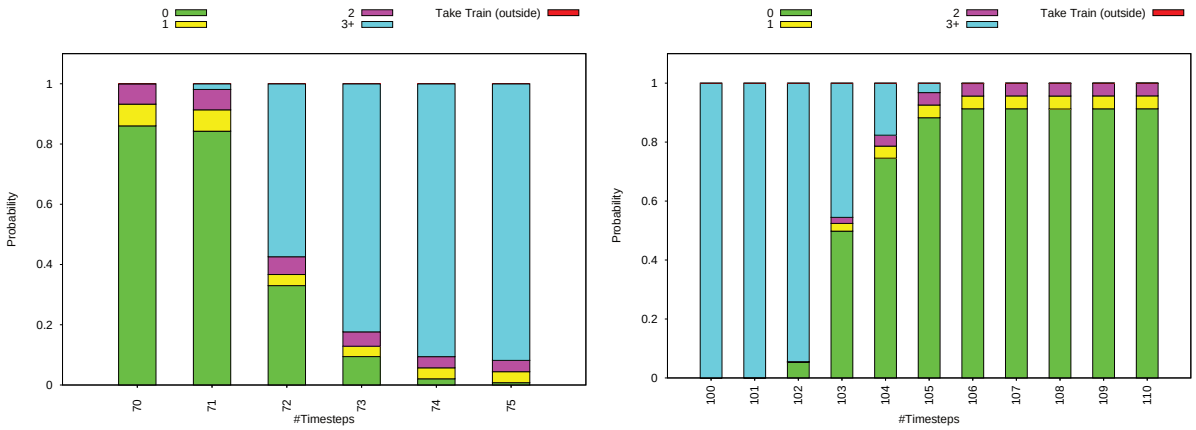
- The abrupt leap observed during the time interval $[18, 30]$ is explained by the fact that the system has not yet identified the exact behavior of the passenger. Indeed, as illustrated in Figure 5.11b, the passenger is not visible and the system maintains two main groups of particles: a group of particles in possession of a ticket (thus, they are going in direction of the ticket barriers) and a group of particles without ticket moving in an opposite direction towards (1) the ATM in order to withdraw cash or (2) the ticket machine for buying one. This divergence on the paths followed by the particles combined with the lack of observation data lead to an increase of the T-MSE.

- The nullity of the T-MSE observed during the time interval $[52, 71]$ (resp. $[85, 102]$) corresponds to periods of time where all the particles are standing in front of an object, here the ATM (resp. the ticket machine) while interacting with it. This can be well observed on Figure 5.11b where there is almost no doubt regarding the activity performed by the passenger at the corresponding time steps. Focusing on the interaction occurring at time step 52, it is interesting to point out that, despite the fact that all the particles are withdrawing cash from the ATM, not all of them are requesting the same amount of money. Indeed, depending on the amount already in possession (see Figure 5.11c on time interval $[50, 70]$), they may want to complete their money to at least 1 (for buying some food/drink) or at least 3 (for buying a ticket). At this time, the system is pretty sure that particles having more than 1 unit of money are not interested in buying food/drink otherwise they would have directly moved toward the direction of the vending machine.

- The abrupt leap observed at time step 72 (resp. 103), which occurs at the end of the interaction with the ATM (resp. the ticket machine), is because the end of the interaction is not synchronized for all the particles. The reasons of this desynchronization are twofold: (i) as previously highlighted, particles may request different quantities of resources and therefore, they have variable interaction durations; (ii) particles requesting the same amount of money may not necessarily start the interaction at the same time. This lack of synchronization can be well observed either in Figure 5.11d at time step 105 (end of interaction with the ticket machine) where there is a portion of particles having no ticket (they are still in interaction) or, in Figure 5.12 where the evolution of the amount of money owned by the particles is illustrated at the end of each interaction.

- Finally, the peak observed at $t = 158$ is because particles, at the end of the escalator, have different velocities and, with no containment restriction (particles were previously subjected to a containment restriction within the escalator), they can go in any direction. The T-MSE finally falls down as soon the target is re-observed.

The regularization procedure introduced at the resampling stage of the filtering process may occasionally result in inconsistencies within the system's estimates, as we explicitly inject ad-



(a) (b)

Figure 5.12: Illustration of the interaction desynchronization of the particles. Figure 5.12a illustrates the evolution of the amount of money owned by different particles at the end of the passenger's interaction with the ATM. Figure 5.12b shows the same evolution at the end of the passenger's interaction with the ticket machine.

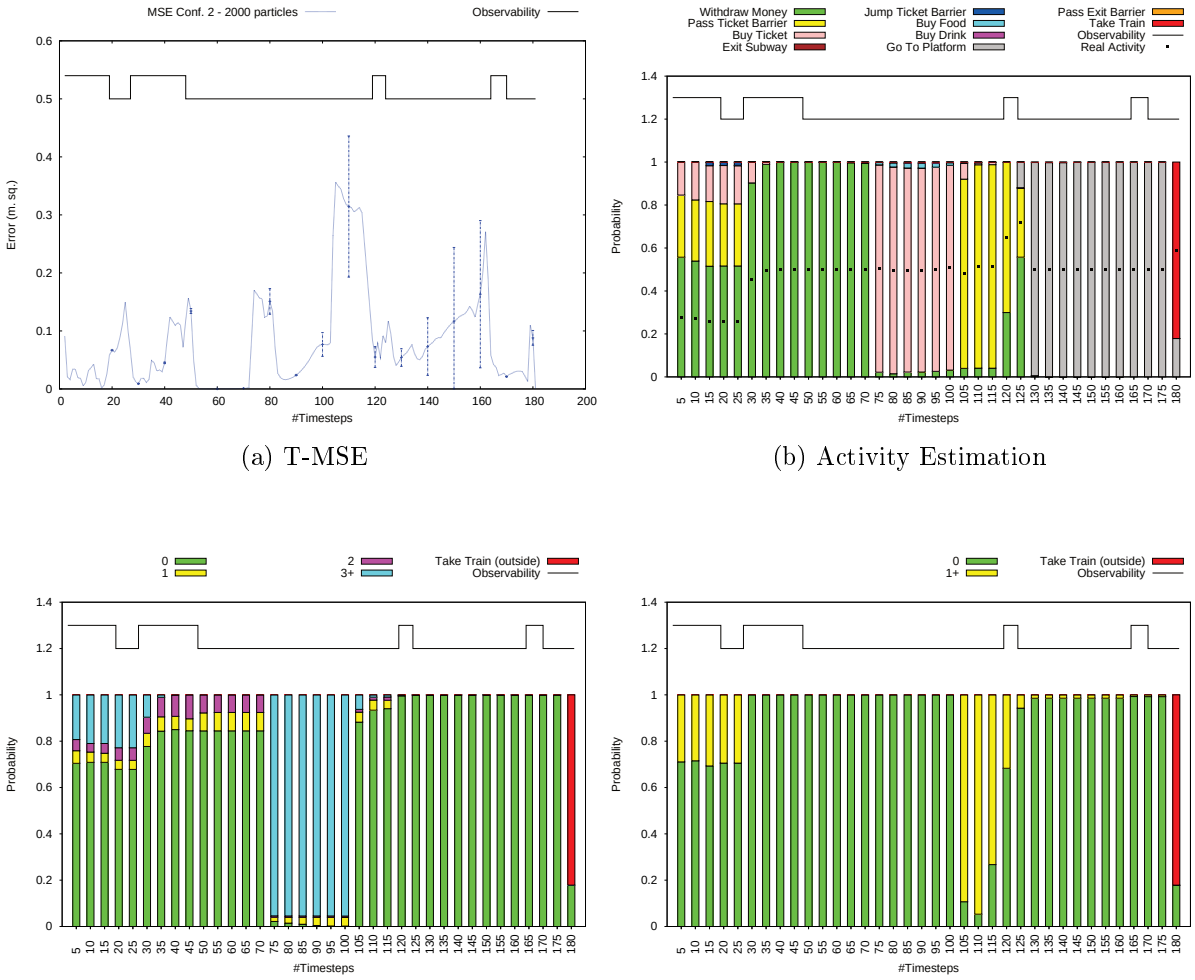
ditional noise in the system. This is typically what happens in Figure 5.11b at time step 85 where the system keeps a non negligible proportion of particles interested in withdrawing some cash from the ATM while it previously infers that almost all of them were interested in buying a ticket from the ticket machine. However, the system manages to subsequently prune such inconsistencies upon receiving new observation data.

Sensor network configuration 2

As previously, the results described in Figure 5.13 are obtained after running the system 50 times under Configuration 2. It comes out that the A-Similarity achieved by the system under this configuration is $95.86\%(\pm 1.70\%)$ despite the passenger being in a non-covered area two-thirds of the time. The two phenomena observed in Fig. 5.11a also appear in Figure 5.13a and the explanations provided in the previous paragraph also hold here. However, unlike Figure 5.11a, the T-MSE is not null during the time interval $[85, 102]$. This is because, after withdrawing cash from the ATM, a particle can either buy a ticket or get some drink/food from the vending machine. Since the passenger is not under sensory coverage, the system simply maintains all these hypotheses within the computed belief (see Figure 5.13b) leading to the peak observed at $t = 103$ in Figure 5.13a. This is an important feature for the robustness of our system. Also, we notice that the variance is higher with respect to the one obtained for Configuration 1, a fact explained by the diversity maintained within the particle set.

Focusing on the time interval $[119, 126]$ in which the passenger briefly enters an area under sensory coverage after a long period of time without being observed, a drop of the T-MSE is observed due to the correction performed by the filtering process. As illustrated in Figure 5.13b at time step $t = 120$, most of the retained particles are interested in the same activity as the tracked passenger ("cross the ticket barrier"). Also, there are still particles that, despite their proximity with the passenger position, are interested in another activity ("withdraw money"). At time step 123 the passenger re-enters a non-covered area and the system is not receiving observations from sensors anymore. However, after few time steps (130), the system manages to infer the exact

activity of the target. Indeed, particles exhibiting the wrong behavior (withdrawing money) need to go back and pass through a covered area, and thus be observable. This contradicts with the observed situation (the system would have received observation data from the sensors) and, as a consequence, the system automatically eliminates them.



(a) T-MSE

(b) Activity Estimation

(c) Money Estimation

(d) Ticket Estimation

Figure 5.13: Scenario 1 - Experimental results under Configuration 2

Finally, we are interested in the rank of the exact passenger's activity within the system for both configurations and the result is depicted in Figure 5.14. It appears that, when not ranked as the best hypothesis, the actual activity occupies the second position most of the time within the system.

Impact analysis of N

This section analyzes the system performance when varying one of its parameters, that is, the number N of particles used to represent the belief over the tracked-target behavior. To this end, we consider running the system with N equal to 100, 500, 1000 and 2000 respectively. For each setting, we carry out 50 runs for both sensor network configurations. The result is presented in Table 5.1. While the effects of the occlusions can be pointed out in general (the

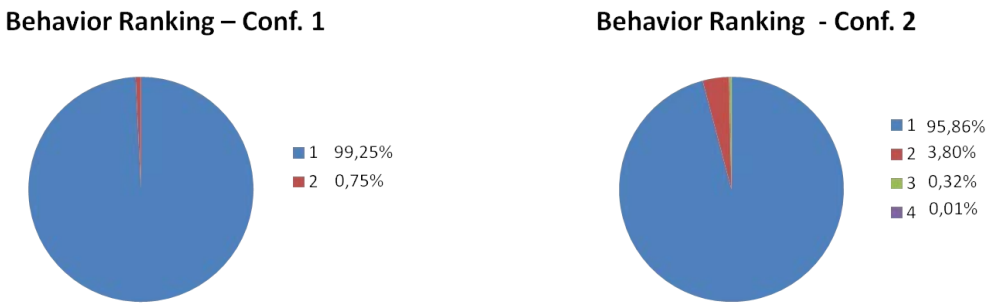


Figure 5.14: Scenario 1 - Activity ranking under Conf. 1 (left) and Conf. 2 (right).

A-Similarity metrics obtained in Configuration 2 are smaller with respect to the ones obtained in Configuration 1), the performance of the system increases with the number of particles used. Also, we obtain a value of 100% for the A-Robustness metric whatever N, meaning that the initial belief provided as input to the system well serves the inference process for this specific scenario. Finally, given the computational resources at disposal, the system manages to achieve a real-time response for all the different settings.

Table 5.1: Scenario 1 - Performance Analysis

	Configuration 1		**Configuration 2**	
N	A-Similarity (%)	A-Robustness (%)	A-Similarity (%)	A-Robustness (%)
100	87.62 ($\pm$2.93)	100	88.40 ($\pm$3.87)	100
500	89.11 ($\pm$3.37)	100	89.92 ($\pm$4.52)	100
1000	97.08 ($\pm$2.25)	100	92.35 ($\pm$4.41)	100
2000	99.25 ($\pm$0.38)	100	95.86 ($\pm$1.70)	100

Summary

In summary, we have shown through this scenario that the proposed system efficiently infers the behavior of a tracked pedestrian, in terms of both location and activity estimations, while reasoning on high-level data such as the resources owned by the target (money, ticket). Also, because of the contextual information represented by the agent behavioral model as well as the smart objects designed within the environment, it is possible to have, at each time step, a clear view of what is happening in the scene. Finally, the system provides accurate results even in case of long periods of occlusion. To highlight the advantages of our system with respect to the existing literature, the next section is dedicated to the comparison of our system with a DBN-based approach [Wilson and Atkeson, 2005] [Manfredotti et al., 2011] on a more challenging scenario.

5.5.2.3 Scenario 2: target with a varying motivation

The purpose of this experiment is to demonstrate the advantages of the proposed approach with respect to an approach where the contextual information is encoded within a DBN. To

this end, we consider a scenario in which the motivation of the target under tracking evolves with the time. In the previous scenario, although the target performed several activities, these ones contributed to the satisfaction of a single motivation, that is, "*taking the train0*". In this scenario, the target is able to modify its motivation over time. Because of this modification of the motivation, the inference process is challenging and the tracking system needs to principally rely on prior environmental knowledge, for better estimate accuracies. This section is organized as follows. After describing the considered scenario, we introduce the settings used in the DBN-based approach. Then, we describe the results obtained for both approaches under each sensor network configuration (cf. Section 5.5.2.1). Finally, a performance analysis of our system is presented.

Scenario description

This scenario is about a starving passenger entering the station and whose motivation consists in taking the train after buying some food. However, he owns nor money, nor a valid ticket. Figure 5.15 shows the trajectory performed by the passenger and used as ground truth for the experiment. First, the passenger withdraws cash (from the ATM) with which he buys some food at the vending machine. After satisfying his hunger, he moves back to the ATM for withdrawing cash again. With the money newly obtained, he buys a ticket at the ticket machine. Then, he crosses a ticket barrier and moves toward the platform where he enters the train.

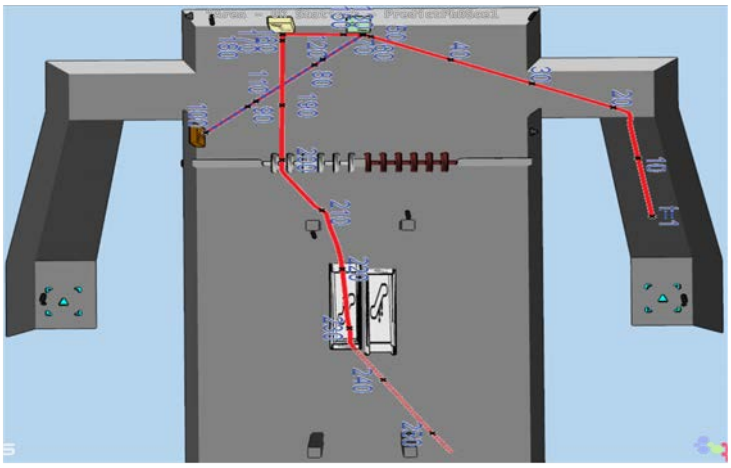


Figure 5.15: Passenger's trajectory (and corresponding time steps) in Scenario 2.

Settings of the DBN-based approach

In the DBN-based approach used for later comparisons, the contextual information regarding the environment is represented using the DBN as illustrated in Figure 5.16. In this experiment, the complexity of the illustrated DBN is similar to the ones encountered in the literature [Wilson and Atkeson, 2005] [Manfredotti et al., 2011]. In the model, the location of the target at the next time step depends on his location as well as his activity at the previous time step. Also, the activity at the next time step only depends on the activity at the previous time step. As it can be noticed, the designed DBN verifies the Markov property of order 1 and thereby, it embeds less memory regarding previous activities. While a planner is used for computing the trajectory of the

target given his current location and the activity he is interested in, the knowledge regarding the activity sequence within the environment is provided by a conditional probability table described in Table 5.2. These probabilities have been manually set for the purpose of the evaluation in such a way to reflect the set of allowed sequence of activities within the environment. For example, after withdrawing money from the ATM, a given passenger has 50% chances to buy a ticket, 20% chances to buy some food (resp. drink) and 10% chances to exit the station. This could be viewed as a general statistics obtained after the analysis of several passenger patterns within the station. In what follows, we present the results obtained with the two approaches.

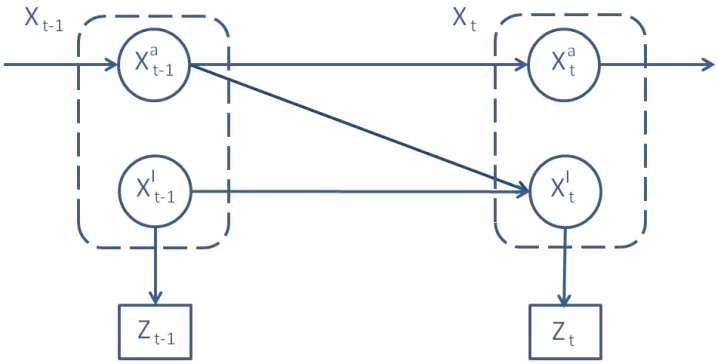


Figure 5.16: Scenario 2 - DBN-based model. $\mathbf{x}^l$ and $\mathbf{x}^a$ respectively symbolize the location and the activity of the target.

Table 5.2: DBN-based context representation. The table describes the transition probability of the activity within the subway station

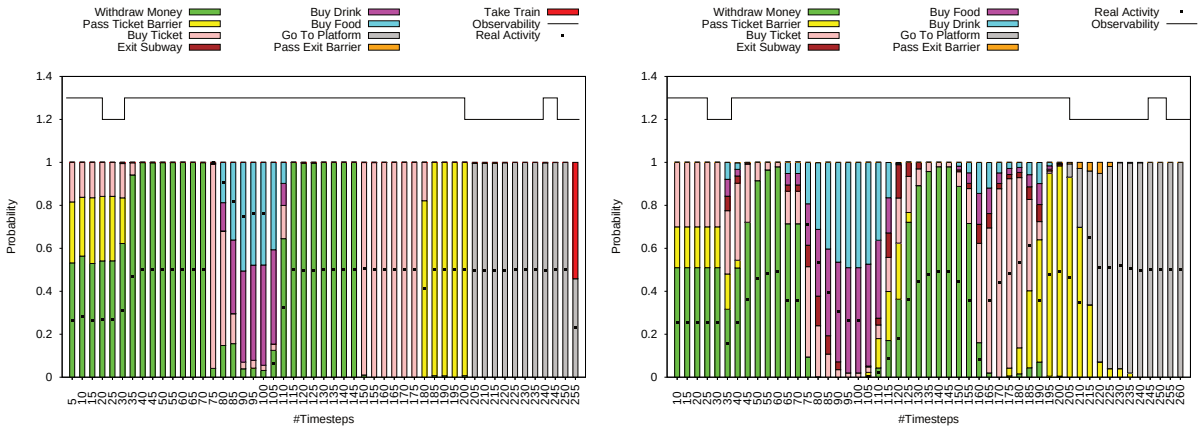
	Next Activity									
Current Activity	Withdraw money	Buy ticket	Buy food	Buy drink	Pass ticket barrier	Go to platform	Pass exit barrier	Take train	Exit subway	Outside
Withdraw money	-	0.50	0.20	0.20	-	-	-	-	0.10	-
Buy ticket	0.10	-	0.10	0.10	0.60	-	-	-	0.10	-
Buy food	0.15	0.15	-	0.20	0.40	-	-	-	0.10	-
Buy drink	0.15	0.15	0.20	-	0.40	-	-	-	0.10	-
Pass ticket barrier	-	-	-	-	-	0.90	0.10	-	-	-
Go to platform	-	-	-	-	-	-	0.05	0.95	-	-
Pass exit barrier	0.15	0.05	0.15	0.15	-	-	-	-	0.50	-
Take train	-	-	-	-	-	-	-	-	-	1.00
Exit subway	-	-	-	-	-	-	-	-	-	1.00
Outside	-	-	-	-	-	-	-	-	-	1.00

Experimental Results

The following results are obtained after performing 50 runs for each sensor network configuration introduced in Section 5.5.2.1. Figures 5.17 and 5.18 respectively show the estimation of the passenger's activity under Configuration 1 and Configuration 2, while Table 5.3 describes the corresponding performance metrics for both approaches. As expected, the proposed approach outperforms the DBN-based approach. While the A-Similarity resulting from the two approaches are relatively close under Configuration 1, their difference is high under Configuration 2. Indeed, under Configuration 1, the passenger is observed most of the time and all

the interactions resulting from the change of the motivation (from "buying food" to "taking the train") are performed under sensory coverage (see Figure 5.17b in the time interval $[105, 200]$). Thereby, both approaches manage to successfully identify the behavior of the passenger.

Under Configuration 2, interactions resulting from the change of motivation mostly occur within areas not observed by the sensor network. However, the difference observed in the performance of both approaches can be explained by the fact that our approach integrate, through the agent behavioral model, high-level (temporal) information regarding the evolution of the thirsty (hunger) of the passenger. More precisely, after our system considers the hypotheses in which the passenger is interested in buying some food/drink (time interval $[65, 100]$), it progressively eliminates them within its belief while the target is still in non-covered areas as illustrated in Figure 5.18a during time interval $[105, 120]$. This is in accordance with what happens in practice as it is more likely that someone who has just satisfied his hunger (resp. thirst) would not be immediately interested in eating (resp. drinking). As a testimony of the proper dynamics of the internal state of the agent, these hypotheses re-appear within the system after a certain period of time (see Figure 5.18a during time interval $[160, 185]$) before being completely pruned as soon as the passenger is re-observed. Unlike our approach, the DBN-based approach keeps maintaining hypotheses regarding these activities (see Figure 5.18b after time step 120) resulting thus in a performance degradation. Basically, nothing prevents the DBN-based system to envision unlikely activity sequences as for example the following loop: withdraw money, buy food, withdraw money. This example can be restrained by increasing of 1 the order of the Markov property within the DBN. However, in general, it may always be possible to find inconsistent examples embedded within the so-obtained model.



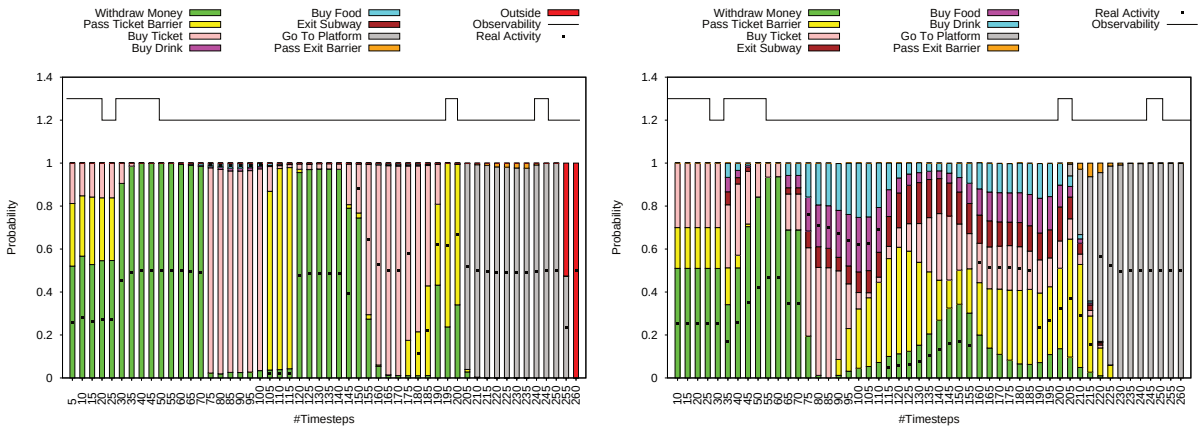
(a) Our approach

(b) DBN-based Approach

Figure 5.17: Scenario 2 - Activity estimation under Configuration 1

Impact analysis of N

In the previous section, we compared our system against a DBN-based system on a challenging scenario using $N = 2000$ particles. This section analyzes the performance of our system on this complex scenario when varying the number N of particles used to represent the system belief. Table 5.4 shows the system outcomes after performing 50 runs with $N = 100$, 500, 1000 and 2000 respectively. While it was possible to obtain a result with $N = 100$ particles for the tracking scenario described in Section 5.5.2.2, it appears that, for this scenario, the degeneracy



(a) Our approach (b) DBN-based Approach

Figure 5.18: Scenario 2 - Activity estimation under Configuration 2

Table 5.3: Scenario 2 - Comparative results

	Configuration 1		**Configuration 2**	
Methodology	A-Similarity	A-Robustness	A-Similarity	A-Robustness
Our Approach	92.54 (±1.54)	100	76.63 (±0.99)	100
DBN-based Approach	89.23 (±5.23)	100	60.99 (±4.04)	100

phenomenon occurs ($N = 100$ particles are insufficient to represent all the plausible behaviors of the tracked target) and the system fails to correctly proceed with the tracking. This illustrates the complexity of this scenario with respect to the previous one given the initial belief provided as input to the system. Nevertheless, the more the particles used, the better the tracking efficiency. Finally, as it was already the case in the previous scenario, the systems achieves a real-time response for all the different settings given the computational resources at disposal.

Table 5.4: Scenario 2 - Performance analysis of our approach

	Configuration 1		**Configuration 2**	
N	A-Similarity (%)	A-Robustness (%)	A-Similarity (%)	A-Robustness (%)
100	-	-	-	-
500	78.95 (±0.47)	89.33 (±0.69)	70.31 (±0.49)	88.27 (±0.18)
1000	92.11 (±2.77)	100	75.63(±0.63)	100
2000	92.54 (±1.54)	100	76.63 (±0.99)	100

Summary

In summary, we have shown, through this scenario, the advantage of relying on agent-based behavioral simulators rather than "simple" graphical models (DBNs) for representing the contex-

tual information useful within the inference process. Indeed, the expressiveness of such simulators is higher than the one of those DBNs. We acknowledge the possibility of designing a DBN which can embed the temporal information regarding the thirsty (or the hunger) of the target as illustrated in this scenario. However, the outcome of such a design process may easily prove to be a large and complex data-structure difficult to manage in practice.

In the next section, we are interested in demonstrating the capacity of our system to effortlessly integrate, in the inference process, exogenous events that may arise within the environment.

5.5.2.4 Scenario 3: exogenous events

The objective of this experiment consists in demonstrating the capacity of the system to effortlessly take into account exogenous events which occur within the environment and which have a significant impact on the behavior of the tracked target. To this end, we consider a scenario simulating an emergency, thus forcing the passenger to simply abandon his current motivation and leave the station. As it was the case for the scenario described in Section 5.5.2.3, the motivation of the target under tracking evolves with time. However, in this scenario, the reason of this evolution is external to the target. This section is organized as follows. After describing the considered scenario, we present the results achieved under each sensor configuration (cf. Section 5.5.2.1).

Scenario description

This scenario involves a passenger entering the station and whose motivation consists in taking the train. However, he has nor money, nor a valid ticket. After withdrawing cash from the ATM, he moves in the direction of the ticket machine where he is about to buy a ticket. In the mean time (time step 80), a fire alarm is triggered in the environment and the passenger subsequently leaves the station. Figure 5.19 illustrates the trajectory followed by the passenger, which is used as ground truth for the experiment.

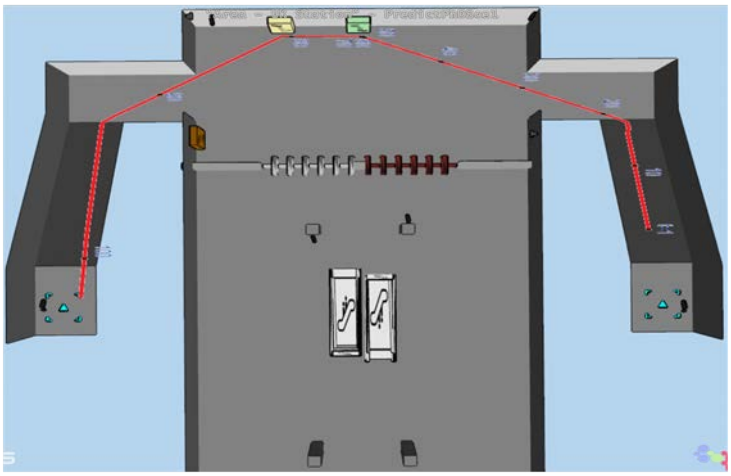
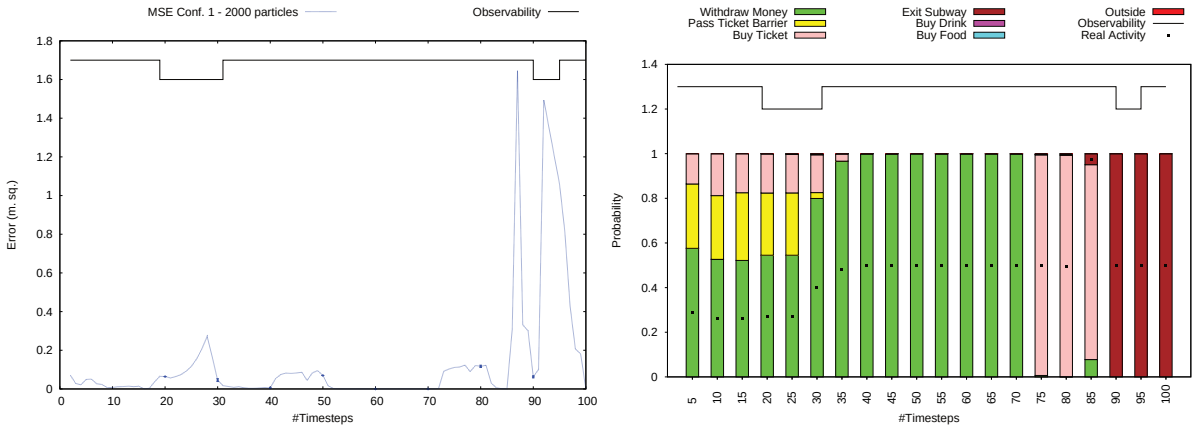


Figure 5.19: Passenger's trajectory (and corresponding time steps) in Scenario 3.

Experimental Results

The results described thereafter are obtained after running the system 50 times with $N = 2000$ particles for the two sensor network configurations at hand. Like it was the case in the previous scenarios, the response time of the system verifies the real-time constraint given the computational resources at disposal. Figures 5.20 and 5.21 show the T-MSE as well as the estimation of the passenger's activity under Configuration 1 and Configuration 2 respectively while Table 5.5 describes the resulting metrics.

It comes out that the system manages to efficiently infer the behavior of the underlying passenger despite the exogenous event and the presence of areas not covered by the sensors. As it can be observed for both configurations, until time step 80 where the alarm is triggered, the shapes of the T-MSE curves (Figures 5.20a and 5.21a) are similar to the ones obtained, under the same tracking conditions, for Scenario 1 in Section 5.5.2.2 (see Figures 5.11a and 5.13a). Upon reception of the signal resulting from the alarm (the system is fed by the changes occurring in the environment), the particles automatically process the information and progressively modify their current motivation for exiting the station as illustrated in Figure 5.20b and 5.21b.

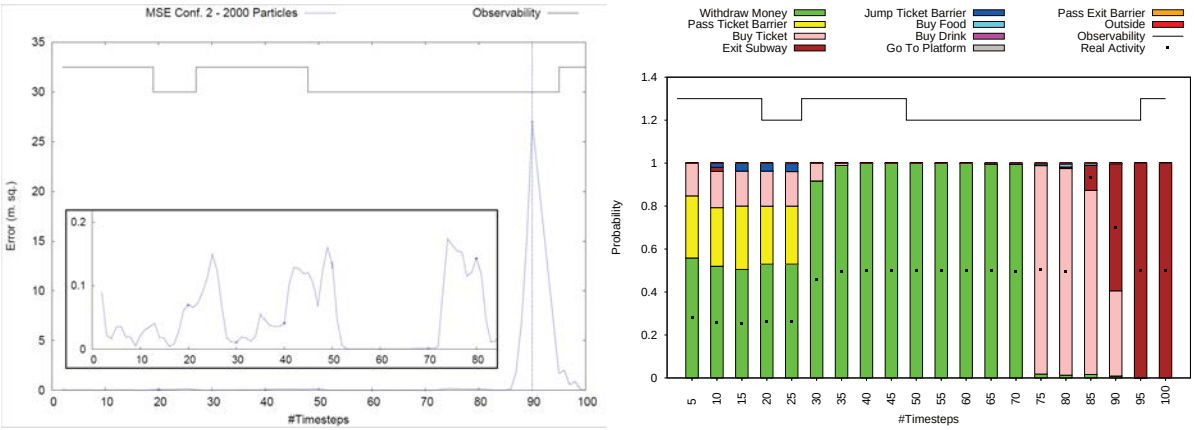


(a) T-MSE (b) Activity Estimation

Figure 5.20: Scenario 3 - Experimental results under Configuration 1

Focusing on Configuration 1, the peak observed during the time interval $[85, 90]$ in Figure 5.20a is the direct consequence of the fact that the information processing, within the particle set, is individualized and depends on each particle characteristics. Indeed, all the particles do not decide to leave the station at the same time. However, because the passenger is currently under sensory coverage, the T-MSE gradually decreases. Nevertheless, the T-MSE increases at time step 90 as soon as the passenger is not observed anymore. This is mainly due to a difference of velocity between particles even if all of them are interested in leaving the station. Subsequently, fast particles are progressively eliminated as they enter in advance areas under sensory coverage while the passenger is still unobserved, thus leading to a drop of the T-MSE.

Similarly in Configuration 2, there is a leap of the T-MSE starting from time step 85 in Figure 5.21a. However, unlike Configuration 1, its height is larger. This is explained by the lack of observation data as the passenger is within a non-covered area. Indeed, besides the fact that particles do not decide to leave the station at the same time, they may want to use one of the two environment exits. As neither the particles, nor the passenger is observed (they are within non-covered areas), the system has no choice than maintaining all of them until those moving in



(a) T-MSE (b) Activity Estimation

Figure 5.21: Scenario 3 - Experimental results under Configuration 2

the wrong exit (with respect to the choice of the passenger) enter the covered areas at time step 90. Since the passenger is still unobserved, while some particles are now in a covered area, the system progressively eliminates invalid particles and, as a consequence, the T-MSE decreases.

Table 5.5: Scenario 3 - Numerical Results

Configuration 1		Configuration 2	
A-Similarity (%)	A-Robustness (%)	A-Similarity (%)	A-Robustness (%)
96.96 (±0.61)	100	93.63 (±3.09)	100

Summary

In summary, through the previously three virtual scenarios, we have demonstrated the advantages of our system with respect to existing STAR's approaches [Wilson and Atkeson, 2005, Manfredotti et al., 2011] in the literature as they rely on DBNs for modeling the environmental context. More precisely, the conducted experiments highlighted that the expressiveness capacity of agent-based behavioral simulators for representing contextual knowledge is higher than the one of the DBNs. Also, it has been shown that the proposed solution is able to effortlessly integrate dynamical changes of the environmental context, a dimension which has not been addressed in previous works. Moreover, the results detailed through these experiments showed very good filtering performances in terms of both location and activity estimations and this still holds even in cases of long periods of occlusion. However, at this stage, one question of particular interest is "how good will the system perform on a real study case?". The next section is devoted to this question.

5.5.3 Real-world based experiments

The purpose of this section is to study the performances of the proposed behavioral tracking system on real situations. In what follows, Section 5.5.3.1 describes the setup under which

the evaluations are performed while Sections 5.5.3.2 and 5.5.3.3 present the different scenarios considered as well as the performances achieved.

5.5.3.1 Experimental Setup

In this section, we describe the environment, the behavioral model, the sensor network, the system parameters as well as the initial belief used in the experimental evaluation.

Environment

To conduct real-world experiments, the surrounding of our office is considered as the reference environment. We provided SE-Star with a 3D representation of the building. The area of the building concerned by the experimental evaluation is depicted in Figure 5.22. This area is equipped with camera sensors, a coffee machine, a copier and a paper tray for printing operations. These objects are modeled using the "smart object" concept and incorporated within the 3D environment in such a way that their locations match with the real environment. In Figure 5.22, we use a common (simple) box graphical model[11] for representing the last three objects (coffee machine is in brown, the printer in cyan, and the tray in light green). In the design, we specify a fixed duration of $15sec.$ for the copier to perform a single operation once it has received a request (of doing so). Similarly, we set up a fixed duration of $10sec.$ for the coffee machine to make a drink and consider that a given employee can make drinks until he is not thirsty anymore. Additionally, the area under consideration contains several rooms in which an employee can go for different purposes. Examples of such rooms include the toilet, the server room, the open space and, of course, the employee's office.

Behavioral Model

To carry out the experimental evaluation, we rely on a simplified employee model. In such a model, an employee can only go into a single office room (the one with the purple triangle in Figure 5.22). Besides, he can go to the toilet if needed as well as to the server room and the open space. For simplicity, we set a fixed time to spend within the toilet, the server room and the open space. Additionally, he can obviously make use of the coffee machine and/or perform some printing operations (possibly after getting some papers from the tray). The model is designed in SE-Star using six numerical attributes, each one associated with a distinct employee's motivation (office, toilet, server room, open space, printing, coffee machine). The domain for these attributes is the interval $[0, 2]$, and their dynamics is encoded within the simulator using hysteresis functions and threshold mechanisms (see Equation 5.1 with $T_1 = 0.1$ and $T_2 = 0.70$). Also, the model includes an additional integer attribute representing the quantity of paper possessed by the employee.

Sensor Network

The real environment is equipped with a sensor network which is set up to partially cover the environment. This network is simulated within the virtual environment (3D representation) and the resulting virtual sensor network is calibrated in such a way that the areas under sensory

[11] This is just an appearance model with no incidence on the outcome of the system's result.

Figure 5.22: 3D representation of the office building environment. The green squares represent areas under sensory coverage. The purple and green triangles are used to model the activities associated with the office room and the server room respectively.

coverage correspond in both worlds (see Figure 5.22 for sensory coverage). Moreover, we rely on a Thales proprietary software for (a) detecting people within a video stream and (b) computing their (noisy) positions which are expressed within the coordinate system of the virtual world as shown in Figure 5.23.

The sensor network is assumed to be subjected to observation noise characterized by a zero-mean Gaussian distribution whose standard deviation is set to 0.8, 0.8 and $0.1m$ (meters) for the x, y and z coordinates respectively.

System Parameters

When not explicitly specified, the system parameters used in the evaluations are the same as the ones introduced in Section 5.5.2.1. We recall that, by default,

- the radius r_h for approximating the error at the edge of covered/non-covered areas is set to $0.5m$;
- the number N of particles used to represent the belief regarding the target behavior is set to 2000;
- the threshold N_T, under which the resampling step is performed within the filtering process, is set to $0.75N$;
- the minimum distance of acceptance d_{min} used when resampling from a particle in interaction is set to $0.5m$.

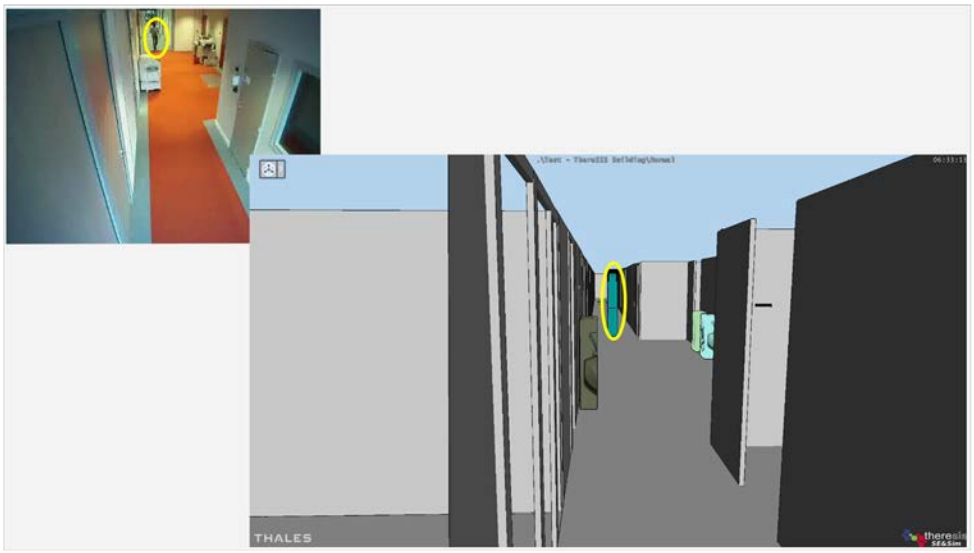


Figure 5.23: Sensor network calibration between virtual and real environments. The position of the target detected (yellow ellipse on top image) in a video stream is expressed in the coordinate system of the 3D virtual representation of the environment (yellow ellipse on bottom image).

Initial Belief

For all the runs performed in the above described environment, the following belief is provided as input to the system:

- an employee has 30% chances to possess a quantity of paper chosen uniformly in $[1, 3]$, and nothing otherwise;
- all the attributes except the one related to the "office" motivation are initialized using the Gaussian distribution $\mathcal{N}(0.80; 0.1)$;
- the "office" related attribute is initialized using the Gaussian distribution $\mathcal{N}(0.60; 0.1)$;
- the velocity of an employee is initialized using the Gaussian distribution $\mathcal{N}(1.42; 0.15)$.

The following sections are dedicated to the assessment of the system on challenging scenarios in this real-world environment.

5.5.3.2 Scenario 1

The purpose of this experiment is to highlight the inference capacity of our system on a real setting. To this end, we consider a scenario representing a situation occurring in the employee everyday life at the office and we study the system efficiency. This section is organized as follows. After describing the scenario, we present the results obtained when using the default parameters introduced in Section 5.5.3.1. Finally, an analysis of the system performance is provided.

Description

In this scenario, an employee leaves his office and goes to the toilet where he spends some time. After that, he moves towards the coffee machine where he makes some drinks. Finally, he returns back to his office. A sketch of the corresponding trajectory within areas under sensory coverage appears in Figure 5.24. For later comparison with the system output, we manually tag each time step of the trajectory with the corresponding employee's behavior.

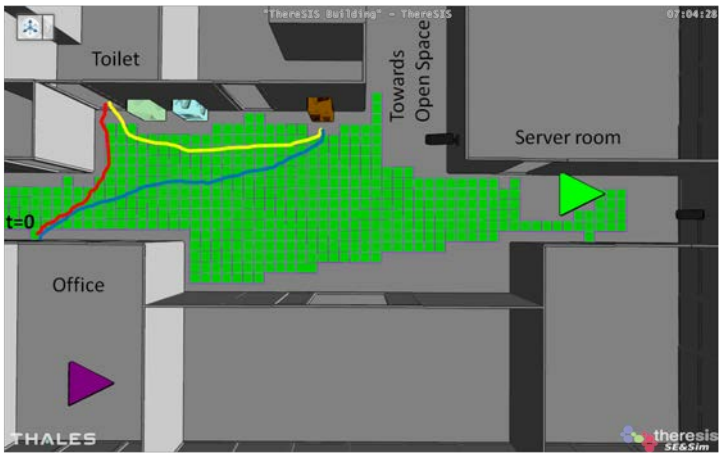


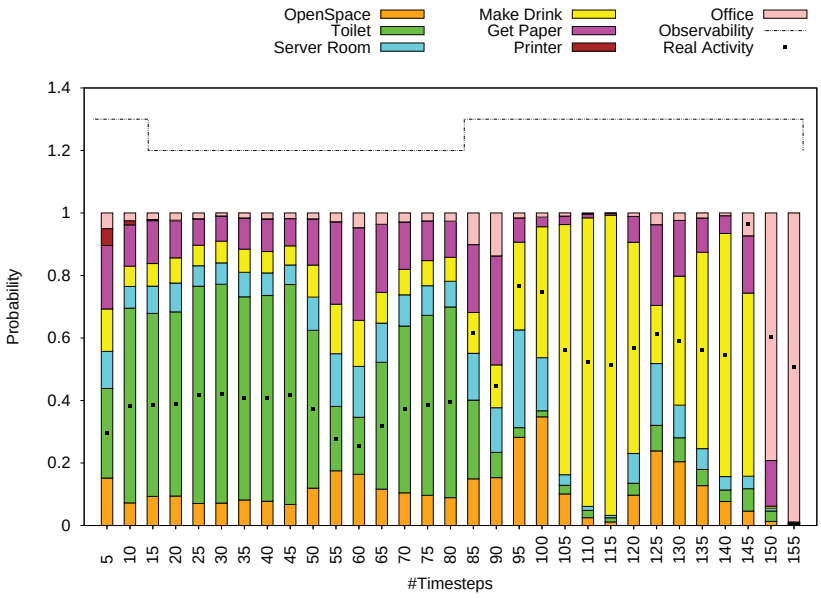
Figure 5.24: Scenario 1 - Sketch of the employee's trajectory within areas under sensory coverage: From the office to the toilet (red), from the toilet to the coffee machine (yellow) and from the machine to the office (blue).

Experimental Results

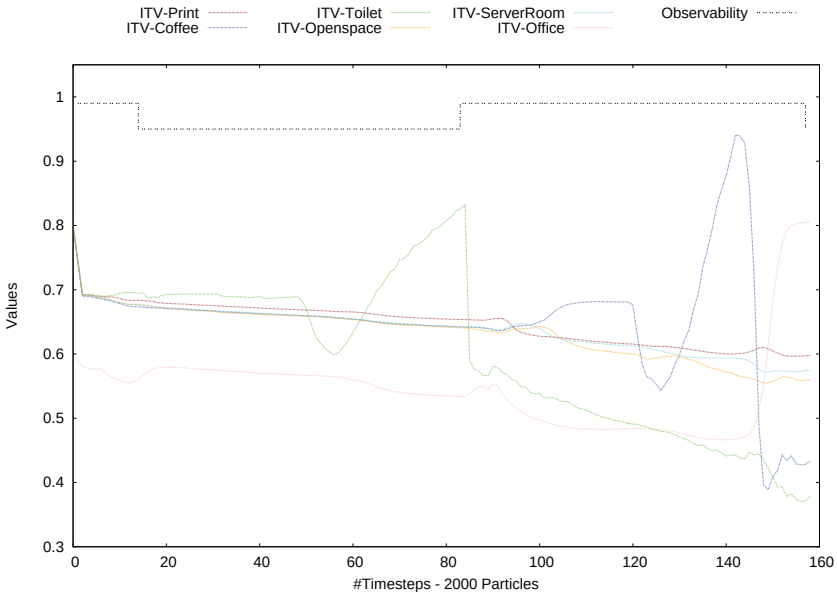
The results presented in Figure 5.25 are an aggregation of the inference outcomes obtained after carrying out 20 runs of the system with the same video data input. Since we do not have the ground truth trajectory (only noisy detections from sensors are available), we restricted the system's evaluation criteria on the A-Similarity and the A-Robustness ratios. Figures 5.25a and 5.25b respectively show the employee's activity estimation and the weighted mean of the model attributes (for all particles) over the time. Also, it appears that the response time of the system satisfies the real-time constraint given the computational resources at disposal.

Focusing on Figure 5.25b, we can clearly identify three main periods in which the attribute associated with the correctly predicted behavior has the highest value. Rapidly, based on the trajectory followed by the employee, the value associated to the "toilet" related attribute becomes greater than the values of other attributes. However at $t = 58$, we observe a basin for the "toilet" related attribute. The reason is that, as we assumed a fixed duration in the toilet, there are some particles for which this time is elapsed and therefore they are willing to go out from the toilet. Since the the employee is still inside, and, because the system does not receive observation data from the sensors (as it should be the case if, like the particles, the target was out), those particles are eliminated, and thus, the correct hypothesis (the one stating that the target is still inside the toilet) is strengthened at the same occasion. This situation corresponds to an increase of the value of the corresponding attribute which finally falls down as soon as the target is re-observed when exiting the toilet.

At this point, the system maintains several hypotheses regarding the next activity of the



(a) Activity estimation



(b) Model attributes' evolution

Figure 5.25: Scenario 1 - Experimental results

employee and, based on the trajectory followed by the employee, it manages to infer the exact one. As it can be observed at time step 97 on Figure 5.25b, the "coffee" related attribute dominates all the others. However, as it was already the case with the "toilet" related attribute, a basin is observed at $t = 127$ for the "coffee" related attribute. Similarly, the explanation comes from the fact that some particles have finished their interaction with the coffee machine while the tracked employee has not. Because the employee remains in proximity of the coffee machine (and therefore the returned noisy locations), the system automatically infers that the interaction with the machine is not over. At the end of the interaction with the coffee machine (the fall of the corresponding attribute's value), there is an increase of the "office" related attribute's value meaning that the system believes that the employee is returning back to the office.

Finally, because of the proximity of all the objects and due to the noisy sensors, the system tends to maintain several hypotheses regarding the employee's activity. Nevertheless, it appears that the system achieved an A-Similarity of 81.51%(±9.5%) as well as an A-Robustness of 100%. Figure 5.26 depicts the details of the ranking of the employee's activity within the system. It appears that, when not ranked as the best hypothesis, the actual activity occupies the second position most of the time in the system. This assertion is in accordance with the experiments previously performed in virtual environments.

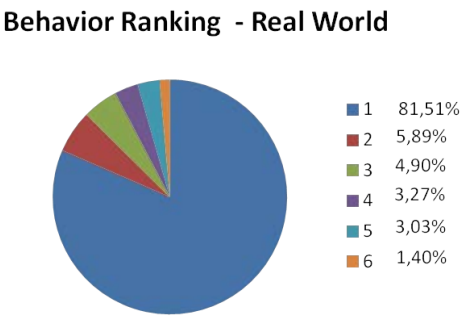


Figure 5.26: Scenario 1 - Activity Ranking in the Office Building

Impact analysis of N

The aim of this paragraph is to provide an analysis of the system performance when varying N, the number of particles used to represent the belief over the tracked-target behavior. To this end, we consider running the system with $N = 100$, 500, 1000 and 2000 respectively. For each setting, we carry out 20 runs and the outcomes are presented in Table 5.6. Unlike the experiments performed in virtual world, the robustness of the system is not guaranteed with small values of N. Indeed, for the system to have a valid hypothesis regarding the true behavior of the tracked employee during all the tracking process, at least 1000 particles are required. Nevertheless, the system performs well and the efficiency of the filtering process increases with the number of particles used.

In summary, we have demonstrated in this section that our system can well be transposed from virtual situations to real ones without significant performance degradation. For generalization purposes, we consider evaluating the system on a second scenario which is described in the next section.

Table 5.6: Scenario 1 - Performance Analysis depending on N.

	$N = 100$	$N = 500$	$N = 1000$	$N = 2000$
A-Similarity (%)	72.50 (±17.28)	75.56 (±10.32)	76.53 (±10.71)	81.51 (±9.50)
A-Robustness (%)	83.40 (±14.60)	95.54 (±4.31)	100	100

5.5.3.3 Scenario 2

In this section, we evaluate the system on a more challenging scenario with respect to the one described in Section 5.5.3.2. As previously, after describing the considered scenario, we present the results obtained using the default system parameters and analyze the system's performance when varying the number of particles.

Description

In this scenario, an employee leaves the office and goes to the open-space where he spends some time. Then, from the open-space, he moves to the toilet for a certain duration. After exiting the toilet, he goes to the server room from which he finally returns back to the open-space. A sketched illustration of the corresponding trajectory in covered areas is depicted in Figure 5.27. Here again, we manually tag each time step of the employee's trajectory with his corresponding behavior for later comparison.

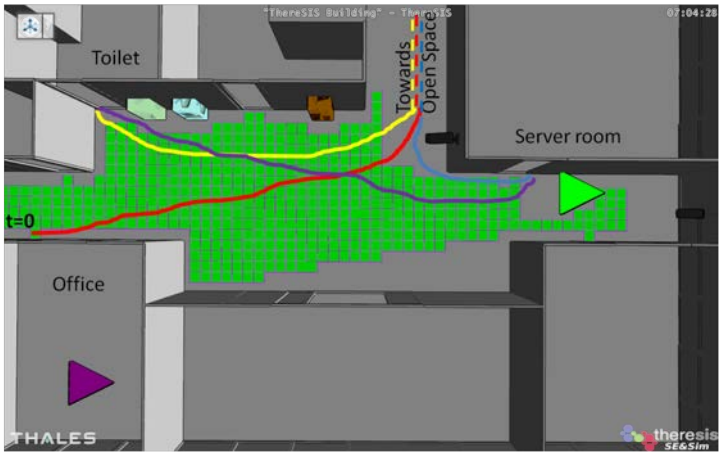
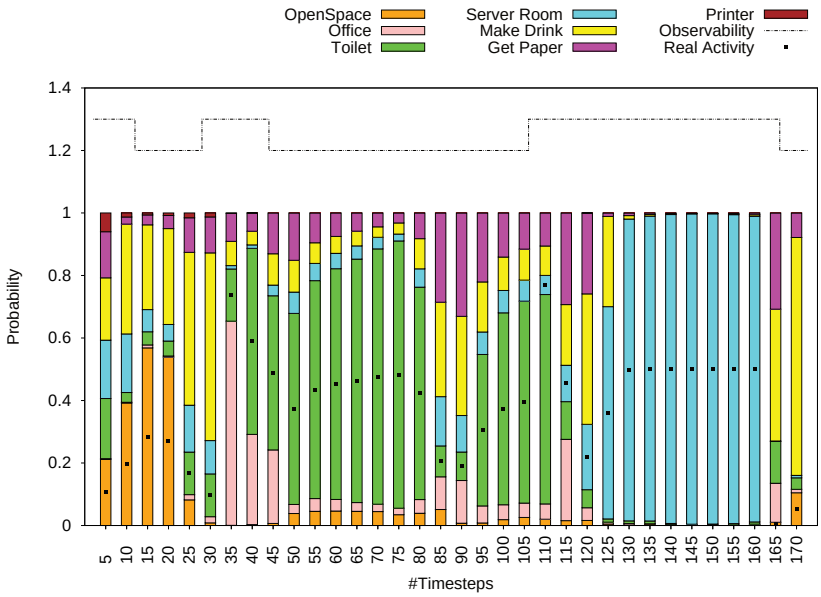


Figure 5.27: Scenario 2 - Sketch of the employee's trajectory within areas under sensory coverage: From the office to the open-space (red), from the open-space to the toilet (yellow), from the toilet to the server room (pink), and from the server room to the open-space (blue).

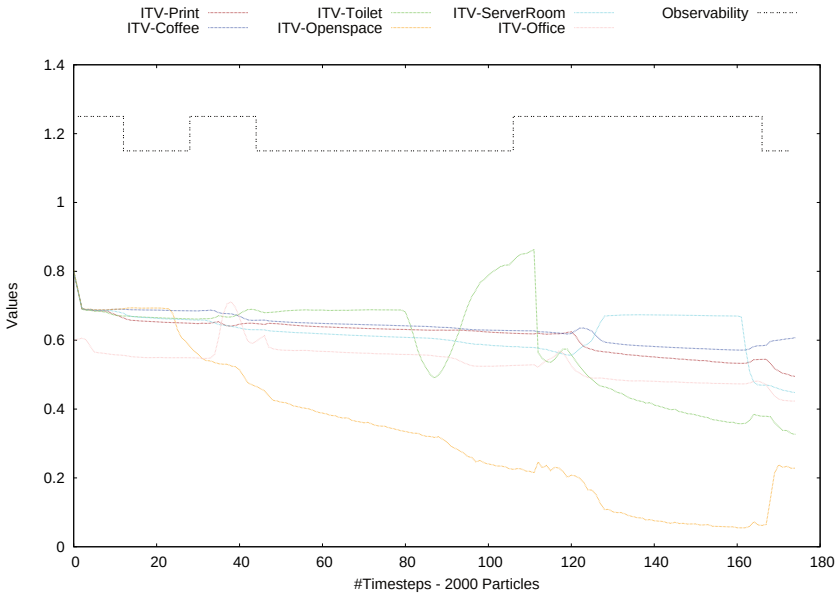
Experimental Results

As previously, the results presented in Figure 5.28 are obtained after performing 20 runs with the same input data described in Figure 5.27. Figures 5.28a shows the estimation of the employee's activity while Figure 5.28b illustrates the evolution of the model attributes (weighted mean from all particles) over the time.

Given the computational resources at hand, the system achieves a real-time response. Also, It appears that, on this scenario, the system achieves 71.98%(±3.18%) of A-Similarity with



(a) Activity estimation



(b) Model attributes' evolution

Figure 5.28: Scenario 2 - Experimental results

respect to the true target behavior as well as an A-Robustness of 100%. Having a close look on Figure 5.28b, we notice a slight domination of the "open-space" related attribute from the moment the employee is not anymore observed by the sensor network (time interval $[12, 22]$). However, at time step 23, the value of the attribute associated with the "open-space" motivation decreases while the employee is still unobserved. This is explained by the fact that we have assumed a fixed duration within the open-space. Thus, there are some particles in the system for which this time has already elapsed. Therefore, they are leaving the open-space and most of them are interested in making some drinks at the coffee machine (domination of the "coffee" related attribute). Nevertheless, when the employee is newly under sensory coverage, the system first assumes that the employee is going back in the office (domination of the "office" related attribute) before correctly inferring that he is interested in the toilet (domination of the "toilet" related attribute). The phenomenon observed in Scenario 1 for the "toilet" related attribute is also observed here with a basin occurring at time step 85. At the exit of the toilet, and because of the trajectory followed by the employee, the value of the "server" related attribute progressively increases until it becomes the dominant one. At time step 130, the system has little doubt regarding the employee behavior (see Figure 5.28a at time step 130) as he is about to use the corridor leading to the server room. Finally, from time step 161, the value of the "server" related attribute decreases while the one of the "open-space" related attribute increases, meaning that the system believes that the employee is moving towards the open-space.

Regarding the system's ranking, when not classified as the best hypothesis, the true activity of the employee occupies, like in Scenario 1, the second position most of the time within the system as illustrated in Figure 5.29.

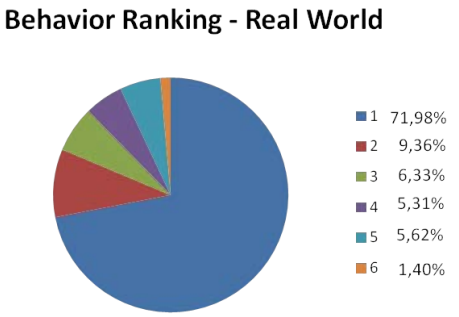


Figure 5.29: Scenario 2 - Activity Ranking in the Office Building

Impact analysis of N

Table 5.7 describes the system performances obtained when considering $N = 100$, 500, 1000 and 2000 particles respectively. As evidence of the difficulty of the scenario, 1000 particles are not enough to yield the perfect robustness of the system as it was the case in scenarios studied so far. However, the higher the number of particles used, the better the tracking efficiency.

Table 5.7: Scenario 2 - Performance Analysis depending on N.

	$N = 100$	$N = 500$	$N = 1000$	$N = 2000$
A-Similarity (%)	51.18 (±15.67)	67.25 (±11.10)	69.19 (±7.78)	71.98 (±3.18)
A-Robustness (%)	73.29 (±17.49)	88.53 (±10.29)	92.00 (±5.98)	100

5.6 Conclusion and Discussion

In this chapter, we addressed the problem of inferring the behavior of a single pedestrian within his environment on the sole basis of observation data from a sensor network. As a baseline of our work, we considered the STAR (simultaneous tracking and activity recognition) framework whose principle consists in exploiting, during the inference procedure, contextual knowledge related to activities in order to improve the estimations of targets' location and inversely.

5.6.1 Contributions

In the literature, existing STAR's methodologies mostly rely on graphical models, and particularly, dynamic Bayesian networks (DBNs) for representing the contextual knowledge of the environment. In this work, instead of DBNs, we considered the use of advanced agent-based behavioral simulators for such purposes. As a solution to the behavioral tracking problem, we proposed to integrate, within a classical Bayesian filter, such an agent-based simulator as a predictive block for behavioral analysis purposes. The designed solution has been implemented using SE-Star, a Thales proprietary agent-based behavioral simulator, and it has been evaluated in both virtual and real environments.

Performing evaluations in virtual environments (therefore with virtual targets) could be viewed as a special case in which the inner model of the tracked pedestrian is perfectly known (this is surely not the case with real humans) in such a way to avoid inexplicable phenomena. The results detailed in Section 5.5.2 showed very good filtering performances in terms of both behavior and location estimations, and this is still verified even in cases of extreme tracking conditions such as long periods of occlusion and/or exogenous events.

Regarding the evaluation in real environments, we considered scenarios taking place in our office building and designed a behavioral model to approximate a typical employee behavior. When designing such a model, a difficulty encountered in the characterization of the duration of real human-object interactions as it may completely differ from one individual to another given the same object. In the experiments, we opted for a solution consisting in embedding, within the object, a unique duration representing the average of different interaction durations observed in the real world. Another solution is to include within the behavioral model, an attribute representing the duration of the interaction a given target is about to perform and which can eventually be defined in a probabilistic way. However, the obtained results, as detailed in Section 5.5.3, were pretty good and the system proved to be quite robust. Of course, the simplified model has been designed to serve the purpose of the experiments in the considered area, but could easily imagine to elaborate, with the help of experts, a more general model applicable to all the employees in the building.

Advantages

The proposed approach presents several advantages:

- unlike tracking approaches where pedestrians are simply characterized by basic properties (e.g., position, velocity) and stereotyped behaviors (e.g., flow of trajectories), we consider finer behavioral models governed by inner action selection mechanisms that make it possible to explain outliers impossible to elucidate with stereotyped behaviors;
- secondly, it is possible to represent objects present in the environment, and to reason on the impact they have on a given pedestrian when the latter is seeking to achieve an objective. Also, it is possible to explicitly handle object's usage operations (i.e., interactions with objects) involving a certain duration.
- finally, it is possible to manage, without any additional effort, exogenous events or dynamic changes that may occur within the environment (e.g., escalator failure, fire alerts) by simply defining the impact of these events on the inner attributes of the underlying behavioral model. For example, the effect of a fire alarm could simply be to (quickly) increase the value of the attribute related to the motivation "exit the environment".

Limitations

In order to be deployed, the proposed solution required to be provided with a virtual and faithful representation of the environment in which the tracking process will be performed. Such a representation may (1) not be available and/or be expensive to realize. However, once it has been acquired, this representation can always be used for tracking pedestrians within the corresponding environment unless the structure of the latter changes. In such a case, a novel representation reflecting the new structure need to be provided.

Similarly, our solution assumes that the system is aware of areas which are not under sensory coverage. Therefore, for any modification of the sensor network configuration, a preliminary task need to be performed for calibrating the virtual sensor network on the real one.

Finally, as pointed out in Section 5.4.3, the response-time of the designed system is limited by an incompressible computational factor corresponding to the time required, by the underlying simulator, for simulating a hypothesis (particle) under consideration. The larger this factor, the larger the computational resources needed by the system to perform efficiently under real-time constraints. In the experiments carried out in this chapter, we achieved real-time performances given the computational resources at disposal. It would be interesting to study the resilience of our system when using less computational resources.

5.6.2 Research directions

This section aims at presenting the possible extensions of the work presented in this chapter.

Single behavioral model *versus* multiple behavioral models

This work assumes the availability of a unique behavioral model characterizing all the potential pedestrians within the considered environment. In practice, it may be difficult to design such a

generic behavioral model for a given environment. In order to avoid the complexity of such a design, a possible extension of this work could be to design several models, each one corresponding to a typical profile of pedestrians in the environment. For example, in the subway station, we could have a model for a controller, a passenger and/or a police officer. On this basis, each model easily embeds specificities related to each profile and the tracking process consists in simultaneously determining both the profile of the underlying targets and their behaviors (in terms of location and activity).

Combination with probabilistic graphical models

While it is clear, and we have demonstrated it through the conducted experiments, that the designed system is suitable for the recognition and the identification of location-based activities, it may not be able to differentiate activities proposed by a same object unless they have noticeable differentiators such as the duration of interaction and/or the resources required for them to be performed. This situation has been observed in Section 5.5.2.3 where the system has no choice but to maintain hypotheses (with the same probability) regarding the "buy food" and "buy drink" activities as both activities perfectly match the target trajectory. In some cases, the differentiators between activities proposed by a given object may simply reside in the gestures the considered target has to perform on the object. In such a situation, combining the behavioral model with probabilistic graphical models for gesture recognition will probably strengthen the capacities of the whole system.

Exploring outdoor environments

The study conducted in this chapter focuses on indoor environments where the choice of the activities for the target under tracking is rather limited. While we consider evaluating the resilience of the system when varying the number of particles used to represent the belief on the tracked target, a possible extension of this work consists in studying the proposed system's performance within an open world where the choice/search space is much broader as for example in the context of traffic simulations.

Multi-target tracking

This work has been described under the assumption that only one target (pedestrian) is present within the environment. This is rarely the case in real situations where several persons simultaneously evolves in a given area. When considering tracking multiple targets, problems not encountered in the case of a single target emerge making the tracking process a more challenging task. These include, among others, (1) the association of the noisy observation data to the corresponding target, (2) the re-identification of the target when re-entering an area under sensory coverage and (3) the management of targets' mutual interactions. While intensive studies have been performed in the literature to address the problem of data association [Reid, 1979, Fortmann et al., 1983, Vermaak et al., 2005], few works are interested in explicitly handling mutual influences of targets within a tracking process. The remaining of this document is devoted to the design of an efficient algorithm for tracking multiple targets and which is able to deal with these interactions.

Part III

Multi-Target Behavioral Tracking

Chapter 6

The Multi-Target Tracking Problem

Contents

6.1 Introduction

In the field of state estimation, the problem of target tracking has been widely studied during the last decades. When more than one target is considered, the problem of tracking is referred to as multi-target tracking (MTT)[Reid, 1979, Fortmann et al., 1983, Schulz et al., 2003, Liu et al., 2007]. The objective of an MTT problem is to estimate, from the history of received observations over time, the number of targets evolving in a given environment together with their states. In an MTT setting, the number of targets can potentially vary with time due to due to targets entering or leaving the environment. Also, the number of atomic observations[12] is not necessarily the same as the number of targets, particularly in case of missing reports (the probability of detection of a target is less than one) and false alarms (the sensor network can generate data which are not related to any target, but come from clutters). Moreover, in the general case, it is not possible to determine in advance from which target an atomic observation originates as well as which atomic observations correspond to clutters.

As for the single-target case, the MTT problem is a filtering problem and therefore, it can be modeled using the generic representation of partially observed dynamical systems (cf. Section 2.1). A natural approach for handling the MTT problem consists in extending the filtering techniques (e.g., Kalman filter in case of linear systems with Gaussian noises, particle filter for nonlinear systems — see Chapters 2 and 3) developed for the single-target scenario to the multi-target scenario. In such an approach, as illustrated by Figure 6.1, the state $\mathbf{x}_t$ of the global system is not anymore made by the state of a single target, but it is composed by a collection of the states of all the targets present in the environment. Similarly, the observation $\mathbf{z}_t$ received at each time step t contains information regarding all detected targets within the coverage area of the sensor network. However, this approach, while theoretically correct, suffers from the curse of dimensionality leading to high computational complexity. Indeed, the size of the state vector $\mathbf{x}_t$ gathering all target information increases with the number of targets, thus making the prediction step of the filtering solution intractable in practice [Liu et al., 2007].

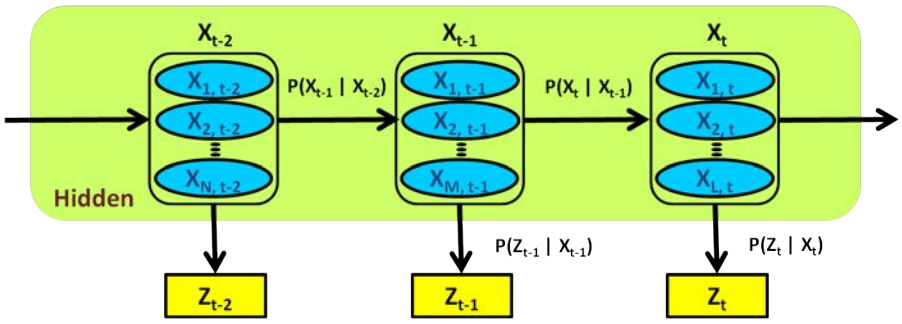


Figure 6.1: A partially observed dynamical system modeling a multi-target tracking problem. The state $\mathbf{x}_t$ of the overall system at time t is a collection $\{\mathbf{x}_{k,t}\}$ of the states of all the targets present in the environment.

An alternative approach for breaking down this computational complexity consists in approximating the distribution regarding the target states by reasoning on each target separately rather than focusing on the global state $\mathbf{x}_t$ as a whole. Consequently, as depicted in Figure 6.2, the

[12] We use the term "atomic observation (or measurement)" to refer to an observation data related to a single target.

overall filtering process can be decomposed — in a sub-optimal way — into a collection of sub-filtering processes, each dedicated to a given target. By doing so, the dimension of the state for each sub-filtering process is limited to a single target and it is fixed over time, which is convenient for computational issues. Nevertheless, this decomposition into target-oriented (sub-)filtering processes raises up two main problems that need to be handled for tracking efficiency purposes:

- **the data association problem**: since each sub-filtering process focuses on a dedicated target, the observation data related to the underlying target is needed for updating the target's posterior distribution. The observation $\mathbf{z}_t$ received from the sensor network at each time step t encapsulates atomic measurements regarding all the detected targets. When these atomic measurements do not contain the identity of the target they originate from, determining the matching between targets and atomic measurements is the purpose of the data association problem.

- **the management of targets' interactions**: each sub-filtering process aims at producing an individual posterior distribution characterizing the state of its associated target. This is appropriate and suitable when the targets behave independently from each other. However, in most scenarios, targets mutually interact and affect each other's behavior. In such a case, in order to not considerably degrade the quality of the individual posterior distributions, it is necessary to design solutions capable of taking into account such interactions while maintaining low computational complexity.

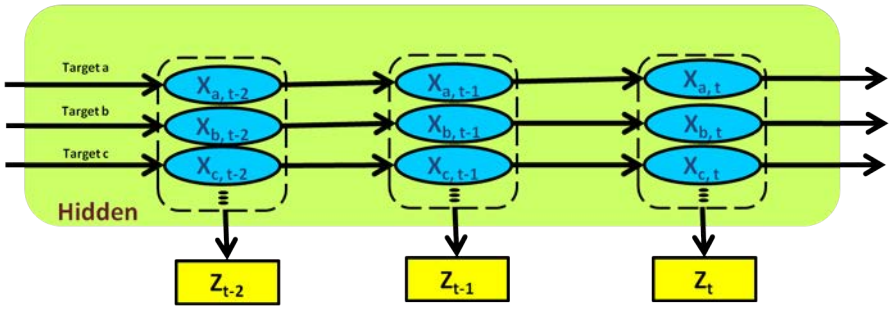


Figure 6.2: A multi-target problem modeled as a collection of target-oriented sub-filtering processes. Each sub-filtering process is dedicated to a specific target in the environment.

Several methodologies have been proposed in the literature for handling the data-association problem. Among them, the most notable are the Multiple Hypothesis Tracker (MHT) [Reid, 1979] and the Joint Probabilistic Data Association Filter (JPDAF) [Fortmann et al., 1983]. Similarly, there have been works addressing the tracking of multiple interacting targets such as the one of Khan et al. [2005].

The remainder of this chapter aims at presenting a non-exhaustive review of these works and is organized as follows. Section 6.2 mathematically formalizes the multi-target tracking problem and introduces the notations used throughout this chapter, while Section 6.3 is dedicated to the description of methodologies developed for addressing the data-association problem. Finally, in Section 6.4, we present solutions designed for managing targets' interactions within the context of multi-target tracking.

6.2 Problem Formulation

Let us consider an environment in which several targets evolve and which is equipped with a sensor network. The purpose of this section consists in describing the MTT problem as a filtering problem and therefore, in terms of system's dynamics and observation model.

6.2.1 System Dynamics

Generally, the number of targets in the considered environment may vary over time. Let K_t be the number of targets in the environment at time t. Also, let $\mathbf{x}_{k,t}$ be the state of the k^{th} target at time t. The state of the overall system at time t, defined as $\mathbf{x}_t = (K_t, \{\mathbf{x}_{k,t}\}_{k=1}^{K_t})$,[13] is composed by the state of individual targets present in the environment at that time. In the general case where it is possible for targets to mutually interact (the behavior of a given target can be influenced by other targets), the system is globally characterized by a probabilistic dynamics of the form $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ defining how targets evolve together. When interactions are non-existent and targets are assumed to behave independently, the system dynamics can be decomposed into individual target dynamics and, assuming the number of targets fixed over time ($K_t = K, \forall t \geq 0$), it can be written as

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{k=1}^{K} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}). \tag{6.1}$$

6.2.2 Observation Model

Regarding the sensor network, it is assumed that each target can be detected with a probability P_D, and, if detected, a noisy atomic observation of its state is generated. Furthermore, the sensor network may return false alarms, and the number of false alarms generated is assumed to follow a Poisson distribution with a parameter $\lambda_{FT}V$ where:

- λ_{FT} is the false alarm rate per unit time, per unit volume;
- V is the volume of $\mathcal{R}$, that is, the region of the environment where targets may evolve.

In what follows, we present the assumptions related to the observation data returned by the sensor network as well as the generation procedure.

6.2.2.1 Observation-Generation-related Assumptions

Let $\mathbf{z}_t = \{\mathbf{z}_{1,t}, \mathbf{z}_{2,t}, \cdots, \mathbf{z}_{M_t,t}\}$ be the observation received from the sensor network at time t, where M_t represents the number of atomic observations. $\mathbf{z}_t$ includes both noisy target measurements and false alarms. Let $\mathbf{z}_{j,t}$ be the j^{th} atomic observation at time t for $j = 1, \cdots, M_t$. The following *assumptions* are made:

[13] In this formulation, at each time step t, targets are numbered from 1 to K_t. In cases where targets may enter or exit the environment at any time, we assume the availability of an additional function which maps, at each time step t, each number $k \in [1, K_t]$ to the original identity of the corresponding target.

- each target generates at most one atomic observation at each time step;
- an atomic observation originates from at most one target;
- the generation of a target-related atomic observation is independent of other targets.

6.2.2.2 Observation Generation Procedure

At each time step t, the generation of the observation data is done according to the following process:

- **Generation of the atomic observation related to tracked targets.** For each target k within an area of the environment under sensory coverage, a noisy atomic observation is generated with a probability P_D. This generation is made according to $h(\mathbf{x}_{k,t}, \mathbf{v}_{\cdot,t})$ where h is a function — related to the sensor network — which returns a noisy atomic observation given the state $\mathbf{x}_{k,t}$ of the underlying target and the observation noise $(\mathbf{v}_{\cdot,t})_{t>0}$ of the sensor network.
- **Generation of the atomic observation related to clutters.** The number N_c of clutter-related atomic observations to generate is sampled from the Poisson distribution $F_{Pois}(\cdot, \lambda_{FT}V)$, that is $N_c \sim F_{Pois}(\cdot, \lambda_{FT}V)$. Then, N_c atomic observations are randomly generated by uniformly sampling the region $\mathcal{R}$ where targets may evolve.
- **Shuffling of the atomic observations.** The so generated atomic observations — target-related and clutter-related — are then put together and shuffled in such a way there is no correlation between the final order of the atomic observations in $\mathbf{z}_t$ and their origin.

Summarizing this description in a mathematical form, we then have:

$$\mathbf{z}_{j,t} = \begin{cases} h(\mathbf{x}_{k,t}, \mathbf{v}_{j,t}) & \text{if } \exists k \text{ s.t. the } j^{th} \text{ atomic observation is} \\ & \text{from the } k^{th} \text{ target,} \\ \mathbf{u}_t & \text{otherwise,} \end{cases} \tag{6.2}$$

where $\mathbf{u}_t \sim Unif(\mathcal{R})$ is a uniform random distribution for false alarms and, the sign $\exists$ is used to illustrate that it is not possible to automatically retrieve k from j. From a probabilistic perspective, the function h is characterized by a probability distribution $p_h(\mathbf{z}_{j,t}|\mathbf{x}_{k,t})$ which allows to relate the noisy measurement obtained from the sensors to the target's state.

6.2.3 Summary

In general, given the considered environment, new targets may appear while some existing targets may disappear at any point in time. Therefore, the purpose of the multi-target tracking problem consists in estimating the number of targets K_t present in the environment as well as their states $\{\mathbf{x}_{k,t}\}_{k=1}^{K_t}$ from the sequence of observations received so far.

As previously mentioned, approaching the MTT problem by applying classical filtering methods for estimating the joint distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ suffers from the curse of dimensionality.

Instead, existing methods [Reid, 1979], [Fortmann et al., 1983], [Schulz et al., 2003], [Frank et al., 2003], [Vermaak et al., 2005] seek to estimate the marginal individual target distributions $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$, for $k = 1, \cdots, K_t$. The basic idea behind these methods can be summarized into two steps. First, they solve the data-association problem upon reception of the observation $\mathbf{z}_t$ at each time step t. Then, based on the resulting target-measurement associations, they subsequently apply standard filtering techniques for updating individual target distributions. In the next section, we focus on the data-association problem and present existing methodologies from the literature for solving it.

6.3 Data-Association Problem

The data-association problem arises when it is not possible to directly determine from the observation data received, the origin of a given measurement. Therefore, this problem is not faced in tracking applications where the considered sensors can provide identity information (e.g., wearable devices or RFID sensors). This section presents methodologies that have been developed in the literature for addressing the data-association problem in the context of multi-target tracking. We focus the discussion on solutions assuming that the tracked targets do not interact with each other and behave independently (interactions will be handled in Section 6.4). Therefore, given any target k, its dynamics can be fully characterized by a probability distribution of the form $p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1})$ (see Equation 6.1).

6.3.1 Global Nearest Neighbor

The global nearest neighbor (GNN) approach [Blackman, 1986] is a solution developed to handle the data-association problem in the context of MTT. The general idea behind the GNN approach is to find a unique joint association of atomic measurements to targets that minimizes a total cost — e.g., the total summed distance — or maximizes a utility function — e.g., the likelihood — under the constraint that an atomic measurement can be associated with at most one target.

Let us consider $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$, the state estimation of target k given observation data up to time t, for $k = 1, \cdots, K_t$. At time $t+1$, the GNN filter first computes the predicted state estimation $p(\mathbf{x}_{k,t+1}|\mathbf{z}_{1:t})$ of each target using standard Bayesian filtering methods, that is

$$p(\mathbf{x}_{k,t+1}|\mathbf{z}_{1:t}) = \int p(\mathbf{x}_{k,t+1}|\mathbf{x}_{k,t})p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})d\mathbf{x}_{k,t}. \tag{6.3}$$

Equation 6.3 can be evaluated, for example, using the Kalman formulations in case of linear and Gaussian assumptions of the target's dynamics or using formulations derived from the particle filter framework otherwise.

Let $\mathbf{z}_{t+1} = \{\mathbf{z}_{1,t+1}, \mathbf{z}_{2,t+1}, \cdots, \mathbf{z}_{M_{t+1},t+1}\}$ be the observation data received at time $t+1$. Then, after computing the predicted states for all targets, the GNN filter proceeds by computing the likelihood D_j^k of each individual atomic measurement j with respect to each target k as follows:

$$D_j^k = \int p_h(\mathbf{z}_{j,t+1}|\mathbf{x}_{k,t+1})p(\mathbf{x}_{k,t+1}|\mathbf{z}_{1:t})d\mathbf{x}_{k,t+1}, \tag{6.4}$$

where the term $p_h(.|.)$ represents the observation model of the sensors.

Once the different likelihood values have been computed, the GNN filter solves the following optimization problem:

$$\begin{aligned} &\arg\max_{(a_j^k)} \quad \sum_{k=1}^{K_t}\sum_{j=1}^{M_{t+1}} D_j^k a_j^k, \\ &\text{such that:} \\ &\qquad \sum_{j=1}^{M_{t+1}} a_j^k \leq 1, \qquad \forall\, k = 1, \cdots, K_t; \\ &\qquad \sum_{k=1}^{K_t} a_j^k \leq 1, \qquad \forall\, j = 1, \cdots, M_{t+1}; \\ &\qquad a_j^k \in \{0,1\}, \qquad \forall\, k = 1, \cdots, K_t;\ \forall\, j = 1, \cdots, M_{t+1}; \end{aligned} \tag{6.5}$$

where the variable a_j^k associates the k^{th} target to the j^{th} atomic observation. Existing algorithms, such as the Hungarian method (also referred to as the Munkres algorithm) [Kuhn, 1955], allow to find the optimal solution to the problem stated in Equation 6.5 in polynomial time.

The assignment solution resulting from the above optimization problem is considered by the GNN filter to be the correct one. Unassociated measurements can subsequently be used for track creation (new target) while unassociated targets can be used for track termination. Finally, the GNN filter proceeds by updating, for each target k, its state estimation using the associated atomic measurement if it exists (j such that $a_j^k = 1$).

Although the GNN scheme is simple to implement, it generally performs poorly in practice. This is due to the greediness of the approach. Indeed, the optimal solution to Equation 6.5 does not always correspond to the true assignment, leading thus to a degradation of performance.

6.3.2 Multiple Hypothesis Tracker

The multiple hypothesis tracking (MHT) filter is an approach proposed by Reid [1979] for handling the data-association problem. Unlike the GNN approach where only a single association is used for updating the individual target's posterior distributions, MHT considers, within its process, all the feasible associations of an atomic observation to either a target (new or existing one) or a clutter. This is done by reasoning, over time, on *trajectory hypotheses* of all the targets in the environment.

Specifically, let the term *target track* refer to a temporal sequence of associations of a given target to atomic observations. A trajectory hypothesis $\Omega_{i,t}$ at time step t encapsulates all the observation data $\mathbf{z}_{1:t}$ received so far, implies the existence of $K_{i,t}$ targets within the environment, and it is defined in terms of target tracks as

$$\Omega_{i,t} = (\tau_0, \tau_1, \cdots, \tau_{K_{i,t}}) \text{ such that:}$$

- $\cup_{p=0}^{K_{i,t}} \tau_p = \mathbf{z}_{1:t}$,
- $\tau_{p_1} \cap \tau_{p_2} = \emptyset \ \ \forall p_1 \neq p_2$,
- $|\tau_p \cap \mathbf{z}_{t_1}| \leq 1 \ \ \forall p = 1, \cdots, K_{i,t}$ and $t_1 = 1, \cdots, t$.

Each $(\tau_p)_{p>1}$ represents a target track while τ_0 is the set of atomic observations considered as false alarms. The first two conditions guarantee — as stated in Section 6.2.2.1 — that all atomic observations must be assigned to either a target or clutter in an exhaustive way, and that there is no conflict in the association as two target tracks can not share the same atomic observations. The last condition states that each target track can be assigned at most one atomic observation at each time step (missing reports).

Basically, if the observation data were related to the identities of their generators, there would be only a unique trajectory hypothesis within the system and their individual posterior distributions would have been estimated using classical filtering techniques (e.g., Kalman filter). Instead, MHT proceeds by maintaining, at each time step, a set of trajectory hypotheses together with their probabilities and, for each such hypothesis, the posterior distributions of the targets — as implied by the hypothesis — are computed using classical filtering techniques while assuming correct the associations described by the hypothesis. Over time, trajectory hypotheses with low probabilities are discarded and the remaining hypotheses are those which likely match the observations received so far, including the one corresponding to the true data-association. In principle, because no formal decision is taken at each time step regarding the origins of the atomic observations, MHT is generally able to recover, at the end of the process, the true data-association and therefore, the best estimates — as allowed by the filtering techniques used — of the individual posterior distributions.

Let $\Omega_t = \{\Omega_{i,t}\}$ be the set of trajectory hypotheses within the system at time step t. Let $p(\Omega_{i,t}|\mathbf{z}_{1:t})$ be the probability of the i^{th} hypothesis. Upon receiving the observation data $\mathbf{z}_{t+1} = \{\mathbf{z}_{1,t+1}, \mathbf{z}_{2,t+1}, \cdots, \mathbf{z}_{M_{t+1},t+1}\}$ at time step $t+1$, MHT iteratively builds the new hypothesis set Ω_{t+1} by processing each $\Omega_{i,t} \in \Omega_t$ according to the following procedure:

- First, it generates the set of possible assignments in which each atomic observation in $\mathbf{z}_{t+1}$ is associated either to clutter, an existing target or a new target. Each such assignment is referred to as a measurement-to-target association. In order to not consider unlikely measurement-to-target associations, MHT usually relies on a *gating procedure* which aims at associating an existing target (in $\Omega_{i,t}$) to an atomic observation if and only if the latter lies within the gate — or the *validation region* — of the target as implied by $\Omega_{i,t}$ (see Section 6.3.2.1).

- Secondly, a new set Ω^i_{t+1} of trajectory hypotheses is created by extending $\Omega_{i,t}$ with the previously generated measurement-to-target associations while respecting the constraints related to the definition of a trajectory hypothesis (see Section 6.3.2.2).

- Finally, the probability of each hypothesis in Ω^i_{t+1} is computed from $p(\Omega_{i,t}|\mathbf{z}_{1:t})$ and the probability corresponding to the involved measurement-to-target associations (see Section 6.3.2.3).

Once each $\Omega_{i,t}$ has been processed and $\hat{\Omega}_{t+1} = \cup_{i=1}^{|\Omega_t|}\Omega^i_{t+1}$ has been obtained, a reduction procedure is performed on $\hat{\Omega}_{t+1}$ for eliminating unlikely hypotheses, resulting thus to Ω_{t+1}. All these steps are detailed in the following sections.

6.3.2.1 Gating Procedure

Once new observation data $\mathbf{z}_{t+1}$ have been received from the sensor network, an important step for the generation of a new set Ω^i_{t+1} of trajectory hypotheses, given the hypothesis $\Omega_{i,t}$, is to determine from which targets current atomic observations have been engendered. A basic approach consists in enumerating all the possible ways that any atomic observation in $\mathbf{z}_{t+1}$ may originate from known —or possibly new— targets. However, this approach tends to be impractical as the number of targets increases [Bar-Shalom and Fortmann, 1987].

The gating technique is a heuristic introduced [Reid, 1979, Bar-Shalom and Fortmann, 1987] for reducing the complexity of the data association step. It suggests that a target can only be associated with an atomic observation if the latter lies within the *validation region* of the target. An illustration of the procedure is shown in Figure 6.3. In this figure, the blue circles represent the different targets while the green squares are the atomic observations. The dashed ellipse around each target represents its validation region. As depicted, the atomic observation Z_3, apart being associated with clutter or a new target, can only be associated with targets T_1 or T_2.

The validation region of a target k is defined from the observation model of the sensor network as well as its posterior distribution computed from $\Omega_{i,t}$. In case (1) the predicted target's posterior distribution is Gaussian with $\bar{\mathbf{x}}_{k,t}$ and $\bar{\mathbf{P}}_{k,t}$ being respectively the mean and the covariance, and (2) the observation model is linear with Gaussian noise, such a validation gate is characterized by the covariance matrix $\mathbf{C}^{\nu}_{k,t}$ defined as

$$\mathbf{C}^{\nu}_{k,t} = \mathbf{H}\bar{\mathbf{P}}_{k,t}\mathbf{H}^T + \mathbf{R}_t, \tag{6.6}$$

where $\mathbf{H}$ and $\mathbf{R}_t$ are known matrices characterizing the sensor model and the observation noise respectively, and $\mathbf{H}^T$ is the transpose of $\mathbf{H}$. Under these assumptions, an atomic observation $\mathbf{z}_{j,t+1}$ belongs to the so defined validation region if and only if

$$(\mathbf{z}_{j,t+1} - \mathbf{H}\bar{\mathbf{x}}_{k,t})^T {\mathbf{C}^{\nu}_{k,t}}^{-1}(\mathbf{z}_{j,t+1} - \mathbf{H}\bar{\mathbf{x}}_{k,t}) \leq \eta, \tag{6.7}$$

where η is a predefined threshold.

Once the measurement-to-target associations have been generated on the basis of the trajectory hypothesis $\Omega_{i,t}$, the next step consists in generating new hypotheses accounting for time step $t+1$ and how this is done is the topic of the following section.

6.3.2.2 Generation of Hypotheses

The purpose of this section is to describe how, at time step $t+1$, a new set Ω^i_{t+1} of trajectory hypotheses is generated from a hypothesis $\Omega_{i,t}$ at time step t and measurement-to-target associations previously generated from the received observation $\mathbf{z}_{t+1}$. The overall procedure is described in Algorithm 6.1. Starting from a (resulting) set containing only hypothesis $\Omega_{i,t}$, the algorithm proceeds by screening each atomic observation composing $\mathbf{z}_{t+1}$ and iteratively updating the resulting set. More specifically, for each atomic observation $\mathbf{z}_{j,t+1}$, each hypothesis currently within the resulting set is extended, on the basis of measurement-to-target associations (generated using the gating procedure) involving $\mathbf{z}_{j,t+1}$, into a subset of hypotheses. The extension of an hypothesis by a measurement-to-target association is made possible if and only if the constraints related to the definition of a trajectory hypothesis are not violated. Once all the hypotheses

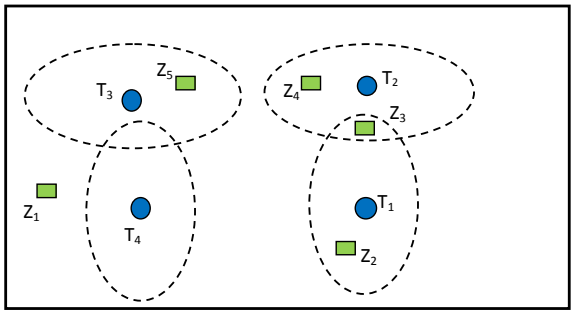


Figure 6.3: Example of the Gating procedure: the blue circles represent targets and the green squares represent atomic observations. The validation region of each target is represented by the ellipse centered around it and only atomic observations falling in this region are considered as feasible for the target.

within the current resulting set have been processed, the so generated subsets of hypotheses are merged together, thus forming the new resulting set from which the next atomic observation will be screened. In MHT, an atomic observation can be assigned to either a clutter, an existing target or a completely new target. By doing so, MHT inherently handles track initiations, and it is therefore appropriate for tracking an unknown and time-varying number of targets.

An illustration of how the hypothesis generation procedure works is shown in Figure 6.4. In the figure, a trajectory hypothesis $\Omega_{i,t}$ implies the existence of 2 targets (T_1 and T_2) at time step t. Upon reception of a measurement composed of 3 atomic observations (Z_1, Z_2, Z_3), a gating procedure is performed and, as a result, T_1 can be associated to Z_1 while T_2 can be associated to either Z_1 or Z_2. Also, all atomic observations can be associated to clutter or a new target. The generation process is described as a hierarchy (tree) where each level corresponds to the screening of each atomic observation. The final trajectory hypotheses correspond to the leaves of the tree and, as illustrated, $\Omega_{i,t}$ engenders 22 hypotheses accounting for time step $t+1$.

Once the set Ω^i_{t+1} has been obtained, the next step consists in computing the probabilities of the generated hypotheses. This is discussed in the next section.

6.3.2.3 Hypothesis Probability Computation

The purpose of this section is to describe how to compute the probability of a given trajectory hypothesis. Let $\Omega_{g,t+1}$ be the trajectory hypothesis of interest. Also, let $\Omega_{i,t}$ be the hypothesis at time step t from which $\Omega_{g,t+1}$ has been generated and, ψ_h be the set of measurement-to-target associations generated from $\mathbf{z}_{t+1}$ used for extending $\Omega_{i,t}$ in order to obtain $\Omega_{g,t+1}$. The probability $p(\Omega_{g,t+1}|\mathbf{z}_{1:t+1})$ is computed using the Bayes rule as follows:

$$\begin{aligned} p(\Omega_{g,t+1}|\mathbf{z}_{1:t+1}) &= p(\Omega_{i,t}, \psi_h|\mathbf{z}_{1:t+1}), \\ &\propto p(\mathbf{z}_{t+1}|\Omega_{i,t}, \psi_h, \mathbf{z}_{1:t}) \times p(\psi_h|\Omega_{i,t}, \mathbf{z}_{1:t}) \times p(\Omega_{i,t}|\mathbf{z}_{1:t}), \end{aligned} \tag{6.8}$$

where $p(\mathbf{z}_{t+1}|\Omega_{i,t}, \psi_h, \mathbf{z}_{1:t})$ represents the likelihood of the observation data $\mathbf{z}_{t+1}$ given the data-associations described by $\Omega_{i,t}$ and ψ_h while $p(\psi_h|\Omega_{i,t}, \mathbf{z}_{1:t})$ is the probability of the measurement-to-target associations given the prior hypothesis $\Omega_{i,t}$. In what follows, these two terms are evaluated.

Algorithm 6.1: Generation Procedure of MHT's Hypotheses

Algorithm `MHT_HYP_GEN`$(\Omega_{i,t},\ \mathbf{z}_{t+1})$
  $A =$ `GatingProcedure` $(\Omega_{i,t},\ \mathbf{z}_{t+1})$*
  $\bar{\Omega}_0 = \{\Omega_{i,t}\}$
  for $j = 1 : |\mathbf{z}_{t+1}|$ **do**
    $A_j = \{a \in A \text{ s.t. } a \text{ involves } \mathbf{z}_{j,t+1}\}$
    $\bar{\Omega}_j = \bar{\Omega}_{j-1}$
    for $l = 1 : |\bar{\Omega}_{j-1}|$ **do**
      for $a \in A_j$ **do**
        if *a does not violate the assignation constraints in* $\bar{\Omega}_{l,j-1}$ **then**
          $\bar{\Omega}_{temp} = \bar{\Omega}_{l,j-1} \cup \{a\}$
          $\bar{\Omega}_j = \bar{\Omega}_j \cup \{\bar{\Omega}_{temp}\}$
  $\Omega^i_{t+1} = \bar{\Omega}_{M_{t+1}}$
  return Ω^i_{t+1}
(*) The **Procedure** `GatingProcedure` $(\Omega_{i,t},\ \mathbf{z}_{t+1})$ returns the set of possible measurement-to-target associations.

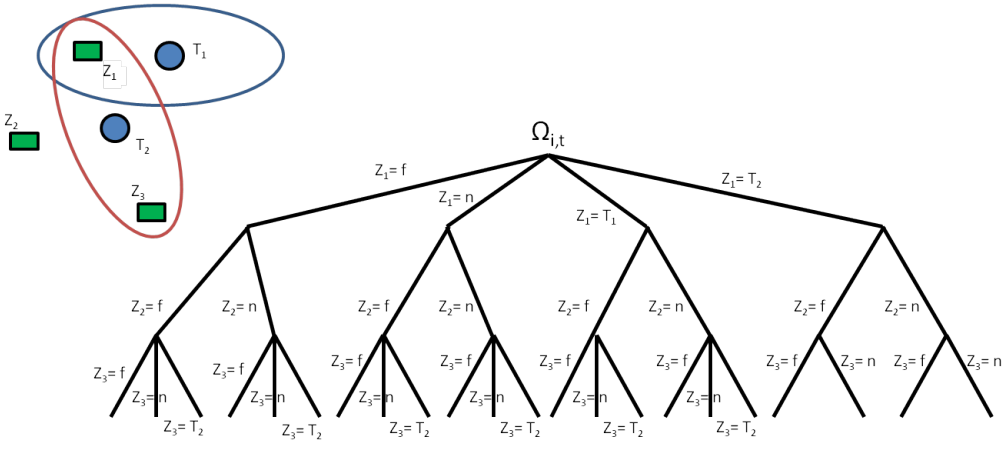


Figure 6.4: MHT's Hypothesis Generation Procedure - The figure shows an illustrative scenario (in the top-left corner) and a tree of hypotheses generated for this scenario - blue circles are targets and green squares are atomic observations. From Hypothesis $\Omega_{i,t}$ at the time step t, 22 hypotheses are generated for the next time step $t+1$. In the figure, n denotes *new target* while f denotes *false alarm* (clutter).

The term $p(\mathbf{z}_{t+1}|\Omega_{i,t}, \psi_h, \mathbf{z}_{1:t})$ of Equation 6.8 is given by:

$$p(\mathbf{z}_{t+1}|\Omega_{i,t}, \psi_h, \mathbf{z}_{1:t}) = \prod_{j=1}^{M_{t+1}} f_p(j), \tag{6.9}$$

where

$$f_p(j) = \begin{cases} 1/V & \text{if the } j^{th} \text{ atomic observation is from} \\ & \text{clutter or a new target,} \\ D_j^k & \text{if the } j^{th} \text{ observation is from a target } k \text{ whose} \\ & \text{existence is implied by the prior hypothesis } \Omega_{i,t}, \end{cases} \tag{6.10}$$

with D_j^k being the predictive likelihood of the j^{th} observation with respect to the k^{th} target as given in Equation 6.4 and V being the volume of the environment's region where targets may evolve (cf. Section 6.2.2).

The term $p(\psi_h|\Omega_{i,t}, \mathbf{z}_{1:t})$ in Equation 6.8 is obtained as follows. Let N_{TGT} denote the number of previously known targets as implied by the prior hypothesis $\Omega_{i,t}$. Also, since ψ_h associates each atomic observation to a specific source (target or clutter), it encapsulates information regarding the number N_{DT} of atomic observations associated with targets whose existence is implied in the prior hypothesis $\Omega_{i,t}$ (meaning that only N_{DT} targets have been detected out of the N_{TGT} ones) as well as the numbers N_{FT} and N_{NT} of atomic observations associated with clutter and new targets respectively. We have

$$M_{t+1} = N_{FT} + N_{DT} + N_{NT}. \tag{6.11}$$

Moreover, it is possible from ψ_h to determine the identity l_j of a known target associated to the j^{th} atomic observation. We have

$$l_j = \begin{cases} k & \text{if the } j^{th} \text{ measurement is from a target } k \text{ whose} \\ & \text{existence is implied by the prior hypothesis } \Omega_{i,t}, \\ 0 & \text{otherwise.} \end{cases} \tag{6.12}$$

As for the number N_{FT} of false alarms, it is assumed that the number N_{NT} of new targets follows a Poisson distribution with a parameter $\lambda_{NT}V$ where λ_{NT} is the density rate of new targets per unit time, per unit volume; and V is defined as previously. From these assumptions, the probability $p(\psi_h|\Omega_{i,t}, \mathbf{z}_{1:t}^L)$ can be expressed as [Reid, 1979]

$$p(\psi_h|\Omega_{i,t}, \mathbf{z}_{1:t}) = \frac{N_{FT}!N_{NT}!}{M_{t+1}!} \times P_D^{N_{DT}}(1-P_D)^{(N_{TGT}-N_{DT})} \times F_{pois}(N_{NT}, \lambda_{NT}V)F_{pois}(N_{FT}, \lambda_{FT}V), \tag{6.13}$$

where $F_{pois}(.,.)$ represents the Poisson distribution function.

Substituting Equation 6.13 and Equation 6.9 into Equation 6.8, we get

$$\begin{aligned} p(\Omega_{g,t+1}|\mathbf{z}_{1:t+1}) \propto & \frac{N_{FT}!N_{NT}!}{M_{t+1}!} \times P_D^{N_{DT}}(1-P_D)^{(N_{TGT}-N_{DT})} \\ & \times F_{pois}(N_{NT}, \lambda_{NT}V)F_{pois}(N_{FT}, \lambda_{FT}V) \\ & \times \left[\prod_{j:l_j\neq 0} D_j^{l_j}\right] \frac{1}{V^{N_{FT}+N_{DT}}} p(\Omega_{i,t}|\mathbf{z}_{1:t}). \end{aligned} \tag{6.14}$$

Finally, by replacing the Poisson processes with their mathematical formulas, the dependence of V is simplified and Equation 6.14 can be rewritten as

$$\begin{aligned} p(\Omega_{g,t+1}|\mathbf{z}_{1:t+1}) \propto & P_D^{N_{DT}}(1-P_D)^{(N_{TGT}-N_{DT})}\lambda_{FT}^{N_{FT}}\lambda_{NT}^{N_{NT}} \\ & \times \left[\prod_{j:l_j\neq 0} D_j^{l_j}\right] p(\Omega_{i,t}|\mathbf{z}_{1:t}). \end{aligned} \tag{6.15}$$

Having obtained the set of trajectory hypotheses together with their probabilities, standard Bayesian filtering techniques are used, independently for each hypothesis, to update the states and trajectories of individual targets.

As it has been noticed, by considering hypotheses in which an atomic observation originated from a new target, MHT can handle track initiations. Additional mechanisms have been developed in the literature [Sittler, 1964] for handling, within the MHT framework, track terminations, that is, the possibility that a target ceases to exist (e.g., it exits the environment).

6.3.2.4 Hypothesis Reduction Techniques

In the MHT approach, the total number of possible trajectory hypotheses increases exponentially with time, thus making the implementation of such a theoretical approach intractable. In practice, pruning heuristics are used to reduce the computational requirements with the objective of keeping the accuracy of the theoretical filter. Applying these heuristics, only a small number of hypotheses is maintained at each time step. The simplest heuristic consists in keeping, at each time step, the hypothesis with the highest posterior distribution, thus leading to the GNN approach (cf. Section 6.3.1). However, this heuristic can significantly degrade the tracking performance, especially when the retained hypothesis does not match the reality [Bar-Shalom and Li, 1995]. An alternative heuristic consists in keeping all the hypotheses whose posterior probabilities are above a specified threshold. The advantage of this heuristic with respect to the first one resides in the fact that subsequent observations may be used to aid in the correlation of prior observations since no strong decision (regarding the data-association) is made at each time step [Blackman, 1986].

Another way for reducing the number of hypotheses consists in combining similar hypotheses. One simple approach to achieve this is to merge, into a unique hypothesis, all the hypotheses which have the last N (corresponding to the last N time steps) measurement-to-target associations in common. The limitation of this approach is that hypotheses differentiating between observations in earlier time steps are eliminated. As an alternative, it is possible to bind together hypotheses with similar effects (e.g., same number of targets and almost identical state

estimations, given each target, in considered hypotheses). For further details on implementation, please refer to [Cox and Hingorani, 1996, Blackman and Popoli, 1999].

6.3.3 Joint Probabilistic Data Association Filter

The joint probabilistic data association filter (JPDAF) is a scheme developed by Fortmann et al. [1983] to handle the data association problem in situations in which the number of targets in the environment is known and fixed ($K_t = K, \forall t \geq 1$). Unlike MHT which reasons on a temporal sequence of measurement-to-target associations over time, JPDAF does not consider the history of associations made and focuses only on feasible associations at the current time step. Practically, given a set of potential candidates (in terms of atomic observations) for association to a given target at the current time step, JPDAF does not keep a track of them for further consideration in the future. Instead, it computes the statistically most probable distribution of these candidates with respect to the considered target while taking into account the presence of other targets (their posterior distributions) and the statistical distribution of clutters. The resulting distribution is subsequently used for updating the posterior distribution regarding the state of the considered target.

To go more into the details, upon receiving a new observation, let us say $\mathbf{z}_{t+1} = \{\mathbf{z}_{j,t+1}\}_{j=1}^{M_{t+1}}$, JPDAF proceeds according to the following procedure:

- First, based on the predicted belief of each target (see Equation 6.3), a set of potential measurement-to-target associations is computed for all the targets. This is usually done using the Gating technique described in Section 6.3.2.1.
- Secondly, from the measurement-to-target associations generated, a set Θ of *feasible joint association hypotheses* (FJAHs) is computed. An FJAH θ is a collection of measurement-to-target associations such that:
  - at most one measurement-to-target association in θ is related to a given target;
  - at most one measurement-to-target association in θ is related to a given atomic observation.

  Considering an FJAH θ, targets not related to any association are assumed undetected (or associated to a fictional atomic observation $\mathbf{z}_{0,t+1}$ — $\mathbf{z}_{0,t+1}$ can therefore be associated to more than one target) while atomic observations not involved in any association are considered as false alarms. The generation of the set Θ is discussed in Section 6.3.3.1.
- Then, from the set Θ, the statistically most probable distribution of the observation data $\mathbf{z}_{t+1}$ with respect to each target k is computed as

$$p(\mathbf{z}_{t+1}|\mathbf{x}_{k,t+1}) = \sum_{j=0}^{M_{t+1}} \beta_{jk}.p_h(\mathbf{z}_{j,t+1}|\mathbf{x}_{k,t+1}), \tag{6.16}$$

  where β_{jk} is the probability that $z_{j,t+1}$ is generated from the k^{th} target (if $j > 0$) or the probability that the k^{th} target is undetected (if $j = 0$), and $p_h(\mathbf{z}_{j,t+1}|\mathbf{x}_{k,t+1})$ corresponds

to the observation model. How to compute the β_{jk} coefficients given Θ is discussed in Section 6.3.3.2.

- Finally, the posterior distribution of each target k is updated using classical filtering techniques as

$$p(\mathbf{x}_{k,t+1}|\mathbf{z}_{1:t+1}) \propto p(\mathbf{z}_{t+1}|\mathbf{x}_{k,t+1})p(\mathbf{x}_{k,t+1}|\mathbf{z}_{1:t}), \tag{6.17}$$

where $p(\mathbf{x}_{k,t+1}|\mathbf{z}_{1:t})$ represents the predicted belief of the k^{th} target (see Equation 6.3).

In what follows, we focus on the generation of the set of feasible joint association hypotheses as well as the computation of the β_{jk} coefficients.

6.3.3.1 Feasible (Joint Association) Hypothesis Generation

The purpose of this section is to describe how to generate the set Θ of feasible joint association hypotheses given a set of potential measurement-to-target associations. The overall procedure is described in Algorithm 6.2. Starting from a resulting set made of an empty set, the algorithm proceeds by screening each atomic observation composing $\mathbf{z}_{t+1}$ and iteratively updating the resulting set. For each atomic observation $\mathbf{z}_{j,t+1}$, each hypothesis currently within the resulting set is extended, on the basis of measurement-to-target associations involving $\mathbf{z}_{j,t+1}$, into a subset of new hypotheses such that the association constraints are not violated. Once all the hypotheses of the current resulting set have been processed, the so generated subsets of new hypotheses are merged together, thus forming the new resulting set from which the next atomic observation will be screened. In JPDAF, an atomic observation can be assigned to either a clutter or an existing target.

Algorithm 6.2: JPDA Hypothesis Generation Procedure

```
1  Algorithm JPDA_HYP_GEN(z_{t+1}, A)
2      //The parameter A is the set of possible measurement-to-target associations.
3      Θ̄_0 = {∅}
4      for j = 1 : |z_{t+1}| do
5          A_j = {a ∈ A s.t. a involves z_{j,t+1}}
6          Θ̄_j = Θ̄_{j−1}
7          for θ ∈ Θ̄_{j−1} do
8              for a ∈ A_j do
9                  if a does not violate the assignation constraints in θ then
10                     θ̄_temp = θ ∪ {a}
11                     Θ̄_j = Θ̄_j ∪ {θ̄_temp}
12     Θ = Θ̄_{M_{t+1}}
13     return Θ
```

6.3.3.2 Computation of the β_{jk} coefficients

Once Θ has been obtained, the next step consists in computing, for each j and k, β_{jk} which is the probability that $z_{j,t+1}$ originates from the k^{th} target (if $j > 0$) or the probability that the k^{th} target is undetected (if $j = 0$).

Let the pair $(j,k) \in \{0, \cdots, M_{t+1}\} \times \{1, \cdots, K\}$ be the representation of a measurement-to-target association which links together the j^{th} atomic observation to the k^{th} target. Let $\Theta_{jk} \subset \Theta$ with $\Theta_{jk} = \{\theta : (j,k) \in \theta\}$ be the set of all feasible joint association hypotheses in which the k^{th} target is associated with the j^{th} atomic observation. Then, we have

$$\begin{aligned} \beta_{jk} &= p(\Theta_{jk}|\mathbf{z}_{1:t+1}) \text{ (by definition)}, \\ &= \sum_{\theta \in \Theta_{jk}} p(\theta|\mathbf{z}_{1:t+1}). \end{aligned} \tag{6.18}$$

The main question of interest now is how to estimate $p(\theta|\mathbf{z}_{1:t+1})$. Given an FJAH θ, it is possible to retrieve the number N_{DT} of atomic observations associated with a target as well as the number N_{FT} of false alarms ($N_{FT} = M_{t+1} - N_{DT}$) as implied by θ. Besides, it is possible to determine the identity l_j of the target associated with the j^{th} atomic observation ($j > 0$). We have

$$l_j = \begin{cases} k & \text{if } (j,k) \in \theta, \\ 0 & \text{otherwise.} \end{cases} \tag{6.19}$$

Using the Bayes rule, $p(\theta|\mathbf{z}_{1:t+1})$ is expressed as

$$p(\theta|\mathbf{z}_{1:t+1}) \propto p(\mathbf{z}_{t+1}|\theta, \mathbf{z}_{1:t}) p(\theta|\mathbf{z}_{1:t}), \tag{6.20}$$

where $p(\mathbf{z}_{t+1}|\theta, \mathbf{z}_{1:t})$ represents the likelihood of the observation data $\mathbf{z}_{t+1}$ given the data-associations described by θ while $p(\theta|\mathbf{z}_{1:t})$ is the probability of θ given the observations received up to time step t.

The term $p(\mathbf{z}_{t+1}|\theta, \mathbf{z}_{1:t})$ of Equation 6.20 is given by

$$p(\mathbf{z}_{t+1}|\theta, \mathbf{z}_{1:t}) = \prod_{j=1}^{M_{t+1}} f_p(j), \tag{6.21}$$

where $f_p(j)$ is defined as

$$f_p(j) = \begin{cases} 1/V & \text{if } l_j = 0, \\ D_j^{l_j} & \text{otherwise,} \end{cases} \tag{6.22}$$

with $D_j^{l_j}$ being the predictive likelihood of the j^{th} observation with respect to the l_j^{th} target as given in Equation 6.4 and V being the volume of the environment's region where targets may evolve (cf. Section 6.2.2).

Assuming that the number N_{FT} of false alarms follows a Poisson distribution with a parameter $\lambda_{FT}V$ (cf. Section 6.2.2), the term $p(\theta|\mathbf{z}_{1:t})$ of Equation 6.20 is expressed as [Vermaak et al.,

2005]

$$p(\theta|\mathbf{z}_{1:t}) = \frac{N_{FT}!}{M_{t+1}!} \times P_D^{N_{DT}}(1-P_D)^{(K-N_{DT})} \times F_{pois}(N_{FT}, \lambda_{FT}V), \tag{6.23}$$

where $F_{pois}(.,.)$ represents the Poisson distribution function.

Substituting Equation 6.23 and Equation 6.21 into Equation 6.20 and replacing the Poisson process with its mathematical formula, we have

$$p(\theta|\mathbf{z}_{1:t+1}) \propto P_D^{N_{DT}}(1-P_D)^{K-N_{DT}} \times \lambda_{FT}^{N_{FT}} \times \left[\prod_{j:l_j \neq 0} D_j^{l_j}\right]. \tag{6.24}$$

As it can be noticed, Equation 6.24 is quite similar to Equation 6.15 obtained in MHT except that there is neither terms regarding the appearance of new targets (JPDAF assumes a fixed number of target over time) nor the prior association probability (JPDAF is not concerned with associations made at previous time steps) anymore.

The main bottleneck of JPDAF resides in the hypothesis generation procedure which may be computationally expensive as it was already the case with MHT. Works from the literature have focused on designing suboptimal strategies for reducing the complexity of such a procedure. Examples of these works include the deterministic suboptimal strategy in [Roecker and Phillis, 1993] and the Markov Chain Monte Carlo (MCMC) based strategy in [Oh et al., 2009]. Another limitation of JPDAF regards the assumption in which the number of targets under tracking is known and fixed over time. Approaches have been proposed to extend the JPDAF scheme in such a way to accommodate with situations where there is a time-varying and unknown number of targets [Bar-Shalom et al., 1989, Schulz et al., 2003, Musicki and Evans, 2004].

6.4 Management of Targets' Interactions

In the previous section, we presented prominent methodologies that have been developed in the literature to deal with the target association problem in the context of muti-target tracking. These methodologies, in their formulation, assume that targets behave independently from each other and are not concerned with issues related with targets' interactions. While this assumption can make sense for radar and sonar applications [Reid, 1979, Fortmann et al., 1983], it can not be generalized on application domains such as crowd monitoring or behavioral analysis. Indeed, in such domains, targets' interactions are prevalent and neglecting them may significantly degrade the quality of the tracking system, leading to the well known *coalescence* phenomenon [Yu and Wu, 2004] where individual (sub)filters lose their initial associated target and falsely associate with other targets.

In the literature, the term "interaction" may refer to two types of phenomena in the context of multi-target tracking:

- **Observation-based interactions** (Section 6.4.1): With respect to the problem description made in Section 6.2.2, the generation of a target-related atomic observation, instead of being made independently of other targets, may depend on other target atomic obser-

vations. This type of interaction well describes targets' mutual occlusions and it is more prevalent in visual tracking applications.

- **Dynamics-based interactions** (Section 6.4.2): Unlike observation-based interactions which focus on target observations, dynamics-based interactions concern the target states and describe targets' mutual influences on their respective individual behaviors over time.

The following sections are dedicated to the presentation of works from the literature for managing each of these types of interactions.

6.4.1 Observation-based Interactions

Approaches have been developed in the literature for solving the data-association issue while managing observation-based interactions in the context of visual tracking applications. These approaches, roughly speaking, can be regrouped into three categories depending on how these interactions are handled: potential function-based approaches, learning-based approaches, and interaction model-based approaches. The following sections are devoted to the description of each of these categories together with the presentation of associated works from the literature.

6.4.1.1 Approaches with observation-based potential functions

These approaches rely on potential functions for expressing the constraints regarding target-related observations. Basically, potential functions, which are usually pairwise, are used for modeling observations which mutually influence each other (e.g., they are sufficiently close or even occluded). As targets evolve over time, the structure of these influences is dynamic and is highly dependent on the spatial relations among targets' observations. Figure 6.5 depicts the underlying graphical model characterizing such approaches. In this figure, the undirected links between observations (red lines) represent their potentials at a given time step. Pairs of observations without direct links are considered as not mutually influencing one another.

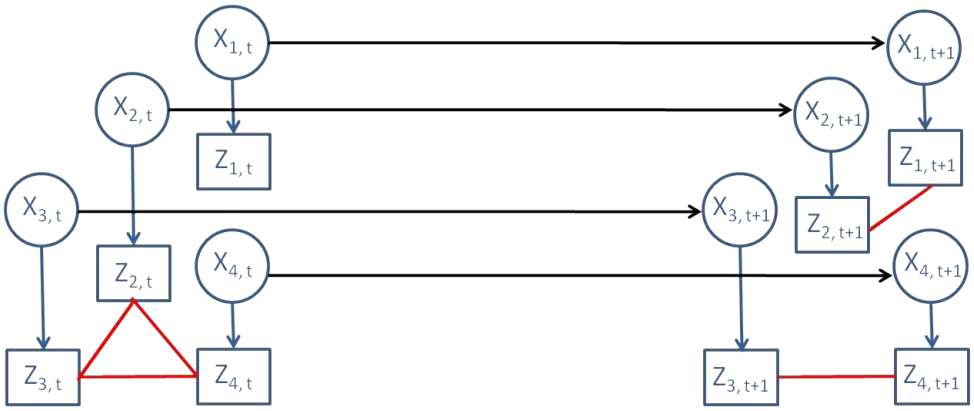


Figure 6.5: Approaches with observation-based potential functions. Circles represent target state and squares represent observations. Black arrows model individual target dynamics. Red lines represents target observations involving in a potential function at a given time step. The structure of the influences can evolve over time.

In [Qu et al., 2007], the authors consider tracking multiple targets, in the context of visual tracking in presence of occlusion phenomena, using multiple sub-filtering systems, each one dedicated to a specific target. In their approach, they seek to maintain unaltered the identity of the target associated to each sub-filter over time. To achieve their goal, they rely on an intuitive metaphor for explaining the coalescence phenomenon (the loss of the identity of the target initially associated to a given sub-filter which, consequently, falsely tracks another target). The metaphor states that the coalescence phenomenon can be explained by the presence of an invisible *gravitational force* among sub-filters whose purpose consists in attracting them to merge together when targets move closer and preventing them from disjoining when targets move apart. From this analogy with the gravitational theory, they propose to introduce a repulsive force to oppose the gravitational force in such a way that it can (1) prevent the sub-filters from falsely merging when targets are very close and/or (2) help the sub-filters detach easily from each other when targets move away. The repulsive force is then modeled using a *magnetic-inertia* potential [Hayt and Buck, 2011] on targets' (estimated) observations at each time step from which an interactive likelihood for each target is computed. These interactive likelihoods are subsequently used to perform a repulsion procedure among targets' (estimated) observations. At the end of the repulsion procedure, new estimates of the targets' observations are computed together with the corresponding interactive likelihoods. A new repulsion procedure is then performed and the overall loop is repeated until an equilibrium is reached. The authors claim that with the magnetic-inertia potential model, the interactive likelihood reduces the probability that target states' estimates occupy the same position in the state space. However, one concern of this approach is the convergence rate of the repulsion procedure at each iteration of the filtering process as low convergence rate may drastically increase the overall computational time and therefore prevent the approach from being used for online tracking.

Lanz and Manduchi [2005] propose a joint separable model to deal with occlusions while tracking multiple targets. The potential function they use is intimately dependent on the proximity of an individual target's estimated state with respect to the sensor position. The general idea is that, given two targets, in case of occlusions, it is more likely that the one closest to the sensor has covered the view of the other. Finally, they derive a particle filtering implementation of the proposed model which uses a representation size that grows linearly with the number of targets and a computational complexity that grows quadratically. However, their approach is proper to "mutual occlusions" and can not be extended to other forms of target dependencies.

6.4.1.2 Learning-based Approaches

The main idea behind learning-based approaches is to view the tracking process, in particular when the tracked targets are very similar in appearance, as a classification problem whose objective consists in determining to which class (target) the data to be processed (atomic observations) belong. In these approaches, when there is no ambiguity regarding the origin of a received atomic observation (e.g., the underlying target is far away from other targets), a learning procedure is launched for extracting features from the corresponding track (temporal sequence of atomic observations associated to a given target). These extracted features are then used in cases of ambiguities for maintaining unaltered the identity of each track.

In [Piater and Crowley, 2001], the authors are interested in tracking multiple objects or groups of objects in the presence of background noise and lighting variations. They propose a modular tracker architecture that combines the advantages of several simple and rapidly performing detection modules (tracking algorithms) such as a background-difference tracker, a color-histogram

tracker as well as a motion-history tracker. The result of each individual detection module are integrated by a recursive estimator which performs high-level control and analysis at the symbolic level in order to maintain the estimates of the underlying targets. Each target is described in terms of a Gaussian estimate of its spatial extent, i.e., the pixel coordinates of its center along with spatial covariance parameters. Such a description allows to define a region of interest of a target within the considered image fame from which a Gaussian mask is estimated. From this Gaussian mask, Gaussian approximations of known nearby targets are subtracted to allow individual tracking of interacting targets, that can be merged and split based on Mahalanobis distances (between their respective masks) [Mahalanobis, 1936]. The authors claim that their adaptive solution can robustly track at video frame rates several targets, each of which corresponds to one or more individual objects. However, as pointed out, the performance of the tracker degrades gracefully with increased system load.

In [Sullivan and Carlsson, 2006], the authors focus on the problem of localizing and maintaining the identities of players in a soccer game. They proceed by identifying periods of time and trajectories when players are isolated. These trajectories, also called players tracks, are then extracted and recorded. For each of these tracks, they define a feature vector which encodes the relative spatial position of the associated player with respect to his team-mates. Subsequently, using appearance and relative depth ordering properties together as well as the motion continuity property, they identify situations in which player tracks "merge" together or "split" from each other. These merges and splits between tracks are recorded to form a large graph, also referred to as a track graph (see Figure 6.6), summarizing the whole game. Finally, a clustering procedure is performed on the basis of the extracted feature vectors, thus allowing temporally separated trajectories of each player to be linked through the analysis of the recorded track graph.

Following the work done in [Sullivan and Carlsson, 2006], instead of using a clustering approach for identifying complete player trajectories within the track graph, Nillius et al. [2006] propose to formulate the problem as a Bayesian network inference problem in which they are able to use standard message propagation [Pearl, 1982] to find the most probable set of paths in an efficient way. To achieve their objective, they exploit the constraints imposed by the graph structure (e.g., which players have merged together) as well as the extracted feature vectors describing the appearance of each target. In their approach, the high complexity unavoidable in large problems is reduced by leveraging dependency links between tracks in the graph. A limitation of both works [Sullivan and Carlsson, 2006, Nillius et al., 2006] regards their applicability in case of on-line tracking situations.

For managing on-line tracking situations, Song et al. [2008] present a supervised learning based method in the context of visual tracking. Their approach is based on the idea that learning and tracking can be integrated in such a way to supplement each other in a unique framework to deal with various complex tracking issues. In the proposed solution, when targets are isolated from each other, multiple independent filters are employed for training a classifier for each target. Given a target's classifier, information from other targets is used as negative samples over time during the learning process in such a way that each classifier presents a strong distinguishability property. Next, when targets are in close proximity or present partial occlusions, the learned classifiers are used to assist in tracking. Specifically, when targets merge together, a new state space representing "merged targets" is assigned and tracked as one target. When the targets split, their classifiers are used to specify a correct identification of the resulting tracks. A general overview of their approach is described in Figure 6.7 in the simple case of two targets. The proposed approach is suitable when the tracking targets are different in appearance; however, in

case of targets with similar appearance, the approach will perform poorly.

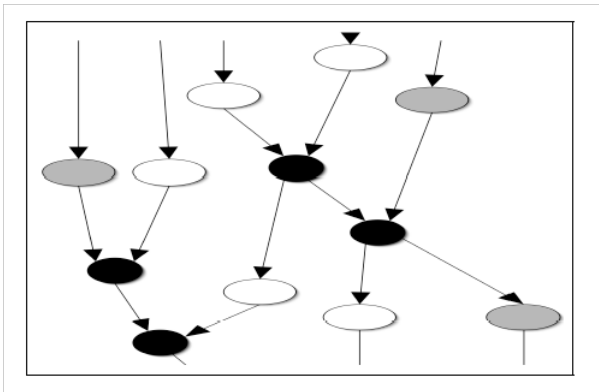


Figure 6.6: A portion of a track graph summarizing a football game. White/gray nodes correspond to team A/B player tracks and black nodes to multiple player tracks. Edges indicate when tracks interact. From [Sullivan and Carlsson, 2006].

6.4.1.3 Interaction model-based Approaches

Unlike previously presented approaches, interaction model-based approaches explicitly represent the probabilistic model of the occlusion phenomenon as well as their impact within the target dynamics. In [del Blanco et al., 2011], the authors are interested in visually tracking targets whose trajectories change during occlusions. To achieve their objective, they propose a Bayesian framework that explicitly models the occlusions for predicting different target trajectories on the basis of well estimated set of data association hypotheses. Basically, in their approach, the occlusions are modeled as a random variable

$$\mathbf{o}_t = \{\mathbf{o}_{k,t} | k = 1, \cdots, K_t\}$$

where each component stores the occlusion information for each target, that is,

$$\mathbf{o}_{k,t} = \begin{cases} l & \text{if the } k^{th} \text{ target is occluded by the } l^{th} \text{ target,} \\ 0 & \text{if the } k^{th} \text{ target is not occluded.} \end{cases}$$

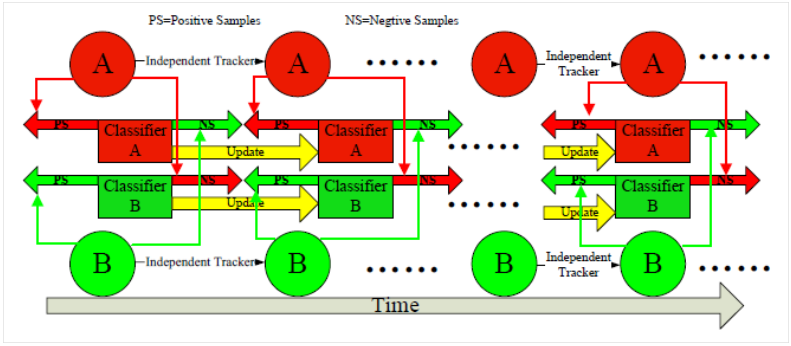
Also, the data association is modeled by a random variable

$$\mathbf{a}_t = \{\mathbf{a}_{j,t} | j = 1, \cdots, M_t\}$$

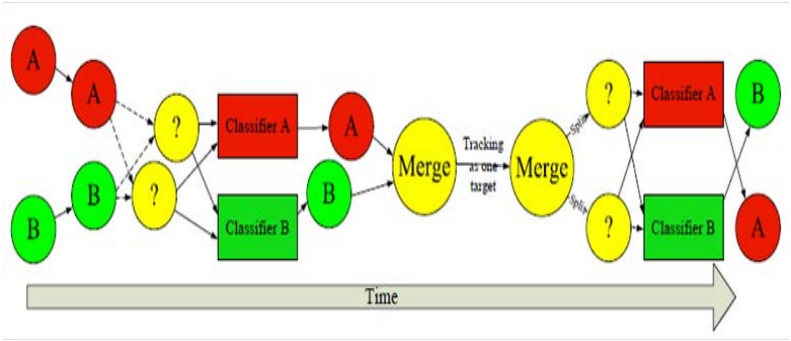
where each component specifies the origin of the j^{th} atomic measurement, that is,

$$\mathbf{o}_{j,t} = \begin{cases} k & \text{if the } k^{th} \text{ target engendered the } j^{th} \text{ atomic observation,} \\ 0 & \text{if the } j^{th} \text{ atomic observation is from clutter.} \end{cases}$$

Finally, since target trajectories may change during occlusions, they represent the dependencies of the variables $\mathbf{a}_t$ and $\mathbf{o}_t$ with respect to the global state $\mathbf{x}_t$ of all the targets and the observation $\mathbf{z}_t$ according to the graphical model depicted in Figure 6.8. On the basis of this graphical model,



(a)



(b)

Figure 6.7: Supervised learning based tracking procedure. In Fig. 6.7a, when A and B do not interact with each other, independent trackers are employed to track them and the tracking results are used for learning. For each target's classifier, the negative samples are based on the other target. In Fig. 6.7b, when the targets are in close proximity or a "merge/split" condition occurs, the classifiers are used to assist in the tracking. From [Song et al., 2008].

they end up with an expression of the target posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = \sum_{\mathbf{a}_t} \sum_{\mathbf{o}_t} p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_{1:t}), \tag{6.25}$$

with

$$\begin{aligned} p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_{1:t}) &= p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_t, \mathbf{z}_{1:t-1}), \\ &\propto p(\mathbf{z}_t|\mathbf{x}_t, \mathbf{a}_t) p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_{1:t-1}). \end{aligned} \tag{6.26}$$

The term $p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_{1:t-1})$ represents the prior distribution that predicts the evolution of $\{\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t\}$ between consecutive time steps using the joint posterior distribution at the previous time step $p(\mathbf{x}_{t-1}, \mathbf{a}_{t-1}, \mathbf{o}_{t-1}|\mathbf{z}_{1:t-1})$ and it is expressed as

$$p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_{1:t-1}) = \int \sum_{\mathbf{a}_{t-1}} \sum_{\mathbf{o}_{t-1}} p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t|\mathbf{z}_{1:t-1}, \mathbf{x}_{t-1}, \mathbf{a}_{t-1}, \mathbf{o}_{t-1}) p(\mathbf{x}_{t-1}, \mathbf{a}_{t-1}, \mathbf{o}_{t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{t-1}, \tag{6.27}$$

where $p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{t-1}, \mathbf{a}_{t-1}, \mathbf{o}_{t-1})$ is factorized according to the graphical model as

$$p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t | \mathbf{z}_{1:t-1}, \mathbf{x}_{t-1}, \mathbf{a}_{t-1}, \mathbf{o}_{t-1}) = p(\mathbf{x}_t | \mathbf{x}_{t-1}, \mathbf{o}_t) p(\mathbf{a}_t) p(\mathbf{o}_t | \mathbf{x}_{t-1}). \tag{6.28}$$

Subsequently, after expressing $p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t | \mathbf{z}_{1:t})$ as $p(\mathbf{x}_t, \mathbf{a}_t, \mathbf{o}_t | \mathbf{z}_{1:t}) = p(\mathbf{x}_t | \mathbf{z}_{1:t}, \mathbf{a}_t, \mathbf{o}_t) p(\mathbf{a}_t, \mathbf{o}_t | \mathbf{z}_{1:t})$, they assume that the target dynamics can be acceptably simulated by a constant velocity model with Gaussian perturbations if the occlusions and the data association are known, that is $p(\mathbf{x}_t | \mathbf{z}_{1:t}, \mathbf{a}_t, \mathbf{o}_t)$ is conditionally Gaussian with an analytic expression computed from the Kalman filter. From this assumption, they use the Rao-Blackwellization strategy [Doucet et al., 2000] to derive a particle filter for computing $p(\mathbf{a}_t, \mathbf{o}_t | \mathbf{z}_{1:t})$ and therefore approximating the global posterior density $p(\mathbf{x}_t | \mathbf{z}_{1:t})$. The authors claim that their approach is able to obtain accurate estimates in situations in which the targets change their trajectories while they are occluded. However, their approach reasons on the global target state and may turn out to be inefficient in state spaces with high dimension.

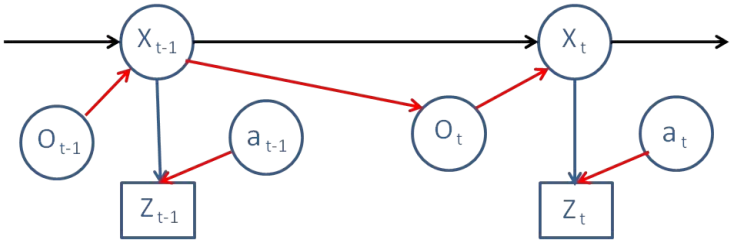


Figure 6.8: Graphical model for occlusion-based interactions among targets. Black and blue arrows show the usual temporal dependencies between state and observation variables. Additional dependencies are represented by red arrows. From [del Blanco et al., 2011].

6.4.2 Dynamics-based Interactions

Roughly speaking, approaches found in the literature for handling dynamics-based interactions can be regrouped into two categories. Like it was previously the case with observation-based interactions, we distinguish between potential function-based approaches and interaction model-based approaches. In what follows, we present the works belonging to each of these categories.

6.4.2.1 Approaches with state-based potential functions

Like approaches with observation-based potential functions described in Section 6.4.1.1, approaches belonging to this category use potential functions to formally specify the influences (or constraints) that can exist among targets on the basis of the knowledge related to the application domains. However, instead of defining the potential functions on observable variables, these approaches explicitly represent the constraints over target states. Basically in such approaches, as illustrated in Figure 6.9, the targets are handled in a first stage as if they were behaving independently from each other. Subsequently, in a second stage, the potential functions, which are usually pairwise, are used for accounting the constraints characterizing the application domain.

In [Khan et al., 2003], the authors consider the application of tracking ants in an arena. In order to represent constraints such that ants cannot overlap with each other and actively avoid collisions, they use a motion prior defined on the basis of Markov random fields (MRF) [Li, 2009].

In their work, the MRF is described by an undirected graph (V, E) where nodes represent targets, and an edge between node k and node l symbolizes the existence of a local interaction between targets l and k. At each time step, an MRF is built on the fly using target state estimates and it is used for "adjusting" the posterior distribution over target states. More specifically, the dynamics of the targets under tracking is represented as

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) \propto \prod_i p(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) \prod_{ij \in E} \psi(\mathbf{x}_{i,t}, \mathbf{x}_{j,t}), \tag{6.29}$$

where the first term $\prod_i p(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1})$ simulates the targets' independent behaviors and the second term $\prod_{ij \in E} \psi(\mathbf{x}_{i,t}, \mathbf{x}_{j,t})$ corresponds to interaction potentials representing *non-overlapping* constraints. Moreover, they propose to express these potentials by means of the Gibbs distribution as

$$\psi(\mathbf{x}_{i,t}, \mathbf{x}_{j,t}) \propto \exp(-g(\mathbf{x}_{i,t}, \mathbf{x}_{j,t})),$$

where $g(\mathbf{x}_{i,t}, \mathbf{x}_{j,t})$ is a penalty function modeling the domain application, in this case the distance between overlapping targets. The function $g()$ is maximal when two targets (ants) coincide and gradually falls off as targets move apart. Finally, for estimating the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, they propose to use multiple nearly independent particle filters — one per target — which interact at each weighting stage of the filtering process. At this stage, for scoring a particle of a filter dedicated to the k^{th} target, estimated states of other filters are taken into account for computing the interaction potentials. However, their approach presents a major drawback: since the effects of the interaction potentials are defined at time step t, there is a waste of computational resources and time for computing non viable particles.

To overcome this limitation and improve the quality of the tracking results, the same authors propose in [Khan et al., 2004] an extension of their work in which a unique particle filter on the joint state $\mathbf{x}_t$ is used. In this approach, they replace the classical importance sampling step of the filtering procedure — which is inefficient in high dimensional state spaces — by a Markov chain Monte Carlo (MCMC) sampling strategy [Gilks and Spiegelhalter, 1996] in which the interaction potentials are used at the prediction step to compute the acceptance ratio of a target's state generated by the proposal density of the filter. While this approach seems to result in better target estimates, one limitation regards the number of Monte Carlo samplings to perform for obtaining such results. Besides, their factorization (Equation 6.29) is specific to the considered problem and can be hard to generalize to other problems, otherwise the question regarding how to obtain the potential function $\psi(.)$ still remains open.

The idea of having nearly independent filters collaborating only under certain conditions for tracking interacting targets is carried further in the work of Yu and Wu [2004] in which a tracking algorithm is proposed to cope with the coalescence problem in visual tracking applications. In their approach, the authors represent the joint prior of the tracked objects using a Markov random network, a special case of Markov random field, and consider the interaction issues as a competition mechanism that enables different targets to compete for the common resources — here, image observations — since the existence of two different targets cannot be supported by the same piece of image evidence. In other words, if one target occupies a region in the state space, it will lower the probability for other targets to occupy the same region. Unlike [Khan et al., 2004] where the MRF structures are built on the fly at each time step, the authors consider the use of variational analysis inference methods [Jordan et al., 1999, Jaakkola, 2000] to iteratively approximate individual target posterior distributions by decreasing at each iteration

the Kullback-Leibler divergences between consecutive approximations until an equilibrium is reached. At each iteration, the approximation of a given target distribution is performed by taking into account not only its proper dynamics and image likelihood, but also influences of other neighboring targets competing for image resources against him. More specifically, given a target i, its posterior distribution $p(\mathbf{x}_{i,t}|\mathbf{z}_{1:t})$ is approximated using $Q_{i,t}(\mathbf{x}_{i,t})$ such that

$$Q_{i,t}(\mathbf{x}_{i,t}) \propto p(\mathbf{z}_{i,t}|\mathbf{x}_{i,t}) \times \left[\int p(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1}) Q_{i,t-1}(\mathbf{x}_{i,t-1}) d\mathbf{x}_{i,t-1} \right] \times M_{i,t}(\mathbf{x}_{i,t}), \tag{6.30}$$

where $M_{i,t}(\mathbf{x}_{i,t})$ accounts for neighboring target influences and is given by

$$M_{i,t}(\mathbf{x}_{i,t}) = \exp\Big(\sum_{j:(i,j)\in E} \int_{\mathbf{x}_{j,t}} Q_{j,t}(\mathbf{x}_{j,t}) \log \psi(\mathbf{x}_{i,t}, \mathbf{x}_{j,t}) d\mathbf{x}_{j,t} \Big). \tag{6.31}$$

Because of the mutual dependency between $Q_{i,t}(\mathbf{x}_{i,t})$ and $Q_{j,t}(\mathbf{x}_{j,t})$ for a pair $(i,j) \in E$, variational analysis methods allow, in a first stage, to estimate $Q^0_{i,t}(\mathbf{x}_{i,t})$, an approximation of $Q_{i,t}(\mathbf{x}_{i,t})$ where the influence $M_{i,t}(\mathbf{x}_{i,t})$ of other targets is neglected, for all the targets. Then in a second stage, initiating from these estimates, subsequent $Q^r_{i,t}(\mathbf{x}_{i,t})$ values are iteratively computed while taking into account other target influences (obtained using $Q^{r-1}(.)$), for all targets until reaching an equilibrium. This way of performing inference process is known as mean field inference. Finally, for practical purposes, the authors develop a mean field Monte Carlo algorithm, a Monte Carlo approach of the mean field inference which allows simulating the competition among a set of low dimensional particle filters. One limitation of this approach regards the convergence rate of the mean field inference procedure at each iteration of the filtering process. Indeed, a low convergence rate may prevent the algorithm from being applied in an online setup.

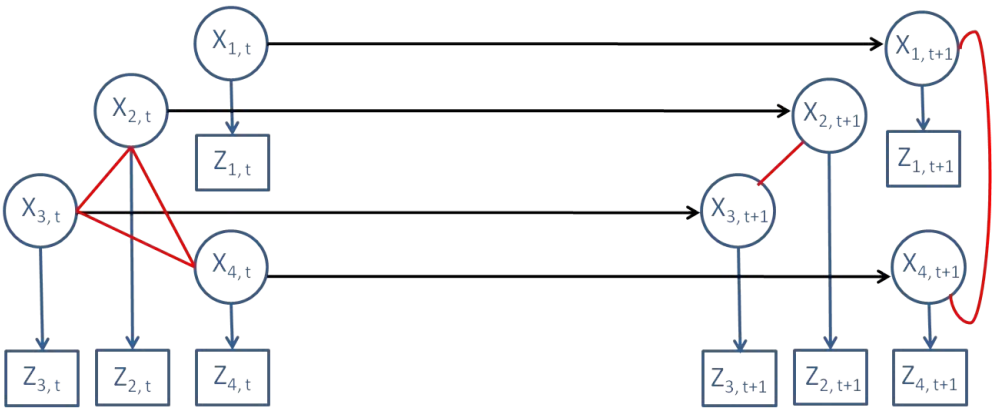


Figure 6.9: Approaches with state-based potential functions. Circles represent target states and squares represent observations. Black arrows represent individual target dynamics. Red lines describe potential functions involving pairs of targets. The structure of the influences can evolve over time.

6.4.2.2 Interaction Model-based Approaches

Unlike approaches presented in the previous section, interaction model-based approaches specifically integrate the interaction model within the dynamics of the targets. Therefore, targets are not anymore handled as if they behave independently from each other and each individual

target's dynamics cannot be expressed in the form $p(\mathbf{x}_{i,t}|\mathbf{x}_{i,t-1})$, where i refers to the target identity.

In [Cattelani et al., 2014], the authors propose to explicitly represent the relationships among the targets under tracking using relational dynamic Bayesian networks [Manfredotti et al., 2011]. They extend the target state space with the "relations" that can exist between entities. By the term "relation", they refer to any property that relates two or more targets. In this way, they are able to model interactions between targets and also how these interactions evolve over time. More specifically, the overall state $\mathbf{x}_t$ at each time step t is composed of two parts: the state of the attributes of the different targets, $\mathbf{x}_t^a$, and the state of the relations between the targets, $\mathbf{x}_t^r$. That is

$$\mathbf{x}_t = (\mathbf{x}_t^a, \mathbf{x}_t^r).$$

From this decomposition, the authors consider the dependency assumptions that are illustrated in Figure 6.10. These assumptions state that:

- the relations are not directly observable, that is $p(\mathbf{z}_t|\mathbf{x}_t) = p(\mathbf{z}_t|\mathbf{x}_t^a)$;
- the relations at time step t only depend on the relations at time step $t-1$ as well as the target attributes at time step t;
- the attributes at time step t depend on attributes and relations at time step $t-1$ but not on relations at time step t. They motivate this assumption by the fact that relations between targets may affect their dynamics and therefore their attributes.

From these assumptions, the authors express the overall target dynamics as

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = p(\mathbf{x}_t^a, \mathbf{x}_t^r|\mathbf{x}_{t-1}^a, \mathbf{x}_{t-1}^r) = p(\mathbf{x}_t^r|\mathbf{x}_t^a, \mathbf{x}_{t-1}^r)p(\mathbf{x}_t^a|\mathbf{x}_{t-1}^a, \mathbf{x}_{t-1}^r),$$

and they end up with an expression of the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) = p(\mathbf{x}_t^a, \mathbf{x}_t^r|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_t^a) \int \left[p(\mathbf{x}_t^r|\mathbf{x}_t^a, \mathbf{x}_{t-1}^r)p(\mathbf{x}_t^a|\mathbf{x}_{t-1}^a, \mathbf{x}_{t-1}^r)p(\mathbf{x}_{t-1}^a, \mathbf{x}_{t-1}^r|\mathbf{z}_{1:t-1})\right] d\mathbf{x}_{t-1}. \tag{6.32}$$

For practical issues, they derive a particle filter on the basis of Equation 6.32 to approximate the posterior distribution of the targets. However, a drawback of their approach resides in the use of a unique filter for estimating the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$, which may turn out to be inefficient in state spaces with high dimension.

6.4.3 Summary

In the previous sections, we presented existing works from the literature dedicated to handling targets' interactions — both observation-based interactions and dynamics-based interactions — in the context of the multi-target tracking problem. These works, despite the nature of the interactions they are interested in, can be classified into three categories: (1) potential function based approaches, (2) learning based approaches, and (3) interaction models based approaches.

Most of the works belonging to the first two categories use nearly independent filters, each one dedicated to a single target, and which collaborate only under certain conditions for tracking

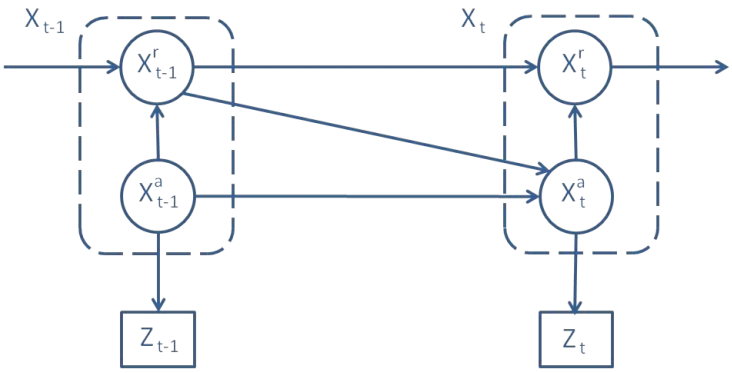


Figure 6.10: Relational dynamic model of interacting targets. $\mathbf{x}_{\cdot}^{a}$ represents the attributes of the targets at a given time step while $\mathbf{x}_{\cdot}^{r}$ models the relationships that may exist among targets. Arrows indicate probabilistic dependencies between variables. From [Cattelani et al., 2014]

interacting targets. While this methodology decreases the complexity of the tracking problem, these approaches suffer from the assumption that, on the one hand, the dynamics of individual targets is independent from other targets and, on the other hand, interactions can either be modeled as constraints over the state space (potential function based approaches) or be resolved using individual target characteristics learned from temporal sequences of related data (learning based approaches). Due to these assumptions, the mathematical formulations developed in each of these works is specific to the problem considered and therefore, it can not be easily generalized.

Regarding works falling into the third category, unlike other approaches, they assume a model of interactions between targets and therefore they imply that the behavior of a target depends on other targets' behaviors. This assumption mainly corresponds to what happens in most real-world scenarios, and therefore these works can be generalized in other domains. However, for these works, a unique filter is considered for estimating the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ of all the involved targets; and thus, they suffer from high computational complexity issues.

A classification of the presented works based on their computational complexity and their capability to be generalized is depicted in Table 6.1.

6.5 Conclusion

In this chapter, we introduced the multi-target tracking problem and the related challenges, that is the data association problem and the management of targets' interactions. Then, we described the most notable methodologies that have been developed for handling the data association problem. These include the well known multiple hypothesis tracker (MHT) and the joint probabilistic data association (JPDA) strategies. After that, we focused on the special case of interacting targets and presented a synthesis of works existing on the topic. As specified in Section 6.4.3 and Table 6.1, the resulting approaches from these works either make problem-specific assumptions at the cost of being less generalizable, which allow to derive nearly independent filters with low computational complexity in the tracking process, or embed interactions within the target dynamics, thus resulting in a more generalized solution but this time with high computational cost. In the next chapter, we are interested in the problem of tracking multiple interacting targets from the point of view of dynamics-based interactions. Unlike previous approaches, we will describe a

Table 6.1: A classification of works for managing interaction in the MTT problem. The superscripts in front of each reference indicates whether they tackle observation-based interactions (1) or dynamics-based interactions (2).

	Non interaction Model (specific)	Interaction model (generalizable)
Unique filter (high complexity)	[Khan et al., 2004][2]	[Cattelani et al., 2014][2], [del Blanco et al., 2011][1]
Nearly independent filters (low complexity)	[Khan et al., 2003][2], [Yu and Wu, 2004][2], [Lanz and Manduchi, 2005][1], [Sullivan and Carlsson, 2006][1], [Qu et al., 2007][1], [Song et al., 2008][1] [Piater and Crowley, 2001][1]	

novel solution based on interaction models with the particularity that the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ is approximated using nearly independent filters, an approach that does not exist yet in the literature (see Table 6.1).

Chapter 7

Tracking Multiple Interacting Targets - An Interaction-Model Based Factored Approach

Contents

7.1 Introduction

This chapter is devoted to the design of an original approach for tracking multiple autonomous and interacting targets. We focus on dynamics-based interactions, and therefore we are not concerned with observation-based interactions.[14] Our solution is particular in that we seek to integrate a generic interaction model within the filtering system while avoiding, at the same time, the complexity related to the curse of dimensionality. The latter, in the context of multi-target tracking, usually occurs when the reasoning is performed on the overall system's state composed of individual states of the underlying targets.

Putting it differently, our purpose is to elaborate a system capable of handling dynamics-based target interactions and which can be viewed as a collection of collaborative sub-filtering systems, each one committed to a particular target. As in Chapter 5, we consider the general case where we assume the availability of a generic simulator or a function modeling the targets' dynamics including their mutual interactions. With respect to the general scheme of a filtering process, the work we present in this chapter mostly falls within the prediction step of the process. In order to assess the impact of our contribution on the tracking efficiency, we consider, for simplicity, situations in which the number of targets under tracking is known and fixed, and we choose to rely on the JPDA (Joint Probabilistic Data Association) framework — see Section 6.3.3 — for solving the data association issues at the correction step of the filtering process.

The remainder of this chapter is organized as follows. Section 7.2 formally introduces the problem we are interested in as well as the objective we seek to achieve. In Section 7.3, we present the main assumptions made and describe the generic model we designed to represent dynamics-based target interactions within our solution. Section 7.4 describes in detail the design of our solution and presents the different heuristics used to reduce its computational complexity. The implementation of the proposed approach using the particle filtering paradigm within the JPDA framework is discussed in Section 7.5, while Section 7.6 is dedicated to experimental evaluations. Finally, Section 7.7 succinctly presents the conclusion derived from these evaluations and identifies some important research lines induced by this work.

7.2 Problem Statement

Let us consider a closed environment equipped with a sensor network in which multiple autonomous and interacting targets evolve. The number K of targets in the environment does not change over time. Also, we consider the general case where the environment under consideration may contain areas which are not under sensory coverage. Recalling from the notations used in Chapter 6, let $\mathbf{x}_{k,t}$ be the state of the k^{th} target at time step t. The state $\mathbf{x}_t$ of the overall system, defined as $\mathbf{x}_t = \{\mathbf{x}_{k,t}\}_{k=1}^{K}$, is the collection of the states of individual targets in the environment. Finally, let $\mathbf{z}_t = \{\mathbf{z}_{1,t}, \mathbf{z}_{2,t}, \cdots, \mathbf{z}_{M_t,t}\}$ denote the observation data received at time t from the sensor network and $\mathbf{z}_{1:t}$ represent the sequence of all the observations received so far. The problem we are interested in consists in efficiently estimating, at each time step t, the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0)$ of the target states given the sequence $\mathbf{z}_{1:t}$ of observations received so far and the prior knowledge Bel_0 regarding the state of each target at time step $t = 0$. For simplicity, the term Bel_0 will be omitted in subsequent notations.

[14] When not explicitly qualified, the term "interaction" is used in this chapter to refer to dynamics-based interactions.

To avoid the computational complexity issues related to high dimensional state spaces, we consider a factorized approach in which we try to approximate the overall posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ using K (sub-)filters, each one being dedicated to a unique target. In other words, we plan to approximately represent the posterior distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ as

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \bar{p}(\mathbf{x}_t|\mathbf{z}_{1:t}) = \prod_{k=1}^{K} p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t}), \tag{7.1}$$

where each $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$ is the posterior distribution corresponding to the k^{th} target and it is obtained via the associated (sub-)filtering system.

However, as targets may potentially influence each other's behavior, the main issue we are concerned with here is how to efficiently approximate $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$ by such a probability distribution factorization while taking into account dynamics-based target interactions. Recalling from Chapter 2 (Section 2.2), focusing on the (sub-)filtering process associated to the k^{th} target, the corresponding estimated distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$ is obtained in two steps (assuming available the $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$):

- **a prediction step** in which the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ of the k^{th} target is computed from the posterior distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ obtained on the basis of observation data received up to time step $t-1$;
- **a correction step** in which the previously computed predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ is adjusted on the basis of the received observation $\mathbf{z}_t$, leading thus to the posterior distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$.

While the second step can be handled using any data association technique presented in Section 6.3 under the assumption that the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ is available, we consider in this chapter the JPDA framework as, on one hand, it is originally designed to cope with situations where the number of targets is fixed and known, and on the other hand, it avoids the storage complexity related to the MHT algorithm. Nevertheless, computing the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ at the prediction step still remains a real challenge. Indeed, at this step, in order to efficiently approximate the true distribution regarding the state of the corresponding target, interactions have to be taken into account. In the next section, we present the representation used to describe dynamics-based target interactions with the aim to efficiently estimate the predicted distribution of each target.

7.3 Dynamics-based Interaction Representation

As previously stated, we are interested in tracking a fixed number K of interacting targets within a given environment. As targets are allowed to mutually interact, the behavior exhibited by the k^{th} target may be influenced by other targets present in the environment. We propose to represent such dynamics-based interactions using **Markovian conditional dependencies** [Daduna and Szekli, 1995, Pattison and Robins, 2002] over target states, that is, we assume that *the evolution of a given target state at the next time step depends solely on all target states at the previous time step*. Following this assumption, the overall target dynamics can be put in the

form

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{k=1}^{K} p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1}). \tag{7.2}$$

More specifically, we assume, for each target k, the availability of a generic function that, given any total number of targets, is able to compute (or simulate) the evolution of target k while accounting for the influence of other targets. Let $F^k_{t,K}$ be the dedicated discrete-time dynamics of the k^{th} target. Then, its proper dynamics can be expressed as

$$\mathbf{x}_{k,t} = F^k_{t,K}(\mathbf{x}_{k,t-1}, \mathbf{x}_{\neg k,t-1}, \mathbf{w}_{k,t}), \tag{7.3}$$

where $\mathbf{x}_{\neg k,t-1}$ represents the collection of the states of all the targets in the environment except target k and $\mathbf{w}_{k,t}$ is the process noise specific to target k. The index K in the notation of the function F ($F^{\cdot}_{\cdot,K}$) denotes the number of targets involved in the computation of $\mathbf{x}_{k,t}$. From a probabilistic point of view, the k^{th} target's dynamics described in Equation 7.3 is well represented by a probability distribution of the form $p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\neg k,t-1})$. This way of modeling dependencies between targets is realistic and it is very common in physical systems [Harel and Pnueli, 1985, Reynolds, 1999].

However, as illustrated in Figure 7.1, it is usual that not all the targets in the environment influence the behavior of a given target at every time step. Indeed, targets are usually affected by the only targets within their neighborhood, an area which is defined according to the application domain. Thereby, it is possible to dynamically refine the probability distribution $p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\neg k,t-1})$ characterizing the k^{th} target in such a way to focus only on relevant targets. How to efficiently proceed with such a refinement at the prediction step of the filtering process is the core of the proposed approach and it is discussed in the next section.

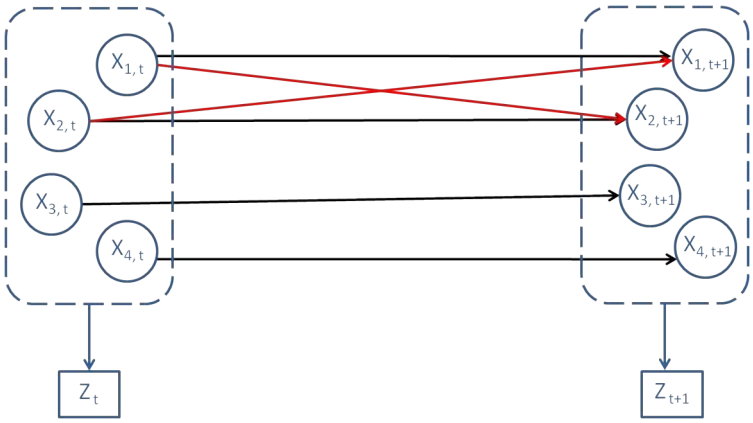


Figure 7.1: Graphical model representing dynamics-based target interactions. The circles represent individual target states while squares denote observation data. Arrows correspond to temporal dependencies between state variables with the red ones modeling targets' interactions. In this figure, it is assumed that only target 1 and 2 are mutually interacting while the other targets behave independently.

7.4 Estimation of Targets' Predicted Distributions

In the previous section, we assumed a generic form of dynamics-based target interactions and we introduced the graphical model used for representing them. The purpose of this section consists in leveraging the structure of such an interaction model to derive an efficient approximation of the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ characterizing each target k in the environment. From now on, we assume the availability of the posterior distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ characterizing the state of each target k at time step $t-1$ given the observations received up to that time step.

One possible approach for estimating the predicted distributions of the individual targets consists in (1) combining the individual posterior distributions $\{p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})\}_{k=1}^{K}$ into a unique distribution $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ on the global system's state, (2) estimating the predicted belief $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ while taking into account the potential interactions inscribed within the structure of the global system's state, and (3) finally decomposing the resulting predicted distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t-1})$ into a collection of marginalized individual predicted distributions $\{p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})\}_{k=1}^{K}$. However, this approach (*i*) is computationally expensive due to the high dimensional state space in which the estimation of the predicted belief is performed, and (*ii*) ruins the whole point of a factorization.

Instead, in this work, we seek to directly approximate individual targets' predicted distributions by focusing on the state space corresponding to each target. We propose an approach which mostly relies on the notion of *local interactions* to derive such approximations. More specifically, given a target k, the approach determines the subset of targets in the environment which may influence its dynamics, and subsequently relies on this subset for computing its predicted distribution. This is done according to a procedure (Section 7.4.3) which is summarized by the following steps:

- **Probability distribution aggregation**: First, we propose to partition each individual's posterior distribution $p(\mathbf{x}_{\cdot,t-1}|\mathbf{z}_{1:t-1})$ into coarse but pertinent regions of interest which are then aggregated into weighted representative states, one per region.
- **Computation of the predicted distribution**: Then, based on the computed representatives, we compute the subset of targets with which target k may potentially interact. These targets are those at least one representative of whom falls within the neighborhood — defined according to the application domain — of a (representative) state of target k. Once this subset of targets has been computed, their corresponding representatives are subsequently used for approximating the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$.

This section is organized as follows. In Section 7.4.1, the Bayesian formulation characterizing the predicted distribution of a given target under the assumptions made in Section 7.3 is described. After that, Section 7.4.2 focuses on the general idea governing the notion of *local interactions* and describes how this idea can be used for reducing the computational complexity of such a Bayesian formulation. Then, in Section 7.4.3, the general intuition behind the aggregation of probability distributions is discussed and the resulting formulation of the estimated predicted distribution is presented. Finally, Section 7.4.4 is dedicated to heuristics for effectively aggregating probability distributions.

7.4.1 Bayesian formulation of target predicted distributions

From the Bayesian perspective and considering the assumptions made in Section 7.3 (see Equation 7.2), the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ of the k^{th} target is computed as

$$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1})p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{t-1}, \tag{7.4}$$

where $p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1})$ refers to the k^{th} target's dynamics which takes into account potential dependencies with other targets in the environment, and $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ is the posterior distribution of the global system's state at time step $t-1$. However, as assumed in Equation 7.1, $p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1})$ is approximated according to the following equation

$$p(\mathbf{x}_{t-1}|\mathbf{z}_{1:t-1}) = \prod_{l=1}^{K} p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1}), \tag{7.5}$$

where $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$, as previously mentioned, is the individual posterior distribution of the l^{th} target at time step $t-1$. Putting Equation 7.5 into Equation 7.4, we then have

$$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{1,t-1}} \cdots \int_{\mathbf{x}_{K,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1}) \prod_{l=1}^{K} p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{1,t-1} \cdots d\mathbf{x}_{K,t-1}. \tag{7.6}$$

Finally, when highlighting the state $\mathbf{x}_{k,t-1}$ of the considered target at time step $t-1$, Equation 7.6 can be rewritten as

$$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{k,t-1}} Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{k,t-1}, \tag{7.7}$$

where $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$ is what we call the **interactive likelihood transition** of $\mathbf{x}_{k,t}$ with respect to $\mathbf{x}_{k,t-1}$ and the history $\mathbf{z}_{1:t-1}$ of observations — which can be viewed as the general dynamics of target k — and it is defined as

$$Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{\neg k,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\neg k,t-1})p(\mathbf{x}_{\neg k,t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{\neg k,t-1}, \tag{7.8}$$

with $\mathbf{x}_{\neg k,t-1}$ being, as previously mentioned, a random variable representing the collection of the states of all the targets in the environment except target k, and $p(\mathbf{x}_{\neg k,t-1}|\mathbf{z}_{1:t-1})$ being given by

$$p(\mathbf{x}_{\neg k,t-1}|\mathbf{z}_{1:t-1}) = \prod_{l:l\neq k} p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1}). \tag{7.9}$$

However, evaluating the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ using Equation 7.7 still remains a real challenge. Indeed, the expression represented by Equation 7.8 can be computationally expensive in the general case where the individual target distributions do not have any particular closed form. In what follows, we define heuristics for reducing the complexity of such an expression without highly degrading the quality of the obtained probability distribution.

7.4.2 Exploiting the locality of interactions

Generally, interactions are very localized in the environment and they do not usually involve all the targets therein. In such a configuration, it is very common to observe a structure of targets' interactions described by a collection of groups of targets such that, on the one hand, targets belonging to a same group are part of the same neighborhood and mutually influence each other, while on the other hand, targets belonging to different groups do not interact with each other. Of course, under this assumption, a target which is the only member of its group behaves independently as if it was alone in the environment. In this work, we propose to capture such a structure of interactions using what we call an *interaction graph* whose graphical representation is very similar to the one of the pairwise Markov random field (MRF) used in [Khan et al., 2004] (see Section 6.4.2.1). Formally, an interaction graph is a graph (V, E) where nodes represent targets, and an edge between node k and node l symbolizes a local interaction between targets k and l. An example of interaction graph corresponding to the configuration described by Figure 7.1 is illustrated in Figure 7.2.

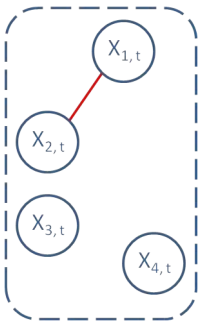


Figure 7.2: Example of an interaction graph capturing the structure of interactions of the configuration shown in Figure 7.1. Here, targets 1 and 2 are interacting together and their behaviors at the next time step will depend on both their states. On the other hand, targets 3 and 4 are not in interaction and will behave independently in the next time step.

The generation of the interaction graph assumes the availability of a context-based function Φ modeling the prior knowledge of the application domain, especially the interaction neighborhood of a given target. Basically, Φ should be able to detect whether two targets are interacting (they are in the interaction neighborhood of each other) or not given their respective state values. A common example of such a context-based function is the one considering that two targets are interacting if their physical distance is below a given threshold. In general, the function Φ is defined as

$$\Phi(\mathbf{x}_{k,t}, \mathbf{x}_{l,t}) = \begin{cases} 1 & \text{if } \mathbf{x}_{k,t} \text{ and } \mathbf{x}_{l,t} \text{ satisfy the condition for} \\ & \text{being in interaction,} \\ 0 & \text{otherwise.} \end{cases} \tag{7.10}$$

As targets are moving, the structure of the interaction graph is not static but evolves over time. Inspired by the work of Khan et al. [2004], we propose to build on the fly the interaction graph at each time step using the function Φ.

From now on, we assume that we are provided with the current interaction graph of the targets within the environment (the computation of such a graph will be discussed later in Section 7.4.3). As mentioned above, we are interested in computing the predicted distribu-

tion $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ of the k^{th} target in the environment and therefore, the interactive likelihood transition $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1})$ as suggested by Equation 7.7. Let $\mathcal{N}(k)$ be the set of targets in the neighborhood of target k within the interaction graph. Also, let $\mathbf{x}_{\mathcal{N}(k),t-1} = \{\mathbf{x}_{l,t-1}\}_{l:l\in\mathcal{N}(k)}$ be the collection of the states of all the targets belonging in $\mathcal{N}(k)$. Under the assumptions made in Section 7.3, the state $\mathbf{x}_{k,t}$ of target k at time step t depends only on its state $\mathbf{x}_{k,t-1}$ at time step $t-1$ as well as $\mathbf{x}_{\mathcal{N}(k),t-1}$. Therefore, Equation 7.3 can be rewritten as

$$\mathbf{x}_{k,t} = F^{k}_{t,|\mathcal{N}(k)|+1}(\mathbf{x}_{k,t-1}, \mathbf{x}_{\mathcal{N}(k),t-1}, \mathbf{w}_{k,t}). \tag{7.11}$$

Equation 7.11 stipulates that, in order to correctly determine the future behavior of target k, it is necessary to simulate only targets belonging to $\mathcal{N}(k)$ instead of simulating all the targets as advocated by Equation 7.3. In terms of probability, this translates into

$$p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1}) = p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\mathcal{N}(k),t-1}). \tag{7.12}$$

On this basis, the expression of $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1})$ in Equation 7.8 is simplified as

$$Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{\mathcal{N}(k),t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\mathcal{N}(k),t-1}) p(\mathbf{x}_{\mathcal{N}(k),t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{\mathcal{N}(k),t-1}, \tag{7.13}$$

where $p(\mathbf{x}_{\mathcal{N}(k),t-1}|\mathbf{z}_{1:t-1})$ is defined as

$$p(\mathbf{x}_{\mathcal{N}(k),t-1}|\mathbf{z}_{1:t-1}) = \prod_{l:l\in\mathcal{N}(k)} p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1}). \tag{7.14}$$

Of course, the complexity of Equation 7.13 is lower than the one of Equation 7.8. Moreover, when targets do not interact at all, $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$ automatically degrades to $p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1})$ and thereby, the formulation in Equation 7.7 corresponds to multiple independent Bayesian filters.

While the formulation in Equation 7.13 sounds very interesting, its computational complexity may still be expensive depending on the state space's dimension of $\mathbf{x}_{\mathcal{N}(k),t-1}$. Besides, the computation of the interaction graph as well as the neighborhood $\mathcal{N}(k)$ of target k may not be a priori straightforward. Indeed, if we were provided with the collection $\{\mathbf{x}_{k,t-1}\}_{k=1}^{K}$ of the states of all the targets in the environment, a simple application of the function Φ on all pairs of states from the collection will lead to the construction of the graph and the computation of the neighborhood of each target. Instead, we are provided with posterior distributions $p(\mathbf{x}_{\cdot,t-1}|\mathbf{x}_{\cdot,t-1})$ characterizing each target. The next section is dedicated to the estimation of $\mathcal{N}(k)$ based on these individual posterior distributions as well as the reduction of the computational complexity without highly degrading the quality of the corresponding estimate.

7.4.3 Probability distribution aggregation

Recalling from Section 7.4, our objective is to estimate the predicted distribution of a given target while taking into account its potential interaction with other targets in the environment. In the previous section, we derived an equation for computing the interactive likelihood transition $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$ of the k^{th} target at time step t given its state $\mathbf{x}_{k,t-1}$ at time step $t-1$ and the historic $\mathbf{z}_{1:t-1}$ of observation data (see Equation 7.13). In this section, we are interested in reducing the complexity of this equation while estimating the subset $\mathcal{N}(k)$ of targets with which target k interacts. To this end, we propose to make an approximation by discretizing

the probability distribution characterizing each target l ($l \neq k$) in such a way to not highly degrade the quality of the corresponding estimate. The general idea governing such a procedure is presented below.

7.4.3.1 Overview

For illustration purposes, we consider the case in which there are two targets, let us say k and l, in the environment. We assume that the state $\mathbf{x}_{k,t-1}$ of target k is known. The approach we propose in this section is based on the (interaction) effects of target l on the future behavior of target k. Basically, given $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$, it may happen that two state values $x^p_{l,t-1}$ and $x^q_{l,t-1}$ from this distribution have quite the same effect on $\mathbf{x}_{k,t-1}$, that is,

$$\bar{\mathbf{x}}^p_{k,t} \approx \bar{\mathbf{x}}^q_{k,t},$$

where $\bar{\mathbf{x}}^p_{k,t}$ (resp. $\bar{\mathbf{x}}^q_{k,t}$) is the state of target k at time step t under the assumption that the true state $\mathbf{x}_{l,t-1}$ of target l at time step $t-1$ is equal to $x^p_{l,t-1}$ (resp. $x^q_{l,t-1}$). In such conditions, for a good estimation of the probability distribution regarding the state of target k at time step t, it is useless to simulate target k with both values $x^p_{l,t-1}$ and $x^q_{l,t-1}$ of target l. Only a simulation with one of these values is necessary provided that the probability of the simulation's outcome is adapted accordingly (by integrating the probability values associated with $x^p_{l,t-1}$ and $x^q_{l,t-1}$ from the distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$).

To generalize this idea for estimating the future behavior of target k, one can proceed according to the following two steps:

- **Partitioning** (Section 7.4.3.2): We partition the state space represented by the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ into regions where each region regroups state values having the same effect on the state of target k. For each resulting region, a representative state is computed (on the basis of all the state values belonging to the region) and associated with a weight corresponding to the probability mass of the region. We use the term of *probability distribution aggregation* to refer to this operation. For illustration, Figure 7.3 depicts an example of the partitioning of a probability distribution based on the related effect on a given reference point.

- **Estimation** (Section 7.4.3.3): The future behavior of target k is approximated using the resulting representative states instead of the whole distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$, thus reducing the complexity of the estimation procedure.

The following sections are dedicated to each of these steps.

7.4.3.2 Effect-based partitioning

In general, the effect of an object on another one is highly correlated with their relative distance. Illustrative examples where this can be observed include electrostatic and/or electromagnetic forces [Faraday, 1832]. Under this assumption, given the state $\mathbf{x}_{k,t-1}$ of target k, it is possible to define a *"distance function"* $dist(\cdot, \mathbf{x}_{k,t-1})$ for characterizing the effect of target l on target k. In such conditions, the partitioning of the distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ can be done according to the

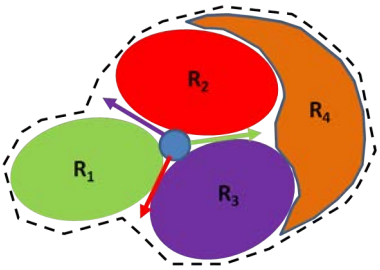


Figure 7.3: Example of effect-based probability distribution partitioning. The blue circle represents target k ($\mathbf{x}_{k,t-1}$) while the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ of target l is represented by region delimited by the dashed black line. $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ is partitioned into four regions R_1, R_2, R_3 and R_4. In this scenario, values of $\mathbf{x}_{l,t-1}$, the state of target l, belonging to the first three regions should normally cause target k to move in the direction indicated by the arrow with the corresponding region's color. On the other hand, values belonging to R_4 do not have any impact (no interaction) on target k.

following rule: two values $x^p_{l,t-1}$ and $x^q_{l,t-1}$ from $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ belong to the same region if the difference of their relative distance to $\mathbf{x}_{k,t-1}$ — that is $|dist(x^p_{l,t-1}, \mathbf{x}_{k,t-1}) - dist(x^q_{l,t-1}, \mathbf{x}_{k,t-1})|$ — is below a predefined threshold.

However, this approach suggests that, for estimating the predicted distribution of target k, a new partitioning procedure has to be performed for each value of $\mathbf{x}_{k,t-1}$ from the posterior distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$. Additionally, this procedure should be repeated for every target in the environment. Therefore, the approach is computationally expensive in practice.

An alternative approach[15] for avoiding such a computational exigence consists in approximating the effect of target l by partitioning $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ independently of $\mathbf{x}_{k,t-1}$. Basically, instead of using the distance function $dist(\cdot, \mathbf{x}_{k,t-1})$, one can rely on an *"affinity function"* $F_{aff}(\cdot,\cdot)$ defined on $\mathbf{x}_{l,t-1}$ to perform the partitioning. More specifically, given two state values $x^p_{l,t-1}$ and $x^q_{l,t-1}$ from $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$, $F_{aff}(x^p_{l,t-1}, x^p_{l,t-1})$ measures how "*close*" they are with respect to the application domain. Specifically, the closer the points, the higher their affinity. The underlying idea behind this approach is that, if $x^p_{l,t-1}$ and $x^q_{l,t-1}$ are relatively close to each other, then their relative distances to any value of $\mathbf{x}_{k,t-1}$ are "nearly" the same. Therefore, they should belong to the same region since their effects on target k are similar.

In Section 7.4.4, we present two methodologies (heuristics) for partitioning $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ based on this idea and thereafter computing the resulting representative states as well as the corresponding weights. For now, we focus on computing the estimation of the predicted distribution of target k based on these representative states.

7.4.3.3 Estimation of the predicted distribution

In this section, we assume that we are provided with the weighted representative states resulting from the posterior distributions of all the targets in the environment at time step $t-1$. Let

[15] Other techniques probably exist in the literature to break down this computational complexity, however they are not addressed here.

$R_{l,t-1}$ be the set of all the weighted representative states characterizing target l. We have

$$R_{l,t-1} = \{(\hat{\mathbf{x}}^r_{l,t-1}, \hat{w}^r_{l,t-1})\}_{r=1}^{|R_{l,t-1}|},$$

where $\hat{\mathbf{x}}^r_{l,t-1}$ is the r^{th} representative state and $\hat{w}^r_{l,t-1}$ is its associated weight. In what follows, we first derive the estimation of the predicted distribution of target k in the special case where all the targets in the environment are characterized by a unique representative state. After that, we proceed with the estimation in the general case of multiple representative states.

Single representative per target

A baseline solution for obtaining a set of representative state values consists in aggregating the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ into a unique representative state $\hat{\mathbf{x}}_{l,t-1}$ with a weight $\hat{w}_{l,t-1} = 1$ (here, we omit the superscript r for simplicity since we have only one representative). Several heuristics exist for computing such a representative state including for instance the mode and/or the expected mean of the distribution.

As mentioned in Section 7.4.2, to compute the interactive likelihood transition $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$, we need to deduce $\bar{\mathcal{N}}(k)$, an estimate of the neighborhood $\mathcal{N}(k)$ of target k. To this end, an interaction graph is built on the basis of the collection $\{\hat{\mathbf{x}}_{l,t-1}\}_{l=1}^{K}$ in conjunction with the function Φ as follows: for each pair of representative states, Φ is used to determine if there is an edge between the nodes associated with corresponding targets. $\bar{\mathcal{N}}(k)$ is then composed of targets such that there is an edge between them and target k in the generated graph. Thereafter, $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1})$ is approximated as

$$Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) = p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \hat{\mathbf{x}}_{\bar{\mathcal{N}}(k),t-1})p(\hat{\mathbf{x}}_{\bar{\mathcal{N}}(k),t-1}, \mathbf{z}_{1:t-1}), \tag{7.15}$$

where $\hat{\mathbf{x}}_{\bar{\mathcal{N}}(k),t-1} = \{\hat{\mathbf{x}}_{l,t-1}\}_{l:l\in\bar{\mathcal{N}}(k)}$ is the collection of the representative states of all the targets in $\bar{\mathcal{N}}(k)$ and $p(\hat{\mathbf{x}}_{\bar{\mathcal{N}}(k),t-1}, \mathbf{z}_{1:t-1}) = \prod_{l:l\in\bar{\mathcal{N}}(k)} p(\hat{\mathbf{x}}_{l,t-1}, \mathbf{z}_{1:t-1})$. As the posterior distribution of each target l has been aggregated into a single representative state $\hat{\mathbf{x}}_{l,t-1}$, we have

$$p(\hat{\mathbf{x}}_{l,t-1}, \mathbf{z}_{1:t-1}) = 1, \text{ for } l = 1, \cdots, K.$$

Therefore, Equation 7.15 can be rewritten as

$$Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) = p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \hat{\mathbf{x}}_{\bar{\mathcal{N}}(k),t-1}). \tag{7.16}$$

Finally, by substituting $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$ with Equation 7.16 in Equation 7.7, the predicted distribution of target k is obtained as

$$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{k,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \hat{\mathbf{x}}_{\bar{\mathcal{N}}(k),t-1})p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{k,t-1}. \tag{7.17}$$

As it can be shown, using Equation 7.17, the complexity of each individual target filter is similar to the one of a classical filter when the target behaves independently. However, while using a single representative state value may be sufficient for characterizing compact probability distributions, it may not be good enough for probability densities with multiple modalities nor probability densities over state spaces embedding categorical variables (e.g., the activity). In the next paragraph, we present a formulation involving multiple representative states.

Multiple representatives per target

In this paragraph, we explore the general case in which the posterior distribution of a target is aggregated into several representative states, each one with an associated weight. We use the expression *representative state's combination* to refer to an element from the space $R_{1,t-1} \times \cdots \times R_{K,t-1}$. In other words, a representative state's combination $\bar{\mathcal{C}}$ is composed of K representative states such that each one is related to a distinct target. Also, the *weight $\bar{w}$ of a representative state's combination $\bar{\mathcal{C}}$* is defined as the product of the weights of all the representative states composing the combination. That is,

$$\bar{w} = \prod_{(l,r):\hat{\mathbf{x}}^r_{l,t-1} \in \bar{\mathcal{C}}} \hat{w}^r_{l,t-1}. \tag{7.18}$$

Let $\mathcal{C} = \{(\bar{\mathcal{C}}_c, \bar{w}_c)\}$ be the set of all representative state's combinations with their associated weights. The cardinality of $\mathcal{C}$ is then given by

$$|\mathcal{C}| = \prod_{l=1}^{K} |R_{l,t-1}|. \tag{7.19}$$

Theoretically, as it was previously the case with the single-representative-based formulation, each element $\bar{\mathcal{C}}_c$ of $\mathcal{C}$ can be used in conjunction with the function Φ for computing an interaction graph $\bar{G}_c$ on the basis of the underlying representative states. Let $G = \{(\bar{G}_c, \bar{w}_c)\}_{c=1}^{|\mathcal{C}|}$ be the set of all the generated interaction graphs, each one being associated with the weight of the representative state's combination from which it is derived. Considering an interaction graph $\bar{G}_c$, it is possible to retrieve the neighborhood $\bar{\mathcal{N}}_c(k)$ of the representative state of target k involved in that graph. Similarly, considering a specific representative state $\hat{\mathbf{x}}^r_{k,t-1}$ of target k, it is possible to compute the set $IG(k,r)$ of combination (or graph) indices in which $\hat{\mathbf{x}}^r_{k,t-1}$ is involved. We have

$$|IG(k,r)| = \prod_{l \in \{1,\cdots,K\}: l \neq k} |R_{l,t-1}|.$$

As assumed, the probability distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ has been partitioned into $|R_{k,t-1}|$ regions from which a set of weighted representative states has been generated. Therefore, considering a value $x_{k,t-1}$ drawn from the distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$, — that is $p(x_{k,t-1}|\mathbf{z}_{1:t-1}) \neq 0$ — it can deterministically be associated with the representative state of the region which it belongs to. Let $\Upsilon_k(.)$ be the function that associates any such value of $\mathbf{x}_{k,t-1}$ to the index of its corresponding representative state. Then, the interactive likelihood transition $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$ is approximated as

$$\begin{aligned} Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) &= \int_{\mathbf{x}_{\mathcal{N}(k),t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\mathcal{N}(k),t-1}) p(\mathbf{x}_{\mathcal{N}(k),t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{\mathcal{N}(k),t-1}, \\ &\approx \sum_{c \in IG(k, \Upsilon_k(\mathbf{x}_{k,t-1}))} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k),t-1}) p(\hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k)|\mathbf{z}_{1:t-1}}), \end{aligned} \tag{7.20}$$

where $\hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k),t-1} = \{\hat{\mathbf{x}}^r_{l,t-1}\}_{(l,r):(l,r) \in \bar{\mathcal{N}}_c(k)}$ represents the joint (representative) state of the neigh-

borhood of target k in the interaction graph $\bar{G}_c$, and $p(\hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k)|\mathbf{z}_{1:t-1}})$ is obtained as

$$\begin{aligned} p(\hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k)|\mathbf{z}_{1:t-1}}) &= \prod_{(l,r):(l,r)\in\bar{\mathcal{N}}_c(k)} p(\hat{\mathbf{x}}^r_{l,t-1}|\mathbf{z}_{1:t-1}), \\ &= \frac{\bar{w}_c}{\hat{w}^{r_k}_{k,t-1}}, \end{aligned} \tag{7.21}$$

where r_k is the index of the representative of target k within the the representative combination $\bar{\mathcal{C}}_c$. Using $\mathbf{x}_{k,t-1}$ in conjunction with the function $\Upsilon_k(\cdot)$, it turns out that

$$r_k = \Upsilon_k(\mathbf{x}_{k,t-1}),$$

and therefore Equation 7.20 can be rewritten as

$$Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1},\mathbf{z}_{1:t-1}) = \sum_{c\in IG(k,\Upsilon_k(\mathbf{x}_{k,t-1}))} \frac{\bar{w}_c}{\hat{w}^{\Upsilon_k(\mathbf{x}_{k,t-1})}_{k,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1},\hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k),t-1}). \tag{7.22}$$

The idea behind Equation 7.22 is to go through all the generated interaction graphs $\{\bar{G}_c\}$ involving the representative state associated to $\mathbf{x}_{k,t-1}$ and exploit the local interaction information obtained from these graphs for computing the transition likelihood. Finally, Equation 7.22 can be used in Equation 7.7 for estimating the predicted distribution of target k, thus leading to:

$$\begin{aligned} p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) \quad = \quad & \int_{\mathbf{x}_{k,t-1}} \left[\sum_{c\in IG(k,\Upsilon_k(\mathbf{x}_{k,t-1}))} \frac{\bar{w}_c}{\hat{w}^{\Upsilon_k(\mathbf{x}_{k,t-1})}_{k,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1},\hat{\mathbf{x}}_{\bar{\mathcal{N}}_c(k),t-1})\right] \\ & p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})d\mathbf{x}_{k,t-1}, \end{aligned} \tag{7.23}$$

As it can be noticed, Equation 7.22 is equivalent to Equation 7.16 when the number of representatives for each target is equal to 1. Regarding the complexity, the computational cost highlighted in Equation 7.8 is drastically reduced in Equation 7.22 provided that the number of representatives generated for each target distribution is not too large. Nevertheless, the complexity induced by Equation 7.22 can still further be reduced. The following paragraph describes how to do it.

Computational complexity reduction

The formulation described in Equation 7.22 can further be optimized at computation time. Let us consider the following example in which we are interested in tracking three targets A, B and C. Let us assume that, at time step $t-1$, target A is characterized by exactly one representative $\hat{A}_1$ while both target B and target C are characterized by two representatives ($\hat{B}_1$ and $\hat{B}_2$ for target B; and $\hat{C}_1$ and $\hat{C}_2$ for target C). Let us assume that $\hat{B}_1$ has a weight of $2/3$ and $\hat{B}_2$ a weight of $1/3$. Similarly, $\hat{C}_1$ has a weight of $2/5$ and $\hat{C}_2$ a weight of $3/5$. Finally, let us assume that the interaction graphs generated from these representatives appear as depicted in Figure 7.4.

According to Figure 7.4, target A is not interacting with either target B or target C. Therefore, it should be updated as if it were alone in the environment using classical Bayesian filtering formulas. Nevertheless, following the formula of Equation 7.22, the interactive likelihood transition

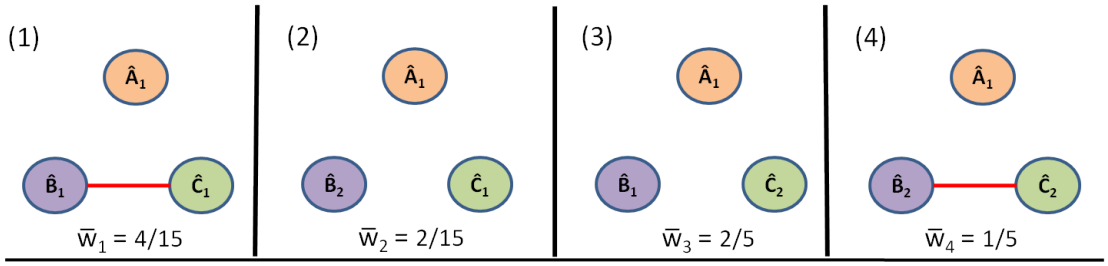


Figure 7.4: Example of a multiple representative aggregation based approach.

of target A is obtained as

$$Q(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}, \mathbf{z}_{1:t-1}) = \frac{4}{15}p(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}) + \frac{2}{15}p(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}) + \frac{2}{5}p(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}) + \frac{1}{5}p(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}).$$

Instead of explicitly computing $p(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1})$ sequentially for all the interaction graphs, one can first, based on the considered target representative state (here $\hat{A}_1$), regroup the interaction graphs into categories where each category encapsulates graphs exhibiting the same neighborhood for the considered representative state. In this way, the computation of the interactive likelihood transition is factorized with the advantage of less computational effort. Therefore, in our example, we obtain

$$Q(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}, \mathbf{z}_{1:t-1}) = \left[\frac{4}{15} + \frac{2}{15} + \frac{2}{5} + \frac{1}{5}\right] p(\mathbf{x}_{A,t}|\mathbf{x}_{A,t-1}).$$

Generalizing this idea, let $\Delta_{k,r} = \{\Delta^g_{k,r}, \upsilon^g_{k,r}\}_{g=1}^{\Xi_{k,r}}$ be the set of different groups of interaction graphs involving the r^{th} representative $\hat{\mathbf{x}}^r_{k,t-1}$ of target k such that graphs belonging to a given group $\Delta^g_{k,r}$ exhibit the same neighborhood $\tilde{\mathcal{N}}_g(k,r)$ for $\hat{\mathbf{x}}^r_{k,t-1}$. $\Xi_{k,r}$ represents the numbers of groups and $\upsilon^g_{k,r}$ represents the weight of the g^{th} group $\Delta^g_{k,r}$. We have

$$IG(k,r) = \bigcup_{g=1}^{\Xi_{k,r}} \Delta^g_{k,r}.$$

The weight $\upsilon^g_{k,r}$ of the group $\Delta^g_{k,r}$ is defined as the sum of the weights of all the interaction graphs belonging to the group at hand. We thus have

$$\upsilon^g_{k,r} = \sum_{c \in \Delta^g_{k,r}} \bar{w}_c. \tag{7.24}$$

On this basis, Equation 7.22 can then be rewritten as

$$Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) = \sum_{g=1}^{\Xi_{k,\Upsilon_k(\mathbf{x}_{k,t-1})}} \frac{\upsilon^g_{k,\Upsilon_k(\mathbf{x}_{k,t-1})}}{\hat{w}^{\Upsilon_k(\mathbf{x}_{k,t-1})}_{k,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,\Upsilon_k(\mathbf{x}_{k,t-1})),t-1}), \tag{7.25}$$

where $\hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,\Upsilon_k(\mathbf{x}_{k,t-1})),t-1} = \{\hat{\mathbf{x}}^r_{l,t-1}\}_{(l,r):(l,r)\in\tilde{\mathcal{N}}_g(k,\Upsilon_k(\mathbf{x}_{k,t-1}))}$. Here we conclude our description of the estimation of the predicted distribution of a given target k in situations in which there are multiple representatives per target in the environment. The next section is dedicated to the

presentation of some heuristics for effectively aggregating probability distributions into representative states.

7.4.4 Heuristics for aggregating probability distributions

In this section, we are interested in aggregating the posterior distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ of a given target l at time $t-1$. In other words, we seek to (1) partition the distribution into regions and (2) compute for each region the corresponding representative state together with its associated weight. Without loss of generality, we assume that $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ is represented by a set of weighted points $\{\mathbf{x}^i_{l,t-1}, w^i_{l,t-1}\}_{i=1}^N$ in the state space where N is the number of points. In the case $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ is not represented by a set of weighted points, the latter can still be obtained by executing the following procedure:

- sampling a large number of points from the distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$; because of the law of large numbers, the generated points approximate the original probability distribution;
- considering the set of sampled points and associating each element from this set with a weight corresponding to its frequency within the list of generated points.

As previously stated in Section 7.4.3.2, we rely on a affinity function F_{aff} for partitioning $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ independently of other targets present in the environment. After formally defining the characteristics of such an affinity function (Section 7.4.4.1), we describe two heuristics for partitioning $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ into non-overlapping regions (Section 7.4.4.2). Finally, we focus on the computation of the representative states corresponding to each of these generated regions (Section 7.4.4.3).

7.4.4.1 Definition of the affinity function

Because the state space may be composed of attributes which are not semantically related, as for example the velocity and the activity of the tracked target, a special care has to be considered when defining F_{aff} in such a way to avoid mixing non-commensurate attributes. To this end, and inspired by the work of Navarro et al. [2011], we propose to regroup the state space attributes into classes of commensurate attributes and to define a distance function for each class. As an illustrative example, let us consider the tracking problem in the subway station introduced in Chapter 5. We can define for instance a class Cl_1 of location attributes (continuous attributes), a class Cl_2 of attributes defining the activity of the target (categorical attributes), and so on ... Then, the distance over Cl_1 can simply be the Euclidean distance between two locations while, for Cl_2, one can define a distance between the different activities (for example, the distance between drinking and eating can be small while the distance between two other distinct activities can be maximal).

Let $\{Cl_b\}_{b=1}^{N_{cl}}$ be the set of N_{cl} attribute classes characterizing the state space of $\mathbf{x}_{l,t-1}$, and let D_{cl_b} be the distance function defined for class Cl_b. The affinity function F_{aff} between two state values $x^{i_1}_{l,t-1}$ and $x^{i_2}_{l,t-1}$ is defined as a combination of the different class distances as

$$F_{aff}(x^{i_1}_{l,t-1}, x^{i_2}_{l,t-1}) = f_{comb}(D_{cl_1}(x^{i_1}_{l,t-1}, x^{i_2}_{l,t-1}), D_{cl_2}(x^{i_1}_{l,t-1}, x^{i_2}_{l,t-1}), \cdots, D_{cl_{N_{cl}}}(x^{i_1}_{l,t-1}, x^{i_2}_{l,t-1})), \tag{7.26}$$

where f_{comb} is a continuous positive function which is strictly decreasing as any of the attribute class's distance D_{cl_b} increases.

In the following section, we discuss techniques for partitioning the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ based on the above defined affinity function.

7.4.4.2 Partitioning of the probability distribution

While data partitioning has been a major concern during the last decades [Jain et al., 1999], we briefly present in this section two partitioning (or clustering) techniques which aim at splitting the set of weighted points representing the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ into non-overlapping regions (here, subsets of points) while relying on the affinity function F_{aff} defined above. The first technique is based on the well known k-means clustering algorithm [MacQueen, 1967] and requires to be provided, in advance, with the number of regions to be generated. On the other hand, the second technique is a density-based clustering approach [Ester et al., 1996] and does not need to be provided with the number of regions to generate.

Centroid-based clustering

This technique is mainly inspired from the k-means algorithm [MacQueen, 1967, Lloyd, 1982] except that, instead of using the means of concerned points for characterizing the centroid of a given region (or cluster), the representative state of the region is used. Basically, assuming we are interested in generating R_l regions, the algorithm proceeds according to the following procedure:

1. Randomly choose R_l points $\{\hat{\mathbf{x}}^r_{l,t-1}\}_{r=0}^{R_l}$ in the state space of $\mathbf{x}_{l,t-1}$ for representing the initial centroids of the regions.
2. Repeat until the convergence criterion is met:
   - assign each point $\mathbf{x}^i_{l,t-1}$ to the region j with the highest affinity with respect to the corresponding centroid, that is
   $$j = \underset{r\in\{1\cdots R_l\}}{\arg\max}\, F_{aff}(\mathbf{x}^i_{l,t-1}, \hat{\mathbf{x}}^r_{l,t-1});$$
   - update the centroid $\hat{\mathbf{x}}^r_{l,t-1}$ of each region r as the representative state of the points belonging in region r.

In the above described algorithm, the convergence is met when the assignments to the different regions no longer change. Once the convergence is achieved, each region is characterized by its centroid which can lately be used as its representative state.

In terms of complexity, each iteration of the clustering process is linear in N (the number of points composing the distribution) and in R_l (the number of regions to generate). If N_c is the number of iterations needed until convergence, then the overall complexity of the clustering process is $\mathcal{O}(N.R_l.N_c)$. Usually, a maximum number of iterations is set and the algorithm stops when this number is reached even if the convergence is not yet met. Because the number of

clusters (and therefore the number of representatives) is set in advance, this algorithm is suitable when we have a hint regarding the shape of the underlying probability distribution. However, it is not always possible to have such an information.

Density-based clustering

In density-based clustering, the definition of cluster is based on the notion of *density reachability*. Basically, two points are *directly* density-reachable if they are not far away, with respect to a predefined distance metric (here, the inverse of the affinity function F_{aff}), than a specified threshold ϵ_A. Besides, two points are density-reachable if there is a path between them in such a way that two successive points within the path are directly density reachable. The most common density-based clustering algorithm is DB-SCAN [Ester et al., 1996]. However, DB-SCAN may treat some points as noise (when the cardinality of their neighborhood is below a preset number), therefore preventing them to be part of the resulting clusters.

We now present a density-based algorithm in which all the provided points are part of a cluster at the end of the clustering process. This algorithm is inspired by the *clearing procedure* introduced in [Petrowski, 1996] as a niching methodology in the context of evolutionary algorithms [Mahfoud, 1995]. The algorithm, which is applied on the set of weighted points representing the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$, seeks to gather in a same cluster points which are density-reachable by a path whose length is at most equal to a specified parameter $MaxSize$. To this end, it proceeds according to the following procedure:

1. Sort all the points in descending order according to their weights and mark them all as "unprocessed". The aim here is to give more importance to most probable points.
2. Repeat until all the points are "processed":
    - consider the heaviest unprocessed point to form a new cluster;
    - repeat $MaxSize$ times:
        - consider the set A of the newly added points in the current cluster and mark them as processed;
        - add the set of all the neighbors (unprocessed points which are directly density-reachable from the considered point) for all the points in A in the current cluster;

A pseudo-code of the described procedure is presented in Algorithm 7.1.

The complexity of the above described clustering procedure is $\mathcal{O}(N \ln N)$. Unlike centroid-based clustering, the number of clusters is not known beforehand and therefore, the algorithm is suitable for situations in which the shape of the distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ varies over time.One particularity of this approach is the ability to adaptively modify the value of the threshold ϵ_A at run-time (and so obtaining coarse or fine aggregations) in such a way to cope with application constraints such as computational resources and/or time; thus making it practical for real-time applications.Alternatively, it is possible to apply on the output of this approach an agglomerate hierarchical clustering algorithm [Sibson, 1973] for obtaining coarser clusters.

Algorithm 7.1: Clearing-Based Clustering

Algorithm `Clearing-Clustering`$(\{\mathbf{x}^i_{l,t-1}, w^i_{l,t-1}\}_{i=1}^N,\ \epsilon_A,\ MaxSize)$

- $processedPts = \emptyset$
- $sortWeightedPoints(\{\mathbf{x}^i_{l,t-1}, w^i_{l,t-1}\}_{i=1}^N)$
- **for** $i = 1, \cdots, N$ **do**
  - $clusters[i] = -1$
- $clusterId = 0$
- **for** $i = 1, \cdots, N$ **do**
  - **if** $(i \notin processedPts)$ **then**
    - $clusterId = clusterId + 1$
    - $clusters[i] = clusterId$
    - $processedPts = processedPts \cup \{i\}$
    - $directlyReachablePts =$ `EpsNeighborhood` $(\{\mathbf{x}^i_{l,t-1}, w^i_{l,t-1}\}_{i=1}^N,\ i,\ processedPts,\ \epsilon_A)$
    - **for** $k = 1, \cdots, MaxSize$ **do**
      - **for** $(j \in directlyReachablePts)$ **do**
        - $clusters[j] = clusterId$
        - $processedPts = processedPts \cup \{j\}$
      - **if** $(k < MaxSize)$ **then**
        - $nextReachablePts = \emptyset$
        - **for** $(j \in directlyReachablePts)$ **do**
          - $tempPts =$ `EpsNeighborhood` $(\{\mathbf{x}^i_{l,t-1}, w^i_{l,t-1}\}_{i=1}^N,\ j,\ processedPts,\ \epsilon_A)$
          - $nextReachablePts = nextReachablePts \cup tempPts$
        - $directlyReachablePts = nextReachablePts$

Procedure `EpsNeighborhood`$(\{\mathbf{x}^i_{l,t-1}, w^i_{l,t-1}\}_{i=1}^N,\ i,\ processedPts,\ \epsilon_A)$

- $neighborhood = \emptyset$
- **for** $j = 1, \cdots, N$ **do**
  - **if** $j \notin processedPts$ *and* $\left(\frac{1}{F_{aff}(\mathbf{x}^i_{l,t-1}, \mathbf{x}^j_{l,t-1})} < \epsilon_A\right)$ **then**
    - $neighborhood = neighborhood \cup \{j\}$
- **return** $neighborhood$

7.4.4.3 Computing the representative states

Once the probability distribution $p(\mathbf{x}_{l,t-1}|\mathbf{z}_{1:t-1})$ has been partitioned into R_l non-overlapping regions (i.e. subsets of points), we need to compute the representative state for each region.

Considering the r^{th} region, its representative state $\hat{\mathbf{x}}^r_{l,t-1}$ can be obtained by combining the attributes of the points belonging to the region using a group of predefined operators such as SUM, MEDIAN, MODE or MEAN according to the semantic of the considered attribute and/or its nature (e.g., continuous real value, categorical). For example, for location attributes, we can use the MEAN operator while, for the activity attribute, which is a categorical attribute, we can use the MODE operator. The choice of the operators is application-dependent. Similarly, the weight $\hat{w}^r_{l,t-1}$ of the representative $\hat{\mathbf{x}}^r_{l,t-1}$ is obtained by summing up the weights of the points belonging to the corresponding region.

7.4.5 Summary

To sum up, from Section 7.4.1 to Section 7.4.4, we have (1) proposed a factored Bayesian approach for estimating the predicted distribution of individual targets involved in dynamics-based interactions, and (2) described useful heuristics for reducing the computational complexity of such an estimation. In what follows, we address the implementation of the proposed approach using the particle filtering paradigm.

7.5 Implementation Using Particle Filtering

In the previous section, we have designed a factored approach for managing dynamics-based interactions during the prediction step of a filtering process, that is, estimating the predicted distribution of each target in the environment on an individual basis. In this section, we seek to integrate the designed approach within a complete filtering process. Consequently, we need to choose a methodology for solving the data association issues. As mentioned in Section 7.1, we consider the use of the JPDA framework (see Section 6.3.3) for such a purpose. More precisely, we consider the general case in which the target dynamics are non-linear and therefore, we rely on the Monte-Carlo JPDA filter (MC-JPDAF) [Schulz et al., 2003, Vermaak et al., 2005] which is based on the particle filtering paradigm [Arulampalam et al., 2002]. However, (MC-)JPDAF presents two limitations with respect to the problem we are interested in (see Section 7.2):

- **Handling of non-covered areas:** while we are concerned with cluttered environments which may potentially include areas which are not under sensory coverage, the JPDAF algorithm, in its initial derivation, has been designed to deal with environments exempt from non-covered areas. We propose (Section 7.5.2) an extension to the MC-JPDAF so as to cope with this limitation.

- **Management of targets' interactions:** the (MC-)JPDAF algorithm assumes a total independence among target dynamics in its implementation. To tackle this limitation (Section 7.5.3), we integrate the factored approach designed in the previous section within the prediction step of the algorithm, thus leading to a new algorithm that we name *IT-MCJPDAF* where the prefix "IT" stands for "interacting targets".

The remainder of this section is organized as follows. Section 7.5.1 introduces the classical MC-JPDAF algorithm while Section 7.5.2 presents our extension of the (MC-)JPDAF filter to cope with environments including non-covered areas. Finally, Section 7.5.3 discusses the implementation details of the *IT-MCJPDAF* algorithm together with the encountered issues.

7.5.1 Monte Carlo JPDAF

Unlike the classical JPDAF algorithm, where the distributions are computed using the Kalman filtering techniques [Kalman, 1960], MC-JPDAF uses common particle filtering techniques for approximating the posterior density function $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ of the target states given the received observation data. In this algorithm, it is assumed that targets behave independently. Like any filter derived from the Bayesian perspective (see Section 2.2), MC-JPDAF proceeds according to two phases, the prediction and the correction steps, which respectively aims at computing the predicted and the posterior distributions of the targets being tracked. Like for JPDAF, the correction step of MC-JPDAF, at each time step t, consists in

- resolving the data association problem,
- and computing the target marginal probability distribution which, in this case, is represented by a set of N weighted points (also called particles) $\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}^N_{i=1}$ for a given target k.

The resolution of the data association problem requires the algorithm to determine the origin of each received atomic observation. To this end, a "soft-gating" technique has been introduced [Daronkolaei et al., 2007] as an adaptation of the original gating heuristic (based on Gaussian densities, see Section 6.3.2.1) to cope with non-Gaussian densities. In the remainder of this section, we first introduce the soft-gating technique and, then, present the MC-JPDAF algorithm.

7.5.1.1 Soft-gating procedure

As already seen in Section 6.3.2.1, the purpose of the gating heuristic is to reduce the complexity of the procedure which consists in determining the origins of the received atomic measurements. It suggests that an atomic observation can be considered to come from a given target if it lies within a validation region — or *gate* — around the target. In what follows, we present the soft-gating technique within the particle filtering paradigm. In this section, it is assumed that the sensor observation model $h(\mathbf{x}_{\cdot,t}, \mathbf{v}_{\cdot,t})$ can be put in the form $h(\mathbf{x}_{\cdot,t}) + \mathbf{v}_{\cdot,t}$, where $\mathbf{v}_{\cdot,t}$ is characterized by a zero-mean Gaussian density with a covariance matrix $\mathbf{R}_t$ ($\mathcal{N}(0, \mathbf{R}_t)$).

Let us consider a given target k and let us assume that its *predicted* distribution at time t is represented by a set of N weighted particles $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}$. Then, the validation gate around target k is approximated as follows [Daronkolaei et al., 2007]:

- compute the mean $\mu^h_{k,t|t-1}$ of the atomic observations resulting from the particles as

$$\mu^h_{k,t|t-1} = \sum_{i=1}^{N} w^i_{k,t|t-1} h(\mathbf{x}^i_{k,t|t-1}); \tag{7.27}$$

- compute the covariance matrix $\mathbf{C}^{\nu}_{k,t}$ of the predicted atomic observation $\mu^{h}_{k,t|t-1}$ as

$$\mathbf{C}^{\nu}_{k,t} = \mathbf{R}_t + \sum_{i=1}^{N} w^{i}_{k,t|t-1}(h(\mathbf{x}^{i}_{k,t|t-1}) - \mu^{h}_{k,t|t-1})(h(\mathbf{x}^{i}_{k,t|t-1}) - \mu^{h}_{k,t|t-1})^{T}. \tag{7.28}$$

Finally, a given atomic observation $\mathbf{z}_{j,t}$ lies within the validation area of target k if and only if

$$(\mathbf{z}_{j,t} - \mu^{h}_{k,t|t-1})^{T} {\mathbf{C}^{\nu}_{k,t}}^{-1} (\mathbf{z}_{j,t} - \mu^{h}_{k,t|t-1}) \leq \eta, \tag{7.29}$$

where η is a preset threshold.

As it can be noticed, the overall idea behind the soft-gating technique is to "cast" the predictive likelihood of the considered target with respect to the observation model into a unique Gaussian density distribution $\mathcal{N}(\mu^{h}_{k,t|t-1}, \mathbf{C}^{\nu}_{k,t})$ from which Mahalanobis distances[16] [Mahalanobis, 1936] can be easily computed.

7.5.1.2 MC-JPDAF algorithm

Recalling from Chapter 6 (Section 6.3.3), the JPDA paradigm solves the data association problem by computing the likelihood of the received observation data $\mathbf{z}_t = \{\mathbf{z}_{1,t}, \cdots, \mathbf{z}_{M_t,t}\}$ with respect to a given target k as follows:

$$p(\mathbf{z}_t|\mathbf{x}_{k,t}) = \sum_{j=0}^{M_t} \beta_{jk}.p_h(\mathbf{z}_{j,t}|\mathbf{x}_{k,t}), \tag{7.30}$$

where $p_h(\cdot|\cdot)$ represents the observation model of the sensor network (it is assumed in this equation that $p_h(\mathbf{z}_{0,t}|\cdot) = 1$) and β_{jk}, which is the probability that the j^{th} atomic observation is generated from the k^{th} target, is given by

$$\beta_{jk} = \sum_{\theta:(j,k)\in\theta} p(\theta|\mathbf{z}_{1:t}). \tag{7.31}$$

In Equation 7.31, θ stands for a feasible joint association hypothesis (as introduced in Section 6.3.3) and the SUM operator is over all feasible hypotheses in which the j^{th} atomic observation is associated with the k^{th} target. The soft-gating technique presented previously is used for substantially reducing the number of feasible hypotheses and therefore, the complexity of their generation. For more details about the generation of feasible joint association hypotheses, refer to Section 6.3. Given θ, it is possible to determine the identity l_j of the target associated with the j^{th} atomic observation ($j > 0$). We thus have

$$l_j = \begin{cases} k & \text{if } (j,k) \in \theta, \\ 0 & \text{otherwise.} \end{cases} \tag{7.32}$$

[16] The **Mahalanobis distance** is a common approach for measuring a distance between a point and a distribution.

Finally, the posterior distribution $p(\theta|\mathbf{z}_{1:t})$ of a feasible joint association hypothesis θ is expressed as

$$p(\theta|\mathbf{z}_{1:t}) \propto (1-P_D)^{K-N_{DT}} P_D^{N_{DT}} \times \lambda_{FT}^{N_{FT}} \times \left[\prod_{j:l_j \neq 0} p^{l_j}(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1}) \right], \tag{7.33}$$

where N_{FT} and N_{DT} respectively represent the number of false alarms and the number of detected targets induced by θ, P_D is the detection probability of the sensor network, λ_{FT} is the false alarm rate, $j : l_j \neq 0$ denotes the set of atomic observations which are considered not to be false alarms but originate from a given target l_j, and $p^l(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1})$ represents the predictive likelihood of the j^{th} atomic observation using the information from the l^{th} target and is given by

$$p^l(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1}) = \int p_h(\mathbf{z}_{j,t}|\mathbf{x}_{l,t}) p(\mathbf{x}_{l,t}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{l,t}. \tag{7.34}$$

The main difference between the classical JPDAF algorithm and the MC-JPDAF algorithm is that Equation 7.34 is approximated using the particles (or samples) representing the considered target predicted distribution. Let $\{\mathbf{x}^i_{l,t|t-1}, w^i_{l,t|t-1}\}_{i=1}^N$ be a set of weighted particles representing the predicted distribution $p(\mathbf{x}_{l,t}|\mathbf{z}_{1:t-1})$ of target l at time step t. $\mathbf{x}^i_{l,t|t-1}$ is obtained from $\mathbf{x}^i_{l,t-1}$ using the proper target l's proposal density function $q(\mathbf{x}_{l,t}|\mathbf{x}_{l,t-1}, \mathbf{z}_t)$ (usually equal to its dynamics $p(\mathbf{x}_{l,t}|\mathbf{x}_{l,t-1})$ which is assumed to be independent from other targets) and $w^i_{l,t|t-1}$ is obtained from $w^i_{l,t-1}$ according to

$$w^i_{l,t|t-1} \propto w^i_{l,t-1} \frac{p(\mathbf{x}^i_{l,t|t-1}|\mathbf{x}^i_{l,t-1})}{q(\mathbf{x}^i_{l,t|t-1}|\mathbf{x}^i_{l,t-1}, \mathbf{z}_t)}. \tag{7.35}$$

Then, Equation 7.34 is approximated as

$$p^l(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1}) = \sum_{i=1}^{N} w^i_{l,t|t-1} p_h(\mathbf{z}_{j,t}|\mathbf{x}^i_{l,t|t-1}). \tag{7.36}$$

A pseudo-code of a one-step MC-JPDAF algorithm is reported in Algorithm 7.2.

7.5.1.3 Summary

Section 7.5.1 was dedicated to the presentation of MC-JPDAF, a Monte Carlo implementation of the JPDA filter in which targets are considered to behave independently. However, (MC-)JPDAF was not initially designed to deal with environments including areas which are not under sensory coverage. In the next section, we present an extension of (MC-)JPDAF for addressing this limitation.

7.5.2 Dealing with non-covered areas in (MC-)JPDAF

In its initial derivation, JPDAF is designed to cope with cluttered environments in an effective way. It is assumed that the environment is fully under sensory coverage and the sensors are entitled to provide, on one hand, noisy observations generated from tracked targets and, on the

Algorithm 7.2: Monte Carlo JPDAF

Algorithm `MC-JPDAF`$(\{\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}^N_{i=1}\}^K_{k=1}, \mathbf{z}_t)$
  `/* Prediction Step */`
  for $k = 1, \cdots, K$ **do**
    $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}\} =$ `TargetPrediction`$(\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}^N_{i=1}, \mathbf{z}_t)$
  `/* Correction Step */`
  $\Theta =$`AssocHypGen`$(\{\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}\}^K_{k=1}, \mathbf{z}_t)$
  for $k = 1, \cdots, K$ **do**
    $\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}^N_{i=1}\} =$ `TargetCorrection`$(\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}, \mathbf{z}_t, \Theta)$
  return $\{\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}^N_{i=1}\}^K_{k=1}$

Procedure `TargetPrediction`$(\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}^N_{i=1}, \mathbf{z}_t)$
  for $i = 1, \cdots, N$ **do**
    sample $\mathbf{x}^i_{k,t|t-1}$ from $q(.|\mathbf{x}^i_{k,t-1}, \mathbf{z}_t)$
    compute $w^i_{k,t|t-1}$ using Equation 7.35
    for $j = 1, \cdots, M_t$ **do**
      compute $p_h(\mathbf{z}_{j,t}|\mathbf{x}^i_{k,t|t-1})$
  return $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}\}$

Procedure `AssocHypGen`$(\{\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}\}^K_{k=1}, \mathbf{z}_t)$
  for $k = 1, \cdots, K$ **do**
    for $j = 1, \cdots, M_t$ **do**
      compute the predictive likelihood $p^k(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1})$ of $\mathbf{z}_{j,t}$ with respect to target k using Equation 7.36
      use the soft-gating to determine if atomic observation j eventually comes from target k (see Section 7.5.1.1)
  compute the set $\Theta = \{\theta\}$ of feasible joint association hypotheses
  for $\theta \in \Theta$ **do** compute $p(\theta|\mathbf{z}_{1:t})$ using Equation 7.33
  return Θ

Procedure `TargetCorrection`$(\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}, \mathbf{z}_t, \Theta)$
  `/* Re-weighting */`
  for $j = 0, \cdots, M_t$ **do**
    compute β_{jk} using 7.31
  for $i = 1, \cdots, N$ **do**
    compute $p(\mathbf{z}_t|\mathbf{x}^i_{k,t|t-1})$ using Equation 7.30
    compute the importance weight $w^i_{k,t} = w^i_{k,t|t-1} \times p(\mathbf{z}_t|\mathbf{x}^i_{k,t|t-1})$
  normalize the importance weights: $w^i_{k,t} = \frac{w^i_{k,t}}{\sum_{i_1=1}^N w^{i_1}_{k,t}}$
  `/* Re-sampling */`
  compute the effective size $\widehat{N}^k_{eff} = \frac{1}{\sum_{i=1}^N (w^i_{k,t})^2}$
  if $\widehat{N}^k_{eff} < N_T$ **then**
    $[\{\mathbf{x}^{i_1}_{k,t}, w^{i_1}_{k,t}\}^N_{i_1=1}] =$ RESAMPLE $[\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t}\}^N_{i=1}]$
  else
    for $i = 1, \cdots, N$ **do** assign $\mathbf{x}^i_{k,t} = \mathbf{x}^i_{k,t|t-1}$
  return $\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}^N_{i=1}\}$

other hand, clutters which mainly correspond to false alarms. In addition, the sensor network is characterized by a non-zero probability of missing a given target even if this target is within its field of view. While these characteristics are sufficient for capturing the essence of most tracking problems, no consideration has been taken for environments including areas which are not under sensory coverage. The purpose of this section is to provide an extension of the classical (MC)JPDAF algorithm to cope with such (partially covered) environments.

The particularity of an environment with non-covered areas is that there are possibly two reasons for explaining why a given target is not detected by the sensor network:

1. the target is within an area under sensory coverage but the sensors were unable to detect it,

2. the target is within an area which is not under sensory coverage and, *de facto*, it cannot be detected.

The essence of what is proposed in this section is a mechanism for reasoning on these two cases. Inspired by the introduction of the zero-indexed observation in the classical (MC)JPDAF which corresponds to a target being undetected in an area under sensory coverage (see Section 6.3.3), we introduce a fictive "-1"-indexed observation corresponding to a situation caused by the second above-mentioned reason. Consequently, any feasible (joint) association hypothesis retrieved within the framework of the classical (MC)JPDAF should next be re-decomposed in what we call "unambiguous" hypotheses to cover all the possibilities. For example, let us assume that we are tracking four targets (A, B, C, D) and that, after receiving an observation $\mathbf{z}_t = \{\mathbf{z}_{1,t}, \mathbf{z}_{2,t}\}$ at time step t, the association hypothesis $\{(A, \mathbf{z}_{1,t}), (D, \mathbf{z}_{2,t})\}$ is part of the feasible ones. Focusing on that particular hypothesis, it should be decomposed into the following four "unambiguous" hypotheses:

$$\{(A, \mathbf{z}_{1,t}), (D, \mathbf{z}_{2,t})\} \Rightarrow \begin{cases} \{(A, \mathbf{z}_{1,t}), (B, \mathbf{z}_{0,t}), (C, \mathbf{z}_{0,t}), (D, \mathbf{z}_{2,t})\} & , \\ \{(A, \mathbf{z}_{1,t}), (B, \mathbf{z}_{0,t}), (C, \mathbf{z}_{-1,t}), (D, \mathbf{z}_{2,t})\} & , \\ \{(A, \mathbf{z}_{1,t}), (B, \mathbf{z}_{-1,t}), (C, \mathbf{z}_{0,t}), (D, \mathbf{z}_{2,t})\} & , \\ \{(A, \mathbf{z}_{1,t}), (B, \mathbf{z}_{-1,t}), (C, \mathbf{z}_{-1,t}), (D, \mathbf{z}_{2,t})\} & , \end{cases}$$

where $\mathbf{z}_{0,t}$ and $\mathbf{z}_{-1,t}$ stand respectively for the zero-indexed and "-1"-indexed observations.

From now on, let Θ be the set of all the generated "unambiguous" hypotheses and let θ be one of these hypotheses. Focusing on θ, one can easily compute the number N_{DT} of detected targets, the number N_{FT} of false alarms, the number N_{HT} of (hidden) targets in non covered areas (those associated with observation $\mathbf{z}_{-1,t}$) and finally, the number of targets in covered areas which are not detected, which is equal to $K - N_{DT} - N_{HT}$, where K is the total number of targets under tracking. Also, it is possible to to determine the identity l_j of the target associated with the j^{th} atomic observation ($j > 0$). We thus have

$$l_j = \begin{cases} k & \text{if } (j,k) \in \theta \text{ and } j > 0, \\ 0 & \text{otherwise.} \end{cases} \tag{7.37}$$

Moreover, it is possible to define the set T_{-1} of all the targets that have been associated with the fictive observation $\mathbf{z}_{-1,t}$ in θ. On this basis, Equation 7.33 for computing the posterior

distribution $p(\theta|\mathbf{z}_{1:t})$ of the hypothesis θ is then rewritten as

$$p(\theta|\mathbf{z}_{1:t}) \propto (1-P_D)^{K-N_{DT}-N_{HT}} P_D^{N_{DT}} \times \lambda_{FT}^{N_{FT}} \times \left[\prod_{j:l_j\neq 0} p^{l_j}(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1})\right] \times \left[\prod_{l\in T_{-1}} p^{l}(\mathbf{z}_{-1,t}|\mathbf{z}_{1:t-1})\right], \tag{7.38}$$

where P_D is the detection probability of the sensor network, λ_{FT} is the false alarm rate, $j : l_j \neq 0$ denotes the set of atomic observations which are considered to originate from a given target, and $l \in T_{-1}$ represents targets which are considered to be in non-covered areas. In Equation 7.38, the term $p^l(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1})$ with $j \notin \{0, -1\}$ for a given target l represents the predictive likelihood of the j^{th} atomic observation using the information from the l^{th} target and it is obtained using Equation 7.34 (or Equation 7.36 in the Monte-Carlo version) defined previously. Besides, the term $p^l(\mathbf{z}_{-1,t}|\mathbf{z}_{1:t-1})$, which represent the predictive likelihood of a given target l being associated with $\mathbf{z}_{-1,t}$, is given by

$$p^l(\mathbf{z}_{-1,t}|\mathbf{z}_{1:t-1}) = \int p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{l,t}) p(\mathbf{x}_{l,t}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{l,t}, \tag{7.39}$$

$$\approx \sum_{i=1}^{N} w_{l,t|t-1}^{i} p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{l,t}^{i}), \text{ (Monte-Carlo version)} \tag{7.40}$$

where $w_{l,t|t-1}^{i}$ is defined according to Equation 7.35 and $p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{l,t})$, which depends on the topology of the environment, corresponds to the probability of be unobservable while being within a non-covered area.

To encourage smooth transitions from covered areas to non-covered areas, we assume that $p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{l,t})$ is not from a $0-1$ distribution (Dirac distribution), but rather a continuous distribution. As the observation noise is assumed Gaussian, this distribution can be assimilated to the portion of the Gaussian belonging to the areas not covered by the sensor network. Such a portion is then approximated following the procedure described below (see Section 5.3.3):

1. As illustrated in Figure 7.5, we first consider a region around the location data $\mathbf{x}_{l,t}$ (represented by P in the figure) with a predefined radius (r_h in the figure).

2. Then, we discretize the resulting region into small cells and compute the probability, with respect to P, of each cell (center) belonging to the region.

3. Afterward, $p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{l,t})$ is estimated as the proportion of the cells which are effectively within the non-covered area, that is,

$$p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{l,t}) = \frac{\sum \text{prob. of region's cells in non-covered areas}}{\sum \text{prob. of all region's cells}}.$$

After computing the probability $p(\theta|\mathbf{z}_{1:t})$ for all the θ in Θ, we compute, for all possible couples (j, k), the probability β_{jk} that the j^{th} atomic observation comes from the k^{th} target according to Equation 7.31. Finally, Equation 7.30 for computing the likelihood $p(\mathbf{z}_t|\mathbf{x}_{k,t})$ of the received

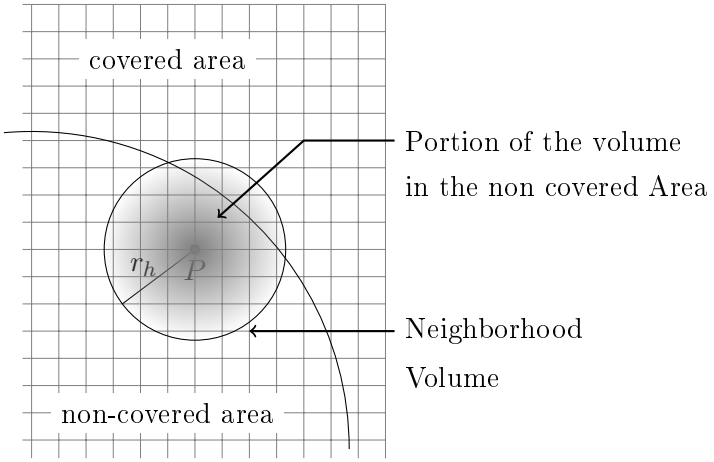


Figure 7.5: An estimation of $p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{.,t})$.

observation data $\mathbf{z}_t$ with respect to a given target k is rewritten as

$$p(\mathbf{z}_t|\mathbf{x}_{k,t}) = \beta_{-1k}.(p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{k,t})) + (1 - (p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{k,t}))).\sum_{j=0}^{M_t} \beta_{jk}.p_h(\mathbf{z}_{j,t}|\mathbf{x}_{k,t}), \tag{7.41}$$

where M_t is the number of the atomic observations composing $\mathbf{z}_t$ and $p_h(\mathbf{z}_{0,t}|\cdot) = 1$. It is interesting to notice that in case the environment is fully under sensory coverage, Equation 7.41 perfectly corresponds to the formulation from classical (MC)JPDAF as the first term of the equation will be canceled out and $p_{Env}(\mathbf{z}_{-1,t}|\mathbf{x}_{k,t})$ will be always equal to zero. This concludes our extension of the (MC)JPDAF for dealing with environments containing non-covered areas.

In the next section, we introduce IT-MCJPDAF, a new tracking algorithm capable of handling interacting targets and which results from the combination of the (extended) MC-JPDAF algorithm with the approach designed in Section 7.4 for computing the predicted distributions of such targets.

7.5.3 Tracking interacting targets with JPDA-like filter

This section describes how the factored approach we designed for computing the predicted distributions of interacting targets can be embedded within the (extended) MC-JPDAF algorithm. Thus, recalling from Section 7.3, we consider modeling dynamics-based interactions between targets as

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{k=1}^{K} p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1}).$$

7.5.3.1 Managing targets' interactions

Let $\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^N$ be the set of weighted particles representing the posterior distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ of a given target k at time step $t-1$. Our main objective here is to compute $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}$, the set of weighted particles representing the predicted distribution

$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ of any target k at time step t while taking into account potential interactions that may exist between targets at time step $t-1$.

To achieve this objective, we proceed according to the following procedure:

1. **Probability aggregation:** First, we aggregate the posterior distribution $\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^N$ and compute the set $\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$ of weighted representative states of each target k (see Section 7.4.3).

2. **Interaction graphs' generation:** Then, based on the obtained representatives and in conjunction with the function Φ, we built on the fly the set $\{(\bar{G}_c, \bar{w}_c)\}$ of weighted interaction graphs where each graph $\bar{G}_c$ involves a distinct combination of K representatives of all the targets, and the weight $\bar{w}_c$ of the graph $\bar{G}_c$ is given by Equation 7.18.

3. **Interaction graphs' regrouping:** The next step consists in computing, for any target k's representative $\hat{\mathbf{x}}^r_{k,t-1}$, the set $\Delta_{k,r} = \{(\Delta^g_{k,r}, \upsilon^g_{k,r})\}_{g=1}^{\Xi_{k,r}}$ of weighted classes of interaction graphs where $\Delta^g_{k,r}$ is a class regrouping interaction graphs which involve $\hat{\mathbf{x}}^r_{k,t-1}$ and which share the same neighborhood $\tilde{\mathcal{N}}_g(k,r)$ of $\hat{\mathbf{x}}^r_{k,t-1}$. This is mainly an optimization step as described in Section 7.4.3.3 since, under the assumptions made, only the neighborhood of a target can affect its behavior. $\upsilon^g_{k,r}$ is the weight of the class $\Delta^g_{k,r}$ given by Equation 7.24 and $\Xi_{k,r}$ is the cardinality of $\Delta_{k,r}$.

4. **Predicted distribution's estimation:** Finally, the predicted distribution of each target k is computed. Basically, given a particle $\mathbf{x}^i_{k,t-1}$ from the posterior distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ of target k, $Q(\cdot|\mathbf{x}^i_{k,t-1}, \mathbf{z}_{1:t-1})$ (see Equation 7.25) is approximated using multiple particles as described below. We first determine $r_i = \Upsilon_k(\mathbf{x}^i_{k,t-1})$, the region of the distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ which it belongs to, and thus, the corresponding representative state $\hat{\mathbf{x}}^{r_i}_{k,t-1}$. Then, we consider the set $\Delta_{k,r_i} = \{(\Delta^g_{k,r_i}, \upsilon^g_{k,r_i})\}_{g=1}^{\Xi_{k,r_i}}$ of classes of interaction graphs and, thereby, the set $\{\tilde{\mathcal{N}}_g(k,r)\}_{g=1}^{\Xi_{k,r}}$ of corresponding neighborhoods of $\hat{\mathbf{x}}^{r_i}_{k,t-1}$ associated with each class. Afterward, the particle $\mathbf{x}^i_{k,t-1}$ is simulated using each of these neighborhoods, thus leading to Ξ_{k,r_i} new weighted particles $\{\mathbf{x}^{i,g}_{k,t|t-1}, w^{i,g}_{k,t|t-1}\}_{g=1}^{\Xi_{k,r}}$ accounting for the predicted distribution of target k. That is,

$$\begin{aligned}
&\mathbf{x}^{i,g}_{k,t|t-1} \sim q(\cdot|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1}, \mathbf{z}_t), && \forall g = 1, \cdots, \Xi_{k,r_i};\\
&w^{i,g}_{k,t|t-1} \propto w^i_{l,t-1} \times \frac{\upsilon^g_{k,r_i}}{\hat{w}^{r_i}_{k,t-1}} \times \frac{p(\mathbf{x}^{i,g}_{k,t|t-1}|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1})}{q(\mathbf{x}^{i,g}_{k,t|t-1}|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1}, \mathbf{z}_t)}, && \forall g = 1, \cdots, \Xi_{k,r_i},
\end{aligned} \tag{7.42}$$

where $q(\cdot|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1}, \mathbf{z}_t)$ and $p(\cdot|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1})$ are respectively the proposal density and the dynamics model of target k. They both take into account the neighborhood of the target.

As it can be noticed from the above description, the number of particles of the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ generated from a given particle $\mathbf{x}^i_{k,t-1}$ of the posterior distribution $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ is equal to the number Ξ_{k,r_i} of corresponding classes of interaction graphs. Therefore, the total number N_p of generated particles is defined as $N_p = \sum_{i=1}^N \Xi_{k,r_i}$. In the special case

where $\Xi_{k,r_i} = 1, \forall i \leq N$ (this is always true when each target posterior distribution is characterized by a single representative), we have $N_p = N$ and we thus obtain the set $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^N$ of weighted particles characterizing the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ of target k (by setting $\mathbf{x}^i_{k,t|t-1} = \mathbf{x}^{i,1}_{k,t|t-1}$ and $w^i_{k,t|t-1} = w^{i,1}_{k,t|t-1}$). However, in general, $N_p > N$ and thereby, we have an increased number of particles within the system. In such a situation, there is a need to find a way to reduce to N the number of particles, otherwise the filtering process will suffer from the curse of dimensionality. The purpose of the next section is to discuss possible reduction policies.

7.5.3.2 Particle reduction policy

We have seen in the previous section that, by applying the factored approach designed in Section 7.4, the number N_p of particles characterizing the predicted distribution of a given target k may be greater than N, the number of allowed particles per target in the filtering process. In this section, we are interested in reducing this number to N without severely degrading the quality of the tracking process. Intuitively, we consider in this work that such a reduction procedure should normally intervene after the re-weighting stage of the filtering process when new weights have already been assigned to the particles on the basis of the received observation data.

Simple approaches may consist in either selecting a single particle from the descendants of each particle at time step $t-1$ according to a specified criterion (e.g., the highest weight), or globally sampling N particles from the whole particle set according to their weights. However, these approaches tend to degrade the performance of the filtering process which will privilege local maxima to the state space exploration. This is particularly noticeable when particles enter an area which is not under sensory coverage. In such a scenario, while there is no additional information for improving the weights of particles, a good hint for the reduction criterion will be to preserve a behavioral variety within the resulting set of particles rather than removing particles based on their weight. On this basis, we propose that the reduction policy should be implemented in such a way to encourage diversity between the selected particles.

The problem we are facing can be easily assimilated to the common *diversity maintenance problem* encountered in the field of evolutionary algorithms (EA), where there is a need to intelligently select, at each iteration, individuals from a given population in order to proceed with them to the next iteration. In that field, each individual is characterized by a *fitness score* symbolizing how close it is to the optimal of a given function and the purpose of an EA is to find such an optimal. Popular approaches have been proposed to address the diversity maintenance problem such as the well studied *Goldberg's fitness sharing algorithm* [Goldberg and Richardson, 1987] or, more recently, *Petrowski's clearing method* [Petrowski, 1996]. These approaches rely on a dissimilarity measure between individuals to determine whether they belong to the same subpopulation (class of individuals) or not. Then, the fitness score of an individual is altered taking into account other individuals belonging to its subpopulation. While a similitude can straightforwardly be established with a set of weighted particles (the fitness is the weight of a particle), these approaches cannot be directly applied in the context of particle filtering (PF) mainly because the objective is not the same in both cases: EAs seek to optimize a provided but static function while the goal of a PF algorithm is to estimate a probability distribution characterizing a system evolving over time.We thus need to consider new diversity maintenance algorithms — in our case, reduction policies — adapted for a particle filtering process.

Like algorithms designed in the field of EAs [Goldberg and Richardson, 1987] [Petrowski,

1996], such a reduction policy should rely on a dissimilarity metric to divide the particle set into several disjoint subsets (subpopulations or clusters), each one regrouping particles having mutual (dissimilarity) distances within a given threshold. Techniques for subdividing a particle set into disjoint clusters have previously been discussed in Section 7.4.4.2 when computing targets' representative states. We can easily leverage such techniques in the reduction process. From now on, we assume that the provided particle set, whose size is N_p s.t. $N < N_p$, has been partitioned into a set $SP = \{SP_s, \check{w}_s\}$ of $|SP|$ clusters (subpopulations) where SP_s denotes the subset of particles of size $|SP_s|$ belonging to the s^{th} cluster and $\check{w}_s$ is the weight (the sum of the particle weights) of the s^{th} cluster. In this work, we choose to assess the diversity within a population according to the following metrics:

- the relative size of each cluster;
- the relative weight of each cluster.

On this basis, we present two strategies for reducing the size of the particle set to N. In the first strategy, it is assumed that the resulting particle set is a subset of the particle set provided as input, that is, the reduction is based on a selection mechanism. On the other hand, the second strategy is based on a sampling mechanism and therefore, a given particle (in the provided particle set) can be duplicated within the resulting particle set.

Selection-based reduction policy

The guideline of this policy consists in keeping the above-defined metrics unchanged after the reduction process. Basically, the general idea governing the reduction policy can be decomposed into two stages:

- **Selection:** The selection of particles is made according to the size of the cluster to which they belong. In other words, the larger the cluster, the higher the ratio of individuals coming from that cluster in the final selection result. More specifically, we select $\sim \frac{|SP_s|}{N_p}N$ individuals from each subpopulation SP_s using a random sampling strategy *without replacement*, where the probability for a particle to be selected is proportional to its weight[17]. This way of performing selection over a population is known in the literature as *stratified sampling* [Kitagawa, 1996] and it guarantees that the relative cluster sizes are maintained after the reduction process.
- **Re-Weighting:** Once the particles from different clusters have been selected, the next step consists in modifying their weights in such a way to keep unchanged the relative weights of the clusters. Let $SEL_s = \{\mathbf{x}_{:,:}^{e,s}, w_{:,:}^{e,s}\}_{e=1}^{\check{E}_s}$ be the set of all the $\check{E}_s$ particles selected from the s^{th} cluster with their associated weights. Let $\check{w}_{SEL_s} = \sum_{e=1}^{\check{E}_s} \mathbf{w}_{:,:}^{e,s}$ be the weight of SEL_s. A simple and intuitive way for performing re-weighting is to assign to each selected particle

[17] Here, we assume that $\sim \frac{|SP_s|}{N_p}N$ is always an integer for all the generated subpopulations. If it is not the case, one can easily rely on the techniques used in the residual resampling strategy [Liu and Chen, 1998], at the cluster level, to determine subpopulations from which additional particles can be selected for completing the overall size of selected particles to N.

$\mathbf{x}_{\cdot,\cdot}^{e,s}$ the weight $\check{w}_{\cdot,\cdot}^{e,s}$ given by

$$\check{w}_{\cdot,\cdot}^{e,s} = w_{\cdot,\cdot}^{e,s} \frac{\check{w}_s}{\check{w}_{SEL_s}}. \tag{7.43}$$

Equation 7.43 simply stipulates that the weights of the selected particles from a given cluster need to be first normalized and then rescaled to the overall weight of the underlying subpopulation.

Sampling-based reduction policy

The main difference between this second policy and the one presented previously is that there could be several instances of a specific particle within the final resulting set. The general idea of this reduction policy consists in two stages:

- **Sampling:** From each cluster SP_s, use the residual sampling strategy [Liu and Chen, 1998] for sampling with replacement $\sim \check{w}_s N$ particles to be part of the final particle set.
- **Re-Weighting:** Assign to each of the obtained samples a weight of $1/N$.

7.5.3.3 Summary

In this section, we have presented how to combine the (extended) MC-JPDAF algorithm with the factored approach designed in Section 7.3 for estimating the predicted distributions of interacting targets without neglecting these interactions.

Up to now, the purpose of the representative states is mostly restricted to the prediction step of the filtering process although they are used for maintaining behavioral diversity within the reduction policy in case of multiple representatives per target. One question of particular interest concerns their utility within the correction step of the filter in such a way to further improve the tracking system. The next section presents an alternative approach to the classical soft-gating procedure in which multiple representatives are used to improve the quality of the data association.

7.5.4 Multiple-representative-based soft-gating

In this section, we present a variant of the soft-gating technique introduced in Section 7.5.1.1. Like in Section 7.5.1.1, it is assumed that the observation model can be put in the form $h(\mathbf{x}_{\cdot,t}) + \mathbf{v}_{\cdot,t}$, where $\mathbf{v}_{\cdot,t}$ is characterized by a zero-mean Gaussian density with a covariance matrix $\mathbf{R}_t$. As stated before, the soft-gating technique aims at "casting" the predicted distribution of a given target with respect to the observation model into a unique Gaussian distribution from which a Mahalanobis distance can be easily computed to determine whether a given atomic observation lies within the considered target's gate or not. This can be viewed as a single-representative-based approach where a representative state (here, the mean of the resulting Gaussian distribution), in conjunction with its related properties (here, the covariance matrix of the resulting Gaussian distribution), is used for computing the validation gate.

However, in case of a target's dynamics with strong non-linearity, it may not always be convenient to approximate the resulting predicted distribution — which may be of any shape (e.g., a multi-modal distribution) — with a single Gaussian distribution. Indeed, in such a case, the variance of the resulting Gaussian distribution is large due to the dispersion of the data through the different modes. As a direct consequence, several non-relevant atomic observations are candidates for potential association with the related target including clutters and/or observations generated by other targets. A typical example illustrating our concern is depicted in Figure 7.6b. To remedy this situation and gain more precision in data-association issues, we propose, instead of using a single representative, to use multiple representatives obtained by regrouping/clustering the particle set into disjoints clusters. In other words, we propose to approximate the underlying distribution not by a single Gaussian distribution, but a mixture of Gaussian distributions, each one characterizing a distinct cluster.

Let $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}^N_{i=1}$ be the particle set characterizing the predicted distribution $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ of target k at time step t. The techniques for partitioning a particle set into significant clusters based on affinity/dissimilarity functions, previously discussed in Section 7.4.4.2, can easily be re-used to serve our purpose. Let $SPO = \{(SPO_s, \check{wo}_s)\}$be the set of $|SPO|$ clusters generated after the application of such a partitioning technique, where SPO_s and $\check{wo}_s$ are respectively the set of particles being part of the s^{th} cluster and the sum of their weights (the total weight of the cluster). Then, the multiple-representative-based soft-gating for target k is described as follows:

- compute the mean $\mu^{s,h}_{k,t|t-1}$ of the atomic observations resulting from the particles belonging to each cluster s as

$$\mu^{s,h}_{k,t|t-1} = \sum_{i \in SPO_s} \frac{w^i_{k,t|t-1}}{\check{wo}_s} h(\mathbf{x}^i_{k,t|t-1}); \tag{7.44}$$

- compute, for each cluster s, the covariance matrix $\mathbf{C}^{s,\nu}_{k,t}$ of the predicted atomic observation $\mu^{s,h}_{k,t|t-1}$ as

$$\mathbf{C}^{s,\nu}_{k,t} = \mathbf{R}_t + \sum_{i \in SPO_s} \frac{w^i_{k,t|t-1}}{\check{wo}_s} (h(\mathbf{x}^i_{k,t|t-1}) - \mu^{s,h}_{k,t|t-1})(h(\mathbf{x}^i_{k,t|t-1}) - \mu^{s,h}_{k,t|t-1})^T. \tag{7.45}$$

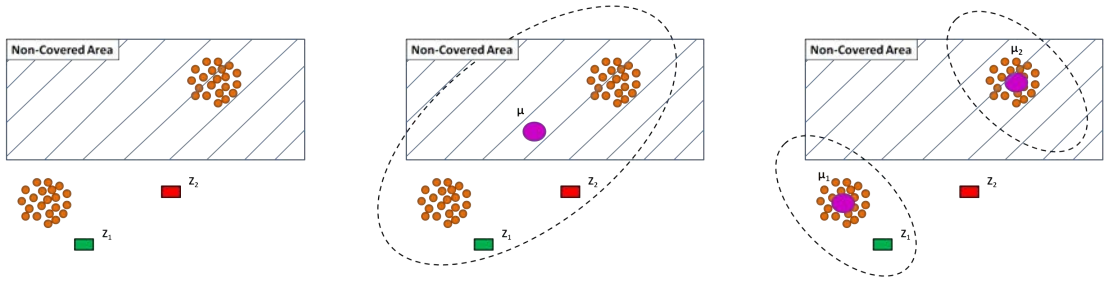
Finally, a given atomic observation $\mathbf{z}_{j,t}$ can be associated to target k if it lies at least within one of these validation areas, that is, there exists s such that

$$(\mathbf{z}_{j,t} - \mu^{s,h}_{k,t|t-1})^T \mathbf{C}^{s,\nu-1}_{k,t} (\mathbf{z}_{j,t} - \mu^{s,h}_{k,t|t-1}) \leq \eta, \tag{7.46}$$

where η is a preset threshold. An illustration of the application of such a procedure is shown in Figure 7.6c. It is interesting to notice that in case $|SPO| = 1$, the proposed approach is completely equivalent to the classical soft-gating procedure.

7.5.5 Summary

From Section 7.5.2 to Section 7.5.4, we have presented IT-MCJPDAF, a new algorithm based on the JPDA framework capable of (1) dealing with environments containing non-covered areas and (2) tracking interacting targets. Table 7.1 recapitulates the different parameters used



(a) Initial Situation (b) Soft-Gating (c) Our approach

Figure 7.6: Example of a multiple-representative-based soft-gating. In (a), a set of particles (colored circles) representing a given target distribution is depicted. Also, two atomic observations, Z_1 and Z_2, are represented in colored rectangles. Z_1 is generated by the considered target while Z_2 is either a clutter or generated by another target. Using the classical soft-gating (Fig. (b)), a single representative is computed as the basis of the gating and the procedure considers both atomic observations as potential candidates. In (c), using multiple representatives, the procedure manages to consider only Z_1 as a potential candidate.

in the algorithm. Also, a pseudo-code of a one-step IT-MCJPDAF algorithm is reported in Algorithms 7.3/ 7.4 where all the steps described previously are described in form of procedures.

The main discriminator of IT-MCJPDAF with respect to the classical (MC-)JPDAF algorithm is that it relies on target representatives for computing in a finer way the different distributions of the filtering process. More specifically, these representatives intervene at three levels in the filtering procedure:

1. **Target dynamics**: The representatives are used within the individual dynamics of each target being tracked when computing the predicted distributions of all the targets at the prediction step of the process.

2. **Gating**: The representatives are used during the resolution of data association issue, at the correction step of the process, for efficiently selecting atomic observations which probably come from each target.

3. **Behavioral diversity**: When the number of particles increases, the representatives are used, during the reduction stage (correction step), for intelligently reducing this number while keeping a diversity within the resulting particle set.

Generally, these representatives are computed using a clustering algorithm for aggregating a probability distribution (see Section 7.4.4) whose nature depends on which above-mentioned level they are meant to be used (posterior distributions from previous time step for the first level, predicted distributions from current time step for the second level, and posterior distributions from current time step for the third level). Thereby, an additional computational cost resulting from these clustering procedures is added on top of the inference complexity. In the next section, we study the global complexity of the IT-MCJPDAF algorithm.

Algorithm 7.3: Monte Carlo JPDAF For Interacting Targets

Algorithm `ITMC-JPDAF`$(\{\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^N\}_{k=1}^K, \mathbf{z}_t)$

`/* Prediction Step */`

for $k = 1, \cdots, K$ **do**

  $\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k} =$ `ProbAgg`$(\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^N)$

$\{(\bar{G}_c, \bar{w}_c)\} =$ `IGraphGen`$(\{\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}\}_{k=1}^K)$

for $k = 1, \cdots, K$ **do**

  $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N_p^k} =$ `ITargetPrediction`$(\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^N, \{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}, \{(\bar{G}_c, \bar{w}_c)\}, \mathbf{z}_t)$

`/* Correction Step */`

$\Theta =$ `AssocHypGen`$(\{\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N_p^k}\}_{k=1}^K, \mathbf{z}_t)$

for $k = 1, \cdots, K$ **do**

  $\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}_{i=1}^N\} =$ `ITargetCorrection`$(\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N_p^k}, \mathbf{z}_t, \Theta)$

return $\{\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}_{i=1}^N\}_{k=1}^K$

Procedure `ProbAgg`$(\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^N)$

  compute the representatives $\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$ (see Section 7.4.3)

  return $\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$

Procedure `IGraphGen`$(\{\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}\}_{k=1}^K)$

  for $c = 1, \cdots, R_1 \times R_2 \times \cdots \times R_k$ **do**

    build the interaction graph $\bar{G}_c$ using the interaction indicator function Φ

    compute the weight $\bar{w}_c$ of $\bar{G}_c$ according to Equation 7.18

  return $\{(\bar{G}_c, \bar{w}_c)\}$

Procedure `AssocHypGen`$(\{\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N_p^k}\}_{k=1}^K, \mathbf{z}_t)$

  for $k = 1, \cdots, K$ **do**

    for $j = 1, \cdots, M_t$ **do**

      compute the predictive likelihood $p^k(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1})$ of $\mathbf{z}_{j,t}$ with respect to target k using Equation 7.36

      use the soft-gating to determine if atomic observation j eventually comes from target k (see Section 7.5.4)

  compute the set $\Theta = \{\theta\}$ of feasible joint association hypotheses

  for $\theta \in \Theta$ **do** compute $p(\theta|\mathbf{z}_{1:t})$ using Equation 7.38

  return Θ

Algorithm 7.4: Monte Carlo JPDAF For Interacting Targets (Continuation)

Procedure `ITargetPrediction`($\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}_{i=1}^{N}$, $\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$, $\{(\bar{G}_c, \bar{w}_c)\}$, $\mathbf{z}_t$)

  $\{\{(\Delta^g_{k,r}, \upsilon^g_{k,r}, \tilde{\mathcal{N}}_g(k,r))\}_{g=1}^{\Xi_{k,r}}\}_{r=1}^{R_k}$ = `IGraphRegroup`($\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$, $\{(\bar{G}_c, \bar{w}_c)\}$);

  for $i = 1, \cdots, N$ **do**

    get the region $r_i = \Upsilon_k(\mathbf{x}^i_{k,t-1})$ of particle i ;

    for $g = 1, \cdots, \Xi_{k,r_i}$ **do**

      sample $\mathbf{x}^{i,g}_{k,t|t-1}$ from $q(.|\mathbf{x}^i_{k,t-1}, \tilde{\mathcal{N}}_g(k, r_i), \mathbf{z}_t)$;

      compute $w^{i,g}_{k,t|t-1}$ using Equation A.21;

      for $j = 1, \cdots, M_t$ **do**

        compute $p_h(\mathbf{z}_{j,t}|\mathbf{x}^{i,g}_{k,t|t-1})$;

  $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N^k_p} = \cup_{i=1}^{N}\{\mathbf{x}^{i,g}_{k,t|t-1}, w^{i,g}_{k,t|t-1}\}_{g=1}^{\Xi_{k,r_i}}$;

  return $\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N^k_p}$;

Procedure `IGraphRegroup`($\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$, $\{(\bar{G}_c, \bar{w}_c)\}$)

  for $r = 1, \cdots, R_k$ **do**

    build the set $\Delta_{k,r} = \{\Delta^g_{k,r}, \upsilon^g_{k,r}\}_{g=1}^{\Xi_{k,r}}$ (see Section 7.4.3.3);

    for $g = 1, \cdots, \Xi_{k,r}$ **do**

      get the neighborhood $\tilde{\mathcal{N}}_g(k,r)$ of the r^{th} representative of target k in $\Delta^g_{k,r}$;

  return $\{\{(\Delta^g_{k,r}, \upsilon^g_{k,r}, \tilde{\mathcal{N}}_g(k,r))\}_{g=1}^{\Xi_{k,r}}\}_{r=1}^{R_k}$;

Procedure `ITargetCorrection`($\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t|t-1}\}_{i=1}^{N^k_p}$, $\mathbf{z}_t$, Θ)

  `/* Re-weighting */`

  for $j = 0, \cdots, M_t$ **do**

    compute β_{jk} using 7.31;

  for $i = 1, \cdots, N^k_p$ **do**

    compute $p(\mathbf{z}_t|\mathbf{x}^i_{k,t|t-1})$ using Equation 7.41 ;

    compute the importance weight $w^i_{k,t} = w^i_{k,t|t-1} \times p(\mathbf{z}_t|\mathbf{x}^i_{k,t|t-1})$;

  normalize the importance weights: $w^i_{k,t} = \frac{w^i_{k,t}}{\sum_{i_1=1}^{N^k_p} w^{i_1}_{k,t}}$;

  `/* Reduction Step */`

  if $N^k_p > N$ **then**

    $[\{\mathbf{x}^{i_1}_{k,t|t-1}, w^{i_1}_{k,t}\}_{i_1=1}^{N}]$ = Reduction $(\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t}\}_{i=1}^{N^k_p})$. see Section 7.5.3.2;

  `/* Re-sampling */`

  compute the effective size $\widehat{N}^k_{eff} = \frac{1}{\sum_{i=1}^{N}(w^i_{k,t})^2}$;

  if $\widehat{N}^k_{eff} < N_T$ **then**

    $[\{\mathbf{x}^{i_1}_{k,t}, w^{i_1}_{k,t}\}_{i_1=1}^{N}]$ = Resample $[\{\mathbf{x}^i_{k,t|t-1}, w^i_{k,t}\}_{i=1}^{N}]$;

  else

    for $i = 1, \cdots, N$ **do** assign $\mathbf{x}^i_{k,t} = \mathbf{x}^i_{k,t|t-1}$;

  return $\{\mathbf{x}^i_{k,t}, w^i_{k,t}\}_{i=1}^{N}$;

Table 7.1: Recapitulation of IT-MCJPDAF parameters

Parameters	**Meaning / Possible Values**
Generic Parameters	
K	Number of targets being tracked
N	Number of particles for representing target distributions
N_T	Resampling threshold
Prediction Step	
Target Representative Computation	Single-representative-based approaches (e.g., MEAN, MEDIAN, MODE)
	Multiple-representative-based approaches (e.g., centroid-based clustering, clearing-based clustering)
Φ	Prior function defining the conditions under which two targets are assumed interacting
Correction Step - Data Association	
Gating Procedure	Single-representative-based approach (classical soft-gating) based on the MEAN
	Multiple-representative-based soft-gating (based on a clustering procedure)
η	Gating threshold
r_h	Radius for controlling smooth transition from covered areas/non-covered areas
Correction Step - Reduction Procedure	
Reduction Policy (based on clustering)	Selection-based Reduction
	Sampling-based Reduction
Centroid-based Clustering	
R	Number of representatives to generate
N_c	Max number of iterations
F_{aff}	Prior affinity function
Clearing-based Clustering	
F_{aff}	Prior affinity function
ϵ_A	Dissimilarity threshold
$MaxSize$	Parameter for controlling the density of the generated clusters

7.5.6 Complexity analysis

The objective of this section is to provide a concise analysis of the complexity of the IT-MCJPDAF algorithm. Here, we intentionally neglect the complexity related to the resolution of the data association issue since it is common to all the JPDA-based approaches.

Based on the description we made in Section 7.4.4.2, it comes out that the number of generated representatives for characterizing the k^{th} target's distribution may vary from one time step to the next depending on whether we are using for example a density-based or a centroid-based clustering methodology. Similarly, the number of connections between targets within the built interaction graphs varies with time depending on whether targets are considered in interaction or not. For all these reasons, it is hard to obtain the precise complexity of the algorithm and therefore, the analysis we perform in this section is based on a pessimist assumption.

Let K be the number of targets being tracked and let N be the number of particles used to represent each target distribution. Let us assume that each target is characterized by at most R representatives at each time step. Under these assumptions, there are R^K possible combinations of representatives of all the targets, each one leading to an interaction graph. Also, the number of interaction graphs in which a given target representative is involved is R^{K-1}. Let N_I be the maximal number of direct connections for each target's representative within any interaction graph in which it appears. To avoid performing the optimization step described in Section 7.4.3.3, let us assume that the neighborhood of each representative is different in each interaction graph in which it is present. If we assume that (1) the required time for simulating a target grows linearly with the number of targets it is interacting with, and (2) the complexity of the clustering procedure for generating representatives is at most $\mathcal{O}(N.\log(N))$ (e.g., density-based clustering), then the overall complexity of the prediction step of a filter associated to any target is given by:

$$\mathcal{O}(N.\log(N)) + \mathcal{O}(N.R^{K-1}.(N_I+1)). \tag{7.47}$$

From the assumptions made, the prediction step described above for each target will not generate N but $M = N.R^{K-1}$ particles which will subsequently be reduced to N. As described in Section 7.5.3.2, the reduction is based on a clustering procedure in such a way to preserve the behavioral diversity and it is the main concern of the filter at the correction step. Therefore, the complexity of the correction step can be easily approximated as done previously by:

$$\begin{aligned}\mathcal{O}(M.\log(M)) &= \mathcal{O}(N.R^{K-1}.\log(N.R^{K-1})),\\ &= \mathcal{O}(N.R^{K-1}.(\log(N) + (K-1)\log(R))).\end{aligned} \tag{7.48}$$

In both Equations 7.47 and 7.48, the term $N.R^{K-1}$ highlights the role of the target's representatives in our algorithm. In situations where each particle is considered as a representative (that is, $R = N$), our algorithm is analogous, in terms of complexity, to the ones reasoning on a joint target state. However, in practice, $R << N$ and thus, the overall complexity of the algorithm can be considered linearithmic with respect to N.

7.6 Experimental Evaluations

This section is devoted to the experimental evaluation of the algorithm we developed throughout this chapter.

7.6.1 Simulator

As stated in Section 7.1, our solution assumes the availability of a generic simulator modeling the targets' dynamics including their mutual interactions. To cope with this requirement, we consider the steering behavior model introduced by Reynolds [1999] for modeling autonomous characters. The steering behavior model, in its generality, describes a set of simple behaviors (comparable to physical forces) that, combined together, allow an autonomous character to exhibit realistic high-level navigational behaviors in games or in multi-agent computer simulations (e.g., obstacle avoidance, collision prevention, group formation).

In this work, we rely on the 2-D java implementation proposed in [Christian and Thomas, 2007] as the baseline simulator of steering behaviors. Also, the following simple behaviors are considered for defining the dynamics of each target (autonomous agent) being tracked:

- **containment behavior**: it allows agents to avoid obstacles;
- **separation behavior**: it makes agents keep a certain distance from each other;
- **wandering behavior**: it allows agents to move randomly;
- **seek behavior**: it allows agents to move toward a specific point (or goal) in the environment.

In the next section, we introduce the metrics used for assessing the performance of the proposed algorithm.

7.6.2 Performance metrics

The purpose of the algorithm proposed in this chapter consists in estimating, with multiple collaborating (sub-)filters, the states of all the targets being tracked. Given the simulator on which we rely as well as the simple behaviors we considered, the state $\mathbf{x}_{k,t}$ of a given target k at time step t is mainly characterized by the location of the underlying target within the environment, although there is a possibility to consider other feature — like the goal/activity — induced by the "seek behavior" when used. On this basis, the following criteria are used for assessing the performance of our algorithm:

- **correct tracks**: the average number of (sub-)filters that are correctly representing their initial target at each time step of the filtering process. The correctness of a track is assessed, at each time step, from the difference between the true target location and the estimated one. This difference should be lower than a preset threshold d_{corr};
- **track jumps**: the average number of (sub-)filters referring to a target which is different from the one it was initially associated with. The track jumping phenomenon generally happens when two targets pass very close to one another, so that a track related to a filter shifts from one target to the other one. This phenomenon is detected whenever the target is close, with a drift of a preset distance d_{jump}, to a filter meant to represent another target;
- **lost tracks**: the average number of (sub-)filters that do not correspond to any target in the environment at each time step. A lost track may occur principally because of two reasons: (1) the detection probability of the sensor network which may be lower than 1 and (2) the gating procedure which considers that an observation is valid if and only if it falls within a validation region of at least one target;
- **average error distance**: the target average error distance, $AvgErr_t$, computes, at each time step t, the average distance between the real location of a target and the estimated

location from its original associated filter, that is,

$$AvgErr_t = \frac{\sum_{k=1}^{K} dist(\mathbf{x}_{k,t}, \hat{\mathbf{x}}_{k,t})}{K},$$

where $dist(.,.)$ is a function computing the distance between two target states and $\hat{\mathbf{x}}_{k,t}$ is the estimated state of the filter initially associated with the k^{th} target;

- **average error distance to closest filter**: because of the track jump phenomenon, a (sub-)filter may be associated to a target different from the one it was initially associated with. In such situations, and considering the fact that the targets may share the same behavior and look identical, it would be unfair to expect the filtering process to re-associate a target to its originally associated filter. The target average error distance to closest filter, $AvgErrClo_t$, computes, at each time step, the average of the distance between the real location of a target and the estimated location from its currently associated filter, that is,

$$AvgErrClo_t = \frac{\sum_{k=1}^{K} dist(\mathbf{x}_{k,t}, \hat{\mathbf{x}}_{l,t})}{K},$$

where at time step t, the k^{th} target is assumed associated with the filter initially associated with the l^{th} target;

- **average goal similarity**: the average goal similarity, $AvgSim_t$ computes, at each time step t, the average similarity between the real goal of a target and the estimated goal from its original associated filter, that is,

$$AvgSim_t = \frac{\sum_{k=1}^{K} Sim(\mathbf{x}_{k,t}, \hat{\mathbf{x}}_{k,t})}{K},$$

where the similarity function $Sim(\cdot,\cdot)$ is worth 1 when the goals inherent to both states are identical, 0 otherwise;

- **run time**: the total time for tracking the targets over a temporal window of a preset length T. The experiments are performed on a 2.4 GHz AMD Opteron 250 CPU machine running under Ubuntu Linux.

The next section is dedicated to the evaluation of the proposed algorithm under the condition that single-representative-based approaches are used throughout the filtering process.

7.6.3 Single-representative-based evaluations

The objective of this section is to highlight the performance of the designed algorithm in tracking multiple interacting targets in complex situations when each target is associated with a single representative whatever the level in which these representatives are used. In other words, there is no clustering procedure to perform and the data-association issue is resolved using the classical soft-gating as found in the literature. Also, since there is a single representative per target, the reduction procedure has no place in this section. We start by presenting, in Section 7.6.3.1, the setup under which the experiments are carried out. Then, from Section 7.6.3.2 to Section 7.6.3.4, we analyze the impact of different parameters on the performance of the algorithm.

7.6.3.1 Experimental setup

In this section, we describe the environment, the behavioral model, the observation model, the initial belief as well as the parameters used in this experimental evaluation.

Environment

The environment under consideration is composed of a polygonal arena inscribed within a rectangle of width $375mm$ and height $300mm$ as depicted in Figure 7.7. It does not contain areas which are not under sensory coverage.

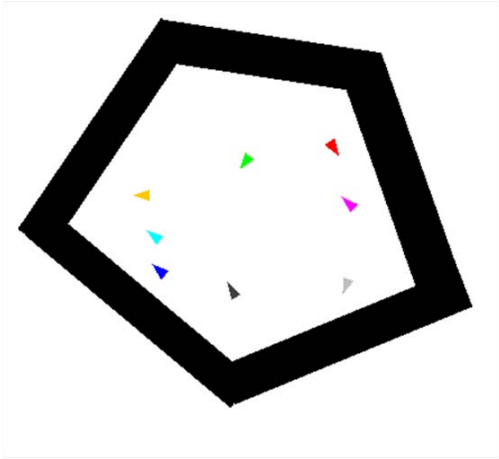


Figure 7.7: Experimental environment of Scenario 1: the considered arena is a pentagon. The colored triangles represent targets that should be tracked. The colors are just used for visual distinction and are **not** taken into account within the tracking algorithm for facilitating data association issues.

Behavioral model

We consider modeling targets whose behaviors consist in moving in a random fashion within the arena while avoiding each other and collisions with obstacles. Thereby, their underlying model results from a combination of the following simple behaviors: *containment* behavior, *separation* behavior, and *wandering* behavior.

Each target k is characterized by a state representing its location and its velocity within the 2-D plane, that is,

$$\mathbf{x}_{k,t} = [x_{k,t}, \dot{x}_{k,t}, y_{k,t}, \dot{y}_{k,t}],$$

where $(x_{k,t}, y_{k,t})$ and $(\dot{x}_{k,t}, \dot{y}_{k,t})$ stand respectively for the location and the velocity of the considered target. As illustrated in Figure 7.7, the physical shape of the target is an isosceles triangle whose height issued from the main summit is equal to $10mm$. Also, the orientation of the triangle corresponds to the one of the velocity vector of the related target. Finally, the simulator is in charge of the evolution of all the targets along the time.

Observation model

To perform the tracking process, we assume the availability of a sensor network which returns noisy location data of a given target when it is detected. Under this assumption, we define an observation function $h(\mathbf{x}_{k,t}, \mathbf{v}_t)$ as

$$h(\mathbf{x}_{k,t}, \mathbf{v}_t) = B\mathbf{x}_{k,t} + \mathbf{v}_t,$$

where B is the observation matrix defined as

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix},$$

and $(\mathbf{v}_t)_{t>0}$ is a bi-dimensional zero-mean Gaussian noise process characterized by a diagonal covariance matrix with a standard deviation of σ_x and σ_y for both the x and y coordinates respectively. Moreover, we assume that the sensor network is characterized by:

- a detection probability P_D of 0.95,
- a false alarm rate λ_{FT} of 0.8.

Next, we consider two settings of this model based on the observation noise:

- *Setting 1*: $\sigma_x = \sigma_y = 7mm$;
- *Setting 2*: $\sigma_x = \sigma_y = 15mm$.

These settings will be used afterward for assessing the performance of the algorithm with respect to the observation noise.

Initial belief

It is assumed that the initial locations of the targets being tracked is known and they are used to initialize the filtering process. Moreover, the target velocities are initialized from the Gaussian density $\mathcal{N}(1, 0.5)$.

Additional parameters

When not explicitly specified, the system parameters are set as follows:

- the number K of targets being tracked is 7;
- the number N of particles used to represent the targets' distributions is set to 500;
- the threshold N_T, under which the resampling step is performed, is set to $0.75N$;
- the radius r_h for managing smooth transition at the edge of covered/non-covered areas is set to $2mm$;

- the threshold η for the gating procedure is $40mm$;
- the single-representative-based aggregator used is the MEAN operator;
- $\Phi(\cdot,\cdot)$ is defined on the basis of a parametrized rule which states that two targets are assumed interacting if their distance is below a preset threshold d, that is,

$$Rule\{d\} : \Phi(\mathbf{x}_{k,t}, \mathbf{x}_{l,t}) = \begin{cases} 1 & \text{if } dist(\mathbf{x}_{k,t}, \mathbf{x}_{l,t}) < d, \\ 0 & \text{otherwise;} \end{cases}$$

- the length T of the temporal window of the filtering process is set to 500;
- the threshold distances d_{corr}/d_{jump} for evaluating the number of correct tracks/track jumps are set to $10mm$.

In the next section, we focus on the analysis of the parameter d of the function $\Phi(\cdot,\cdot)$ on the quality of the filtering process.

7.6.3.2 Impact analysis of Φ

In this section, we are interested in studying the impact of the function $\Phi(\cdot,\cdot)$ on the quality of the filtering process. As we do not know the true mechanism governing the dynamics-based interactions within the simulator, we consider running our algorithm with the following rules: *Rule0*, *Rule10*, *Rule20*, *Rule30*, *Rule40*, *Rule50*, *Rule100*, and *Rule200*. *Rule0* assumes that targets are never in interaction and, thus, behave independently. On the contrary, *Rule200* assumes that targets always interact with each other since the threshold distance ($200mm$) is close to the environment size.

To enrich our analysis, we further consider the two settings (*Setting 1*, *Setting 2*) of the observation model defined earlier. For each combination of one these settings with one of the rules defined above, we perform 100 runs of the algorithm. Also, for each of these settings, we carry out a single-target tracking process with the same experimental conditions (above-defined parameters). The outcomes of this single-target tracking process can be used as a reference during the evaluation since (1) no interaction are considered in this case and (2) the data-association issues are simplified. The results obtained are presented below.

Figure 7.8 gives an insight of the conducted experiments and presents some obtained trajectories. As it can be noticed from Figures 7.8c and 7.8d, targets are indeed evolving in a small space and can frequently interact. On the other hand, Figures 7.9a and 7.9b depict the average errors obtained for each rule under *Setting 1* and *Setting 2* respectively. In both settings, it can be observed that *Rule0* yields the poorest results. Also, it appears that the larger the interaction distance d, the better the result. Since *Rule0* considers that targets behave independently, the difference between results of *Rule200* and *Rule0* can be interpreted as the gain in tracking performance for reasoning on interactions on the joint targets state.

However, for both settings, we observe a stagnation of the improvement beyond a given distance threshold. This means that an attention has to be paid in determining the interaction neighborhood in our scenario as distances below the threshold are too short for predicting the correct behavior of the targets. This is even more important as the computational time increases with the size of the neighborhood considered (see Table 7.2). In particular, for *Setting1*, the

quality of the results are not improved from *Rule30* and the performance obtained is probably the best one can ever achieve given the similarity with the results obtained for the single target case. This shows the efficiency of our approach for tracking interacting targets.

Regarding *Setting2*, the results are not improved beyond *Rule40*. The standard deviations presented in Figure 7.10 show that *Rule20*, *Rule30* and *Rule50* exhibit significant differences. Also, these deviations are quite large for each of the above-mentioned rules. However, unlike results obtained in *Setting1*, we were unable to achieve performance comparable with the one obtained in the single target case. This is mainly because the observation noise is bigger and therefore, the overall filtering process is more prone to track jumps, which are hard to correct as targets behave identically. This is confirmed by Figure 7.11 where we depict the targets' average error distance to their closest filter ($AvgErrClo_t$) for each interaction rules. As it can be observed, the errors obtained are closer to the ones obtained in the single target case.

For a deeper analysis of our algorithm, numerical results obtained for *Setting2* are reported in Table 7.2 where the last column highlights the average number of interactions per time step considered by the algorithm. We note that the average computational time increases with the distance used to define the interaction rule. When *Rule0*, *Rule10*, *Rule20* or *Rule30* are considered, this time is close to 150 sec whereas there is a huge difference with *Rule200* where the computational time rises to almost 505 sec. The benefit of rules with low interaction distance is the consequence of considering less interactions (and therefore, less target representatives) to compute the prediction step: the number of launched simulations is the same but simulations with low interaction distance involve less targets than simulations with a high interaction distance. On the contrary, the number of lost tracks increases when the interaction distance decreases. This means that the filter is less and less accurate because it does not consider interactions between "actually" interacting targets. Filters *Rule40* and *Rule50* seem to be good compromises since their efficiencies are close to the filter *Rule200* which considers that targets always interact but their computation times are close to the one of the filtering process assuming independent targets.

In summary, we have shown through this scenario that the proposed algorithm is able to efficiently track interacting targets with a computational complexity close to the one of filtering systems involving independent filters. However, as shown by the results obtained for *Setting2*, the quality of the results may depend on other tracking conditions such as the observation noise. In particular, the larger the observation noise, the more the filtering system is entitled to track jumps, thus affecting the quality of the outputs. In the following section, we keep performing analysis under *Setting2* and study the efficiency of the algorithm when varying parameters such as the number N of particles used to represent each target's distribution or the number K of targets to track.

7.6.3.3 Impact analysis of N

In the previous section, we saw that, even when considering a good function Φ, the performance of the filtering process is affected by the track jump phenomenon under high observation noise. In this section, our aim is to analyze whether this situation can be improved by modifying a parameter of the algorithm, and particularly the number N of particles used to represent each target distribution. To this end, we consider the tracking scenario described previously using *Rule50* and *Setting2*. Next, to perform our analysis, we consider the following values of N: 50, 200, 500, and 1000. For each of these values, we carry out 100 runs and report the obtained

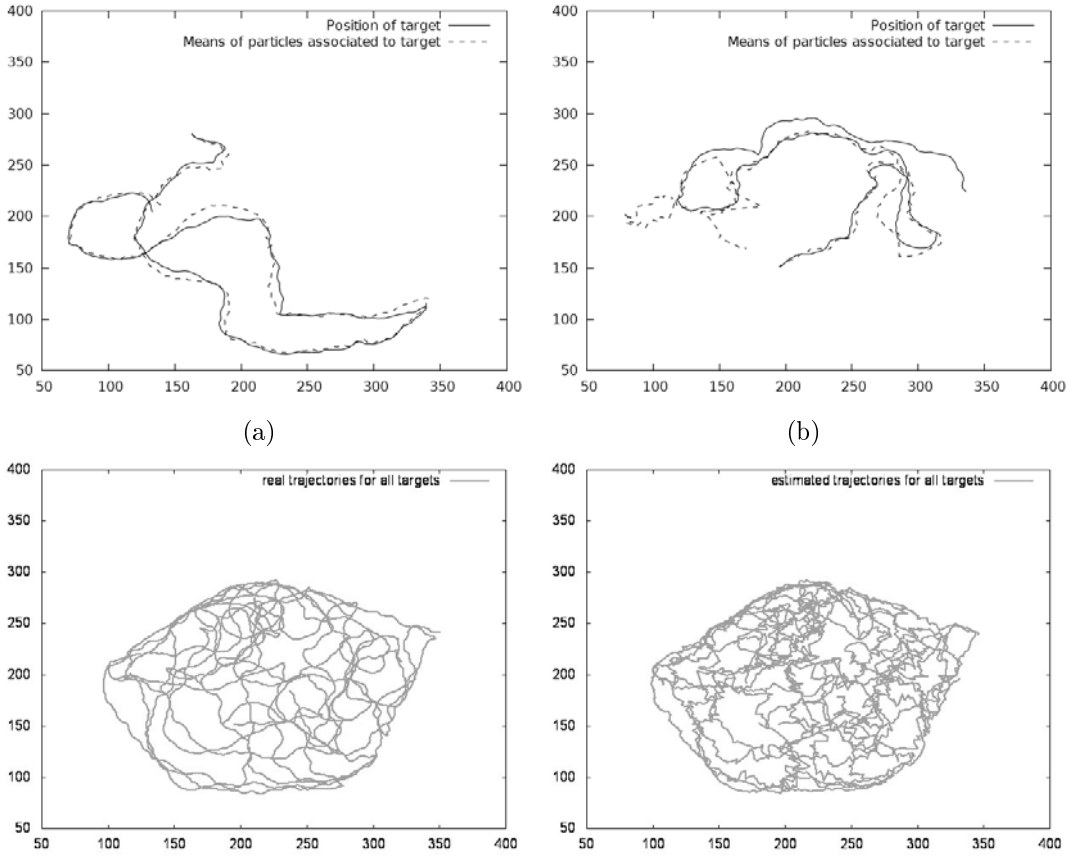
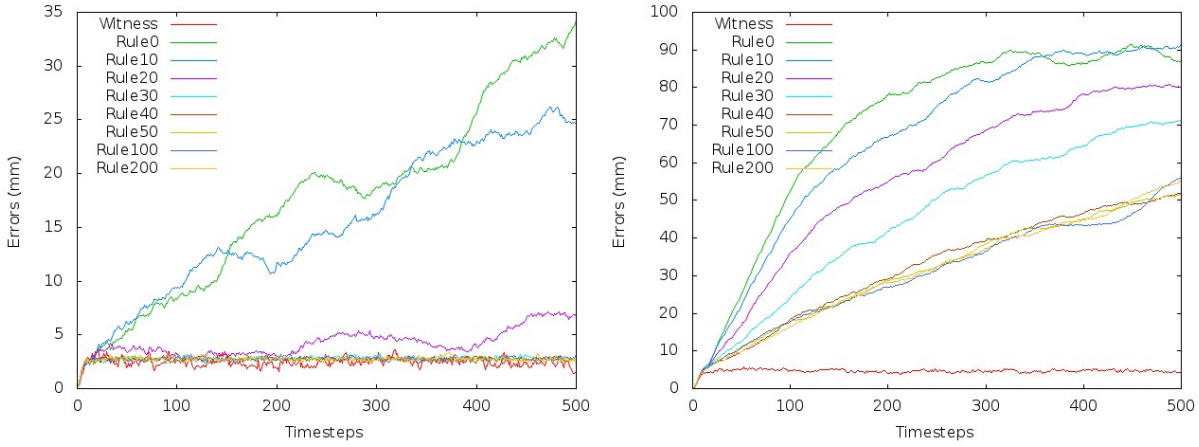
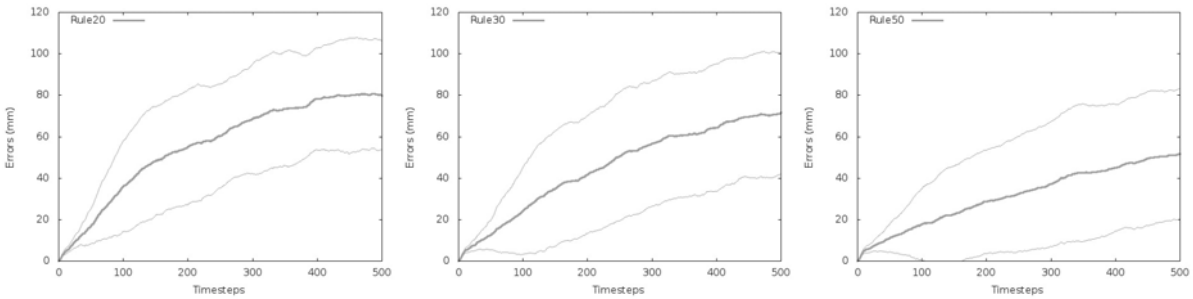


Figure 7.8: Insight of conducted experiments in Scenario 1: trajectories of targets and their associated filters for a given *Rule50* experiment under *Setting2*. (a) and (b) present trajectories of two randomly chosen targets. (c) illustrates the trajectories for all targets while (d) shows the trajectories resulting from the particle mean, at each time step, for all the (sub-)filters.



(a) Setting 1: $\sigma_x = \sigma_y = 7mm$ (b) Setting 2: $\sigma_x = \sigma_y = 15mm$

Figure 7.9: Impact analysis of Φ. Each graph depicts the average error distance $AvgErr_t$ obtained for each observation noise's setting.



(a) Rule20 (b) Rule30 (c) Rule50

Figure 7.10: Comparison of $AvgErr_t$ for different interaction rules under *Setting2*. Each graph presents the mean and associated standard deviation.

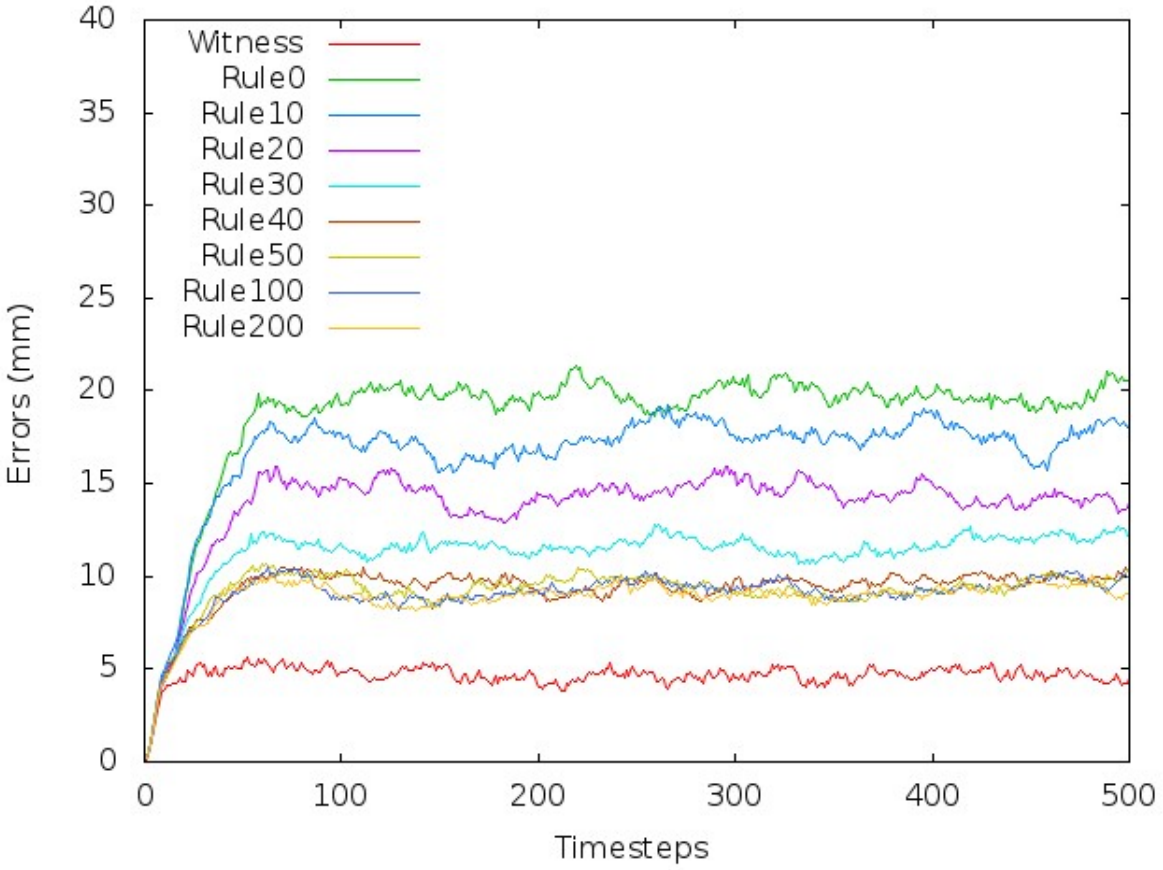


Figure 7.11: Average error between targets and their closest filter under *Setting2*

Table 7.2: Impact analysis of Φ - Tracking performances under *Setting2*. Here, a single representative has been used for characterizing each target both for interaction management and soft-gating. Each column contains mean and standard deviation over performed experiments.

Distance (in mm)	time=250			time=500			End process (sec)	Interactions
	Correct Tracks	Track Jumps	Lost Tracks	Correct Tracks	Track Jumps	Lost Tracks		
0	1.0 (±0.93)	3.48 (±1.26)	2.52 (±1.15)	0.72 (±0.85)	3.78 (±1.34)	2.5 (±1.28)	144.2 (±1.98)	0.0 (±0.0)
10	1.41 (±1.23)	3.27 (±1.38)	2.32 (±1.25)	0.74 (±0.78)	4.01 (±1.24)	2.25 (±1.13)	148.8 (±2.64)	0.77 (±0.21)
20	2.29 (±1.60)	3.07 (±1.38)	1.64 (±1.09)	1.41 (±1.32)	4.06 (±1.46)	1.53 (±1.11)	147.2 (±3.48)	0.96 (±0.22)
30	3.1 (±1.81)	2.69 (±1.67)	1.21 (±0.19)	1.95 (±1.61)	3.66 (±1.60)	1.39 (±1.10)	154.3 (±2.09)	1.08 (±0.22)
40	4.43 (±2.02)	1.69 (±1.64)	0.88 (±0.94)	3.08 (±1.82)	2.91 (±1.74)	1.01 (±0.93)	173.4 (±6.83)	2.10 (±0.35)
50	4.39 (±2.02)	1.62 (±1.65)	0.99 (±1.09)	3.13 (±2.02)	2.76 (±1.75)	1.11 (±1.10)	198.4 (±7.86)	3.13 (±0.44)
100	4.4 (±1.99)	1.6 (±1.69)	1.0 (±1.08)	3.32 (±2.06)	2.8 (±1.86)	0.88 (±0.95)	329.2 (±20.46)	10.37 (±0.86)
200	4.47 (±1.75)	1.59 (±1.49)	0.94 (±1.01)	3.17 (±1.82)	2.98 (±1.77)	0.85 (±0.88)	502.8 (±10.64)	20.62 (±0.2)

results.

Figure 7.12 shows the average errors obtained for each previously listed number of particles while Figure 7.13 depicts a direct comparison between some values of particle's number including the corresponding standard deviation. As we can observe, there is a slight increase of the overall performance of the filtering system as the number of particles increases. This is in accordance with the observations found in the literature [Arulampalam et al., 2002] and one can confidently assume that, with a very high number of particles, the quality of the result will get closer to the optimal solution one can achieve given the observation noise. Finally, the execution times obtained for all the considered values of N are reported in Table 7.3. It comes out that the execution time increases with the value of N. However, as we simply consider a single representative to characterize a target in this scenario, these execution times are of the same order of magnitude.

Table 7.3: Impact analysis of N - Execution time on Scenario 1.

Number of particles (N)	**Execution time** (in *sec*)
50	124.0 (±5.51)
200	176.74 (±6.37)
500	198.4 (±7.86)
1000	264.66 (±13.10)

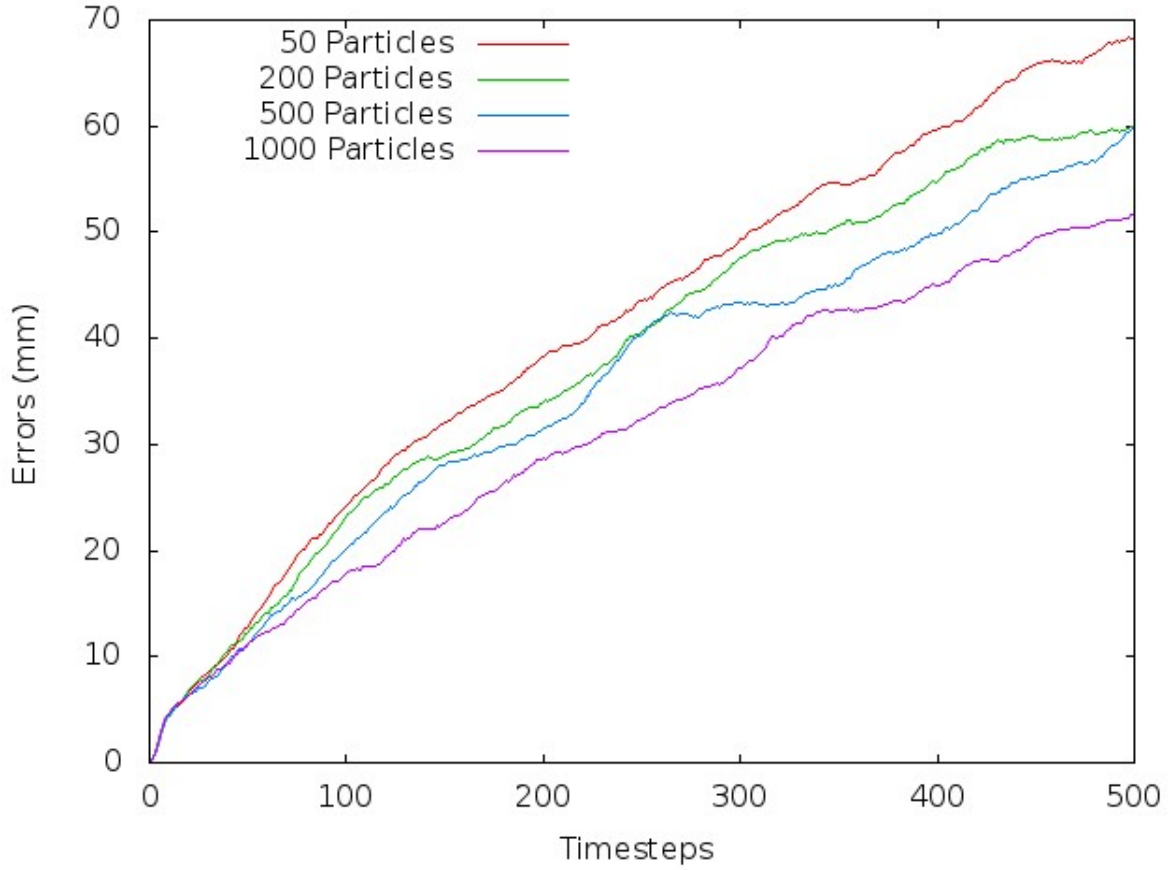
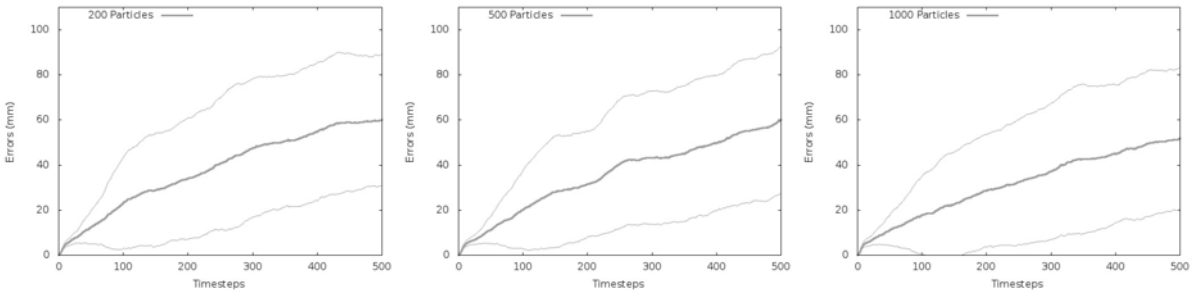


Figure 7.12: Impact analysis of N - Evaluation against the number of particles. The graph depicts the average error distance $AvgErr_t$ obtained under *Setting2* and *Rule50*.



(a) 200 Particles (b) 500 Particles (c) 1000 Particles

Figure 7.13: Comparison of $AvgErr_t$ for different numbers of particles under *Setting2* and *Rule50*. Each graph presents the mean and associated standard deviation.

7.6.3.4 Impact analysis of K

In this section, we study the impact of the density of the environment on the performance of the filtering process. To this end, we still keep the previous scenario with the observation noise defined by *Setting2* and the interaction rule *Rule50*. Subsequently, we consider the problem of tracking $K = 2$, 5, 7 and 9 targets respectively. For each of these setups, we perform 100 runs and present the results below.

Figure 7.14 illustrates the average errors obtained for each value of K as a function of time while Figure 7.15 exhibits these errors with their corresponding standard deviation. Additionally, Table 7.4 reports the execution times obtained for each value of K. It appears that the smaller the number of targets, the better the quality of the results. This can be explained by the small number of interactions occurring in less dense environments, and therefore, few opportunities for a (sub-)filter to mislead the track related to the target it was initially associated with. Putting it differently, in denser environments (9 targets), there are more interactions between targets and, due to the presence of large observation noise (*Setting2*), the filtering process suffers from track jump phenomena leading to an increase of the error. This is assessed by plotting the relative error of all the targets with respect to their closest filter which is illustrated in Figure 7.16.

Regarding the execution times, they increase with the number of targets in the environment. This is mainly because of two factors. The first one, which intervenes during the prediction step, is related to the number of interactions considered by the filtering process for computing the predicted distributions of all the targets. Indeed, as previously mentioned, the larger the number of targets, the larger the number of considered interactions (and thus, the number of simulations to be performed at the prediction step). The second factor intervenes during the correction step and is related to the data-association issue. As stated in the literature [Fortmann et al., 1983], the required time for resolving the data-association issue in the JPDAF framework exponentially increases with the number of targets. This second factor probably explains the difference observed between the execution time obtained for 7 targets (198.4 ($\pm$7.86)) and the one obtained for 9 targets (605.42 ($\pm$151.88)).

Table 7.4: Impact analysis of K - Execution time on Scenario 1.

Number of targets (K)	**Execution time** (in *sec*)
2	34.7 ($\pm$1.12)
5	63.49 ($\pm$3.29)
7	198.4 ($\pm$7.86)
9	605.42 ($\pm$151.88)

7.6.3.5 Summary

Throughout Section 7.6.3, we use a single representative (computed as the mean of the particles) for approximating the information required by targets to correctly interact with each other in the environment. The scenario we designed and the experiments performed show promising results. They highlighted the fact that each individual (sub-)filter takes into account the presence of other targets within the environment when updating the posterior distribution regarding the state of the associated target. However, in several real-world situations, a single representative is

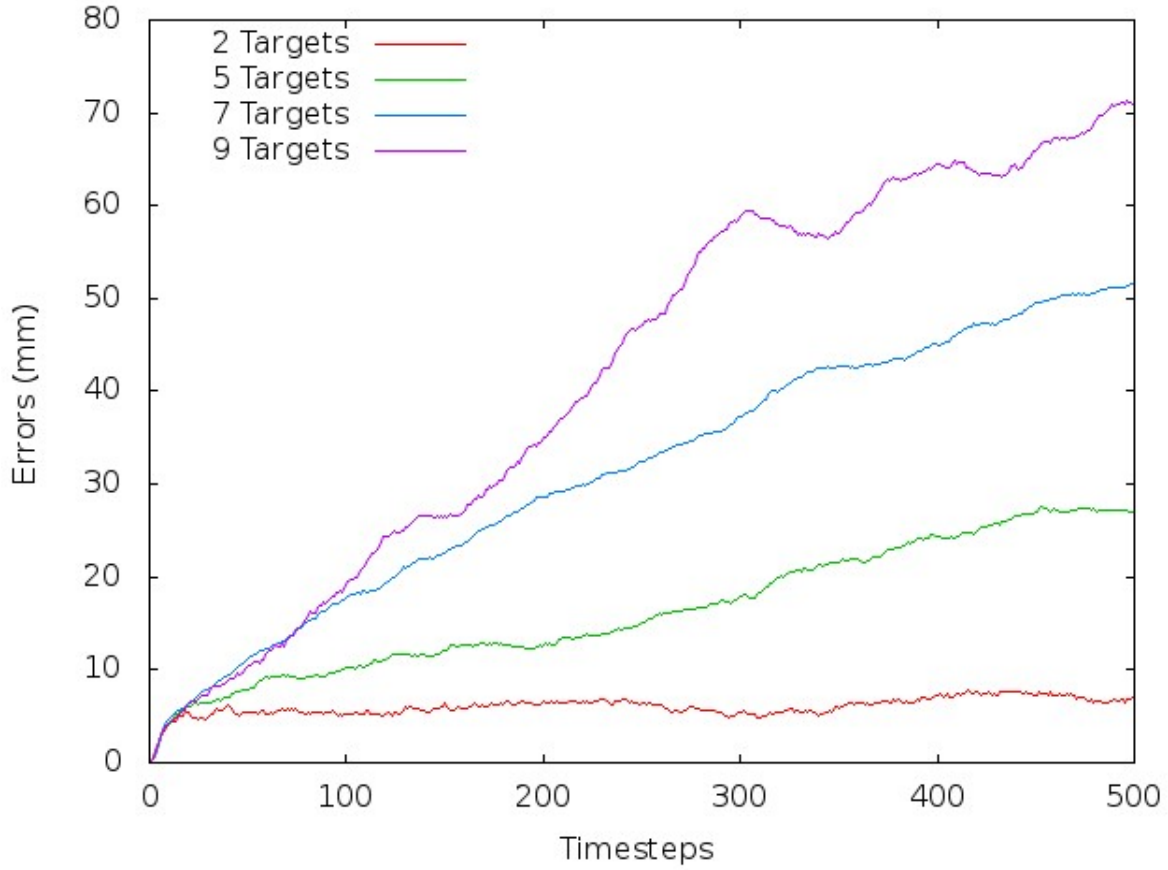
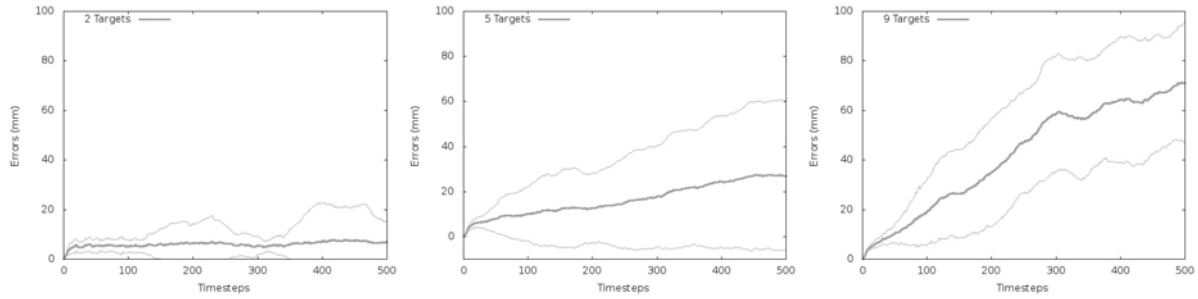


Figure 7.14: Impact analysis of K. The graph depicts the average error distance $AvgErr_t$ obtained under *Setting2* and *Rule50*.



(a) 2 Targets (b) 5 Targets (c) 9 Targets

Figure 7.15: Comparison of $AvgErr_t$ for different numbers of targets under *Setting2* and *Rule50*. Each graph presents the mean and associated standard deviation.

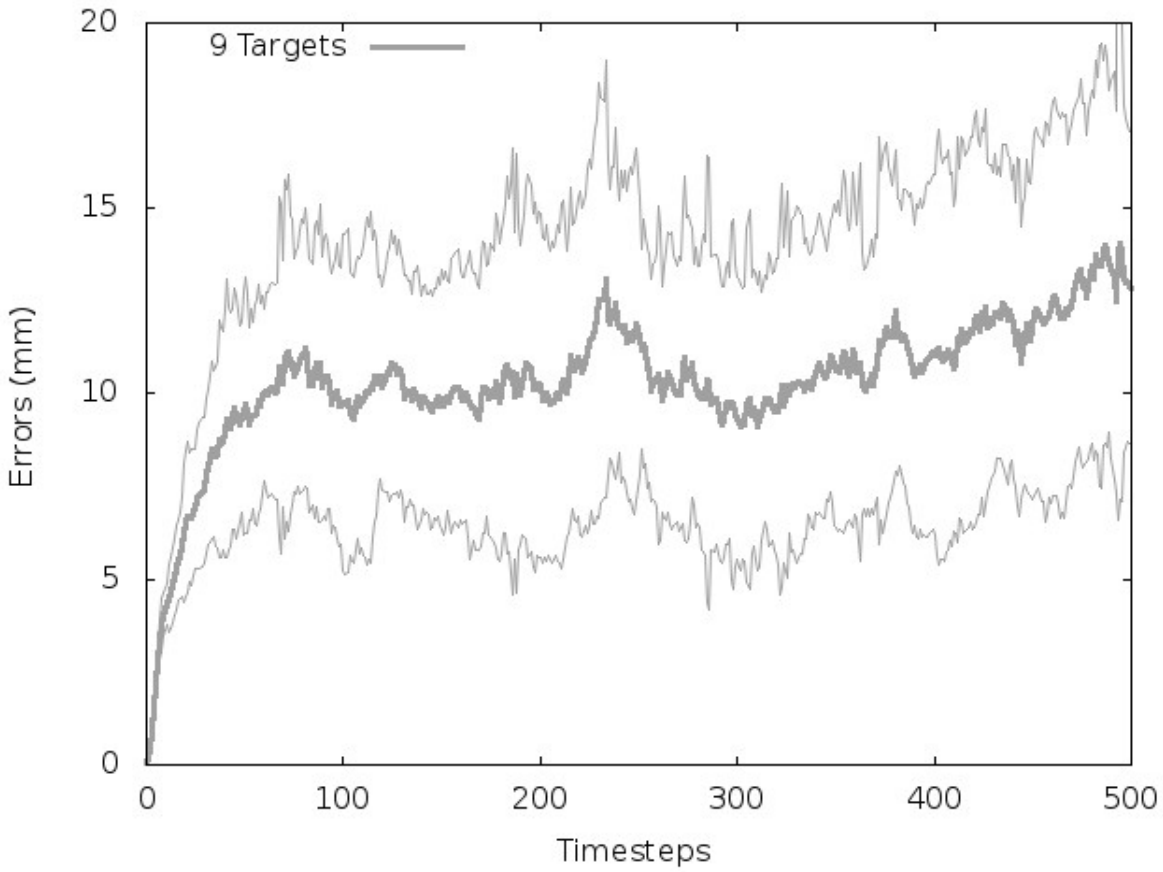


Figure 7.16: Average error to closest filter under *Setting2* and *Rule50* for 9 targets. The plot presents the mean and associated standard deviation.

not always sufficient for capturing the essence of a given target-related information, particularly when its dynamics presents strong non-linearity as for example with multi-modal distributions (see Section 7.4.3.3). The next section is devoted to providing a clear insight, on a basic example, of the advantages of considering multiple representatives when tracking interacting targets whose distributions may be multi-modal.

7.6.4 Multiple-representative-based evaluations: an illustrative scenario

The objective of this section is to study, on a simple example, the advantages of considering multiple representatives versus a single representative for tracking interacting targets with strongly non-linear distributions. The environment designed in the previous section is not suitable for this study since it is assumed that the environment is fully under sensory coverage, and thus, the target distributions are hardly multi-modal. Thereby, we need to design for this evaluation an environment containing areas which are not under sensory coverage. This section is organized as follows. Section 7.6.4.1 describes the experimental setup under which the study is performed. Then, Section 7.6.4.2 and Section 7.6.4.3 focus on the impact analysis of using multiple representatives at the prediction and the correction steps of the filtering process respectively.

7.6.4.1 Experimental setup

This section focuses on describing the conditions under which the above-mentioned study is conducted.

Environment

The environment under consideration is depicted in Figure 7.17 and it has a Y-shape. Moreover, it contains an area which is not under sensory coverage. This non-covered area is represented in Figure 7.17 by a hatched rectangle.

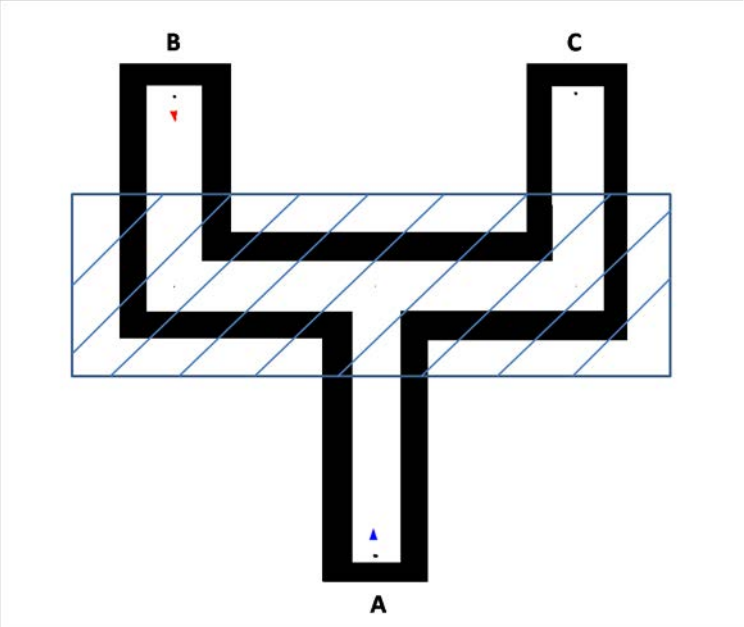


Figure 7.17: Experimental environment of Scenario 2. The hatched rectangle represents an area which is not covered by the sensors.

Behavioral model

In this study, we consider tracking two targets within the environment described above. Like in Section 7.6.3, the targets are equipped with the *containment* and *separation* behaviors. Moreover, each target is provided with the *seek* behavior making thus possible, for the target, to choose its final destination among a set of preassigned goals. More specifically, at the beginning of the simulation, the blue target — which starts from A — randomly chooses, between destinations B and C, the goal it wishes to reach. Similarly, the red target — which starts from B — has the possibility to choose its final destination between A and C. Also, in order to prevent direct trajectories, each target has, at each time step, a probability of $1/3$ to wander while moving towards his destination.

On this basis, each target k is characterized by the state

$$\mathbf{x}_{k,t} = [x_{k,t}, \dot{x}_{k,t}, y_{k,t}, \dot{y}_{k,t}, g_{k,t}],$$

where $(x_{k,t}, y_{k,t})$ and $(\dot{x}_{k,t}, \dot{y}_{k,t})$ are defined as previously in Section 7.6.3.1, and $g_{k,t} \in \{A, B, C\}$ stands for the goal (destination) of target k at time t. In this scenario, once a goal has been chosen by a target, the latter cannot change it.

Observation model

Like in Section 7.6.3, the sensor network provides a noisy estimate of the location of a target when it is detected. Here, the sensor network is characterized by the observation function

$$h(\mathbf{x}_{k,t}, \mathbf{v}_t) = B\mathbf{x}_{k,t} + \mathbf{v}_t,$$

where B is the observation matrix defined as

$$B = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{bmatrix},$$

and $(\mathbf{v}_t)_{t>0}$ is a bi-dimensional zero-mean Gaussian noise process characterized by a diagonal covariance matrix with a standard deviation of $\sigma_x = 15mm$ and $\sigma_y = 15mm$ for the x and y coordinates respectively (similar to *Setting2* introduced previously). Moreover, like in Section 7.6.3, we assume that the sensor network is characterized by:

- a detection probability P_D of 0.95,
- a false alarm rate λ_{FT} of 0.8.

Initial belief

The initial belief regarding the location and the velocities of the targets being tracked is similar to what has been described in Section 7.6.3. Moreover, regarding the goal attribute, the filtering process is not aware of the choices made by the targets and should rather infer them. As the initial belief characterizing the goal attribute, we consider the uniform distribution over the possible choices.

Additional parameters

As previously mentioned in this document (see Section 7.5.5), representatives may intervene at three levels within the proposed algorithm: (*i*) at the prediction step when computing the predicted distribution of each target, (*ii*) at the correction step when using the gating procedure (multiple-representative-based gating) during the resolution of the data-association issues and (*iii*) when reducing the number of particles while maintaining a behavioral diversity. The computation of these representatives (in case a target may be characterized by multiple representatives) requires the execution of a clustering procedure (see Section 7.4.4.2). In this section, we opt for a density-based clustering approach (clearing-based clustering).

Furthermore, when not explicitly specified, the system parameters are set as follows:

- the number K of targets being tracked is 2;

- the number N of particles used to represent the targets' distributions is set to 500;
- the threshold N_T, under which the resampling step is performed, is set to $0.75N$;
- the radius r_h for managing smooth transitions at the edge of covered/non-covered areas is set to $2mm$;
- the threshold η for the gating procedure is set to $40mm$;
- $\Phi(\cdot,\cdot)$ is defined on the basis of *Rule50* introduced in Section 7.6.3.2;
- the length T of the temporal window of the filtering process is set to 500;
- the reduction policy used is the selection-based reduction policy (see Section 7.5.3.2);
- the single-representative-based aggregator used is a combination of the MEAN operator for the continuous attribute and the MODE operator for the categorical attribute;
- the multiple-representative-based aggregator used is based on the clearing-based clustering characterized by the affinity function F_{aff} defined as

$$F_{aff}(\mathbf{x}_k^i, \mathbf{x}_k^j) = \begin{cases} \frac{1}{\alpha D_E^2(\mathbf{x}_k^i,\mathbf{x}_k^j)+\beta D_G^2(\mathbf{x}_k^i,\mathbf{x}_k^j)} & \text{if } D_E(\mathbf{x}_k^i, \mathbf{x}_k^j) \neq 0 \text{ or } D_G(\mathbf{x}_k^i, \mathbf{x}_k^j) \neq 0 \ , \\ \infty & \text{otherwise,} \end{cases} \tag{7.49}$$

  where $D_E(\cdot,\cdot)$ stands for the Euclidean distance (in mm) between the states in parameters and $D_G(\cdot,\cdot)$ is a distance we defined for distinguishing between the goals of the targets and is expressed as

$$D_G(\mathbf{x}_k^i, \mathbf{x}_k^j) = \begin{cases} 1 & \text{if } g_k^i \neq g_k^j, \\ 0 & \text{otherwise.} \end{cases}$$

  In equation 7.49, α and β are parameters defining the importance of a given distance with respect to the other one. For this scenario, α and β are respectively equal to $2.7 \cdot 10^{-4}$ and 0.35. Also, the affinity threshold ϵ_A is preset to $\frac{2}{3}$. In this way, the maximum Euclidean distance between particles having the same goal within a given cluster is nearly $50mm$.[18] Also, particles having different goals can still be part of the same cluster if their distance is lower than $35mm$.[19] Finally, $MaxSize$ is preset to 1.

In the following sections, we separately assess, on this illustrative scenario, the impact of multiple representatives at levels (*i*) and (*ii*).

7.6.4.2 Impact of multiple representatives at the prediction step

The purpose of this section is to evaluate the impact of considering multiple representatives for computing the predicted distributions of interacting targets in situations in which the environ-

[18] The maximum Euclidean distance is obtained as $\sqrt{\frac{\epsilon_A}{\alpha}}$.

[19] The maximum Euclidean distance between particles with different goals is obtained as $\sqrt{\frac{\epsilon_A-\beta}{\alpha}}$.

ment includes areas which are not under sensory coverage. Here, we assume that the gating performed at the correction step of the filtering process is done on the basis of the classical soft-gating procedure. To carry out our evaluation, we run our algorithm on the scenario defined in Section 7.6.4.1 using the following two configurations when computing the predicted distributions of the targets being tracked:

- target representatives are computed using a single-representative-based aggregator,
- target representatives are computed using a multiple-representative-based aggregator.

For assessing the performance under each of these configurations, we simply consider $AvgErr_t$, the "*average error distance*" criterion defined in Section 7.6.2, as there is a low probability of track jump phenomena since we are tracking only two targets with distinct behaviors. For this assessment, we consider two variants of the estimated state $\hat{\mathbf{x}}_{k,t}$ of target k:

- *Variant 1*: the mean of the particles within the corresponding (sub-)filter;
- *Variant 2*: the representative whose state is the closest to the true target state. We acknowledge the bias introduced by this estimator but the latter has the advantage of evaluating the filtering process from a point of view satisfying the behavioral diversity maintenance.

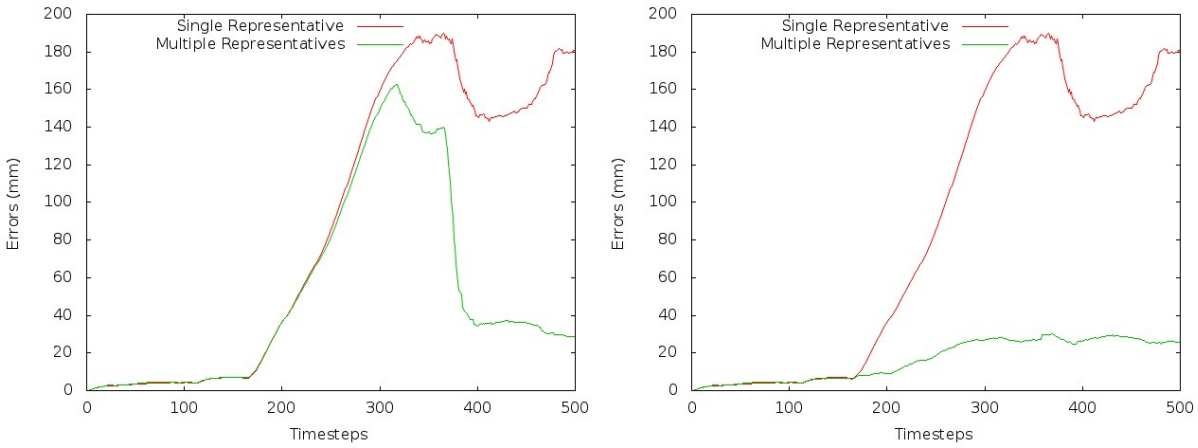
Finally, for each of these configurations, we perform 100 runs and the results are reported below.

Figure 7.18a and Figure 7.18b compare the errors obtained in both configurations when using estimates defined on the basis of *Variant 1* and *Variant 2* respectively. As we can observe, the multiple-representative-based configuration outperforms the single-representative-based configuration whatever the estimation variant used. Based on these graphs, the difference between both configurations roughly appears at time $t = 170$ when the blue target effectively follows the direction corresponding to the destination choice it made between B and C. From that time, using the single-representative-based approach, the filter associated with the red target is provided with an inaccurate information regarding the blue target's state. Indeed, whatever the real choice of the blue target, the mean of the blue target's particles is stuck in the center of the non-observed corridor and therefore, the red target's particles will mostly evolve in the first half of the corridor as if they were not subject to any interaction (which is wrong if the blue target's final destination is B) as depicted in Figure 7.19c. On the other hand, with the multiple-representative-based approach, the red-target-related filter is able to take into account both hypotheses regarding the choice made by the blue target and evolves accordingly (see Figure 7.19d). Furthermore, at time $t = 385$, when both targets are not within the non-covered area anymore, we observe that the multiple-representative-based approach manages to maintain the tracking errors low while the single-representative-based approach cannot. This means that the distributions computed on the basis of multi-representative-approach are more robust and they efficiently represent what is going on in the environment.

A similar comparison could be made regarding the computation of the estimated states of the tracked targets. While the simple mean of the particle set appears to be inaccurate for estimating the true state of a given target when its distribution is multi-modal, considering the distribution's representative states offers a much better alternative (see Figure 7.18b) even if,

as in our context, it is not always possible to say in advance (without knowing the true state of the target) which representative is the best. Here, the objective is to maintain all possible hypotheses (representatives) available until we have enough observation data to determine which one(s) match with the reality. Therefore, evaluating the performance of the filter posteriorly (the closest representative to the true target state) is not so biased and perfectly meets our needs.

Through the remainder of this chapter, in order to assess the performance of the proposed algorithm, the filter estimates are computed on the basis of *Variant 2*. Also, be aware that *Variant 2* is equivalent to *Variant 1* in case of single-representative-based approaches.



(a) Variant 1: particle mean

(b) Variant 2: closest representative

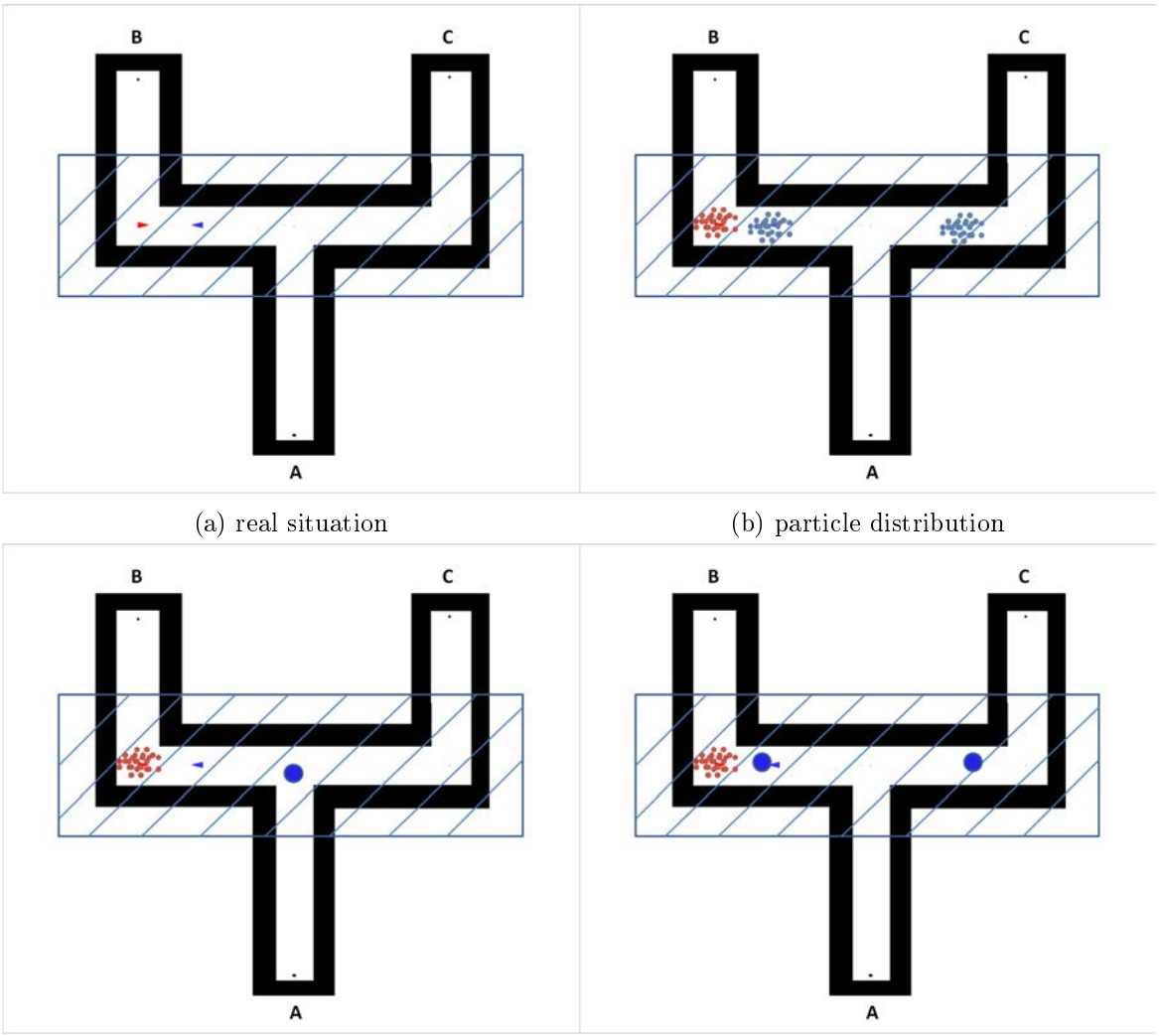
Figure 7.18: Multiple versus single representative(s) - prediction step: Each graph depicts the average error distance $AvgErr_t$ obtained for each estimation variant.

7.6.4.3 Impact of multiple representatives at correction step

In the previous section, we have shown that, in cases of multi-modal distributions, considering multiple representatives at prediction step of the filtering process results in improving the performance of the process when tracking interacting targets. In this section, we are interested in studying the impact of using multiple representatives at the correction step of the filtering process and more particularly during the gating procedure. To this end, we consider the scenario defined in Section 7.6.4.1 and apply our algorithm using a multiple-representative-based aggregator when computing the predicted distributions of the targets at the prediction step. However, unlike Section 7.6.4.2, we assume here that the gating procedure is performed on the basis of the multiple-representative-based soft-gating introduced in Section 7.5.4.

In order to compute representatives for the gating procedure, we rely on the affinity function defined in Equation 7.49 with the particularity that β is set to 0 since the internal goal of a target does not affect its current observation (by the sensors) given its location. Finally, 100 runs are conducted and the resulting performance (based on the closest target representative) is shown in Figure 7.20.

As one can observe, the multiple-representative-based soft-gating yields better performances than the classical soft-gating, even if, in this particular scenario, both approaches cannot be characterized by a real statistical difference given the variances obtained (although the standard



(a) real situation (b) particle distribution

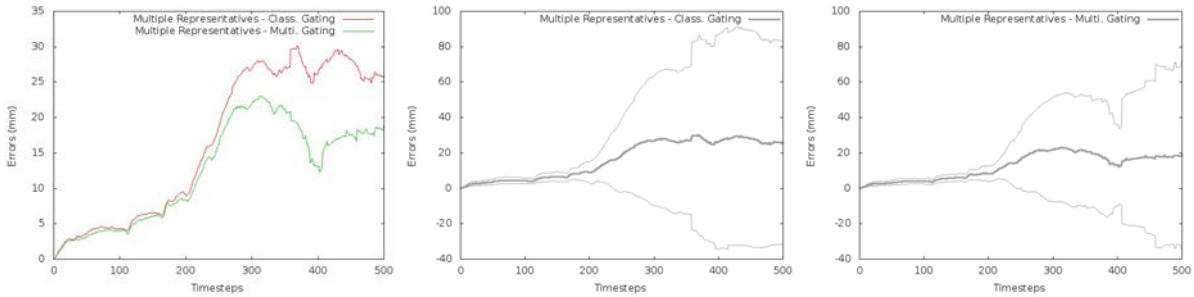
(c) single representative (d) multiple representatives

Figure 7.19: Insight of single-representative versus multiple-representative based approaches. (a) and (b) capture, at some point in time, the real situation of both targets and the distribution represented by the related particles. (c) (resp. (d)) focuses on the red target and shows which information from the blue target is considered for updating the underlying particle states in case of single(resp. multiple)-representative-based approach.

deviation associated with the multiple-representative-based soft-gating is smaller). This is mainly because of the following two reasons: (*i*) the number of targets is low and (*ii*) the targets have distinct behaviors. Indeed, the system may face observation dilemma only a quarter of time (when both targets chose point C as final destination). Also, as previously mentioned in Section 7.5.4, clutters also contribute to the degradation of the performance in case of classical soft-gating.

7.6.4.4 Summary

Throughout Section 7.6.4, we have demonstrated on a simple case the importance of considering multiple representatives both at the prediction and correction steps of the filtering process for



(a) Comparison (b) Classical soft-gating (c) Mult. rep. soft-gating

Figure 7.20: Multiple versus single representative(s) - correction step (gating): (a) shows a comparison of the error $AvgErr_t$ obtained from the two approaches. (b) and (c) depict the resulting standard deviation for both cases.

improving the performance when tracking interacting targets in environments containing non-covered areas. In the next section, we continue to investigate the performance of our algorithm, but this time, on a more complex scenario.

7.6.5 Multiple-representative-based evaluations: a complex scenario

In the previous section, we have presented an insight of the multiple-representative-based approach on a simple example. Here, we are interested in assessing the performance of the proposed algorithm in more complex situations. This section is organized as follows. Section 7.6.5.1 describes the experimental setup under which the assessment is performed. Then, from Section 7.6.5.2 to Section 7.6.5.7, we analyze the impact of various parameters on the performance of the algorithm.

7.6.5.1 Experimental setup

This section focuses on describing the conditions under which the above-mentioned assessment is performed. In particular, we present the environment, the behavioral model, the observation model, the initial belief as well as the parameters used.

Environment

The environment under consideration is illustrated in Figure 7.21. It includes an area which is not under sensory coverage (the hatched area in Figure 7.21). Moreover, the environment is characterized by six predefined points of interest (points A to F in Figure 7.21) for the targets evolving therein. Some of these points of interest are localized within the area which is not under sensory coverage (C, D and E).

Behavioral model

Like in Section 7.6.4.1, each target evolving in the above-described environment is equipped with the *containment*, *separation* and *seek* steering behaviors. Moreover, the navigational policy

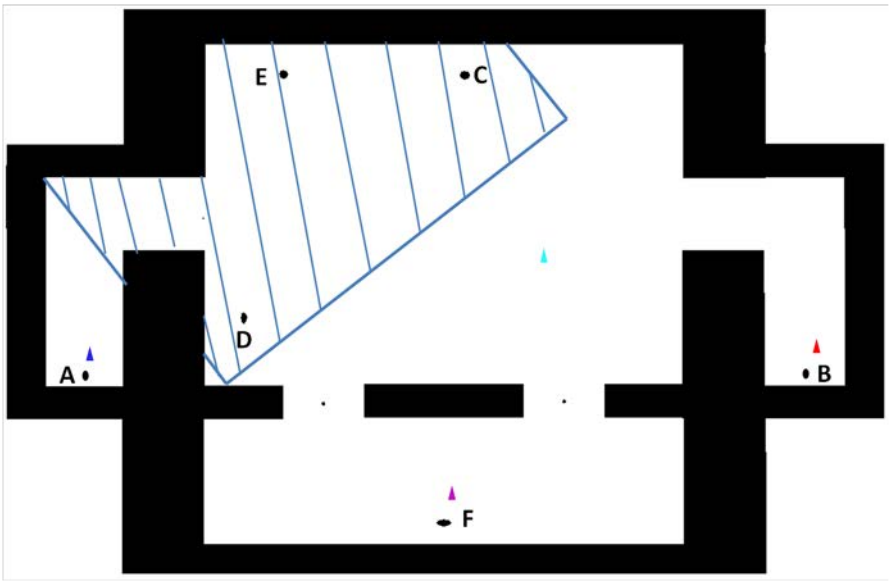


Figure 7.21: Experimental environment of Scenario 3. The hatched area is not under sensory coverage.

described in Table 7.5 is encoded within each target and dictates how to probabilistically evolve within the environment after reaching a specific destination. This policy is designed to roughly reproduce complex trajectories that one may encounter in the virtual train station introduced in Chapter 5 without the high-level reasoning characterizing each virtual agent. In addition, each target has, at each time step, a probability of $\frac{1}{3}$ for wandering while moving towards its current destination.

On this basis, each target k is characterized by the state

$$\mathbf{x}_{k,t} = [x_{k,t}, \dot{x}_{k,t}, y_{k,t}, \dot{y}_{k,t}, g_{k,t}]$$

defined as previously in Section 7.6.4.1.

Table 7.5: Scenario 3 - Navigational Policy encoded within each tracked target.

	Next Destination					
Current Destination	A	B	C	D	E	F
A	0.04	0	0.24	0.24	0.24	0.24
B	0	0.04	0.24	0.24	0.24	0.24
C	0	0	0.6	0.2	0.2	0
D	0	0	0.4	0.5	0.1	0
E	0	0	0	0	0.6	0.4
F	0.3	0.3	0	0.36	0	0.04

Observation model

The observation model used for this study is identical to the one described in Section 7.6.4.1.

Initial belief

As for the observation model, the initial belief used in this section is identical to the one described in Section 7.6.4.1.

Additional parameters

When not explicitly specified, the system parameters are set according to the description made in Section 7.6.4.1. Recalling from that description, it is assumed, by default, that:

- $\Phi(\cdot,\cdot)$ is defined on the basis of *Rule50* introduced in Section 7.6.3.2;
- the reduction policy used is the selection-based reduction policy (see Section 7.5.3.2);
- the single-representative-based aggregator used is a combination of the MEAN operator for the continuous attribute and the MODE operator for the categorical attribute;
- the multiple-representative-based aggregator used is based on the clearing-based clustering characterized by the affinity function F_{aff} defined in Equation 7.49.

The next section is dedicated to the evaluation of various configurations of the proposed algorithm on this setup.

7.6.5.2 Single *versus* multiple representative(s)

In this section, we consider the problem of tracking $K = 4$ targets within the above-described environment for a period of $T = 500$ frames. Our aim consists in evaluating the tracking performance for different configurations of the proposed algorithm. To this end, we consider the configurations reported in Table 7.6.Furthermore, the assessment of the performance is based on $AvgErr_t$ — the "*average error distance*" criterion defined in Section 7.6.2 — as well as the execution time. Finally, for each considered configuration, 100 runs are performed and the results are reported below.

Figure 7.22 describes the average error $AvgErr_t$ obtained for each configuration while Figure 7.23 shows the standard deviation of the performance for some of these configurations. More specifically, Figure 7.22a puts together *Configuration 1* and *Configuration 2*, as they rely on a single-representative-based approach at the prediction step, in front of the classical JPDAF algorithm (*Configurations 0*). As it can be noticed, *Configuration 1* and *Configuration 2* perform slightly better than the classical JPDAF algorithm and the difference can be viewed as the minimum gain resulting from the integration of our interaction model within the whole JPDA paradigm. Also, the difference observed between *Configuration 1* and *Configuration 2*, although small, clearly illustrates the advantage of the multiple-representative-based soft-gating with respect to the classical soft-gating. Regarding Figure 7.22b, it compares the performances resulting

Table 7.6: Various configurations for evaluation purposes

	Interactions	**Parameters Representative-based approach**		
Configurations	$\Phi(\cdot,\cdot)$	**Prediction step (Target dynamics)**	**Correction step (Gating)**	**Correction step (Reduction)**
Configuration 0 (JPDAF)	*Rule0*	-	Single representative	-
Configuration 1	*Rule50*	Single representative	Single representative	-
Configuration 2	*Rule50*	Single representative	Multiple representatives	-
Configuration 3	*Rule50*	Multiple representatives	Single representative	Selection-based policy
Configuration 4	*Rule50*	Multiple representatives	Multiple representatives	Selection-based policy

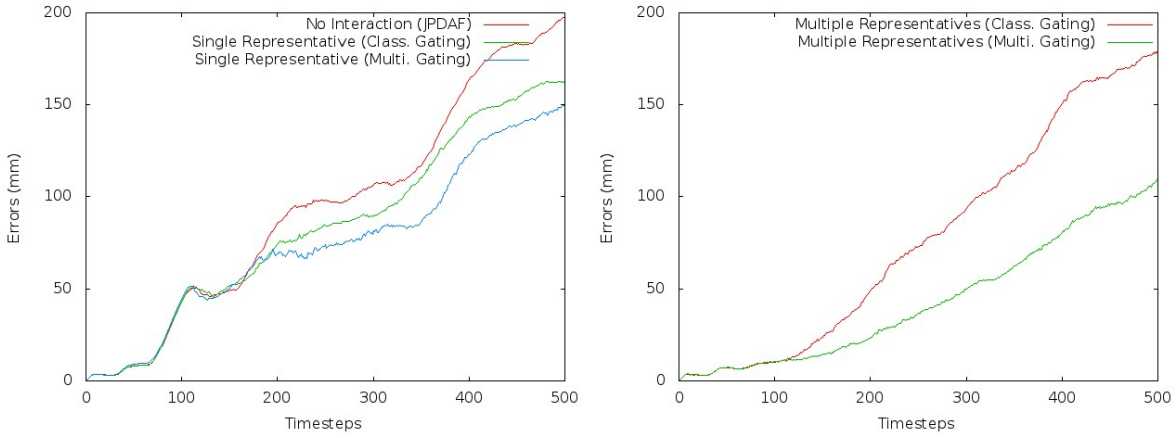
from configurations (*Configuration 3* and *Configuration 4*) involving multiple representatives at the prediction step. With respect to *Configuration 1* and *Configuration 2*, their outcomes are much more better. However, these performances highly depend on the gating procedure that has been used.

Having a close look at Figure 7.22, we can see that all the curves are globally similar from time step 0 until time step 60 where some targets start reflecting the choices they made within the non-covered area. From that point, because information reflecting target behaviors are roughly encapsulated within the mean of the distribution, filtering processes relying on a single representative at the prediction step consider that targets still behave independently and only take into account interactions between them around time step 160 where we can observe the difference with the classical JPDAF algorithm. However, considering multiple representatives per targets at the prediction step, the system can efficiently and early deal with the targets' interactions which normally occur in the environment around time step 90 on average.

The importance of multiple-representative-based soft-gating can be observed on Figure 7.22b around time step 120. Indeed, it corresponds, on average, to the period in which, for the first time, a target previously in the non-covered area re-enters the area under sensory coverage. As illustrated in Figure 7.24, given the dispersion of the underlying target's particles, the classical gating procedure usually fails to consider the received atomic observation as potential candidate[20] for the target, therefore resulting in a fast degradation of the performance.

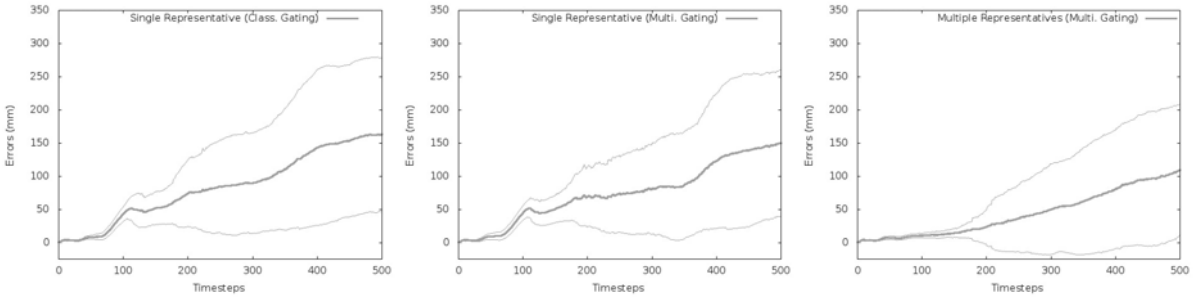
From the plots in Figure 7.22, it is clear that the usage of multiple representatives at the prediction step (interaction management) as well as the correction step (gating procedure) yields the best performance of the system. Nevertheless, the scenario on which our system is evaluated presents another layer of complexity which is not negligible: **the re-identification problem** which consists in determining the ID of the tracked target when it reappears. Our system can help re-identify a target exiting a non-covered area and can deal with this issue efficiently when there are **no** representatives from **other** targets exiting the area at the same time (see Figure 7.25a). Otherwise, the best our system can provide is a subset of targets (with corresponding probabilities) which may correspond to the one exiting the area (see Figure 7.25b) and a lack

[20] An atomic observation is a potential candidate for a target if the probability that it originates from that target is assumed non-zero.



(a) (b)

Figure 7.22: Comparison of the performances from different configurations. (a) confronts together configurations relying on a single representative at the prediction step with classical JPDAF while (b) compares configurations involving multiple-representatives at the prediction step.

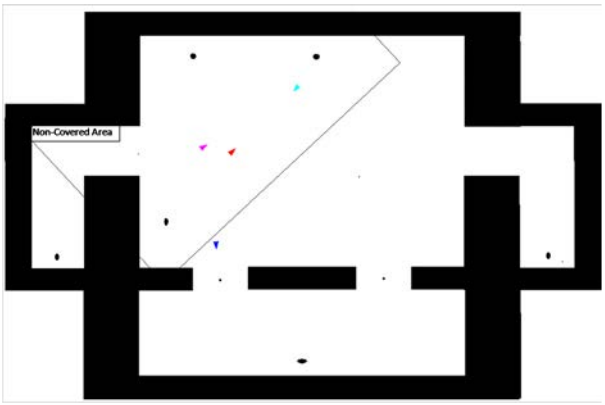


(a) *Configuration 1* (b) *Configuration 2* (c) *Configuration 4*

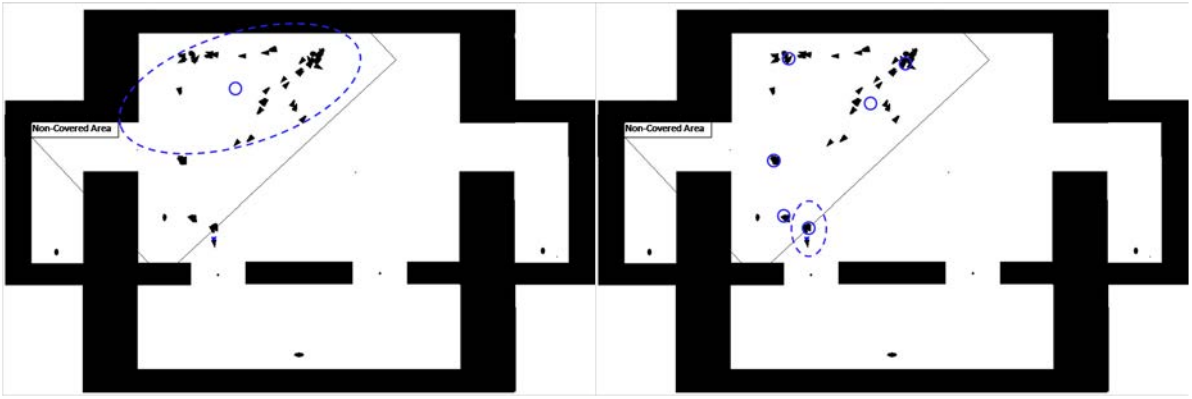
Figure 7.23: Configuration performance evaluation - standard deviations.

of re-identification — as far as the filtering process is carried on — has a significant impact on the metric ($AvgErr_t$) used for assessing the tracking performance. This explains the subsequent degradation of the green curve (*Configuration 4*) in Figure 7.22b and the standard deviation observed in Figure 7.23c. However, as illustrated in Figure 7.26, when there is enough information (observation data) for distinguishing targets, our system demonstrates interesting features. Despite this re-identification issue, our system performs pretty well half of the time as demonstrated in Figure 7.27 where we plot the quartiles resulting from the undertaken experiments.

Regarding the execution times, Table 7.7 describes the outcomes for each configuration. As it can be expected and because they are performing more simulations, configurations involving multiple representatives at the prediction step require more computational resources with respect to the classical JPDAF algorithm and/or configurations involving a single representative. However, these numerical values have been obtained without constraining the maximal number of representatives per target at each time step, which can easily increase within a non-covered area. Figure 7.28 focuses on *Configuration 4* (multiple Representatives used at both the prediction and the correction steps) and illustrates the performance obtained when constraining the maximal number of representatives per target to 4. As it can be observed, there is no significant



(a)

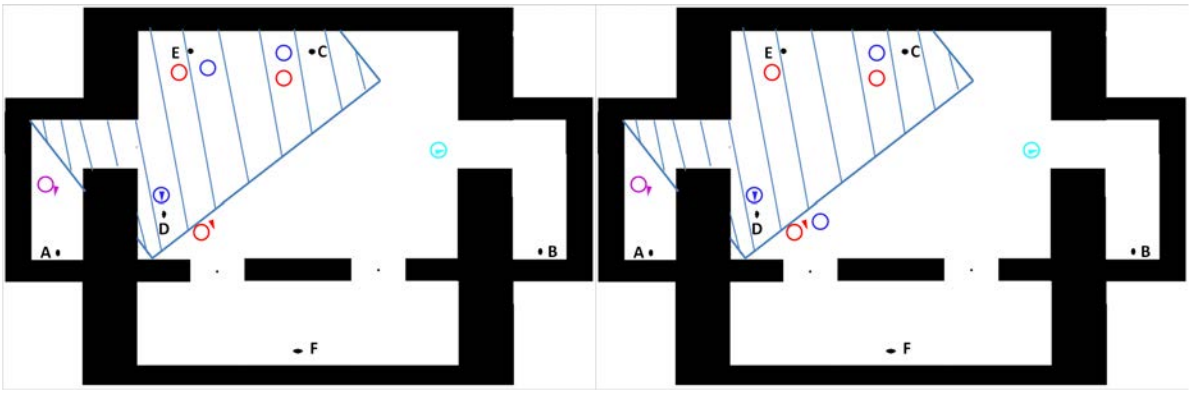


(b) classical soft-gating (c) mult. repr. based soft-gating

Figure 7.24: Classical soft-gating *vs* multiple-representative-based soft-gating. In (a), the blue target is exiting the non-covered area and an atomic observation data is provided to the filtering process. Using the classical soft-gating, the system fails to consider the atomic observation as potential candidate for the target (Fig. (b)) while it is not the case with multiple-representative-based approach (Fig. (c)).

difference, in terms of performance quality, with the non-constraining case and here, the average execution time obtained on 100 runs is 292.16($\pm$150.58) which is about half the one obtained previously (430.50($\pm$277.80)). Finally, Figure 7.29 shows, as a function of time, the average goal similarity ($AvgSim_t$) within the filtering process under *Configuration 4*. As illustrated, most of the time, the filtering process manages to correctly infer the goal of the underlying targets despite the presence of non-covered areas.

In summary, we have evaluated in this section different configurations of the proposed algorithm on a complex and challenging scenario. The results obtained show the benefit of handling target interactions in an environment with non-covered areas (or more generally, in an environment inducing multi-modal distributions) using multiple representatives rather than a single representative. Also, they show that, in such environments, multiple representatives are useful at correction level for efficiently dealing with data association issues. From Section 7.6.5.3 to Section 7.6.5.7, we focus on *Configuration 4* (multiple representatives are used at both the prediction and the correction steps) and study the performance of the algorithm when varying some parameters such as the clustering methodology, the reduction policy, the number of particles



(a) (b)

Figure 7.25: Illustration of re-identification issues. The colored circles depict the representatives of corresponding targets. The red target is exiting the non-covered area. In (a), there is no doubt about the identification of the exiting target and our system will subsequently eliminate other hypotheses (representatives) related to the red target. In (b), there is no certainty regarding the identity of the exiting target. The blue and the red targets are potential candidates. And since we are in a JPDAF paradigm, the survival of the each of these representatives, as the filtering process is carried on, will depend on its relative weight with respect to other hypotheses of the corresponding target.

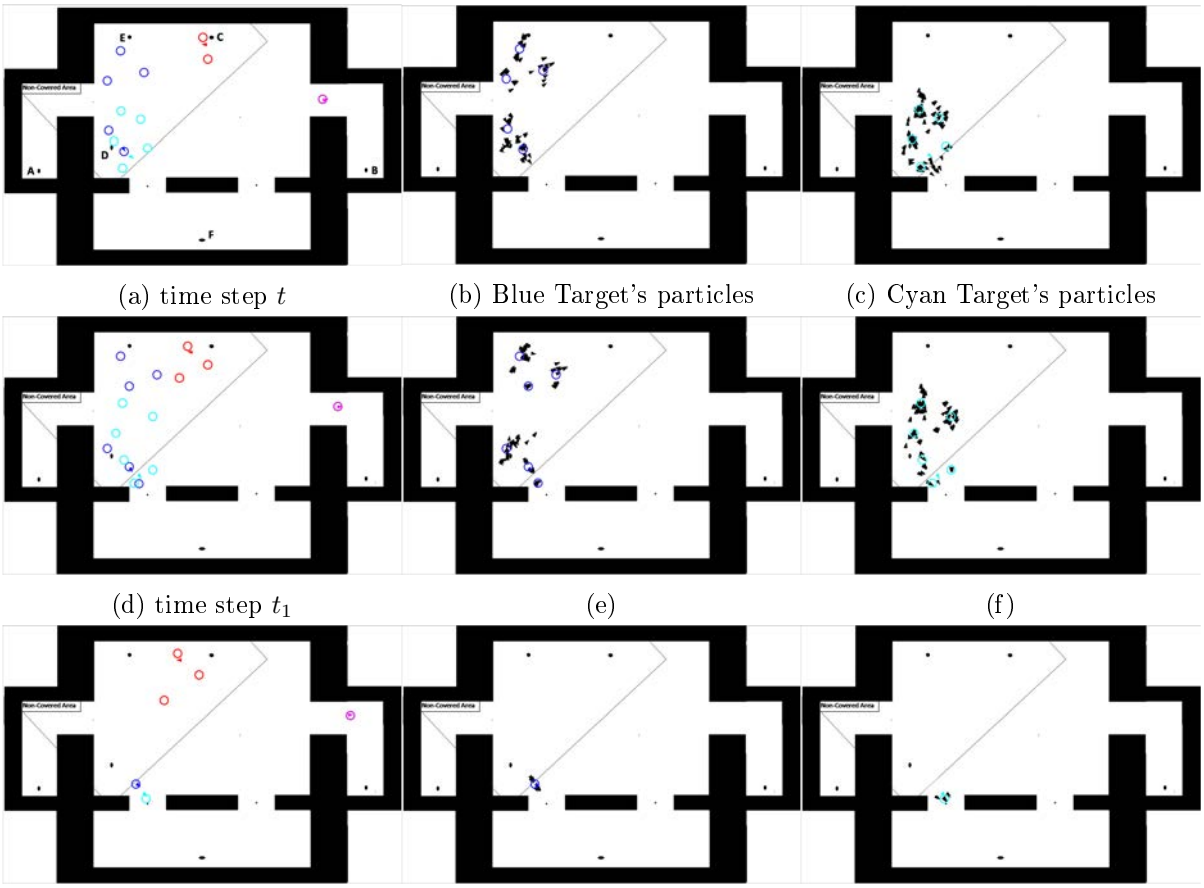
and/or the number of targets being tracked.

Table 7.7: Configuration's Execution time on Scenario 3.

Configurations	**Execution time** (in *sec*)
Configuration 0: JPDAF	105.98 ($\pm$3.72)
Configuration 1: Single Representative	119.85 ($\pm$8.06)
Configuration 2: Single Representative (Multi. Gating)	133.94 ($\pm$5.80)
Configuration 3: Multiple Representatives	307.62 ($\pm$200.79)
Configuration 4: Multiple Representatives (Multi. Gating)	430.50 ($\pm$277.80)

7.6.5.3 Impact analysis of the clustering methodology

In the previous section, we saw that the configuration yielding a best tracking performance within the considered environment is the one using multiple representatives both at the prediction and correction (gating) steps of the filtering process. Also, representatives were computed using a density-based clustering methodology. In this section, we still consider the configuration in which multiple representatives are used at both steps (prediction and the correction (gating)), and our objective is to study the performance variation when modifying the clustering methodology. To this end, we consider the centroid-based clustering methodology and we fix to 4 the number R of clusters (or representatives) to generate. Also, the maximum number N_c of iterations is set to 50. Afterward, we perform 100 runs and we compare the results obtained with those achieved in the previous section (density-based clustering).



(a) time step t (b) Blue Target's particles (c) Cyan Target's particles

(d) time step t_1 (e) (f)

(g) time step t_2 (h) (i)

Figure 7.26: Example of a resolved re-identification issue. On this example, we focus on the blue and cyan targets and we show their particles at some time steps. At time step t, the red, blue and cyan targets are not under sensory coverage. At time step t_1, the cyan target exits the non-covered area and the system is unable to determine the ID of the exiting targets. However, only blue and cyan targets are potential candidates. At the reception of new (unambiguous) observation at time step t_2, the system resolves the target identities.

Figure 7.30 puts together the average error $AvgErr_t$ resulting from both clustering methodologies. Although we can notice a light degradation of the performance when using the centroid-based clustering methodology (see Figure 7.30a), the difference is not statistically significant as illustrated by Figures 7.30b and 7.30c. The difference observed can be imputed to the approximations made regarding the convergence of each centroid-based clustering procedure (maximum number N_c of iterations is 50) in order to maintain the computational time at a reasonable level. On this basis, we can state that the performance of our system may depend on the reliability and the stability of the clustering methodology used.

Table 7.8 reports the execution times obtained for both clustering methodologies. While, in terms of performance quality, both methodologies lead to similar results, the execution time associated to the centroid-based clustering methodology is larger than the one associated to the density-based clustering methodology. This is because, with the centroid-based clustering methodology, the number of representatives obtained at each time step for each target is constant

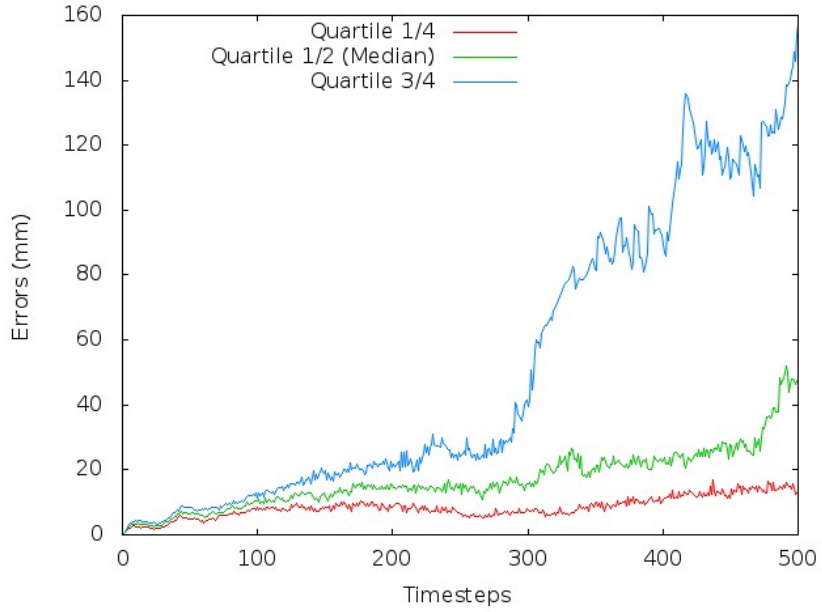
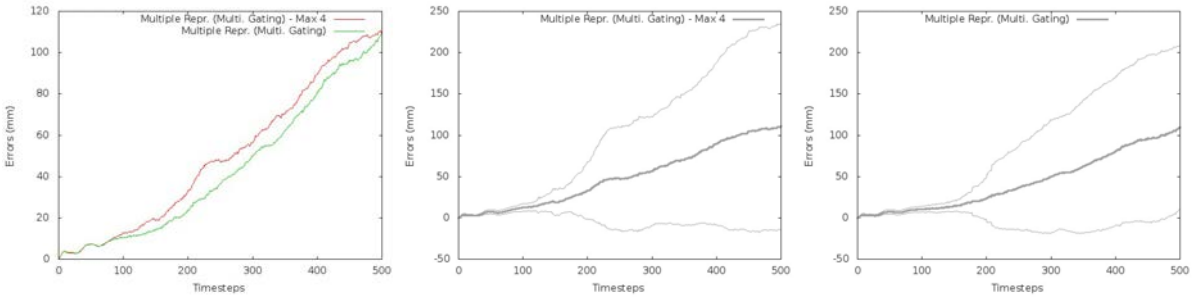


Figure 7.27: Performance's quartiles under *Configurations 4* (Mult. Rep. (Multi. Gating)).



(a) Mean (b) $Num.Max.Repr. = 4$ (c) Unconstrained case

Figure 7.28: Performance Illustration: Constraining the number of representatives. (a) puts together the average error obtained in constrained and unconstrained cases while (b) and (c) show the standard deviations in each case.

even if the shape of the underlying distribution is not multi-modal, thus leading to unnecessary simulations. On the other hand, with the density-based clustering methodology, the number of representatives for a given target adjusts with the shape of the distribution.

Table 7.8: Clustering methodology - Execution time on Scenario 3.

Clustering methodology	**Execution time** (in *sec*)
Density-based clustering	430.50 ($\pm$277.80)
Centroid-based clustering	703.1 ($\pm$187.96)

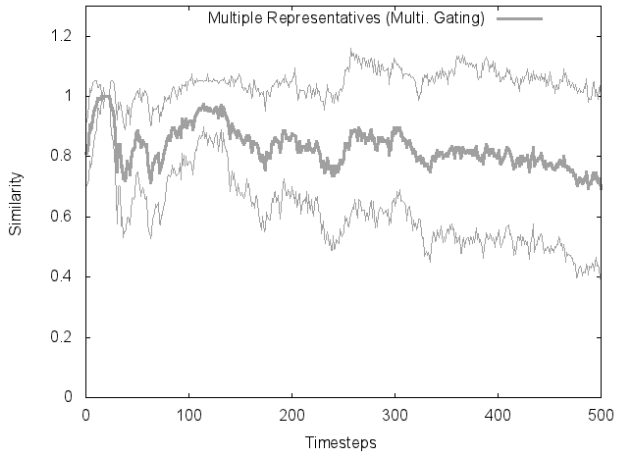
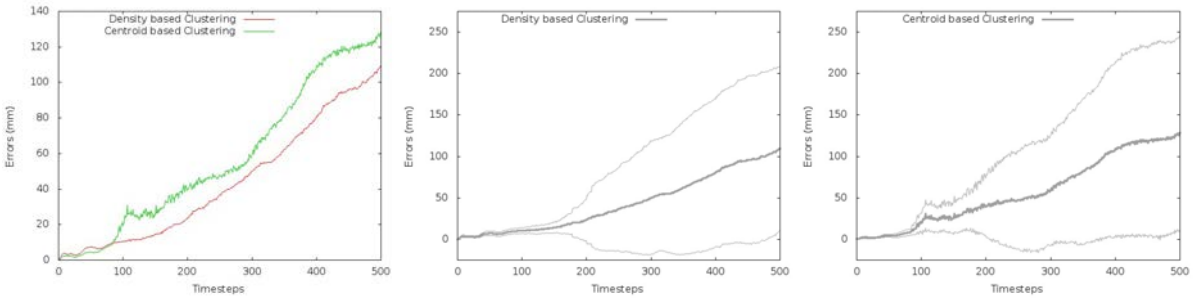


Figure 7.29: Average Goal Similarity ($AvgSim_t$) under *Configurations 4*: the plot depicts the mean and associated standard deviations.



(a) Mean (b) Density-based Clustering (c) Centroid-based Clustering

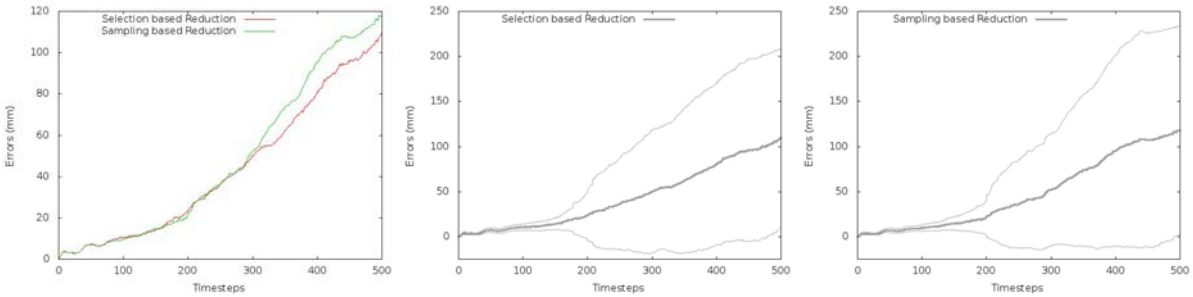
Figure 7.30: Centroid-based clustering *versus* Density-based clustering.

7.6.5.4 Impact analysis of the reduction policy

As we may now have understood, considering multiple representatives for managing target interactions at the prediction step of the filtering process leads to an increase of the number of particles. In order to avoid a combinatorial explosion over time, we need to reduce this number of particles and keep it constant from a given time step to the next one without losing the behavioral diversity within the global particle set. The objective of this section is to analyze the impact of the reduction policies introduced in Section 7.5.3.2 on the tracking performance. The experiments described in Section 7.6.5.2 were performed using the selection-based reduction policy. In this section, we consider using the sampling-based reduction policy under the configuration in which multiple representatives are still used at the prediction step (for managing target interactions) and the correction step (gating procedure). On this basis, 100 runs are performed and the obtained results are described in Figure 7.31 while the execution times are reported in Table 7.9.

As illustrated, the tracking performance resulting from the sampling-based reduction policy is approximately equivalent to the performance obtained when using the selection-based policy meaning that, for the underlying tracking scenario, both policies capture the same information

regarding the diversity within a given population.



(a) Mean (b) Selection-based Reduction (c) Sampling-based Reduction

Figure 7.31: Selection-based Reduction *versus* Sampling-based Reduction.

Table 7.9: Reduction policy - Execution time on Scenario 3.

Reduction policy	**Execution time** (in *sec*)
Selection-based reduction	430.50 ($\pm$277.80)
Sampling-based reduction	415.42 ($\pm$282.04)

7.6.5.5 Impact analysis of N

In this section, our aim is to analyze whether or not the performance of the proposed algorithm on the considered tracking scenario can further be improved by varying the number N of particles used to characterize a target distribution. To this end, we consider the following values of N: 100, 1000 and 2000. For each of these values, 100 runs are performed (still under *Configuration 4*) and the results are presented below.

Figure 7.32 shows the average error $AvgErr_t$ obtained for each value while Figure 7.33 presents the corresponding standard deviations. The graphics in Figure 7.32 suggest that the larger the number of particles used, the better the tracking quality. This is very interesting *per se* given the re-identification problem which is inherent to the considered tracking scenario. In other words, the proposed algorithm does not have the ambition to solve the re-identification problem, but contributes, on average, to disambiguating targets as they re-enter covered areas. Moreover, the contribution is more important as the number of particles per target increases.

Finally, Table 7.10 reports the execution times obtained for each of the considered values of N. As one can expect, it appears that the execution time increases with the value of N.

7.6.5.6 Impact analysis of K

Our concern in this section is to analyze the general behavior of our system with respect to the density of the environment under supervision. To cope with this objective, we subsequently consider tracking 2, 3, 4 and 6 targets within the above-described environment using the *Configuration 4* defined previously. For each of these numbers, we perform 100 runs in which each target

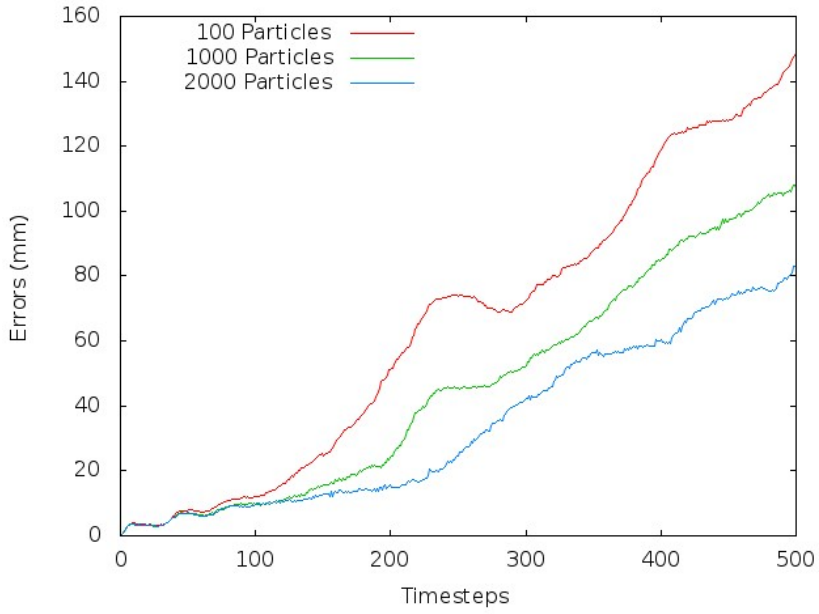
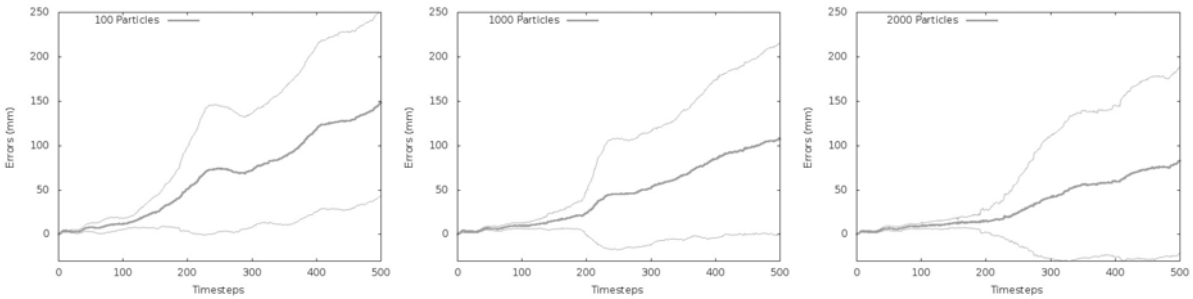


Figure 7.32: Impact analysis of N - $AvgErr_t$



(a) 100 Particles (b) 1000 Particles (c) 2000 Particles

Figure 7.33: Impact analysis of N - Means and standard deviations.

Table 7.10: Impact analysis of N - Execution time on Scenario 3.

Number of particles (N)	**Execution time** (in *sec*)
100	79.3 ($\pm$17.12)
500	430.5 ($\pm$277.80)
1000	1578.1 ($\pm$316.63)
2000	2232.61 ($\pm$593.59)

is characterized by a population of $N = 500$ particles. The resulting performances on average are shown in Figure 7.34 while Figure 7.35 illustrates the corresponding standard deviations. Finally, Table 7.11 presents the execution times obtained.

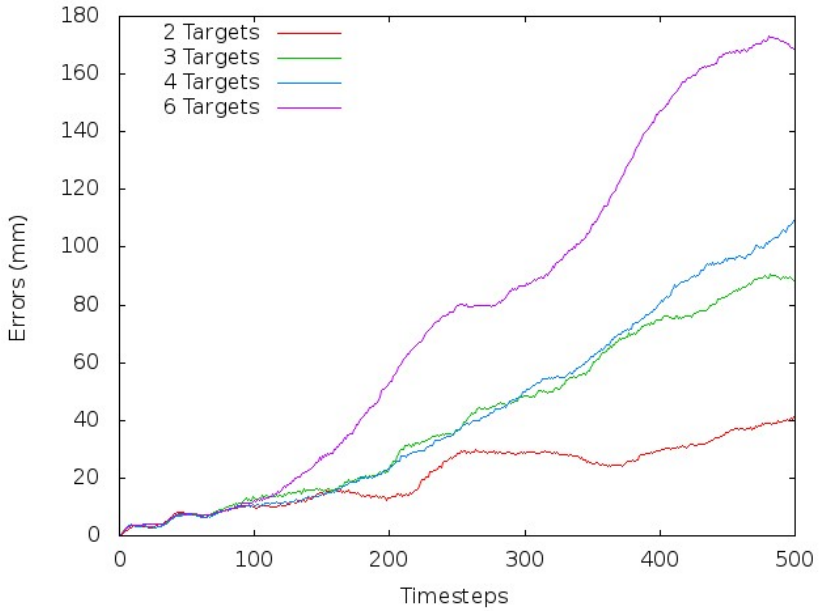
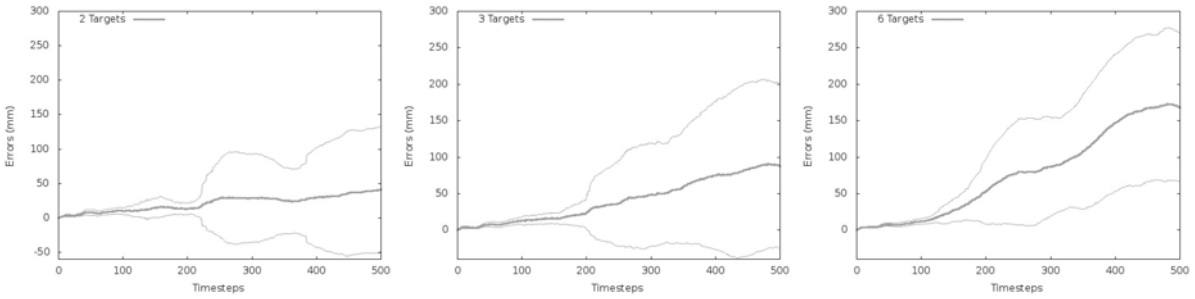


Figure 7.34: Impact analysis of K - $AvgErr_t$

As we can expect, the larger the density, the worse the tracking performance. This is in perfect accordance given the re-identification issues we are facing in this environment. Indeed, the percentage of lack of re-identification increases with the number of targets. However, these results do not depreciate the proposed algorithm as we are only relying on behavioral inference against the complex re-identification problem. Nowadays, solutions based on feature extraction have been developed in parallel for minimizing the rate of false re-identification in a supervision system. In such a solution, features characterizing a given target are learned while the latter is under sensory coverage and are subsequently used to recover the target when it exits non-covered areas. The features to be extracted depend on the application domain and some examples include the clothing's color, the height, the shape [Sullivan and Carlsson, 2006, Song et al., 2008]. One question of particular interest is "how would the proposed algorithm behave if it were coupled with such a re-identification solution?" and it is explored in the next section.

Table 7.11: Impact analysis of K - Execution time on Scenario 3.

Number of targets (K)	**Execution time** (in *sec*)
2	115.1 ($\pm$26.14)
3	280.37 ($\pm$165.73)
4	430.50 ($\pm$277.80)
6	2126.18 ($\pm$500.74)



(a) 2 Targets (b) 3 Targets (c) 6 Targets

Figure 7.35: Impact analysis of K - Standard deviations.

7.6.5.7 Coupling with an external re-identification module

The purpose of this section is to analyze the performance of the proposed algorithm if we consider the presence of a re-identification module which helps, **only when necessary**, disambiguating among potential candidates with respect to a given atomic observation upon leaving non-covered areas. To this end, we simulate the re-identification module as follows: we assume that an observation generated from a target exiting a covered area is tagged with the ID of the target and this holds for a predefined number of time steps (3 in our case) from the moment it exists. It has to be pointed out that the information provided by the simulated re-identification module is then used only during the gating procedure to clarify disambiguated situations and serves under no circumstances to re-initialize a filter which has lost its track. Finally, to quantify the contribution of the proposed algorithm with respect to the re-identification module, we perform experiments using the following two configurations defined earlier: *Configuration 1* and *Configuration 4*. We recall that *Configuration 1* extends the classical JPDAF by managing targets interactions using a single-representative-based approach. On the other hand, *Configuration 4* explores the usage of multiple representatives for target interaction management and for the gating procedure.

To study the performances, we consider tracking $K = 4$ targets in the environment and each target distribution is represented by $N = 500$ particles. For each of the above configurations, 100 runs are performed and the results are presented below.

Figure 7.36 depicts the average error $AvgErr_t$ obtained for both configurations while Figure 7.37 illustrates the performance quartiles. As it can be observed, *Configuration 4* outperforms *Configuration 1* demonstrating therefore, even in presence of a re-identification module, the advantages of multiple-representative-based approaches with respect to single-representative-based approaches in case of multi-modality. Also, because of the re-identification module, the results obtained are better than the ones presented in Figure 7.22. Moreover, they are more stable as acknowledged by Figure 7.37b where the performance's quartiles for *Configuration 4* are presented. These results demonstrates the reliability of the proposed algorithm even in cases of complex tracking scenarios.

7.7 Conclusion and Discussion

In this chapter, we addressed the problem of tracking a fixed number of autonomous and interacting targets within a complex environment under the assumption that the behavior model

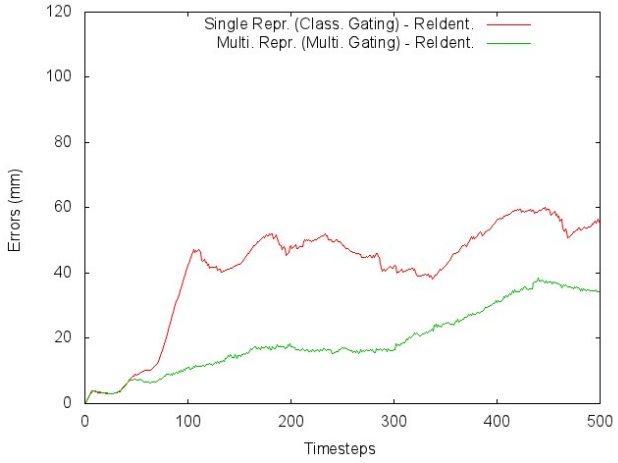
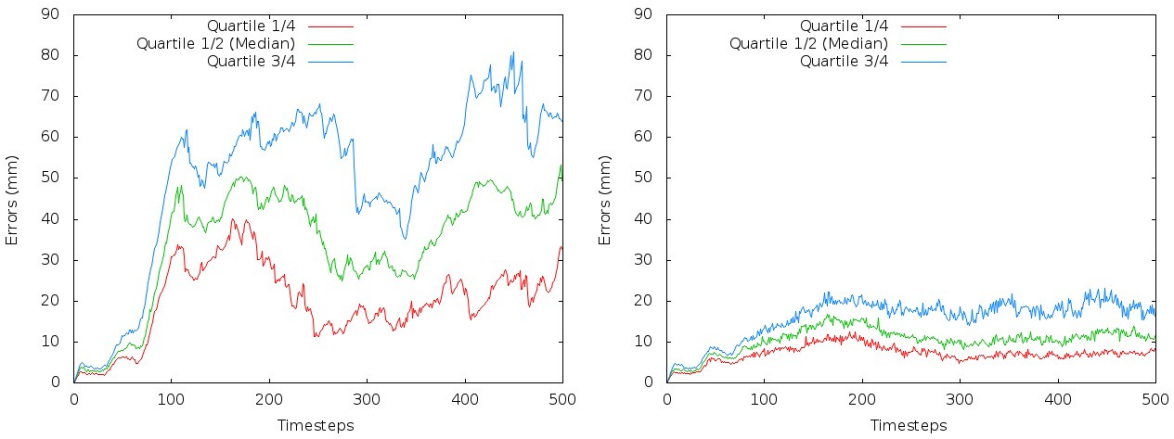


Figure 7.36: Evaluation with an external re-identification module - $AvgErr_t$



(a) *Configuration 1* (b) *Configuration 4*

Figure 7.37: Performance's quartiles in presence of a re-identification module in Configurations 1 and 2.

of each target as well as the interaction model between targets is fully available and encoded within a simulator. Unlike previous works in the literature, we propose for our tracking solution a representation of target dependencies (interactions) which facilitates the approximation of targets' posterior distributions using a factored Bayesian filter (composed of collaborative individual filters, each per target).

After formally introducing the interaction model, we propose a methodology for estimating the predicted distribution of each target. In order to reduce the complexity of such an estimation, we rely on two main heuristics: (1) the exploitation of the "locality of interaction" concept which stipulates that two targets may interact if and only if they are in a close neighborhood, and (2) the aggregation of individual target's distribution into representative states in such a way that the only information required by a given target's associated filter regarding other targets is provided by these representative states. As a result, an interaction graph is built on the fly at each time step, based on target representatives as well as a neighborhood criterion, to capture

the essence of target interactions within the environment.

Then, we describe, within the joint probabilistic data association (JPDA) framework, a Monte-Carlo implementation of the proposed approach in which local interactions between targets (as suggested by the interaction graph) are taken into account when updating the track of each target. This description leads us, in case of multiple representatives per target, to reconsider how classical gating solutions have been designed and to propose a brand new soft-gating procedure based on multiple representatives. Also, because the interest of multiple representatives can be well observed in areas which are not under sensory coverage, we propose an extension of the classical JPDAF algorithm to cope with environments having such areas since the original algorithm was not designed to deal with these characteristics. Finally, we perform a general evaluation of the designed algorithm.

Dynamics-based interactions

The results obtained from this evaluation illustrate a significant improvement of the quality of the tracking performance with respect to a classical JPDAF filter (which neglects the interactions among the targets). Also, these improvements are obtained at low computational cost. Indeed, when a single representative per target is used, the complexity of the proposed algorithm is nearly equal to the complexity of the classical JPDAF filter. When multiple representatives per target are considered, the complexity depends, in average, on the number of representatives required for capturing individual target properties at each time step.

Despite these improvements, we show that attention has to be paid when designing the neighborhood criterion used for building the interaction graphs. Interaction graphs underestimating local interaction potentials (as implemented within the simulator) will yield poor tracking results, while interaction graphs overestimating these interaction potentials will result in an algorithm wasting time in simulating unnecessary situations in the sense that they would not improve the quality of the tracking process. Nevertheless, we show that it is possible to find a compromise between the computation time and the performance quality of the filter, at least on the specific problem we focused on. The same analysis can be done regarding the maximum number of representatives per targets given a specific tracking scenario. Having a low (maximal) number of representatives will lead to a fast execution of the filtering process at the expense of its quality. On the other hand, overestimating this number increases the computational cost without necessary improving the tracking performance. Here again, a compromise between the computation time and the filtering performance needs to be found. Works should be undertaken to pursue in this direction.

Partially covered environments

In environments characterized by the presence of areas which are not under sensory coverage and where targets are subject to a multitude of behaviors, it comes out that the application of the multiple-representative-based soft-gating presents a valuable advantage with respect to the classical soft-gating procedure as it allows to perform, in a finer way, the data association between the targets and the related atomic observations and therefore, avoid the insertion of unnecessary noise within the filtering process. Moreover, by simply reasoning on a multiple representative basis, the proposed algorithm provides a significant premise for addressing the re-identification issue faced in several tracking scenarios. Indeed, because of the maintenance of a behavioral

diversity within the particle set characterizing each target when the latter is not observed, the proposed algorithm is able to provide a valuable clue regarding the identity of the targets exiting a non-covered area.

Of course, when there is no ambiguity (only representatives from a unique target exhibit the same behavior and thus, they are the only ones exiting the area at the same time), the proposed algorithm perfectly recovers the ID of the target. However, as it has been shown, in case of ambiguous situations (a subset of targets can be matched with the exiting one), the knowledge of this information within the JPDAF framework does not guarantee a correct re-identification of the involved targets as the filtering process is carried on. This is because, when focusing on the correction step (weight assignment and resampling) of the (MC)JPDAF filter, we realize that the survival of the matched representatives (the particle composing the representatives) depends not only on upcoming observation data but also on their relative weight with respect to other hypotheses (representatives) related to the target they represent.

At the moment, in such an environment, the best performance of the proposed algorithm is obtained when it is coupled with an external re-identification module (e.g. based on feature extraction or, an human operator in the loop) which is able to provide additional information for disambiguating targets exiting non-covered areas. Nevertheless, we believe that one promising perspective would be to investigate how to extend the JPDAF framework in such a way to get rid of external re-identification modules while achieving good filtering performances in such environments.

Limitations of the described work

Despite the improvements observed on experiments we carried out with respect to classical solutions, the proposed algorithm has currently several limitations. The first one corresponds to the way the interaction graph is generated. Indeed, we rely on a simple function which is based on a basic heuristic and elementary information (distance between two targets). More work has to be done on how to build these interaction graphs when using a more advanced multi-agent simulator designed to represent the dynamics of a real-world phenomenon like crowd behavior in a dedicated context. A lot of work [Davidsson, 2002, Kubera et al., 2008] has been undertaken to represent groups of interacting agents in multi-agent systems, and one interesting direction for future work would be to envision the possibility of taking advantage of a specific interaction formalism in order to build, on the fly, an interaction graph consistent with the considered simulations.

Another limitation of the presented work concerns the "memory of interactions" from one time step to the next. Indeed, the interaction graph is built based on properties inherent to the states of the targets at the current time step. This approximation is not sufficient to properly manage interactions which are persistent over time. An initial solution to this limitation would be to define the interaction's function in such a way that it depends not only on current target states, but also on their states at previous time steps. However, a particular attention needs to be paid to the temporal window to be considered in such a way to prevent storage limitations. More generally, a question of particular interest would be "how to memorize only useful information from past timesteps in such a way to efficiently predict the behaviors of the involved target in the future". As an inceptive thought to the answer of this question, one could imagine to keep track of the identity of the targets a given target's particle is interacting with at the previous time steps.

Chapter 8

Conclusion

Contents

This chapter reviews the work presented through this thesis while highlighting our contributions as well as their limitations. Moreover, we describe some tasks we would like to perform together with interesting directions to investigate in the future. Finally, we present some examples of application domains in which our work can be integrated.

8.1 Contributions

In this thesis, we were concerned with the design of a supervision system for surveillance purpose. More specifically, we were interested in the behavioral tracking problem which consists in inferring pedestrians' behaviors within an environment while relying on observation data received from a sensor network. We considered the general case in which the sensor network may not completely cover the environment. This problem is generally challenging, mainly because of the following two factors:

- **Individual and context-dependent behavior**: the behavior exhibited by a given pedestrian depends on his personal characteristics (e.g., velocity, age, stamina) as well as the environment in which he is immersed. Indeed, the objects present in the environment greatly influence such a behavior, especially, the activities in which the underlying pedestrian may be interested. On this basis, it is crucial to have a good representation of the environmental context in which the tracking process is performed. Additionally, both the location and the activity of a given pedestrian are of great importance and they are inter-related.

- **Pedestrians' interactions**: in a multi-target setting, pedestrians in a close neighborhood mutually influence each other. These dependencies add another layer of complexity to the above-mentioned problem since, for better efficiency, it is not recommended to reason on each pedestrian individually.

The work conducted within this PhD thesis has been centered on these factors and the following sections present our contributions for efficiently dealing with each of them.

8.1.1 Simultaneous tracking and activity recognition

For estimating pedestrian behaviors, we considered the STAR (Simultaneously Tracking and Activity Recognition) paradigm introduced by Wilson and Atkeson [2005]. Its principle consists in exploiting, during the inference procedure, contextual knowledge related to activities in order to improve the estimations of targets' location and inversely.

Existing works designed under this paradigm [Wilson and Atkeson, 2005, Manfredotti et al., 2011] mostly rely on dynamic Bayesian networks (DBNs) to represent the environmental context in which the tracking process is performed. While DBNs are suitable for representing event sequences and causality effects, their expressiveness is limited. Indeed, because of the strict first-order assumption characterizing such models, one may need quite large data structures — which may prove to be difficult to manage in practice — for representing common contextual information as for example temporal data (e.g., the duration of an activity).

Another limitation of the above-mentioned approaches is the assumption that the environment is not subjected to any dynamical changes which may affect pedestrian behaviors. This is usually not the case in real situations (e.g., escalator failure, fire alarms).

In this thesis, we proposed an orthogonal approach (cf. Chapter 5). Instead of relying on DBNs, we used an advanced behavioral simulator of autonomous agents for modeling the environmental context. Such a simulator allows to represent, in a finer and less restrictive way, the details of the environment at hand together with the objects therein as well as the related services. For example, it is possible to easily describe the behavioral process of a pedestrian when interacting with a given object (e.g., an ATM) as well as the duration of such an interaction. Also, it is possible to characterize a pedestrian (assimilated to an agent) not only by basic properties (e.g., location, velocity), but with action-selection mechanisms involving more abstract internal properties (e.g., thirst). Additionally, these simulators are able to generate, for any simulated agent, consistent and realistic behaviors in relation with its internal state and, more importantly, the environmental context. As a solution to the behavioral tracking problem, we proposed, on the one hand, to integrate such a simulator as a predictive block within a particle filter for behavioral analysis purposes, and on the other hand, to feed the considered simulator with environmental changes (e.g., escalator failures, fire alarms) in such a way to adapt on the fly.

The proposed solution has been implemented using SE-Star, a Thales proprietary simulator, and experiments have been conducted, for the special case of a single pedestrian tracking, in both virtual and real conditions. We obtained very good results, thus demonstrating the efficiency and the robustness of the proposed approach in terms of both location and activity estimations. Also, the quality of the achieved performance is still outstanding in cases of difficult tracking conditions such as long periods of occlusion and/or exogenous events.

Despite these improvements, our solution presents a limitation. Indeed, its response time is limited by an incompressible computational factor corresponding to the time required by the underlying simulator for simulating a hypothesis (particle). The larger this factor, the larger the computational resources needed by the system to perform efficiently under real-time constraints. In an effort to alleviate this limitation, we implemented a distributed version of the designed solution.

8.1.2 Management of target interactions

The main reason for considering the simplest case of single-target tracking in the above described experiments was the ability to focus on the representation of contextual information and how this can be leveraged to improve a given target behavior estimation. However, rare are environments in which only a unique pedestrian is authorized to evolve at any time. In a multi-target setting, additional difficulties — not encountered in the single-target case — increase the complexity of the behavioral tracking problem. These include:

- associating observation data to corresponding targets: when the sensors do not provide tagged observation data, one problem of major interest consists in determining from which target an atomic measurement originates. In the literature, this problem is usually referred to as the data association (DA).

- managing dynamics-based interactions: the dynamics of a given target may be influenced

by the presence of other targets in its neighborhood.

DA has been studied for decades and classical solutions have been proposed such as the multiple hypothesis tracker (MHT) [Reid, 1979] and/or the joint probabilistic data association (JPDA) [Fortmann et al., 1983]. On the other hand, few are works [Khan et al., 2003, Cattelani et al., 2014] which do not assume complete independence between targets' dynamics. However, the approaches proposed in these works either make problem-specific assumptions for breaking down the complexity and are therefore less generalizable, or embed interaction models within the target dynamics thus resulting in generalizable solutions but this time with high computational cost as they work on the joint state composed of individual target states.

In this thesis, we focused on the management of dynamics-based interactions. As part of the second contribution, we were guided by the idea of developing an approach which combines the advantages from both categories. Putting differently, we sought for a generic solution based on interaction models from which the posterior distribution of tracked targets can be approximated at lower computational cost using nearly independent (sub)filters, each one committed to a particular target. Under the assumptions that we are provided with a simulator (e.g., advanced behavioral simulators of autonomous agents) or a function modeling the targets' behaviors and their mutual dependencies, we proposed, in Chapter 7, a factored Bayesian filter for estimating the predicted distribution of each target while taking into account dynamics-based interactions.

In our work, we assumed that the evolution of a given target state at the next time step depends solely on neighboring targets states at the previous time step. This allows (*i*) factoring the dynamics of all the targets into individual target dynamics at each time step and, (*ii*) designing a set of individual (sub-)filter processes — one per target — which *collaborate*, when necessary (i.e., in case of interactions), for the tracking purposes. Additionally, we proposed two heuristics (Section 7.4) for efficiently approximating the distributions resulting from these nearly independent (sub-)filters:

- the first one suggests to aggregate, at each time step, individual target distributions into few representative states in such a way that these states are the only information needed by other target (sub)filters for updating their posterior distributions (cf. Section 7.4.4);
- the second heuristic proposes to capture on the fly the structure of the dynamics-based interactions at each time step using an interaction graph that we build based on the so computed representative states.

To assess our contribution, we relied on the JPDA framework for solving the data association. Within this framework, we described a Monte Carlo (MC) implementation of the designed system. Besides its ability to manage targets involved in dynamics-based interactions, this implementation allowed us to bring to light two other improvements with respect to the classical MC-JPDA filter:

- multiple-representative-based soft-gating: the classical gating approach aims at "casting" the predictive likelihood of a target into a unique Gaussian density distribution from which distance computations can be performed for determining which atomic measurements potentially originate from the considered target. This conversion is not always convenient in case of strong non-linearity. To cope with this limitation, we proposed a brand new soft-gating procedure based on target's representatives for finer associations.

- management of non-covered areas: by revising the observation model, we proposed an extension to the classical JPDAF algorithm for partially covered environments as the original filter has been designed under the assumption that the environment under consideration is fully covered by the sensor network.

Finally, we evaluated our implementation on challenging virtual tracking scenarios using the steering behavior simulator provided in [Christian and Thomas, 2007] as the reference simulator. The results obtained were very interesting and they demonstrated the efficiency of the approach in managing dynamics-based interactions. Moreover, in presence of non-covered areas where targets are subject to a multitude of behaviors, the experiments highlighted the advantages of using the multiple-representative-based soft-gating procedure — instead of the classical procedure — as it attenuates the presence of association noise within the filtering system. Also, our system proved to be valuable for addressing the re-identification issue (i.e., the re-identification of a given target when re-entering an area under sensory coverage). This is made possible by the ability of the designed algorithm to maintain, by means of the multiple representatives, a behavioral diversity within the particle set characterizing each target distribution when the underlying target is within a non-covered area.

8.2 Future Work

This section presents the perspectives of the work carried out throughout this thesis.

8.2.1 Real-world evaluations

Unlike the work performed in the first part of this thesis, the approach we proposed for tracking multiple targets was essentially evaluated on virtual scenarios using a steering-based behavioral engine as baseline simulator. The main reason for this choice was to demonstrate the effectiveness of the proposed approach on the most common pedestrian interactions, i.e., those related to navigation. However, behavioral dependencies in real-life situations are not limited to navigational issues. The next step we would like to undertake is to evaluate the performance of our system on real scenarios using an advanced behavioral simulator of autonomous agents like SE-Star. Nevertheless, considering tracking multiple interacting targets in real life using such an advanced simulator raises several implementation issues not yet faced in the single target case.

One of these issues is directly related to the plethora of types of pedestrian interactions that exist in the real world. Therefore, one question of particular interest is how to efficiently build an interaction graph which is consistent with the environmental context and best represents all types of interactions encountered. A lot of work has been undertaken to represent groups of interacting agents in multi-agent systems [Davidsson, 2002, Kubera et al., 2008], and a promising direction for further investigation would be to examine the possibility of leveraging these interaction formalisms for building such a graph.

Another implementation issue is related to the fact that, in real life, dynamics-based interactions may be persistent over time, that is they have a certain duration. In such a case, our system may perform poorly given that we have approximated these interactions on the sole basis of target representatives obtained at the previous time step. To cope with this issue, an initial

approach may consist in considering building the interaction graphs using all representatives generated within a temporal window. However, a particular attention needs to be paid regarding the length of the window to be used in such a way to prevent storage limitations. Another direction consists in investigating a way to only memorize information which is judged to be relevant for the estimation of future target behaviors. This problematic is usually encountered in the field of decision theory, and particularly, in case of (decentralized) partially observable Markov decision processes (POMDPs) [Pineau et al., 2003, Dibangoye et al., 2015].

8.2.2 Target re-identification

As mentioned before, in presence of environments with areas that are not under sensory coverage, the designed system proved to be an inceptive clue for facing the common re-identification issue by maintaining a behavioral diversity regarding targets within such an area. Consequently, when a target is exiting a non-covered area, our system can provide, as output, an accurate subset of target identities, each of which being associated to at least one maintained behavioral hypothesis (representative) that matches the observations received.

When the cardinality of the returned subset is higher than one, the results obtained in Chapter 7 show that the correct re-identification of the target is not guaranteed. Indeed, the survival of candidate hypotheses depends not only on upcoming observation data but also on their relative weight within the set of hypotheses associated to the target they represent.

After analysis, we found that the main reason resides in the fact that the correction step (weight assignment and resampling) in the (MC-)JPDA filter does not naturally integrate additional information regarding the status (i.e., is it candidate for a re-identification or not) of a maintained hypothesis. Therefore, after proposing in this thesis improvements regarding the prediction step (management of dynamics-based interactions) as well as the gating procedure within the JPDA framework, one possible extension of our work could be to investigate how to improve the correction step of the framework to cope with this limitation.

8.2.3 Sensor control decision policies

The general application domain motivating the work performed throughout this thesis was the supervision (or surveillance) of critical environments. When the sensors under consideration are controllable (e.g., using PTZ (pan tilt zoom) cameras), it may be interesting to explicitly control them for better information gathering. This could be useful for disambiguating maintained hypotheses within the system, and particularly those that seem most critical (e.g., maximal coverage of areas with high criticality levels, refinement of the constructed belief, i.e., elimination of hypotheses which prove to be incorrect). On this basis, the sensors' control policy, in order to be effective, needs to reason on all the possible future behaviors of the underlying targets.

Such a control (or decision) problem is generally modeled in the literature as a POMDP [Pineau et al., 2003] (or a rho-POMDP [Araya et al., 2010]) which allows to derive, mostly from an offline planning stage, the optimal control policy. However, in this case, the space of information states to consider is too large to envision the offline planning. It would be interesting to investigate how to compute accurate policies, in an online fashion, for controlling the environment sensors while relying on the distributions computed from the designed system.

8.2.4 Beyond behavioral tracking

Through this thesis, we have developed an approach for factorizing a Bayesian filter (and specifically, a particle filter) within the framework of multi-target tracking. This allows to break down the complexity related to the filtering problem within a high dimensional state space (here, the joint state is composed of all the individual target states).

More generally, let us consider a partially observed dynamical system whose internal state is represented using a DBN involving multiple random variables. Under the assumption that there are no dependencies between variable instances at the same time step, the methodology proposed in this work could be used as a baseline for estimating, at low computational cost, the state of the system.

8.3 Application domains

This section provides examples of application domains, other than (video) surveillance, in which the work achieved throughout this thesis can be applied.

8.3.1 Behavioral model calibration

One of the main difficulties encountered by people interested in designing realistic behavioral models with respect to an environment is the evaluation of the generated model to determine whether or not it sufficiently represents the behaviors of the underlying pedestrians. In this scope, one can easily use our system as part of the evaluation process to iteratively tune the behavioral model. More specifically, given the current behavioral model to evaluate, it is possible to wrap it within a simulator whose objective will be to simply execute the model. Subsequently, the generated simulator can be coupled with our system and thus, they can be used together against the initial contextual environment for inference purposes. In such a setting, the quality of the inference results is a fair indication of the effectiveness of the generated model. Indeed, the higher the inference score, the better the model. Additionally, it will be possible to track down whether or not modifications performed on the model between two successive tuning iterations have led to an improvement.

8.3.2 Telemedicine monitoring service

Nowadays, the development of telemedicine services is growing steadily [Zouba et al., 2010, Dubois and Charpillet, 2013]. Healthcare integrated environments are used to allow patients to benefit from their own environment while they undergo a therapy. Usually, the therapy consists of a sequence of daily activities and processes that the patient should undertake on a regular basis.In this case, one problem of particular interest is to monitor whether or not the patient actually follows the therapy. Assuming we are provided with a behavioral model representing the normal behavior of a patient following the therapy, it is possible to rely on the work described in this thesis for the monitoring purposes. Indeed, based on dedicated sensors installed within the environment, our system can compute a score representing the difference, with respect to the reference behavioral model, of the behavior exhibited by the patient. As soon as the computed score is higher than a preset threshold, a notice is sent to the patient's supervisor.

Bibliography

J.K. Aggarwal and M.S. Ryoo. Human activity analysis: A review. *ACM Computing Surveys*, pages 1–43, Apr 2011.

M. Albanese, R. Chellappa, V. Moscato, A. Picariello, V. S. Subrahmanian, P. Turaga, and O. Udrea. A constrained probabilistic Petri net framework for human activity detection in video. *IEEE Transactions on Multimedia*, 10(6):982–996, October 2008.

Saad Ali and Mubarak Shah. Floor fields for tracking in high density crowd scenes. In *Proc. of European Conference on Computer Vision*, pages 1–14, 2008.

James F. Allen. Maintaining knowledge about temporal intervals. *Commun. ACM*, 26(11): 832–843, nov 1983.

G. Antonini, M. Bierlaire, and M. Weber. Simulation of pedestrian behaviour using a discrete choice model calibrated on actual motion data. In *4th Swiss Transport Research Conference*, 2004a.

Gianluca Antonini, Santiago Venegas, Jean-Philippe Thiran, and Michel Bierlaire. A discrete choice pedestrian behavior model for pedestrian detection in visual tracking systems. In *Proc. of the International Conference on Advanced Concepts for Intelligent Vision Systems (ACIVS)*, 2004b.

Mauricio Araya, Olivier Buffet, Vincent Thomas, and Françcois Charpillet. A POMDP extension with belief-dependent rewards. In *Advances in Neural Information Processing Systems*, pages 64–72, 2010.

Ronald C. Arkin. *Behavior-based Robotics*. MIT Press, Cambridge, MA, USA, 1st edition, 1998.

Kai O. Arras, Slawomir Grzonka, Matthias Luber, and Wolfram Burgard. Efficient people tracking in laser range data using a multi-hypothesis leg-tracker with adaptive occlusion probabilities. In *Proc. IEEE Int. Conf. on Robotics and Automation*, 2008.

M. Sanjeev Arulampalam, Simon Maskell, and Neil Gordon. A tutorial on particle filters for online nonlinear/non-Gaussian Bayesian tracking. *IEEE Trans. on Sig. Processing*, 50:174–188, 2002.

Stefania Bandini, Sara Manzoni, and Giuseppe Vizzari. Agent based modeling and simulation: An informatics perspective. *J. Art. Societies and Social Simulation*, 12, 2009.

Y. Bar-Shalom and Thomas E. Fortmann. *Tracking and data association*, volume 179 of *Mathematics in Science and Engineering*. Academic Press Professional, Inc., San Diego, CA, USA, 1987. ISBN 0-120-79760-7.

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

Y. Bar-Shalom, K.C. Chang, and H.A. Blom. Automatic track formation in clutter with a recursive algorithm. In *Proceedings of the 28th IEEE Conference on Decision and Control, 1989*, pages 1402–1408. IEEE, 1989.

W. Bauer. The Monte Carlo method. *Journal of the Society for Industrial and Applied Mathematics*, 6(4):438–451, 1958.

T. Bayes. An essay towards solving a problem in the doctrine of chances. *Phil. Trans. of the Royal Soc. of London*, 53:370–418, 1763.

Maren Bennewitz, Wolfram Burgard, Grzegorz Cielniak, and Sebastian Thrun. Learning motion patterns of people for compliant robot motion. *Int. Journal of Robotics Research*, 24:31–48, 2005.

Niclas Bergman. *Recursive Bayesian Estimation Navigation and Tracking Applications.* PhD thesis, Linkoping University, 1999.

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

Samuel S. Blackman. *Multiple-target tracking with radar applications.* Artech House radar library. Norwood, Mass. Artech House, 1986.

Victor J. Blue and Jeffrey L. Adler. Cellular automata microsimulation for modeling bi-directional pedestrian walkways. *Transportation Research Part B: Methodological*, 35(3):293–312, March 2001.

Aaron F. Bobick and James W. Davis. The recognition of human movement using temporal templates. *IEEE Trans. Pattern Anal. Mach. Intell.*, 23(3):257–267, March 2001.

Aaron F. Bobick and Andrew D. Wilson. A state-based approach to the representation and recognition of gesture. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19: 1325–1337, 1997.

Wilfried Bohlken, Patrick Koopmann, Lothar Hotz, and Bernd Neumann. Towards ontology-based realtime behaviour interpretation. *Human Behavior Recognition Technologies: Intelligent Applications for Monitoring and Security*, pages 33–64, 2013.

M. Brand. Understanding manipulation in video. In *Proceedings of the 2nd International Conference on Automatic Face and Gesture Recognition*, FG '96, pages 94–99, Washington, DC, USA, 1996. IEEE Computer Society.

Allison Bruce and Geoffrey Gordon. Better motion prediction for people-tracking. In *Proceedings of the IEEE International Conference of Robotics and Automation*, 2004.

Olivier Brun, Vincent Teuliere, and Jean-Marie Garcia. Parallel particle filtering. *Journal of Parallel and Distributed Computing*, 62(7):1186–1202, 2002.

J. J. Bryson. Hierarchy and sequence vs. full parallelism in action selection. In *From Animals to Animats 6: Proceedings of the sixth International Conference on Simulation of Adaptive Behavior*, pages 147–156, Cambridge, MA, 2000. MIT Press.

C. Burstedde, K. Klauck, A. Schadschneider, and J. Zittartz. Simulation of pedestrian dynamics using a two-dimensional cellular automaton. *Physica A: Statistical Mechanics and its Applications*, 295(3-4):507–525, Jun 2001.

Matthias Butenuth, Florian Burkert, Florian Schmidt, Stefan Hinz, Dirk Hartmann, Angelika Kneidl, André Borrmann, and Beril Sirmaçek. Integrating pedestrian simulation, tracking and event detection for crowd analysis. In *ICCV Workshops*, pages 150–157. IEEE, 2011.

L. W. Campbell and A. F. Bobick. Recognition of human body motion using phase space constraints. In *Proceedings of the Fifth International Conference on Computer Vision*, ICCV '95, Washington, DC, USA, 1995. IEEE Computer Society.

W. B. Cannon. *The Wisdom of the Body*. Norton, New York, 1932.

Charles Castel, Laurent Chaudron, and Catherine Tessier. What is going on? a high level interpretation of sequences of images. In *European Conference on Computer Vision (ECCV), Workshop on Conceptual Descriptions from Images*, pages 13–27, 1996.

Luca Cattelani, Cristina E. Manfredotti, and Enza Messina. A particle filtering approach for tracking an unknown number of objects with dynamic relations. *Journal of Math. Model. Algorithms in OR*, 13(1):3–21, 2014.

O. Chomat, J. Martin, and J.L. Crowley. A probabilistic sensor for the perception and the recognition of activities. In *Proceedings of the sixth European Conference on Computer Vision (ECCV)*, pages 487–503. Springer, June 2000.

Schnellhammer Christian and Feilkas Thomas. The Steering Behaviour project, 12 2007. URL http://www.steeringbehaviors.de/.

Ingemar J. Cox and Sunita L. Hingorani. An efficient implementation of Reid's multiple hypothesis tracking algorithm and its evaluation for the purpose of visual tracking. *IEEE Trans. Pattern Anal. Mach. Intell.*, 18(2):138–150, February 1996.

Jinshi Cui, Hongbin Zha, Huijing Zhao, and Ryosuke Shibasaki. Tracking multiple people using laser and vision. In *Proc. of the IEEE International Conference on Intelligent Robots and Systems (IROS)*, pages 2116–2121, 2005.

Hans Daduna and Ryszard Szekli. Dependencies in Markovian networks. *Advances in applied probability*, pages 226–254, 1995.

Dima Damen and David Hogg. Recognizing linked events: Searching the space of feasible explanations. In *CVPR*, pages 927–934. IEEE, 2009.

Aliakbar Gorji Daronkolaei, Saeed Shiry, and Mohammad Bagher Menhaj. Multiple target tracking for mobile robots using the JPDAF algorithm. In *Proceedings of the 19th IEEE International Conference on Tools with Artificial Intelligence - Volume 01*, ICTAI '07, pages 137–145, Washington, DC, USA, 2007. IEEE Computer Society.

René David and Hassane Alla. Petri nets for modeling of dynamic systems: A survey. *Automatica*, 30(2):175 – 202, 1994.

Paul Davidsson. Agent based social simulation: A computer science view. *Journal of artificial societies and social simulation*, 5(1), 2002.

Colin de la Higuera. Current trends in grammatical inference. In *Proceedings of the Joint IAPR International workshop on advances in pattern recognition*, Lecture Notes in Computer Science, pages 28–31. Springer, 2000.

E. De Sevin and D. Thalmann. A motivational model of action selection for virtual humans. In *Proceedings of the Computer Graphics International 2005*, CGI '05, pages 213–220, 2005.

Carlos R. del Blanco, Fernando Jaureguizar, and Narciso N. García. An advanced Bayesian model for the visual tracking of multiple interacting objects. *EURASIP Journal of Advances in Signal Processing*, 2011:130, 2011.

Julien Diard, Pierre Bessière, and Emmanuel Mazer. A survey of probabilistic models, using the Bayesian programming methodology as a unifying framework. In *Proc. of the 2nd Int. Conf. on Computational Intelligence, Robotics and Autonomous Systems*, 2003.

Jilles Steeve Dibangoye, Christopher Amato, Olivier Buffet, and François Charpillet. Optimally solving Dec-POMDPs as continuous-state MDPs: Theory and algorithms. *Journal of Artificial Intelligence Research*, 2015.

P. Dollar, V. Rabaud, G. Cottrell, and S. Belongie. Behavior recognition via sparse spatio-temporal features. In *Proc. of the International Conference on Computer Communications and Networks (ICCCN)*, pages 65–72, 2005.

Roland Donat, Philippe Leray, Laurent Bouillaut, and Patrice Aknin. A dynamic Bayesian network to represent discrete duration models. *Neurocomputing*, 73(4):570–577, 2010.

Randal Douc, Olivier Cappe, and Eric Moulines. Comparison of resampling schemes for particle filtering. In *Proc. of the 4th International Symposium on Image and Signal Processing and Analysis (ISPA)*, pages 64–69, 2005.

Arnaud Doucet, Nando de Freitas, Kevin P. Murphy, and Stuart J. Russell. Rao-Blackwellised particle filtering for dynamic Bayesian networks. In *Proceedings of the 16th Conference on Uncertainty in Artificial Intelligence*, UAI '00, pages 176–183, San Francisco, CA, USA, 2000. Morgan Kaufmann Publishers Inc.

Amandine Dubois and François Charpillet. Detecting and preventing falls with depth camera, tracking the body center. In *AAATE - 12th European Association for the Advancement of Assistive Technology in Europe - 2013*, Vilamoura, Algarve, Portugal, September 2013.

Séverine Dubuisson, Christophe Gonzales, and Xuan Son Nguyen. DBN-based combinatorial resampling for articulated object tracking. In *Proceedings of the International Conference on Uncertainty in Artificial Intelligence (UAI'12)*, pages 237–246, August 2012.

Jay Earley. An efficient context-free parsing algorithm. *Commun. ACM*, 13(2):94–102, feb 1970.

B. Efron and R. J. Tibshirani. *An Introduction to the Bootstrap*. Chapman & Hall, New York, 1993.

G.A. Einicke. Smoothing, filtering and prediction: Estimating the past, present and future. *New York: InTech*, 2012.

G.A. Einicke and L.B. White. Robust extended Kalman filtering. *IEEE Transactions on Signal Processing*, 47(9):2596–2599, Sep 1999.

Martin Ester, Hans-Peter Kriegel, Jorg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data mining*, pages 226–231. AAAI Press, 1996.

Brett R. Fajen, William H. Warren, Selim Temizer, and Leslie P. Kaelbling. A dynamical model of visually-guided steering, obstacle avoidance, and route selection. *Int. Journal of Computer Vision*, 54:13–34, 2003.

Michael Faraday. Experimental researches in electricity. *Philosophical Transactions of the Royal Society of London*, 122:125–162, 1832.

Ajo Fod, Andrew Howard, and Maja J Mataric. Laser-based people tracking. In *Proc. of the IEEE International Conference on Robotics and Automation (ICRA)*, pages 3024–3029, 2002.

Thomas Fortmann, Yaakov Bar-Shalom, and Molly Scheffé. Sonar tracking of multiple targets using joint probabilistic data association. *IEEE J. Oceanic Eng.*, 8, July 1983.

Oliver Frank, Juan Nieto, Jose Guivant, and Steve Scheding. Multiple target tracking using sequential Monte Carlo methods and statistical data association. In *Proc. of the IEEE / RSJ International Conference on Intelligent Robots and Systems*, 2003.

John Funge, Xiaoyuan Tu, and Demetri Terzopoulos. Cognitive modeling: Knowledge, reasoning and planning for intelligent characters. In *Proceedings of SIGGRAPH '99*, pages 29–38, New York, NY, USA, 1999.

Christopher W. Geib. The intentional planning system: ItPlanS. In *Proceedings of the International Conference on Artificial Intelligence Planning and Scheduling (AIPS)*, pages 55–60. AAAI, 1994.

C.W. Geib, L. Levison, and M.B. Moore. *SodaJack: An Architecture for Agents that Search for and Manipulate Objects.* Technical report. University of Pennsylvania, School of Engineering and Applied Science, Department of Computer and Information Science, 1994.

Nagia M. Ghanem. *Petri Net Models for Event Recognition in Surveillance Videos.* PhD thesis, University of Maryland, College Park, MD, USA, 2007. AAI3260402.

W.R. Gilks and D.J. Spiegelhalter. *Markov chain Monte Carlo in practice.* Chapman & Hall, 1996.

P.G. Gipps and B. Marksjo. A micro-simulation model for pedestrian flows. *Mathematics and Computers in Simulation (MATCOM)*, 27(2):95–105, 1985.

David E. Goldberg and Jon Richardson. Genetic algorithms with sharing for multimodal function optimization. In *Proceedings of the Second International Conference on Genetic Algorithms on Genetic Algorithms and Their Application*, pages 41–49, Hillsdale, NJ, USA, 1987. L. Erlbaum Associates Inc.

Abhinav Gupta and Larry S. Davis. Objects in action: An approach for combining action understanding and object perception. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR).* IEEE Computer Society, 2007.

A. Hanisch, J. Tolujew, K. Richter, and T. Schulze. Online simulation of pedestrian flow in public buildings. In *Proc. of the 2003 Winter Simulation Conference*, volume 2, pages 1635–1641 vol.2, 2003.

David Harel and Amir Pnueli. *On the development of reactive systems.* Springer, 1985.

Ismail Haritaoglu, David Harwood, and Larry S. Davis. W4: Real-time surveillance of people and their activities. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 22:809–830, 2000.

P.E. Hart, N.J. Nilsson, and B. Raphael. A formal basis for the heuristic determination of minimum cost paths. *Systems Science and Cybernetics, IEEE Transactions on*, 4(2):100–107, July 1968.

Johannes Hartz and Bernd Neumann. Learning a knowledge base of ontological concepts for high-level scene interpretation. In *Proceedings of the Sixth International Conference on Machine Learning and Applications*, ICMLA '07, pages 436–443, Washington, DC, USA, 2007. IEEE Computer Society.

W. Hayt and J. Buck. *Engineering Electromagnetics.* McGraw-Hill Education, 2011.

D. Helbing and P. Molnár. Social force model for pedestrian dynamics. *Phys. Rev. E*, 51(5): 4282–4286, may 1995.

Dirk Helbing. A mathematical model for the behavior of pedestrians. *Behavioral Science*, 36, 1991.

Dirk Helbing. A fluid dynamic model for the movement of pedestrians. *Complex Systems*, 6:391 – 415, 1992.

Dirk Helbing and Tamás Vicsek. Optimal self-organization. *New Journal of Physics*, 1999.

L.F. Henderson. On the fluid mechanics of human crowd motion. *Transportation Research*, 8: 509 – 515, 1974.

Stephen S. Intille and Aaron F. Bobick. A framework for recognizing multi-agent action from visual evidence. In *Proceedings of the Sixteenth National Conference on Artificial Intelligence and the Eleventh Innovative Applications of Artificial Intelligence Conference Innovative Applications of Artificial Intelligence*, AAAI '99/IAAI '99, pages 518–525, Menlo Park, CA, USA, 1999. American Association for Artificial Intelligence.

Yuri A. Ivanov and Aaron F. Bobick. Recognition of visual activities and interactions by stochastic parsing. *IEEE Trans. Pattern Anal. Mach. Intell.*, 22(8):852–872, aug 2000.

Tommi S. Jaakkola. Tutorial on variational approximation methods. In *Advanced Mean Field Methods: Theory and Practice*, pages 129–159. MIT Press, 2000.

A. K. Jain, M. N. Murty, and P. J. Flynn. Data clustering: A review. *ACM Comput. Surv.*, 31 (3):264–323, September 1999.

Andrew Jazwinski. *Stochastic Processes and Filtering Theory.* Academic Press, apr 1970.

Harold Jeffreys. *Scientific inference.* Cambridge University Press, 1973.

S. Joo and R. Chellappa. Recognition of multi-object events using attribute grammars. In *Proceedings of IEEE International Conference on Image Processing*, pages 2897–2900, Oct 2006.

Michael I. Jordan, Zoubin Ghahramani, Tommi S. Jaakkola, and Lawrence K. Saul. An introduction to variational methods for graphical models. *Mach. Learn.*, 37(2):183–233, nov 1999.

Pieter Jorissen, Maarten Wijnants, and Wim Lamotte. Dynamic interactions in physically realistic collaborative virtual environments. *IEEE Trans. Vis. Comput. Graph.*, 11(6):649–660, 2005.

Simon J. Julier and Jeffrey K. Uhlmann. A new extension of the Kalman filter to nonlinear systems. In *Proceedings of AeroSense - the 11th International Symposium on Aerospace/Defense Sensing, Simulations and Controls*, pages 182–193, 1997.

S.J. Julier and J.K. Uhlmann. Unscented filtering and nonlinear estimation. *Proceedings of the IEEE*, 92(3):401–422, March 2004.

M. Kallmann and D. Thalmann. Modeling objects for interaction tasks. In *Proc. Eurographics Workshop on Animation and Simulation*, pages 73–86, 1998.

Rudolph Emil Kalman. A New Approach to Linear Filtering and Prediction Problems. *Trans. of the ASME–Journal of Basic Engineering*, 82(Series D):35–45, 1960.

Yan Ke, Rahul Sukthankar, and Martial Hebert. Spatio-temporal shape and flow correlation for action recognition. In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2007.

Zia Khan, Tucker Balch, and Frank Dellaert. Efficient particle filter-based tracking of multiple interacting targets using an MRF-based motion model. In *IEEE Proc. of the International Conference on Intelligent Robots and Systems*, pages 254–259. IEEE, 2003.

Zia Khan, Tucker Balch, and Frank Dellaert. An MCMC-based particle filter for tracking multiple interacting targets. In *Proc. of the European Conference on Computer Vision (ECCV)*, pages 279–290, 2004.

Zia Khan, Tucker Balch, and Frank Dellaert. MCMC-based particle filtering for tracking a variable number of interacting targets. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 27, 2005.

G. Kitagawa. Monte Carlo filter and smoother for non-Gaussian nonlinear state space models. *Journal of computational and graphical statistics*, 5(1):1–25, 1996.

Augustine Kong, Jun S. Liu, and Wing H. Wong. Sequential imputations and Bayesian missing data problems. *Journal of the American Statistical Association*, 89(425):278–288, March 1994.

John Krumm, Steve Harris, Brian Meyers, Barry Brumitt, Michael Hale, and Steve Shafer. Multi-camera multi-person tracking for easyliving. In *Proceedings of the third IEEE International Workshop on visual surveillance*, pages 3–10, 2000.

Yoann Kubera, Philippe Mathieu, and Sébastien Picault. Interaction-oriented agent simulations: From theory to implementation. In *Proceedings of the European Conference on Artificial Intelligence*, pages 383–387, 2008.

Harold W. Kuhn. The hungarian method for the assignment problem. *Naval Research Logistics Quarterly*, 2(1–2):83–97, March 1955.

Taras I. Lakoba, D. J. Kaup, and Neal M. Finkelstein. Modifications of the Helbing-Molnár-Farkas-Vicsek social force model for pedestrian evolution. *Simulation*, 81(5):339–352, 2005.

Oswald Lanz and Roberto Manduchi. Hybrid joint-separable multibody tracking. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 413–420. IEEE Computer Society, 2005.

Laura Leal-Taixé, Gerard Pons-Moll, and Bodo Rosenhahn. Everybody needs somebody: Modeling social and grouping behavior on a linear programming multiple people tracker. In *ICCV Workshops*, pages 120–127. IEEE, 2011.

François LeGland, Christian Musso, and Nadia Oudjane. An analysis of regularized interacting particle methods for nonlinear filtering. In *Proceedings of the 3rd IEEE European Workshop on Computer-Intensive Methods in Control and Signal Processing*, pages 167–174, 1998.

Stan Z. Li. *Markov Random Field Modeling in Image Analysis*. Springer Publishing Company, Incorporated, 3rd edition, 2009.

Lin Liao, Dieter Fox, Jeffrey Hightower, Henry Kautz, and Dirk Schulz. Voronoi tracking: Location estimation using sparse and noisy sensor data. In *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2003.

Jingen Liu, Jiebo Luo, and Mubarak Shah. Recognizing realistic actions from videos "in the wild". In *Proc. of IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, 2009.

Juan Liu, Chu M., and Reich J.E. Multi-target Tracking in distributed sensor networks. *Signal Processing Magazine, IEEE*, 24(3):36–46, 2007.

Jun S. Liu and Rong Chen. Blind deconvolution via sequential imputations. *Journal of the American Statistical Association*, 90:567–576, 1995.

Jun S. Liu and Rong Chen. Sequential Monte Carlo methods for dynamic systems. *Journal of the American Statistical Association*, 93:1032–1044, 1998.

S. Lloyd. Least squares quantization in PCM. *IEEE Trans. Inf. Theor.*, 28(2):129–137, March 1982.

M. Luber, J.A. Stork, G.D. Tipaldi, and K.O. Arras. People tracking with human motion predictions from social forces. In *proc. of the IEEE Int. Conf. on Robotics and Automation*, pages 464–469, 2010.

J. B. MacQueen. Some methods for classification and analysis of multivariate observations. In L. M. Le Cam and J. Neyman, editors, *Proceedings of the fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pages 281–297. University of California Press, 1967.

Pattie Maes. Modeling adaptive autonomous agents. *Artificial Life*, 1:135–162, 1994.

P. C. Mahalanobis. On the generalised distance in statistics. In *Proceedings National Institute of Science, India*, volume 2, pages 49–55, April 1936.

Samir W. Mahfoud. *Niching Methods for Genetic Algorithms.* PhD thesis, University of Illinois, 104 South Mathews Avenue, Urbana - Illinois, 1995.

Udi Manber and Gene Myers. Suffix arrays: a new method for on-line string searches. *SIAM (Society for Industrial and Applied Mathematics) Journal on Computing*, 22(5):935–948, 1993.

Cristina Manfredotti, David J. Fleet, Howard J. Hamilton, and Sandra Zilles. Simultaneous tracking and activity recognition. In *Proceedings of the International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 189–196, Washington, DC, USA, 2011.

A. A. Markov. The theory of algorithms. *Trudy Mat. Inst. Steklov*, 42:3–375, 1954.

J.-A. Meyer. From natural to artificial life: Biomimetic mechanisms in animat designs. *Robotics and Autonomous Systems*, 22:3–21, 1997.

Jean-Arcady Meyer. The animat approach: Simulation of adaptive behavior in animals and robots. In *NPI*, pages 1–21, 1998.

D. Minnen, I Essa, and T. Starner. Expectation grammars: leveraging high-level expectations for activity recognition. In *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2, pages 626–632, June 2003.

Darnell Moore and Irfan Essa. Recognizing multitasked activities from video using stochastic context-free grammar. In *Proc. of the Eighteenth National Conference on Artificial Intelligence*, pages 770–776, Menlo Park, CA, USA, 2002. American Association for Artificial Intelligence.

Darnell J. Moore, Irfan A. Essa, and Monson H. Hayes. Exploiting human actions and object context for recognition tasks. In *Proceedings of the International Conference on Computer Vision (ICCV)*, pages 80–86, 1999.

D. Musicki and Rob Evans. Joint integrated probabilistic data association: Jipda. *IEEE Transactions Aerospace and Electronic Systems*, 40(3):1093–1099, July 2004.

Christian Musso, Nadia Oudjane, and Francois Le Gland. Improving regularised particle filters. In *Sequential Monte Carlo methods in practice*, pages 247–271. New-York - Springer-Verlag, 2001.

A. Nareyek. Intelligent agents for computer games. In *Second International Conference on Computers and Games*, volume 2063, pages 414–422, 2000.

Laurent Navarro, Fabien Flacher, and Vincent Corruble. Dynamic level of detail for large scale agent-based urban simulations. In *Proc. of the International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS)*, pages 701–708, 2011.

Laurent Navarro, Fabien Flacher, and Christophe Meyer. SE-Star: A large-scale human behavior simulation for planning, decision-making and training. In *Proceedings of the International Conference on Autonomous Agents and Multi-agent Systems, (AAMAS)*, pages 1939–1940, 2015.

Bernd Neumann and Ralf Möller. On scene interpretation with description logics. *Image Vision Comput.*, 26(1):82–101, January 2008.

Ram Nevatia, Tao Zhao, and Somboon Hongeng. Hierarchical language-based representation of events in video streams. In *IEEE Workshop on Event Mining*, 2003.

Ram Nevatia, Jerry Hobbs, and Bob Bolles. An ontology for video event representation. In *Computer Vision and Pattern Recognition*, Washington, DC, USA, 2004. IEEE.

Nam T. Nguyen, Dinh Q. Phung, Svetha Venkatesh, and Hung Bui. Learning and detecting activities from movement trajectories using the hierarchical hidden Markov models. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, volume 2 of *CVPR '05*, pages 955–960, Washington, DC, USA, 2005. IEEE Computer Society.

Juan Carlos Niebles, Hongcheng Wang, and Li Fei-Fei. Unsupervised learning of human action categories using spatial-temporal words. *Int. J. Comput. Vision*, 79(3):299–318, September 2008.

Peter Nillius, Josephine Sullivan, and Stefan Carlsson. Multi-target tracking - linking identities using Bayesian network inference. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2187–2194. IEEE Computer Society, 2006.

Songhwai Oh, Stuart J. Russell, and Shankar Sastry. Markov chain Monte Carlo data association for multi-target tracking. *IEEE Trans. Automat. Contr.*, 54(3):481–497, 2009.

Shigeyuki Okazakia and Satoshi Matsushitaa. A study of simulation model for pedestrian movement with evacuation and queuing. In *Proceedings of the International Conference on Engineering for Crowd Safety*, pages 271–280, March 1993.

Nuria Oliver, Eric Horvitz, and Ashutosh Garg. Layered representations for human activity recognition. In *Proceedings of the 4th IEEE International Conference on Multimodal Interfaces*, ICMI '02, pages 3–, Washington, DC, USA, 2002. IEEE Computer Society.

Toshihiro Osaragi. Modeling of pedestrian behavior and its applications to spatial evaluation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 2*, AAMAS '04, pages 836–843, Washington, DC, USA, 2004. IEEE Computer Society.

Sangho Park and J. K. Aggarwal. Recognition of two-person interactions using a hierarchical Bayesian network. In *First ACM SIGMM International Workshop on Video Surveillance*, IWVS '03, pages 65–76, New York, NY, USA, 2003. ACM.

Ann Parker. *Dealing with data overload in the scientific realm.* Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550, Jan 2013. URL https://str.llnl.gov/january-2013/williams.

Philippa Pattison and Garry Robins. Neighborhood-based models for social networks. *Sociological Methodology*, 32(1):301–337, 2002.

Judea Pearl. Reverend Bayes on inference engines: a distributed hierarchical approach. In *Proceedings of the National Conference on Artificial Intelligence*, pages 133–136, 1982.

Judea Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference.* Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988. ISBN 0-934613-73-7.

S. Pellegrini, A. Ess, K. Schindler, and L. Van Gool. You'll never walk alone: Modeling social behavior for multi-target tracking. In *Proc. IEEE 12th Int. Conf. on Computer Vision*, pages 261–268, 2009.

Carl Adam Petri. *Communication with automata*. PhD thesis, Universitat Hamburg, Hamburg, 1966.

A. Petrowski. A clearing procedure as a niching method for genetic algorithms. In *Proceedings of IEEE International Conference on Evolutionary Computation*, pages 798–803, May 1996.

Justus Piater and James Crowley. Multi-modal tracking of interacting targets using Gaussian approximations. In *proceedings of the second IEEE International Workshop on Performance Evaluation of Tracking and Surveillance (PETS)*, volume 14, page 58, 2001.

Joelle Pineau, Geoff Gordon, Sebastian Thrun, et al. Point-based value iteration: An anytime algorithm for POMDPs. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 3, pages 1025–1032, 2003.

C. S. Pinhanez and A. F. Bobick. Human action detection using PNF propagation of temporal constraints. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, CVPR '98, pages 898–, Washington, DC, USA, 1998. IEEE Computer Society.

Wei Qu, Dan Schonfeld, and Magdi A. Mohamed. Real-time distributed multi-object tracking using multiple interactive trackers and a magnetic-inertia potential model. *IEEE Transactions on Multimedia*, 9(3):511–519, April 2007.

Donald B. Reid. An algorithm for tracking multiple targets. *IEEE Trans. Automat. Contr.*, 24: 843–854, 1979.

Craig Reynolds. Steering Behaviors for Autonomous Characters. In *Game Developers Conference 1999*, pages 763–782, 1999.

J.A. Roecker and G.L. Phillis. Suboptimal joint probabilistic data association. *Aerospace and Electronic Systems, IEEE Transactions on*, 29(2):510–517, 1993.

Nathanael Rota and Monique Thonnat. Activity recognition from video sequences using declarative models. In *Proc. of the European Conference on Artificial Intelligence (ECAI)*, pages 673–680. IOS Press, 2000.

M. S. Ryoo and J. K. Aggarwal. Recognition of composite human activities through context-free grammar based representation. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 1709–1718, Washington, DC, USA, 2006. IEEE Computer Society.

M. S. Ryoo and J. K. Aggarwal. Hierarchical recognition of human activities interacting with objects. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. IEEE Computer Society, 2007.

Shlomo Sawilowsky. You think you have got trivials? In *Journal of Modern Applied Statistical Methods*, pages 218–225, 2003.

J. Schlenzig, E. Hunter, and R. Jain. Recursive identification of gesture inputs using hidden Markov models. In *Applications of Computer Vision, 1994., Proceedings of the Second IEEE Workshop on*, pages 187–194, Dec 1994.

Dirk Schulz, Wolfram Burgard, Dieter Fox, and Armin B. Cremers. People tracking with a mobile robot using sample-based joint probabilistic data association filters. *Int. Journal of Robotics Research*, 22(2):99–116, 2003.

Wei Shao and Demetri Terzopoulos. Autonomous pedestrians. *Graph. Models*, 69(5-6):246–274, sep 2007.

Yaser Sheikh, Mumtaz Sheikh, and Mubarak Shah. Exploring the space of a human action. In *Proceedings of the Tenth IEEE International Conference on Computer Vision*, pages 144–149, Washington, DC, USA, 2005. IEEE Computer Society.

Yifan Shi, Yan Huang, David Minnen, Aaron Bobick, and Irfan Essa. Propagation networks for recognition of partially ordered sequential action. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 862–869, Washington, DC, USA, 2004. IEEE Computer Society.

Yifan Shi, Aaron Bobick, and Irfan Essa. Learning temporal sequence model from partially labeled data. In *Proceedings of the 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition - Volume 2*, CVPR '06, pages 1631–1638, Washington, DC, USA, 2006. IEEE Computer Society.

R. Sibson. SLINK: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

B. W. Silverman and P. J. Green. *Density Estimation for Statistics and Data Analysis*. Chapman and Hall, London, 1986.

Robert W. Sittler. An optimal data association problem in surveillance theory. *IEEE Transactions on Military Electronics*, 8(2):125–139, 1964.

Xuan Song, Jinshi Cui, Hongbin Zha, and Huijing Zhao. Vision-based multiple interacting targets tracking via on-line supervised learning. In *Proceedings of the 10th European Conference on Computer Vision: Part III*, ECCV '08, pages 642–655, Berlin, Heidelberg, 2008. Springer-Verlag.

Soteris Stylianou, Marios M. Fyrillas, and Yiorgos Chrysanthou. Scalable pedestrian simulation for virtual cities. In *Proc. of the ACM symposium on Virtual reality software and technology*, pages 65–72, New York, NY, USA, 2004. ACM.

Josephine Sullivan and Stefan Carlsson. Tracking and labelling of interacting multiple targets. In *Proc. of the European Conference on Computer Vision (3)*, volume 3953 of *Lecture Notes in Computer Science*, pages 619–632. Springer, 2006.

C. Suppitaksakul, G. Sexton, and P.D. Minns. A pedestrian tracking using the association of cellular automata and a neural network. In *Proceeding of Fifth International Symposium-Communication systems networks and digital signal processing*, pages 19–21, 2006.

Bulent Tastan and Gita Sukthankar. Leveraging human behavior models to predict paths in indoor environments. *Pervasive Mobile Computing*, 7(3):319–330, jun 2011.

K. Teknomo. *Microscopic pedestrian flow characteristics: Development of an image processing data collection and simulation model.* PhD thesis, Tohoku University, Japan, 2002.

Christopher Town. Ontology-driven Bayesian networks for dynamic scene understanding. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop - Volume 7*, CVPRW '04, pages 116–123, Washington, DC, USA, 2004. IEEE Computer Society.

Son D. Tran and Larry S. Davis. Event modeling and recognition using markov logic networks. In *Proceedings of the 10th European Conference on Computer Vision: Part II*, ECCV '08, pages 610–623, Berlin, Heidelberg, 2008. Springer-Verlag.

Xiaoyuan Tu and Demetri Terzopoulos. Artificial fishes: Physics, locomotion, perception, behavior. In *Proc. of SIGGRAPH '94*, pages 43–50, New York, NY, USA, 1994.

Toby Tyrell. Defining the action selection problem. In *Proc. of the 14th Annual conference on cognitive science society*. Lawrence Erlbaum Associates, 1993.

Jaco Vermaak, Simon J. Godsill, and Patrick Perez. Monte Carlo filtering for multi-target tracking and data association. *IEEE Trans. on Aerospace and Electronic Systems*, 41:309–332, 2005.

Giuseppe Vizzari and Lorenza Manenti. An agent-based model for pedestrian and group dynamics: Experimental and real-world scenarios. In *Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems - Volume 3*, AAMAS '12, pages 1341–1342, 2012.

John von Neumann and Stanislav Ulam. Games of chance to study interesting phenomena. *Los Alamos Science*, 1940.

Van-Thinh Vu, François Brémond, and Monique Thonnat. Automatic video interpretation: A recognition algorithm for temporal scenarios based on pre-compiled scenario models. In *Proceedings of the 3rd International Conference on Computer Vision Systems*, ICCVS'03, pages 523–533, Berlin, Heidelberg, 2003a. Springer-Verlag.

Van-Thinh Vu, Francois Bremond, and Monique Thonnat. Automatic video interpretation: A novel algorithm for temporal scenario recognition. In *Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI)*, volume 3, pages 1295–1300, 2003b.

Jan Oliver Wallgrün. Autonomous construction of hierarchical Voronoi-based route graph representations. In *Spatial Cognition IV. Reasoning, Action, Interaction*, pages 413–433. Springer, 2005.

Zhaowen Wang, Ercan E Kuruoğlu, Xiaokang Yang, Yi Xu, and Thomas S Huang. Time varying dynamic Bayesian network for nonstationary events modeling and online inference. *IEEE Transactions on Signal Processing*, 59(4):1553–1568, 2011.

D. H. Wilson and C. Atkeson. Simultaneous tracking and activity recognition (STAR) using many anonymous, binary sensors. In *Proc. of the 3rd Int. conf. on Pervasive Computing*, pages 62–79, Berlin, Heidelberg, 2005. Springer-Verlag.

S. Wolfram. *Theory and Applications of Cellular Automata*. World Scientific, Singapore, 1986.

J. Yamato, Jun Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden Markov model. In *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 379–385, Jun 1992.

Lizhong Yang, Weifeng Fang, Jian Li, Rui Huang, and Weicheng Fan. Cellular automata pedestrian movement model considering human behavior. *Chinese Science Bulletin*, 48(16):1695–1699, 2003.

Ting Yu and Ying Wu. Collaborative tracking of multiple targets. In *Proc. of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 834–841, 2004.

Nadia Zouba, Francois Bremond, and Monique Thonnat. Multisensor fusion for monitoring elderly activities at home. In *Proceedings of the sixth IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 98–103. IEEE, September 2009.

Nadia Zouba, Francois Bremond, and Monique Thonnat. An activity monitoring system for real elderly at home: Validation study. In *Proceedings of the seventh IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 278–285, 2010.

Appendix A

Résumé étendu

Contents

A.1 Introduction

Dans ce mémoire, nous nous intéressons au problème d'inférence de comportements des piétons dans un environnement sous couverture sensorielle partielle. Ce problème, d'une manière générale, appartient à la catégorie des problèmes consistant à déterminer l'état d'un système dynamique partiellement observé sur la seule base des observations — généralement bruitées — fournies par un réseau de capteurs connectés au système. Dans la littérature, cette classe de problèmes est connue sous le nom de *problème de filtrage*. En outre, lorsque l'objectif est d'estimer les comportements d'un ensemble de cibles (dans notre cas, piétons), le terme *suivi comportemental* peut être utilisé pour désigner le problème de filtrage correspondant. Dans un tel cas, le système dynamique est constitué d'un ensemble de cibles évoluant dans l'environnement au cours du temps.

Le *suivi comportemental* des piétons est généralement un problème difficile même dans le cas particulier où le nombre de cibles est réduit à une seule unité. Cette difficulté est probablement parce que le comportement adopté par un piéton, en plus d'être *individuel*, est *adaptatif*. Par *individuel*, nous entendons qu'un tel comportement dépend des propriétés internes (ou personnelles) caractérisant le piéton considéré telles que son endurance, son âge et/ou son niveau d'éducation. Par exemple, dans une station de métro, un passager moins civique serait susceptible de tricher sur les barrières à tickets.

Par *adaptatif*, nous signifions qu'un tel comportement est influencée par le *contexte environnemental* dans lequel le piéton est immergé. Le contexte d'un environnement est caractérisé en grande partie des objets qui s'y trouvent. Ces objets, du simple fait de leur présence, influencent les comportements des différents piétons (ex., un piéton ne peut prélever de l'argent que s'il y a un distributeur automatique dans le voisinage). En plus, le contexte environnemental peut être sujet à des changements dynamiques — causés soit par les actions initiées par les piétons, soit par des événements exogènes — qui peuvent avoir une incidence sur les comportements des piétons (ex., alarme incendie).

En plus, dans le cas général où plusieurs cibles sont considérées, le comportement d'un piéton peut également être influencé par la présence d'autres piétons dans son voisinage. Par exemple, dans une station de métro, un passager souhaitant tricher sur les barrières à tickets se raviserait probablement en présence d'agents de contrôle. Nous utilisons le terme générique *interactions entre cibles* pour désigner cette forme d'influences comportementales entre piétons. Ces interactions complexifient davantage le problème du suivi comportemental. En effet, il est impératif de raisonner sur l'ensemble des comportements des différents piétons plutôt que de décomposer le problème en se focalisant sur chaque piéton individuellement. Ceci peut rapidement s'avérer computationnellement impraticable si aucune attention n'est prise lors de la conception d'une solution au problème du suivi comportemental.

A.1.1 Aperçu de la littérature et limites actuelles

De la description faite ci-dessus, une solution au problème du suivi comportemental pourrait, en grosso modo, être évaluée qualitativement sur la base des critères suivants:

- **Caractérisation des piétons**: la solution doit offrir une interface permettant de définir des propriétés spécifiques liées à chaque piéton.

- **Modélisation contextuelle**: la solution doit être en mesure (1) de représenter et (2) de raisonner sur le contexte environnemental dans lequel les piétons sont immergés. Cela comprend, entre autres, les objets présents dans l'environnement, le service qu'ils offrent ainsi que leur utilisation par un piéton donné.

- **Changements environnementaux**: la solution doit pouvoir intégrer dans son processus d'inférence des modifications dynamiques du contexte environnemental.

- **Gestion des interactions entre cibles**: la solution doit être capable de gérer efficacement les interactions entre piétons tout en conservant la charge computationnelle à un niveau raisonnable.

Dans la littérature, les problèmes de filtrage ont généralement été abordés dans le cadre du filtrage Bayésien[Diard et al., 2003] et plusieurs solutions ont été proposées pour traiter le problème du suivi comportemental [Ke et al., 2007, Aggarwal and Ryoo, 2011]. Deux aspects principaux du comportement des piétons sont souvent considérés. Le premier concerne la *trajectoire* suivie par le piéton considéré tandis que le second aspect est lié à l' *activité*, au sens général, réalisée par celui-ci au sein de l'environnement.

Une grande partie des solutions trouvées dans la littérature se focalise uniquement sur l'un de ces deux aspects et, par conséquent, ces solutions sont intéressées soit au *suivi de trajectoire* [Khan et al., 2003, Pellegrini et al., 2009, Luber et al., 2010, Tastan and Sukthankar, 2011], soit à *la reconnaissance d'activité* [Dollar et al., 2005, Niebles et al., 2008, Liu et al., 2009]. Toutefois, des approches [Wilson and Atkeson, 2005, Manfredotti et al., 2011, Cattelani et al., 2014] ont été proposées, sous le sigle STAR (*Simultaneous tracking and activity recognition — Suivi de trajectoire et reconnaissance d'activité en simultané*), dans lesquelles il est possible traiter simultanément ces deux aspects. La caractéristique principale d'une approche STAR réside sur le fait que les déplacement des piétons sont généralement dictés par une motivation interne en relation avec l'activité qu'ils souhaitent réaliser dans l'environnement. Par conséquent, il est possible d'améliorer l'estimation de la trajectoire d'un individu sur la base des informations contextuelles concernant son activité et réciproquement.

Cependant, sur la base des critères mentionnés ci-dessus, toutes ces méthodes présentent au moins l'une des limites suivantes:

- caractérisation des piétons: les piétons sont caractérisés par des propriétés basiques (ex., position, vitesse, activité) et aucune interface n'est fournie pour représenter les caractéristiques internes (ex., endurance) relatives à la nature intrinsèque de chaque individu [Pellegrini et al., 2009, Tastan and Sukthankar, 2011];

- modélisation contextuelle: la représentation du contexte environnemental est plutôt limitée. Ils fournissent en général peu de détails pour raisonner explicitement sur des objets présents dans l'environnement, en particulier, sur les services qu'ils proposent et sur leur utilisation[Wilson and Atkeson, 2005, Niebles et al., 2008, Manfredotti et al., 2011];

- changements environnementaux: la plupart des solutions sont généralement conçus pour faire face aux situations dans lesquelles le contexte environnemental est supposé statique au fil du temps [Dollar et al., 2005, Wilson and Atkeson, 2005, Luber et al., 2010];

- gestion des interactions entre cibles: on distingue deux catégories. La première concerne les

méthodologies qui s'appuient sur des hypothèses spécifiques au problème traité concernant la nature des interactions entre cibles. Bien que ces hypothèses permettent de gérer les interactions avec une faible charge computationnelle, les solutions dérivées ne sont généralement pas applicable à d'autres problèmes [Khan et al., 2003]. D'autre part, les approches appartenant à la seconde catégorie intègrent des modèles d'interactions plus génériques. Cependant, ces approches affichent généralement une charge computationnelle élevée, leur rendant impraticable en présence de plus de 5 cibles [Cattelani et al., 2014].

A.1.2 Contributions

Dans cette thèse, nous souhaitons développer une solution générique au problème du suivi comportemental capable de faire face aux limites énoncées ci-dessus. À cette fin, nous articulons notre travail autour de deux axes principaux.

Le premier axe se focalise sur les questions liées aux trois premiers critères introduits précédemment (la caractérisation des piétons, la modélisation contextuelle, et les changements environnementaux). Comme décrit dans la littérature, nous partageons l'idée de réaliser simultanément le suivi de trajectoire et la reconnaissance d'activité. Cependant, contrairement aux approches existantes, nous proposons une solution innovante — dans le cadre du filtrage Bayésien — dans laquelle des simulateurs avancés de comportements à base d'agents [Shao and Terzopoulos, 2007] sont utilisés lors du processus d'inférence. Ces simulateurs, d'une manière générale, visent à reproduire de façon réaliste les comportements humains dans des environnements virtuels en utilisant des agents autonomes dotés de sens de perception et d'un moteur "cognitif". Grâce à ces simulateurs, il est possible de:

- représenter chaque piéton par un agent virtuel et définir, outre les propriétés basiques tels que la position et/ou la vitesse, des propriétés internes de haut niveau pouvant influencer son comportement (ex., niveau de soif);
- raisonner sur des modèles contextuels plus riches en représentant virtuellement, dans le simulateurs, l'environnement considéré ainsi que les différents objets qui s'y trouvent;
- simuler les changements dynamiques intervenant dans l'environnement réel au sein du simulateur de façon à tenir compte de leurs effets sur les comportements des agents virtuels.

Le deuxième axe porte sur la gestion des interactions entre cibles. Contrairement aux approches existantes, nous cherchons une solution reposant sur un modèle générique d'interactions, tout en évitant, dans le même temps, les questions liées à la complexité de calcul. Nous proposons un algorithme factorisée pouvant être vue comme une collection de plusieurs processus *collaboratifs* de filtrage, chacun dédié à une cible unique. Dans notre solution, afin de réduire la complexité sans trop dégrader la qualité des résultats, nous concevons des heuristiques pour agréger, en quelques états représentatifs (ou *représentants*) la croyance relative au comportement de chaque cible de sorte que ces représentants soient les seules informations nécessaires à chaque processus de filtrage pour la mise à jour des estimations concernant le comportement de la cible associée.

A.1.3 Plan

Ce mémoire (et son résumé) introduit d'abord aux chapitres 2-3 (Section A.2) le problème de filtrage, et donc des méthodologies classiques dans le cadre Bayésien que sont le filtre de Kalman et le filtre particulaire. Ensuite, les chapitres 4-5 (Section A.3) s'intéressent au problème du suivi comportemental dans le cas spécial d'une cible unique et démontrent les avantages liés à l'utilisation des simulateurs avancés de comportements dans le cadre du paradigme STAR. Les chapitres 6-7 (Section A.4) sont consacrés au cas général du suivi multi-cibles et décrivent des solutions permettant de gérer efficacement les interactions entre cibles dans un processus d'inférence comportementale.

A.2 Le problème de filtrage

Un système dynamique est un système dont l'état interne, représenté par une variable aléatoire $\mathbf{x}$, évolue au fil du temps de manière causale, c-à-d, l'état actuel du système dépend seulement de ses états antérieures. Lorsque cette dépendance est restreinte uniquement à l'état au pas de temps précédent, on dit que le système vérifie la propriété de Markov [Markov, 1954]. Dans ce mémoire, nous nous intéressons aux systèmes vérifiant cette propriété. En particulier, on se focalise sur des systèmes dont la dynamique peut s'exprimer sous la forme

$$\mathbf{x}_t = \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{w}_t), \forall t \geq 1, \tag{A.1}$$

où $\mathbf{x}_t$ et $\mathbf{x}_{t-1}$ sont respectivement l'état du système aux pas de temps t et $t-1$, $\mathbf{f}$ est une fonction déterministe modélisant la dynamique du système, et $(\mathbf{w}_t)_{t\geq 1}$ est un bruit représentant des facteurs exogènes pouvant influencer l'évolution du système. D'un point de vue probabiliste, cette dynamique peut être caractérisée par une distribution de probabilité $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ aussi appelée le *modèle de transition d'états*. En outre, une connaissance a-priori Bel_0 relative à l'état initial du système (au pas de temps $t = 0$) est souvent disponible sous la forme d'une distribution de probabilité $p(\mathbf{x}_0)$. Formellement, $p(\mathbf{x}_0)$ définit l'ensemble des valeurs possibles de l'état initial du système ainsi que leur probabilité.

Généralement, l'état $\mathbf{x}$ n'est pas complètement accessible et la seule information disponible (aussi appelée observation ou mesure) concernant le système, représentée par une variable aléatoire $\mathbf{z}$, s'obtient au moyen d'un réseau de capteurs connectés au système. On dit alors que le système est partiellement observé. Au pas de temps t, l'information $\mathbf{z}_t$ reçue du réseau de capteurs peut s'exprimer sous la forme

$$\mathbf{z}_t = \mathbf{h}(\mathbf{x}_t, \mathbf{v}_t), \forall t \geq 1, \tag{A.2}$$

où h est une fonction d'observation déterministe caractérisant le réseau de capteurs et $\mathbf{v}_t$ est un bruit — indépendant de $\mathbf{w}_t$ — qui représente les imperfections des capteurs utilisés. La Figure A.1 montre une représentation graphique d'un système dynamique partiellement observé. De même, une probabilité de distribution $p(\mathbf{z}_t|\mathbf{x}_t)$, aussi appelée *modèle d'observation*, peut être utilisée pour caractériser l'équation A.2.

Sur la base de cette description, le problème de filtrage consiste à inférer l'état caché d'un système dynamique partiellement observé en s'appuyant notamment sur les observations reçues du réseau des capteurs et de la connaissance a-priori du système. En d'autre termes, l'objectif est

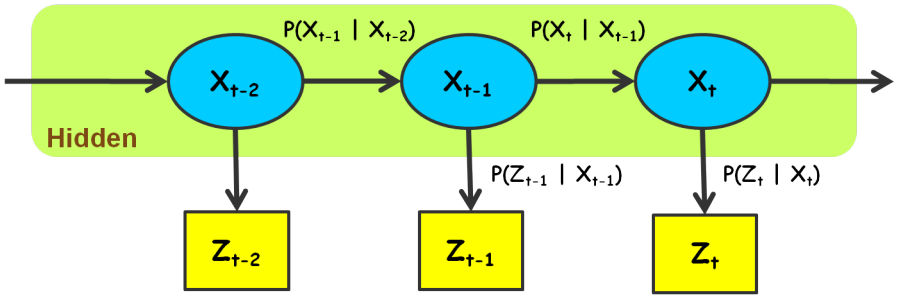


Figure A.1: Système dynamique partiellement observé.

de calculer la distribution de probabilité Bel_t (aussi appelée croyance) relative à l'état courant du système $\mathbf{x}_t$ en fonction de tous les observations passées $\mathbf{z}_{1:t}$ et de Bel_0. on a

$$Bel_t(\mathbf{x}_t) = p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0),\ \forall t \geq 1. \tag{A.3}$$

Nous allons maintenant présenter un ensemble de méthodologies permettant d'adresser le problème de filtrage.

A.2.1 Le filtre Bayésien

Le filtre Bayésien [Jeffreys, 1973] est une méthode générique permettant d'adresser de façon optimale le problème de filtrage. Il s'appuie sur le théorème de Bayes qui décrit comment de nouvelles évidences peuvent être utilisées pour mettre à jour des croyances précédentes. Le filtre Bayésien est une méthode récursive dans la mesure où la croyance $Bel_t(\mathbf{x}_t)$ au pas de temps t est calculée à partir de la croyance $Bel_{t-1}(\mathbf{x}_{t-1})$ au pas de temps précédent. Pour ce faire, le filtre Bayésien procède en deux étapes dites de *prédiction* et de *correction* à chaque pas de temps.

A l'étape de prédiction, la croyance précédente $Bel_{t-1}(\mathbf{x}_{t-1})$ est modifiée sur la base de la dynamique du système $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ permettant ainsi d'obtenir la *croyance prédite* $Bel_{t|t-1}(\mathbf{x}_t)$ de la manière suivante:

$$\begin{aligned} Bel_{t|t-1}(\mathbf{x}_t) &= p(\mathbf{x}_t|\mathbf{z}_{1:t-1}, Bel_0) \\ &= \int p(\mathbf{x}_t|\mathbf{x}_{t-1}) Bel_{t-1}(\mathbf{x}_{t-1}) d\mathbf{x}_{t-1}. \end{aligned} \tag{A.4}$$

A l'étape de correction, l'observation $\mathbf{z}_t$ reçue du réseau des capteurs est utilisée pour raffiner la $Bel_{t|t-1}(\mathbf{x}_t)$ permettant ainsi d'obtenir $Bel_t(\mathbf{x}_t)$ de la manière suivante:

$$\begin{aligned} Bel_t(\mathbf{x}_t) &= p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0) \\ &= \frac{p(\mathbf{x}_t, \mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)} \\ &= \frac{p(\mathbf{z}_t|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{z}_{1:t-1}, Bel_0)}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)} \\ &= \frac{1}{p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0)} p(\mathbf{z}_t|\mathbf{x}_t) Bel_{t|t-1}(\mathbf{x}_t), \end{aligned} \tag{A.5}$$

où $p(\mathbf{z}_t|\mathbf{z}_{1:t-1}, Bel_0) = \int p(\mathbf{z}_t|\mathbf{x}_t) Bel_{t|t-1}(\mathbf{x}_t) d\mathbf{x}_t$ est une constante de normalisation.

L'application du filtre Bayésien nécessitent le calculs des intégrales qui sont généralement fastidieux à effectuer dans les cas où (1) les variables sont définies dans des espaces multidimensionnelles et/ou (2) la dynamique du système considéré est fortement non-linéaire. Néanmoins, des solutions ont été dérivées afin d'approximer efficacement le filtre Bayésien sous certaines conditions particulières. Parmi eux, on peut citer le très populaire filtre de Kalman (KF) [Kalman, 1960] qui est utilisé pour calculer analytiquement la croyance $Bel_t(\mathbf{x}_t)$ lorsque le système considéré est linéaire avec des bruits Gaussiens additifs. Un autre dérivé est le filtre particulaire qui est généralement utilisé lorsque la dynamique du système est non-linéaire. Dans ce qui suit, nous présentons les détails du filtre de Kalman.

A.2.2 Le filtre de Kalman

Dans cette section, nous faisons l'hypothèse selon laquelle le système considéré est linéaire avec des bruit Gaussiens. En d'autres termes, les équations A.1 et A.2 peuvent être réécrites comme

$$\begin{aligned}\mathbf{x}_t &= \mathbf{f}(\mathbf{x}_{t-1}, \mathbf{w}_t) \\ &= \mathbf{F}\mathbf{x}_{t-1} + \mathbf{w}_t, \forall t \geq 1,\end{aligned} \tag{A.6}$$

$$\begin{aligned}\mathbf{z}_t &= \mathbf{h}(\mathbf{x}_t, \mathbf{v}_t) \\ &= \mathbf{H}\mathbf{x}_t + \mathbf{v}_t, \forall t \geq 1,\end{aligned} \tag{A.7}$$

où $\mathbf{F}$ et $\mathbf{H}$ sont des matrices caractérisant respectivement la dynamique du système et le modèle d'observation du réseau des capteurs, $\mathbf{w}_t \sim \mathcal{N}(0, \mathbf{Q}_t)$ (resp. $\mathbf{v}_t \sim \mathcal{N}(0, \mathbf{R}_t)$) est issu d'une distribution Gaussienne de moyenne nulle et de covariance $\mathbf{Q}_t$ (resp. $\mathbf{R}_t$).

En supposant que $Bel_{t-1}(\mathbf{x}_{t-1}) = \mathcal{N}(\mathbf{x}_{t-1|t-1}, \Sigma_{t-1|t-1})$, le filtre de Kalman procède de manière récursive via deux étapes comme dans le filtre Bayésien:

- l'étape de prédiction:

$$\begin{aligned}Bel_{t|t-1}(\mathbf{x}_t) &= \mathcal{N}(\mathbf{x}_{t|t-1}, \Sigma_{t|t-1}) \text{ où} \\ \mathbf{x}_{t|t-1} &= \mathbf{F}\mathbf{x}_{t-1|t-1}, \text{ et} \\ \Sigma_{t|t-1} &= \mathbf{Q}_t + \mathbf{F}\Sigma_{t-1|t-1}\mathbf{F}^T;\end{aligned} \tag{A.8}$$

- l'étape de correction:

$$\begin{aligned}Bel_t(\mathbf{x}_t) &= \mathcal{N}(\mathbf{x}_{t|t}, \Sigma_{t|t}) \text{ où} \\ \mathbf{x}_{t|t} &= \mathbf{x}_{t|t-1} + \mathbf{K}_t(\mathbf{z}_t - \mathbf{H}\mathbf{x}_{t|t-1}), \\ \Sigma_{t|t} &= \Sigma_{t|t-1} - \mathbf{K}_t\mathbf{H}\Sigma_{t|t-1}, \text{ et} \\ \mathbf{K}_t &= \Sigma_{t|t-1}\mathbf{H}^T[\mathbf{H}\Sigma_{t|t-1}\mathbf{H}^T + \mathbf{R}_t]^{-1}.\end{aligned} \tag{A.9}$$

Dans les équations ci-dessus, $\mathbf{K}_t$ est appelé le gain de Kalman. Il est intéressant de souligner que les matrices de covariance $\Sigma_{t|t-1}$ et $\Sigma_{t|t}$ ainsi que $\mathbf{K}_t$ ne dépendent pas de l'observation $\mathbf{z}_t$ et par conséquent, ils peuvent être pré-calculés afin d'accélérer le processus d'inférence.

En pratique, plusieurs problèmes, y compris celui qui nous intéresse dans cette thèse, concernent des systèmes non-linéaires. Malheureusement, le filtre de Kalman ne convient pas lorsque la

linéarité du système n'est pas garantie. Des solutions approximatives ont été développées pour adresser le problème de filtrage en présence des systèmes non-linéaires. C'est notamment le cas des filtres particulaires qui sont des approches inspirées des méthodes de Monte Carlo.

A.2.3 Le filtre particulaire

Selon Sawilowsky [Sawilowsky, 2003], une méthode de Monte Carlo est une technique qui permet de résoudre un problème mathématique ou statistique en s'appuyant sur des échantillonnages aléatoires répétés. Le filtre particulaire (PF) est une approximation du filtre Bayésien qui repose sur le principe des méthodes de Monte Carlo. Formellement, le PF modélise, à chaque pas de temps t, la croyance $Bel_t(\mathbf{x}_t)$ à l'aide d'un ensemble $\mathcal{S}_t = \{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$ de points pondérés aussi appelés particules. Chaque particule $\mathbf{x}_t^i$ est une hypothèse de l'état actuel du système et le poids associé w_t^i représente la vraisemblance de cette hypothèse. Par conséquent, les particules avec des poids élevés sont près des modes de la distribution a posteriori $p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0)$, tandis que ceux avec un faible poids sont près de la queue de celle-ci. Afin de générer les particules, le PF utilise une *densité de proposition* $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t})$ — qui peut être vue comme un simulateur — à partir de laquelle il est facile d'échantillonner. L'ensemble $\mathcal{S}_t$ des particules est calculé à partir de $\mathcal{S}_{t-1}$ et $\mathbf{z}_t$ suivant une procédure comportant trois étapes [Arulampalam et al., 2002]:

- **prédiction:** un échantillon $\mathbf{x}_{t|t-1}^i$ est généré à partir de chaque particule $\mathbf{x}_{t-1}^i$ de l'ensemble $\mathcal{S}_{t-1}$ en utilisant la densité de proposition $q(\mathbf{x}_t|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$.
- **pondération:** a chaque échantillon $\mathbf{x}_{t|t-1}^i$, on assigne un poids w_t^i obtenu suivant la formule

$$w_t^i = w_{t-1}^i . \frac{p(\mathbf{z}_t|\mathbf{x}_{t|t-1}^i).p(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i, \mathbf{z}_t)}.$$

- **ré-échantillonnage:** il consiste à supprimer ou dupliquer les échantillons suivant leur poids. Ceci est généralement fait en générant un nouvel ensemble de particules $\{\mathbf{x}_t^j\}_{j=1}^N$ à partir d'une représentation approchée et discrète $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ donnée par

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \sum_{i=1}^{N} w_t^i \delta_{\mathbf{x}_{t|t-1}^i}(\mathbf{x}_t) \tag{A.10}$$

  de telle sorte que $p(\mathbf{x}_t^j = \mathbf{x}_{t|t-1}^i) = w_t^i$. A la fin, on assigne à chaque particule résultant $\mathbf{x}_t^j$ un poids $w_t^j = 1/N$.

En général, l'étape de ré-échantillonnage n'est effectuée que lorsque l'estimation de la *taille effective des particules* $\hat{N}_{eff_t} = \frac{1}{\sum_{i=1}^N (w_t^i)^2}$ est inférieure à un seuil prédéfini $N_{T_{eff}}$. Le choix de la densité de proposition $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_{1:t})$ joue un rôle important dans le bon fonctionnement du filtre particulaire. L'un des choix généralement considéré dans la littérature est $q(\mathbf{x}_t|\mathbf{x}_{t-1}, \mathbf{z}_t) = p(\mathbf{x}_t|\mathbf{x}_{t-1})$. La Figue A.2 montre une illustration graphique du filtre particulaire tandis que l'algorithme A.1 décrit le pseudo-code à un pas de temps d'un filtre particulaire générique.

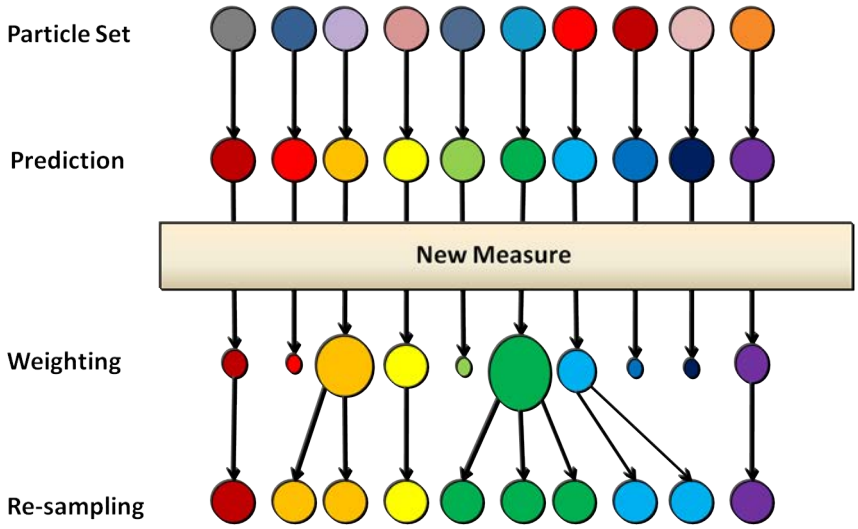


Figure A.2: Illustration graphique du filtre particulaire. L'état (ou la valeur) et le poids de chaque particule sont respectivement représentés par sa couleur et son rayon. De [Parker, 2013].

Algorithm A.1: Filtre Particulaire Générique

Algorithme PF$(\{\mathbf{x}_{t-1}^i, w_{t-1}^i\}_{i=1}^N,\ \mathbf{z}_t)$
  for $i = 1 : N$ **do**
    Échantillonne $\mathbf{x}_{t|t-1}^i \sim q(.|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$
    $w_t^i = w_{t-1}^i . \frac{p(\mathbf{z}_t|\mathbf{x}_{t|t-1}^i).p(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i,\mathbf{z}_t)}$
  for $i = 1 : N$ **do**
    $w_t^i = \frac{w_t^i}{\sum_{j=1}^N w_t^j}$
  $\hat{N}_{eff_t} = \frac{1}{\sum_{i=1}^N (w_t^i)^2}$
  if $\hat{N}_{eff_t} < N_{T_{eff}}$ **then**
    $\{\mathbf{x}_t^i\}_{i=1}^N =$ Reechantillonne $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)^*$
    for $i = 1 : N$ **do**
      $w_t^i = \frac{1}{N}$
  else
    for $i = 1 : N$ **do**
      $\mathbf{x}_t^i = \mathbf{x}_{t|t-1}^i$
  return $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$

(*) la **Procedure** Reechantillonne $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)$ effectue l'opération de ré-échantillonnage.

A.2.4 Le filtre particulaire régularisé

Lorsque la dynamique du système sous-jacent est soumise à peu de bruit, le filtre particulaire tel que décrit ci-dessus ne convient plus. Ceci est parce qu' à l'étape de ré-échantillonnage, la duplication d'une particule engendrera une particule fille dont l'évolution sera identique à celle de la particule mère, conduisant ainsi à long terme au phénomène d'*appauvrissement des échantillons* [Arulampalam et al., 2002]. Des filtres particulaires régularisés (RPFs) ont été introduits [LeGland et al., 1998, Musso et al., 2001] dans le but de prévenir ce phénomène.

L'idée principale des RPFs est de ré-échantillonner les particules (étape 3) à partir d'une approximation continue de la densité de probabilité $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ au lieu d'une approximation discrète (voir l'équation A.10) comme dans le cas du filtre particulaire générique. Par conséquent, les particules nouvellement générés sont différentes des particules mères correspondantes. L'approximation continue de $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ se fait au moyen d'une *fonction noyau* [Silverman and Green, 1986] $K(.)$ de la manière suivante (voir la figure A.3):

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \hat{p}_\lambda(\mathbf{x}_t|\mathbf{z}_{1:t}) = \sum_{i=1}^{N} w_t^i . K_\lambda(\mathbf{x}_t - \mathbf{x}_t^i), \tag{A.11}$$

où $K_\lambda(\mathbf{x}) = \frac{1}{\lambda^{n_x}} K(\frac{\mathbf{x}}{\lambda})$ est la fonction de noyau redimensionnée, λ est la largeur du noyau, et n_x est la dimension de l'espace d'état. Des exemples de fonctions noyaux comprennent le noyau Gaussien, le noyau Epanechnikov et le noyau cosinus. Généralement, la matrice de covariance des échantillons sous-jacents est utilisée comme une information supplémentaire lors de la définition de la fonction noyau à utiliser. Par exemple, cette matrice peut être utilisée comme matrice de covariance du noyau Gaussien. L'algorithme A.2 décrit le pseudo-code à un pas de temps d'un filtre particulaire régularisé.

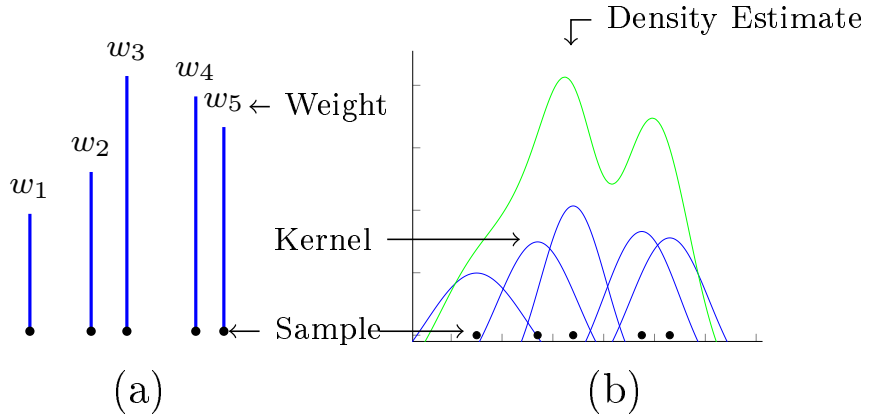


Figure A.3: Procédure de régularisation: (a) mesure empirique pondérée; (b) mesure régularisée. Les nouveaux échantillons seront tirés à partir de la distribution continue représentée en (b) au lieu de la distribution discrète obtenu en (a).

A.3 Le problème du suivi comportemental: cas mono-cible

Dans cette section, nous nous focalisons principalement sur des approches STAR qui permettent de traiter simultanément le problème du suivi de trajectoire et le celui de la reconnaissance

Algorithm A.2: Filtre particulaire régularisé

Algorithme RPF($\{\mathbf{x}_{t-1}^i, w_{t-1}^i\}_{i=1}^N$, $\mathbf{z}_t$)
  for $i = 1 : N$ **do**
    Échantillonne $\mathbf{x}_{t|t-1}^i \sim q(.|\mathbf{x}_{t-1}^i, \mathbf{z}_t)$
    $w_t^i = w_{t-1}^i . \frac{p(\mathbf{z}_t|\mathbf{x}_{t|t-1}^i).p(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i)}{q(\mathbf{x}_{t|t-1}^i|\mathbf{x}_{t-1}^i,\mathbf{z}_t)}$
  for $i = 1 : N$ **do**
    $w_k^i = \frac{w_t^i}{\sum_{j=1}^N w_t^j}$
  $\hat{N}_{eff_t} = \frac{1}{\sum_{i=1}^N (w_t^i)^2}$
  if $\hat{N}_{eff_t} < N_{T_{eff}}$ **then**
    Calcule la matrice de covariance empirique S_t de $\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N$
    Calcule D_t tel que $D_t \times D_t^T = S_t$
    $\{\mathbf{x}_t^i\}_{i=1}^N$ = Reechantillonne $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)$*
    for $i = 1 : N$ **do**
      Échantillonne $\epsilon^i \sim K(.)$ (la fonction noyau)
      $\mathbf{x}_t^i = \mathbf{x}_t^i + \lambda D_t \epsilon^i$
      $w_t^i = \frac{1}{N}$
  else
    for $i = 1 : N$ **do**
      $\mathbf{x}_t^i = \mathbf{x}_{t|t-1}^i$
  return $\{\mathbf{x}_t^i, w_t^i\}_{i=1}^N$
(*) La **Procedure** Reechantillonne $(\{\mathbf{x}_{t|t-1}^i, w_t^i\}_{i=1}^N)$ effectue l'opération de ré-échantillonnage.

d'activité. Nous présentons le principe fondamental du paradigme STAR, certains travaux de la littérature ainsi que leur limites. Finalement, nous décrivons notre solution permettant de faire face à ces limites.

A.3.1 Le paradigme STAR: principes

Généralement, les déplacements d'un piéton sont guidés par une motivation interne en rapport avec l'activité que celui-ci souhaite réaliser dans l'environnement. Par conséquent, il existe une dépendance intrinsèque, au sein de l'environnement, entre l'emplacement et l'activité d'un piéton. Cette dépendance fait partie de ce que nous appelons, de manière globale dans ce mémoire, des *connaissances contextuelles* ou tout simplement le *contexte environnemental*.

En guise d'illustration, un environnement (ex., une maison) est souvent structuré de telle sorte que seul un sous-ensemble d'activités peut être réalisé d'une zone donnée. Dans le cas d'une maison, des activités telles que *"ouvrir le réfrigérateur"* ou *"cuisiner"* ne peuvent se produire que dans la cuisine tandis que des activités telles que *"regarder la télévision* ou *"s'asseoir sur le sofa"* ont généralement lieu au salon. D'une manière générale, les connaissances contextuelles peuvent être définies par exemple autour de (1) une cible (ex., sa nature {*agent de contrôle*, *passager*}), (2) un groupe de cibles (ex., la nature de leur relation {*famille* ou *amis*}), (3) un emplacement (ex.,

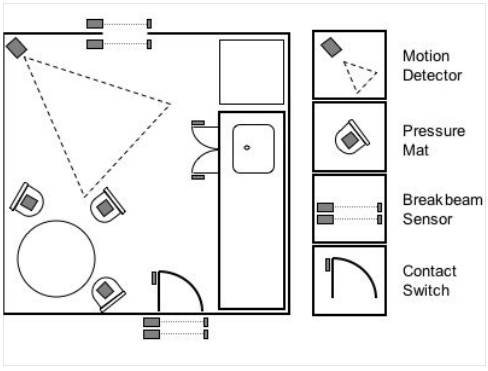
la liste des activités accessibles à partir d'un endroit donné), (4) une activité (ex., les endroits où l'activité peut se produire, la durée de l'activité), ou (5) une règle de causalité (ex., la liste des activités habituellement effectuée après l'exécution d'une activité donnée).

Le principe fondamental des approches STAR consiste à exploiter ces connaissances contextuelles lors du processus d'inférence afin d'améliorer la qualité des estimations. Plus précisément, les connaissances en rapport avec l'emplacement du piéton peuvent être utilisées pour raffiner l'estimation de l'activité courante du piéton. Inversement, les connaissances en rapport avec l'activité courante du piéton peuvent servir à améliorer l'estimation de sa trajectoire. On est donc en présence d'une structure en boucle telle qu'illustrée par la figure A.4. Toutefois, il se pose le problème de la représentation de ces connaissances contextuelles.

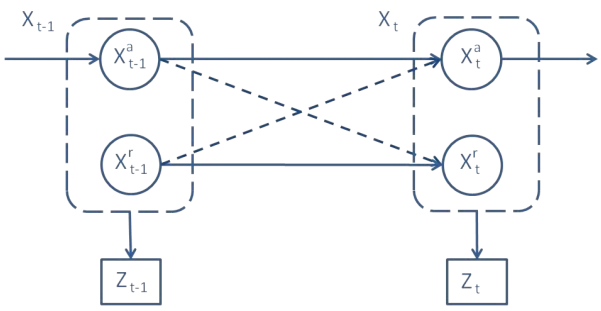
Figure A.4: les bénéfices du paradigme STAR.

A.3.2 État de l'art et limites actuelles

Le paradigme STAR a été formellement introduit par Wilson and Atkeson [2005] qui s'intéressent au monitoring automatique des personnes âgées dans une maison instrumentalisée (voir la figure A.5a) équipées des capteurs binaires de présence. Afin de représenter les connaissances contextuelles de l'environnement, ils utilisent les réseaux dynamiques Bayésiens (DBNs) [Pearl, 1988] (voir la figure A.5b) qui mettent en relation la salle dans laquelle se trouve la cible et les activité de ce dernier. Par la suite, ils apprennent les probabilités caractérisant ce DBN et s'appuient sur le filtre particulaire pour estimer le comportement de la cible considérée.



(a)



(b)

Figure A.5: Illustration d'une approche STAR. (a) la maison instrumentalisée. Les carrés gris représentent les capteurs. (b) la représentation de la connaissance contextuelle sous forme d'un DBN. $\mathbf{X}^r_.$ symbolise la salle dans laquelle se trouve la cible tandis que $\mathbf{X}^a_.$ modélise l'activité de la cible. De [Wilson and Atkeson, 2005].

Dans [Manfredotti et al., 2011], les auteurs essayent d'identifier le type de relation entre plusieurs piétons tout en suivant leurs trajectoires. À cette fin, ils introduisent la notion d'*état*

relationnel permettant de caractériser chaque cible. l'état relationnel d'une cible donnée est composé de deux parties: l'état de ses propres attributs et l'état de ses relations avec d'autres cibles. Par la suite, ils considèrent l'utilisation des DBNs sur ces états relationnels afin de représenter les connaissances contextuelles en rapport avec la dynamique des relations au fil du temps, et ils utilisent un filtre particulaire pour les besoins d'inférence. Ils démontrent que la reconnaissance explicite de la relation entre les cibles améliore l'estimation de leurs trajectoires. En outre, ils soutiennent que leur approche a des avantages significatifs par rapport aux méthodes qui n'intègre pas les connaissances liées aux relations dans le processus d'inférence.

Comme illustré dans les travaux sus-mentionnés, les approches STAR issues de la littérature s'appuient essentiellement sur des modèles graphiques (et notamment des DBNs) afin d'encoder les connaissances contextuelles de l'environnement. Bien que les DBNs soient adaptés pour représenter les séquences d'événements et les effets de causalité, leur capacités d'expressivité sont globalement limitées. En effet, en raison de l'hypothèse stricte de premier ordre caractérisant ces modèles, la structure d'un tel modèle graphique peut rapidement devenir très large et difficilement manœuvrable lorsque l'on souhaite représenter des données contextuelles plus fines telles que les durées des activités et/ou la procédure d'utilisation d'un objet présent dans l'environnement.

Outre ces limites intrinsèques aux modèles graphique utilisés, les approches mentionnées ci-dessus ont été conçues sous l'hypothèse selon laquelle le contexte environnemental n'est soumis à aucun changement dynamique pouvant affecter le comportements des cibles suivies. Cette hypothèse n'est généralement pas vérifiée en situations réelles (ex., un escalator qui arrête de fonctionner, une alerte incendie).

A.3.3 Les simulateurs avancés de comportements à base d'agents autonomes: une alternative aux DBNs

Afin de faire face aux limites évoquées ci-dessus, nous explorons dans ce mémoire une approche orthogonale. Au lieu de s'appuyer sur des DBN, nous envisageons l'utilisation d'un simulateur avancé de comportement à bases d'agents autonomes [Shao and Terzopoulos, 2007] pour la modélisation du contexte environnemental. Un tel simulateur permet de représenter, d'une manière plus fine et moins restrictif, les détails de l'environnement, les objets qui s'y trouvent ainsi que l'ensemble des services offerts par ces objets et qui sont susceptibles d'intéresser une cible (piéton) suivie.

Par exemple, il est possible de décrire facilement la procédure à suivre par un piéton lors de l'interaction avec un objet donné (ex., un guichet automatique), ainsi que la durée d'une telle interaction. De même, il est possible de caractériser un piéton (assimilé à un agent) non seulement par les propriétés basiques (ex., l'emplacement, la vitesse), mais également par des mécanismes motivationnels impliquant des propriétés internes plus abstraits (ex., le niveau de soif).

En outre, ces simulateurs sont capables de générer, pour tout agent simulé, des comportements cohérents et réalistes en relation avec son état interne et, plus important encore, le contexte environnemental. Pour atteindre un tel niveau de réalisme, l'architecture de ces simulateurs s'organise en général autour de trois dimensions [Shao and Terzopoulos, 2007]:

- **La représentation des services** (Section 5.2.1). La première dimension est la capacité d'incorporer, dans le monde virtuel, tous les services qui peuvent être d'un intérêt particulier pour un piéton donné. Cette dimension comprend la description des interactions avec

des objets présents dans l'environnement ainsi que des informations concernant la façon dont ces interactions sont effectuées (ex., comment utiliser un guichet automatique).

- **La sélection des actions de l'agent simulé** (Section 5.2.2). La seconde dimension concerne la capacité d'un agent virtuel d'exprimer, d'une manière cohérente et persistante, son désir pour un service donné dans l'environnement. Cette dimension implique la conception des mécanismes de sélection d'action capable de s'adapter rapidement à des modifications du contexte environnemental.

- **planification** (Section 5.2.3). Enfin, la troisième dimension concerne la capacité d'un agent virtuel à naviguer intelligemment vers un objet de l'environnement offrant le service auquel il est intéressé. Cette dimension implique la conception d'un planificateur chargé de déterminer les trajectoires à suivre sur la base du contexte environnemental et les caractéristiques de l'agent considéré (ex., si un agent souhaitant retirer de l'argent est pressé, il évitera les guichets automatiques où il y'a une file d'attente).

La figure A.6 illustre une vue d'ensemble du fonctionnement d'un tel simulateur. Essentiellement, étant donné un agent virtuel, des facteurs externes issus de l'environnement (données contextuelles) affectent son état interne. A chaque pas de temps, il exprime un désir (ex., besoin d'argent) à l'aide de son mécanisme motivationnel. Ce désir est par la suite mis en correspondance avec les différents services — et donc les objets offrant ce service — (ex., un guichet automatique) de l'environnement permettant de le réaliser. Enfin, une fois les objets correspondants ont été identifiés, les trajectoires vers ces objets sont effectuées par l'agent. Par conséquent, les agents se déplacent de manière **auto-explicative** car il est possible de construire, à partir d'une trajectoire observée, un ensemble de schémas pouvant expliquer les motivations liée à l'activité de l'agent. Le pipeline ainsi décrit matérialise explicitement la synergie existant entre l'activité et l'emplacement et, pour cette raison, ces simulateurs se révèlent être utile en pratique dans le cadre du paradigme STAR. Nous allons maintenant décrire notre solution au problème de suivi comportemental dans laquelle nous tirons avantage de ces simulateurs comportementaux.

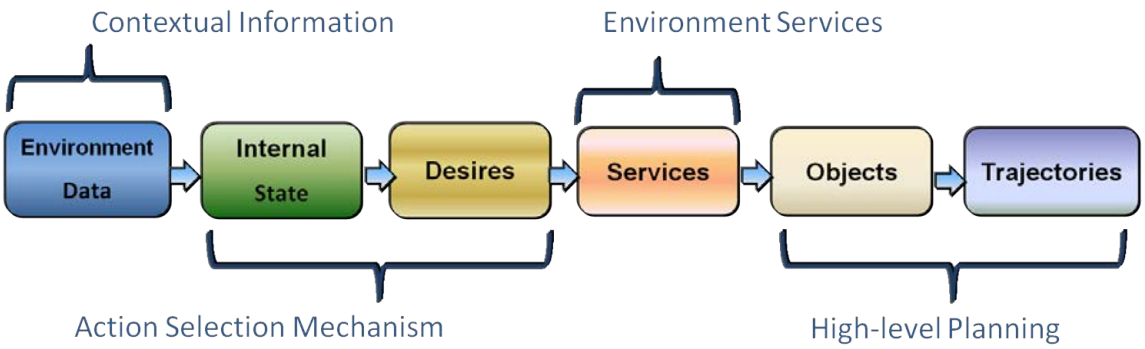


Figure A.6: Vue d'ensemble du fonctionnement d'un simulateur comportemental d'agents autonomes.

A.3.4 Suivi comportemental à base d'agents

La solution au problème de suivi comportemental que nous proposons est illustrée par la figure A.7. Dans cette approche, le simulateur vise à reproduire artificiellement des comportements humains au travers des agents autonomes situés dans un environnement virtuel représentant l'environnement réel. Un filtre particulaire est utilisé à des fins d'inférence. Dans ce filtre, un ensemble d'hypothèses en rapports avec les états internes des agents virtuels est considéré. Ensuite, sur la base de ces hypothèses, le simulateur est utilisé, lors de l'étape de prédiction du filtre, pour simuler le comportement des différents agents et, par conséquent, estimer aussi bien l'emplacement et l'activité de la cible suivie.

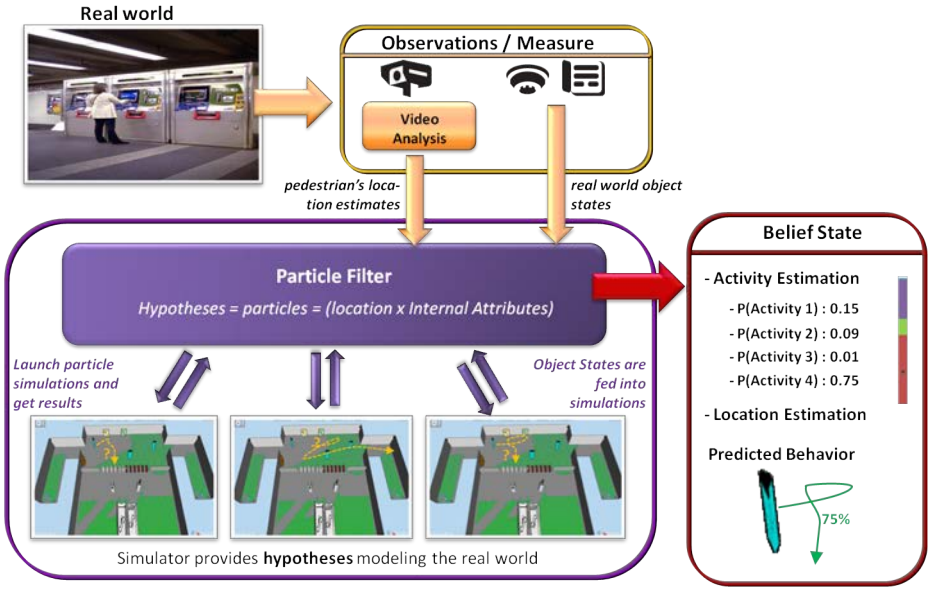


Figure A.7: Vue d'ensemble de la solution proposée: le simulateur de comportements est utilisé comme un bloc prédictif au sein de la procédure de filtrage (filtre particulaire). Les observations provenant des cameras sont utilisées pour raffiner la croyance concernant le comportement de la cible suivie. En outre, les états des objets de l'environnement réel sont simulés dans l'environnement virtuel afin de préserver une cohérence entre les deux mondes.

Les observations fournies par les caméras (déployés dans l'environnement réel) sont ensuite utilisés lors de l'étape de correction du processus de filtrage pour raffiner la croyance calculée à l'étape de prédiction. Ces observations sont notamment des estimations bruitées de l'emplacement de la cible suivie lorsque celle-ci est observée. Ainsi, les hypothèses peu probables sont éliminées et les poids des particules sont mis à jour. Le réseau des caméras peut couvrir totalement ou partiellement l'environnement. Cependant, nous connaissons à l'avance les zones de l'environnement sous couverture sensorielle. En outre, nous supposons qu'il est possible de récupérer, à chaque pas de temps, les états des différents objets de l'environnement réel. Finalement, afin que les agents virtuels puissent prendre en compte des changements dynamiques qui se produisent dans l'environnement réel et s'adapter en conséquence, le simulateur est mis à jour avec les changements concernant les états des objets réels (ex., les pannes d'escalators) ainsi que des événements exogènes (ex., alerte incendie) de manière à préserver la cohérence entre le monde réel et le monde simulé.

A.3.5 Implémentation et Expérimentations

L'approche proposée a été implémentée en utilisant SE-Star, un simulateur comportemental propriétaire de la société Thales[21]. Cependant, SE-Star est un simulateur avec peu d'aléatoire dans la dynamique du modèle de comportement. Par conséquent, son utilisation dans un filtre particulaire peut conduire au au phénomène d'*appauvrissement des échantillons* [Arulampalam et al., 2002]. Pour cette raison, nous avons mis en œuvre un filtre particulaire régularisé (voir Section A.2.4) afin d'introduire explicitement de l'aléatoire (par l'intermédiaire de bruit) dans le processus de filtrage.

Des expérimentations ont été conduites (dans le cas spécial du suivi mono-cible) aussi bien dans un environnement simulé — une station de métro — (voir Section 5.5.2) que dans un environnement réel — (bâtiment de bureaux) — (voir Section 5.5.3). Les résultats obtenus sont très satisfaisants, même en présence de longues périodes d'occlusions et/ou d'événement exogènes. Ces résultats démontrent ainsi l'efficacité et la robustesse de l'approche proposée dans le cadre STAR.

A.3.6 Travaux futurs

Les expérimentations menées dans cette section ont été conduites dans des environnements intérieurs où le choix des activités pour le nombre d'activités susceptible d'intéresser la cible suivi est plutôt limité (une dizaine). Une éventuelle extension de ce travail consisterait à étudier la performance du système proposé dans un monde ouvert où l'espace des possibilités est beaucoup plus large comme par exemple dans le cadre de simulations de trafic urbain.

Dans la section suivante, nous nous intéressons au cas général du suivi multi-cibles et présentons des solutions permettant de gérer efficacement les interactions pouvant exister entre plusieurs cibles

A.4 Le problème du suivi comportemental: cas multi-cibles

Comme dans le cas mono-cible, le suivi multi-cibles est un problème de filtrage et par conséquent, il peut être modélisé à l'aide des système dynamique partiellement observés. Dans ce cas l'état globale du système $\mathbf{x}_t$ est composé de l'ensemble des états de tous les cibles suivies. De même, l'observation $\mathbf{z}_t$ obtenue des capteurs est composée d'un ensemble d'observations atomiques correspondant chacune à une cible observée. Les approches de filtrage classiques peuvent aisément être appliquées au problème du suivi multi-cibles. Toutefois, parce qu'elles raisonnent dans un espace d'état de très grande dimension (en rapport avec le nombre de cibles), ces approches sont généralement inapplicable lorsque le nombre de cibles est supérieure à 3.

Lorsque les cibles sont considérées indépendantes entre elles, une alternative consiste à raisonner sur chaque cible individuellement au lieu de considérer l'état global du système $\mathbf{x}_t$. En conséquence, comme illustrée par la figure A.8, le processus de filtrage peut ainsi être décomposé en plusieurs (sous-)processus de filtrage, chacun assigné à une cible unique.

Toutefois, de manière générale, le suivi multi-cibles présente deux difficultés supplémentaires en comparaison au suivi mono-cible:

[21] Thales est une société française multinationale spécialisée dans l'aérospatial, la défense et les technologies de l'information.

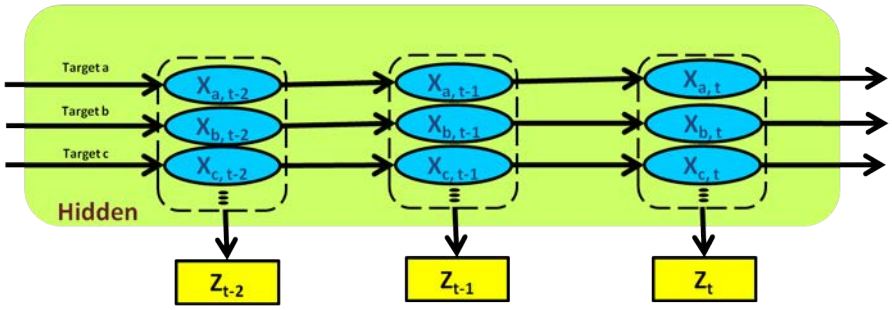


Figure A.8: Suivi multi-cibles comme une collection de suivis mono-cible.

- **l'association de données observationnelles** (**DA**): l'association de données observationnelles consiste à déterminer les identités des cibles associées à chaque observation reçue des capteurs;

- **les interactions (dépendances comportementales) entre cibles**: la dynamique d'une cible donné peut être influencée par la présence d'autres cibles dans son voisinage. Il est donc primordiale de prendre en compte ces dépendances lors du processus de filtrage.

Le problème d'association de données observationnelles a été étudié pendant des décennies et des solutions classiques ont été proposées telles que le *traqueur à multiples hypothèses* (MHT) [Reid, 1979] et/ou l'*association de données probabiliste jointe* (JPDA) [Fortmann et al., 1983]. D'autre part, peu sont des travaux [Khan et al., 2003, Yu and Wu, 2004, Cattelani et al., 2014] qui ne présupposent pas d'une indépendance totale des cibles et traitent explicitement les interactions entre elles. Cependant, soit ces travaux ([Khan et al., 2003, Yu and Wu, 2004]) s'appuient sur des hypothèses spécifiques au problème adressé — ce qui leur permet d'avoir une solution d'une complexité faible, soit ces travaux ([Cattelani et al., 2014]) proposent des méthodologies génériques intégrant des modèles d'interactions pouvant s'appliquer à d'autres problèmes. Toutefois, dans ce dernier cas, les méthodologies proposées souffrent malheureusement d'une forte complexité computationnelle due au fait qu'elles reposent sur l'état joint des différentes cibles suivies.

Dans ce mémoire, nous nous intéressons principalement au cas spécial du suivi multi-cibles dans lequel le nombre de cibles, connu à l'avance, ne change pas au cours du temps. Nous focalisons notre attention sur la gestion des interaction entre cibles et nous nous appuyons sur l'approche JPDA pour résoudre les problèmes d'association de données observationnelles. Plus précisément, nous sommes motivés par l'idée de développer une approche générique (c-à-d, à base de modèles d'interactions) permettant de gérer, à faible coûts computationnels, les interactions entre cibles dans un processus d'inférence comportementale. Dans les sections suivantes, nous allons formaliser le problème du suivi-multi (Section A.4.1), introduire l'approche JPDA pour la résolution du problème d'association de données observationnelles (Section A.4.2) et présenter notre solution pour la gestion des interactions entre cibles (Section A.4.3).

A.4.1 Formalisation

Considérons un environnement fermé, doté d'un réseau de capteurs, dans laquelle plusieurs cibles pouvant interagir entre elles évoluent. Le nombre K de cibles dans l'environnement ne change pas au cours du temps. Aussi, l'environnement peut contenir des zones non couvertes par les

capteurs. Une cible à une probabilité P_D d'être détectée lorsqu'elle est dans une zone couverte par les capteurs. En outre, le réseau de capteurs peut retourner de *fausses alarmes* correspondant à des observations erronées n'émanant d'aucune cibles. On suppose que le nombre de fausses alarmes générées suit une distribution de Poisson de paramètre $\lambda_{FT}V$ où λ_{FT} est le taux de fausses alarme par unité de temps et unité de volume, et V est le volume de l'environnement.

Soit $\mathbf{x}_{k,t}$ l'état de la k^e cible au pas de temps t. L'état $\mathbf{x}_t$ de l'ensemble du système, défini comme $\mathbf{x}_t = \{\mathbf{x}_{k,t}\}_{k=1}^K$ est l'ensemble des états individuels des différents cibles. De même, soit $\mathbf{z}_t = \{\mathbf{z}_{1,t}, \mathbf{z}_{2,t}, \cdots, \mathbf{z}_{M_t,t}\}$ l'observation fournie par le réseau de capteurs au pas de temps t. Nous représentons par $\mathbf{z}_{1:t}$ la séquence de toutes les observations reçues à ce jour. Le problème qui nous intéresse consiste à estimer efficacement, à chaque pas de temps t, la distribution de probabilité a-posteriori $p(\mathbf{x}_t|\mathbf{z}_{1:t}, Bel_0)$ concernant les états des différentes cibles dans l'environnement sur la base de la séquence d'observations $\mathbf{z}_{1:t}$ des capteurs et la connaissance a-priori Bel_0. Pour plus de simplicité dans l'écriture, le terme Bel_0 sera omis par la suite dans les différentes notations.

Afin d'éviter les problèmes de complexité liés à l'espace d'état à très grande dimension, nous envisageons une approche factorisée dans laquelle nous essayons d'approximer la distribution a-posteriori $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ en utilisant K filtres "collaboratifs", chacun étant associé à une cible donnée. En d'autres termes, nous voulons représenter approximativement $p(\mathbf{x}_t|\mathbf{z}_{1:t})$ par

$$p(\mathbf{x}_t|\mathbf{z}_{1:t}) \approx \bar{p}(\mathbf{x}_k|\mathbf{z}_{1:t}) = \prod_{k=1}^{K} p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t}), \tag{A.12}$$

où chaque $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$ est la distribution a-posteriori relative à la k^e cible et obtenue via le k^e filtre. Cependant, puisque les cibles peuvent potentiellement s'influencer, la difficulté à laquelle nous faisons face est celle de prendre en compte ces interactions dans le calcul factorisé de la distribution $p(\mathbf{x}_t|\mathbf{z}_{1:t})$.

Comme mentionné à la Section A.2, la distribution a-posteriori $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$ relative à la k^e cible peut être obtenue — via le filtre associé — en deux étapes:

- **une étape de prédiction** dans laquelle la distribution prédite $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ de la k^e cible est calculée sur la base de la distribution a-posteriori $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ au pas de temps précédent;

- **une étape de correction** dans laquelle la distribution prédite $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ est corrigée sur la base de l'observation reçue $\mathbf{z}_t$ conduisant ainsi à la distribution a-posteriori $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t})$.

Dans le problème du suivi multi-cibles, l'étape de correction traite du problème d'association de données observationnelles tandis que nous adressons la gestion des interaction entre cibles à l'étape de prédiction. Dans ce qui suit, nous allons présenter le paradigme JPDA permettant de gérer l'association des données. Notre solution permettant de prendre en compte les interactions entre cibles de manière efficace sera décrite en Section A.4.3.

A.4.2 Le paradigme JPDA

Cette section fait l'hypothèse de la disponibilité de la distribution prédite $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ relative à la k^e cible ($\forall k = 1, \cdots, K$). A la réception de $\mathbf{z}_t = \{\mathbf{z}_{1,t}, \mathbf{z}_{2,t}, \cdots, \mathbf{z}_{M_t,t}\}$, le paradigme JPDA

résout le problème d'association de données suivant la procédure décrite ci-dessous.

Soient $\{\beta_{jk}\}_{j=0}^{M_t}$ les probabilités que la k^e cible soit associée à la j^e observation atomique $\mathbf{z}_{j,t}$. $\mathbf{z}_{0,t}$ est utilisée pour signifier que la k^e cible n'ait pas été détectée. Dans le paradigme JPDA, la vraisemblance de $\mathbf{z}_t$ par rapport à la k^e cible est exprimée comme

$$p(\mathbf{z}_t|\mathbf{x}_{k,t}) = \sum_{j=0}^{M_t} \beta_{jk}.p(\mathbf{z}_{j,t}|\mathbf{x}_{k,t}), \tag{A.13}$$

où $p(\mathbf{z}_{j,t}|\mathbf{x}_{k,t})$ correspond au modèle d'observation du réseau des capteurs. Sur cette base, la distribution a-posteriori s'obtient de la manière suivante (voir Équation A.5):

$$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t}) \propto p(\mathbf{z}_t|\mathbf{x}_{k,t}).p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}). \tag{A.14}$$

Tout ce qui reste pour l'application des équations A.13 et A.14 est le calcul des probabilités β_{jk}.

Soit une hypothèse θ constituée un ensemble de paires $(j,k) \in \{0,\cdots,M_t\} \times \{1,\cdots,K\}$. $(j,k) \in \theta$ signifie que, au sein de l'hypothèse θ, la k^e cible ait générée la j^e observation atomique. θ est une *hypothèse jointe valide* si elle détermine, sans équivoque, l'origine de chaque observation atomique de telle sorte que (1) deux cibles ne peuvent avoir engendré une même observation atomique et (2) deux observations atomiques ne peuvent avoir été engendrées par une même cible. Ainsi, à partir d'une hypothèse jointe valide θ, il est possible de déterminer le nombre N_{DT}^{θ} de cibles détectées de même que le nombre de fausses alarmes $N_{FT}^{\theta} = M_t - N_{DT}^{\theta}$. Soit Θ l'ensemble des hypothèses jointes valides. Soit $\Theta_{jk} \subset \Theta$ l'ensemble d'hypothèses jointes valides dans lesquelles la k^e cible est associée à la j^e observation atomique ($\Theta_{jk} = \{\theta : (j,k) \in \theta\}$). Nous avons

$$\beta_{jk} = p(\Theta_{jk}|\mathbf{z}_{1:t}) = \sum_{\theta:(j,k)\in\theta} p(\theta|\mathbf{z}_{1:t}), \tag{A.15}$$

où $p(\theta|\mathbf{z}_{1:t})$ est la probabilité de l'hypothèse θ et elle se calcule selon [Vermaak et al., 2005]

$$p(\theta|\mathbf{z}_{1:t}) \propto \lambda_{FT}^{N_{FT}^{\theta}} \times (1-P_D)^{K-N_{DT}^{\theta}} \times P_D^{N_{DT}^{\theta}} \times \prod_{j:l_j\neq 0} p^{l_j}(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1}), \tag{A.16}$$

où $j : l_j \neq 0$ dénotes des paires (j,l_j) dans θ dans lesquelles l'observation atomique associée n'est pas une fausse alarme, et $p^k(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1})$, donnée par

$$p^k(\mathbf{z}_{j,t}|\mathbf{z}_{1:t-1}) = \int p(\mathbf{z}_{j,t}|\mathbf{x}_{k,t}) p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{k,t}, \tag{A.17}$$

est la vraisemblance prédictive de la j^e observation atomique en se basant sur les informations de la k^e cible.

Comme mentionné ci-dessus, le paradigme JPDA nécessite l'énumération de toutes les hypothèses jointes valides, ce qui peut être fastidieux. Afin de réduire la complexité d'une telle énumération, Bar-Shalom and Fortmann [1987] a introduit une technique, appelée *gating*, qui suggère qu'une cible ne peut être associée à une observation atomique donnée que si cette dernière appartient à une région de validation définie autour de la dite cible. La figure A.9 illustre cette technique.

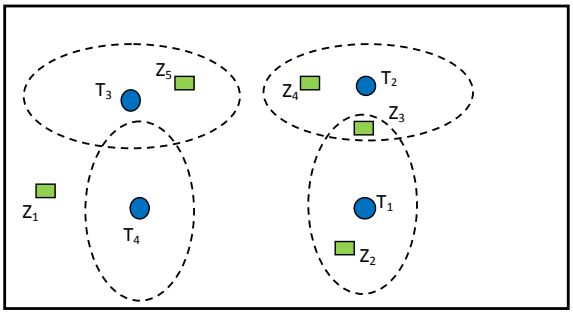


Figure A.9: Exemple illustratif de la technique de "gating": les cercles bleues représentent les cibles et les carrés verts désignent les observations atomiques. La région de validation de chaque cible est représentée par l'ellipse centrée autour d'elle et seules les observations atomiques se situant à l'intérieur de cette région sont considérées comme candidates potentielles pour la dite cible.

A.4.3 La gestion des interactions entre cibles

Cette section fait l'hypothèse de la disponibilité de la distribution a-posteriori $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ relative à la k^e cible $(\forall k = 1, \cdots, K)$ au pas de temps $t-1$.

Afin de gérer les interactions entre cibles, nous proposons de les représenter à l'aide d'une dépendance conditionnelle Markovienne. Plus précisément, nous faisons l'hypothèse selon laquelle l'évolution de l'état d'une cible donnée au pas de temps courant dépend uniquement des états des différentes cibles au pas de temps précédent. Par conséquent, la dynamique de l'ensemble des cibles peut s'exprimer sous la forme

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = \prod_{k=1}^{K} p(\mathbf{x}_{k,t}|\mathbf{x}_{t-1}). \tag{A.18}$$

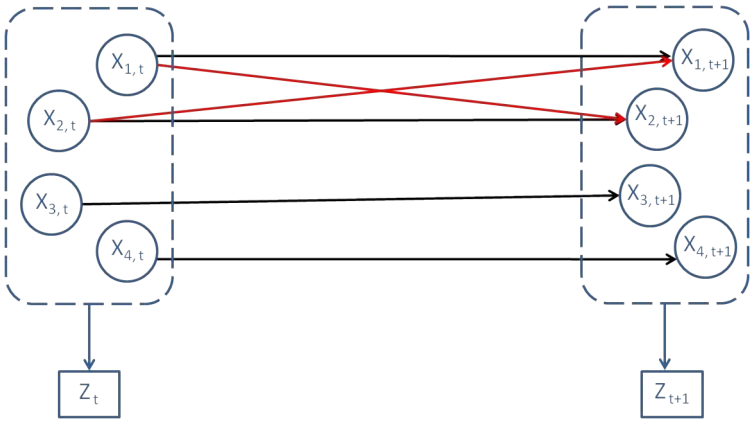


Figure A.10: Modèle graphique représentant les interactions entre cibles. Les cercles symbolisent les états des différentes cibles tandis que les carrés représentent les observations. Les flèches correspondent aux dépendances temporelles entre variables d'états. Les flèches rouges modélisent les interactions entre cibles. Dans cette figure, on suppose que seules les cibles 1 et 2 s'influencent mutuellement tandis que les autres cibles agissent de manière indépendante.

Cependant, comme illustré par la figure A.10, il est courant que, étant donnée une cible k,

seule un sous-ensemble de cibles (celles appartenant à son voisinage) l'influence réellement. Dans ce mémoire, nous cherchons à approximer la distribution prédite $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ correspondant à chaque cible k. Nous proposons une approche qui s'appuie sur la *localité des interactions* pour dériver cette approximation. Plus précisément, pour une cible k, nous avons l'intention (1) de déterminer le sous-ensemble $\mathcal{N}(k)$ des cibles influençant sur sa dynamique et (2) de s'appuyer par la suite sur ce sous-ensemble pour calculer la distribution prédite correspondante. Ceci est fait suivant la procédure suivante:

- **agrégation des distribution de probabilité** (Section 7.4.3): tout d'abord, nous proposons de partitionner la distribution a-posteriori $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ de chaque cible k en un ensemble intuitif de R_k régions d'intérêt et d'agréger chaque région ainsi obtenue en un *état représentatif* aussi appelé *représentant* de la région; Nous avons ainsi $\{\hat{\mathbf{x}}^r_{k,t-1}, \hat{w}^r_{k,t-1}\}_{r=1}^{R_k}$ où $\hat{\mathbf{x}}^r_{k,t-1}$ et $\hat{w}^r_{k,t-1}$ sont respectivement le représentant et le poids de la r^e région;

- **construction des graphes d'interactions** (Section 7.4.2): Ensuite, sur la bases des représentants ainsi obtenus, nous construisons, en conjonction avec une fonction Φ de voisinage modélisant la connaissance a-priori du domaine d'application, un ensemble $\{(\bar{G}_c, \bar{w}_c)\}$ de graphes pondérés d'interactions où chaque graphe $\bar{G}_c$ implique une combinaison distincte de K représentants des différentes cibles et $\bar{w}_c$ est le produit des poids des différents représentants impliqués;

- **regroupement des graphes d'interactions** (Section 7.4.3.3): La prochaine étape consiste à calculer, pour chaque représentant $\hat{\mathbf{x}}^r_{k,t-1}$ d'une cible k donnée, l'ensemble $\Delta_{k,r} = \{(\Delta^g_{k,r}, \upsilon^g_{k,r})\}_{g=1}^{\Xi_{k,r}}$ des classes pondérées des graphes d'interactions où $\Delta^g_{k,r}$ est la classe regroupant des graphes d'interactions impliquant $\hat{\mathbf{x}}^r_{k,t-1}$ tel que le voisinage $\tilde{\mathcal{N}}_g(k,r)$ de $\hat{\mathbf{x}}^r_{k,t-1}$ est identique, $\upsilon^g_{k,r}$ est le poids de la classe $\Delta^g_{k,r}$ et il est égal à la somme des poids des graphes d'interactions composant la dite classe, et $\Xi_{k,r}$ est la cardinalité de $\Delta_{k,r}$;

- **Estimation des distributions prédites** (Section 7.4.3.3): Finalement, la distribution prédite $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ de chaque cible k est calculée comme

$$p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1}) = \int_{\mathbf{x}_{k,t-1}} Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{k,t-1}, \tag{A.19}$$

  où $Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1})$ est ce que nous appelons la *vraisemblance interactive de transition* relative à $\mathbf{x}_{k,t-1}$ et il est défini par

$$\begin{aligned} Q(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{z}_{1:t-1}) &= \int_{\mathbf{x}_{\mathcal{N}(k),t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \mathbf{x}_{\mathcal{N}(k),t-1}) p(\mathbf{x}_{\mathcal{N}(k),t-1}|\mathbf{z}_{1:t-1}) d\mathbf{x}_{\mathcal{N}(k),t-1}, \\ &\approx \sum_{g=1}^{\Xi_{k,\Upsilon_k(\mathbf{x}_{k,t-1})}} \frac{\upsilon^g_{k,\Upsilon_k(\mathbf{x}_{k,t-1})}}{\hat{w}^{\Upsilon_k(\mathbf{x}_{k,t-1})}_{k,t-1}} p(\mathbf{x}_{k,t}|\mathbf{x}_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,\Upsilon_k(\mathbf{x}_{k,t-1})),t-1}). \end{aligned} \tag{A.20}$$

En principe, lorsque nous sommes dans un cadre de filtrage particulaire ($p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ est représentée par un ensemble $\{\mathbf{x}^i_{k,t-1}, w^i_{k,t-1}\}$ de particules pondérées), pour une particule $\mathbf{x}^i_{k,t-1}$ de $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$, $Q(\cdot|\mathbf{x}^i_{k,t-1}, \mathbf{z}_{1:t-1})$ (voir Equation A.20) est approximée en utilisant plusieurs

particules de la manière suivante. Tout d'abord, on détermine $r_i = \Upsilon_k(\mathbf{x}^i_{k,t-1})$, la région de $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$ à laquelle la particule $\mathbf{x}^i_{k,t-1}$ appartient ainsi que le représentant correspondant $\hat{\mathbf{x}}^{r_i}_{k,t-1}$. Ensuite, on considère l'ensemble $\Delta_{k,r_i} = \{(\Delta^g_{k,r_i}, \upsilon^g_{k,r_i})\}_{g=1}^{\Xi_{k,r_i}}$ des classes de graphes d'interactions et, de ce fait, l'ensemble $\{\tilde{\mathcal{N}}_g(k,r)\}_{g=1}^{\Xi_{k,r}}$ du voisinage de $\hat{\mathbf{x}}^{r_i}_{k,t-1}$ associé à chaque classe. La particule $\mathbf{x}^i_{k,t-1}$ est simulée en utilisant chacun de ces voisinages, conduisant ainsi à Ξ_{k,r_i} nouvelles particules pondérées $\{\mathbf{x}^{i,g}_{k,t|t-1}, w^{i,g}_{k,t|t-1}\}_{g=1}^{\Xi_{k,r}}$ comptant pour la distribution prédite $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ de la cible k où

$$\begin{aligned}
&\mathbf{x}^{i,g}_{k,t|t-1} \sim q(.|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1}, \mathbf{z}_t), && \forall g = 1, \cdots, \Xi_{k,r_i};\\
&w^{i,g}_{k,t|t-1} \propto w^i_{l,t-1} \times \frac{\upsilon^g_{k,r_i}}{\hat{w}^{r_i}_{k,t-1}} \times \frac{p(\mathbf{x}^{i,g}_{k,t|t-1}|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1})}{q(\mathbf{x}^{i,g}_{k,t|t-1}|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1}, \mathbf{z}_t)}, && \forall g = 1, \cdots, \Xi_{k,r_i}, \quad \text{(A.21)}
\end{aligned}$$

avec $q(.|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1}, \mathbf{z}_t)$ et $p(.|\mathbf{x}^i_{k,t-1}, \hat{\mathbf{x}}_{\tilde{\mathcal{N}}_g(k,r_i),t-1})$ qui sont respectivement la densité de proposition et le modèle de la dynamique de la cible k.

Comme on peut le remarquer de la description ci-dessus, le nombre de particules représentant $p(\mathbf{x}_{k,t}|\mathbf{z}_{1:t-1})$ est supérieure au nombre de particules représentant $p(\mathbf{x}_{k,t-1}|\mathbf{z}_{1:t-1})$. Afin d'éviter un nombre croissant de particules dans le système de filtrage, nous proposons des heuristiques permettant de maintenir constant ce nombre de particules au fil du temps tout en s'assurant d'un maintien de diversité à l'intérieur de l'ensemble des particules (voir Section 7.5.3.2).

A.4.4 Autres contributions

La mise en œuvre de notre solution dans le cadre du paradigme JPDA nous a amené à proposer deux contributions supplémentaires à savoir:

- **la gestion des zones non-couvertes dans le paradigme JPDA** (voir Section 7.5.2): le paradigme JPDA a été originellement conçu sous l'hypothèse selon laquelle l'environnement considéré soit totalement sous couverture sensorielle. Dans ce mémoire, nous révisons les hypothèses du modèle d'observation et proposons une extension du paradigme JPDA afin de faire face aux environnements partiellement couvert par le réseau des capteurs;

- **la technique de "gating" à base de multiples représentants** (voir Section 7.5.4): l'approche classique du gating vise à approcher la distribution prédite d'une cible donnée en une distribution Gaussienne à partir de laquelle des calculs de distance peuvent être effectués pour déterminer quelles observations atomiques appartiennent à la région de validation de la dite cible. Cette approximation n'est pas toujours convenant, en particulier en cas de forte non-linéarité dans la dynamique de la cible. Pour faire face à cette limitation, nous proposons une nouvelle approche de gating, basée sur des représentants des cibles, qui permettent d'utiliser plusieurs Gaussiennes pour des associations plus fines de données observationnelles.

A.4.5 Expérimentations

Nous avons évalué l'approche proposée sur des scénarios virtuels de suivi extrêmement difficiles en utilisant comme simulateur de référence celui du comportement à base de steering fourni par [Christian and Thomas, 2007] (voir Section 7.6). Les résultats obtenus sont très intéressant et ils démontrent l'efficacité de l'approche à gérer des interactions entre cibles. En outre, en présence de zones non couvertes où les cibles sont assujettis à une multitude de comportements, les expériences ont mis en évidence les avantages de l'utilisation de la technique de gating à base de multiples représentants puisque cette technique atténue la présence de bruit au sein du processus de filtrage. En outre, notre approche s'est avérée être une prémisse précieuse face au problème de ré-identification (c-à-d, la ré-identification d'une cible donnée lorsqu'elle entre de nouveau dans une zone sous couverture sensorielle) essentiellement rencontrée. Ceci est rendu possible par la capacité de l'algorithme conçu à maintenir, au moyen des multiples représentants introduits, une diversité comportementale au sein de l'ensemble des particules caractérisant la distribution de chaque cible lorsque ce dernier est dans une zone non couverte.

A.4.6 Travaux futurs

Malgré les améliorations observées sur les expériences que nous avons menées, l'algorithme proposé a actuellement plusieurs limites. La première correspond à la façon dont les graphes d'interactions sont générés. En effet, nous nous appuyons sur une fonction simple basée sur une heuristique basique (distance entre deux cibles). Des efforts supplémentaires doivent être faits sur la façon de construire ces graphes d'interactions lors de l'utilisation d'un simulateur comportemental avancé (ex., SE-Star) conçu pour représenter la dynamique d'un phénomène du monde réel. Des travaux ont été entrepris pour représenter des groupes d'agents en interaction dans des systèmes multi-agents [Davidsson, 2002, Kubera et al., 2008], et une direction intéressante pour les travaux futurs serait d'étudier la possibilité de profiter de ces formalismes d'interactions afin de construire, à la volée, des graphes d'interactions compatibles avec les simulateurs considérés.

Une autre limitation du travail présenté concerne la "mémoire des interactions" d'un pas de temps au suivant. En effet, le graphe d'interaction est construit sur la base des états des différentes cibles au pas de temps précédent. Cette approximation ne suffit pas pour gérer efficacement les interactions qui sont persistantes dans le temps. Une première solution à cette limitation serait de définir le modèle d'interaction de telle sorte qu'elle ne dépend pas seulement des états des cibles au pas de temps précédent, mais également des états des cibles à des pas de temps antérieurs. Toutefois, une attention particulière doit être mise sur la fenêtre temporelle à considérer de manière à éviter les problèmes de stockage.

A.5 Conclusion

Dans ce mémoire, nous nous sommes intéressés au problème de suivi comportemental qui consiste à déduire les comportements de différentes cibles (piétons) dans un environnement en se basant basant uniquement sur des observations reçues d'un réseau de capteurs. Nous avons considéré le cas général selon lequel le réseau de capteur peut partiellement couvrir l'environnement. Ce problème est généralement difficile en raison des deux facteurs suivants:

- **comportement individuel et dépendant du contexte**: le comportement présenté par une cible donnée dépend essentiellement de ses caractéristiques personnelles (ex., la vitesse, l'âge, l'endurance) ainsi que l'environnement dans lequel il est immergé. En effet, les objets présents dans l'environnement influencent grandement un tel comportement, en particulier, les activités auxquelles la cible sous-jacente peut s'intéresser. Sur cette base, il est crucial d'avoir une bonne représentation du contexte environnemental dans lequel le processus de suivi est effectué.

- **les interactions entre cibles**: dans un cadre multi-cibles, les cibles situées dans un voisinage proche s'influencent mutuellement. Ces dépendances ajoutent une autre couche de complexité au problème de suivi mentionné ci-dessus puisque, pour des résultats de bonne qualité, il est recommandé de ne pas raisonner sur chaque cible individuellement.

Le travail effectué dans de cette thèse s'articule autour de ces deux facteurs et les Sections A.3 et A.4 présentent nos contributions relatives à chacune de ces facteurs respectivement.

Abstract

In this thesis, we are interested in the problem of pedestrian behavioral tracking within a critical environment partially under sensory coverage. While most of the works found in the literature usually focus only on either the location of a pedestrian or the activity a pedestrian is undertaking, we stands in a general view and consider estimating both data simultaneously. The contributions presented in this document are organized in two parts. The first part focuses on the representation and the exploitation of the environmental context for serving the purpose of behavioral estimation. The state of the art shows few studies addressing this issue where graphical models with limited expressiveness capacity such as dynamic Bayesian networks are used for modeling prior environmental knowledge. We propose, instead, to rely on richer contextual models issued from autonomous agent-based behavioral simulators and we demonstrate the effectiveness of our approach through extensive experimental evaluations. The second part of the thesis addresses the general problem of pedestrians' mutual influences, commonly known as targets' interactions, on their respective behaviors during the tracking process. Under the assumption of the availability of a generic simulator (or a function) modeling the tracked targets' behaviors, we develop a yet scalable approach in which interactions are considered at low computational cost. The originality of the proposed approach resides on the introduction of density-based aggregated information, called "representatives", computed in such a way to guarantee the behavioral diversity for each target, and on which the filtering system relies for computing, in a finer way, behavioral estimations even in case of occlusions. We present the modeling choices, the resulting algorithms as well as a set of challenging scenarios on which the proposed approach is evaluated.

Keywords: Behavioral tracking, Simultaneous tracking and activity recognition, Autonomous agents, Agent-based simulations, Particle filter, Interactions, JPDAF, Interacting targets, Representative states, Density-based aggregation, Clustering, Behavioral diversity, Occlusions.

Résumé

Dans cette thèse, nous nous intéressons au problème du suivi comportemental des piétons au sein d'un environnement critique partiellement observé. Tandis que plusieurs travaux de la littérature s'intéressent uniquement soit à la position d'un piéton dans l'environnement, soit à l'activité à laquelle il s'adonne, nous optons pour une vue générale et nous estimons simultanément à ces deux données. Les contributions présentées dans ce document sont organisées en deux parties. La première partie traite principalement du problème de la représentation et de l'exploitation du contexte environnemental dans le but d'améliorer les estimations résultant du processus de suivi. L'état de l'art fait mention de quelques études adressant cette problématique. Dans ces études, des modèles graphiques aux capacités d'expressivité limitées, tels que des réseaux Bayésiens dynamiques, sont utilisés pour modéliser des connaissances contextuelles a-priori. Dans cette thèse, nous proposons d'utiliser des modèles contextuelles plus riches issus des simulateurs de comportements d'agents autonomes et démontrons l'efficacité de notre approche au travers d'un ensemble d'évaluations expérimentales. La deuxième partie de la thèse adresse le problème général d'influences mutuelles — communément appelées interactions — entre piétons et l'impact de ces interactions sur les comportements respectifs de ces derniers durant le processus de suivi. Sous l'hypothèse que nous disposons d'un simulateur (ou une fonction) modélisant ces interactions, nous développons une approche de suivi comportemental à faible coût computationnel et facilement extensible dans laquelle les interactions entre cibles sont prises en compte. L'originalité de l'approche proposée vient de l'introduction des "représentants", qui sont des informations agrégées issues de la distribution de chaque cible de telle sorte à maintenir une diversité comportementale, et sur lesquels le système de filtrage s'appuie pour estimer, de manière fine, les comportements des différentes cibles et ceci, même en cas d'occlusions. Nous présentons nos choix de modélisation, les algorithmes résultants, et un ensemble de scénarios difficiles sur lesquels l'approche proposée est évaluée.

Mots-clés: Suivi comportemental, Suivi de trajectoire et reconnaissance d'activité en simultané, Agents autonomes, Simulations à base d'agents, Filtre particulaire, Interactions, JPDAF, Cibles en interaction, États représentatifs, Agrégation à base de densité, Clusterisation, Diversité comportementale, Occlusions.